GaussianUpdate: Continual 3D Gaussian Splatting Update for Changing Environments
Supplementary Material

Lin Zeng* Boming Zhao* Jiarui Hu Xujie Shen
Ziqiang Dang Hujun Bao Zhaopeng Cui†
State Key Lab of CAD & CG, Zhejiang University

In this supplementary material, we provide additional details about the GaussianUpdate framework, including: (1) a detailed explanation of the Appearance Model (Sec. A); (2) the Sky Ball (Sec. B); (3) additional experimental results (Sec. C) on the Synthetic NeRF dataset (Sec. D) and the WAT dataset (Sec. E); and (4) additional experiments (Sec. F). We also discuss how our method requires fewer Gaussian points compared to training a 3D Gaussian model at each time step (Sec. G). Additionally, a video supplement is provided to summarize our method and showcase the results.

A. Further Details on the Appearance Model

Our appearance model consists of a 4D hash grid and a tiny MLP. The 4D hash grid consists of L=16 layers of hash tables, each layer containing T=2^{21} features of dimension F=4, with a coarsest grid resolution of 16. The tiny MLP consists of two layers, each with 128 neurons. The appearance model takes position xyz$\in \mathbb{R}^3$ and timestamp t$\in \mathbb{R}^1$as inputs and outputs features of 51 dimensions, comprising the spherical harmonics coefficients$\in \mathbb{R}^{48}$ and scaling$\in \mathbb{R}^3$.

B. Sky Ball

The WAT dataset contains numerous unbounded sky areas across scene communities, spa, and streets. Since SFM or MVS does not initialize points in sky regions, directly using these initialization points for training can result in suboptimal outcomes. To address this, we introduce a "sky ball" during the training of the initial 3D Gaussian Splatting model with $\mathbf{T}_0$in these scenes. Specifically, we first determine the center and radius of the MVS initial point cloud. We then uniformly distribute 100,000 points within a 3D sphere, with a radius ten times that of the initial point cloud, as the initialized Gaussian. This approach significantly enhances the performance of 3D Gaussian Splatting in unbounded scenes with extensive sky regions.

*Equal contribution.
†Corresponding author.

Method	WAT		Synthetic NeRF	
	PSNR(↑)	SSIM(↑)	PSNR(↑)	SSIM(↑)
NT	16.89	0.631	22.27	0.812
CLNeRF [1]	25.45	0.764	32.16	0.957
4DGS [4]	25.70	0.816	32.18	0.959
Ours	26.31	0.83	33.44	0.967
UB	26.47	0.83	33.87	0.969

Table 1. **Quantitative comparison** . Best results are highlighted as **first** , **second.**

C. Further Quantitative Results

Table 1 presents the average PSNR and SSIM results of various methods across different datasets. It is evident that our method achieves higher PSNR and SSIM values compared to CLNeRF, and its performance is very close to the upper bound of our approach.

Method	Kitchen	Breville	Car	Average
3DGStream [3]	26.40	24.90	20.05	23.78
Ours	28.02	30.11	23.81	27.31

Table 2. **Comparison of the average PSNR between our method and 3DGS across the Kitchen, Breville, and Car scenes in the WAT dataset**. Best results are highlighted as **first** , **second.**

D. Additional Qualitative Results for the Synthetic NeRF Dataset

We further show the quantitative results of different methods on the Synthetic NeRF dataset in Figure 1.

