Splat-based Gradient-domain Fusion for Seamless View Transition

Supplementary Material

1. Mesh Reconstruction Details

We render depth maps $\hat{D}$ from each training camera $\mathbf{P}$ using Gaussian Splatting and reconstruct a mesh via depth back-projection. In the mesh construction, adjacent pixels along both vertical and horizontal directions are connected to form triangles. However, back-projecting all depths produces excessively large triangles in regions of sharp depth discontinuity (referred to as *depth edge* in the main text). These regions incorrectly fill empty space, causing significant inconsistencies with the global geometry. To address this, we detect pixels with abrupt depth changes based on the area of the meshed triangle as:

$$\frac{(\hat{D} - \nabla_x \hat{D}) \times (\hat{D} - \nabla_y \hat{D})}{\|\hat{D}\|^2} < \tau_{\text{depth}}, \tag{1}$$

and exclude them from back-projection during mesh generation. Here, ∇_x and ∇_y denote the depth values of the neighboring pixel one step along the x and y axes. For all experiments, we set $\tau_{\text{depth}} = 1e-4$. Using this strategy, we obtain a clean and well-formed mesh.

2. Point Cloud Reconstruction Details

We convert disparity maps obtained from a stereo model [10] into depth to generate dense depth point clouds $\mathbf{X}$ for Gaussian initialization. To prevent the generation of erroneous points at depth edge, we apply the same thresholding strategy used in mesh reconstruction. Unlike mesh reconstruction, we do not generate faces and simply back-project the valid depth points into world space.

Predicted disparities with extremely small values can result in abnormally large depth points, causing Gaussians to grow excessively to fill the space. This also hinders the creation of new Gaussians at precise locations. To address this issue, we adopt a contraction function inspired by Mip-NeRF 360 [1] with two boundaries, B_{near} and B_{far}. We define the contraction function as:

$$\text{Contract}(\mathbf{X}) = \begin{cases} \mathbf{X} - \mathbf{T}, & \text{if } d \leq B_{\text{near}}, \\ (\mathbf{X} - \mathbf{T}) \cdot \left(\frac{B_{\text{far}} - B_{\text{near}}}{d}\right), & \text{if } d > B_{\text{near}} \end{cases} \tag{2}$$

where $d = \|\mathbf{X} - \mathbf{T}\|_2$ is the distance from the depth point $\mathbf{X}$ to the reference camera of contraction $\mathbf{T}$. B_{near} is determined as the largest depth value within the top 10% of depths across all training views, and B_{far} is set to twice this value. The camera containing B_{near} is identified as $\mathbf{T}$, which serves as the center for the contraction process.

Through confidence-based hierarchical point cloud reconstruction, we progressively accumulate high-confidence points, resulting in a more stable and reliable point cloud compared to directly using all stereo depth points simultaneously. In stereo pairs, depths predicted in regions without stereo coverage often resemble those obtained from monocular depth estimation models. While depths farther than those predicted by two-view stereo generally do not pose significant issues, closer depths can lead to a degradation in overall quality (Figure 1). Therefore, our hierarchical approach to reconstructing the point cloud enhances the stability and robustness of our method.

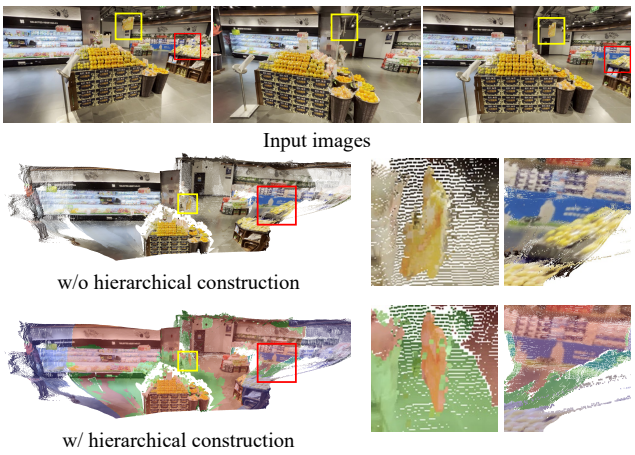


Figure 1. **Ablation over hierarchical point cloud reconstruction.** Our point cloud reconstruction approach enhances the multi-view consistency. Color coding is the same as in the main paper Figure 4: red ($\mathcal{R}^{\text{c}}$) for pairwise consistent areas, green ($\mathcal{R}^{\text{o}}$) for overlap regions with occlusion or mismatching, and blue ($\mathcal{R}^{\text{s}}$) for single-view areas.

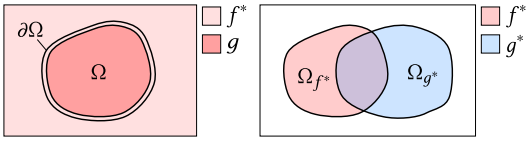
3. Additional Implementation Details

We use Adam to optimize all Gaussian attributes. The position learning rate is configured to decay exponentially from 0.00016 to 0.0000016. The diffuse colors are optimized with a fixed learning rate of 0.0025, while the spherical harmonics coefficients are trained with a learning rate of 0.000125. For other parameters, we set the learning rates for opacity, scaling, and rotation to 0.05, 0.005, and 0.001, respectively.

4. Gradient-domain Fusion Comparison Details

Here, we provide a more detailed explanation of each method introduced in Section 5.3.

