CompGS: Smaller and Faster Gaussian Splatting with Vector Quantization

K L Navaneet* Kossar Pourahmadi Meibodi*
Soroush Abbasi Koohpayegani Hamed Pirsiavash

University of California, Davis
{nkadur,kmeibodi,soroush,hpirsiav}@ucdavis.edu

In this supplementary pdf, we compare the performance of our CompGS with state-of-the-art approaches on the NeRF-Synthetic dataset (Section 1). Section 2 shows exploratory results on generalization of the learnt vector codebook across scenes and Section 4 provides insights on the learnt codebook assignments. We also provide scene-wise results (section 5), ablations on baselines (section 6) and additional visualizations and qualitative comparisons on the ARKit-200 dataset (section 8). The anonymized code for our approach is provided along with this pdf in the zipped supplementary material file.

1 Results on NeRF-Synthetic and DeepBlending Datasets

The results (PSNR) for the NeRF-Synthetic dataset [7] are presented in Table 1. Our CompGS approach achieves an impressive average improvement of 1.13 points in PSNR compared to the 3DGS-No-SH baseline while using less than half its memory. As reported in the main submission, we report metrics for 3DGS both from the original paper and using our own runs. We observe an improvement for 3DGS [5] over their official reported numbers by 0.5 points. The comparison between our CompGS and SOTA methods for novel view synthesis on Deep Blending dataset is shown in Table 2. CompGS maintains the speed and performance advantages of 3DGS while being 40× to 50× smaller. CompGS-32K BitQ is the post-training bit quantized version of CompGS-32K, in which position parameters are 16-bits, opacity is 8 bits, and the rest are 32 bits.

2 Generalization of codebook across scenes

We train our vector quantization approach including the codebook and the code assignments on a single scene ('Counter') of the Mip-NeRF360 dataset. We then freeze the codebook and learn only assignments for the rest of the eight scenes in the dataset and report the averaged performance metrics over all scenes. In addition to the results in Fig.5 of the main submission, we provide results with our 32K variant here in table 3. Interestingly, we observe that the shared codebook generalizes well across all scenes with a small drop in performance compared to learning a codebook for each scene. Sharing learnt codebook can further reduce

* Equal contribution

Table 1: Results on NeRF-Synthetic dataset. Here, we present the PSNR values for the synthesized novel views on the NeRF-Synthetic dataset [7]. Our CompGS approach achieves an impressive average improvement of 1.13 points in PSNR compared to the 3DGS-No-SH baseline while using less than half its memory. As reported in the main submission, we report metrics for 3DGS both from the original paper and using our own runs. We observe an improvement of 3DGS over the reported numbers by 0.5points. * indicates our own run.

	Mic	Chair	Ship	Materials	Lego	Drums	Ficus	Hotdog	Avg.
Plenoxels	33.26	33.98	29.62	29.14	34.10	25.35	31.83	36.81	31.76
INGP-Base	36.22	35.00	31.10	29.78	36.39	26.02	33.51	37.40	33.18
Mip-Nerf	36.51	35.14	30.41	30.71	35.70	25.48	33.29	37.48	33.09
Point-NeRF	35.95	35.40	30.97	29.61	35.04	26.06	36.13	37.30	33.30
3DGS	35.36	35.83	30.80	30.00	35.78	26.15	34.87	37.72	33.32
3DGS*	36.80	35.51	31.69	30.48	36.06	26.28	35.49	38.06	33.80
3DGS-No-SH	34.37	34.09	29.86	28.42	34.84	25.48	32.30	36.43	31.97
CompGS 4k	35.99	34.92	31.05	29.74	35.09	25.93	35.04	37.04	33.10

the memory requirement and can help speed up the training of CompGS. The quality of the codebook can be improved by learning it over multiple scenes. Fig. 3 shows qualitative comparison of the same. There are no apparent differences between CompGS and CompGS-Shared-Codebook approaches.

3 Variance in 3DGS Performance

We observe huge differences in the reported performance of 3DGS across different published works. To analyze this, we run the 3DGS baseline method 20 times on all the scenes of Mip-NeRF360 and Tanks&Temples datasets with different seeds and report the statistics of the performance in table 4. We only report the PSNRs for brevity. Some of the scenes (Bonsai, Kitchen, Room and Train) have a huge difference in performance between their best and worst runs with a PSNR difference more than 0.4. Thus, one must be careful when comparing performances of methods with small differences. The median of the standard deviation of these runs across all scenes is 0.048.

