Supplementary Material for RadSplat: Radiance Field-Informed Gaussian Splattingfor 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 Federico Tombari
Google
m-niemeyer.github.io/radsplat

Abstract

*In this supplementary material, we discuss additional implementation details in Sec. 1 and report additional quantitative and qualitative results in Sec. 2. In the **supplementary video**, we show camera trajectories for our models and baselines on unbounded as well as large-scale scenes.*

1. Additional Implementation Details

1.1. 3D Gaussian Splatting with Exposure Module

To ensure a fair visual comparison to 3DGS [6], we follow [3] and utilize an exposure module that can compensate for varying exposure in the input images for the "Alameda" scene from the ZipNeRF dataset (see Fig. 1). More specifically, this module consists of an MLP that maps a predicted color and an per image exposure metric to a exposure-corrected color value. We use a 4-layer MLP with 10 units and define the exposure metric of image i using the ISO_i value and the shutter time Δt_i[1]

$$\epsilon_i = \log\left(\frac{ISO_i * \Delta t_i}{1000}\right) \tag{1}$$

The module is initialized with a pre-optimization procedure leading to an identity mapping between the input and output color (see [3] for more details). Afterwards, the MLP weights are optimized together with the 3DGS scene. During evaluation for every test image i, we apply the input exposure value ϵ_i for predicting color values as close as possible to the GT values. For rendering of videos, we fix the exposure value to the median value leading to a more consistent video.

1.2. FPS Measurement

We follow [3] and render every test image 100 times on a desktop computer with a NVIDIA GeForce RTX 3090 Ti for evaluating the frames per second. We take the GPU timings of the rendering forward step to measure the frame render times and calculate the FPS value from the mean of the frame render times. During rendering, we use the individual test image resolution from the datasets and apply 100 warm up renderings before measuring the frame render time following [3].

1.3. Viewpoint-Based Visibility Filtering

In Sec. 3.4 of the main paper, we discuss our viewpoint-based visibility filtering that increases rendering speed without a drop in quality. As mentioned in the Visibility Filtering subsection, we compute importance scores and respective visibility indicator lists from the input and random cameras to increase robustness. More specifically, we uniformly sample 100k novel viewpoint positions in a region around the training cameras. The set of viewpoint positions is defined as the union of the positions of training cameras and uniformly sampled novel 3D points around training cameras. More specifically, after uniformly sampling novel viewpoint positions in space, we select the positions that are closer to the closest train camera than

[1] It is worth noting that the exposure module for our own model (during the NeRF training) does not require this additional ISO input.

(a) 3DGS [6] (b) 3DGS w/ expos. module (c) Ours

Figure 1. **Additional Robustness Results.** On complex captures with lighting variations, 3DGS [6] leads to degraded results (1a). When equipped with exposure handling modules [3, 6], results improve yet still suffer from floating artifacts and are overly smooth (1b). Our radiance field-informed approach instead achieves high quality even for challenging captures (1c).

a threshold value t_{dist}. We arbitrarily choose the threshold t_{dist} as 10 times the average smallest distance between training cameras. Finally, we apply visibility filtering by rendering with six random camera rotations from the selected viewpoints.

1.4. Datasets

We report results on all scenes from the commonly-used MipNeRF360 dataset [1] and follow common practice to use a downsampling factor or two for indoor scenes and a downsampling factor of four for outdoor scenes. For the ZipneRF dataset [2], we follow [3] to use the undistorted variant and report results for all four scenes with a downsampling factor of two where for Berlin, we limit the maximal image extent to 1600 pixels to avoid OOM. As discussed in the main paper, we utilize the robust NeRF prior among others as supervision which enables us, in contrast to, e.g. 3DGS [6], to also train from data captured with fisheye or other lens types. We show a comparison to ZipNeRF on "Alameda" in the supplemental video where both models were trained on the original data. Please note that all quantitative and qualitative comparisons in the paper are reported wrt. the undistorted dataset to ensure a fair comparison for all methods.

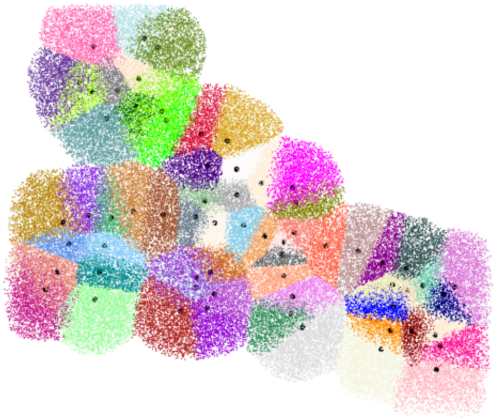


Figure 2. **Viewpoint-Based Visibility Filtering.** To increase robustness of our viewpoint-based visibility filtering, we densely sample novel viewpoints (colored points) around each cluster center (black). The figure shows the 2D projection along the z-axis.

