Supplementary Material for: Disentangled Generation and Aggregation for Robust Radiance Fields

Shihe Shen[1⋆], Huachen Gao[1⋆], Wangze Xu[1], Rui Peng[1,2], Luyang Tang[1,2], Kaiqiang Xiong[1,2], Jianbo Jiao[3], and Ronggang Wang[1,2,✉]

[1] School of Electronic and Computer Engineering, Peking University
[2] Peng Cheng Laboratory
[3] School of Computer Science, University of Birmingham
{shshen0308, gaohuachen712}@gmail.com rgwang@pkusz.edu.cn

A Detailed Architecture of Triplane Generator

As we mentioned in the previous chapter, the shape of triplane noise tokens is set to $(3 \times 8 \times 20 \times 20)$, where 3 represents the number of feature planes, 8 represents the hidden dimension and 20×20 is the spatial shape. The triplane tokens are reshaped to $(3 \times 400 \times 8)$, and each plane $(1 \times 400 \times 8)$ is applied as the query to do cross-attention with the extracted DINO feature tokens $(1 \times 3889 \times 384)$ separately. The attended tokens are then reshaped back to $(3 \times 8 \times 20 \times 20)$ for the input of the triplane generator. The generator for each plane is composed of one mid-block and L up-sample-blocks ($L = 5$). The mid-block comprises two 2D convolution layers with residual connections (res-conv-layers) and one attention layer. A group normalization and a SiLU activation follow each res-conv layer. For the up-sample-blocks, we similarly adopt two res-conv-layers followed by group normalization and SiLU activation. Then, a bilinear upsampler is appended in each block, except for the last one. The upsample layers expand the spatial size of noised tokens $(3 \times 8 \times 20 \times 20)$ to the shape of final triplane grids $(3 \times 64 \times 320 \times 320)$. Taking one feature plane $\mathcal{P}_{XY}$ for instance, the noise tokens t_{XY} with size of $(d_t \times r_X \times r_Y)$ are lifted from d_t to D_P in channels and upscaled $(L-1)$ times in the spatial dimensions to the final plane's shape $(D_P \times R_X \times R_Y)$, where $r_X = 20, r_Y = 20, d_t = 8, D_P = 64, R_X = 2^{L-1} \times r_X$ and $R_Y = 2^{L-1} \times r_Y$.

B Detailed Derivation Formulation in DPA

Commonly used aggregations introduced entanglement to pose with triplane features. One operation is Hadamard product used in [4]. On account of the multiplicative nature, when gradient update is unstable in the early steps, product among planes causes violent vibration on pose optimization and may bring interference into gradients back propagated from triplane feature, further resulting

⋆ Equal contribution.

in updating collision on pose. However, in forward-facing scenes, angles of images or videos are generally consistent, with all objects primarily facing towards cameras. Since scenes emphasize the front view of objects, the frontal features gain prime focus and accordingly only one or two planes may receive a good optimization signal. Meanwhile, unlike the relatively independent update of planes, the parameters of each pose receive optimization signals from all different planes, as the 3D points for feature query are obtained by sampling rays from the corresponding camera.

To disentangle pose with planes, we design our DPA, which is formulated as:

$$\begin{aligned}\mathbf{DPA}(\mathcal{P}, \mathbf{x}) = &\prod_k \psi\left(\mathcal{P}_k, \mathcal{D}(\pi_k(\mathbf{x}))\right) + \sum_m \psi(\mathcal{D}(\mathcal{P}_m), \pi_m(\mathbf{x})) \\ &+ \sum_{k\neq m}^{k,m} \psi(\mathcal{D}(\mathcal{P}_k), \mathcal{D}(\pi_k(\mathbf{x})))\psi(\mathcal{D}(\mathcal{P}_m), \mathcal{D}(\pi_m(\mathbf{x}))) + 1,\end{aligned} \tag{1}$$