4 Analysis of learnt code assignments

In Fig. 1, we plot the sorted histogram of the code assignments (cluster to which each Gaussian belongs to) for each parameter on the 'Train' scene of Tanks&Temples dataset. We observe that just a single code out of the 512 in total is assigned to nearly 5% of the Gaussians for both the SH and DC parameters. Similarly, a few clusters dominate even in the case of rotation and scale parameters, albeit to a lower extent. Such a non-uniform distribution of cluster

Table 2: Comparison with SOTA methods for novel view synthesis on Deep Blending dataset. CompGS is a vector quantized version of 3DGS that maintains the speed and performance advantages of 3DGS while being **40×** to **50× smaller**. CompGS 32K BitQ is the post-training bit quantization version of CompGS 32K, in which position parameters are 16-bits, opacity is 8 bits, and rest are 32 bits. *Reproduced using official code. † Reported from 3DGS [5]. Our timings for 3DGS and CompGS are reported using a RTX6000 GPU while those with † used A6000 GPU. We boldface entries for emphasis.

	Deep Blending					
Method	SSIM↑	PSNR↑	LPIPS↓	FPS	Mem (MB)	Train Time(m)
Plenoxels† [4]	0.795	23.06	0.510	11.2	2700	27.5
INGP-Base† [8]	0.797	23.62	0.423	3.26	13	6.31
INGP-Big † [8]	0.817	24.96	0.390	2.79	48	8.00
M-NeRF360† [1]	0.901	29.40	0.245	0.09	8.6	48h
3DGS † [5]	0.903	29.41	0.243	137	676	36.2
3DGS * [5]	0.899	29.49	0.246	**151**	**662**	19.3
LigthGaussian [3]	-	-	-	-	-	-
CGR [6]	0.900	29.73	0.258	181	23.8	-
CGS [9]	0.898	29.38	0.253	-	25.30	-
CompGS 16K	0.906	29.90	0.252	**485**	**12**	20.3
CompGS 32K	0.907	29.90	0.251	484	13	26.2
CompGS 32K BitQ	0.907	29.89	0.251	**484**	**8**	26.2

sizes suggest that further compression can be achieved by using Huffman coding to store the assignment indices.

5 Scene-wise Metrics

For brevity, we reported the averaged metrics over all scenes in a dataset in our main submission. Here in table 5, we provide the detailed scene-wise metrics for MipNerf-360, Tanks and Temples and DeepBlending datasets.

6 Ablations for Gaussian count reduction

We perform ablations to choose the right hyperparameters for the baseline approaches to reduce the Gaussian count. The metrics for the chosen settings are reported in table 3 of the main submission. The ablations for minimum opacity, densification interval and end iteration and gradient threshold are shown in tables 6, 7, 8 and 9 respectively. Among these baselines, modify gradient threshold provides the best trade-off between model size and performance.

7 Details of ARKit-200 dataset.

Unlike visual recognition community that uses large scale benchmarks, interestingly, radiance field modeling community traditionally has used small datasets

Table 3: Effect of shared codebook. We train our vector quantization approach including the codebook and the code assignments on a single scene ('Counter') of the Mip-NeRF360 dataset. We then freeze the codebook and learn only assignments for the rest of the eight scenes in the dataset and report the averaged performance metrics over all scenes. Interestingly, we observe that the shared codebook generalizes well across all scenes with a small drop in performance compared to learning a codebook for each scene. Sharing learnt codebook can further reduce the memory requirement and can help speed up the training of CompGS. The quality of the codebook can be improved by learning it over multiple scenes.

Dataset	Mip-NeRF360		
Method	SSIM$^{\uparrow}$	PSNR$^{\uparrow}$	LPIPS$^{\downarrow}$
3DGS	0.815	27.21	0.214
3DGS *	0.813	27.42	0.217
CompGS 4K	0.804	26.97	0.234
CompGS 32K	0.806	27.12	0.240
CompGS Shared Codebook 4K	0.797	26.64	0.242
CompGS Shared Codebook 32K	0.800	26.780	0.247

Scene	1	2	3	4	5	6	7	8	9	10	11
Min	25.140	31.774	28.854	21.414	27.084	30.864	31.134	26.522	22.418	21.733	25.250
Max	25.271	32.183	29.068	21.562	27.293	31.425	31.545	26.624	22.585	22.193	25.415
Mean	25.221	32.038	28.997	21.491	27.227	31.228	31.427	26.583	22.495	21.940	25.333
Diff	0.131	0.409	0.215	0.148	0.210	0.561	0.410	0.102	0.167	0.460	0.165
Std	0.032	0.109	0.051	0.037	0.048	0.155	0.105	0.024	0.045	0.123	0.046

Table 4: Variance in 3DGS performance: We run the 3DGS baseline method 20 times on all the scenes of Mip-NeRF360 and Tanks&Temples datasets with different seeds and report the statistics of the performance. We only report the PSNRs for brevity. Some of the scenes (Bonsai, Kitchen, Room and Train) have a huge difference in performance between their best and worst runs (denoted as 'Diff') with a PSNR difference more than 0.4. The median of the standard deviation of these runs across all scenes is 0.048.