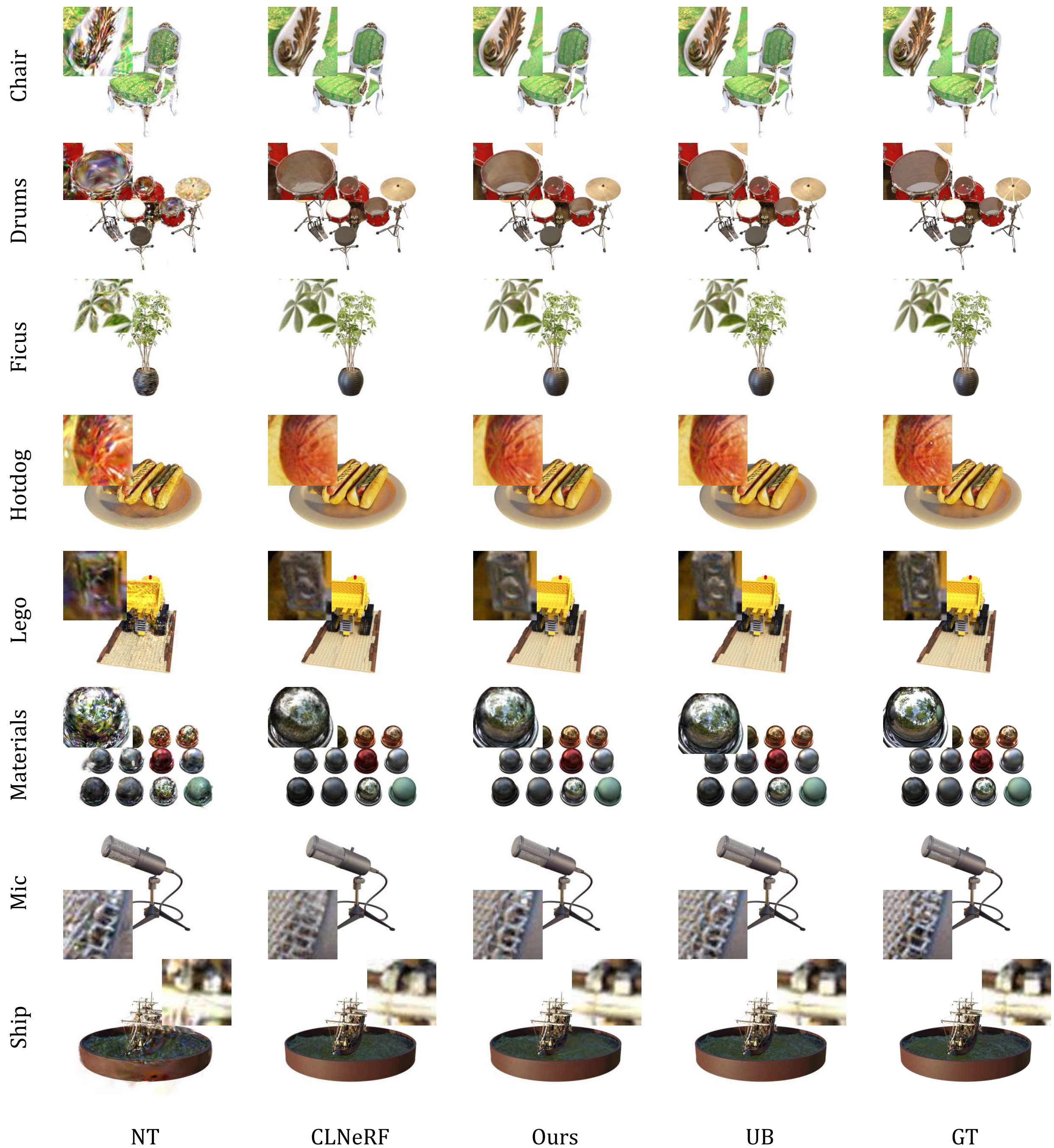

Figure 1. **Visualizaton results on Synthetic NeRF dataset**.

E. Quantitative Results of Our Method for Individual Scenes in the WAT Dataset

Table 5 and Table 6 present the PSNR of our method at each moment in all scenarios of the WAT dataset. Since the SSIM does not change significantly after each model update (for example, the SSIM measured on the test dataset at the first moment with the model updated at the first moment is almost identical to the SSIM measured on the test dataset at the first moment with the model updated at the last moment), we only show the SSIM results at the last moment.

Method	Playground		Truck	
	PSNR(↑)	SSIM(↑)	PSNR(↑)	SSIM(↑)
Baseline	12.59	0.416	12.75	0.479
CLNeRF [1]	22.37	0.643	22.61	0.695
Ours	22.93	0.744	22.44	0.770
UB	23.34	0.771	23.38	0.804

Table 3. **Quantitative comparison on Tanks&Temples dataset**. Best results are highlighted as **first**, **second**.

F. Additional Experiment

We further compared our method with 3DGStream [3] on the WAT [1] dataset. Although the training process of 3DGStream is similar to ours, it requires a Neural Transformation Cache (NTC) to be saved for every moment, leading to increased memory consumption. Moreover, as shown in Table 2 and Figure 2, 3DGStream struggles to handle dynamic scenes with significant changes between consecutive moments, making it less suitable for such scenarios.

We also selected several scenes from Tanks&Temples [2] dataset for our experiments, with the results presented in Table 3. The datasets were utilized in a manner similar to Synthetic NeRF and were divided into 10 sequences for training.

In addition, we also tested the effect of setting different training iterations at different training stages. We kept the total training iterations for the three training stages at 30k, and changed the training steps in the first and second stages. As shown in the table 4, varying iteration counts yields no significant performance differences. These demonstrate that our method is robust to stage iteration settings.

1st stage \ 2nd stage	7K Iters	8K Iters	9K Iters
6K Iters	30.19	30.16	30.10
7K Iters	30.17	30.11	30.08
8K Iters	30.16	30.15	30.13

Table 4. Effect of different iterations settings PSNR results on WAT Breville scene.

