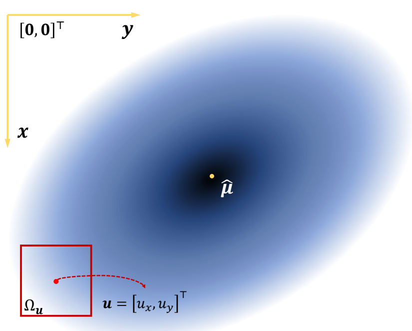
Analytic-Splatting: Anti-Aliased 3D Gaussian Splatting via Analytic Integration - Supplementary Material

Zhihao Liang[1,*], Qi Zhang[2,†], Wenbo Hu[2],
Lei Zhu[3], Ying Feng[2], and Kui Jia[4,†]

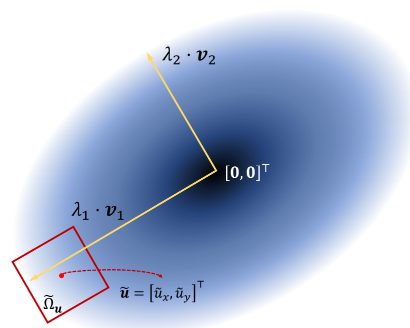
[1]South China University of Technology
[2]Tencent AI Lab [3]City University of Hong Kong
[4]School of Data Science, The Chinese University of Hong Kong, Shenzhen
https://lzhnb.github.io/project-pages/analytic-splatting/

1 Shading Module

Since we greatly improved the shading module, our implementation is quite different from the vanilla 3DGS, especially the backward part. In this section, we give the forward and backward propagation step by step so that everyone can learn the details and reproduce our shading module easily.



(a) Integral over the original pixel area.



(b) Integral over the rotated pixel area.

Fig. 1: Example diagram of the pixel integration domain and the domain after rotation. The yellow lines in Fig. 1a are the coordinate axes of 2D screen space; And the yellow lines in Fig. 1b are the eigenvectors scaled by the eigenvalues.

1.1 Forward

Given a pixel with center $\boldsymbol{u} = [u_x, u_y]^\top$, we use a 2D Gaussian signal g^{2D} to respond to the pixel and shade it. Assuming that the 2D Gaussian signal g^{2D} has a mean vector $\hat{\boldsymbol{\mu}} \in \mathbb{R}^2$ and a real-symmetric covariance matrix $\hat{\Sigma} \in \mathbb{R}^{2\times 2}$.

For better formulation, we express them as:

$$\hat{\boldsymbol{\mu}} = \begin{bmatrix} \hat{\mu}_x \\ \hat{\mu}_y \end{bmatrix}, \quad \hat{\boldsymbol{\Sigma}} = \begin{bmatrix} \hat{\Sigma}_{11} & \hat{\Sigma}_{12} \\ \hat{\Sigma}_{12} & \hat{\Sigma}_{22} \end{bmatrix}. \tag{1}$$

In Analytic-Splatting, we first perform eigendecomposition on $\hat{\boldsymbol{\Sigma}}$ to achieve diagonalization. After the decomposition, we obtain the eigenvalues $\{\lambda_1, \lambda_2\}$ and eigenvectors $\{\boldsymbol{v}_1, \boldsymbol{v}_2\}$ of $\hat{\boldsymbol{\Sigma}}$:

$$\begin{gathered}
\lambda_1 = \frac{\mathrm{Tr}(\hat{\boldsymbol{\Sigma}}) + \sqrt{\mathrm{Tr}(\hat{\boldsymbol{\Sigma}})^2 - 4\det(\hat{\boldsymbol{\Sigma}})}}{2}, \quad \lambda_2 = \frac{\mathrm{Tr}(\hat{\boldsymbol{\Sigma}}) - \sqrt{\mathrm{Tr}(\hat{\boldsymbol{\Sigma}})^2 - 4\det(\hat{\boldsymbol{\Sigma}})}}{2}, \\
\hat{\boldsymbol{v}}_1 = \begin{bmatrix} \hat{\Sigma}_{12} \\ \lambda_1 - \hat{\Sigma}_{11} \end{bmatrix}, \quad \hat{\boldsymbol{v}}_2 = \begin{bmatrix} \lambda_2 - \hat{\Sigma}_{22} \\ \hat{\Sigma}_{12} \end{bmatrix}, \\
\boldsymbol{v}_1 = \begin{bmatrix} v_{1x} \\ v_{1y} \end{bmatrix} = \frac{\hat{\boldsymbol{v}}_1}{\|\hat{\boldsymbol{v}}_1\|} = \begin{bmatrix} \frac{\hat{\Sigma}_{12}}{\|\hat{\boldsymbol{v}}_1\|} \\ \frac{\lambda_1 - \hat{\Sigma}_{11}}{\|\hat{\boldsymbol{v}}_1\|} \end{bmatrix}, \quad \boldsymbol{v}_2 = \begin{bmatrix} v_{2x} \\ v_{2y} \end{bmatrix} = \frac{\hat{\boldsymbol{v}}_2}{\|\hat{\boldsymbol{v}}_2\|} = \begin{bmatrix} \frac{\lambda_2 - \hat{\Sigma}_{22}}{\|\hat{\boldsymbol{v}}_1\|} \\ \frac{\hat{\Sigma}_{12}}{\|\hat{\boldsymbol{v}}_1\|} \end{bmatrix},
\end{gathered} \tag{2}$$

where $\mathrm{Tr}(\hat{\boldsymbol{\Sigma}}) = \hat{\Sigma}_{11} + \hat{\Sigma}_{22}$, $\det(\hat{\boldsymbol{\Sigma}}) = \hat{\Sigma}_{11}\hat{\Sigma}_{22} - \hat{\Sigma}_{12}^2$.

Then we use the eigenvectors $\{\boldsymbol{v}_1, \boldsymbol{v}_2\}$ to construct a new coordinate system with its origin at $\hat{\boldsymbol{\mu}} = [\hat{\boldsymbol{\mu}}_x, \hat{\boldsymbol{\mu}}_y]^\top$ (refer to the yellow lines in Fig. 1b) to unravel the correlation in the covariance $\hat{\boldsymbol{\Sigma}}$. In this way, the coordinate of the pixel center $\boldsymbol{u}$ is transformed into $\tilde{\boldsymbol{u}}$:

$$\begin{aligned}
\tilde{\boldsymbol{u}} = \begin{bmatrix} \tilde{u}_x \\ \tilde{u}_y \end{bmatrix} &= \begin{bmatrix} - \ \boldsymbol{v}_1 \ - \\ - \ \boldsymbol{v}_2 \ - \end{bmatrix} (\boldsymbol{u} - \hat{\boldsymbol{\mu}}) = \begin{bmatrix} v_{1x} & v_{1y} \\ v_{2x} & v_{2y} \end{bmatrix} \begin{bmatrix} u_x - \hat{\mu}_x \\ u_y - \hat{\mu}_y \end{bmatrix} \\
&= \begin{bmatrix} v_{1x}(u_x - \hat{\mu}_x) + v_{1y}(u_y - \hat{\mu}_y) \\ v_{2x}(u_x - \hat{\mu}_x) + v_{2y}(u_y - \hat{\mu}_y) \end{bmatrix}
\end{aligned}, \tag{3}$$

then the intensity response of the pixel center can be written as:

$$g^{2\mathrm{D}}(\boldsymbol{u}) = \exp\left(-\frac{\tilde{u}_x^2}{2\lambda_1}\right) \exp\left(-\frac{\tilde{u}_y^2}{2\lambda_2}\right). \tag{4}$$

