

Appendix

This appendix introduces additional results (Section A1) and ablations (Section A2). We then discuss our reconstructed scene structure (Section A3), possible failure cases (Section A4), and provide visualizations for the different types of initializations mentioned in the main paper (Section A5). Finally, we discuss the parallels between 3DGS and NeRF rendering, which allows training a radiance field using the rendering from 3DGS (Section A6).

A1. Detailed results

Table A1 and Table A2 present results for the no-budget scenario, which were used for plots in Figure 6. We use various values of the β parameter without specifying a budget for our method, except for the last cell, which uses the budget set by the number of Gaussians generated by 3D Gaussian Splatting. For each cell, we also compare MCMC [25] and EDGS [27] where they are set to match the number of Gaussians produced in each of the cells. The comparisons show that our method is able to produce high-quality results even without specifying a budget, instead adjusting the number of primitives based on the scene complexity, while still remaining sparse in the number of primitives. Notably, even when setting $\beta = 0$, which effectively disables densification, our method still performs well due to a dense initialization and effective filtering of unnecessary primitives enforced by the opacity penalty, consistent with

Ablation	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	FPS \uparrow	# Gaussians
Ours ($\beta = 0$)	29.12	0.873	0.183	185	542k
MCMC (SfM)	28.86	0.865	0.209	136	
EDGS	28.87	0.868	0.187	194	
Ours ($\beta = 0.01$)	29.25	0.877	0.174	169	674k
MCMC (SfM)	29.06	0.870	0.199	119	
EDGS	28.97	0.873	0.178	167	
Ours ($\beta = 0.02$)	29.34	0.880	0.166	146	942k
MCMC (SfM)	29.15	0.876	0.189	101	
EDGS	29.10	0.878	0.167	134	
Ours ($\beta = 0.04$)	29.38	0.881	0.161	105	1.66M
MCMC (SfM)	29.32	0.882	0.174	77	
EDGS	29.19	0.881	0.159	102	
Ours	29.35	0.879	0.159	80	2.57M
MCMC (SfM)	29.56	0.887	0.164	55	
EDGS	29.37	0.884	0.153	67	
3DGS	29.03	0.870	0.184	69	

Table A1. **No-budget Mip-NeRF360.** Comparison of reconstruction quality without a fixed limit on the number of primitives. For each cell, our method is run with a different β parameter, which determines the number of Gaussians generated, and the other methods are limited to this number. In the last cell, the number of Gaussians is set to match the amount produced by 3DGS, with all other methods constrained accordingly. Results are averaged over the Mip-NeRF360 [3] dataset.

Ablation	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	FPS \uparrow	# Gaussians
Ours ($\beta = 0$)	28.54	0.879	0.195	253	352k
MCMC (SfM)	28.37	0.877	0.207	199	
EDGS	28.62	0.885	0.183	219	
Ours ($\beta = 0.01$)	29.02	0.890	0.176	237	438k
MCMC (SfM)	28.55	0.882	0.199	175	
EDGS	28.84	0.890	0.173	211	
Ours ($\beta = 0.02$)	29.19	0.894	0.166	217	581k
MCMC (SfM)	28.79	0.888	0.190	157	
EDGS	29.07	0.896	0.163	188	
Ours ($\beta = 0.04$)	29.32	0.898	0.157	116	927k
MCMC (SfM)	29.05	0.895	0.177	105	
EDGS	29.35	0.902	0.151	134	
Ours	29.60	0.904	0.144	114	1.75M
MCMC	29.46	0.904	0.157	89	
EDGS	29.85	0.909	0.137	99	
3DGS	29.30	0.896	0.171	88	

Table A2. **No-budget OMMO.** Comparison of reconstruction quality without a fixed limit on the number of primitives. For each cell, our method is run with a different β parameter, which determines the number of Gaussians generated, and the other methods are limited to this number. In the last cell, the number of Gaussians is set to match the amount produced by 3DGS, with all other methods constrained accordingly. Results are averaged over the OMMO [33] dataset.

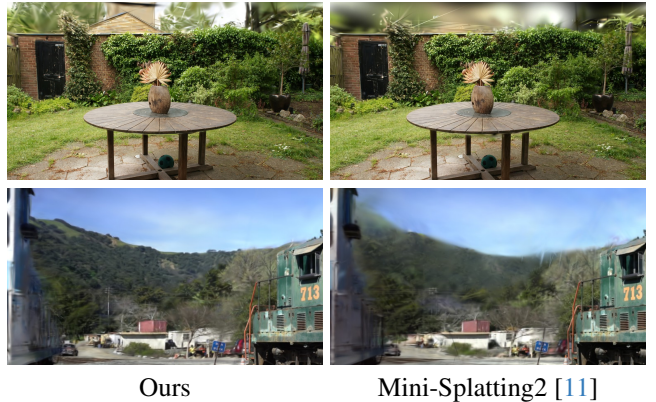


Figure A1. **Qualitative comparison** between our approach and Mini-Splatting2 [11] on the garden scene from Mip-NeRF360 [3] and on the train scene from Tanks & Temples [26].

findings in [27]. However, this configuration does not allow explicit control over the number of primitives and may limit the achievable reconstruction quality.

If an even lower number of Gaussians is desired while still using a no-budget scenario, the number of initialized Gaussians can be reduced, effectively lowering the number of primitives generated at the end.