where $\mathcal{D}(x)$ is the gradient-detached copy of x and $\frac{\partial \mathcal{D}(x)}{\partial x}$ is *zero*. Thus, denote that the feature obtained from the proposed aggregation $\hat{F} = \mathbf{DPA}(\mathcal{P}, \mathbf{x})$, the core part $\frac{\partial \hat{F}}{\partial \mathbf{x}}$ of pose updating gradients can be expressed as:

$$\begin{aligned}\frac{\partial \hat{F}}{\partial \mathbf{x}} &= \frac{\partial \prod_k \psi\left(\mathcal{P}_k, \mathcal{D}(\pi_k(\mathbf{x}))\right)}{\partial \mathbf{x}} + \frac{\partial \sum_m \psi(\mathcal{D}(\mathcal{P}_m), \pi_m(\mathbf{x}))}{\partial \mathbf{x}} \\ &+ \frac{\partial \sum_{k\neq m}^{k,m} \psi(\mathcal{D}(\mathcal{P}_k), \mathcal{D}(\pi_k(\mathbf{x})))\psi(\mathcal{D}(\mathcal{P}_m), \mathcal{D}(\pi_m(\mathbf{x})))}{\partial \mathbf{x}} + \frac{1}{\partial \mathbf{x}} \\ &= \sum_m \left(\frac{\partial \psi\left(\mathcal{P}_m, \mathcal{D}(\pi_m(\mathbf{x}))\right)}{\partial \mathbf{x}} \cdot \prod_{k\neq m} \psi\left(\mathcal{P}_k, \mathcal{D}(\pi_k(\mathbf{x}))\right) \right) \\ &+ \frac{\partial \sum_m \psi(\mathcal{D}(\mathcal{P}_m), \pi_m(\mathbf{x}))}{\partial \mathbf{x}} \\ &+ \frac{\partial \sum_{k\neq m}^{k,m} \psi(\mathcal{D}(\mathcal{P}_k), \mathcal{D}(\pi_k(\mathbf{x})))\psi(\mathcal{D}(\mathcal{P}_m), \mathcal{D}(\pi_m(\mathbf{x})))}{\partial \mathbf{x}} + \frac{1}{\partial \mathbf{x}}.\end{aligned} \tag{2}$$

Since $\frac{\partial \mathcal{D}(x)}{\partial x}$ is *zero* and $\frac{\partial \psi(\mathcal{P}, \mathcal{D}(x))}{\partial x}$ is *zero*, we get the final expanded expression of $\frac{\partial \hat{F}}{\partial \mathbf{x}}$ as:

$$\frac{\partial \hat{F}}{\partial \mathbf{x}} = \sum_m \frac{\partial \psi\left(\mathcal{D}(\mathcal{P}_m), \pi_m(\mathbf{x})\right)}{\partial \mathbf{x}}, \tag{3}$$

where the gradients back propagated to pose from different planes are combined with sum, which is beneficial to pose optimization. Meanwhile, with the gradients for plane XY as:

$$\frac{\partial \hat{F}}{\partial \mathcal{P}_{XY}} = \frac{\partial \prod_k \psi\left(\mathcal{P}_k, \mathcal{D}(\pi_k(\mathbf{x}))\right)}{\partial \mathcal{P}_{XY}}, \tag{4}$$

and so as the other two planes YZ, XZ, our triplane features preserve the expressive ability for scene representation.

C More Experimental Details

Proposal Sampling and Density Field. We use a proposal sampling strategy for 3D point sampling and implement it similarly to the one in K-Planes [4], which is a more compact variant from the proposal sampling strategy in Mip-NeRF 360 [1]. K-Planes designed a density model with a triplane structure similar to its feature model, and trained it with histogram loss. We borrow the two-stage proposal sampling and basic density models from K-Planes and histogram loss from Mip-NeRF 360 but bond them with the pose-scene joint optimization. Therefore, our density field are forged and updated by another triplane generator.

Datasets. We conduct experiments on two datasets: LLFF [8] and NeRF-synthetic dataset [9]. The LLFF [8] is a real-world dataset consisting of eight forward-facing scenes captured by mobile phones. The NeRF-Synthetic dataset [9] contains pathtraced images of eight objects that exhibit complex geometry and non-Lambertian material.

