MaskGaussian: Adaptive 3D Gaussian Representation from Probabilistic Masks

Supplementary Material

6. Computing Gradients of Masks

For a pixel $\mathbf{x}$, we compute the gradient of the loss $\mathcal{L}$ with respect to the mask $\mathcal{M}_i$ by backpropagating the gradient of $\mathcal{L}$ with respect to the output pixel color $\mathbf{c(x)}$.

We split $\mathbf{c(x)}$ in two parts, the part colored by N Gaussian $\mathbf{c}_g\mathbf{(x)}$ and the part colored by background $\mathbf{c}_b\mathbf{(x)}$

$$\mathbf{c(x)} = \mathbf{c}_g\mathbf{(x)} + \mathbf{c}_b\mathbf{(x)}, \tag{9}$$

where $\mathbf{c}_b\mathbf{(x)}$ is contributed by the final transmittance T_{N+1} and background color $\mathbf{c}_{bg}$

$$\mathbf{c}_b\mathbf{(x)} = T_{N+1} \cdot \mathbf{c}_{bg}. \tag{10}$$

Now we compute the gradient of the first part with respect to $\mathcal{M}_i$. Recall that the definition of $\mathbf{b}_i$ is the color rendered from the i-th Gaussian to the last Gaussian, so we have the following equation from alpha blending:

$$\mathbf{b}_i = \mathcal{M}_i \cdot \alpha_i \cdot \mathbf{c}_i + (1 - \mathcal{M}_i \cdot \alpha_i) \cdot \mathbf{b}_{i+1}, \tag{11}$$

where $\mathbf{c}_i$ is the color of the i-th Gaussian.

According to the definition of $\mathbf{b}_1$, we have

$$\frac{\partial \mathcal{L}}{\partial \mathbf{b}_1} = \frac{\partial \mathcal{L}}{\partial \mathbf{c}_g\mathbf{(x)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{c(x)}}. \tag{12}$$

We can use (11) and (12) to compute the gradient with respect to $\mathbf{b}_i$ with standard chain rule

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \mathbf{b}_i} &= \frac{\partial \mathcal{L}}{\partial \mathbf{b}_1} \sum_{j=1}^{i-1} \frac{\partial \mathbf{b}_j}{\partial \mathbf{b}_{j+1}} \\ &= \frac{\partial \mathcal{L}}{\partial \mathbf{c(x)}} \sum_{j=1}^{i-1} (1 - \mathcal{M}_j \cdot \alpha_j)\mathbb{I}_3 \\ &= T_i \frac{\partial \mathcal{L}}{\partial \mathbf{c(x)}}. \end{aligned} \tag{13}$$

From (11) we have the gradient of $\mathbf{b}_i$ with respect to the mask

$$\frac{\partial \mathbf{b}_i}{\partial \mathcal{M}_i} = \alpha_i \cdot (\mathbf{c}_i - \mathbf{b}_{i+1}). \tag{14}$$

Therefore we write the gradient with respect to the mask as

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \mathcal{M}_i} &= \frac{\partial \mathcal{L}}{\partial \mathbf{b}_i} \frac{\partial \mathbf{b}_i}{\partial \mathcal{M}_i} + \frac{\partial \mathcal{L}}{\partial \mathbf{c(x)}} \frac{\partial \mathbf{c}_b\mathbf{(x)}}{\partial \mathcal{M}_i} \\ &= \alpha_i T_i \frac{\partial \mathcal{L}}{\partial \mathbf{c(x)}} (\mathbf{c}_i - \mathbf{b}_{i+1}) + \frac{\partial \mathcal{L}}{\partial \mathbf{c(x)}} \frac{\partial \mathbf{c}_b\mathbf{(x)}}{\partial \mathcal{M}_i}. \end{aligned} \tag{15}$$

For the gradient of the second part, notice that

$$T_{N+1} = \sum_{i=1}^{N} (1 - \alpha_i \cdot \mathcal{M}_i). \tag{16}$$

We can thus compute

$$\frac{\partial \mathbf{c}_b\mathbf{(x)}}{\partial \mathcal{M}_i} = \frac{-\alpha_i \cdot T_{N+1}}{1 - \alpha_i \cdot \mathcal{M}_i} \mathbf{c}_{bg}. \tag{17}$$