We show in Table A3 the results on the selected benchmarks for the budget of 1M primitives. Additionally, we present per-scene results for 100k Gaussians in Table A8,

	Mip-NeRF360 [3]			OMMO [33]			Tanks & Temples [26]			DeepBlending [19]		
	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
Number of Gaussians limited to 1M												
3DGS [24] (SfM init.)	28.71	0.846	0.222	29.40	0.894	0.178	23.60	0.826	0.245	28.93	0.881	0.292
Foroutan et al. [14] [†]	29.22	0.876	0.177	29.32	0.896	0.163	23.75	0.829	0.230	29.60	0.886	0.282
MCMC [25] (rand. init.)	28.98	0.865	0.204	28.83	0.888	0.185	23.41	0.824	0.249	28.94	0.877	0.303
MCMC [25] (SfM init.)	29.23	0.875	0.190	29.18	0.895	0.174	23.93	0.836	0.233	29.67	0.887	0.288
MCMC [25] (iNGP init.)	29.32	0.879	0.173	28.76	0.887	0.181	23.37	0.804	0.268	29.87	0.892	0.276
GaussianPro [9]	28.56	0.845	0.226	28.79	0.885	0.189	23.01	0.818	0.256	29.27	0.887	0.291
Perceptual-GS [64]	29.27	0.876	0.176	29.00	0.888	0.179	23.26	0.828	0.240	29.41	0.889	0.286
EDGS [27]	29.18	0.877	0.168	29.58	0.903	0.149	23.66	0.842	0.200	29.43	0.889	0.271
Ours	29.37	0.880	0.168	29.44	0.899	0.154	23.70	0.835	0.207	29.69	0.889	0.273

[†] Original code was not publicly available. Our implementation uses iNGP initialization and does not include the additional depth-based loss.

Table A3. **Quantitative results** with a Gaussian number limit of 1M. We highlight the **best**, **second best** and **third best** results among methods with comparable numbers of Gaussians.

	Ours	3DGS (SfM)	MCMC (SfM)	EDGS	Perceptual-GS
# Memory (MiB)	9545	9049	8671	14517	12403

Table A4. **Peak GPU memory usage** on the 15 scene from the OMMO dataset [33] on the maximum budget of 500k primitives. We highlight the **best**, **second best** and **third best** results among all.

	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	FPS \uparrow	Train (min) \downarrow
ConeGS	29.14	0.892	0.170	217	25
ConeGS (iNGP retrain)	29.27	0.893	0.168	212	29

Table A5. **Ablation on additional iNGP training** on the OMMO [33] dataset with 500k primitives, evaluating the effect of continuing training iNGP in parallel with the full 3DGS optimization. We highlight the **best**, **second best** and **third best** results among all.

500k Gaussians in Table A9, 1M Gaussians in Table A10, and 2M Gaussians in Table A11. We also show additional qualitative results with Mini-Splatting2 [11] in Figure A1.

A2. Additional ablations and comparisons

We evaluate the peak GPU memory usage during optimization for several methods in Table A4. The results show that our method is very close to 3DGS [24] and MCMC [25] in terms of memory consumption, while remaining considerably lower than EDGS [27] and Perceptual-GS [64]. This efficiency allows our method to run on a wider range of GPUs, making it more accessible to devices with limited memory.

We expand on ablation (c) from the main paper, where iNGP continues training in parallel with the full 3DGS optimization. Since iNGP training, like densification, is guided by the L1 error from 3DGS, the iNGP model can better focus on regions where 3DGS reconstruction may fall short, potentially improving densification. The original ablation was tested on a budget of 100k primitives, which may not

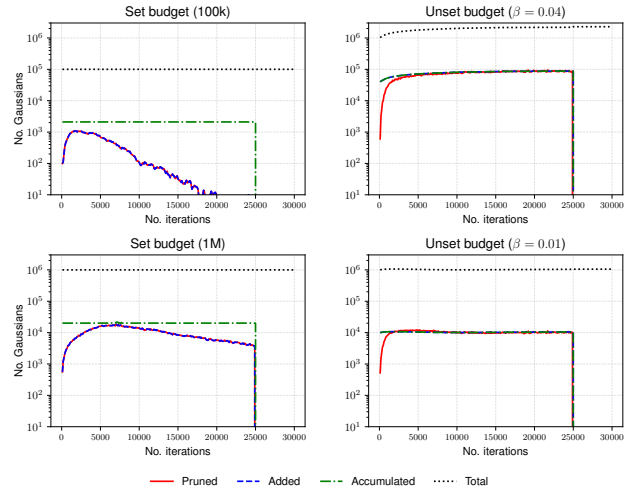


Figure A2. **Number of primitives** pruned, added, accumulated, as well as the total number, during each densification, on different budget scenarios. Results obtained from the garden scene from Mip-NeRF360 [3].

fully reveal this effect. To explore further, we run experiments with larger budgets. On the 500k budget for challenging OMMO [33] scenes (Table A5), we observe additional performance gains, though at the cost of longer training time. We also test this setup with a budget of 1M primitives across all datasets (Table A6), finding minimal improvements on difficult scenes and slight decreases on Mip-NeRF360, which may be caused by overfitting to certain areas.

