Compressed 3D Gaussian Splatting for Accelerated Novel View Synthesis

Supplementary Material

8. Supplementary

A. Detailed Scene Analysis

For all scenes used in the paper, we report the PSNR, SSIM, LPIPS, memory consumption, and compression ratio of our approach. See Tab. 8 for Mip-Nerf360 [2], Tab. 10 for Deep Blending [9], Tab. 9 for Tanks&Temples [14], Tab. 11 for the synthetic scenes from [16].

B. Image Quality

For all scenes used in the paper, a random test view is selected. The ground truth images are compared to the renderings of the uncompressed (baseline) and our compressed (Compressed) scene representation. See Figs. 11 and 12 for Mip-Nerf360 [2], Fig. 10 for Deep Blending [9], Fig. 9 for Tanks & Temples [14], Figs. 13 and 14 for the synthetic scenes from [16].

C. Memory Requirements

Fig. 7 illustrates the memory requirements of different scene parameters. The coordinates of the 3D Gaussian center points and the codebook indices take up the most memory in general. The amount of memory required by the color codebook varies significantly between different scenes.

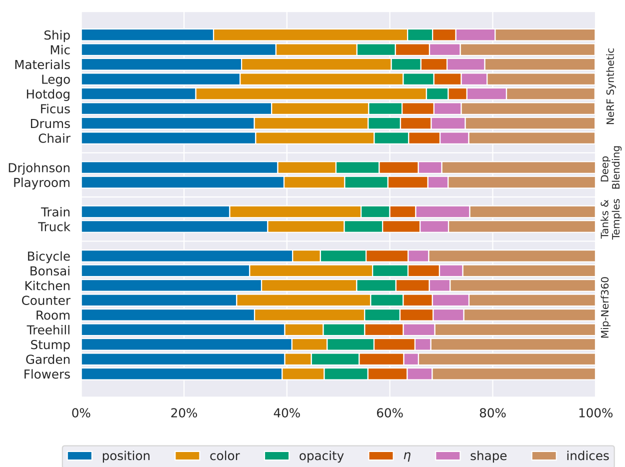

Figure 7. Storage size of different scene parameters in the compressed representation. Color is the codebook with all SH coefficients. Shape is the codebook with the Gaussian parameters and η is the scaling factor.

D. Timing Statistics

We provide timings for the different stages of our compression pipeline. Tab. 6 shows the average and maximum time required by each stage. It can be seen that the fine-tuning stage takes up 70% of the total time.

	Average Time ↓	Maximum Time ↓
Sensitivity Calculation	8.05	11.38
Clustering	75.11	78.41
QA Fine-tuning	213.30	278.05
Encoding	2.69	5.13
Total	299.15	365.94

Table 6. Time requirements of the individual stages of the compression pipeline. We report the average and maximum time of each stage in seconds. The entropy and run-length encoding are grouped into the Encoding stage. Measurements were taken with an NVIDIA RTX A5000 graphics card.

Additionally, we report timings for each stage of the novel view renderer. Tab. 7 shows the average times for two different scenes. It can be seen that the preprocessing stage is accelerated by a factor of $5\times$ when using the compressed scene representation.

		Preprocess ↓	Sorting ↓	Rasterization ↓	Total ↓
Bicycle	Uncompressed	1.46	0.55	2.81	4.82
	Compressed	0.28	0.48	2.45	3.22
Bonsai	Uncompressed	0.44	0.20	1.81	2.44
	Compressed	0.09	0.19	1.67	1.95

Table 7. Timings in milliseconds for the different stages of our renderer. Evaluated on an NVIDIA RTX A5000 with scenes from Mip-Nerf360 [2]

(a) Baseline (b) Compressed

Figure 8. Pruning failure case. Compared to the baseline reconstruction, some leaves have been removed in the compressed version due to pruning.

E. Sensitivity Calculation and Pruning

The sensitivity of a parameter is calculated using the gradient of the total image energy wrt. this parameter (see Eq. (3)). Kerbl *et al.* [13] clamp negative direction-dependent colors (i.e., resulting from the evaluation of the SH coefficients) to zero. For the clamped values, the partial derivatives are set to zero in the backward pass. This results in a sensitivity of zero for the respective SH coefficients, which is not desired since they possibly contribute to the training images. Therefore, we do not clamp colors when calculating the sensitivity.