Table 5: Scene-wise metrics. We report the scene-wise metrics on all the scenes for both 3DGS and CompGS 32K. As observed in the averaged metrics in the main submission, CompGS achieves a high level of compression and fast rendering without losing much on rendering quality.

Scene	Method	SSIM$^{\uparrow}$	PSNR$^{\uparrow}$	LPIPS$^{\downarrow}$	Train Time(s)	FPS	Mem(MB)	#Gauss
Bicycle	3DGS	0.763	25.169	0.212	1845	68	1422	6026079
	CompGS 32K	0.755	25.068	0.244	2123	242	29	1314018
Bonsai	3DGS	0.940	31.918	0.206	876	276	293	1241520
	CompGS 32K	0.932	31.195	0.223	1405	492	10	377227
Counter	3DGS	0.906	29.018	0.202	968	208	283	1200091
	CompGS 32K	0.895	28.467	0.222	1571	356	10	385515
Flowers	3DGS	0.602	21.456	0.339	1192	133	851	3605827
	CompGS 32K	0.588	21.262	0.367	1744	343	23	1037132
Garden	3DGS	0.863	27.241	0.108	1870	76	1347	5709543
	CompGS 32K	0.848	26.822	0.140	2177	258	30	1370624
Kitchen	3DGS	0.926	31.510	0.127	1206	158	425	1801403
	CompGS 32K	0.918	30.774	0.142	1729	317	13	564382
Room	3DGS	0.917	31.346	0.221	1020	190	364	1541909
	CompGS 32K	0.912	31.131	0.235	1439	444	9	327191
Stump	3DGS	0.772	26.651	0.215	1445	104	1136	4815087
	CompGS 32K	0.770	26.605	0.236	1936	303	27	1226044
Treehill	3DGS	0.633	22.504	0.327	1213	122	879	3723675
	CompGS 32K	0.634	22.747	0.355	1744	336	23	1002290
Train	3DGS	0.811	21.991	0.209	563	253	254	1077461
	CompGS 32K	0.804	21.789	0.231	1169	456	12	500811
Truck	3DGS	0.878	25.385	0.148	897	159	611	2588966
	CompGS 32K	0.872	25.092	0.165	1306	494	13	540081
DrJohnson	3DGS	0.898	29.089	0.247	1312	121	772	3270679
	CompGS 32K	0.906	29.445	0.249	1717	379	17	714902
Playroom	3DGS	0.901	29.903	0.246	1003	181	551	2335846
	CompGS 32K	0.908	30.347	0.253	1422	589	10	393414

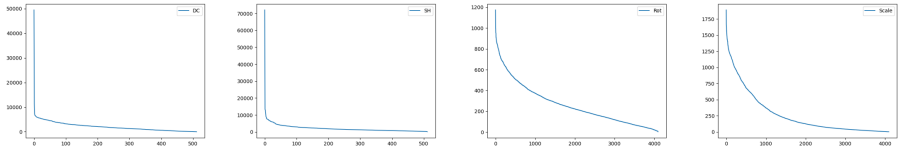

Fig. 1: Histogram of code assignments. We plot the sorted histogram of the code assignments (cluster to which each Gaussian belongs to) for each parameter on the 'Train' scene of Tanks&Temples dataset. We observe that just a single code out of the 512 in total is assigned to nearly 5% of the Gaussians for both the SH and DC parameters. Similarly, a few clusters dominate even in the case of rotation and scale parameters, albeit to a lower extent. Such a non-uniform distribution of cluster sizes suggest that further compression can be achieved by using Huffman coding to store the assignment indices.

Table 6: Ablation study on opacity threshold. We changed the default value of 0.005 for minimum opacity as a baseline approach for compressing 3DGS by reducing the number of Gaussians. Gaussians with opacity values below the threshold are pruned, resulting in smaller models for lower thresholds. Table 3 in main paper shows the results of this experiment when min opacity is 0.1.