Figure A2 illustrates the number of primitives added and removed during each densification and pruning step. It also tracks the accumulation buffer of primitives. When no budget is specified, all accumulated primitives are added to the scene (see Eq. 17). Under a fixed budget, however, not all of them are used. This is because accumulation happens every iteration and must remain available for pruning and

	Mip-NeRF360 [3]			OMMO [33]			Tanks & Temples [26]			DeepBlending [19]		
	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
Number of Gaussians limited to 1M												
Ours	29.37	0.880	0.168	29.44	0.899	0.154	23.70	0.835	0.207	29.69	0.889	0.273
Ours, iNGP training during 3DGS	29.25	0.879	0.168	29.54	0.900	0.152	23.72	0.836	0.207	29.73	0.889	0.273

Table A6. **Ablation on additional iNGP training** for the budget of 1M primitives. We highlight the **best**, **second best** and **third best** results among methods with comparable numbers of Gaussians.

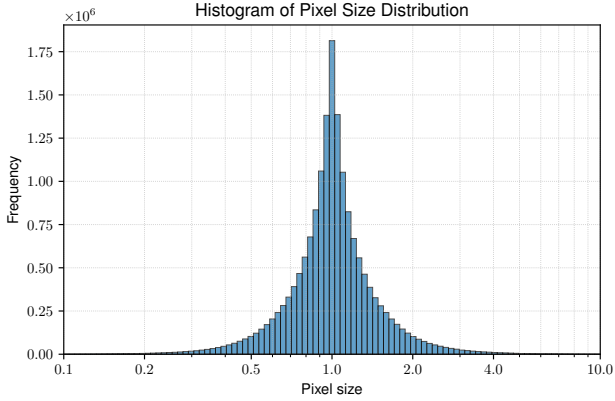


Figure A3. **The histogram of perceived image-space size** of pixel-sized Gaussians rendered from different viewpoints. Size expressed in pixel widths. Analysis doesn't include the low-pass filter from 3DGS rasterization.

	Ours	MCMC (SfM)	MCMC	3DGS (SfM)
# Gaussians per pixel	30.72	49.55	48.45	34.74

Table A7. **Blending analysis.** Mean number of Gaussians contributing to alpha blending per pixel, averaged across the Mip-NeRF360 [3] dataset using a budget of 1M Gaussians. We highlight the **best**, **second best** and **third best** results among all.

densification, while the exact number that can be added is unknown beforehand. As a result, the system accumulates more primitives than are usually required to keep the total count close to the budget after pruning (see Eq. 16).

The primitive size is defined to be approximately one pixel from a single viewpoint. From other viewpoints, this size is not exactly one pixel, but should remain close. To confirm this, we measure the size of newly added pixel-sized primitives from multiple viewpoints and report the results in Figure A3. The distribution shows that the apparent size remains near one pixel, with very few cases above four pixels or below 0.2 pixels.

A3. Scene structure

To confirm that our method produces reconstructions with desirable characteristics, such as a balanced distribution of scaling values and placement close to surfaces, which

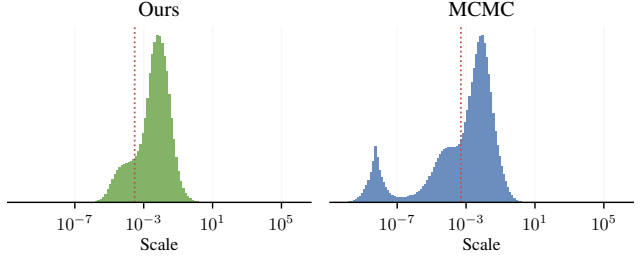


Figure A4. **Histograms of the Gaussian scaling values** for our method and MCMC with SfM initialization on 2M Gaussians. The red lines indicate the minimum scaling required for a Gaussian to cover at least one pixel, disregarding the low-pass filter.



Figure A5. **Shrunk Gaussians.** Visual comparison between the proposed method and MCMC, rendered using Gaussians scaled to half their original size. Both methods were trained with a limit of 1 million primitives on the bicycle and bonsai scenes from Mip-NeRF360 [3], and the 10 scene from OMMO [33].

are often important for downstream tasks, we analyze the scenes created with our method in comparison to MCMC [25].

Although Gaussian Splatting with the MCMC densifica-

tion strategy explores the scene effectively, it also has clear shortcomings. Its cloning strategy and scaling penalty often cause Gaussians to shrink strongly along certain dimensions. The low-pass filter used by 3DGS renderer can hide this effect visually, but it prevents Gaussians from expanding in those directions and interferes with tasks that depend on accurate scales, such as MCMC position error. As shown in Figure A4, many of its Gaussians shrink below the pixel radius, in contrast to the more balanced distribution produced by our method.

Figure A5 shows that our approach produces Gaussians that are more uniform in size and placed closer to object surfaces. This avoids an overreliance on oversized background primitives located far from the surface. This primitive distribution not only improves geometric alignment but also increases rendering speed by reducing blending and sorting overhead during rasterization. To support this hypothesis, Table A7 reports the mean number of Gaussians blended per pixel. Our method consistently requires less blending compared to the selected benchmarks. The difference is especially large compared to MCMC, which blends over 60% more Gaussians per pixel on average.

A4. Failure cases

While our method generally produces strong reconstructions on almost all tested scenes, its reliance on iNGP can make it susceptible to floaters in challenging scenarios such as noisy poses, very large or sparse-view scenes, or other cases where reliable iNGP reconstruction is difficult. One such example is shown in Figure A6, where the scene contains distortions and lacks sufficient viewpoint coverage near the cameras. This leads to spurious high-density regions in iNGP close to the cameras, which in turn degrades densification quality by placing Gaussians in incorrect regions. Although this can reduce performance, the method still produces high-quality reconstructions overall (see per-scene results).