Implementation Details. To achieve the joint optimization of camera poses and triplane, we follow the architecture of K-Planes [4] with some modifications for pose refinement and thus make the joint pose-triplane optimization baseline. Assuming known camera intrinsics, we follow [7, 11] to parameterize camera extrinsics as learnable variables $T \in SE(3)$, where the rotations are optimized in axis-angle $\phi_i \in \mathfrak{so}(3)$.

In the first stage, we utilize two separate Adam optimizers to independently optimize the triplane generator and camera poses. Specifically, the learning rate for the generator linearly increases from 0 to 0.002 in the first 128 steps of training and decreases with cosine-annealing until the second stage. The learning rate for the camera pose is set to 0.001. We switch to the second stage of learning after 4000 steps with our proposed warm-start strategy.

In the second stage, we discard the triplane generator and set the generated triplane as learnable variables, utilizing a new Adam optimizer for optimization. The learning rates for triplane and camera parameters linearly increase from 0 to 0.03 and 0.001 respectively within the first 128 steps from the second stage, ensuring a smooth transition to the second-stage direct optimization approach. Similar to the first stage, these learning rates decrease exponentially to 1×10^{-5} in the remaining steps.

In both stages, we randomly sample 4096-pixel rays at each optimization step. Following [4], we employ proposal sampling to sample 48 points along each ray for subsequent volume rendering. For forward-facing LLFF, we utilize normalized device coordinates (NDC) to better allocate our resolution and enable unbounded depth. During the evaluation, we follow [7] to run additional steps of test-time learning on the frozen trained models and only optimize poses for testing during this procedure. Our model is implemented with PyTorch and trained 60k (for NeRF-synthetic [9] and 70k for LLFF [8]) epochs per scene on an NVIDIA Tesla V100 GPU.

Evaluation Criteria. For novel view synthesis evaluation, we first conduct test-time optimization on each testing pose as proposed in [7] to eliminate minor errors caused by the misalignment of the test phase and training phase camera poses. We report PSNR, SSIM [10] for novel view synthesis evaluation. For camera pose evaluation, we follow previous works [2, 7] to perform Procrustes analysis for aligning the training poses and the GT poses before calculating the rotation error (in degree) and translation error (scaled by 100).

D More Experimental Results

Additional Qualitative Results of Novel View Synthesis. We provide additional novel view synthesis comparisons as shown in Fig. 1. Since the implementation of GARF [3] and HASH [5] are unavailable, we directly use the results reported in their paper for comparison.