In Sec. 4.1 of the main page, we propose to use a conditioned logistic function to approximate the cumulative distribution function (CDF) of the standard Gaussian distribution $g(x)$ as:

$$\begin{aligned}
G(x) &= \int_{-\infty}^{x} g(u)\mathrm{d}u = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right) \mathrm{d}u \\
&\approx S(x) = \frac{1}{1 + \exp(-1.6 \cdot x - 0.07 \cdot x^3)}
\end{aligned}, \tag{5}$$

and for the Gaussian distribution with standard deviation $\sigma \neq 1$, we use the reciprocal of σ to scale x and express the logistic function S_σ as:

$$\begin{aligned} G_\sigma(x) &= \int_{-\infty}^{x} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{u^2}{2\sigma^2}\right) \mathrm{d}u \\ &\approx S_\sigma(x) = S\left(\frac{x}{\sigma}\right) = \frac{1}{1+\exp\left(-1.6\cdot\frac{x}{\sigma} - 0.07\cdot\left(\frac{x}{\sigma}\right)^3\right)}. \end{aligned} \tag{6}$$

Given the CDF approximation Eq. (6), we approximate the response of 1-width window around sample x as:

$$\begin{aligned} \mathcal{I}_g(x) &= \int_{x-\frac{1}{2}}^{x+\frac{1}{2}} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{u^2}{2\sigma^2}\right) \mathrm{d}u \\ &= G_\sigma\left(x+\frac{1}{2}\right) - G_\sigma\left(x-\frac{1}{2}\right) \approx S_\sigma\left(x+\frac{1}{2}\right) - S_\sigma\left(x-\frac{1}{2}\right) \end{aligned}. \tag{7}$$

In Analytic-Splatting, we calculate the intensity response $\mathcal{I}_g^{\mathrm{2D}}$ by approximating the integral of Gaussian signal Eq. (4) over the domain $\tilde{\Omega}_{\boldsymbol{u}}$ in Fig. 1, and the integral over 2D domain $\tilde{\Omega}_{\boldsymbol{u}}$ can be represented as the product of the integrals of two one-dimensional Gaussian signals:

$$\begin{aligned} \mathcal{I}_g^{\mathrm{2D}} &\approx \int_{\tilde{\Omega}_{\boldsymbol{u}}} g^{\mathrm{2D}}(\boldsymbol{u})\mathrm{d}\boldsymbol{u} \\ &= \int_{\tilde{u}_x-\frac{1}{2}}^{\tilde{u}_x+\frac{1}{2}} \exp\left(-\frac{x^2}{2\lambda_1}\right)\mathrm{d}x \int_{\tilde{u}_y-\frac{1}{2}}^{\tilde{u}_y+\frac{1}{2}} \exp\left(-\frac{y^2}{2\lambda_2}\right)\mathrm{d}y \\ &= \sqrt{2\pi\lambda_1}\int_{\tilde{u}_x-\frac{1}{2}}^{\tilde{u}_x+\frac{1}{2}} \frac{\exp\left(-x^2/2\lambda_1\right)}{\sqrt{2\pi\lambda_1}}\mathrm{d}x \cdot \sqrt{2\pi\lambda_2}\int_{\tilde{u}_y-\frac{1}{2}}^{\tilde{u}_y+\frac{1}{2}} \frac{\exp\left(-y^2/2\lambda_2\right)}{\sqrt{2\pi\lambda_2}}\mathrm{d}y \\ &\approx 2\pi\, \underbrace{\sigma_1\left[S_{\sigma_1}(\tilde{u}_x+\frac{1}{2}) - S_{\sigma_1}(\tilde{u}_x-\frac{1}{2})\right]}_{\mathcal{I}_{\sigma_1}} \underbrace{\sigma_2\left[S_{\sigma_2}(\tilde{u}_y+\frac{1}{2}) - S_{\sigma_2}(\tilde{u}_y-\frac{1}{2})\right]}_{\mathcal{I}_{\sigma_2}}, \end{aligned} \tag{8}$$

where $\sigma_1 = \sqrt{\lambda_1}, \sigma_2 = \sqrt{\lambda_2}$ denote the standard derivations of the independent Gaussian signals along two eigenvectors respectively. In summary, the volume shading in Analytic-Splatting is given by:

$$\begin{aligned} \boldsymbol{C}(\boldsymbol{u}) &= \sum_{i\in N} T_i \mathcal{I}_{g-i}^{\mathrm{2D}}(\boldsymbol{u}|\hat{\boldsymbol{\mu}}_i, \hat{\boldsymbol{\Sigma}}_i)\alpha_i \boldsymbol{c}_i, \quad T_i = \prod_{j=1}^{i-1}(1-\mathcal{I}_{g-j}^{\mathrm{2D}}(\boldsymbol{u}|\hat{\boldsymbol{\mu}}_j, \hat{\boldsymbol{\Sigma}}_j)\alpha_j), \\ \mathcal{I}_g^{\mathrm{2D}}(\boldsymbol{u}) &= 2\pi\sigma_1\sigma_2\left[S_{\sigma_1}(\tilde{u}_x+\frac{1}{2}) - S_{\sigma_1}(\tilde{u}_x-\frac{1}{2})\right]\left[S_{\sigma_2}(\tilde{u}_y+\frac{1}{2}) - S_{\sigma_2}(\tilde{u}_y-\frac{1}{2})\right] \\ &= 2\pi\mathcal{I}_{\sigma_1}\mathcal{I}_{\sigma_2}. \end{aligned} \tag{9}$$

1.2 Backward

Derivation of the Conditioned Logistic Function Before introducing the backpropagation in Analytic-Splatting, we first give the derivation of the conditioned logistic function $S(x)$ in Eq. (5):

$$\begin{aligned}\frac{\partial S(x)}{\partial x} &= \left(1.6 + 0.21 \cdot x^2\right) \frac{\exp\left(-1.6 \cdot x - 0.07 \cdot x^3\right)}{\left[1 + \exp\left(-1.6 \cdot x - 0.07 \cdot x^3\right)\right]^2}, \\ &= (1.6 + 0.21 \cdot x^2) S(x) [1 - S(x)]\end{aligned} \tag{10}$$

further for the derivation of $S_\sigma(x)$ in Eq. (6), we can get $\frac{\partial S_\sigma(x)}{\partial x}$ and $\frac{\partial S_\sigma(x)}{\partial \sigma}$ through the chain rule:

$$\begin{aligned}\frac{\partial S_\sigma(x)}{\partial x} &= \frac{\partial S\left(\frac{x}{\sigma}\right)}{\partial \frac{x}{\sigma}} \frac{\partial \frac{x}{\sigma}}{\partial x} = \left(1.6 + 0.21 \cdot \left(\frac{x}{\sigma}\right)^2\right) S_\sigma\left(\frac{x}{\sigma}\right)\left[1 - S_\sigma\left(\frac{x}{\sigma}\right)\right] \cdot \frac{1}{\sigma} \\ \frac{\partial S_\sigma(x)}{\partial \sigma} &= \frac{\partial S\left(\frac{x}{\sigma}\right)}{\partial \frac{x}{\sigma}} \frac{\partial \frac{x}{\sigma}}{\partial \sigma} = \left(1.6 + 0.21 \cdot \left(\frac{x}{\sigma}\right)^2\right) S_\sigma\left(\frac{x}{\sigma}\right)\left[1 - S_\sigma\left(\frac{x}{\sigma}\right)\right] \cdot -\frac{x}{\sigma^2}\end{aligned} \tag{11}$$

Backpropogation in Shading Our backpropagation aims to derive the gradients of $\mathcal{I}_g^{\mathrm{2D}}$ with respect to $\hat{\boldsymbol{\mu}}$ and $\hat{\boldsymbol{\Sigma}}$, as $\frac{\partial \mathcal{I}_g^{\mathrm{2D}}}{\partial \hat{\boldsymbol{\mu}}}$ and $\frac{\partial \mathcal{I}_g^{\mathrm{2D}}}{\partial \hat{\boldsymbol{\Sigma}}}$:

$$\frac{\partial \mathcal{I}_g^{\mathrm{2D}}}{\partial \hat{\boldsymbol{\mu}}} = \begin{bmatrix} \partial \mathcal{I}_g^{\mathrm{2D}} / \partial \hat{\mu}_x \\ \partial \mathcal{I}_g^{\mathrm{2D}} / \partial \hat{\mu}_y \end{bmatrix} \in \mathbb{R}^2, \quad \frac{\partial \mathcal{I}_g^{\mathrm{2D}}}{\partial \hat{\boldsymbol{\Sigma}}} = \begin{bmatrix} \partial \mathcal{I}_g^{\mathrm{2D}} / \partial \hat{\Sigma}_{11} & \partial \mathcal{I}_g^{\mathrm{2D}} / \partial \hat{\Sigma}_{12} \\ \partial \mathcal{I}_g^{\mathrm{2D}} / \partial \hat{\Sigma}_{12} & \partial \mathcal{I}_g^{\mathrm{2D}} / \partial \hat{\Sigma}_{22} \end{bmatrix} \in \mathbb{R}^{2\times 2}. \tag{12}$$