A5. Initialization visualizations

The choice of initialization has a strong impact on the performance of 3DGS reconstruction. A sufficiently good initialization can even remove the need for additional densification ([27], Table A1, Table A2). We visualize different initialization types in Figure A7. Initialization from a sparse SfM point cloud often leaves large gaps that are difficult to fill correctly, while our initialization produces a more uniform coverage. Another important strategy is initialization based on pixel width. This approach does not provide a useful inductive bias of larger primitives and can lead to more gaps in unobserved regions, although it may offer faster rendering due to reduced blending (Section 5.2). Finally, our formulation allows scaling the initial primitives in image

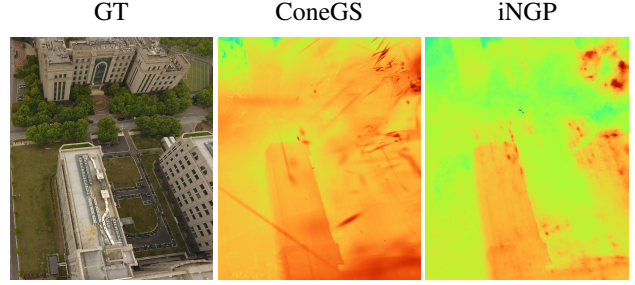


Figure A6. **Depth maps.** Depth maps for ConeGS and the iNGP reconstruction model on the 01 scene from the OMMO [33] dataset. Since the Gaussians are generated from an iNGP reconstruction, the presence of floaters in it leads to floaters also appearing in the Gaussian Splatting results.

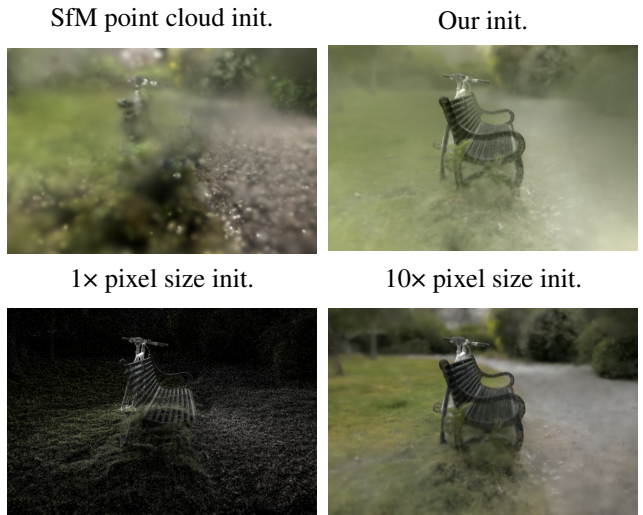


Figure A7. **Initialization Methods Comparison.** Visual comparison of four initialization methods for 3DGS reconstruction on the bicycle scene from Mip-NeRF 360 dataset [3]: (1) SfM point cloud initialization, (2) iNGP initialization with kNN-based sizing, (3) iNGP initialization with 1x sizing, and (4) iNGP initialization using the smaller of 10x pixel size or kNN-based sizing.

space, producing a balance between these two types of initialization.

A6. Gaussian-Based Radiance Field training

To establish a more direct connection between the volumetric ray marching procedure in Neural Radiance Fields (NeRF) and the rendering in 3D Gaussian Splatting (3DGS), the set of conical frustums sampled along a cone in Mip-NeRF and its derivatives [2–4] can be reinterpreted as a set of 3D Gaussian primitives, covering approximately the same area. The purpose of this reformulation is to enable training a neural radiance field model using only the 3DGS renderer, without using the traditional volumetric ray integration.

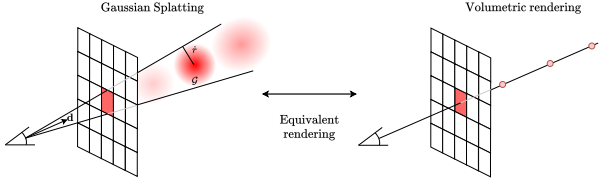


Figure A8. **Training equivalence.** Illustration of the equivalence between training an implicit radiance field model using NeRF-style ray marching and 3D Gaussian Splatting rasterization.

To align the 3DGS and NeRF rendering formulations, it is necessary to demonstrate that an entire ray can be equivalently represented as a sequence of Gaussians such that, when rendered, the result is equal to the volumetric integration process. This is achieved by casting cones along pixel directions and subdividing each into a set of conical frustums. Each frustum is then mapped to a Gaussian primitive, where the midpoint

$$t_{\mu,i} = \frac{t_i + t_{i+1}}{2} \quad (\text{A1})$$

is used to define the Gaussian center \mathbf{p}_i . The view-dependent RGB color \mathbf{c}_i and scalar density σ_i , which is converted to opacity o_i using Eq. 6, are predicted by a neural network. As detailed in Section 4.1, the predicted RGB color \mathbf{c}_i is encoded as spherical harmonics coefficients, which are compatible with the 3DGS rasterizer.

Since the Gaussians lie along the cone and their centers are co-linear with the pixel center and the camera origin, their projected 2D means fall directly on the target pixel. Consequently, the maximum opacity contribution aligns with the pixel center, and the kernel evaluation satisfies

$$K(\mathbf{p}_c, \boldsymbol{\mu}_i^{2D}, \boldsymbol{\Sigma}_i^{2D}) = 1. \quad (\text{A2})$$

This condition ensures that the Gaussian opacity o_i corresponds directly to the opacity α_i used in blending operations, as defined in Eq. 6 and Eq. 2. The opacity is therefore determined by the predicted density and the length of the corresponding frustum.