We observe that a notable number of Gaussians (up to 15%) do not have any impact on the training images. These particular splats exhibit zero sensitivity in the color parameters. Consequently, we opt to eliminate these splats from the scene (called Pruning in Tab. 3).

Experiments with higher pruning thresholds have shown that more Gaussians can be removed with minimal loss in PSNR. However, this can lead to fine details in the scene being removed, which we consider undesirable. An example of this can be seen in Fig. 8, where small leaves were removed from the reconstruction due to pruning.

F. Covariance Matrix Clustering

Given a rotation matrix $R \in \mathbb{R}^{3\times 3}$ and a scaling vector $\mathbf{s} \in \mathbb{R}^3_{>0}$. The covariance matrix Σ is defined as [13]

$$\Sigma = RSSR^T = RS^2R^T, \tag{8}$$

with $S = \mathrm{diag}(\mathbf{s})$. Since Σ is real and symmetric it holds that

$$S^2 = \mathrm{diag}([\lambda_1, \lambda_2, \lambda_3]^T) = \mathrm{diag}([s_1^2, s_2^2, s_3^2]^T), \tag{9}$$

where λ_i are the eigenvalues of Σ. By using the trace of Σ, the squared length of $\mathbf{s}$ can be calculated as

$$\mathrm{Tr}(\Sigma) = \sum_{i=1}^{3} \lambda_i = \sum_{i=1}^{3} s_i^2 = \|\mathbf{s}\|_2^2 \tag{10}$$

Clustering Update Step In the following, we show that the clustering update step results in normalized covariance matrices as cluster centroids. Given N normalized covariance matrices $\hat{\Sigma}_i$ with $\|\mathbf{s_i}\|_2 = 1$ and respective weighting factors $w_i \in \mathbb{R}_{>0}$. Their centroid $\hat{\Sigma}_c$ is calculated as

$$\hat{\Sigma}_c = \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^{N} w_i \hat{\Sigma}_i \tag{11}$$

. By using Eq. (10) it holds that

$$\begin{aligned}
\|\mathbf{s_c}\|_2^2 &= \mathrm{Tr}(\hat{\Sigma}_c) && (12)\\
&= \mathrm{Tr}(\frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^{N} w_i \hat{\Sigma}_i) && (13)\\
&= \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^{N} w_i \mathrm{Tr}(\hat{\Sigma}_i) && (14)\\
&= \frac{1}{\sum_{i=1}^N w_i} \sum_{i=1}^{N} w_i \|\mathbf{s_i}\|_2^2 && (15)\\
&= 1 && (16)
\end{aligned}$$

This proves that the covariance matrix $\hat{\Sigma}_c$ has a normalized scaling vector and thus iteself is in a normalized form.

Covariance Matrix Normalization The following derivation proofs that a covariance matrix Σ can be transformed into its normalized form $\hat{\Sigma}$ by dividing it by its trace, i.e.,

$$\begin{aligned}
\frac{\Sigma}{\mathrm{Tr}(\Sigma)} &= R\frac{S^2}{\mathrm{Tr}(\Sigma)}R^T = R\frac{S}{\|\mathbf{s}\|_2}\frac{S}{\|\mathbf{s}\|_2}R^T && (17)\\
&= R\hat{S}^2R^T = \hat{\Sigma} && (18)
\end{aligned}$$

	3D Gaussian Splatting				Ours				
Scene	PSNR ↑	SSIM ↑	LPIPS ↓	SIZE ↓	PSNR ↑	SSIM ↑	LPIPS ↓	SIZE ↓	Ratio ↑
bicycle	25.171	0.762	0.216	1450.277	24.970	0.751	0.240	47.147	30.761
bonsai	31.979	0.938	0.208	294.415	31.347	0.930	0.217	12.794	23.011
counter	28.888	0.905	0.204	289.244	28.671	0.896	0.215	13.789	20.977
flowers	21.448	0.602	0.341	860.062	21.152	0.584	0.358	31.140	27.619
garden	27.179	0.861	0.115	1379.993	26.746	0.844	0.144	46.565	29.636
kitchen	30.713	0.923	0.130	438.099	30.262	0.914	0.140	18.874	23.211
room	31.341	0.916	0.223	376.853	31.138	0.911	0.231	15.033	25.068
stump	26.562	0.770	0.219	1173.522	26.285	0.757	0.250	40.569	28.926
treehill	22.303	0.631	0.328	894.903	22.256	0.620	0.351	33.318	26.859
average	27.287	0.812	0.220	795.263	26.981	0.801	0.238	28.803	26.230

Table 8. Mip-Nerf360 [2] results.