	bicycle	flowers	garden	stump	treehill	room	counter	kitchen	bonsai	mean
BakedSDF [10]	0.570	0.452	0.751	0.595	0.559	0.870	0.808	0.817	0.851	0.697
MERF [9]	0.595	0.492	0.763	0.677	0.554	0.874	0.819	0.842	0.884	0.722
INGP [8]	0.540	0.378	0.709	0.654	0.547	0.893	0.845	0.857	0.924	0.705
SMERF [3]	0.760	0.626	0.844	0.784	0.682	0.918	0.892	0.916	0.941	0.818
LightG [4]	0.738	0.560	0.836	0.768	0.623	0.926	0.893	0.914	0.930	0.799
CompactG [7]	0.723	0.556	0.832	0.757	0.638	0.919	0.902	0.919	0.939	0.798
EAGLES [5]	0.750	0.570	0.840	0.770	0.640	0.930	0.910	0.930	0.940	0.809
3DGS [6]	0.771	0.605	0.868	0.775	0.638	0.914	0.905	0.922	0.938	0.815
Ours Light	0.764	0.647	0.850	0.803	0.676	0.923	0.902	0.924	0.944	0.826
Ours	0.800	0.665	0.881	0.820	0.683	0.933	0.918	0.937	0.952	0.843
ZipNeRF [2]	0.784	0.655	0.872	0.807	0.688	0.927	0.909	0.926	0.953	0.836

Table 1. **Per-Scene SSIM on MipNeRF360.**

	bicycle	flowers	garden	stump	treehill	room	counter	kitchen	bonsai	mean
BakedSDF [10]	22.04	19.53	24.94	23.59	22.25	28.68	25.69	26.72	27.17	24.51
MERF [9]	22.62	20.33	25.58	25.04	22.39	29.28	25.82	27.42	28.68	25.24
INGP [8]	22.79	19.19	25.26	24.80	22.46	30.31	26.21	29.00	31.08	25.68
SMERF [3]	25.58	22.24	27.66	27.19	23.93	31.38	29.02	31.68	33.19	27.99
LightG [4]	24.96	21.17	26.73	26.70	22.55	31.27	28.11	30.40	31.01	26.99
CompactG [7]	24.77	20.89	26.81	26.46	22.65	30.88	28.71	30.48	32.08	27.08
EAGLES [5]	24.91	21.00	26.88	26.59	22.57	31.84	28.47	30.70	31.44	27.16
3DGS [6]	25.25	21.52	27.41	26.55	22.49	30.63	28.70	30.32	31.98	27.20
Ours Light	25.32	22.08	27.40	27.38	23.35	30.95	28.40	31.41	31.76	27.56
Ours	25.99	22.24	28.23	27.74	23.35	31.62	29.05	32.40	32.67	28.14
ZipNeRF [2]	25.80	22.40	28.20	27.55	23.89	32.65	29.38	32.50	34.46	28.54

Table 2. **Per-Scene PSNR on MipNeRF360.**

	bicycle	flowers	garden	stump	treehill	room	counter	kitchen	bonsai	mean
BakedSDF [10]	0.368	0.429	0.213	0.371	0.366	0.251	0.286	0.237	0.259	0.309
MERF [9]	0.371	0.406	0.215	0.309	0.414	0.292	0.307	0.224	0.262	0.311
INGP [8]	0.398	0.441	0.255	0.339	0.420	0.242	0.255	0.170	0.198	0.302
SMERF [3]	0.225	0.305	0.141	0.220	0.266	0.208	0.212	0.134	0.191	0.211
LightG [4]	0.265	0.396	0.155	0.261	0.385	0.220	0.218	0.147	0.204	0.250
CompactG [7]	0.286	0.399	0.161	0.278	0.363	0.209	0.205	0.131	0.193	0.247
EAGLES [5]	0.260	0.380	0.160	0.260	0.360	0.200	0.200	0.130	0.190	0.238
3DGS [6]	0.201	0.336	0.103	0.210	0.317	0.221	0.204	0.130	0.206	0.214
Ours Light	0.234	0.312	0.144	0.213	0.316	0.192	0.199	0.135	0.175	0.213
Ours	0.170	0.267	0.092	0.169	0.263	0.164	0.160	0.103	0.150	0.171
ZipNeRF [2]	0.188	0.254	0.104	0.180	0.221	0.189	0.173	0.117	0.167	0.177

Table 3. **Per-Scene LPIPS on MipNeRF360.**

	bicycle	flowers	garden	stump	treehill	room	counter	kitchen	bonsai	mean
LightG [4]	1.92	1.07	1.92	1.55	1.02	0.51	0.40	0.59	0.43	1.05
CompactG [7]	2.22	1.53	2.21	1.73	2.01	0.53	0.54	1.13	0.60	1.39
EAGLES [5]	2.46	1.55	1.92	2.44	1.79	0.80	0.71	1.28	2.46	1.71
3DGS [6]	5.72	3.41	5.64	4.55	3.47	1.48	1.17	1.74	1.25	3.16
Ours Light	0.50	0.52	0.54	0.39	0.45	0.22	0.20	0.27	0.24	0.37
Ours	3.14	2.15	2.51	2.74	2.39	1.15	1.04	1.21	0.98	1.92

Table 4. **Per-Scene Num. Gaussians (mio) on MipNeRF360.**

	bicycle	flowers	garden	stump	treehill	room	counter	kitchen	bonsai	mean
BakedSDF [10]	485	368	385	552	571	535	476	893	585	539
MERF [9]	196	190	202	148	102	229	184	155	137	171
INGP [8]	9.1	8.8	7.6	13.0	8.2	15.1	8.3	6.4	6.8	9.3
SMERF [3]	228	213	210	259	247	330	238	181	141	228
LightG [4]	124	180	155	229	159	311	213	204	308	209
CompactG [7]	76.4	142	89.5	121	110	183	120	114	196	128
EAGLES [5]	97.0	161	123	141	132	143	147	107	184	137
3DGS [6]	177	342	288	218	402	169	322	312	303	281
Ours Light	962	913	942	1040	992	953	748	734	879	907
Ours	333	483	378	404	423	475	372	335	485	410
ZipNeRF [2]	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25

