BAD-Gaussians: Bundle Adjusted Deblur Gaussian Splatting

Supplementary Materials

A Introduction

In this supplementary material, we present the derivation of the analytical jacobian of the Gaussians w.r.t the camera poses and some additional qualitative and quantitative evaluation of our BAD-Gaussians.

B Jacobian of the Gaussians w.r.t the Camera Poses

In

$$\frac{\partial \mathcal{L}}{\partial \mathbf{T}_i} \coloneqq \underbrace{\sum_{k=0}^{K-1} \frac{\partial \mathcal{L}}{\partial \mathbf{B}_k} \cdot \frac{1}{n} \sum_{i=0}^{n-1} \frac{\partial \mathbf{B}_k}{\partial \mathbf{C}_i}}_{\text{auto-diff}} \frac{\partial \mathbf{C}_i}{\partial \boldsymbol{\theta}} \frac{\partial \boldsymbol{\theta}}{\partial \mathbf{T}_i}, \tag{1}$$

we have

$$\frac{\partial \boldsymbol{\theta}}{\partial \mathbf{T}_i} = \left[\frac{\partial \boldsymbol{\mu}'}{\partial \mathbf{T}_i} \cancel{\frac{\partial \boldsymbol{\Sigma}'}{\partial \mathbf{T}_i}} \cancel{\frac{\partial \mathbf{c}}{\partial \mathbf{T}_i}} \cancel{\frac{\partial \mathbf{o}}{\partial \mathbf{T}_i}} \right], \tag{2}$$

where color $\mathbf{c}$ and opacity $\mathbf{o}$ of the Gaussian are independent with the virtual camera pose $\mathbf{T}_i = \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix} \in \mathbf{SE}(3)$ of the i^{th} virtual sharp image $\mathbf{C}_i$. Also, following GS-SLAM [12], we ignore $\frac{\partial \boldsymbol{\Sigma}'}{\partial \mathbf{T}_i}$ for efficiency.

As for $\frac{\partial \boldsymbol{\mu}'}{\partial \mathbf{T}_i}$, since the motion-blurred image synthesis is implemented in PyTorch [8], the first part of the gradient in Eq. (1) can be computed by the auto-diff module of PyTorch. The remaining part of the gradient in Eq. (1) can be computed as follows:

$$\frac{\partial \mathbf{C}_i}{\partial \boldsymbol{\mu}'} \frac{\partial \boldsymbol{\mu}'}{\partial \mathbf{T}_i} = \frac{\partial \mathbf{C}_i}{\partial \boldsymbol{\mu}'} \frac{\partial \boldsymbol{\mu}'}{\partial \boldsymbol{\mu}} \frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\mu}_c} \frac{\partial \boldsymbol{\mu}_c}{\partial \mathbf{T}_i}, \tag{3}$$

where $\boldsymbol{\mu}_c = \mathbf{R}\boldsymbol{\mu} + \mathbf{t}$ represents $\boldsymbol{\mu}$ transformed into the camera's coordinate space.

The first term $\frac{\partial \mathbf{C}_i}{\partial \boldsymbol{\mu}'} \frac{\partial \boldsymbol{\mu}'}{\partial \boldsymbol{\mu}} = \frac{\partial \mathbf{C}_i}{\partial \boldsymbol{\mu}}$ is the Jacobian w.r.t. the mean position of the Gaussians, which is already computed in the CUDA backend of the diffenrentiable projection and rasterization. The second term can be simplified as follows:

$$\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\mu}_c} = \frac{\partial \boldsymbol{\mu}}{\partial \mathbf{R}\boldsymbol{\mu} + \mathbf{t}} = \mathbf{R}^\top; \tag{4}$$

And the last term:

$$\frac{\partial \boldsymbol{\mu}_c}{\partial \mathbf{T}_i} = \begin{bmatrix} \boldsymbol{\mu}_c^\top & 1 \end{bmatrix} \otimes \mathbf{I}_3 \in \mathbb{R}^{3 \times 12}, \tag{5}$$

Table A: Full quantitative comparisons for **novel view synthesis** on the real captured dataset of Deblur-NeRF [6].