	3D Gaussian Splatting				Ours				
Scene	PSNR ↑	SSIM ↑	LPIPS ↓	SIZE ↓	PSNR ↑	SSIM ↑	LPIPS ↓	SIZE ↓	Ratio ↑
train	21.770	0.805	0.217	242.782	21.863	0.798	0.226	13.249	18.324
truck	24.940	0.871	0.155	601.030	24.823	0.867	0.161	21.316	28.196
average	23.355	0.838	0.186	421.906	23.343	0.832	0.194	17.282	23.260

Table 9. Tanks&Temples [14] results

	3D Gaussian Splatting				Ours				
Scene	PSNR ↑	SSIM ↑	LPIPS ↓	SIZE ↓	PSNR ↑	SSIM ↑	LPIPS ↓	SIZE ↓	Ratio ↑
drjohnson	28.938	0.896	0.248	805.358	28.871	0.895	0.254	28.938	27.830
playroom	29.926	0.901	0.244	602.186	29.891	0.900	0.252	21.660	27.802
average	29.432	0.898	0.246	703.772	29.381	0.898	0.253	25.299	27.816

Table 10. Deep Blending [9] results

	3D Gaussian Splatting				Ours				
Scene	PSNR ↑	SSIM ↑	LPIPS ↓	SIZE ↓	PSNR ↑	SSIM ↑	LPIPS ↓	SIZE ↓	Ratio ↑
chair	35.864	0.987	0.012	70.105	35.297	0.985	0.014	3.575	19.609
drums	26.072	0.954	0.038	83.665	25.941	0.952	0.040	3.829	21.848
ficus	34.736	0.987	0.012	70.177	34.559	0.986	0.013	3.059	22.937
hotdog	37.646	0.985	0.021	34.079	37.367	0.984	0.022	2.725	12.505
lego	35.399	0.981	0.017	76.071	34.802	0.979	0.020	4.314	17.633
materials	29.861	0.959	0.035	71.833	29.602	0.957	0.038	4.021	17.862
mic	35.155	0.991	0.006	77.563	34.913	0.991	0.007	3.025	25.640
ship	30.954	0.905	0.111	75.659	31.005	0.905	0.111	4.938	15.322
average	33.211	0.969	0.031	69.894	32.936	0.967	0.033	3.686	19.170

Table 11. NeRF Synthetic [16] results

Figure 9. Random test views for each scene from Tanks&Temples [14]

