Supplementary for GStex: Per-Primitive Texturing of 2D Gaussian Splatting for Decoupled Appearance and Geometry Modeling

1. Video Results

Video results can be viewed on our project page website: https://lessvrong.com/cs/gstex. We include novel view synthesis renderings for all key figures of our paper. The videos are best viewed on the Chrome browser.

2. Implementation Details

Texture implementation. Let $T_j = \sum_{i=1}^{j} U_i V_i$ be the number of texels of the first j Gaussians. We represent our texture as a $T_n \times 3$ tensor, where the i Gaussian's texture map can be found flattened from indices T_{i-1} to $T_i - 1$. During rendering via the CUDA kernel, along with the typical Gaussian parameters passed in 2DGS, the texture along with arrays T_i, U_i, and V_i are passed in as well. When a ray intersects Gaussian i in the CUDA kernel, knowledge of T_{i-1} is needed to locate the subarray corresponding to the ith Gaussian's texture map, and U_i and V_i are necessary to simulate reshaping into a 2D grid and querying into the correct indices. Note that the ith Gaussian's texture map are not loaded into shared memory like with the other Gaussian parameters. The individual texture maps are simply too large. We do, however, load T_{i-1}, U_i, V_i into shared memory.

Texture painting. We elaborate on our texture painting method, which casts an edited image onto the textured Gaussian splats from a given viewpoint. For each texel of the GStex model, we aggregate the RGBA values of the rays which hit it so as to propose an updated texel color. To accommodate the transparency of Gaussians, we weight these aggregated values by the transmittance of each ray when it hits the texel, as well as the coefficient from bilinear interpolation. In actuality, some pixels of the edited image may be unedited, or have low alpha from filtering of a stroke. As such, the final texel colors should be interpolated between the original texel colors and the aggregated edited texel color. More precisely, for texel τ that is intersected with rays of RGBA values $(\hat{\mathbf{c}}_i, \alpha_i)$ with combined (between interpolation and transmittance) weight ω_i, we compute

$$\tau_{\text{edit}} = \sum \hat{\mathbf{c}}_i \alpha_i \omega_i,$$
$$w_0 = \sum \alpha_i \omega_i,$$
$$w_1 = \sum (1 - \alpha_i) \omega_i.$$

Then the updated texel color τ_{new} interpolates between the edited texel value τ_{edit} and the original texel value τ_{orig} with weight $w_0/(w_0 + w_1)$:

$$\tau_{\text{new}} = \frac{w_0 \tau_{\text{edit}} + w_1 \tau_{\text{orig}}}{w_0 + w_1}.$$

When updating the texture values, we ensure that only texels within 1×10^{-2} (in normalized depth coordinates) of the median depth from the corresponding view are altered. After this editing process, we can visualize the edited scene from any viewpoint in a 3D-consistent fashion.

Method	Blender PSNR	Blender SSIM	Blender LPIPS	DTU PSNR	DTU SSIM	DTU LPIPS
GStex	33.25	0.969	0.024	32.87	0.956	0.038
w/o stop gradient	33.07	0.968	0.024	32.82	0.956	0.039
reset never	33.23	0.969	0.024	32.94	0.956	0.037
reset every	33.01	0.967	0.026	32.88	0.955	0.040

Table 1. Ablation experiments for novel view synthesis and rendering performance on the synthetic Blender dataset and DTU dataset. The first row, GStex, refers to our method exactly as described in the paper. The next three rows give various metrics upon adjusting certain design choices. We report PSNR ↑, SSIM ↑, LPIPS ↓.

3. Additional Experiments

3.1. Evaluation Over Design Choices.

We show the results of three of our design choices in Table 1. In the first, we check the stop on the gradient between texture colors and Gaussian geometries. Allowing the gradient ("w/o stop gradient") decreases the visual quality slightly. In the other two rows, we evaluate the effect of

