# EVER: Exact Volumetric Ellipsoid Rendering for Real-time View Synthesis

Supplementary Material

## A. Backpropagation

We use an adjoint rendering approach, starting from the last state of each ray, which we store, and reconstruct each ray state backwards using $p^{-1}(x, i)$. We can then use this reconstructed state to backpropagate the error to each state, then propagate this error to each primitive. The gradient with respect to mean, scale, and orientation, all come from the derivative of the ray-ellipsoid intersection function. To retrieve the list of primitives intersected, we store the list of intersections on the forward pass. Since rays tend to terminate before 300 total intersections, this turns out to be relatively cheap, at 1.2 KB per a ray. Although we experimented with using ray tracing to retrieve the list of surfaces in reverse order, we found the instability too high.

## B. Hyperparameters, Etc

We change the opacity learning rate to $0.0125$, the initial position learning rate to $4 \times 10^{-5}$ and the final position learning rate to $4 \times 10^{-7}$. We change the parameter known as "percent dense" in 3DGS to $0.001785$, which controls the size threshold above which primitives are split instead of cloned. We perform this every 200 iterations, instead of 100, and set the splitting gradient threshold to $2.5 \times 10^{-7}$ and the clone gradient threshold to $0.1$. We also stop splitting and cloning at 7 million primitives, or at 16000 iterations, which ever comes first, and start at 1500 iterations.

For color, we apply a softplus activation ($\beta = 10$) to the output of the spherical harmonics (instead of 3DGS's relu activation), which we find avoids certain local minima where primitives get locked into a color. We increase the spherical harmonic degree every $2{,}000$ iterations, instead of $1{,}000$. We set the max primitive size to $25$ units, which improves performance.

### B.1. Inverse Contraction Initialization

To help initialize the primitives in a scene, we supplement the SfM initialization with $10{,}000$ additional primitives. We generate these primitives by sampling their means uniformly from a radius-2 sphere. The radius is set to a constant value based the max radius at which the spheres could be packed into the radius-2 sphere, and colors are set to a constant value of $0.5$. These primitives are then transformed by "uncontracting" the resulting means and covariances using the inverse of the contraction used in mip-NeRF 360 [3]. We found that highly anisotropic primitives at initialization can cause issues, so we scale the primitives to be isotropic.

To review, the mip-NeRF 360 contraction function $\mathcal{C}$ that maps from a 3D coordinate in Euclidean space $\mathbf{x}$ to a 3D coordinate in contracted space $\mathbf{z}$ is:

$$\mathcal{C}(\mathbf{x}) = \mathbf{x} \cdot \frac{2\sqrt{\max(1, ||\mathbf{x}||^2)} - 1}{\max(1, ||\mathbf{x}||^2)} \tag{8}$$

The inverse of $\mathcal{C}(\mathbf{x})$ can be defined straightforwardly:

$$\mathcal{C}^{-1}(\mathbf{z}) = \frac{\mathbf{z}}{\sqrt{\max(1, ||\mathbf{z}||^2)}(2 - \min(2, \sqrt{\max(1, ||\mathbf{z}||^2)}))} \tag{9}$$

To apply this inverse contraction to a Gaussian instead of a point, we use the same Kalman-esque approach as was used in mip-NeRF 360: we linearize the contraction around $\mathbf{z}$ into a Jacobian-vector product, which we apply twice to the input covariance matrix.