Table 5. **Per-Scene FPS on MipNeRF360.**

2. Additional Experimental Results

2.1. Unbounded MipNeRF360 Scenes

We report per-scene SSIM in Tab. 1, PSNR in Tab. 2, LPIPS in Tab. 3, point count in Tab. 4, and FPS in Tab. 5. We observe that our model achieves state-of-the-art real time view synthesis on all metrics and even outperforms ZipNeRF in SSIM, the offline state-of-the-art. We further find that our models achieve higher FPS than prior works, in particular our lightweight variant enables high-quality view synthesis at 907 FPS, which is 1.7x faster than the state-of-the-art mesh-based method BakedSDF [10] and 4x faster than the previous state-of-the-art real-time method SMERF [3]. In Fig. 4, Fig. 5, Fig. 6, Fig. 7, Fig. 8 we show additional qualitative comparisons on the MipNeRF360 dataset.

2.2. Large-Scale ZipNeRF Scenes

We report per-scene SSIM in Tab. 6a, PSNR in Tab. 6b, LPIPS in Tab. 7a, and FPS in Tab. 7b. We observe that our models improve over prior real-time view synthesis methods in SSIM and even slightly outperform ZipNeRF, the offline state-of-the-

	Berlin	NYC	Alameda	London	mean
MERF [9]	0.840	0.765	0.646	0.737	0.747
SMERF [3] ($K = 1$)	0.852	0.771	0.701	0.780	0.776
SMERF [3] ($K = 5$)	0.887	0.844	0.758	0.829	0.829
3DGS [6]	0.879	0.829	0.733	0.797	0.809
Ours Light	0.899	0.858	0.760	0.835	0.838
Ours	0.901	0.860	0.760	0.836	0.839
ZipNeRF [2]	0.891	0.850	0.767	0.835	0.836

(a) Per-Scene SSIM on ZipNeRF.

	Berlin	NYC	Alameda	London	mean
MERF [9]	25.27	24.82	20.34	23.53	23.49
SMERF [3] ($K = 1$)	26.79	25.40	23.71	25.86	25.44
SMERF [3] ($K = 5$)	28.52	28.21	25.35	27.05	27.28
3DGS [6]	26.60	26.41	23.52	25.45	25.50
Ours Light	27.76	27.54	22.55	26.60	26.11
Ours	27.96	27.59	22.48	26.64	26.17
ZipNeRF [2]	28.59	28.42	25.41	27.06	27.37

(b) Per-Scene PSNR on ZipNeRF.

Table 6. **Per-Scene SSIM and PSNR on ZipneRF.**

	Berlin	NYC	Alameda	London	mean
MERF [9]	0.395	0.426	0.501	0.456	0.445
SMERF [3] ($K = 1$)	0.380	0.417	0.445	0.407	0.412
SMERF [3] ($K = 5$)	0.325	0.321	0.370	0.342	0.340
3DGS [6]	0.335	0.343	0.407	0.392	0.369
Ours Light	0.341	0.348	0.410	0.372	0.368
Ours	0.338	0.342	0.406	0.369	0.364
ZipNeRF [2]	0.297	0.281	0.338	0.304	0.305

(a) Per-Scene LPIPS on ZipNeRF.

	Berlin	NYC	Alameda	London	mean
MERF [9]	185	470	270	348	318
SMERF [3] ($K = 1$)	220	481	358	364	356
SMERF [3] ($K = 5$)	131	262	230	260	221
3DGS [6]	369	417	467	625	470
Ours Light	533	859	788	811	748
Ours	467	677	666	709	630
ZipNeRF [2]	0.25	0.25	0.25	0.25	0.25

(b) Per-Scene FPS on ZipNeRF.

Table 7. **Per-Scene LPIPS and FPS on ZipneRF.**

art. We find that for PSNR and LPIPS, we rival the large SMERF model with $5^3 = 128$ submodels, the state-of-the-art for large-scale real-time view synthesis, while rendering 3.4x (our light variant) and 2.9x (our default) faster. In Fig. 9 and Fig. 10, we show additional qualitative results on the ZipNeRF dataset.

2.3. Ablation of Initial Point Set

	SSIM↑	PSNR↑	LPIPS↓	#G(M)↓
Ours	0.800	26.02	0.171	3.14
Mean D.	0.797	25.95	0.175	3.02

(a) Median vs Mean Depth.

	SSIM↑	PSNR↑	LPIPS↓	#G(M)↓
Ours	0.800	26.02	0.171	3.14
1.5M	0.802	26.06	0.166	3.25
2M	0.803	26.05	0.164	3.39
2.5M	0.803	25.98	0.161	3.51
3M	0.805	26.06	0.160	3.67

(b) Ablation of Initial Point Count.

Table 8. **Initial Point Set Ablation Studies on Bicycle Scene.** Left: We find that our median depth initialization leads to improved results compared to, e.g., the commonly chosen mean depth (Mean D.). Right: For simplicity, we always use an initial point set of size one million. We find that larger initial point sets lead to increased final scene sizes while only marginally improving view synthesis metrics.