Method	Outdoor			Indoor				
	Bicycle	Garden	Stump	Bonsai	Counter	Kitchen	Room	Mean
3DGS	25.18	27.23	26.55	32.13	28.99	31.31	31.36	28.96
2DGS	24.79	26.74	26.19	31.32	28.08	30.40	30.57	28.30
GStex (10^7)	24.84	27.05	26.30	31.90	28.62	31.00	31.25	28.71
GStex (10^8)	24.91	27.15	26.36	32.05	28.68	31.12	31.37	28.81
3DGS	0.763	0.862	0.771	0.940	0.906	0.925	0.917	0.869
2DGS	0.742	0.850	0.759	0.933	0.895	0.919	0.911	0.858
GStex (10^7)	0.747	0.857	0.763	0.938	0.902	0.924	0.916	0.864
GStex (10^8)	0.752	0.861	0.766	0.939	0.905	0.926	0.919	0.867
3DGS	0.212	0.109	0.216	0.205	0.202	0.127	0.221	0.185
2DGS	0.214	0.098	0.192	0.147	0.183	0.111	0.188	0.162
GStex (10^7)	0.201	0.088	0.176	0.136	0.169	0.102	0.177	0.150
GStex (10^8)	0.192	0.083	0.172	0.125	0.158	0.097	0.167	0.142
3DGS	6000K	5700K	4900K	1300K	1200K	1800K	1500K	3200K
2DGS	5300K	3200K	3500K	810K	660K	850K	890K	2200K
GStex (10^7)	5300K	3200K	3500K	810K	660K	850K	890K	2200K
GStex (10^8)	5300K	3200K	3500K	810K	660K	850K	890K	2200K

Table 2. Novel view synthesis metrics for individual scenes on the MipNeRF-360 dataset. We report PSNR ↑, SSIM ↑, LPIPS ↓, and number of Gaussians, respectively.

resetting the texel size more or less often. In "reset never", we do not change the texel size at all after initialization. In "reset every", we change it after every initialization. In the actual method, GStex, we reset every 100 iterations. There is not a strong trend in either direction, suggesting that the choice of reset frequency is not significant.

3.2. Large-Scale Scenes

Though our focus is on object-centric scenes, our method still functions for large-scale scenes such as the MipNeRF-360 dataset [1]. We evaluate our method in Table 2. We use the same experimental set-up as with Table 1 of the main paper except that we use 10^7 and 10^8 texels rather than 10^6, as the number of Gaussians in the scenes are much greater. During the initial 2DGS training, normal regularization was enabled with a coefficient of 0.05 while it was disabled during GStex training. No distortion regularization was used in either stage. Similarly to the Blender and DTU scenes, we observe minor improvements in visual metrics compared to 2DGS. Even with 10^8 texels, the texels may be fairly large compared to the details in training and test images, as shown in Figures 2 and 3. In unbounded scenes especially, there is large variation in the range between camera views and individual Gaussians. As a result, the texels of foreground Gaussians cover a disproportionate area in the renders while background texels are undersampled and exhibit aliasing artifacts.

4. Experiment Details

We give additional details for the experiments presented in the paper.

4.1. Captured Scene

To validate the usefulness of GStex on real data, we capture our own scene, *whiteboard*. We choose a standing whiteboard with four visually apparent pieces of paper taped onto one side, and red marker drawings on another. We captured a move-around video with a Galaxy S21 mobile phone. During capture, we kept ISO and shutter speed fixed to minimize photometric inconsistency. From the video, we selected frames that represent wide range of viewpoints while minimizing motion blur, and we utilize SwinIR [5] to remove compression artifacts. Segmentation masks are generated using Segment Anything [4] with manual point prompts, and the output masks are further manually painted to remove visible holes and islands. After running COLMAP on our data, 126 images remained, which we undistorted and split into a train and test sets following the intervals of 8 practice typically used for real-world captures [1].

4.2. Novel View Synthesis

Full results to the novel view synthesis are listed in Tables 3 and 4. We use the official implementations of 3DGS

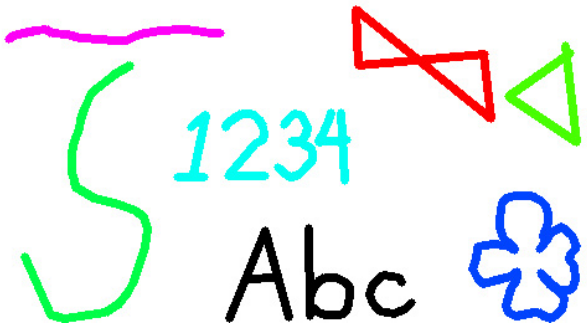

Figure 1. We drew various shapes and cast them from different views onto the Lego model.