Combining the first and second part, we get

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \mathcal{M}_i} &= \alpha_i T_i \frac{\partial \mathcal{L}}{\partial \mathbf{c(x)}} (\mathbf{c}_i - \mathbf{b}_{i+1}) \\ &\quad + \frac{-\alpha_i \cdot T_{N+1}}{1 - \alpha_i \cdot \mathcal{M}_i} \frac{\partial \mathcal{L}}{\partial \mathbf{c(x)}} \mathbf{c}_{bg}. \end{aligned} \tag{18}$$

For brevity, the second part was omitted from the main manuscript in Eq 5.

7. Comparison with LightGaussian (training from scratch)

LightGaussian [5] is a method developed to compress an already trained 3D Gaussian Splatting (3DGS) model through additional training steps. Besides the post-training setting, we compare with LightGaussian by training from scratch and apply Gaussian pruning at the 20,000th iteration[1]. Our model apples λ_m= 0.1 during 19,000 to 20,000 iterations.

Table 9. Comparison with LightGaussian on Mip-NeRF360.

Dataset	Mip-NeRF360			
Metrics	PSNR↑	SSIM↑	LPIPS↓	#GS (M)↓
3DGS	27.45	0.811	0.223	3.204
LightGaussian	27.10	0.800	0.246	1.090
Ours	27.44	0.811	0.227	1.205

Table 10. Comparison with LightGaussian on Tanks & Temples.

Dataset	Tanks &Temples			
Metrics	PSNR↑	SSIM↑	LPIPS↓	#GS (M)↓
3DGS	23.74	0.848	0.176	1.825
LightGaussian	23.04	0.822	0.222	0.625
Ours	23.72	0.847	0.181	0.590

Table 11. Comparison with LightGaussian on Deep Blending.

Dataset	Deep Blending			
Metrics	PSNR↑	SSIM↑	LPIPS↓	#GS (M)↓
3DGS	29.53	0.903	0.243	2.815
LightGaussian	27.29	0.877	0.294	0.752
Ours	29.69	0.907	0.244	0.694

[1]This configuration follows the setup used in LightGaussian's official implementation.

Table 8. Ablation study for masking loss.

Dataset	Mip-NeRF360				Tanks&Temples				Deep Blending			
Metrics	PSNR↑	SSIM↑	LPIPS↓	#GS (M)↓	PSNR↑	SSIM↑	LPIPS↓	#GS (M)↓	PSNR↑	SSIM↑	LPIPS↓	#GS (M)↓
Compact3DGS-L1	27.32	0.805	0.233	1.533	23.61	0.846	0.180	0.960	29.58	0.903	0.248	1.310
Compact3DGS-L2	27.33	0.805	0.231	1.745	23.69	0.846	0.180	1.066	29.58	0.904	0.246	1.660
Ours-L1	27.42	0.811	0.225	1.811	23.56	0.845	0.179	0.900	29.71	0.905	0.243	1.208
Ours-L2	27.44	0.811	0.226	1.520	23.66	0.846	0.180	0.740	29.76	0.907	0.244	0.913

8. Ablation for the Masking Loss

Through experiments, we observe that the L2 masking loss Eq. 7 outperforms the L1 loss for our proposed method. However, when applied to Compact3DGS [26], the L2 masking loss does not demonstrate a performance advantage over the L1 loss. The results are shown in Tab. 8.

9. Spatial Distribution of Gaussians

We visualize the Gaussian centers in Fig. 5. While 3DGS occasionally exhibits dense clusters of Gaussians in some regions and sparse distributions in others, leading to spatial inefficiency, Compact3DGS[26] attempts to prune Gaussian points but still suffers from clustering issues. In contrast, our method achieves a more uniform spatial distribution of Gaussians, effectively mitigating this problem.

10. Per-Scene Results

We provide the per-scene results of Tab. 1 in Mip-NeRF360 (Tab. 12), Tanks & Temples and Deep Blending (Tab. 13).