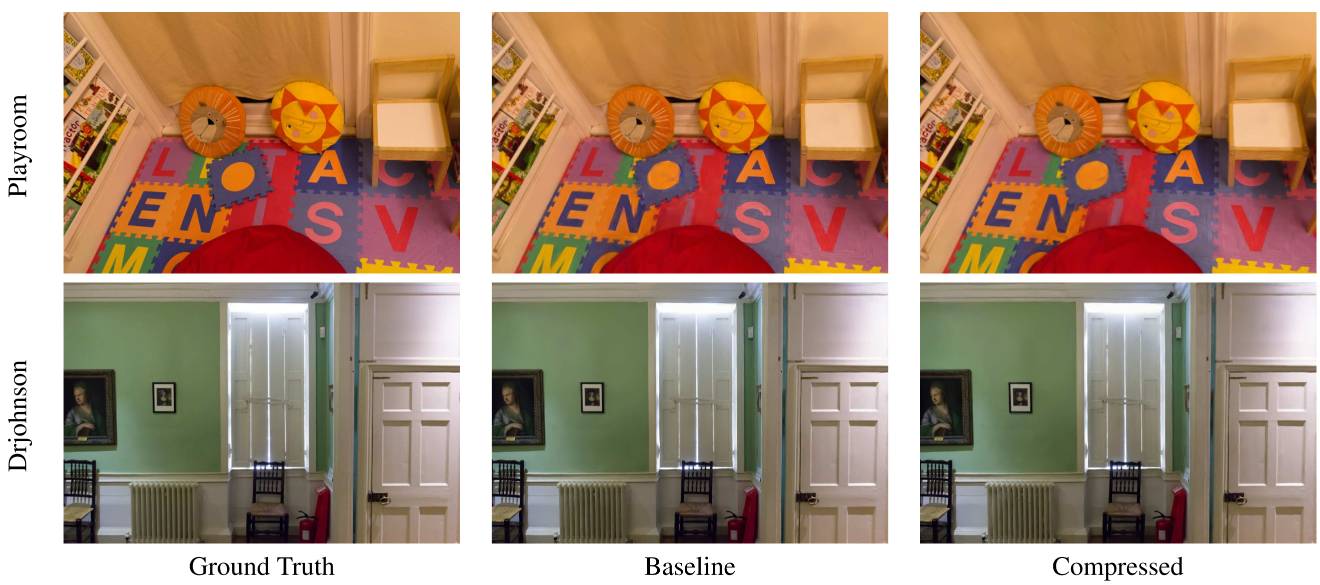

Figure 10. Random test views for each scene from Deep Blending [9]