In Tab. 8, we report additional ablation studies on the initial point set that is used as initialization for our point-based scene optimization (see Sec. 3.2 of the main paper for more context). As outlined in the main paper, we use the "median depth" from our NeRF model to calculate the initial point set. We find that using the commonly-chosen mean depth (also called "expected depth") leads to reduced quality. We hypothesize that using exact sample point estimates, as done for the median depth, ensures a better initialization.

As discussed in the main manuscript, for simplicity, we always use an initial point set of size one million. We observe that using a larger number of initial points leads to an increase in final scene size while only marginally improving in view synthesis metrics.

2.4. Ablation of the NeRF Prior

Overall, our approach is robust to the prior choice (see Tab. 9). For lightweight variants, the difference is small and INGP can be used as prior to save computation. To obtain best possible results, however, the ZipNeRF prior is still required. It is worth nothing that our contributions are efficient test-time rendering with SOTA quality, but not more efficient training.

	SSIM ↑	PSNR ↑	LPIPS
Ours L. (INGP)	0.826	27.35	0.259
Ours L. (ZipNeRF)	0.826	27.56	0.213
Ours (INGP)	0.837	27.37	0.213
Ours (ZipNeRF)	0.843	28.14	0.171

Table 9. **Robustness to the Choice of the NeRF Prior.** We report our model with the ZipNeRF (our default) vs. INGP as prior on the mip-NeRF 360 dataset. We find that our model is robust to the prior and best results are obtained with our default variant.

Initialization from SfM. Initialization from NeRF (Ours).

Figure 3. **Qualitative Ablation of the NeRF Prior.** We find that, compared to the 3DGS standard SfM initialization, our NeRF prior captures the scene more homogeneously leading to more stable GS optimization with less local minima.

	MERF [9]	SMERF$_1$ [3]	SMERF$_5$ [3]	3DGS [6]	Ours Light	Ours	ZipNeRF [2]
mip-NeRF360 Dataset							
Size (GB) ↓	0.16	0.14	-	0.74	0.07	0.40	0.61
SSIM ↑	0.722	0.818	-	0.815	0.826	0.843	0.836
ZipNeRF Dataset							
Size (GB) ↓	0.13	0.12	4.11	0.21	0.24	0.41	0.61
SSIM ↑	0.747	0.776	0.829	0.809	0.838	0.839	0.836

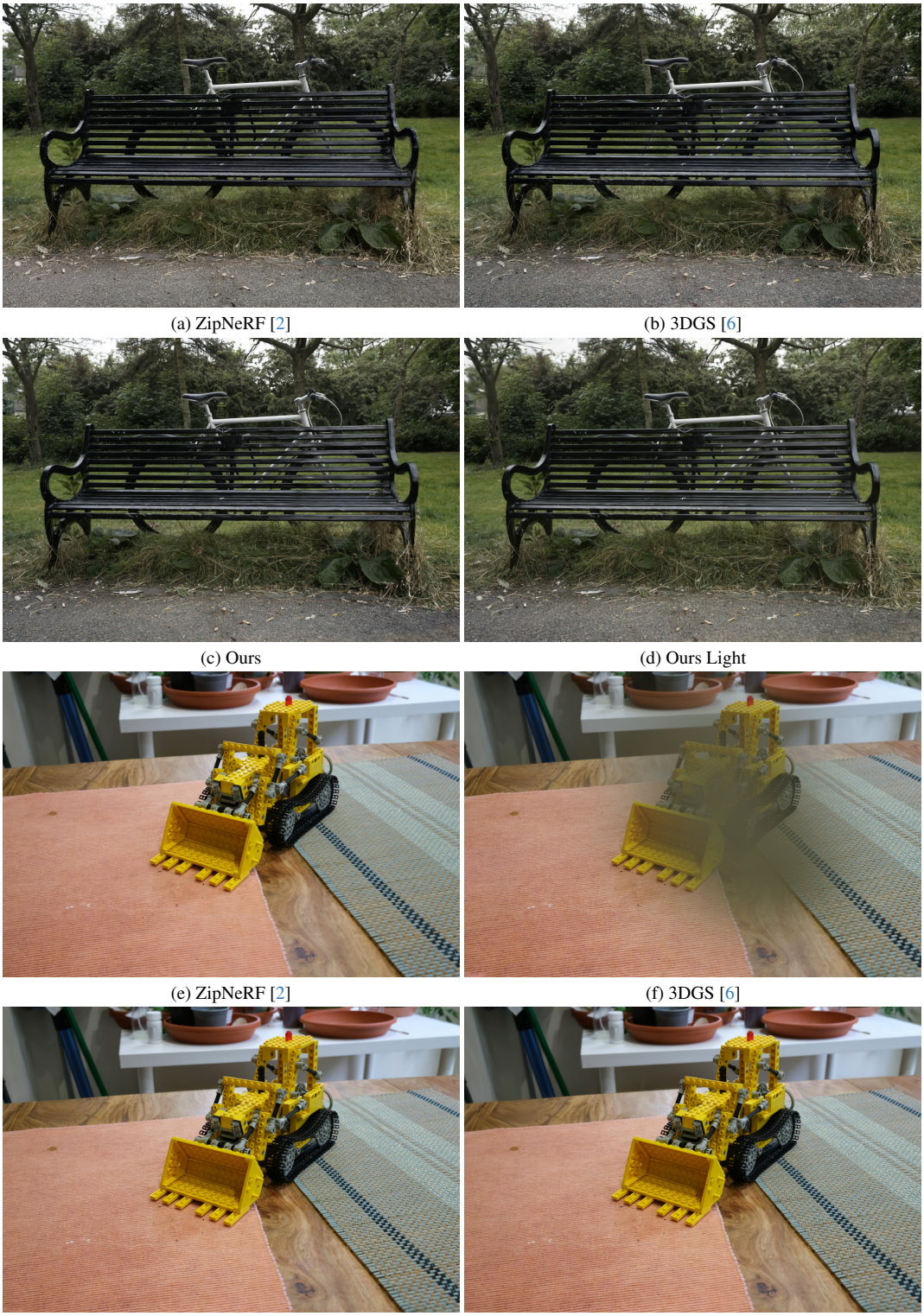
Table 10. **File Size Comparison.** We compare our raw file sizes to baseline methods in gigabytes (GB) above. While obtaining decent raw sizes (e.g., 70MB on average for our light model on the mip-NeRF 360 dataset), early experiments show that we can get a 10× reduction for ours and 3DGS when compression is applied (see text).

