3DGStream: On-the-Fly Training of 3D Gaussians for Efficient Streaming of Photo-Realistic Free-Viewpoint Videos

Supplementary Material

1. Implementation Details

We implement 3DGStream upon the codebase of 3D Gaussian Splatting (3DG-S) [4] and use tiny-cuda-nn [7] to implement Neural Transformation Cache (NTC). All experiments were conducted on an NVIDIA RTX 3090 GPU. In training of the initial frame, we let the densification of 3DG-S end at iteration 5000. For the scenes in the N3DV dataset, we use the 3DGs of iteration 15000 as the initial 3DGs, while for the scenes in the Meet Room dataset, we use the results of iteration 10000. For convenience, we set the maximum degree of spherical harmonics (SH) to 1, and all other hyperparameters are consistent with the 3DG-S.

Training NTC. We set the learning rate of NTC to 0.002. For the scenes in the N3DV dataset [6], the hash table size of the multi-resolution hash encoding is 2^{15}, the feature vector dimension is 4, and there are 16 resolution levels. For the Meet Room dataset [5], the hash table length is 2^{14}, with all other hyperparameters matching those specified for the N3DV dataset. For all scenes, our fully-fused MLP comprises 2 hidden layers with 64 neurons each, employing ReLu as the activation function. Given that the N3DV dataset and the Meet Room dataset both record indoor dynamic scenes, and multi-resolution hash encoding requires normalized coordinates for input, we create an axis-aligned bounding box that roughly encloses the house to normalize the 3D points and discard any points outside the bounding box to prevent distant landscapes from influencing the training.

Training the additional 3DGs. Compared to training on the initial frame, we increase the learning rate in the second stage for faster convergence. Specifically, the learning rates for the mean, SH coefficient, opacity value, scaling vector, and rotation quaternion of the 3DGs are set to 0.0024, 0.0375, 0.75, 0.075, and 0.015, respectively. Note that these learning rates were not individually fine-tuned; instead, their proportions are following the default settings of 3DG-S.

2. SH Rotation

In order to preserve theoretical soundness, we also rotate the SH after transforming the 3DGs. The zeroth-degree SH does not require rotation; therefore, we only need to rotate the first-degree SH coefficients.

We utilize the projection function [10] to project normal vectors onto the first-order SH. Given a rotation matrix R, we seek a matrix M that can rotate the first-degree SH. Because rotating a vector before projecting it to SH produces the same outcome as projecting the vector first and then rotating the SH, we have the following relationship:

$$MP(N) = P(RN), \tag{1}$$

where N is a normal vector. For any three normal vectors N_0, N_1, and N_2 we denote $A = [P(N_0), P(N_1), P(N_2)]$. Consequently, we obtain:

$$MA = [P(RN_0), P(RN_1), P(RN_2)]. \tag{2}$$

And hence:

$$M = [P(RN_0), P(RN_1), P(RN_2)]A^{-1} \tag{3}$$

For computational convenience, we choose $N_0 = [1, 0, 0]^T, N_1 = [0, 1, 0]^T$, and $N_2 = [0, 0, 1]^T$.

Method	Coffee Martini	Cook Spinach	Cut Beef	Flame Salmon	Flame Steak	Sear Steak	Mean
Plenoxels [3]†	27.65	31.73	32.01	28.68	32.24	32.33	30.77
I-NGP [8]†	25.19	29.84	30.73	25.51	30.04	30.40	28.62
3DG-S [4]†	27.78	34.10	34.03	28.66	34.41	33.48	32.08
DyNeRF [6]	–	–	–	29.58	–	–	29.58
NeRFPlayer [11]	31.53	30.58	29.35	31.65	31.93	29.13	30.69
HexPlane [2]	–	32.04	32.55	29.47	32.08	32.39	31.70
K-Planes [9]	29.99	32.60	31.82	30.44	32.38	32.52	31.63
HyperReel [1]	28.37	32.30	32.92	28.26	32.20	32.57	31.10
MixVoxels [12]	29.36	31.61	31.30	29.92	31.43	31.21	30.80
StreamRF [5]†	27.84	31.59	31.81	28.26	32.24	32.36	28.26
Ours	27.75	33.31	33.21	28.42	34.30	33.01	31.67

Table 1. **Quantitative comparison** of PSNR values across all scenes in the N3DV dataset, with the metric for each scene calculated as the average over 300 frames. †Obtained in our own experiments with the official codes.

Method	Coffee Martini	Cook Spinach	Cut Beef	Flame Salmon	Flame Steak	Sear Steak	Mean
Baseline	27.68	33.19	33.10	28.39	33.54	32.79	31.45
Full Model	27.75	33.31	33.21	28.42	34.30	33.01	31.67

Table 2. **Ablation Study** of the Adaptive 3DG Addition strategy across all scenes in the N3DV dataset, with the metric for each scene calculated as the average over 300 frames. We take PSNR to measure the image quality.

3. More Results

3.1. Quantitative Results

We provide a quantitative comparison of image quality, measured by PSNR, across all scenes in the N3DV dataset

Step	Overhead (ms)	FPS
Render *w/o* NTC	1.75	571
+ Query NTC	+0.46	
+ Transformation	+0.02	
+ SH Rotation	+1.24	
Total	3.47	288

Table 3. **Rendering profilling** on the Meet Room dataset.