	Mip-NeRF360				Tanks&Temples				Deep Blending			
	SSIM	PSNR	LPIPS	#Gauss	SSIM	PSNR	LPIPS	#Gauss	SSIM	PSNR	LPIPS	#Gauss
0.05	0.810	27.308	0.230	1.93M	0.839	23.508	0.194	1.04M	0.902	29.542	0.251	1.47M
0.1	0.802	27.120	0.244	1.46M	0.833	23.439	0.204	780K	0.902	29.504	0.255	1.01M

with a handful of 3D scenes: maximum of 13 total real world scenes in several papers (e.g., 3DGS [5]). We believe this is due to the computational cost of NeRF-based methods (several hours of training for each scene). Hence, with the recent advancements in this field including 3DGS that takes only a few minutes to learn a scene, it may be time to move beyond small benchmarks and evaluate methods on larger scale benchmarks. Therefore, we use an existing scene-understanding dataset [2] as a benchmark for radiance field modeling that is an order of magnitude larger than the traditional datasets (200 vs 13 scenes). We believe the community will benefit from using this larger benchmark to evaluate future radiance field methods with a reasonable computational demand. Note that DL3DV-140 is another large dataset that is concurrent work to ours, so we include that one too in the results of our main paper.

ARKit-200 [2] is an indoor scene understanding dataset comprising $5,048$ scans encompassing $1,661$ distinct scenes. The videos are recorded using the 2020 iPad Pro and have a resolution of 1920×1440. We exclusively utilize the RGB frames from each video. To construct the dataset subset for view synthesis, we randomly select 200 raw videos from the ARKit-200 dataset, extracting a uniform sample of 300 frames from each. Subsequently, we employ the code provided by 3DGS [5] to extract undistorted images and Structure-from-Motion

Table 7: Ablation study on densification interval. We modify the densification interval in 3DGS as a baseline approach for compressing 3DGS by reducing the number of Gaussians. Higher intervals results in less frequent densification and thus smaller number of Gaussians. Table 3 in main paper shows the results of this experiment when interval is 500.

	Mip-NeRF360				Tanks&Temples				Deep Blending			
	SSIM	PSNR	LPIPS	#Gauss	SSIM	PSNR	LPIPS	#Gauss	SSIM	PSNR	LPIPS	#Gauss
300	0.803	27.201	0.241	1.70M	0.837	23.517	0.195	1.00M	0.902	29.705	0.253	1.30M
500	0.794	26.98	0.255	1.07M	0.832	23.36	0.206	709K	0.902	29.76	0.258	844K

Table 8: Ablation study on densification end iteration. Early or late stopping of densification process impacts the number of Gaussians and the model performance of 3DGS. We report the results with the value set to 3000 in our main submission (table 3).

	Mip-NeRF360				Tanks&Temples				Deep Blending			
	SSIM	PSNR	LPIPS	#Gauss	SSIM	PSNR	LPIPS	#Gauss	SSIM	PSNR	LPIPS	#Gauss
5000	0.797	27.199	0.241	1.92M	0.838	23.599	0.188	1.12M	0.897	29.486	0.256	1.34M
3000	0.780	27.02	0.267	1.12M	0.835	23.55	0.194	810K	0.896	29.42	0.264	795K

(SfM) information from the input images. The dataset can easily be extended in the future by including more of the remaining scenes from the ARKit dataset.

8 Qualitative comparison on ARKit-200 dataset.

Figures 4 and 5 provide qualitative results on the ARKit-200 dataset.

References

1. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5470–5479 (2022) 3
2. Baruch, G., Chen, Z., Dehghan, A., Feigin, Y., Fu, P., Gebauer, T., Kurz, D., Dimry, T., Joffe, B., Schwartz, A., Shulman, E.: ARKitscenes: A diverse real-world dataset for 3d indoor scene understanding using mobile RGB-d data. In: Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1) (2021), https://openreview.net/forum?id=tjZjv_qh_CE 6
3. Fan, Z., Wang, K., Wen, K., Zhu, Z., Xu, D., Wang, Z.: Lightgaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ fps. arXiv preprint arXiv:2311.17245 (2023) 3
4. Fridovich-Keil, S., Yu, A., Tancik, M., Chen, Q., Recht, B., Kanazawa, A.: Plenoxels: Radiance fields without neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5501–5510 (2022) 3
5. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics (ToG) **42**(4), 1–14 (2023) 1, 3, 6