In Fig. 3, we further show a qualitative ablation study comparing our NeRF-based intialization to the 3DGS standard SfM initialization. We find that our approach covers the scene more homogeneously, resulting in a more stable optimization with less local minima.

2.5. File Size Comparison

We compare our raw file sizes to baselines in Tab. 10. While obtaining decent raw file sizes (e.g., on average 70MB for our light model on the mip-NeRF 360 dataset), we can obtain a 10× reduction for ours and 3DGS when adding compression[2], leading to file sizes below 40MB for our default model and below 24MB for our light model. In comparison, top performing baselines such as SMERF require more than 4GB of storage.

[2]See github.com/aras-p/UnityGaussianSplatting



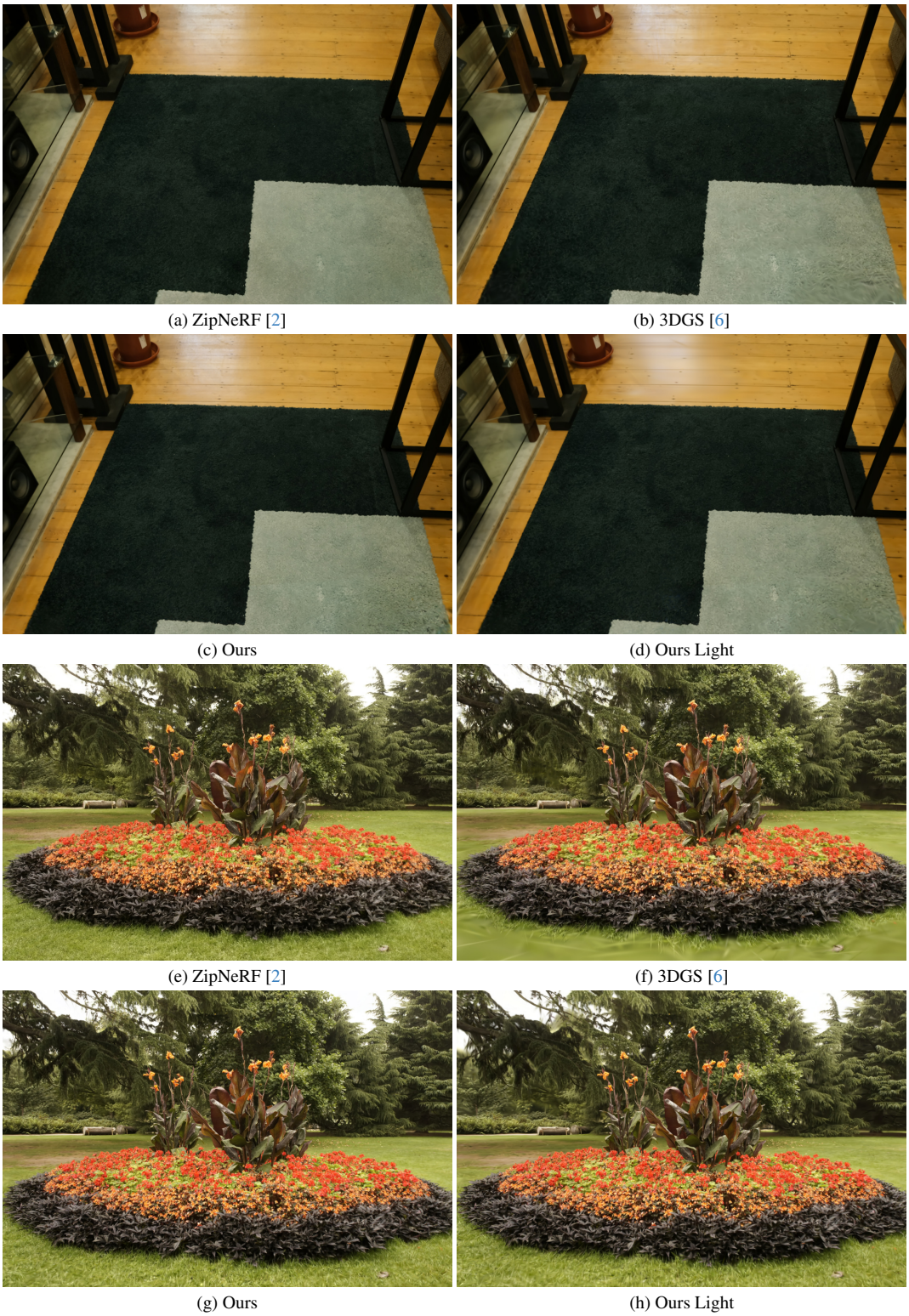
(a) ZipNeRF [2] (b) 3DGS [6]