	Ball			Basket			Buick			Coffee			Decoration		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
NeRF [7]	24.08	.6237	.3992	23.72	.7086	.3223	21.59	.6325	.3502	26.48	.8064	.2896	22.39	.6609	.3633
3D-GS [2]	23.72	.6321	.3210	23.96	.7466	.2600	21.53	.6630	.2743	27.44	.6630	.2208	22.29	.6826	.2868
DB-NeRF [6]	27.15	.7641	.2112	27.35	.8367	.1347	24.93	.7791	.1545	30.72	.8949	.1070	24.15	.7730	.1700
DP-NeRF [3]	26.86	.7522	.2066	27.71	.8434	.1301	25.52	.7847	.1433	31.35	.9013	.0987	24.42	.7820	.1611
BAD-NeRF [11]	26.71	.7480	.2120	26.40	.8024	.1268	23.06	.7104	.2099	29.08	.8401	.1941	22.09	.6067	.3079
ExBluRF [4]	25.82	.7228	.2359	25.64	.8130	.1495	24.27	.7486	.1719	27.59	.8519	.1417	21.30	.6125	.3074
Ours	28.11	.8041	.2058	27.41	.8632	.0987	24.22	.8064	.1161	32.17	.9280	.0659	25.53	.8442	.0937

	Girl			Heron			Parterre			Puppet			Stair		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
NeRF [7]	20.07	.7075	.3196	20.50	.5217	.4129	23.14	.6201	.4046	22.09	.6093	.3389	22.87	.4561	.4868
3D-GS [2]	19.97	.7276	.2613	20.28	.5254	.3109	22.98	.6326	.2967	22.38	.6463	.2645	22.68	.4709	.3911
DB-NeRF [6]	22.27	.7976	.1687	22.63	.6874	.2099	25.82	.7597	.2161	25.24	.7510	.1577	25.39	.6296	.2102
DP-NeRF [3]	23.43	.8148	.1459	22.79	.7010	.1891	25.90	.7658	.1893	25.56	.7560	.1469	25.53	.6326	.1778
BAD-NeRF [11]	19.72	.6194	.3378	21.81	.6249	.2340	24.86	.7066	.2131	24.14	.7073	.1833	25.64	.6370	.1768
ExBluRF [4]	19.35	.6923	.2748	21.10	.6091	.3049	23.14	.6794	.2471	22.41	.6639	.2006	23.51	.5446	.2508
Ours	21.28	.8152	.1040	24.52	.7657	.1187	25.94	.8133	.0983	25.25	.7991	.0948	26.63	.7177	.0685

where $\otimes$ is the Kronecker operator and $\mathbf{I}_3$ is a 3×3 identity matrix [1] [13].

Finally, note that the virtual camera pose $\mathbf{T}_i$ is interpolated from the control knots of the $\mathbf{SE}(3)$ coutinuous trajectory, e.g. $\mathbf{T}_{\text{start}}, \mathbf{T}_{\text{end}} \in \mathbf{SE}(3)$ for linear interpolation, and $\mathbf{T}_1, \mathbf{T}_2, \mathbf{T}_3, \mathbf{T}_4 \in \mathbf{SE}(3)$ for cubic B-spline. We use PyPose [10] to implement the interpolations, thus the corresponding Jacobian of $\mathbf{T}_i$ w.r.t. the pose adjustments (the actual parameters being optimized), e.g. $\boldsymbol{\varepsilon}_{\text{start}}, \boldsymbol{\varepsilon}_{\text{end}} \in \mathfrak{se}(3)$ for linear interpolation and $\boldsymbol{\varepsilon}_1, \boldsymbol{\varepsilon}_2, \boldsymbol{\varepsilon}_3, \boldsymbol{\varepsilon}_4 \in \mathfrak{se}(3)$ for cubic B-spline, are handled by auto-diff of PyTorch [8].

C Experimental Details

C.1 Full Table of Novel View Synthesis Results on Real Datasets

In this section, we present full quantitative results of novel view synthesis on real datasets from Deblur-NeRF [6]. The results are shown in Table A. The results demonstrate that our method outperforms the state-of-the-art methods in terms of PSNR, SSIM, and LPIPS.

C.2 Full Table of Ablations on Trajectory Representations

The full results of our ablation study on trajectory representations are presented in Table B. The results demonstrate that linear interpolation adequately represents the camera motion trajectory for synthetic datasets, such as *MBA-VO* and *Deblur-NeRF-Synthetic*. However, cubic B-spline outperforms linear interpolation in real data scenarios (i.e. *Deblur-NeRF-Real*), attributed to the extended exposure time.

Table B: Ablation studies on the effect of trajectory representations. The results demonstrate that cubic interpolation improves performance in scenes with complex camera trajectories (i.e. *MBA-VO* and *Deblur-NeRF-Real*).