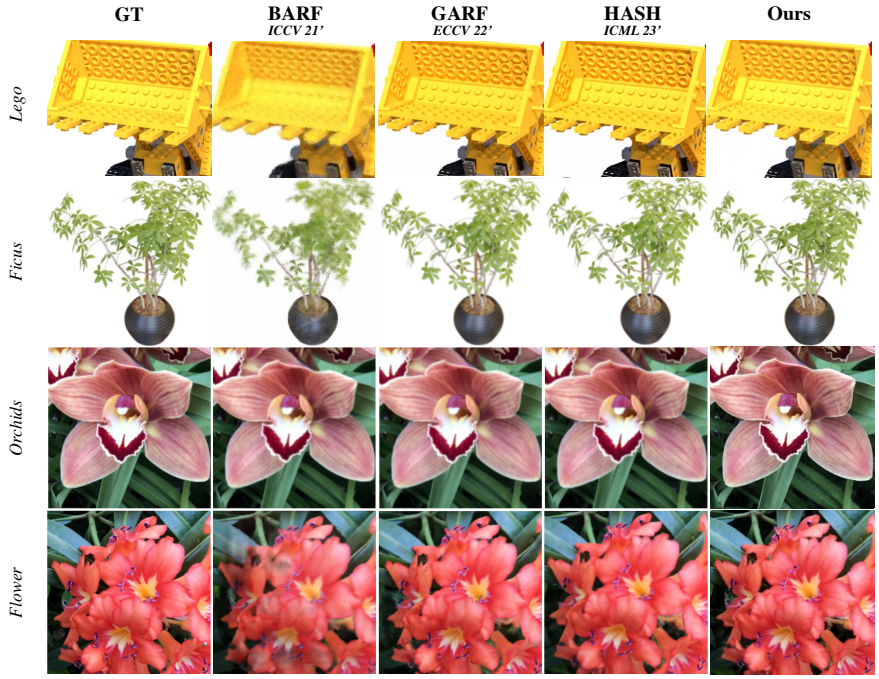

Fig. 1: Additional qualitative results of novel view synthesis.

Ablation on Scene Texture Embedding. As shown in Sec. D, we additionally perform an ablation of the scene texture embedding module in the triplane generator. The results show a degradation in the quality of the novel view

rendering after removing this module from our model. This demonstrates that our Scene Texture Embedding module introduces more scene texture prior for triplane generation, thus enhancing the triplane representation.

Table 1: Ablations on applying the Scene Texture Embedding in the triplane generator in real-world LLFF dataset [8].

Settings	PSNR↑	SSIM↑
Ours with Scene Texture Embedding	25.90	0.828
Ours without Scene Texture Embedding	25.35	0.813

Besides, we further provide results across varied scenes in Table below, the proposed STE improves a minimum of 0.18 db on *Lego* and a maximum of 1.23 db on *Fortress*, indicating consistent improvements.

Table 2: More detailed ablations on the Scene Texture Embedding module.

	Fern	Fortress	Room	Flower	Lego	Drums
Ours without STE	24.68	29.56	32.84	26.67	32.01	25.73
Ours	25.70 (+**1.02**)	30.79 (+**1.23**)	33.95 (+**1.11**)	27.06 (+**0.39**)	32.19 (+**0.18**)	26.10 (+**0.37**)

More Empirical Experiments on Inappropriate Learning Signals from Different Feature Grids. We further provide empirical experiments from *Lego* as shown in Fig. 2. The 'forward-facing-like' cameras are gathered within a limited view angle towards PlaneXZ and the 'surrounded-like' set almost covers the whole scene. Triplane receives less supervision (PlaneXY and PlaneYZ) in 'forward-facing-like' scenes with more noisy and incomplete feature textures.

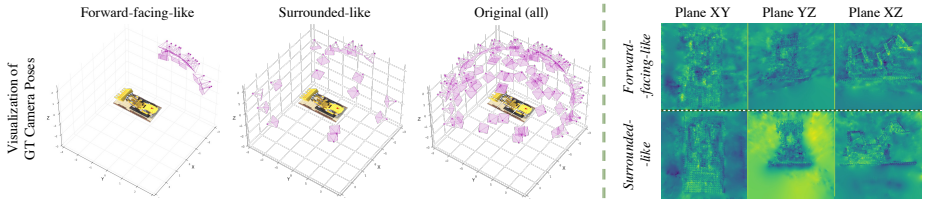

Fig. 2: Visualization of camera distribution and plane features.

Additional Ablations on Two-Stage Warm-Start Training. We perform a two-stage system ablation experiment on the challenging scene *orchids* by

switching from the first stage to the second stage at different training steps as shown in Fig. 3. We can see our two-stage strategy (orange) stably improves the performance regardless of the quality of the first stage (blue).

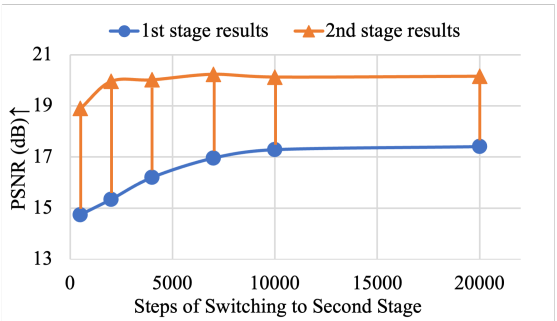

Fig. 3: Comparisons of different steps of switching to second stage.