(c) Ours (d) Ours Light

(e) ZipNeRF [2] (f) 3DGS [6]

(g) Ours (h) Ours Light

Figure 4. **Additional Qualitative Comparison on MipNeRF360.**



(a) ZipNeRF [2] (b) 3DGS [6]

(c) Ours (d) Ours Light

(e) ZipNeRF [2] (f) 3DGS [6]

(g) Ours (h) Ours Light

Figure 5. **Additional Qualitative Comparison on MipNeRF360.**

(a) ZipNeRF [2] (b) 3DGS [6]

(c) Ours (d) Ours Light

(e) ZipNeRF [2] (f) 3DGS [6]

(g) Ours (h) Ours Light

Figure 6. **Additional Qualitative Comparison on MipNeRF360.**

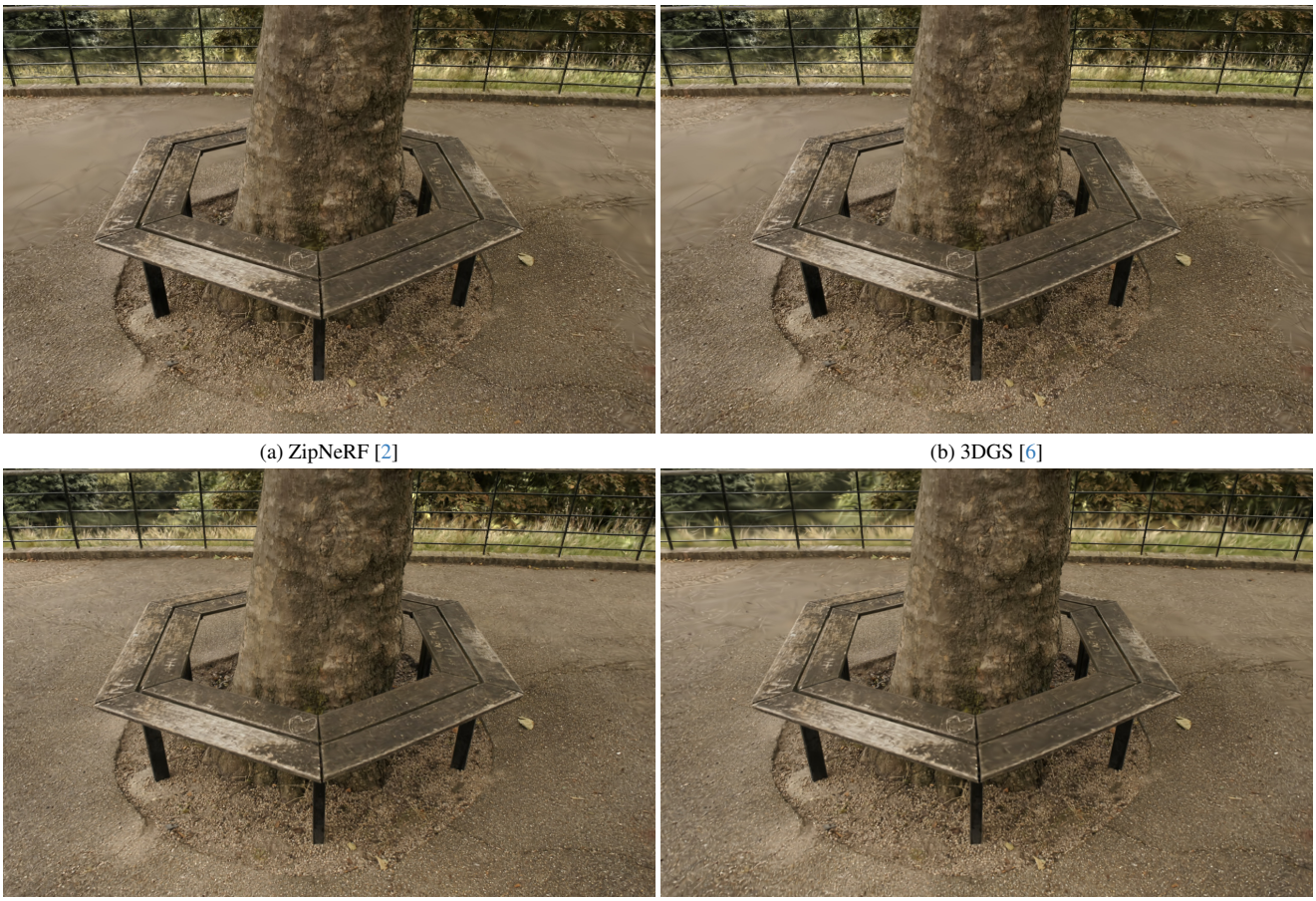
(a) ZipNeRF [2] (b) 3DGS [6]