G. Memory

Compared to training a separate 3D Gaussian model for each moment, our method requires only a single model to encapsulate information from multiple scenes. On the WAT dataset, our method uses an average of 870,000 Gaussian points to model each scene, whereas retraining a 3D Gaussian model at each time step would require an average of 1.5 million Gaussian points. This represents an average reduction of 42% in the number of Gaussian points needed.

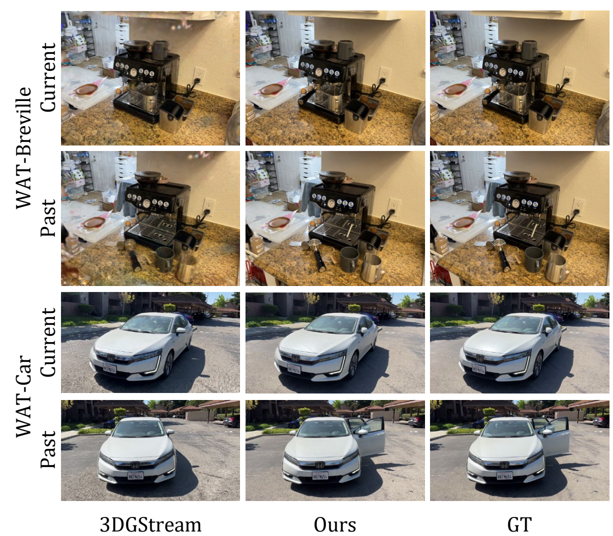

Figure 2. **Visualizaton results on WAT dataset with 3DGStream**.

Moment	Community										
	PSNR ↑										SSIM ↑
	$\mathbf{T}_0$	$\mathbf{T}_1$	$\mathbf{T}_2$	$\mathbf{T}_3$	$\mathbf{T}_4$	$\mathbf{T}_5$	$\mathbf{T}_6$	$\mathbf{T}_7$	$\mathbf{T}_8$	$\mathbf{T}_9$	
$\mathbf{T}_0$	25.43	-	-	-	-	-	-	-	-	-	0.837
$\mathbf{T}_1$	25.43	23.01	-	-	-	-	-	-	-	-	0.707
$\mathbf{T}_2$	25.43	23.05	24.39	-	-	-	-	-	-	-	0.809
$\mathbf{T}_3$	25.43	23.05	24.37	26.76	-	-	-	-	-	-	0.839
$\mathbf{T}_4$	25.43	23.03	24.28	26.70	24.80	-	-	-	-	-	0.822
$\mathbf{T}_5$	25.43	22.98	24.24	26.55	24.90	24.62	-	-	-	-	0.809
$\mathbf{T}_6$	25.43	22.99	24.12	26.44	24.87	24.63	23.44	-	-	-	0.779
$\mathbf{T}_7$	25.43	22.96	24.08	26.38	24.79	24.57	23.30	25.16	-	-	0.831
$\mathbf{T}_8$	25.43	22.96	24.05	26.31	24.71	24.52	23.17	24.95	22.65	-	0.697
$\mathbf{T}_9$	25.43	22.98	24.02	26.22	24.65	24.46	23.17	24.92	22.56	20.25	0.531

Table 5. The results of Community scene at each moment.

Moment	Breville						Kitchen					
	PSNR $\uparrow$					SSIM $\uparrow$	PSNR $\uparrow$					SSIM $\uparrow$
	$\mathbf{T}_0$	$\mathbf{T}_1$	$\mathbf{T}_2$	$\mathbf{T}_3$	$\mathbf{T}_4$		$\mathbf{T}_0$	$\mathbf{T}_1$	$\mathbf{T}_2$	$\mathbf{T}_3$	$\mathbf{T}_4$	
$\mathbf{T}_0$	29.93	-	-	-	-	0.933	28.94	-	-	-	-	0.920
$\mathbf{T}_1$	29.93	32.12	-	-	-	0.948	28.94	28.28	-	-	-	0.909
$\mathbf{T}_2$	29.93	32.07	28.41	-	-	0.907	28.94	28.29	28.71	-	-	0.909
$\mathbf{T}_3$	29.93	32.03	28.28	29.29	-	0.916	28.94	28.23	28.64	28.62	-	0.925
$\mathbf{T}_4$	29.93	32.05	28.04	29.15	30.67	0.928	28.94	28.24	28.66	28.65	26.22	0.904