Comparison of Utilizing Noise Tokens and Image Tokens as Input. We initialize fixed triplane noise tokens to introduce *spatial priors*. We provide more comparison as shown in Tab. 3 and Fig. 4.

Table 3: Qualitative comparison between using image tokens and noise tokens as input.

Scenes	Room	Fern	Lego	Chair
Image Tokens Only	31.90	24.48	30.07	34.27
Ours	**33.95**	**25.70**	**32.19**	**37.78**

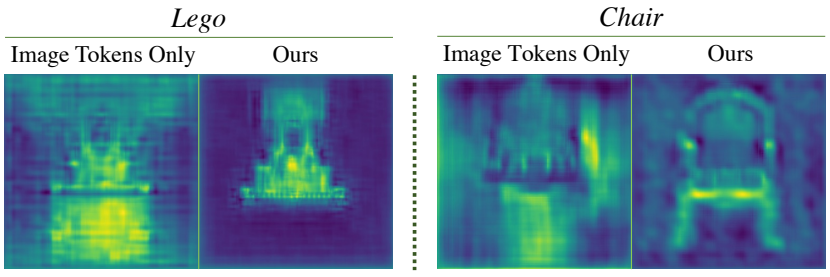

Fig. 4: Visual comparisons of feature plane between only image feature tokens inputs and our full inputs with noise tokens at 1000 training steps.

Visual Comparisons with Baseline Joint Estimation. We provide additional visualization comparisons as shown in Fig. 5 to illustrate the effectiveness of our approach. Compared to the baseline simple combination of pose estimation and triplane-NeRF optimization, our approach achieves higher-quality visual effects, which demonstrates that our proposed method mitigates the errors caused by local updating and entanglement, leading to better pose estimation and novel view rendering results.

Limitations and Future Works. Although our method can recover the camera pose and triplane radiance fields effectively, there is still room for improvement. In the future, we will explore introducing more powerful 3D representations such as 3D Gaussian Splatting [6] into the joint optimization pipeline and try to further reduce the dependence on camera pose initialization.

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 (CVPR). pp. 5470–5479 (June 2022)
2. Chen, Y., Chen, X., Wang, X., Zhang, Q., Guo, Y., Shan, Y., Wang, F.: Local-to-global registration for bundle-adjusting neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8264–8273 (2023)
3. Chng, S.F., Ramasinghe, S., Sherrah, J., Lucey, S.: Gaussian activated neural radiance fields for high fidelity reconstruction and pose estimation. In: European Conference on Computer Vision. pp. 264–280. Springer (2022)
4. Fridovich-Keil, S., Meanti, G., Warburg, F.R., Recht, B., Kanazawa, A.: K-planes: Explicit radiance fields in space, time, and appearance. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 12479–12488 (June 2023)
5. Heo, H., Kim, T., Lee, J., Lee, J., Kim, S., Kim, H.J., Kim, J.H.: Robust camera pose refinement for multi-resolution hash encoding. In: Proceedings of the 40th International Conference on Machine Learning (2023)
6. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**(4) (2023)
7. Lin, C.H., Ma, W.C., Torralba, A., Lucey, S.: Barf: Bundle-adjusting neural radiance fields. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV). pp. 5721–5731. IEEE Computer Society (2021)
8. Mildenhall, B., Srinivasan, P.P., Ortiz-Cayon, R., Kalantari, N.K., Ramamoorthi, R., Ng, R., Kar, A.: Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. ACM Transactions on Graphics (TOG) (2019)
9. 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. pp. 405–421. Springer (2020)
10. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing **13**(4), 600–612 (2004)

11. Wang, Z., Wu, S., Xie, W., Chen, M., Prisacariu, V.A.: NeRF−−: Neural radiance fields without known camera parameters. arXiv preprint arXiv:2102.07064 (2021)