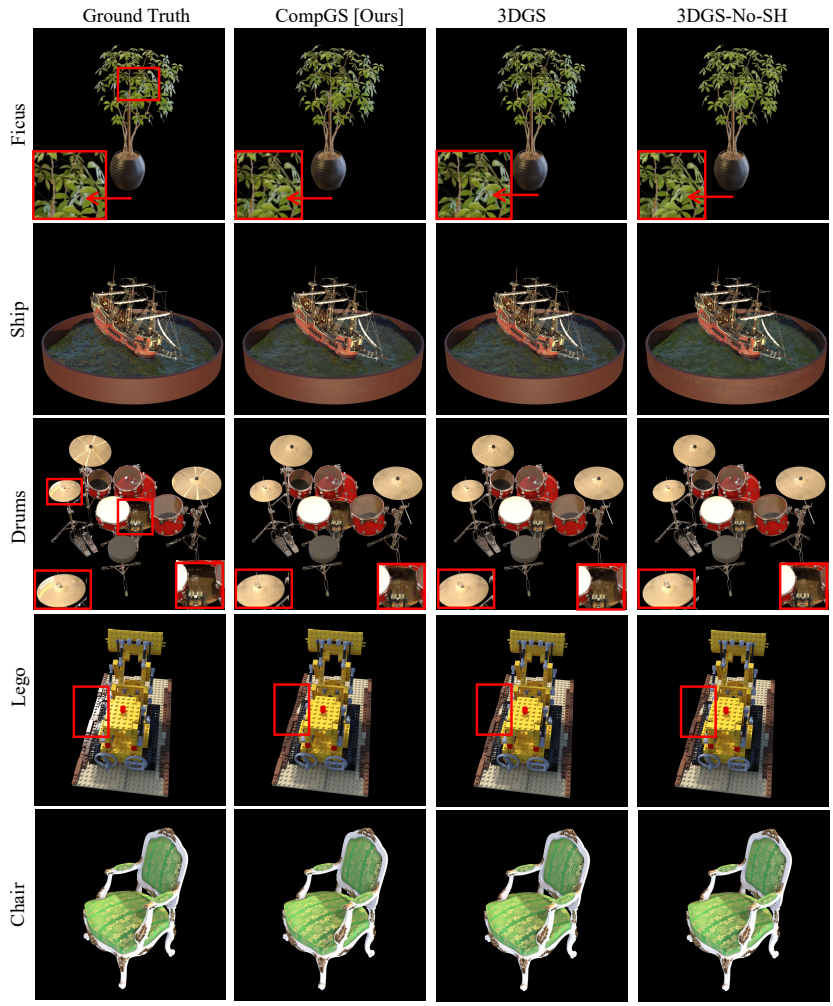

Fig. 2: Visualization of results on Sythetic-NeRF dataset. We compare the performance of our compressed CompGS with the original 3DGS and 3DGS-No-SH approaches on different scenes of the NeRF-Synthetic dataset. The difference between CompGS and 3DGS-No-SH is apparent in some of these scenes. E.g., 3DGS-No-SH fails to effectively model the brown color of branches and shadows and bright light on the leaves of the 'Ficus' scene. All approaches including 3DGS have imperfect reconstruction in some of the scenes like 'Drums' and 'Lego'. The scenes and views used for visualization were chosen at random.

Fig. 3: Qualitative analysis of shared codebook. We show the generalization of codebook learned using a single scene on various scenes of the Mip-NeRF360 dataset. The codebook was trained on the 'Counter' scene (row-1) and frozen for the remaining scenes. The codebooks for all four parameters (DC, SH, Scale, Rot) are shared across scenes. Both CompGS and CompGS-Shared-Codebook are visually similar to the uncompressed 3DGS with no conspicuous differences between them. 3DGS-No-SH requires twice more memory than CompGS while 3DGS is ten times bigger than CompGS . The scenes and views used for visualization were chosen at random.

Table 9: Ablation study on gradient threshold. We modify the gradient threshold, used as a criterion in the densification process of 3DGS. A higher threshold results in a smaller 3DGS model. Among the baselines considered for reducing Gaussian count, modify gradient threshold provides the best trade-off between model size and performance. Table 3 in main paper shows the results of this experiment for gradient threshold of 0.00045.