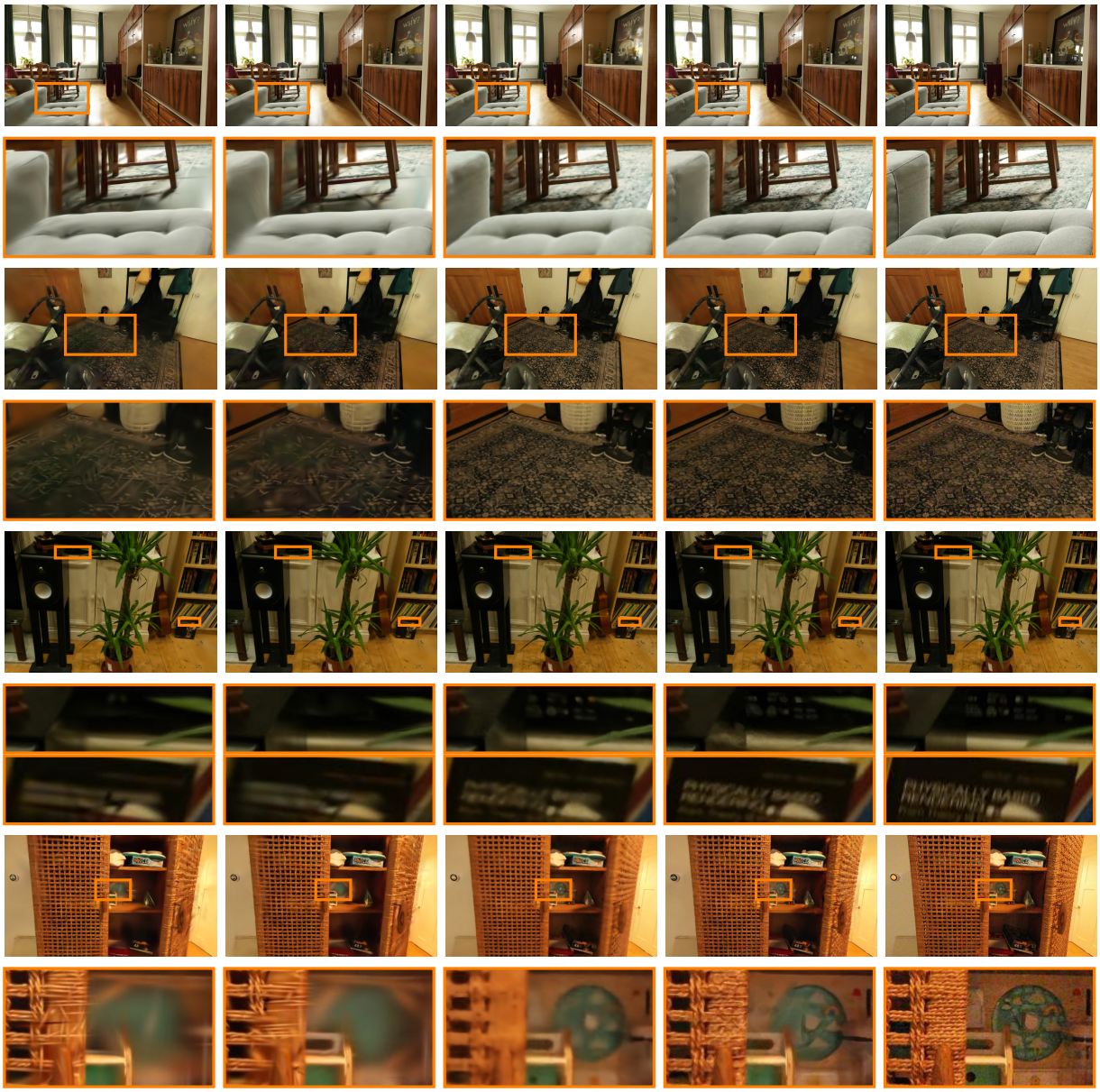


Figure 8. Additional visual comparison of our method on the Mip-NeRF 360 dataset [4].