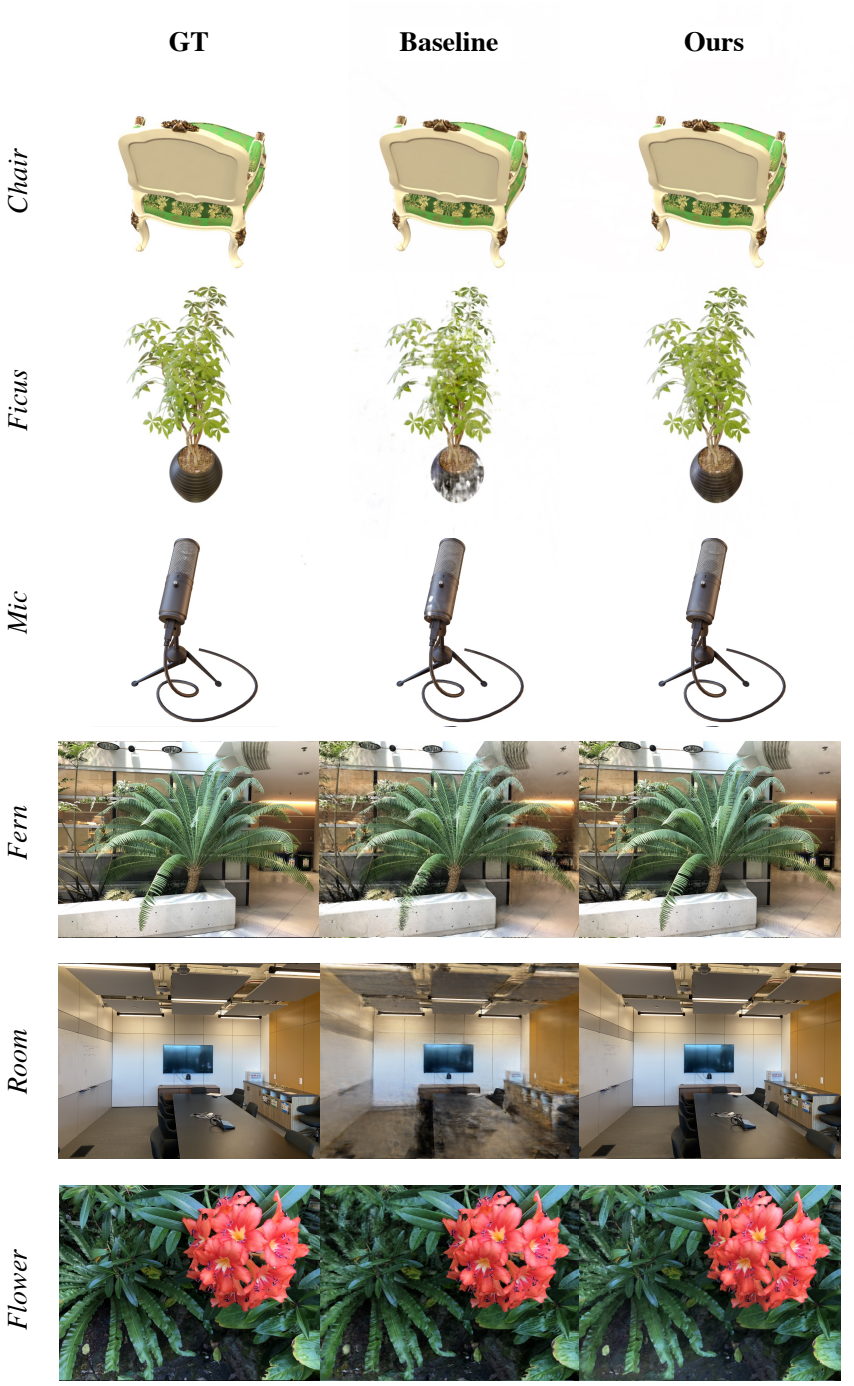

Fig. 5: Visual comparisons between our full model and the baseline direct joint pose-triplane optimization.