[3], 2DGS [2], and Texture-GS [6]. We use the default hyperparameters for all. For 2DGS, their codebase had updated since the paper release and had tweaked the densification condition. More specifically, the version of the codebase which we worked with intentionally zeroed out gradients for Gaussians when they were sufficiently small to use their anti-aliasing blur. This noticeably reduced the number of Gaussians that were densified at a small cost to photometric quality, which the authors recommend.

For our evaluation of Texture-GS, we use their default training pipeline which has three stages composed of 30,000, 20,000, and 40,000 iterations, totalling to 90,000 iterations. Though this is $3\times$ that of GStex as well as 2DGS and 3DGS, we find that the visual quality of Texture-GS is noticeably behind others.

To calculate all metrics, we first quantize render RGBs to unsigned 8-bit integers before converting back to the float range $[0, 1]$ and applying the metric calculations. For LPIPS, we use version 0.1 with AlexNet activations.

4.3. Geometric Level of Detail

We give additional qualitative results for the geometric level of detail experiment. In Figures 4 and 5, we compare 2DGS and GStex in discrete level of detail for the remaining scenes not shown in Figure 6 of the main paper.

4.4. Appearance Editing

Texture painting. In Figure 1, we include the 2D images drawn on the Lego models in Figure 7 of the main paper.

Procedural textures. We evaluate the colors resulting from the procedural textures at the centers of each texel in world coordinates. In our implementation, we do this within the PyTorch framework. In Figure 8 of the main paper, we demonstrate the procedural texture

$$
\begin{aligned}
d(x, y, z) &= ||(x, y, z) - (\lfloor x \rceil, \lfloor y \rceil, \lfloor z \rceil)||_2 \\
R_1(x, y, z) &= 0.5(\sin(d) + 1), \\
G_2(x, y, z) &= 0, \\
B_2(x, y, z) &= 0.5\left(1 - \sin(d)\right),
\end{aligned}
$$

where $\lfloor t \rceil$ is t rounded to the nearest integer, creating circle-like patterns of blue and red. We also have the simple

$$
\begin{aligned}
R_2(x, y, z) &= 0.5\left(\sin(x) + 1\right), \\
G_2(x, y, z) &= 0.5\left(\sin(y) + 1\right), \\
B_2(x, y, z) &= 0.5\left(\sin(z) + 1\right),
\end{aligned}
$$

inducing axis-aligned stripes of color.

References

[1] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-NeRF: A multiscale representation for anti-aliasing neural radiance fields. In *Proc. CVPR*, 2021. 2

[2] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2D Gaussian splatting for geometrically accurate radiance fields. In *Proc. SIGGRAPH*, 2024. 3

[3] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian splatting for real-time radiance field rendering. In *Proc. SIGGRAPH*, 2023. 3

[4] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. In *Proc. ICCV*, 2023. 2

[5] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. SwinIR: Image restoration using Swin transformer. *arXiv preprint arXiv:2108.10257*, 2021. 2

[6] Tian-Xing Xu, Wenbo Hu, Yu-Kun Lai, Ying Shan, and Song-Hai Zhang. Texture-GS: Disentangling the geometry and texture for 3D Gaussian splatting editing. In *Proc. ECCV*, 2024. 3

Figure 2. **MipNeRF-360 outdoor scenes.** We provide test-set renders and texel visualizations of GStex with 10^8 texels applied to the outdoor scenes of MipNeRF-360. Each cell of the checkered Gaussians indicates a single texel. Note that the texel visualizations omit Gaussians with opacity < 0.5.