It is quite difficult to directly express the above gradient. Still, we can use the chain rule to boil down the above results layer by layer to be available. We note that our key insight is to construct a new coordinate system using the mean vector $\hat{\boldsymbol{\mu}}$, eigenvalues $\{\lambda_1, \lambda_2\}$ and eigenvectors $\{\boldsymbol{v}_1, \boldsymbol{v}_2\}$. Therefore, we can use them as an intermediary to solve the final gradient by the chain rule. For the gradients of $\mathcal{I}_g^{\mathrm{2D}}$ with respect to the mean vector $\hat{\boldsymbol{\mu}}$, according to Eq. (3) and Eq. (9), we have:

$$\begin{aligned}\frac{\partial \mathcal{I}_g^{\mathrm{2D}}}{\partial \hat{\boldsymbol{\mu}}} =& 2\pi\left(\mathcal{I}_{\sigma_2}\frac{\partial \mathcal{I}_{\sigma_1}}{\partial \hat{\boldsymbol{\mu}}} + \mathcal{I}_{\sigma_1}\frac{\partial \mathcal{I}_{\sigma_2}}{\partial \hat{\boldsymbol{\mu}}}\right) = 2\pi\left(\mathcal{I}_{\sigma_2}\frac{\partial \mathcal{I}_{\sigma_1}}{\partial \tilde{u}_x}\frac{\partial \tilde{u}_x}{\partial \hat{\boldsymbol{\mu}}} + \mathcal{I}_{\sigma_1}\frac{\partial \mathcal{I}_{\sigma_2}}{\partial \tilde{u}_y}\frac{\partial \tilde{u}_y}{\partial \hat{\boldsymbol{\mu}}}\right) \\ =& 2\pi\mathcal{I}_{\sigma_2}\left[\frac{\partial S_{\sigma_1}(\tilde{u}_x + \frac{1}{2})}{\partial \tilde{u}_x}\frac{\partial \tilde{u}_x}{\partial \hat{\boldsymbol{\mu}}} - \frac{\partial S_{\sigma_1}(\tilde{u}_x - \frac{1}{2})}{\partial \tilde{u}_x}\frac{\partial \tilde{u}_x}{\partial \hat{\boldsymbol{\mu}}}\right] + \\ & 2\pi\mathcal{I}_{\sigma_1}\left[\frac{\partial S_{\sigma_2}(\tilde{u}_y + \frac{1}{2})}{\partial \tilde{u}_y}\frac{\partial \tilde{u}_y}{\partial \hat{\boldsymbol{\mu}}} - \frac{\partial S_{\sigma_2}(\tilde{u}_y - \frac{1}{2})}{\partial \tilde{u}_y}\frac{\partial \tilde{u}_y}{\partial \hat{\boldsymbol{\mu}}}\right],\end{aligned} \tag{13}$$

where $\frac{\partial \tilde{u}_x}{\partial \hat{\boldsymbol{\mu}}} = [-v_{1x}, -v_{1y}]^\top$, $\frac{\partial \tilde{u}_y}{\partial \hat{\boldsymbol{\mu}}} = [-v_{2x}, -v_{2y}]^\top$, and the gradient $\frac{\partial S_\sigma(x)}{\partial x}$ has been solved in Eq. (11).

According to Eq. (9), we have the gradients of $\mathcal{I}_g^{2D}$ to eigenvalues $\{\lambda_1, \lambda_2\}$:

$$
\begin{aligned}
\frac{\partial \mathcal{I}_g^{2D}}{\partial \lambda_1} &= \frac{\partial \mathcal{I}_g^{2D}}{\partial \sigma_1}\frac{\partial \sigma_1}{\partial \lambda_1} = 2\pi \mathcal{I}_{\sigma_2}\frac{\partial \mathcal{I}_{\sigma_1}}{\partial \sigma_1}\cdot \frac{1}{2\sqrt{\lambda_1}} \\
&= \frac{\pi \mathcal{I}_{\sigma_2}}{\sqrt{\lambda_1}}\left(\left[S_{\sigma_1}(\tilde{u}_x + \frac{1}{2}) - S_{\sigma_1}(\tilde{u}_x - \frac{1}{2})\right] + \sigma_1\left[\frac{\partial S_{\sigma_1}(\tilde{u}_x + \frac{1}{2})}{\partial \sigma_1} - \frac{\partial S_{\sigma_1}(\tilde{u}_x - \frac{1}{2})}{\partial \sigma_1}\right]\right), \\
\frac{\partial \mathcal{I}_g^{2D}}{\partial \lambda_2} &= \frac{\partial \mathcal{I}_g^{2D}}{\partial \sigma_2}\frac{\partial \sigma_2}{\partial \lambda_2} = 2\pi \mathcal{I}_{\sigma_1}\frac{\partial \mathcal{I}_{\sigma_2}}{\partial \sigma_2}\cdot \frac{1}{2\sqrt{\lambda_2}} \\
&= \frac{\pi \mathcal{I}_{\sigma_1}}{\sqrt{\lambda_2}}\left(\left[S_{\sigma_2}(\tilde{u}_y + \frac{1}{2}) - S_{\sigma_2}(\tilde{u}_y - \frac{1}{2})\right] + \sigma_2\left[\frac{\partial S_{\sigma_2}(\tilde{u}_y + \frac{1}{2})}{\partial \sigma_2} - \frac{\partial S_{\sigma_2}(\tilde{u}_y - \frac{1}{2})}{\partial \sigma_2}\right]\right),
\end{aligned}
\tag{14}
$$

please note that the gradient $\frac{\partial S_\sigma(x)}{\partial \sigma}$ has been solved in Eq. (11). Given Eq. (3) and Eq. (9), the gradients of $\mathcal{I}_g^{2D}$ with respect to the eigenvectors $\{\boldsymbol{v}_1, \boldsymbol{v}_2\}$ are:

$$
\begin{aligned}
\frac{\partial \mathcal{I}_g^{2D}}{\partial \boldsymbol{v}_1} &= \frac{\partial \mathcal{I}_g^{2D}}{\partial \tilde{u}_x}\frac{\partial \tilde{u}_x}{\partial \boldsymbol{v}_1} = 2\pi \mathcal{I}_{\sigma 2}\frac{\partial \mathcal{I}_{\sigma_1}}{\partial \tilde{u}_x}\begin{bmatrix} u_x - \hat{\mu}_x \\ u_y - \hat{\mu}_y \end{bmatrix} \\
&= 2\pi \mathcal{I}_{\sigma 2}\left(\frac{\partial S_{\sigma_1}(\tilde{u}_x + \frac{1}{2})}{\partial \tilde{u}_x} - \frac{\partial S_{\sigma_1}(\tilde{u}_x - \frac{1}{2})}{\partial \tilde{u}_x}\right)\begin{bmatrix} u_x - \hat{\mu}_x \\ u_y - \hat{\mu}_y \end{bmatrix} \in \mathbb{R}^2, \\
\frac{\partial \mathcal{I}_g^{2D}}{\partial \boldsymbol{v}_2} &= \frac{\partial \mathcal{I}_g^{2D}}{\partial \tilde{u}_y}\frac{\partial \tilde{u}_y}{\partial \boldsymbol{v}_2} = 2\pi \mathcal{I}_{\sigma 1}\frac{\partial \mathcal{I}_{\sigma_2}}{\partial \tilde{u}_y}\begin{bmatrix} u_x - \hat{\mu}_x \\ u_y - \hat{\mu}_y \end{bmatrix} \\
&= 2\pi \mathcal{I}_{\sigma 1}\left(\frac{\partial S_{\sigma_2}(\tilde{u}_y + \frac{1}{2})}{\partial \tilde{u}_y} - \frac{\partial S_{\sigma_2}(\tilde{u}_y - \frac{1}{2})}{\partial \tilde{u}_y}\right)\begin{bmatrix} u_x - \hat{\mu}_x \\ u_y - \hat{\mu}_y \end{bmatrix} \in \mathbb{R}^2.
\end{aligned}
\tag{15}
$$