(c) Ours (d) Ours Light

(e) ZipNeRF [2] (f) 3DGS [6]

(g) Ours (h) Ours Light

Figure 7. **Additional Qualitative Comparison on MipNeRF360.**



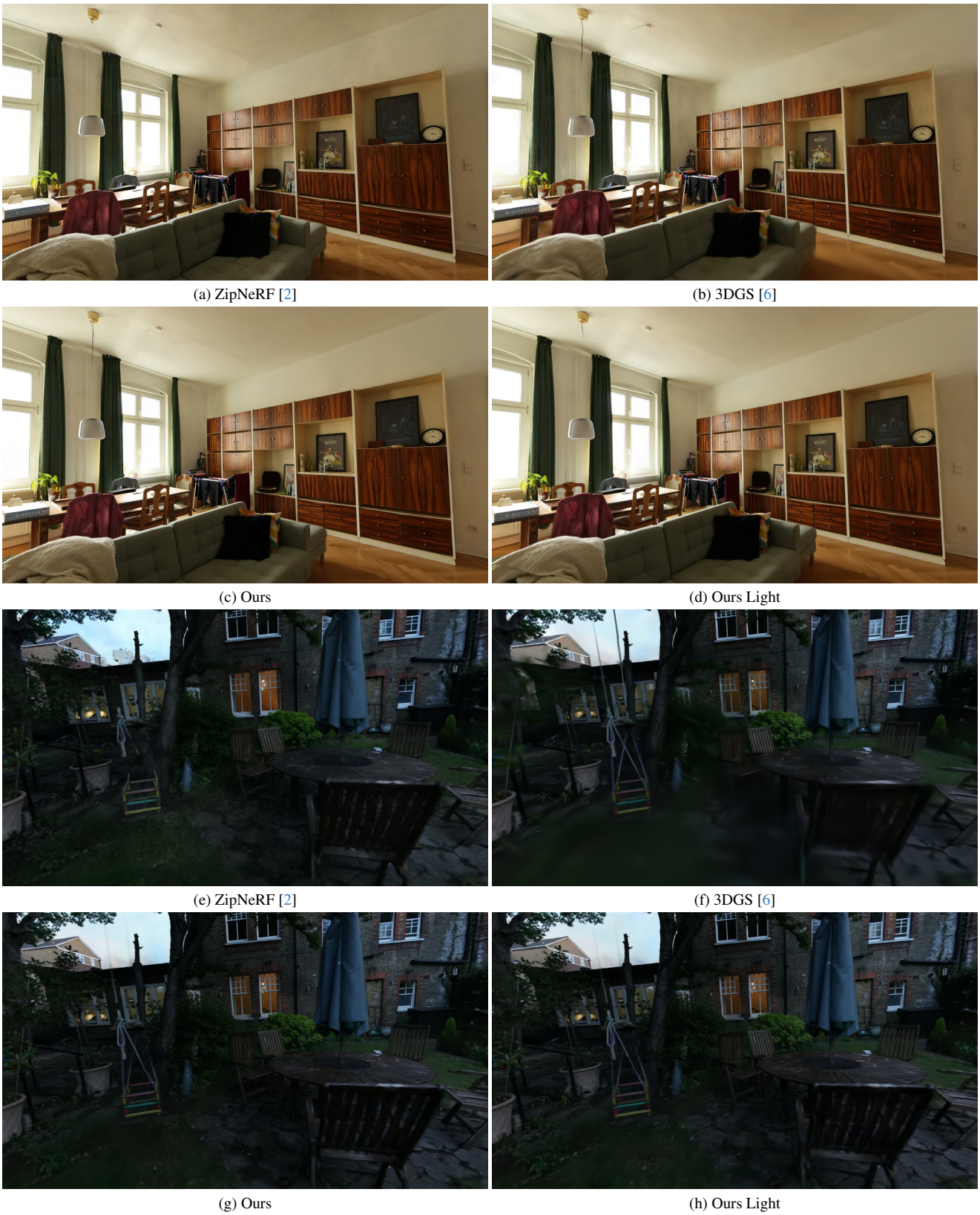
(a) ZipNeRF [2]

(b) 3DGS [6]