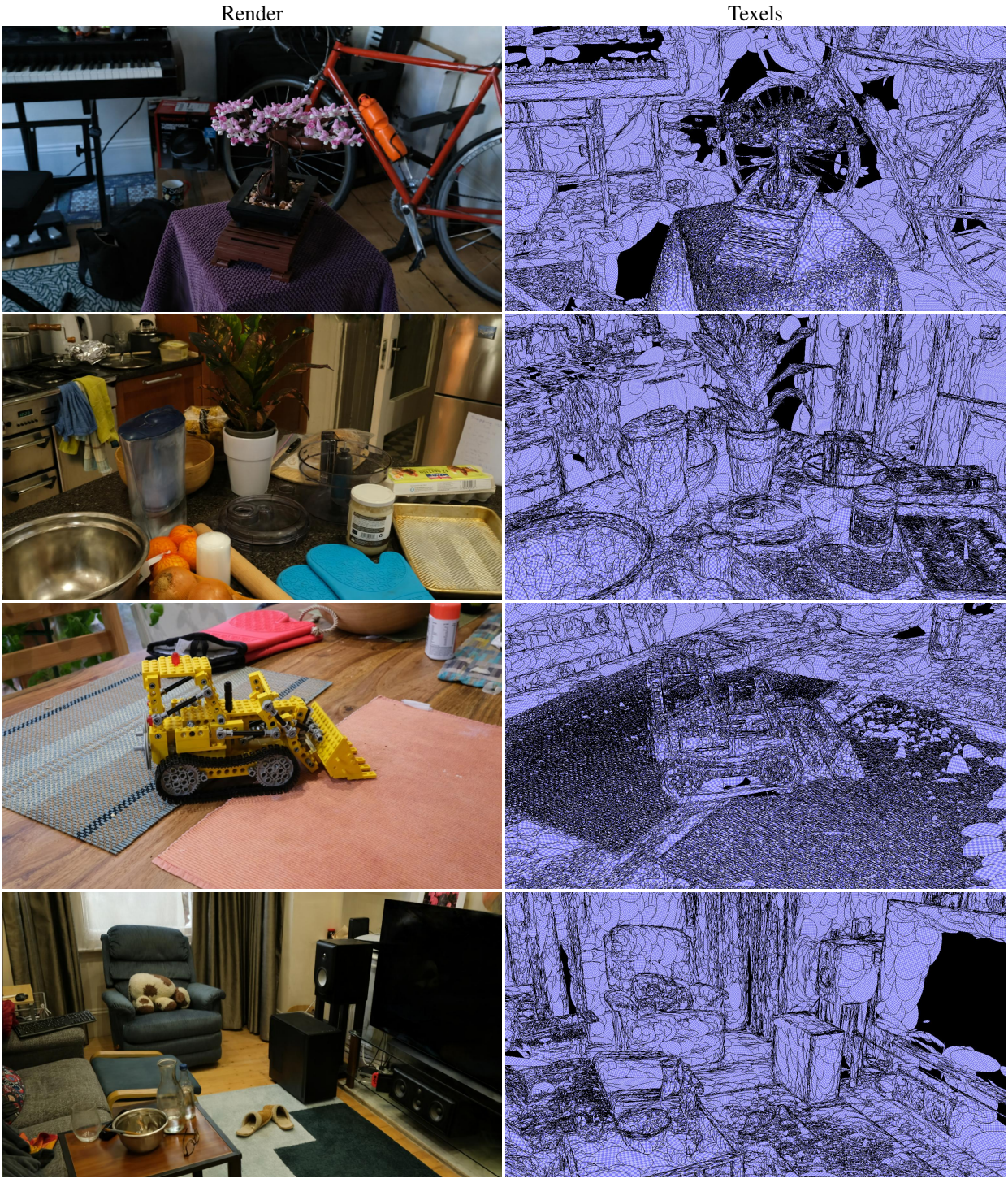

Figure 3. **MipNeRF-360 indoor scenes.** We provide test-set renders and texel visualizations of GStex with 10^8 texels applied to the indoor scenes of MipNeRF-360. Each cell of the checkered Gaussians indicates a single texel. Note that the texel visualizations omit Gaussians with opacity < 0.5.