In addition, please refer to [12] and learn the gradients of eigenvalues to the covariance $\{\frac{\partial \lambda_1}{\partial \hat{\Sigma}_{ij}}, \frac{\partial \lambda_2}{\partial \hat{\Sigma}_{ij}}\}$ and the gradients of eigenvectors with respect to the covariance $\{\frac{\partial \boldsymbol{v}_1}{\partial \hat{\Sigma}_{ij}}, \frac{\partial \boldsymbol{v}_2}{\partial \hat{\Sigma}_{ij}}\}$. According to Eq. (14) and Eq. (15), we get the final gradient:

$$
\begin{aligned}
\frac{\partial \mathcal{I}_g^{2D}}{\partial \hat{\Sigma}_{11}} &= \frac{\partial \mathcal{I}_g^{2D}}{\partial \lambda_1}\frac{\partial \lambda_1}{\partial \hat{\Sigma}_{11}} + \frac{\partial \mathcal{I}_g^{2D}}{\partial \lambda_2}\frac{\partial \lambda_2}{\partial \hat{\Sigma}_{11}} + \left(\frac{\partial \mathcal{I}_g^{2D}}{\partial \boldsymbol{v}_1}\right)^\top \frac{\partial \boldsymbol{v}_1}{\partial \hat{\Sigma}_{11}} + \left(\frac{\partial \mathcal{I}_g^{2D}}{\partial \boldsymbol{v}_2}\right)^\top \frac{\partial \boldsymbol{v}_2}{\partial \hat{\Sigma}_{11}} \\
\frac{\partial \mathcal{I}_g^{2D}}{\partial \hat{\Sigma}_{12}} &= \frac{\partial \mathcal{I}_g^{2D}}{\partial \lambda_1}\frac{\partial \lambda_1}{\partial \hat{\Sigma}_{12}} + \frac{\partial \mathcal{I}_g^{2D}}{\partial \lambda_2}\frac{\partial \lambda_2}{\partial \hat{\Sigma}_{12}} + \left(\frac{\partial \mathcal{I}_g^{2D}}{\partial \boldsymbol{v}_1}\right)^\top \frac{\partial \boldsymbol{v}_1}{\partial \hat{\Sigma}_{12}} + \left(\frac{\partial \mathcal{I}_g^{2D}}{\partial \boldsymbol{v}_2}\right)^\top \frac{\partial \boldsymbol{v}_2}{\partial \hat{\Sigma}_{12}}. \\
\frac{\partial \mathcal{I}_g^{2D}}{\partial \hat{\Sigma}_{22}} &= \frac{\partial \mathcal{I}_g^{2D}}{\partial \lambda_1}\frac{\partial \lambda_1}{\partial \hat{\Sigma}_{22}} + \frac{\partial \mathcal{I}_g^{2D}}{\partial \lambda_2}\frac{\partial \lambda_2}{\partial \hat{\Sigma}_{22}} + \left(\frac{\partial \mathcal{I}_g^{2D}}{\partial \boldsymbol{v}_1}\right)^\top \frac{\partial \boldsymbol{v}_1}{\partial \hat{\Sigma}_{22}} + \left(\frac{\partial \mathcal{I}_g^{2D}}{\partial \boldsymbol{v}_2}\right)^\top \frac{\partial \boldsymbol{v}_2}{\partial \hat{\Sigma}_{22}}
\end{aligned}
\tag{16}
$$

2 Additional Results

In this section, we report more quantitative and qualitative results in detail. The 3D filtering mechanism proposed in Mip-Splatting [13] can be adopted into our method and improve the novel-view synthesis results on the scene datasets, thus we report our results using the 3D filtering mechanism here.

	PSNR ↑				SSIM ↑				LPIPS ↓			
	Truck	Train	Johnson	Playroom	Truck	Train	Johnson	Playroom	Truck	Train	Johnson	Playroom
Plenoxels [5]	23.22	18.93	23.14	22.98	0.774	0.663	0.787	0.802	0.335	0.422	0.521	0.499
INGP-Base [11]	23.26	20.17	27.75	19.48	0.779	0.666	0.839	0.754	0.274	0.386	0.381	0.465
INGP-Big [11]	23.83	20.46	28.26	21.67	0.800	0.689	0.854	0.779	0.249	0.360	0.352	0.428
Mip-NeRF 360 [2]	24.91	19.52	29.14	29.66	0.857	0.660	0.901	0.900	0.159	0.354	0.237	0.252
3DGS [8]	25.19	21.09	28.77	30.04	0.879	0.802	0.899	0.906	0.148	0.218	0.244	0.241
Mip-Splatting [13]	25.48	22.09	29.09	30.28	0.884	0.818	0.901	0.908	0.149	0.208	0.248	0.248
Ours	25.48	22.20	29.18	30.32	0.883	0.819	0.901	0.909	0.148	0.207	0.247	0.248
Ours + 3D filter	25.60	22.22	29.17	30.25	0.887	0.820	0.903	0.909	0.147	0.206	0.247	0.248

Table 1: Quantitative comparison of our method against previous methods on Tanks&Temples [9] and Deep Blending [6]. All methods are trained on full-resolution images and tested on the same-resolution images.

2.1 Single-scale Training and Single-scale Testing on Scene Datasets.

We evaluate our Analytic-Splatting against other methods on complex scene datasets (*i.e.* Mip-NeRF 360 [2], Tanks & Temples [9], and Deep Blending [6]) under the single-scale training and testing setting, which is the most widely used setting. The overall results are shown in the Tab. 3 on the main page, our method shows great generalization across different datasets and almost outperforms other methods. Here, we report per-scene metrics for Tanks & Temples [9] and Deep Blending [6] in Tab. 1, and for Mip-NeRF 360 [2] in Tab. 2.

For Mip-NeRF 360, images from indoor and outdoor scenes are downsampled by 2× and 4×, respectively, as full-resolution input for training and testing. The official dataset provides the downsampled images and stores them in different folders. Specifically, the results reported in Mip-Splatting did not use officially provided downsampled images as input, but use the bicubically downsampled images as input. The quantitative results in Tab. 2 show these two downsampling schemes greatly affect the metrics. Therefore, for fairness, we mark the methods that use bicubically downsampled images for training with * (*i.e.* 3DGS*, Mip-Splatting*, and Ours*). And the remaining methods without * marks use officially provided downsampled images for training.

2.2 Multi-scale Training and Multi-scale Testing on the Multi-scale Blender Synthetic Dataset

We evaluate our Analytic-Splatting against other cutting-edge methods on the Blender Synthetic dataset under the multi-scale training and testing setting. Since per-resolution metrics have been mentioned in the main paper, we report per-object metrics in Tab. 4 for more comprehensive comparisons. More qualitative results are shown in Fig. 2. Our method almost surpasses other methods and performs better than other methods in terms of detail capturing and anti-aliasing. We further provide our method's per-resolution and per-object metrics in Tab. 3 so that subsequent methods can better refer to our results.