Dataset	Sequence	Linear Interpolation			Cubic B-spline		
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Deblur-NeRF Synthetic [6]	Cozy2room	34.68	.9521	.0258	33.74	.9446	.0346
	Factory	31.88	.9270	.0952	31.87	.9324	.0897
	Pool	36.95	.9434	.0225	34.77	.9107	.0440
	Tanabata	32.12	.9481	.0464	32.09	.9477	.0464
	Trolley	33.97	.9628	.0209	33.73	.9619	.0220
MBA-VO [5]	ArchViz-low	32.28	.9167	.1134	32.43	.9165	.1118
	ArchViz-high	29.64	.8568	.1847	29.68	.8601	.1789
Deblur-NeRF Real [6]	Ball	23.10	.6423	.2778	28.11	.8041	.2078
	Basket	27.03	.8564	.0998	27.41	.8632	.0987
	Buick	23.44	.7939	.1118	24.22	.8064	.1161
	Coffee	30.52	.9017	.0913	32.17	.9280	.0659
	Decoration	24.99	.8305	.0992	25.53	.8442	.0937
	Girl	21.23	.8028	.1449	21.28	.8152	.1040
	Heron	21.70	.6929	.1495	24.52	.7657	.1187
	Parterre	25.03	.7817	.1173	25.94	.8133	.0983
	Puppet	24.78	.7811	.1021	25.25	.7991	.0948
	Stair	25.87	.6944	.0975	26.63	.7177	.0685

C.3 Details of Ablations on the Number of Virtual Cameras

In our ablation study on the number of virtual cameras n, for a fair comparison, we make the number of the densified Gaussians roughly the same by adjusting the threshold of the gradient in densification with n. This is based on the following derivation: In Eq. (1), during the synthesis of motion-blurred image, the gradient of every Gaussian is scaled by $\frac{1}{n}$. Therefore, if we change n to n', the densification threshold should be multiplied by $\frac{n}{n'}$, in order to match the scaled gradients.

D Additional Qualitative Evaluation

We provide further qualitative experimental results on both the synthetic and real datasets, showcased in Fig. A and Fig. B respectively. These results demonstrate the superior performance of our method over previous state-of-the-art approaches.

D.1 Visualization of Pose Estimation Results

In this section, we present the visualization results in terms of camera pose estimation. The experiments are conducted on the synthetic dataset of Deblur-NeRF [6]. We present the comparison result of BAD-Gaussians against COLMAP [9] and BAD-NeRF [11] in Fig. C. It demonstrates that our method recovers motion trajectories more accurately.

D.2 Video of Novel View Synthesis Results

To showcase the effectiveness of our approach, we provide supplementary videos illustrating the capability of BAD-Gaussians to recover high-quality latent sharp video from blurry images. The videos contain results on both synthetic and real scenes from Deblur-NeRF [6]. In the provided videos, on the left are our rendered novel view images and on the right are the input blurry images.

Notably, in the provided videos, due to the fast training speed and low GPU memory requirements of our method, we are able to train real scenes at the native resolution 2400×1600 to achieve maximum reconstruction quality in about 1.5 hours, compared to the resolution of 600×400 that we used in all experiments above in this paper for a fair comparison.

References

1. Blanco, J.L.: A tutorial on SE(3) transformation parameterizations and on-manifold optimization. Tech. Rep. 012010, University of Malaga (2010), http://ingmec.ual.es/~jlblanco/papers/jlblanco2010geometry3D_techrep.pdf 2
2. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3D Gaussian Splatting for Real-Time Radiance Field Rendering. ACM Transactions on Graphics (TOG) **42**(4) (July 2023), https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/ 2, 6, 7
3. Lee, D., Lee, M., Shin, C., Lee, S.: DP-NeRF: Deblurred Neural Radiance Field With Physical Scene Priors. In: Computer Vision and Pattern Recognition (CVPR) (2023), https://dogyoonlee.github.io/dpnerf/ 2, 6, 7
4. Lee, D., Oh, J., Rim, J., Cho, S., Lee, K.M.: ExBluRF: Efficient Radiance Fields for Extreme Motion Blurred Images. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 17639–17648 (2023) 2
5. Liu, P., Zuo, X., Larsson, V., Pollefeys, M.: MBA-VO: Motion Blur Aware Visual Odometry. In: International Conference on Computer Vision (ICCV) (2021), https://github.com/ethliup/MBA-VO 3, 6
6. Ma, L., Li, X., Liao, J., Zhang, Q., Wang, X., Wang, J., Sander, P.V.: Deblur-NeRF: Neural Radiance Fields from Blurry Images. In: Computer Vision and Pattern Recognition (CVPR) (2022), https://limacv.github.io/deblurnerf/ 2, 3, 4, 6, 7
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: European Conference on Computer Vision (ECCV) (2020), https://www.matthewtancik.com/nerf 2
8. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An Imperative Style, High-performance Deep Learning Library. Advances in neural information processing systems **32** (2019), https://pytorch.org/ 1, 2
9. Schonberger, J.L., Frahm, J.M.: Structure-from-motion Revisited. In: Computer Vision and Pattern Recognition (CVPR) (2016), https://github.com/colmap/colmap 3, 8
10. Wang, C., Gao, D., Xu, K., Geng, J., Hu, Y., Qiu, Y., Li, B., Yang, F., Moon, B., Pandey, A., Aryan, Xu, J., Wu, T., He, H., Huang, D., Ren, Z., Zhao, S., Fu,