Method	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship	Mean
3DGS	35.90	26.16	34.85	37.70	35.78	30.00	35.42	30.90	33.34
2DGS	35.41	26.12	35.39	37.47	35.25	29.74	35.20	30.66	33.15
Texture-GS	30.20	24.93	30.63	32.52	28.77	26.26	30.81	27.63	28.97
GStex	35.51	26.06	35.65	37.49	35.51	29.72	35.28	30.80	33.25
3DGS	0.987	0.955	0.987	0.985	0.983	0.960	0.992	0.907	0.969
2DGS	0.987	0.954	0.988	0.985	0.981	0.958	0.991	0.903	0.968
Texture-GS	0.950	0.937	0.968	0.962	0.935	0.925	0.975	0.853	0.938
GStex	0.987	0.954	0.988	0.985	0.982	0.958	0.991	0.904	0.969
3DGS	0.012	0.037	0.012	0.020	0.016	0.034	0.006	0.107	0.030
2DGS	0.009	0.039	0.008	0.014	0.012	0.021	0.006	0.087	0.024
Texture-GS	0.049	0.056	0.023	0.042	0.047	0.055	0.020	0.146	0.055
GStex	0.009	0.038	0.008	0.013	0.011	0.020	0.006	0.085	0.024
3DGS	270K	350K	290K	150K	320K	280K	310K	330K	290K
2DGS	100K	140K	50K	70K	160K	130K	150K	160K	120K
Texture-GS	60K	80K	20K	40K	100K	60K	30K	40K	50K
GStex	100K	140K	50K	70K	160K	130K	150K	160K	120K

Table 3. Novel view synthesis metrics for the synthetic Blender dataset. We report PSNR ↑, SSIM ↑, LPIPS ↓, and number of Gaussians, respectively, for individual scenes.

Method	24	37	40	55	63	65	69	83	97	105	106	110	114	118	122	Mean
3DGS	30.48	26.97	29.57	31.76	35.44	31.29	28.24	38.42	30.08	34.12	35.07	34.61	31.08	37.59	38.34	32.87
2DGS	29.99	26.18	28.77	31.30	34.35	30.76	27.90	37.72	29.77	33.46	34.93	33.32	30.58	36.69	37.60	32.22
Texture-GS	27.63	25.31	27.83	27.04	33.62	30.02	27.72	37.22	28.43	31.91	32.73	30.41	28.95	34.50	34.65	30.53
GStex	30.77	26.91	29.63	31.85	35.28	31.17	28.10	38.42	30.21	33.92	35.27	34.55	31.16	37.65	38.23	32.87
3DGS	0.937	0.919	0.917	0.963	0.971	0.965	0.938	0.981	0.950	0.963	0.963	0.970	0.951	0.975	0.980	0.956
2DGS	0.909	0.888	0.881	0.950	0.958	0.958	0.927	0.972	0.937	0.945	0.949	0.952	0.936	0.963	0.970	0.940
Texture-GS	0.875	0.886	0.847	0.893	0.949	0.955	0.908	0.970	0.923	0.924	0.927	0.936	0.907	0.944	0.955	0.920
GStex	0.937	0.921	0.916	0.965	0.971	0.965	0.935	0.980	0.949	0.961	0.962	0.968	0.954	0.974	0.979	0.956
3DGS	0.056	0.064	0.103	0.039	0.033	0.050	0.076	0.030	0.057	0.049	0.054	0.060	0.053	0.037	0.025	0.052
2DGS	0.089	0.087	0.147	0.045	0.074	0.093	0.144	0.059	0.098	0.094	0.096	0.093	0.095	0.074	0.060	0.090
Texture-GS	0.086	0.080	0.154	0.089	0.051	0.061	0.123	0.035	0.065	0.093	0.080	0.118	0.103	0.063	0.047	0.083
GStex	0.032	0.042	0.079	0.029	0.022	0.039	0.069	0.020	0.040	0.037	0.039	0.040	0.042	0.024	0.018	0.038
3DGGS	670K	700K	1010K	740K	130K	190K	210K	80K	360K	220K	280K	110K	300K	170K	180K	360K
2DGS	320K	440K	590K	330K	80K	90K	110K	50K	180K	120K	130K	60K	110K	100K	90K	190K
Texture-GS	60K	100K	120K	110K	20K	30K	50K	20K	60K	40K	50K	20K	50K	40K	40K	50K
GStex	320K	440K	590K	330K	80K	90K	110K	50K	180K	120K	130K	60K	110K	100K	90K	190K

Table 4. Novel view synthesis metrics for individual scenes on the DTU dataset. We report PSNR ↑, SSIM ↑, LPIPS ↓, and number of Gaussians, respectively.