	PSNR ↑									
	bicycle	flowers	garden	stump	treehill	room	counter	kitchen	bonsai	Avg.
Plenoxels [5]	21.91	20.10	23.49	20.66	22.25	27.59	23.62	23.42	24.67	23.08
INGP-Base [11]	22.19	20.35	24.60	23.63	22.36	29.27	26.44	28.55	30.34	25.30
INGP-Big [11]	22.17	20.65	25.07	23.47	22.37	29.69	26.69	29.48	30.69	25.59
Mip-NeRF 360 [2]	24.37	21.73	26.98	26.40	22.87	31.63	29.55	32.23	33.46	27.69
3DGS [8]	25.25	21.52	27.41	26.55	22.49	30.63	28.70	30.32	31.98	27.21
Mip-Splatting [13]	25.31	21.62	27.45	26.62	22.62	31.62	29.11	31.53	32.30	27.57
Ours	25.18	21.61	27.39	26.65	22.54	31.75	29.11	31.56	32.43	27.58
Ours + 3D filter	25.32	21.64	27.51	26.68	22.59	31.66	29.04	31.50	32.30	27.58
3DGS* [8]	25.63	21.77	27.70	26.87	22.75	31.69	29.08	31.56	32.29	27.76
Mip-Splatting* [13]	25.72	21.93	27.76	26.94	22.98	31.74	29.16	31.55	32.31	27.79
Ours*	25.63	21.92	27.73	26.92	22.79	31.89	29.24	31.66	32.60	27.82
Ours + 3D filter*	25.70	21.92	27.78	26.98	22.95	31.86	29.17	31.74	32.55	27.85

	SSIM ↑									
	bicycle	flowers	garden	stump	treehill	room	counter	kitchen	bonsai	Avg.
Plenoxels [5]	0.496	0.431	0.606	0.523	0.509	0.842	0.759	0.648	0.814	0.625
INGP-Base [11]	0.491	0.450	0.649	0.574	0.518	0.855	0.798	0.818	0.890	0.671
INGP-Big [11]	0.512	0.486	0.701	0.594	0.542	0.871	0.817	0.858	0.906	0.699
Mip-NeRF 360 [2]	0.685	0.583	0.813	0.744	0.632	0.913	0.894	0.920	0.941	0.792
3DGS [8]	0.771	0.605	0.868	0.775	0.638	0.914	0.905	0.922	0.938	0.815
Mip-Splatting [13]	0.767	0.608	0.868	0.776	0.636	0.920	0.909	0.928	0.943	0.817
Ours	0.763	0.606	0.866	0.772	0.633	0.921	0.910	0.928	0.943	0.816
Ours + 3D filter	0.769	0.608	0.869	0.773	0.636	0.921	0.909	0.928	0.943	0.817
3DGS* [8]	0.777	0.620	0.871	0.784	0.655	0.927	0.916	0.933	0.948	0.825
Mip-Splatting* [13]	0.780	0.623	0.875	0.786	0.655	0.928	0.916	0.933	0.948	0.827
Ours*	0.777	0.623	0.873	0.783	0.651	0.929	0.917	0.933	0.943	0.826
Ours + 3D filter*	0.780	0.624	0.876	0.785	0.654	0.929	0.917	0.934	0.949	0.827

	LPIPS ↓									
	bicycle	flowers	garden	stump	treehill	room	counter	kitchen	bonsai	Avg.
Plenoxels [5]	0.506	0.521	0.386	0.503	0.540	0.419	0.441	0.447	0.398	0.463
INGP-Base [11]	0.487	0.481	0.312	0.450	0.489	0.301	0.342	0.254	0.227	0.371
INGP-Big [11]	0.446	0.441	0.257	0.421	0.450	0.261	0.306	0.195	0.205	0.331
Mip-NeRF 360 [2]	0.301	0.344	0.170	0.261	0.339	0.221	0.204	0.127	0.176	0.237
3DGS [8]	0.205	0.336	0.103	0.210	0.317	0.220	0.204	0.129	0.205	0.214
Mip-Splatting [13]	0.213	0.340	0.108	0.216	0.329	0.221	0.201	0.127	0.208	0.218
Ours	0.212	0.336	0.110	0.218	0.328	0.221	0.200	0.127	0.206	0.217
Ours + 3D filter	0.211	0.340	0.108	0.218	0.327	0.220	0.202	0.127	0.207	0.218
3DGS* [8]	0.205	0.329	0.103	0.208	0.318	0.192	0.178	0.113	0.174	0.202
Mip-Splatting* [13]	0.206	0.331	0.103	0.209	0.320	0.192	0.179	0.113	0.173	0.203
Ours*	0.207	0.329	0.105	0.210	0.320	0.194	0.180	0.114	0.176	0.204
Ours + 3D filter*	0.206	0.333	0.104	0.210	0.321	0.194	0.181	0.114	0.177	0.204

Table 2: Quantative results of Single-Scale Training and Single-Scale Testing on the Mip-NeRF 360 [2] dataset. All methods are trained on full-resolution images and tested on the same-resolution images.

2.3 Multi-scale Training and Multi-scale Testing on the Mip-NeRF 360 Dataset

We evaluate our Analytic-Splatting against other methods on the Mip-NeRF 360 dataset under the multi-scale training and testing setting. As mentioned in Sec. 2.2, we use the officially provided downsampled images (2× for indoor scenes, and 4× for outdoor scenes) as full-resolution images. Under the multi-scale training and testing setting, we convert each full-resolution image into a set of four images for training via bicubically downsampling it by $[1\times, 2\times, 4\times, 8\times]$. Some qualitative results are shown in Fig. 3. Our method performs better anti-aliasing capability and detail fidelity.

We further provide per-resolution and per-scene metrics in Tab. 5. The results of Mip-NeRF 360 [2] and Zip-NeRF [3] are reported from the official Zip-NeRF paper [3]. Please note that Mip-NeRF 360 and Zip-NeRF struggle with real-time rendering, and our Analytic-Splatting, like 3DGS and its variants, is capable of real-time rendering.

Super-Resolution Evaluation on Mip-NeRF 360 Dataset. We further evaluate our method against other 3DGS and its variants under the super-resolution setting (2× Res.) on the Mip-NeRF 360 dataset. All methods are trained on the Multi-Scale Mip-NeRF 360 dataset. The quantitative results are

	PSNR ↑								
	chair	drums	ficus	hotdog	lego	materials	mic	ship	Avg.
Full Res.	35.76	26.16	35.76	37.54	35.06	29.59	35.18	30.76	33.22
1/2 Res.	38.48	27.30	36.46	39.46	36.46	31.05	37.61	32.54	34.92
1/4 Res.	39.58	28.61	36.21	40.64	36.53	32.77	39.33	34.12	35.97
1/8 Res.	39.24	29.87	36.01	40.24	34.93	33.54	39.02	35.10	35.99
All.	38.26	27.98	36.11	39.47	35.75	31.74	37.78	33.13	35.03

	SSIM ↑								
	chair	drums	ficus	hotdog	lego	materials	mic	ship	Avg.
Full Res.	0.986	0.952	0.988	0.984	0.980	0.958	0.990	0.901	0.967
1/2 Res.	0.993	0.960	0.993	0.990	0.989	0.974	0.994	0.926	0.977
1/4 Res.	0.995	0.952	0.988	0.984	0.992	0.986	0.995	0.948	0.984
1/8 Res.	0.995	0.977	0.994	0.995	0.992	0.992	0.997	0.967	0.989
All.	0.992	0.964	0.992	0.991	0.988	0.977	0.994	0.936	0.979

	LPIPS ↓								
	chair	drums	ficus	hotdog	lego	materials	mic	ship	Avg.
Full Res.	0.015	0.040	0.011	0.023	0.020	0.039	0.007	0.111	0.033
1/2 Res.	0.007	0.029	0.006	0.011	0.009	0.018	0.004	0.065	0.019
1/4 Res.	0.006	0.026	0.006	0.006	0.007	0.010	0.004	0.036	0.013
1/8 Res.	0.005	0.022	0.006	0.005	0.008	0.008	0.004	0.021	0.010
All.	0.008	0.029	0.007	0.011	0.011	0.018	0.005	0.058	0.018

Table 3: Quantatitive Comparison of Analytic-Splatting against 3DGS and its variants under the 2× super-resolution setting on the Mip-NeRF 360 dataset [2]. These methods are trained on images with down-sampling rates covering [1, 2, 4, 8].