To maintain consistent 2D projection footprints across different depths, each Gaussian’s 3D covariance must be adjusted based on its location along the cone. Because elongating Gaussians along the ray direction does not affect the resulting 2D projection, the scaling can be derived using the pixel footprint size and set isotropically using the formulation in Eq. 15, while the quaternion can be set to the identity quaternion.

Given these assignments for position, color, opacity, scale, and orientation, each Gaussian can be rendered using the 3DGS renderer to produce the same pixel color as that produced by volumetric rendering. Assuming no low-pass filtering is applied and that only a single image is ren-

dered at a time, it becomes possible to train an Instant-NGP-style model using only the 3DGS rendering pipeline. Although this approach is marginally slower than traditional ray marching, it yields equivalent radiance field representations. A visual comparison of both approaches is shown in Figure A8.

This reinterpretation also provides a more principled foundation for the proposed densification strategy. Instead of using depth, the strategy can be understood as selecting a Gaussian that corresponds to a surface-level conical frustum of pixels with high photometric error. This establishes a clearer theoretical link between cone-based densification and neural implicit models such as NeRF.

Scene		Ours	3DGS (SfM)	MCMC (SfM)	EDGS
Mip-NeRF360 [3]	Bicycle	24.05 / 0.652 / 0.379 / 368	20.99 / 0.470 / 0.543 / 466	23.34 / 0.630 / 0.400 / 339	23.68 / 0.635 / 0.393 / 444
	Garden	24.69 / 0.709 / 0.336 / 431	22.87 / 0.589 / 0.445 / 430	24.51 / 0.710 / 0.350 / 569	24.47 / 0.706 / 0.341 / 493
	Stump	25.71 / 0.707 / 0.351 / 456	22.79 / 0.547 / 0.498 / 506	25.16 / 0.680 / 0.370 / 402	25.08 / 0.673 / 0.377 / 588
	Room	31.09 / 0.903 / 0.254 / 275	27.02 / 0.859 / 0.328 / 267	30.37 / 0.900 / 0.270 / 224	30.08 / 0.896 / 0.265 / 317
	Counter	28.22 / 0.879 / 0.251 / 229	25.64 / 0.845 / 0.307 / 252	27.83 / 0.880 / 0.260 / 198	27.77 / 0.877 / 0.251 / 245
	Kitchen	29.73 / 0.896 / 0.183 / 239	20.53 / 0.671 / 0.440 / 297	28.37 / 0.890 / 0.210 / 246	28.84 / 0.891 / 0.186 / 269
	Bonsai	30.67 / 0.920 / 0.241 / 276	25.41 / 0.866 / 0.331 / 326	29.84 / 0.910 / 0.260 / 281	29.73 / 0.906 / 0.256 / 298
	Average	27.74 / 0.809 / 0.285 / 325	23.61 / 0.692 / 0.413 / 363	27.06 / 0.800 / 0.303 / 323	27.09 / 0.798 / 0.296 / 379
OMMO [33]	01	22.58 / 0.625 / 0.458 / 212	20.53 / 0.543 / 0.575 / 159	22.66 / 0.620 / 0.470 / 180	22.50 / 0.608 / 0.475 / 222
	03	25.39 / 0.842 / 0.244 / 356	24.09 / 0.803 / 0.295 / 112	24.30 / 0.810 / 0.290 / 217	24.85 / 0.827 / 0.259 / 313
	05	27.95 / 0.863 / 0.246 / 415	26.87 / 0.841 / 0.296 / 354	27.70 / 0.860 / 0.270 / 318	27.69 / 0.856 / 0.259 / 398
	06	27.30 / 0.913 / 0.202 / 321	26.53 / 0.899 / 0.229 / 107	26.25 / 0.910 / 0.210 / 339	26.60 / 0.910 / 0.198 / 339
	10	29.31 / 0.853 / 0.252 / 338	28.86 / 0.833 / 0.291 / 125	28.70 / 0.840 / 0.280 / 337	28.78 / 0.839 / 0.275 / 341
	13	30.36 / 0.899 / 0.211 / 449	29.79 / 0.889 / 0.247 / 171	29.45 / 0.880 / 0.250 / 378	28.98 / 0.878 / 0.234 / 468
	14	29.73 / 0.924 / 0.145 / 394	27.09 / 0.874 / 0.220 / 237	29.26 / 0.920 / 0.160 / 360	28.92 / 0.909 / 0.169 / 374
	15	28.10 / 0.893 / 0.182 / 405	27.80 / 0.877 / 0.213 / 153	27.72 / 0.890 / 0.200 / 427	27.58 / 0.877 / 0.215 / 420
	Average	27.59 / 0.852 / 0.242 / 361	26.45 / 0.820 / 0.296 / 177	27.00 / 0.841 / 0.266 / 320	26.99 / 0.838 / 0.261 / 359
Tanks & Temples [26]	Truck	24.54 / 0.822 / 0.291 / 184	23.59 / 0.803 / 0.318 / 128	23.86 / 0.810 / 0.316 / 155	23.33 / 0.799 / 0.314 / 183
	Train	21.70 / 0.759 / 0.328 / 204	21.17 / 0.744 / 0.349 / 118	21.13 / 0.749 / 0.347 / 160	21.31 / 0.756 / 0.334 / 180
	Average	23.12 / 0.790 / 0.309 / 194	22.38 / 0.774 / 0.334 / 123	22.49 / 0.780 / 0.332 / 158	22.32 / 0.778 / 0.324 / 182
Deep Blending [19]	Dr Johnson	28.78 / 0.875 / 0.338 / 447	23.58 / 0.808 / 0.440 / 521	28.16 / 0.869 / 0.343 / 550	27.66 / 0.865 / 0.347 / 444
	Playroom	30.09 / 0.885 / 0.318 / 489	25.72 / 0.845 / 0.384 / 489	29.72 / 0.883 / 0.323 / 552	29.19 / 0.879 / 0.327 / 486
	Average	29.44 / 0.880 / 0.328 / 468	24.65 / 0.827 / 0.412 / 505	28.94 / 0.876 / 0.333 / 551	28.43 / 0.872 / 0.337 / 465