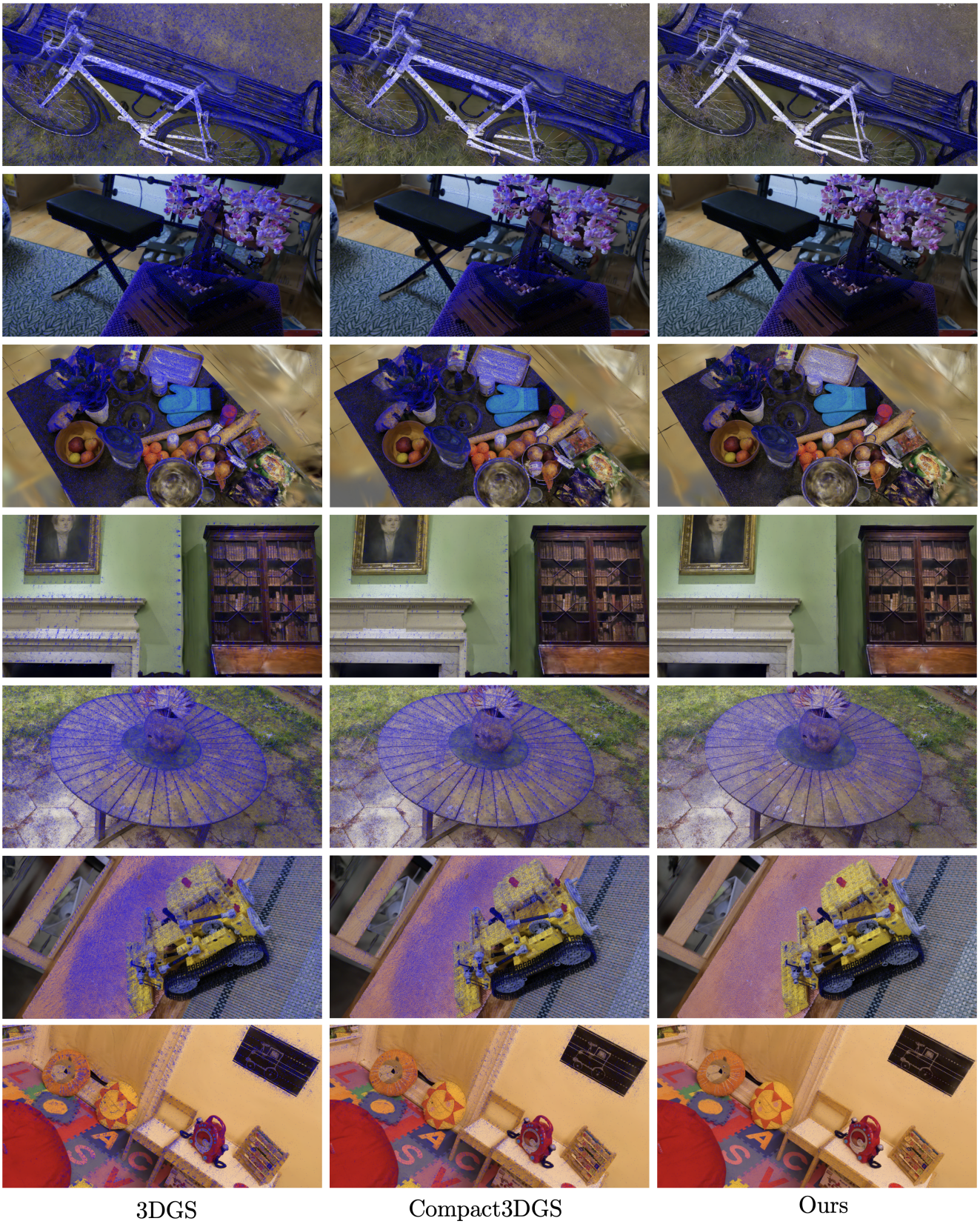

Figure 5. Gaussian centers visualized as blue points. Best viewed zoomed in.

Table 12. Per Scene Results on Mip-NeRF 360 dataset.