(c) Ours

(d) Ours Light

Figure 8. **Additional Qualitative Comparison on MipNeRF360.**



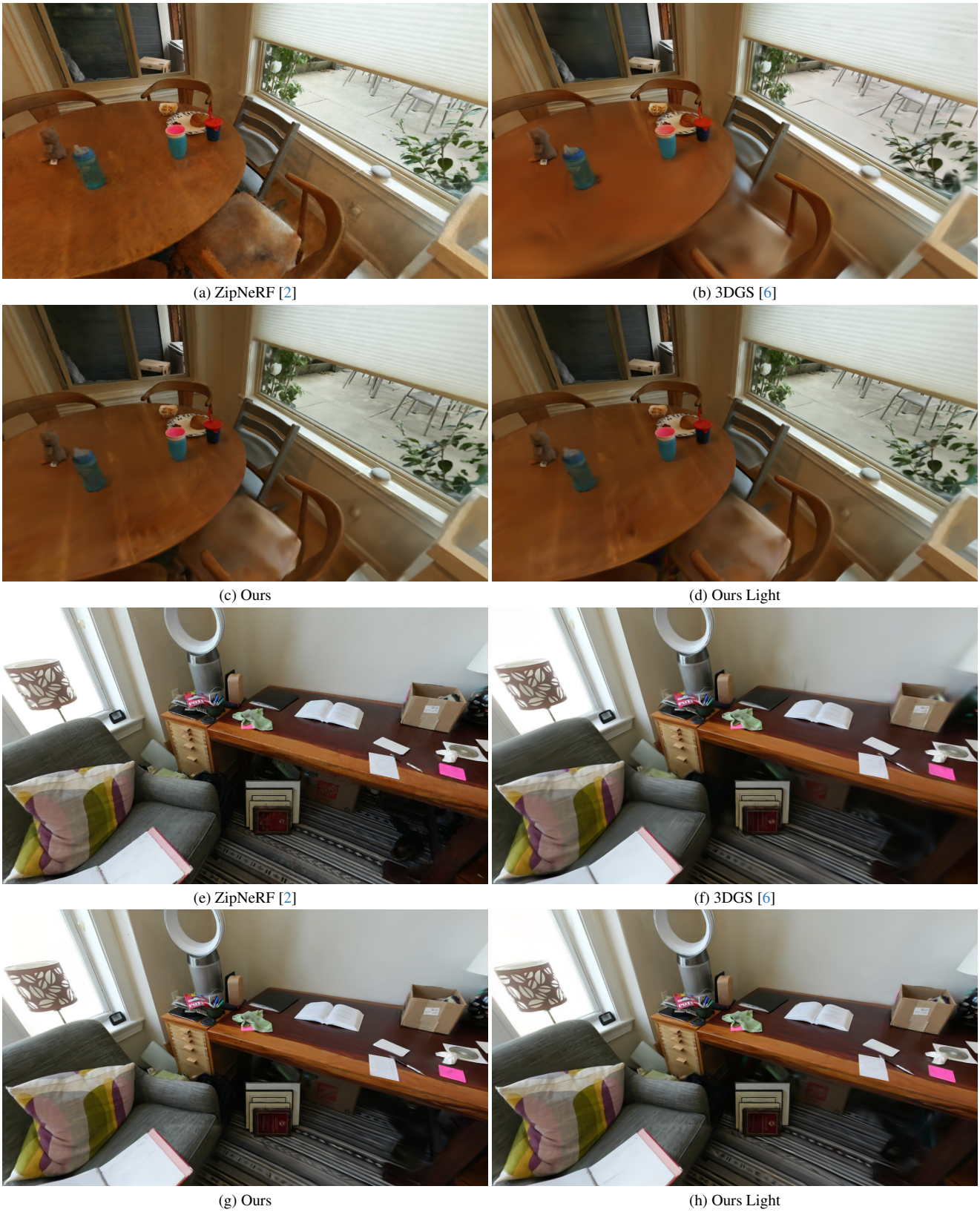
(a) ZipNeRF [2] (b) 3DGS [6]

(c) Ours (d) Ours Light

(e) ZipNeRF [2] (f) 3DGS [6]

(g) Ours (h) Ours Light

Figure 9. **Additional Qualitative Comparison on ZipNeRF.**



(a) ZipNeRF [2] (b) 3DGS [6]

(c) Ours (d) Ours Light

(e) ZipNeRF [2] (f) 3DGS [6]

(g) Ours (h) Ours Light

Figure 10. **Additional Qualitative Comparison on ZipNeRF.**

References

[1] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *CVPR*, 2022. 2

[2] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *ICCV*, 2023. 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13

[3] Daniel Duckworth, Peter Hedman, Christian Reiser, Peter Zhizhin, Jean-François Thibert, Mario Lucic, Richard Szeliski, and Jonathan T. Barron. SMERF: streamable memory efficient radiance fields for real-time large-scene exploration. *arXiv*, 2023. 1, 2, 3, 4, 5, 6

[4] Zhiwen Fan, Kevin Wang, Kairun Wen, Zehao Zhu, Dejia Xu, and Zhangyang Wang. Lightgaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ FPS. *arXiv*, 2023. 3, 4

[5] Sharath Girish, Kamal Gupta, and Abhinav Shrivastava. Eagles: Efficient accelerated 3d gaussians with lightweight encodings. *arXiv*, 2023. 3, 4

[6] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *SIGGRAPH*, 2023. 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13

[7] Joochan Lee, Daniel Rho, Xiangyu Sun, Jong Hwan Ko, and Eunbyung Park. Compact 3d gaussian representation for radiance field. *arXiv*, 2023. 3, 4

[8] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *SIGGRAPH*, 2022. 3, 4

[9] Christian Reiser, Richard Szeliski, Dor Verbin, Pratul P. Srinivasan, Ben Mildenhall, Andreas Geiger, Jonathan T. Barron, and Peter Hedman. MERF: memory-efficient radiance fields for real-time view synthesis in unbounded scenes. *SIGGRAPH*, 2023. 3, 4, 5, 6

[10] Lior Yariv, Peter Hedman, Christian Reiser, Dor Verbin, Pratul P. Srinivasan, Richard Szeliski, Jonathan T. Barron, and Ben Mildenhall. Bakedsdf: Meshing neural sdfs for real-time view synthesis. In *SIGGRAPH*, 2023. 3, 4