Table A8. **Detailed results** on a selection of datasets and methods with the number of Gaussians limited to 100k. Each field contains PSNR, SSIM, LPIPS and FPS respectively. We highlight the **best**, **second best** and **third best** results among all. Slight discrepancies from the main table are due to rounding.

Scene		Ours	3DGS (SfM)	MCMC (SfM)	EDGS
Mip-NeRF360 [3]	Bicycle	25.25 / 0.758 / 0.242 / 234	24.06 / 0.651 / 0.370 / 174	24.90 / 0.740 / 0.282 / 162	24.95 / 0.751 / 0.251 / 248
	Garden	26.87 / 0.837 / 0.157 / 246	25.16 / 0.743 / 0.296 / 297	26.36 / 0.823 / 0.189 / 221	26.49 / 0.833 / 0.161 / 252
	Stump	27.03 / 0.793 / 0.215 / 255	25.27 / 0.688 / 0.348 / 226	26.81 / 0.779 / 0.250 / 175	26.50 / 0.768 / 0.237 / 296
	Room	32.02 / 0.924 / 0.205 / 192	31.47 / 0.913 / 0.234 / 112	31.72 / 0.921 / 0.221 / 114	31.36 / 0.923 / 0.203 / 168
	Counter	28.87 / 0.906 / 0.194 / 148	28.86 / 0.901 / 0.213 / 113	28.95 / 0.907 / 0.205 / 93	29.05 / 0.912 / 0.184 / 137
	Kitchen	31.38 / 0.929 / 0.124 / 172	30.76 / 0.918 / 0.143 / 111	31.04 / 0.921 / 0.141 / 111	31.35 / 0.928 / 0.122 / 154
	Bonsai	32.16 / 0.944 / 0.191 / 190	31.95 / 0.936 / 0.218 / 135	31.97 / 0.939 / 0.210 / 128	32.02 / 0.942 / 0.191 / 170
	Average	29.08 / 0.870 / 0.190 / 205	28.22 / 0.821 / 0.260 / 167	28.82 / 0.861 / 0.214 / 143	28.82 / 0.865 / 0.193 / 204
OMMO [33]	01	23.46 / 0.690 / 0.356 / 140	23.81 / 0.682 / 0.385 / 102	23.73 / 0.685 / 0.378 / 90	23.63 / 0.686 / 0.363 / 84
	03	27.01 / 0.887 / 0.181 / 197	26.18 / 0.868 / 0.220 / 111	26.50 / 0.875 / 0.212 / 105	27.16 / 0.892 / 0.176 / 93
	05	28.61 / 0.877 / 0.199 / 232	28.53 / 0.874 / 0.235 / 162	28.71 / 0.877 / 0.233 / 154	28.79 / 0.881 / 0.196 / 122
	06	27.79 / 0.932 / 0.159 / 195	27.59 / 0.931 / 0.161 / 120	26.99 / 0.934 / 0.159 / 152	27.38 / 0.941 / 0.135 / 105
	10	31.16 / 0.905 / 0.165 / 235	31.04 / 0.894 / 0.194 / 143	30.56 / 0.892 / 0.192 / 168	31.11 / 0.906 / 0.165 / 129
	13	33.02 / 0.949 / 0.115 / 280	32.40 / 0.939 / 0.149 / 165	32.18 / 0.934 / 0.155 / 175	31.95 / 0.944 / 0.123 / 151
	14	31.57 / 0.950 / 0.096 / 225	31.57 / 0.946 / 0.107 / 135	31.14 / 0.946 / 0.108 / 130	31.51 / 0.950 / 0.095 / 112
	15	30.50 / 0.944 / 0.090 / 231	30.59 / 0.934 / 0.114 / 149	29.92 / 0.933 / 0.113 / 162	30.52 / 0.942 / 0.095 / 125
	Average	29.14 / 0.892 / 0.170 / 217	28.96 / 0.884 / 0.196 / 136	28.72 / 0.885 / 0.194 / 142	29.01 / 0.893 / 0.169 / 115
Tanks & Temples [26]	Truck	25.41 / 0.858 / 0.204 / 122	24.80 / 0.842 / 0.253 / 93	25.24 / 0.851 / 0.242 / 68	24.66 / 0.850 / 0.212 / 114
	Train	21.97 / 0.799 / 0.253 / 140	22.28 / 0.790 / 0.277 / 72	22.12 / 0.798 / 0.275 / 86	22.27 / 0.812 / 0.246 / 104
	Average	23.69 / 0.829 / 0.229 / 131	23.54 / 0.816 / 0.265 / 82	23.68 / 0.825 / 0.259 / 77	23.47 / 0.831 / 0.229 / 109
Deep Blending [19]	Dr Johnson	29.20 / 0.890 / 0.291 / 270	28.85 / 0.881 / 0.307 / 151	28.71 / 0.884 / 0.313 / 180	28.60 / 0.888 / 0.292 / 269
	Playroom	30.51 / 0.892 / 0.279 / 287	29.66 / 0.883 / 0.298 / 161	30.16 / 0.889 / 0.303 / 209	30.06 / 0.890 / 0.281 / 277
	Average	29.86 / 0.891 / 0.285 / 278	29.26 / 0.882 / 0.303 / 156	29.44 / 0.887 / 0.308 / 194	29.33 / 0.889 / 0.287 / 273

Table A9. **Detailed results** on a selection of datasets and methods with the number of Gaussians limited to 500k. Each field contains PSNR, SSIM, LPIPS and FPS respectively. We highlight the **best**, **second best** and **third best** results among all. Slight discrepancies from the main table are due to rounding.