Fig. 2: Qualitative comparison of full-resolution and low-resolution (1/8) on Multi-Scale Blender [1]. All methods are trained on images only from the training set with downsampling rates covering [1, 2, 4, 8]. Our method can better overcome the artifacts in 3DGS with better fidelity of details.

shown in Tab. 6 and the qualitative results are shown in Fig. 4. Both results demonstrate the superior performance of our method, further supporting the capability of Analytic-Splatting in capturing details. Conversely, Mip-Splatting is insufficient in capturing details due to pre-filtering.

2.4 Approximation Error Analysis

In Sec. 5.1 of the main page, we study the approximation error produced by different schemes. In this analysis, we concentrate on the standard deviation $\sigma \in [0.3, 6.6]$ and the samples within the 99% confidence interval (*i.e.* $\|x\| < 3\sigma$).

We plot the curve of the approximation error under different conditions.

In detail, we sample uniformly in the logarithmic coordinate system of $[0.3, 6.6]$ to provide the standard deviation samples and plot the approximation error to the standard deviation as Fig. 5(a) on the main page. For calculating the approximation error in the 1D case, we execute erf-function `scipy.special.erf` to produce the integral reference, and the floating point precision is set to 64 bits (`numpy.float64`). For the approximation error caused by the rotation of the integral domain, to calculate the integral of the original integral domain (Fig. 1a) as reference results, we perform Monte Carlo sampling in the original integral

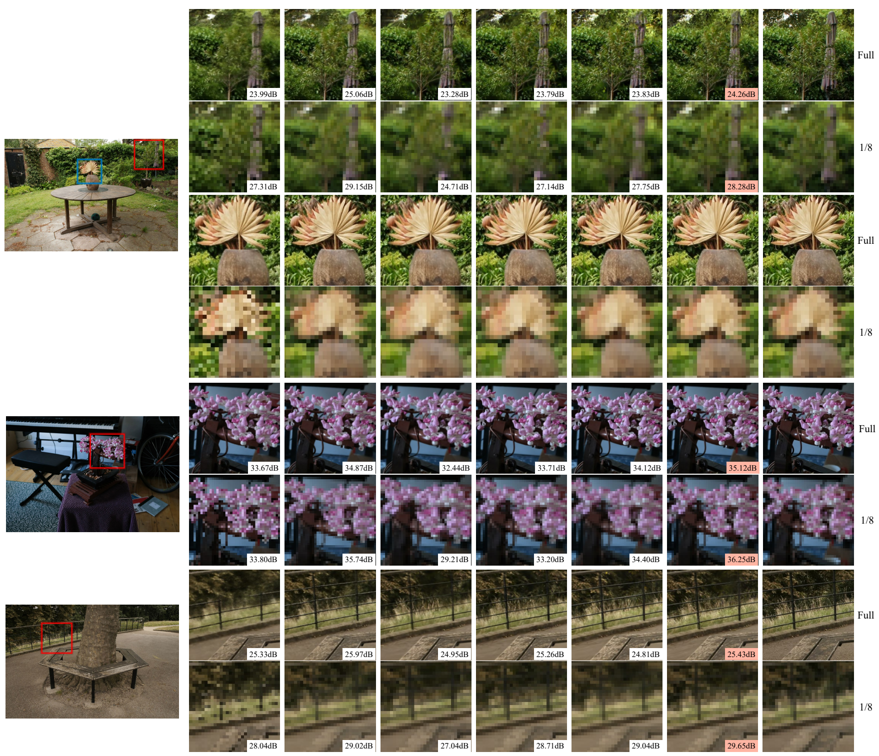


Fig. 3: Qualitative comparison of full-resolution and low-resolution (1/8) on Multi-Scale Mip-NeRF 360 [2]. All methods are trained on images with downsampling rates covering [1, 2, 4, 8]. Our method can better overcome the artifacts in 3DGS with better fidelity of details.

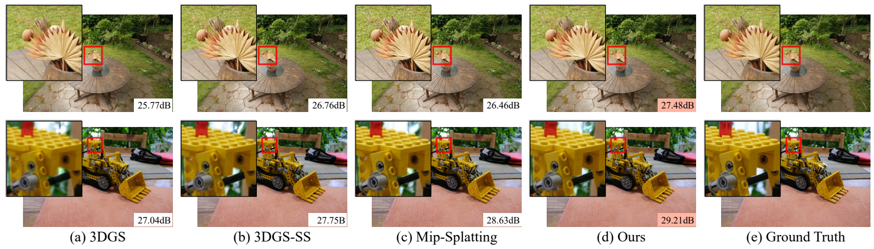


Fig. 4: Qualitative comparison of super-resolution (2×) on Multi-Scale Mip-NeRF 360. [2,3] All methods are trained on images with downsampling rates covering [1, 2, 4, 8]. Please note the high-frequency aliasing of 3DGS-SS and the over-smoothing of Mip-Splatting. Our results are closest to the ground truth.

domain and calculate the average response of 65536 samples as the integration reference.

2.5 Visualization

We follow Mip-Splatting [13] and provide a real-time viewer implemented by WebGL. Specifically, we slightly modified its vertex and fragment shader code to implement our version. We equip a desktop platform with an i7 13700k CPU, an RTX 2080 GPU, and a 1080p 144Hz monitor. We test the visualization on the latest Chrome browser. For scenes, the average inference FPS of Mip-Splatting is around 135, ours is around 132. The biggest gap is about 11 FPS (nearly 10% loss). For objects, both can stably run at 144 FPS.

References

1. 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. pp. 5855–5864 (2021)
2. 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. pp. 5470–5479 (2022)
3. 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 (2023)
4. Chen, A., Xu, Z., Geiger, A., Yu, J., Su, H.: Tensorf: Tensorial radiance fields. In: European Conference on Computer Vision. pp. 333–350. Springer (2022)
5. 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. pp. 5501–5510 (2022)
6. Hedman, P., Philip, J., Price, T., Frahm, J.M., Drettakis, G., Brostow, G.: Deep blending for free-viewpoint image-based rendering. ACM Transactions on Graphics (ToG) **37**(6), 1–15 (2018)

7. Hu, W., Wang, Y., Ma, L., Yang, B., Gao, L., Liu, X., Ma, Y.: Tri-miprf: Tri-mip representation for efficient anti-aliasing neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 19774–19783 (2023)
8. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**(4) (2023)
9. Knapitsch, A., Park, J., Zhou, Q.Y., Koltun, V.: Tanks and temples: Benchmarking large-scale scene reconstruction. ACM Transactions on Graphics (ToG) **36**(4), 1–13 (2017)
10. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: European Conference on Computer Vision. pp. 405–421 (2020)
11. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Transactions on Graphics (ToG) **41**(4), 1–15 (2022)
12. Petersen, K.B., Pedersen, M.S., et al.: The matrix cookbook. Technical University of Denmark **7**(15), 510 (2008)
13. Yu, Z., Chen, A., Huang, B., Sattler, T., Geiger, A.: Mip-splatting: Alias-free 3d gaussian splatting. arXiv preprint arXiv:2311.16493 (2023)

	PSNR ↑								
	chair	drums	ficus	hotdog	lego	materials	mic	ship	Avg.
NeRF w/o $\mathcal{L}_{\text{area}}$	29.92	23.27	27.15	32.00	27.75	26.30	28.40	26.46	27.66
NeRF [10]	33.39	25.87	30.37	35.64	31.65	30.18	32.60	30.09	31.23
MipNeRF [1]	37.14	27.02	33.19	39.31	35.74	32.56	38.04	33.08	34.51
Plenoxels [5]	32.79	25.25	30.28	34.65	31.26	28.33	31.53	28.59	30.34
TensoRF [4]	32.47	25.37	31.16	34.96	31.73	28.53	31.48	29.08	30.60
Instant-NGP [11]	32.95	26.43	30.41	35.87	31.83	29.31	32.58	30.23	31.20
Tri-MipRF [7]	37.67	27.35	33.57	38.78	35.72	31.42	37.63	32.74	34.36
3DGS [8]	32.73	25.30	29.00	35.03	29.44	27.13	31.17	28.33	29.77
3DGS-SS [8]	35.62	27.02	33.12	37.46	33.27	29.90	34.69	30.63	32.71
Mip-Splatting [13]	37.48	27.74	34.71	39.15	35.07	31.88	37.68	32.80	34.56
Ours	38.26	27.98	36.11	39.47	35.75	31.74	37.78	33.13	35.03
Ours + 3D filter	37.53	27.77	35.85	39.17	35.26	31.80	37.61	32.95	34.74