| **PSNR** ↑ | berlin | nyc | alameda | london | Average |
|---|---|---|---|---|---|
| 3DGS | 26.83 | 26.90 | 24.14 | 25.48 | 25.84 |
| Mip Splatting | 27.30 | 27.52 | 24.76 | 26.28 | 26.46 |
| StopThePop | 26.81 | 27.14 | 24.12 | 25.61 | 25.92 |
| SMERF | 28.52 | 28.21 | 25.35 | 27.05 | 27.28 |
| Ours | 27.24 | 27.93 | 24.72 | 26.49 | 26.60 |
| Zip-NeRF | 28.59 | 28.42 | 25.41 | 27.06 | 27.37 |
| **SSIM** ↑ | berlin | nyc | alameda | london | Average |
| 3DGS | .899 | .861 | .776 | .830 | .842 |
| Mip Splatting | .892 | .853 | .768 | .822 | .834 |
| StopThePop | .885 | .844 | .748 | .801 | .819 |
| SMERF | .887 | .844 | .758 | .829 | .830 |
| Ours | .900 | .863 | .779 | .837 | .845 |
| Zip-NeRF | .891 | .850 | .767 | .835 | .836 |
| **LPIPS** ↓ | berlin | nyc | alameda | london | Average |
| 3DGS | .406 | .380 | .441 | .446 | .418 |
| Mip Splatting | .392 | .356 | .410 | .411 | .392 |
| StopThePop | .402 | .373 | .433 | .438 | .411 |
| SMERF | .391 | .361 | .416 | .390 | .389 |
| Ours | .371 | .337 | .389 | .374 | .368 |
| Zip-NeRF | .378 | .331 | .387 | .360 | .364 |

Table 4. Full results for Zip-NeRF dataset

| **PSNR** ↑ | bicycle | flowers | garden | stump | treehill | room | counter | kitchen | bonsai |
|---|---|---|---|---|---|---|---|---|---|
| 3DGS | 25.24 | 21.55 | 27.38 | 26.56 | 22.43 | 31.53 | 29.00 | 31.45 | 32.21 |
| StopThePop | 25.23 | 21.62 | 27.33 | 26.65 | 22.44 | 30.91 | 28.79 | 31.13 | 31.85 |
| 3DGRT | 25.13 | 21.58 | 26.99 | 26.57 | 22.40 | 30.92 | 28.78 | 30.60 | 31.85 |
| SMERF | 25.58 | 22.24 | 27.66 | 27.19 | 23.93 | 31.38 | 29.02 | 31.68 | 33.19 |
| Our model | 25.34 | 21.70 | 27.46 | 26.41 | 22.74 | 31.39 | 28.91 | 31.36 | 32.24 |
| ZipNeRF | 25.80 | 22.40 | 28.20 | 27.55 | 23.89 | 32.65 | 29.38 | 32.50 | 34.46 |
| **SSIM** ↑ | bicycle | flowers | garden | stump | treehill | room | counter | kitchen | bonsai |
| 3DGS | .766 | .606 | .866 | .771 | .633 | .919 | .909 | .928 | .942 |
| StopThePop | .768 | .607 | .866 | .775 | .635 | .919 | .907 | .927 | .941 |
| 3DGRT | .770 | .624 | .858 | .779 | .636 | .917 | .908 | .924 | .942 |
| SMERF | .760 | .626 | .844 | .784 | .682 | .918 | .892 | .916 | .941 |
| Our model | .776 | .639 | .869 | .781 | .656 | .922 | .910 | .926 | .943 |
| ZipNeRF | .769 | .642 | .860 | .800 | .681 | .925 | .902 | .928 | .949 |
| **LPIPS** ↓ | bicycle | flowers | garden | stump | treehill | room | counter | kitchen | bonsai |
| 3DGS | .240 | .367 | .123 | .251 | .376 | .287 | .258 | .155 | .252 |
| StopThePop | .233 | .362 | .120 | .244 | .366 | .281 | .253 | .154 | .249 |
| 3DGRT | .226 | .335 | .134 | .243 | .364 | .280 | .248 | .156 | .242 |
| SMERF | .239 | .317 | .147 | .243 | .302 | .259 | .256 | .155 | .222 |
| Our model | .220 | .307 | .120 | .230 | .318 | .275 | .240 | .155 | .236 |
| ZipNeRF | .228 | .309 | .127 | .236 | .281 | .238 | .223 | .134 | .196 |

Table 5. Full results for Mip-NeRF 360 dataset