Dataset	NTC (KB)	New 3DGs (KB)	Total (KB)
N3DV	7781.5	49.1	7830.6
MeetRoom	3941.5	195.3	4136.8

Table 4. **Detailed "Storage" entry of our method** in Tabs. 1 and 2.

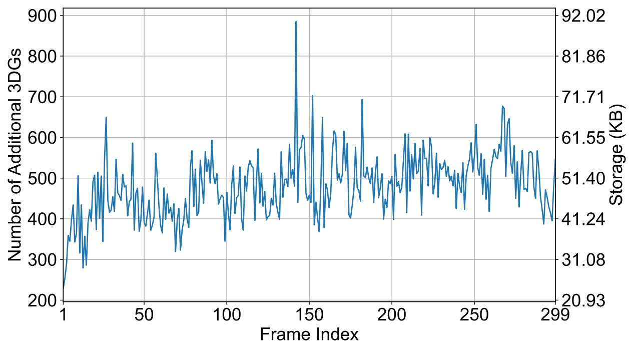

Figure 1. **Number of additional 3DGs and corresponding storage requirement** of each frame on the *flame salmon* scene.

in Tab. 1. Furthermore, we provide the quantitative result of the ablation study across all scenes in the N3DV dataset in Tab. 2, Additionally, we provide rendering profilling on the Meet Room Dataset in Tab. 3.

3.2. Qualitative Results

We provide videos to show the free view synthesis results on various scenes from the N3DV dataset in `https://sjojok.github.io/3dgstream`.

4. More Evaluations

4.1. Storage Requirements

Except the initial 3DGs, we only need to store per-frame NTCs and per-frame additional 3DGs for each FFV frame, as detailed in Tab. 4.

4.2. Quantity of 3DGs

In our experiments on the N3DV datasets, the quantity of initial 3DGs (*i.e.*, the transformed ones) is on the order of 10^5, while the quantity of frame-specific additional 3DGs is on the order of 10^2. We show how the number of the frame-specific additional 3DGs changes as the frame number increases in Fig. 1.

Scene	Stage 1		Stage 2	
	150	250	100	200
Flame Salmon	28.39	28.44	28.46	28.46
Flame Steak	33.54	33.81	34.44	34.46
Sear Steak	32.79	33.02	33.18	33.19
Cook Spinach	33.19	33.50	33.56	33.57
Cut Roasted Beef	33.10	33.39	33.44	33.44
Coffee Martini	27.68	27.77	27.83	27.83

Table 5. **Evaluation on the impact of training iterations** conducted on the N3DV dataset. The result of Stage 2 is is obtained after 250 iterations of optimization at Stage 1. We take PSNR to measure the image quality.

4.3. Impact of Training Iterations

In the main text, we discuss the trade-off between training efficiency and reconstruction quality, noting that limiting the number of training iterations enables efficient on-the-fly training at the expense of reduced quality. To show this trade-off, we conduct experiments to evaluate the impact of training iterations, and show the quantitative results in Tab. 5. As shown in Tab. 5, increasing training iterations in Stage 1 significantly enhances the reconstruction quality. However, an additional 100 iterations result in an increment of 3 seconds in the per-frame training duration. Incrementing training iterations in the second stage has a minimal impact on quality, which can be attributed to the higher learning rate employed in this phase and the smaller number of additional 3DGs, facilitating rapid convergence.

References

[1] Benjamin Attal, Jia-Bin Huang, Christian Richardt, Michael Zollhoefer, Johannes Kopf, Matthew O'Toole, and Changil Kim. Hyperreel: High-fidelity 6-dof video with ray-conditioned sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16610–16620, 2023. 1

[2] Ang Cao and Justin Johnson. Hexplane: A fast representation for dynamic scenes. *CVPR*, 2023. 1

[3] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510, 2022. 1

[4] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4), 2023. 1

[5] Lingzhi Li, Zhen Shen, Zhongshu Wang, Li Shen, and Ping

Tan. Streaming radiance fields for 3d video synthesis. In *NeurIPS*, 2022. 1

[6] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, et al. Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5521–5531, 2022. 1

[7] Thomas Müller. tiny-cuda-nn, 2021. 1

[8] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, 2022. 1

[9] Sara Fridovich-Keil and Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *CVPR*, 2023. 1

[10] Peter-Pike Sloan. Stupid spherical harmonics (sh) tricks. In *Game developers conference*, page 42, 2008. 1

[11] Liangchen Song, Anpei Chen, Zhong Li, Zhang Chen, Lele Chen, Junsong Yuan, Yi Xu, and Andreas Geiger. Nerfplayer: A streamable dynamic scene representation with decomposed neural radiance fields. *IEEE Transactions on Visualization and Computer Graphics*, 29(5):2732–2742, 2023. 1

[12] Feng Wang, Sinan Tan, Xinghang Li, Zeyue Tian, Yafei Song, and Huaping Liu. Mixed neural voxels for fast multi-view video synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19706–19716, 2023. 1