Scene		Ours	3DGS (SfM)	MCMC (SfM)	EDGS
Mip-NeRF360 [3]	Bicycle	25.45 / 0.778 / 0.198 / 156	24.40 / 0.688 / 0.325 / 129	25.34 / 0.768 / 0.238 / 106	25.26 / 0.778 / 0.199 / 161
	Garden	27.42 / 0.861 / 0.115 / 162	26.70 / 0.827 / 0.172 / 142	26.83 / 0.847 / 0.147 / 135	27.05 / 0.857 / 0.118 / 158
	Stump	27.23 / 0.802 / 0.184 / 164	25.64 / 0.719 / 0.301 / 162	27.21 / 0.798 / 0.213 / 110	26.71 / 0.783 / 0.203 / 179
	Room	32.30 / 0.928 / 0.196 / 139	31.62 / 0.918 / 0.221 / 82	32.09 / 0.926 / 0.208 / 79	31.71 / 0.928 / 0.191 / 111
	Counter	29.21 / 0.911 / 0.181 / 107	29.11 / 0.907 / 0.201 / 75	29.24 / 0.913 / 0.191 / 61	29.27 / 0.917 / 0.171 / 94
	Kitchen	31.66 / 0.932 / 0.117 / 118	31.18 / 0.924 / 0.131 / 78	31.44 / 0.927 / 0.130 / 72	31.83 / 0.933 / 0.114 / 101
	Bonsai	32.33 / 0.945 / 0.183 / 132	32.34 / 0.941 / 0.205 / 95	32.46 / 0.944 / 0.199 / 81	32.44 / 0.947 / 0.181 / 115
	Average	29.37 / 0.880 / 0.168 / 140	28.71 / 0.846 / 0.222 / 109	29.23 / 0.875 / 0.189 / 92	29.18 / 0.878 / 0.168 / 131
OMMO [33]	01	23.73 / 0.711 / 0.315 / 104	24.18 / 0.703 / 0.350 / 73	24.18 / 0.709 / 0.341 / 63	24.02 / 0.709 / 0.321 / 84
	03	27.39 / 0.896 / 0.167 / 127	26.96 / 0.886 / 0.197 / 64	27.20 / 0.890 / 0.189 / 67	27.72 / 0.905 / 0.157 / 85
	05	28.56 / 0.877 / 0.185 / 157	28.72 / 0.877 / 0.227 / 115	28.98 / 0.883 / 0.217 / 98	29.15 / 0.885 / 0.181 / 101
	06	27.87 / 0.936 / 0.150 / 136	27.68 / 0.933 / 0.158 / 110	27.46 / 0.939 / 0.146 / 103	27.75 / 0.945 / 0.127 / 102
	10	31.60 / 0.914 / 0.147 / 170	31.47 / 0.906 / 0.171 / 104	30.89 / 0.904 / 0.169 / 120	31.82 / 0.920 / 0.138 / 116
	13	33.55 / 0.957 / 0.098 / 190	33.13 / 0.949 / 0.129 / 116	32.83 / 0.945 / 0.131 / 124	32.83 / 0.955 / 0.100 / 125
	14	31.80 / 0.953 / 0.090 / 146	31.93 / 0.951 / 0.097 / 89	31.44 / 0.950 / 0.099 / 96	32.08 / 0.955 / 0.086 / 92
	Average	29.44 / 0.899 / 0.154 / 148	29.40 / 0.894 / 0.178 / 96	29.18 / 0.895 / 0.174 / 97	29.58 / 0.903 / 0.149 / 101
Tanks & Temples [26]	Truck	25.34 / 0.864 / 0.181 / 97	25.06 / 0.851 / 0.234 / 70	25.57 / 0.862 / 0.217 / 45	25.02 / 0.862 / 0.181 / 82
	Train	22.05 / 0.805 / 0.232 / 109	22.13 / 0.801 / 0.255 / 51	22.29 / 0.811 / 0.249 / 47	22.29 / 0.823 / 0.220 / 78
	Average	23.70 / 0.835 / 0.207 / 103	23.60 / 0.826 / 0.245 / 60	23.93 / 0.837 / 0.233 / 46	23.66 / 0.843 / 0.201 / 80
Deep Blending [19]	Dr Johnson	28.98 / 0.886 / 0.282 / 194	28.85 / 0.884 / 0.293 / 106	29.06 / 0.884 / 0.291 / 164	28.75 / 0.890 / 0.277 / 180
	Playroom	30.39 / 0.891 / 0.264 / 200	29.02 / 0.877 / 0.290 / 112	30.28 / 0.890 / 0.284 / 173	30.11 / 0.889 / 0.265 / 182
	Average	29.69 / 0.889 / 0.273 / 197	28.94 / 0.881 / 0.292 / 109	29.67 / 0.887 / 0.288 / 168	29.43 / 0.890 / 0.271 / 181

Table A10. **Detailed results** on a selection of datasets and methods with the number of Gaussians limited to 1M. Each field contains PSNR, SSIM, LPIPS and FPS respectively. We highlight the **best**, **second best** and **third best** results among all. Slight discrepancies from the main table are due to rounding.