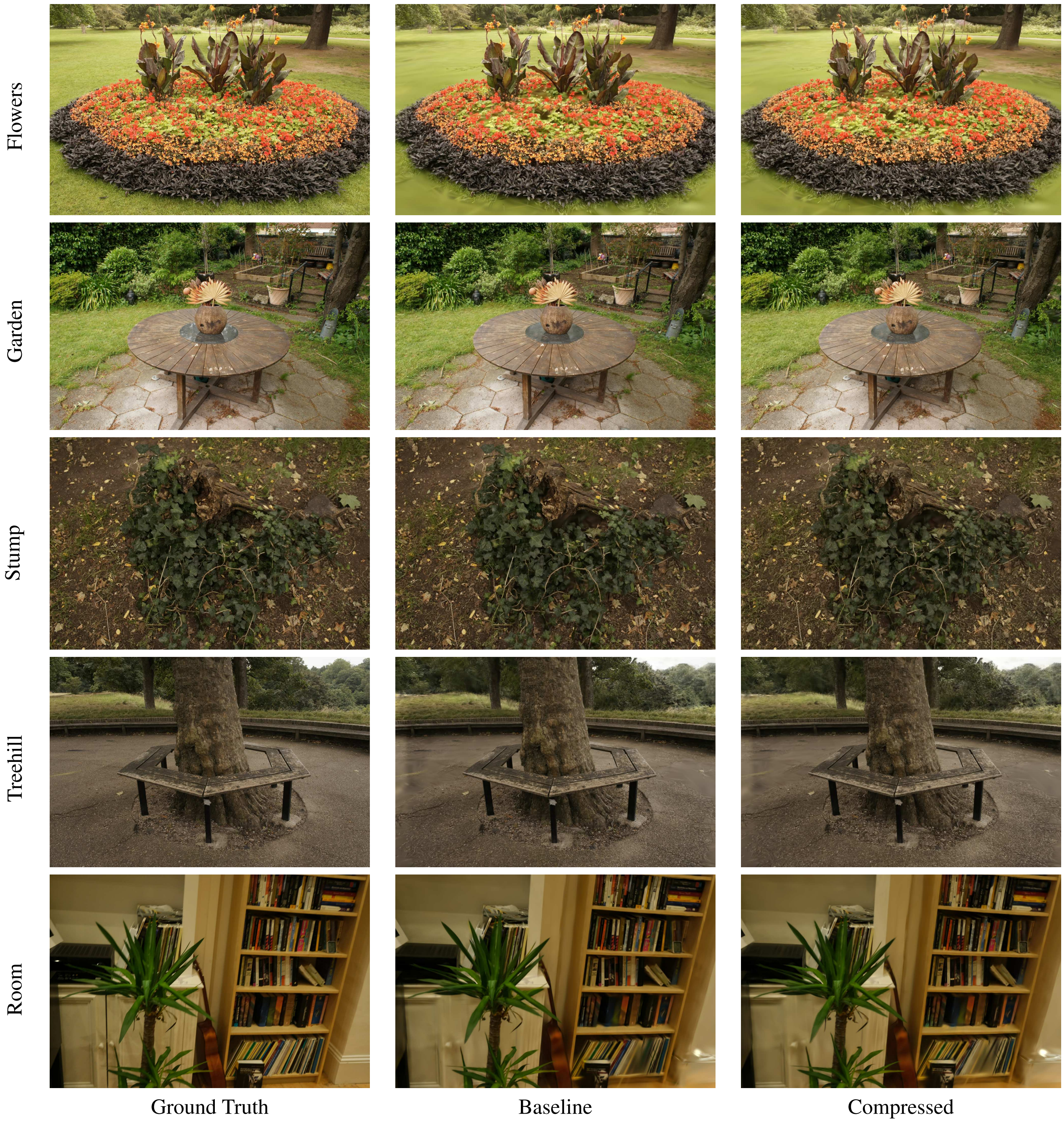

Figure 11. Random test views for each scene from Mip-NeRF360 [2]

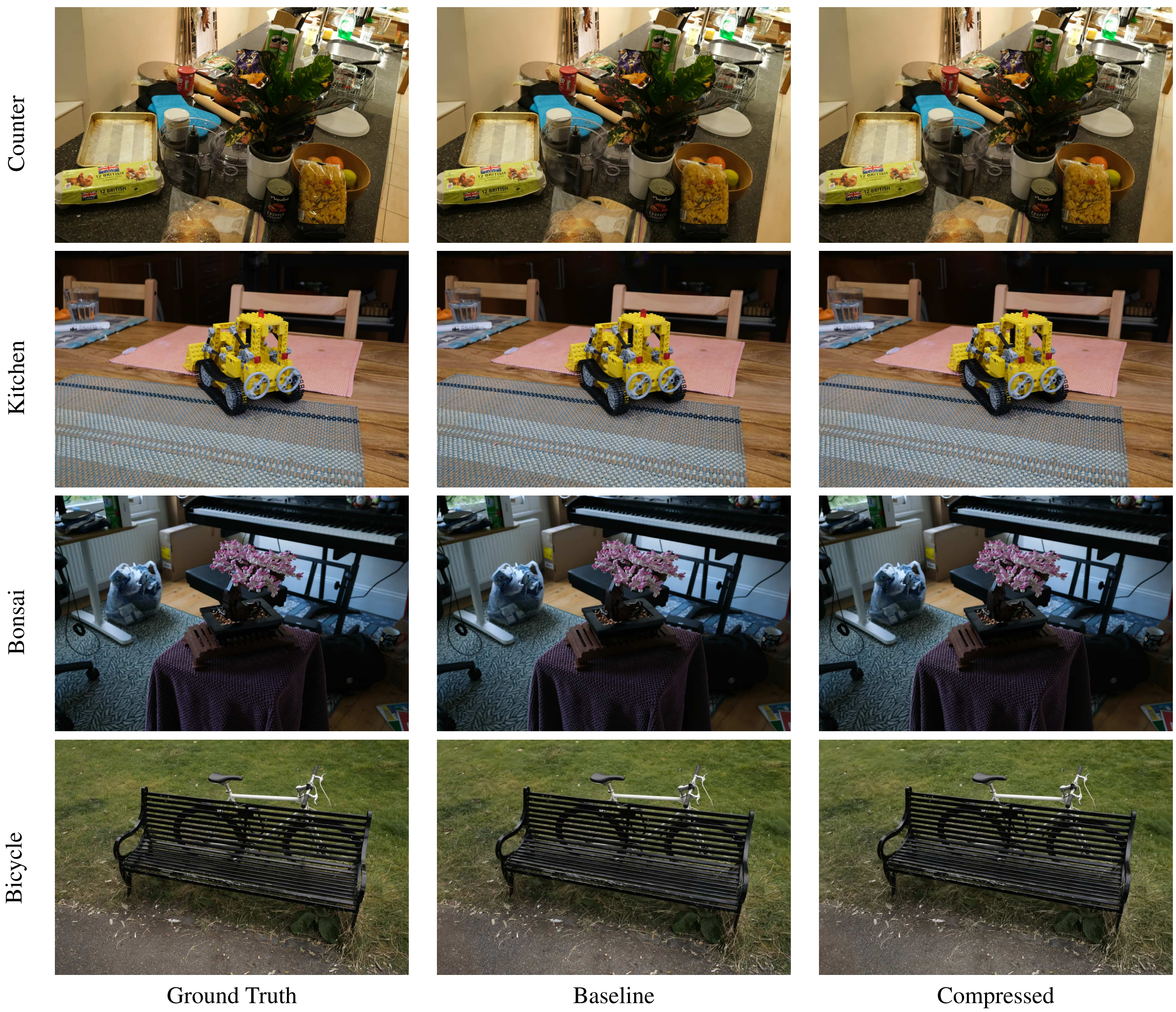

Figure 12. Random test views for each scene from Mip-NeRF360 [2]