T., Reddy, P., Lin, X., Wang, W., Shi, J., Talak, R., Cao, K., Du, Y., Wang, H., Yu, H., Wang, S., Chen, S., Kashyap, A., Bandaru, R., Dantu, K., Wu, J., Xie, L., Carlone, L., Hutter, M., Scherer, S.: PyPose: A library for robot learning with physics-based optimization. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2023), https://github.com/pypose/pypose 2

11. Wang, P., Zhao, L., Ma, R., Liu, P.: BAD-NeRF: Bundle Adjusted Deblur Neural Radiance Fields. In: Computer Vision and Pattern Recognition (CVPR) (2023), https://wangpeng000.github.io/BAD-NeRF/ 2, 3, 6, 7, 8
12. Yan, C., Qu, D., Wang, D., Xu, D., Wang, Z., Zhao, B., Li, X.: GS-SLAM: Dense Visual SLAM with 3D Gaussian Splatting. arXiv preprint arXiv:2311.11700 (2023) 1
13. Ye, V., Kanazawa, A.: Mathematical supplement for the gsplat library. arXiv preprint arXiv:2312.02121 (2023), https://github.com/nerfstudio-project/gsplat 2

Fig. A: Qualitative deblurring results of different methods with synthetic datasets from MBA-VO [5] and Deblur-NeRF [6]. The scenes, from left to right, encompass *ArchViz-high*, *Cozy2room*, *Factory*, and *Trolley*. Despite being trained with ground truth poses (*), BAD-Gaussians outperforms Deblur-NeRF* and DP-NeRF* in recovering high-quality scenes from motion-blurred images with inaccurate camera poses, showcasing its superior performance.

Fig. B: Qualitative novel view synthesis results of different methods with the real datasets from Deblur-NeRF [6]. The scenes, from left to right, encompass *Basket*, *Coffee*, *Girl*, and *Stair*. The experimental results demonstrate that our method achieves superior performance over prior methods on the real dataset as well.

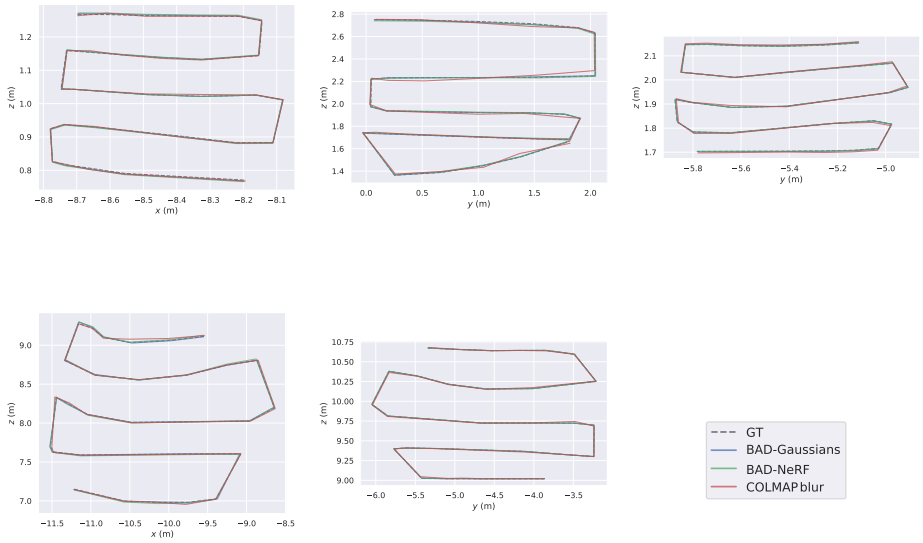

Fig. C: Qualitative Comparisons of estimated camera poses on Deblur-NeRF dataset. These are results on *Cozy2room*, *Factory*, *Pool*, *Tanabata* and *Trolley* sequences respectively. The results demonstrate that our method recovers motion trajectories more accurately compared with both COLMAP [9] and BAD-NeRF [11].