Scene		Ours	3DGS (SfM)	MCMC (SfM)	EDGS
Mip-NeRF360 [3]	Bicycle	25.43 / 0.781 / 0.175 / 99	24.85 / 0.729 / 0.267 / 79	25.56 / 0.786 / 0.205 / 71	25.48 / 0.792 / 0.168 / 93
	Garden	27.59 / 0.870 / 0.097 / 97	27.22 / 0.852 / 0.129 / 84	27.33 / 0.862 / 0.122 / 82	27.46 / 0.870 / 0.097 / 89
	Stump	27.03 / 0.796 / 0.177 / 100	26.18 / 0.749 / 0.255 / 100	27.39 / 0.808 / 0.189 / 69	26.79 / 0.788 / 0.186 / 100
	Room	32.32 / 0.929 / 0.189 / 92	31.80 / 0.919 / 0.218 / 63	32.14 / 0.929 / 0.199 / 53	31.89 / 0.930 / 0.184 / 81
	Counter	29.23 / 0.912 / 0.175 / 70	29.07 / 0.907 / 0.201 / 68	29.41 / 0.917 / 0.181 / 42	29.37 / 0.918 / 0.164 / 65
	Kitchen	31.80 / 0.934 / 0.114 / 72	31.60 / 0.927 / 0.126 / 54	32.00 / 0.931 / 0.122 / 47	32.06 / 0.935 / 0.111 / 65
	Bonsai	32.63 / 0.948 / 0.177 / 84	32.34 / 0.941 / 0.204 / 84	32.80 / 0.948 / 0.189 / 53	32.62 / 0.947 / 0.175 / 85
	Average	29.43 / 0.881 / 0.158 / 88	29.01 / 0.861 / 0.200 / 76	29.52 / 0.883 / 0.172 / 60	29.38 / 0.883 / 0.155 / 83
OMMO [33]	01	23.92 / 0.725 / 0.283 / 73	24.51 / 0.721 / 0.321 / 49	24.51 / 0.731 / 0.303 / 45	24.24 / 0.725 / 0.285 / 64
	03	27.73 / 0.902 / 0.155 / 70	27.05 / 0.888 / 0.193 / 47	27.73 / 0.900 / 0.171 / 45	27.77 / 0.909 / 0.149 / 82
	05	28.65 / 0.877 / 0.176 / 91	28.69 / 0.877 / 0.227 / 112	29.20 / 0.887 / 0.201 / 67	29.31 / 0.887 / 0.169 / 90
	06	27.93 / 0.937 / 0.143 / 79	27.67 / 0.933 / 0.157 / 109	27.58 / 0.942 / 0.138 / 63	27.93 / 0.947 / 0.123 / 97
	10	31.81 / 0.920 / 0.135 / 104	31.77 / 0.915 / 0.153 / 70	31.59 / 0.913 / 0.151 / 80	32.26 / 0.928 / 0.123 / 89
	13	33.88 / 0.961 / 0.088 / 112	33.79 / 0.957 / 0.112 / 73	33.30 / 0.953 / 0.111 / 80	33.55 / 0.961 / 0.087 / 90
	14	32.02 / 0.955 / 0.086 / 83	31.95 / 0.951 / 0.096 / 82	31.75 / 0.953 / 0.093 / 62	31.76 / 0.956 / 0.084 / 86
	Average	31.25 / 0.953 / 0.075 / 86	31.35 / 0.947 / 0.090 / 83	30.80 / 0.947 / 0.086 / 67	31.63 / 0.954 / 0.072 / 87
Tanks & Temples [26]	Truck	25.43 / 0.865 / 0.167 / 69	25.39 / 0.859 / 0.217 / 53	25.86 / 0.869 / 0.193 / 38	25.09 / 0.865 / 0.162 / 55
	Train	21.95 / 0.809 / 0.217 / 77	22.29 / 0.802 / 0.253 / 55	22.35 / 0.822 / 0.228 / 40	21.84 / 0.827 / 0.204 / 58
	Average	23.69 / 0.837 / 0.192 / 73	23.84 / 0.831 / 0.235 / 54	24.11 / 0.846 / 0.211 / 39	23.47 / 0.846 / 0.183 / 56
Deep Blending [19]	Dr Johnson	29.01 / 0.884 / 0.274 / 127	28.75 / 0.886 / 0.282 / 77	28.83 / 0.882 / 0.285 / 104	28.65 / 0.887 / 0.268 / 113
	Playroom	30.50 / 0.887 / 0.247 / 127	29.32 / 0.880 / 0.279 / 83	30.22 / 0.890 / 0.272 / 105	30.11 / 0.886 / 0.248 / 112
	Average	29.76 / 0.886 / 0.261 / 127	29.04 / 0.883 / 0.281 / 80	29.53 / 0.886 / 0.279 / 104	29.38 / 0.887 / 0.258 / 112

Table A11. **Detailed results** on a selection of datasets and methods with the number of Gaussians limited to 2M. Each field contains PSNR, SSIM, LPIPS and FPS respectively. We highlight the **best**, **second best** and **third best** results among all. Slight discrepancies from the main table are due to rounding.