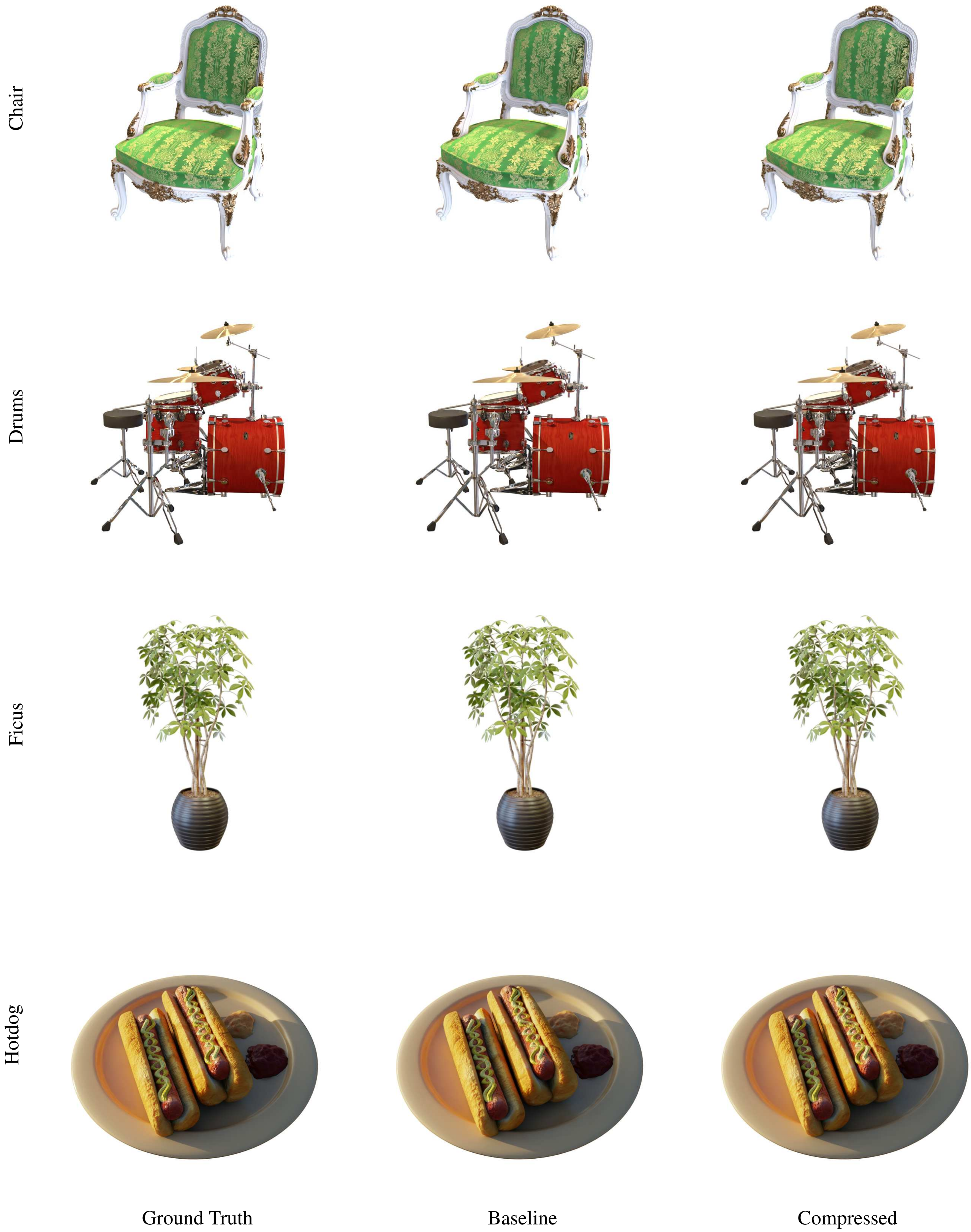

Figure 13. Random test views for each scene from NeRF Synthetic [16]