Scene		bicycle	bonsai	counter	flowers	garden	kitchen	room	stump	treehill	Avg.
3DGS	PSNR	25.09	32.29	29.09	21.35	27.38	31.3	31.44	26.59	22.53	27.45
	SSIM	0.745	0.946	0.915	0.587	0.857	0.931	0.919	0.767	0.633	0.811
	LPIPS	0.244	0.179	0.183	0.358	0.122	0.116	0.217	0.244	0.347	0.223
	#GS (M)	5.70	1.25	1.16	3.47	5.81	1.75	1.56	4.65	3.49	3.204
	FPS	89.7	332.6	244.1	182.8	100.9	195.1	231.3	148.8	164.8	187.7
Compact3DGS	PSNR	24.8	32.23	29.02	21.31	27.06	31.13	31.45	26.43	22.45	27.32
	SSIM	0.729	0.946	0.913	0.578	0.846	0.93	0.917	0.758	0.628	0.805
	LPIPS	0.264	0.181	0.185	0.371	0.139	0.119	0.223	0.26	0.354	0.232
	#GS (M)	2.66	0.66	0.56	1.69	2.46	1.07	0.58	1.97	2.15	1.533
	FPS	146.8	449.2	337.4	301.9	178.4	253.1	383.3	259.0	220.6	281.1
RadSplat	PSNR	25.07	32.3	29.08	21.38	27.31	31.49	31.44	26.58	22.43	27.45
	SSIM	0.745	0.946	0.915	0.588	0.856	0.932	0.919	0.768	0.632	0.811
	LPIPS	0.244	0.179	0.183	0.359	0.123	0.116	0.218	0.244	0.348	0.223
	#GS (M)	3.85	0.868	0.778	2.56	4.01	1.26	0.874	2.94	2.52	2.184
	FPS	132.6	411.0	318.2	228.6	139.3	244.8	347.3	199.1	209.8	247.8
Ours	PSNR	25.08	31.9	29.01	21.33	27.34	31.54	31.42	26.67	22.6	27.43
	SSIM	0.746	0.944	0.913	0.587	0.856	0.931	0.918	0.77	0.634	0.811
	LPIPS	0.248	0.184	0.188	0.361	0.125	0.119	0.222	0.244	0.352	0.227
	#GS (M)	2.34	0.353	0.328	1.41	2.13	0.516	0.366	1.90	1.51	1.205
	FPS	200.5	619.3	492.1	347.7	227.2	435.0	542.0	289.2	309.3	384.7

Table 13. Per Scene Results on Tanks & Temple and Deep Blending dataset.

Dataset		Tanks & Temples			Deep Blending		
Scene		train	truck	Avg.	drjohnson	playroom	Avg.
3DGS	PSNR	22.06	25.43	23.74	29.05	29.99	29.52
	SSIM	0.815	0.882	0.848	0.900	0.906	0.903
	LPIPS	0.206	0.146	0.176	0.244	0.242	0.243
	#GS (M)	1.08	2.57	1.825	3.30	2.33	2.815
	FPS	296.6	212.5	254.5	163.0	239.5	201.2
Compact3DGS	PSNR	21.89	25.33	23.61	29.14	30.02	29.58
	SSIM	0.812	0.880	0.846	0.900	0.906	0.903
	LPIPS	0.210	0.150	0.180	0.248	0.248	0.248
	#GS (M)	0.81	1.11	0.960	1.62	1.00	1.310
	FPS	358.6	359.3	358.9	293.3	439.1	366.2
RadSplat	PSNR	21.81	25.4	23.605	29.06	30.04	29.55
	SSIM	0.813	0.882	0.847	0.900	0.907	0.903
	LPIPS	0.208	0.147	0.177	0.244	0.243	0.243
	#GS (M)	0.737	1.37	1.053	1.75	1.28	1.515
	FPS	433.5	359.4	396.4	296.9	393.7	345.3
Ours	PSNR	22.01	25.43	23.72	29.21	30.18	29.695
	SSIM	0.812	0.882	0.847	0.904	0.910	0.907
	LPIPS	0.214	0.148	0.181	0.244	0.245	0.244
	#GS (M)	0.402	0.779	0.590	0.880	0.508	0.694
	FPS	607.0	509.7	558.3	466.1	598.7	532.4