	SSIM ↑								
	chair	drums	ficus	hotdog	lego	materials	mic	ship	Avg.
NeRF w/o $\mathcal{L}_{\text{area}}$	0.944	0.891	0.942	0.959	0.926	0.934	0.958	0.861	0.927
NeRF [10]	0.971	0.932	0.971	0.979	0.965	0.967	0.980	0.900	0.958
MipNeRF [1]	0.988	0.945	0.984	0.988	0.984	0.977	0.993	0.922	0.973
Plenoxels [5]	0.968	0.929	0.972	0.976	0.964	0.959	0.979	0.892	0.955
TensoRF [4]	0.967	0.930	0.972	0.976	0.964	0.959	0.979	0.892	0.955
Instant-NGP [11]	0.971	0.940	0.973	0.979	0.966	0.959	0.981	0.904	0.959
Tri-MipRF [7]	0.990	0.951	0.985	0.988	0.986	0.969	0.992	0.929	0.974
3DGS [8]	0.976	0.941	0.968	0.982	0.964	0.956	0.979	0.910	0.960
3DGS-SS [8]	0.988	0.958	0.985	0.988	0.982	0.973	0.990	0.928	0.974
Mip-Splatting [13]	0.991	0.963	0.990	0.990	0.987	0.978	0.994	0.936	0.979
Ours	0.992	0.964	0.992	0.991	0.988	0.977	0.994	0.936	0.979
Ours + 3D filter	0.991	0.963	0.990	0.990	0.987	0.977	0.994	0.936	0.979

	LPIPS ↓								
	chair	drums	ficus	hotdog	lego	materials	mic	ship	Avg.
NeRF w/o $\mathcal{L}_{\text{area}}$	0.035	0.069	0.032	0.028	0.041	0.045	0.031	0.095	0.052
NeRF [10]	0.028	0.059	0.026	0.024	0.035	0.033	0.025	0.085	0.044
MipNeRF [1]	0.011	0.044	0.014	0.012	0.013	0.019	0.007	0.062	0.026
Plenoxels [5]	0.040	0.070	0.032	0.037	0.038	0.055	0.036	0.104	0.051
TensoRF [4]	0.042	0.070	0.032	0.037	0.038	0.055	0.036	0.104	0.051
Instant-NGP [11]	0.035	0.066	0.029	0.028	0.040	0.051	0.032	0.095	0.047
Tri-MipRF [7]	0.011	0.046	0.016	0.014	0.013	0.033	0.008	0.069	0.026
3DGS [8]	0.025	0.056	0.030	0.022	0.038	0.040	0.023	0.086	0.040
3DGS-SS [8]	0.013	0.036	0.014	0.014	0.017	0.023	0.008	0.068	0.024
Mip-Splatting [13]	0.010	0.031	0.009	0.011	0.012	0.018	0.005	0.059	0.019
Ours	0.008	0.029	0.007	0.011	0.011	0.018	0.005	0.058	0.018
Ours + 3D filter	0.009	0.031	0.009	0.011	0.012	0.019	0.005	0.059	0.019

Table 4: Quantitative Comparison of Analytic-Splatting against several cutting-edge methods on the Multi-scale Blender Synthetic dataset [1]. We report the metrics for each object in this table and all methods are trained on images only from the training set with downsampling rates covering [1, 2, 4, 8].

Outdoors

PSNR↑	bicycle				flowers				garden				stump				treehill			
	1×	2×	4×	8×	1×	2×	4×	8×	1×	2×	4×	8×	1×	2×	4×	8×	1×	2×	4×	8×
Mip-NeRF 360 [2]	24.51	26.93	28.53	29.24	21.64	23.90	26.01	27.35	26.71	29.59	31.35	32.52	26.27	27.68	28.82	29.27	22.93	24.63	26.06	27.12
Zip-NeRF [3]	25.57	28.25	30.20	31.37	22.37	24.91	27.51	29.50	27.71	30.53	32.60	33.83	27.17	28.62	30.30	31.73	23.63	25.47	27.27	28.84
3DGS [8]	24.19	26.23	26.46	25.83	20.96	23.07	24.54	23.93	26.16	28.54	29.13	28.66	25.84	27.24	27.96	27.64	22.50	24.13	25.31	25.35
3DGS-SS	20.96	27.16	28.25	27.95	21.51	23.07	25.89	26.11	26.81	28.54	30.71	30.91	26.56	27.24	29.36	29.69	22.67	24.13	25.81	26.62
Mip-Splatting [13]	24.90	27.24	28.81	29.10	21.42	23.75	26.13	28.19	26.69	29.37	30.92	31.66	26.49	27.94	29.58	31.17	22.52	24.36	26.22	27.84
Ours	25.20	27.39	28.97	29.81	21.76	24.05	26.52	28.57	27.13	29.53	31.20	32.19	26.74	28.15	29.90	31.50	22.70	24.41	26.13	27.67
Ours + 3D filter	25.32	27.50	29.04	29.88	21.79	24.04	26.49	28.52	27.19	29.52	31.14	32.08	26.80	28.15	29.87	31.48	22.72	24.33	25.97	27.49

SSIM↑	bicycle				flowers				garden				stump				treehill			
	1×	2×	4×	8×	1×	2×	4×	8×	1×	2×	4×	8×	1×	2×	4×	8×	1×	2×	4×	8×
Mip-NeRF 360 [2]	0.666	0.815	0.890	0.912	0.567	0.727	0.834	0.881	0.791	0.903	0.939	0.959	0.726	0.819	0.874	0.882	0.615	0.748	0.839	0.893
Zip-NeRF [3]	0.758	0.872	0.926	0.948	0.635	0.774	0.864	0.914	0.850	0.929	0.960	0.974	0.791	0.865	0.914	0.939	0.671	0.780	0.865	0.922
3DGS [8]	0.703	0.831	0.864	0.855	0.545	0.690	0.784	0.795	0.810	0.904	0.920	0.919	0.729	0.810	0.850	0.830	0.602	0.725	0.811	0.839
3DGS-SS	0.736	0.849	0.902	0.911	0.585	0.724	0.824	0.864	0.834	0.920	0.947	0.956	0.763	0.837	0.887	0.891	0.620	0.738	0.830	0.878
Mip-Splatting [13]	0.739	0.849	0.912	0.940	0.591	0.724	0.825	0.891	0.832	0.917	0.949	0.966	0.768	0.837	0.892	0.932	0.619	0.737	0.839	0.905
Ours	0.750	0.855	0.913	0.940	0.601	0.732	0.834	0.898	0.847	0.921	0.951	0.966	0.772	0.842	0.899	0.933	0.627	0.739	0.835	0.899
Ours + 3D filter	0.754	0.858	0.915	0.941	0.602	0.733	0.835	0.897	0.848	0.922	0.951	0.965	0.773	0.842	0.898	0.933	0.628	0.739	0.834	0.898