	Mip-NeRF360				Tanks&Temples				Deep Blending			
	SSIM	PSNR	LPIPS	#Gauss	SSIM	PSNR	LPIPS	#Gauss	SSIM	PSNR	LPIPS	#Gauss
0.00035	0.786	26.934	0.265	1.27M	0.833	23.579	0.203	833K	0.901	29.574	0.254	1.41M
0.00045	0.769	26.57	0.292	809K	0.825	23.31	0.217	578K	0.900	29.49	0.260	1.01M
0.00055	0.754	26.290	0.312	552K	0.818	23.041	0.228	433K	0.899	29.544	0.265	764K

6. Lee, J.C., Rho, D., Sun, X., Ko, J.H., Park, E.: Compact 3d gaussian representation for radiance field. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21719–21728 (2024) 3
7. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: Proceedings of the European Conference on Computer Vision (ECCV) (2020), http://arxiv.org/abs/2003.08934v2 1, 2
8. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Transactions on Graphics (ToG) **41**(4), 1–15 (2022) 3
9. Niedermayr, S., Stumpfegger, J., Westermann, R.: Compressed 3d gaussian splatting for accelerated novel view synthesis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10349–10358 (2024) 3

Fig. 4: Visualization of ARKit-200 dataset. ARKit-200 is a 3D indoor scene dataset captured using a iPads/iPhones. The dataset consists of videos of indoor environments like houses and office space from multiple view-points. We uniform sample images from each video to form our benchmark dataset for novel view synthesis. Some sample images from different scenes are shown in this figure. The dataset presents unique challenges such as the presence of motion blur due to the use of videos.

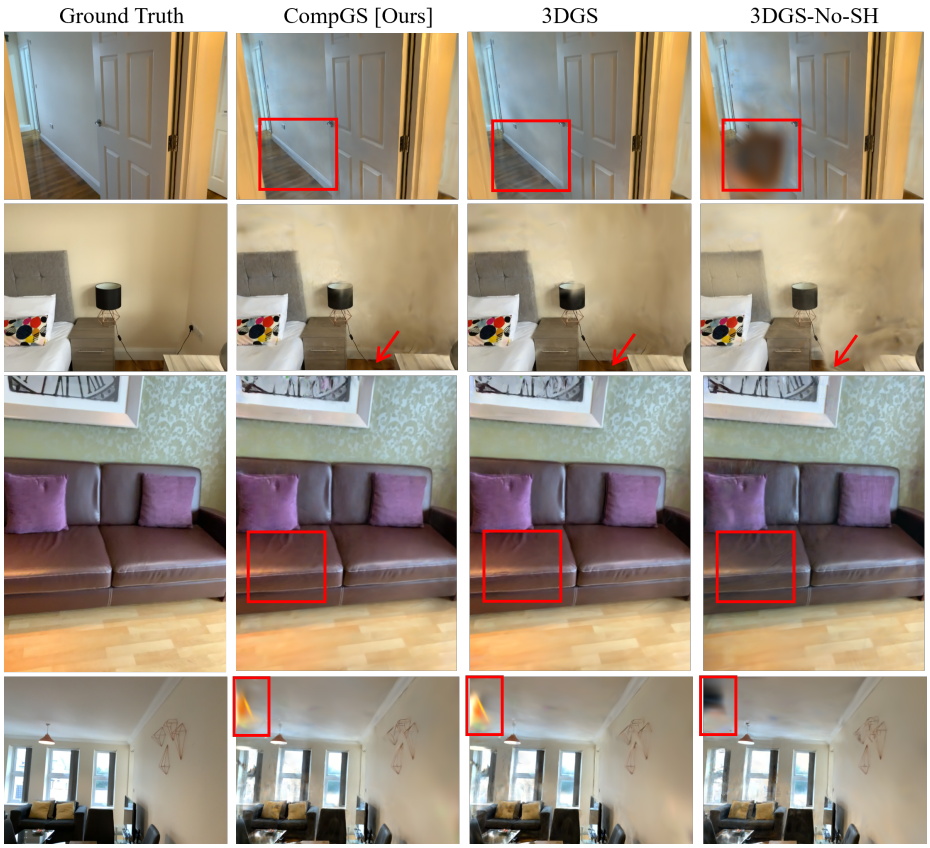

Fig. 5: Qualitative analysis on ARKit-200 dataset. We visualize the results of CompGS along with the uncomressed 3DGS and its variant 3DGS-No-SH . Presence of large noisy blobs is a common error mode for 3DGS-No-SH on this dataset. It also fails to faithfully reproduce the colors and lighting in several scenes. The visual quality of the synthesized images for all methods is lower on this dataset compared to the scenes present in standard benchmarks like Mip-NeRF360 , indicating its utility as a novel benchmark. Further comparison with various NeRF based approaches and more analysis can help improve the results on this dataset.