= Moment	Living room						Ninja					
	PSNR $\uparrow$					SSIM $\uparrow$	PSNR $\uparrow$					SSIM $\uparrow$
	$\mathbf{T}_0$	$\mathbf{T}_1$	$\mathbf{T}_2$	$\mathbf{T}_3$	$\mathbf{T}_4$		$\mathbf{T}_0$	$\mathbf{T}_1$	$\mathbf{T}_2$	$\mathbf{T}_3$	$\mathbf{T}_4$	
$\mathbf{T}_0$	25.13	-	-	-	-	0.879	28.86	-	-	-	-	0.943
$\mathbf{T}_1$	25.13	25.90	-	-	-	0.883	28.86	29.20	-	-	-	0.927
$\mathbf{T}_2$	25.13	25.88	25.91	-	-	0.866	28.86	29.20	27.36	-	-	0.917
$\mathbf{T}_3$	25.13	25.83	25.84	27.03	-	0.905	28.86	29.20	27.41	27.78	-	0.929
$\mathbf{T}_4$	25.13	25.70	25.78	26.98	26.82	0.876	28.86	29.14	27.36	27.66	26.07	0.895

Moment	Spa						Street					
	PSNR $\uparrow$					SSIM $\uparrow$	PSNR $\uparrow$					SSIM $\uparrow$
	$\mathbf{T}_0$	$\mathbf{T}_1$	$\mathbf{T}_2$	$\mathbf{T}_3$	$\mathbf{T}_4$		$\mathbf{T}_0$	$\mathbf{T}_1$	$\mathbf{T}_2$	$\mathbf{T}_3$	$\mathbf{T}_4$	
$\mathbf{T}_0$	28.89	-	-	-	-	0.911	26.20	-	-	-	-	0.862
$\mathbf{T}_1$	28.89	27.78	-	-	-	0.887	26.20	24.20	-	-	-	0.771
$\mathbf{T}_2$	28.89	27.75	27.11	-	-	0.887	26.20	24.21	21.44	-	-	0.659
$\mathbf{T}_3$	28.89	27.67	27.09	27.23	-	0.869	26.20	24.20	21.41	21.83	-	0.668
$\mathbf{T}_4$	28.89	27.74	27.11	27.03	28.36	0.911	26.20	24.20	21.40	21.83	21.17	0.602

Moment	Car						Grill					
	PSNR $\uparrow$					SSIM $\uparrow$	PSNR $\uparrow$					SSIM $\uparrow$
	$\mathbf{T}_0$	$\mathbf{T}_1$	$\mathbf{T}_2$	$\mathbf{T}_3$	$\mathbf{T}_4$		$\mathbf{T}_0$	$\mathbf{T}_1$	$\mathbf{T}_2$	$\mathbf{T}_3$	$\mathbf{T}_4$	
$\mathbf{T}_0$	23.66	-	-	-	-	0.750	27.23	-	-	-	-	0.825
$\mathbf{T}_1$	23.66	23.46	-	-	-	0.721	27.23	25.12	-	-	-	0.769
$\mathbf{T}_2$	23.66	23.45	23.49	-	-	0.738	27.23	25.12	24.70	-	-	0.765
$\mathbf{T}_3$	23.66	23.42	23.44	24.03	-	0.747	27.23	25.10	24.65	25.25	-	0.758
$\mathbf{T}_4$	23.66	23.40	23.38	23.97	24.59	0.757	27.23	25.08	24.61	25.19	24.66	0.741

Moment	Mac						
	PSNR $\uparrow$						SSIM $\uparrow$
	$\mathbf{T}_0$	$\mathbf{T}_1$	$\mathbf{T}_2$	$\mathbf{T}_3$	$\mathbf{T}_4$	$\mathbf{T}_5$	
$\mathbf{T}_0$	32.77	-	-	-	-	-	0.956
$\mathbf{T}_1$	32.77	30.13	-	-	-	-	0.943
$\mathbf{T}_2$	32.77	30.08	29.07	-	-	-	0.922
$\mathbf{T}_3$	32.77	29.92	28.86	29.07	-	-	0.930
$\mathbf{T}_4$	32.77	29.88	28.66	28.94	30.90	-	0.936
$\mathbf{T}_5$	32.77	29.80	28.67	28.91	30.81	29.64	0.928

Table 6. The results of each scene at each moment.

References

[1] Zhipeng Cai and Matthias Müller. Clnerf: Continual learning meets nerf. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23185–23194, 2023. 1, 3

[2] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: benchmarking large-scale scene reconstruction. *ACM Trans. Graph.*, 36(4):78:1–78:13, 2017. 3

[3] Jiakai Sun, Han Jiao, Guangyuan Li, Zhanjie Zhang, Lei Zhao, and Wei Xing. 3dgstream: On-the-fly training of 3d gaussians for efficient streaming of photo-realistic free-viewpoint videos. *arXiv preprint arXiv:2403.01444*, 2024. 1, 3

[4] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20310–20320, 2024. 1