LPIPS↓	bicycle				flowers				garden				stump				treehill			
	1×	2×	4×	8×	1×	2×	4×	8×	1×	2×	4×	8×	1×	2×	4×	8×	1×	2×	4×	8×
Mip-NeRF 360 [2]	0.322	0.177	0.089	0.066	0.367	0.215	0.114	0.071	0.194	0.079	0.045	0.029	0.279	0.171	0.114	0.107	0.362	0.236	0.144	0.096
Zip-NeRF [3]	0.222	0.112	0.061	0.041	0.287	0.156	0.083	0.050	0.129	0.055	0.030	0.020	0.206	0.122	0.077	0.057	0.263	0.163	0.103	0.068
3DGS [8]	0.295	0.180	0.113	0.100	0.404	0.288	0.184	0.147	0.197	0.085	0.059	0.054	0.284	0.186	0.130	0.134	0.398	0.279	0.185	0.140
3DGS-SS	0.257	0.144	0.078	0.067	0.365	0.251	0.154	0.107	0.165	0.065	0.038	0.032	0.243	0.150	0.097	0.088	0.364	0.250	0.160	0.110
Mip-Splatting [13]	0.258	0.153	0.083	0.050	0.363	0.262	0.165	0.093	0.169	0.073	0.045	0.027	0.233	0.156	0.106	0.072	0.373	0.265	0.172	0.105
Ours	0.239	0.134	0.070	0.047	0.344	0.239	0.148	0.084	0.141	0.061	0.037	0.027	0.224	0.138	0.087	0.062	0.349	0.244	0.160	0.098
Ours + 3D filter	0.237	0.134	0.070	0.048	0.347	0.241	0.147	0.086	0.141	0.061	0.037	0.027	0.224	0.140	0.089	0.063	0.350	0.245	0.161	0.100

Indoors

PSNR↑	room				counter				kitchen				bonsai			
	1×	2×	4×	8×	1×	2×	4×	8×	1×	2×	4×	8×	1×	2×	4×	8×
Mip-NeRF 360 [2]	31.44	32.53	33.17	32.96	29.30	30.12	30.81	30.52	31.90	33.39	34.69	34.92	32.85	33.97	34.63	33.80
Zip-NeRF [3]	32.20	33.33	34.12	34.26	29.17	29.93	30.70	31.11	32.33	33.76	35.20	35.71	34.08	35.25	36.18	36.32
3DGS [8]	30.53	31.42	31.46	29.81	28.25	28.91	29.21	27.66	29.90	31.04	31.50	29.57	30.63	31.42	31.02	31.58
3DGS-SS	31.12	32.13	32.75	32.12	28.81	29.47	30.16	29.85	30.84	32.05	33.13	32.59	31.57	32.44	32.96	31.58
Mip-Splatting [13]	31.32	32.26	32.79	32.88	28.91	29.50	30.03	30.32	31.11	32.05	32.33	32.79	31.48	32.21	32.26	31.97
Ours	31.26	32.23	32.92	33.07	29.03	29.65	30.39	30.90	31.44	32.57	33.56	33.78	32.12	32.92	33.60	33.39
Ours + 3D filter	31.32	32.28	32.98	33.13	29.03	29.64	30.38	30.88	31.07	32.15	33.09	33.25	31.97	32.73	33.42	33.26

SSIM↑	room				counter				kitchen				bonsai			
	1×	2×	4×	8×	1×	2×	4×	8×	1×	2×	4×	8×	1×	2×	4×	8×
Mip-NeRF 360 [2]	0.906	0.944	0.963	0.967	0.887	0.916	0.936	0.942	0.916	0.949	0.968	0.975	0.935	0.959	0.969	0.968
Zip-NeRF [3]	0.921	0.955	0.971	0.977	0.899	0.926	0.944	0.955	0.926	0.956	0.975	0.982	0.947	0.968	0.978	0.980
3DGS [8]	0.903	0.936	0.952	0.947	0.886	0.912	0.928	0.920	0.907	0.941	0.956	0.950	0.924	0.948	0.954	0.938
3DGS-SS	0.911	0.943	0.962	0.965	0.898	0.922	0.941	0.947	0.919	0.949	0.968	0.938	0.933	0.955	0.967	0.966
Mip-Splatting [13]	0.913	0.944	0.962	0.969	0.899	0.921	0.936	0.949	0.920	0.948	0.960	0.974	0.935	0.954	0.960	0.966
Ours	0.914	0.946	0.964	0.971	0.902	0.924	0.942	0.955	0.924	0.951	0.967	0.974	0.939	0.958	0.969	0.972
Ours + 3D filter	0.915	0.946	0.964	0.971	0.903	0.924	0.943	0.955	0.924	0.951	0.967	0.973	0.939	0.958	0.969	0.972

LPIPS↓	room				counter				kitchen				bonsai			
	1×	2×	4×	8×	1×	2×	4×	8×	1×	2×	4×	8×	1×	2×	4×	8×
Mip-NeRF 360 [2]	0.227	0.101	0.052	0.042	0.216	0.114	0.068	0.059	0.134	0.063	0.033	0.023	0.185	0.065	0.033	0.033
Zip-NeRF [3]	0.199	0.084	0.041	0.028	0.189	0.095	0.055	0.039	0.117	0.055	0.028	0.018	0.173	0.052	0.023	0.017
3DGS [8]	0.254	0.127	0.066	0.053	0.235	0.128	0.078	0.067	0.159	0.081	0.046	0.041	0.234	0.104	0.056	0.051
3DGS-SS	0.241	0.114	0.054	0.039	0.217	0.112	0.066	0.050	0.142	0.068	0.034	0.024	0.220	0.092	0.044	0.032
Mip-Splatting [13]	0.235	0.115	0.062	0.040	0.213	0.116	0.077	0.043	0.138	0.075	0.047	0.027	0.214	0.095	0.058	0.037
Ours	0.234	0.111	0.052	0.035	0.208	0.109	0.065	0.045	0.134	0.065	0.037	0.028	0.210	0.088	0.042	0.029
Ours + 3D filter	0.233	0.110	0.052	0.034	0.209	0.110	0.065	0.045	0.134	0.065	0.037	0.029	0.209	0.088	0.042	0.030

Table 5: Quantative comparisons of Analyti-Splatting against several cutting-edge methods on the multi-scale Mip-NeRF 360 dataset [2,3]. These methods conduct multi-scale training and testing.

	PSNR ↑									
	bicycle	flowers	garden	stump	treehill	room	counter	kitchen	bonsai	Avg.
3DGS	23.14	20.28	24.62	25.44	21.90	30.27	28.08	29.51	30.34	25.95
3DGS-SS	23.98	20.84	25.48	26.24	**22.12**	30.90	28.69	30.53	31.34	26.68
Mip-Splatting	23.82	20.60	24.97	25.78	21.82	30.95	28.68	30.45	31.07	26.46
Ours	**24.22**	**20.97**	**25.72**	**26.29**	22.04	**31.04**	**28.90**	**31.10**	**31.83**	**26.90**

	SSIM ↑									
	bicycle	flowers	garden	stump	treehill	room	counter	kitchen	bonsai	Avg.
3DGS	0.639	0.492	0.707	0.706	0.578	0.902	0.891	0.893	0.916	0.747
3DGS-SS	0.675	0.527	0.747	**0.739**	**0.596**	0.908	0.901	0.907	0.926	0.769
Mip-Splatting	0.671	0.526	0.737	0.728	0.589	0.906	0.898	0.900	0.924	0.764
Ours	**0.683**	**0.535**	**0.761**	**0.739**	**0.596**	**0.910**	**0.904**	**0.911**	**0.930**	**0.774**

	LPIPS ↓									
	bicycle	flowers	garden	stump	treehill	room	counter	kitchen	bonsai	Avg.
3DGS	0.382	0.471	0.325	0.378	0.462	0.324	0.314	0.241	0.321	0.358
3DGS-SS	0.345	0.438	0.281	0.340	0.435	0.314	0.297	0.220	0.307	0.331
Mip-Splatting	**0.341**	0.433	0.291	**0.338**	0.439	0.309	0.295	0.216	**0.300**	0.329
Ours	0.342	**0.429**	**0.268**	**0.338**	**0.434**	**0.307**	**0.291**	**0.209**	**0.300**	**0.324**

Table 6: Quantatitive comparison of super-resolution (2×) on Multi-Scale Mip-NeRF 360. [2, 3] All methods are trained on images with downsampling rates covering [1, 2, 4, 8]. we boldly mark the best results.