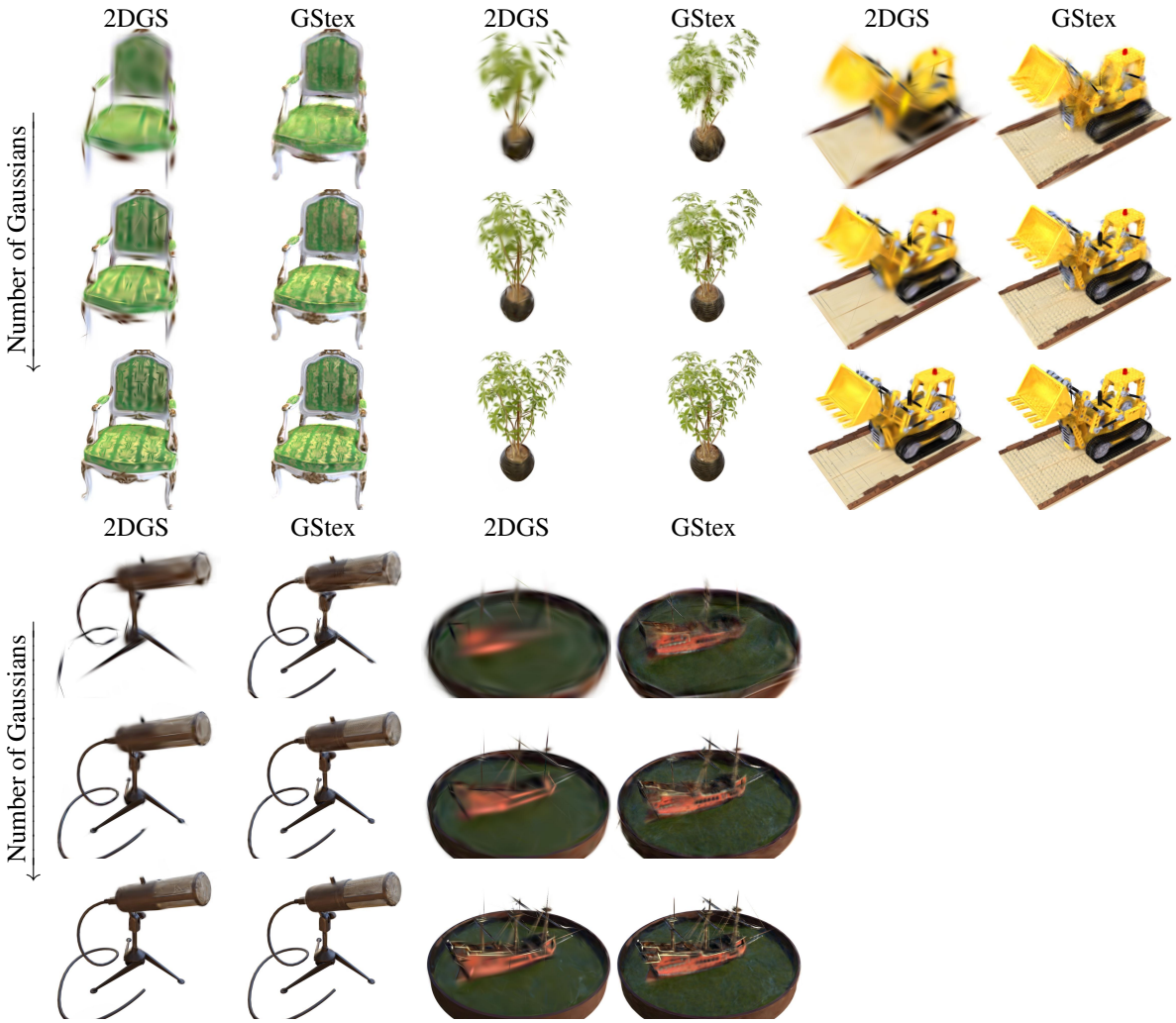

Figure 4. **Novel view synthesis for discrete levels of detail.** We show test set renders of GStex and 2DGS models in three settings of levels of detail: with 128, 512, and 2048 Gaussians. We show our results on the Blender synthetic scenes not shown in the main paper.