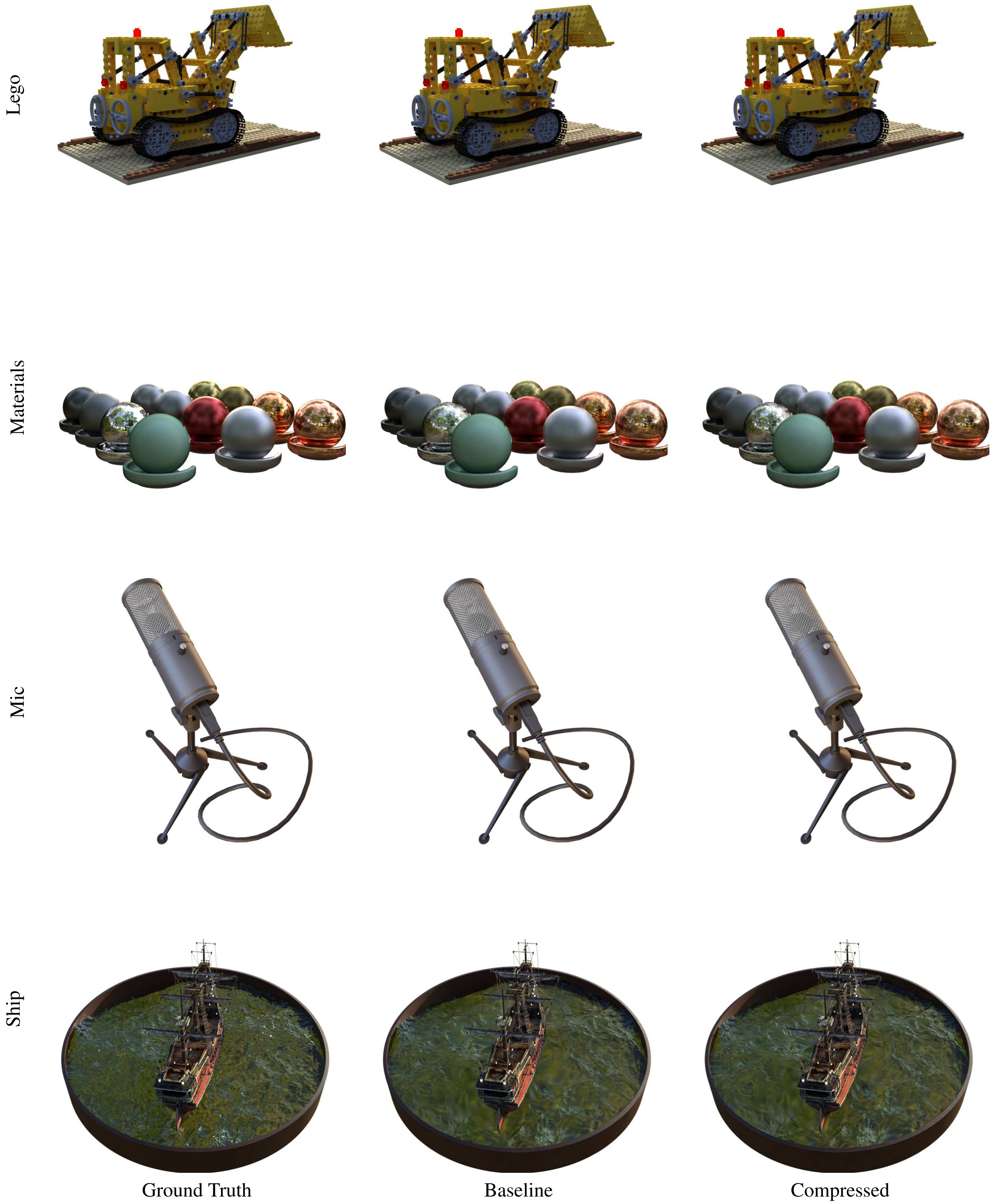

Figure 14. Random test views for each scene from NeRF Synthetic [16]