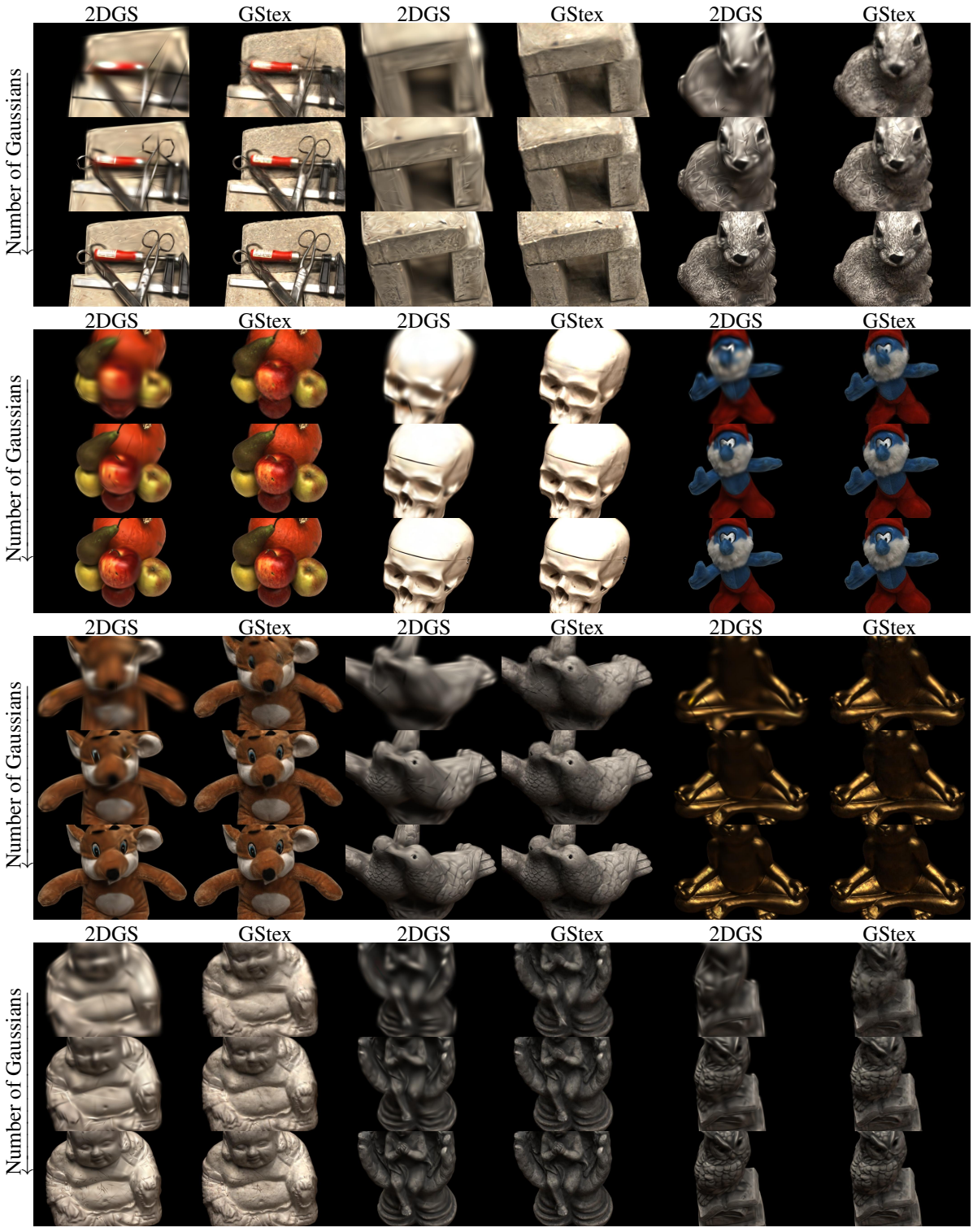

Figure 5. **Novel view synthesis for discrete levels of detail.** We show test set renders of GStex and 2DGS models in three settings of levels of detail: with 128, 512, and 2048 Gaussians. We show our results on the DTU scenes not shown in the main paper.