(a) For Gaussian initialization, we sample only 1% of the points from the hierarchical point cloud reconstruction (compared to 10% in our method). We aim to reduce complexity and achieve smoother rendered results by decreasing the number of appearance functions used in



(a) Poisson blending (b) Our occlusion-aware Poisson blending

Figure 2. Standard Poisson blending vs. occlusion-aware Poisson blending. Our occlusion-aware Poisson blending extends standard Poisson blending, which transfers the source image g^* within a single mask Ω onto the target image f^*, by enabling seamless blending between two images with distinct masks Ω_{f^*} and Ω_{g^*} that each vary in their boundary constraint suitability.

Splatting.

(b) Every 100 iterations, we use a k-NN algorithm to identify the 50 nearest neighbors for each Gaussian and apply a L2 regularizer that aligns its color component with the neighbors' average. The weight of this regularizer is set to 10 throughout training. Empirically, varying this weight from 10 to 1000 did not produce any noticeable change during view transition.

(c) We render the virtual views using Gaussian Splatting in the same manner as our method and apply L1 TV regularization to encourage local consistency in the rendered images, with the regularizer weight set to 1.

(d) While prior methods minimize the difference between the gradient fields of two reference images and the rendered image, we use an L1 loss to directly enforce the rendered virtual view to match the reference image. The regularizer weight is set to the same value as in our method.

(e) Two reprojected adjacent input images are blended using Poisson blending to create a pseudo ground-truth image, which is then used to supervise the rendered image via an L1 loss. The regularizer weight is set to the same value as in the prior method.

Standard Poisson blending typically transfers the gradients of a source image to a target image. In this case, however, both images act as sources and targets simultaneously, making the boundary conditions more complex as illustrated in Figure 2. To address this, we use an occlusion-aware blending approach to seamlessly combine two images f^* and g^* with their corresponding masks Ω_{f^*} and Ω_{g^*} by adapting the boundary conditions. When only one of the masks is valid at a given pixel, the pixel value is used as the boundary condition. For pixels where both masks are valid, we compute a guidance field $\mathbf{w}$ as the weighted sum of the gradients from both images. The Poisson equation is formulated as:

$$\begin{aligned} \Delta f = \nabla \cdot \mathbf{w} \quad \text{over} \quad (\Omega_{f^*} \cap \Omega_{g^*}), \\ f|_{(\Omega_{f^*} \cap \Omega^c_{g^*})} = f^*|_{(\Omega_{f^*} \cap \Omega^c_{g^*})}, \\ f|_{(\Omega_{g^*} \cap \Omega^c_{f^*})} = g^*|_{(\Omega_{g^*} \cap \Omega^c_{f^*})}. \end{aligned} \tag{3}$$

We blend two reprojected input images by solving this equation via sparse matrix solver.

5. User Study Details

To assess the naturalness of color transitions and overall perceptual quality, we conduct two user studies:

- **User study (A):** Comparison among FreeNeRF [12], CoherentGS [5], MVPGS [11], and ours.
- **User study (B):** Comparison of alternatives replacing only the GF in Section 5.3 (a), (b), (d), (e), including ours.

Each study involves 18 participants and is conducted on a Samsung UN46ES7100F 46-inch FHD monitor, using 8-second, 20 fps novel view synthesis videos from *Tanks and Temples* datasets. The user interface, shown in Figure 3, presents two videos side-by-side, and participants answer the following questions using the two-alternative forced choice (2AFC) paradigm in a pairwise comparison setting to ensure consistent and unbiased evaluation:

Q1: (**Natural color transition**) Which video shows more natural color transitions as the viewpoint changes?

Q2: (**Overall quality**) Which video has higher overall quality as the viewpoint changes?

The scene, method pairing, and left–right order are randomized, and all participants complete the study using identical hardware under the same controlled environment. Each participant completes 48 comparisons in **(A)** and 80 in **(B)**, covering all pair combinations, taking approximately 40 minutes in total. As a result, we obtain 144 responses for each method pair in both studies **(A)** and **(B)**.

We employ a Bayesian Bradley–Terry model to estimate the overall preference ranking of all methods based on pairwise comparison data. In the model, each method's skill value, representing its overall preference strength, follows a normal prior distribution $\mathcal{N}(0, 2)$ at the initial state, and the observed wins and losses are modeled using a Bernoulli likelihood. Comparisons are indexed by winners and losers, and MCMC sampling is performed using PyMC with 4,000 draws, 4,000 tuning steps with a target acceptance rate of 0.95. Posterior skill values are then used to compute win probabilities, which are visualized to show the preference ranking among methods.

6. Additional Results

We include the full pairwise win rates and p-values for each method in User Studies **(A)** and **(B)** in Table 1 and Table 2, respectively. We also provide additional results for further comparison. In Figure 5, we compare our method with NeRF-based methods [8, 9, 12], and include results from MVSplat [3] in Figure 4. MVSplat, which focuses on generalization, shows lower quality compared to per-scene optimization methods. In Figure 6, we present the error map compared to MVPGS [11], analyzing the limitation that, while our method excels qualitatively with sparse inputs, it struggles to achieve high quantitative performance. Figures 7 and 8 show the full-shot results of GS-based methods.

Table 1. Pairwise win rates of all baseline methods.

FreeNeRF [12]	**Q1 Win (%)**	**Q1 p-value**	**Q2 Win (%)**	**Q2 p-value**
vs. CoherentGS [5]	28.47	<0.0001	24.31	<0.0001
vs. MVPGS [11]	7.64	<0.0001	1.39	<0.0001
vs. Ours	1.39	<0.0001	0	<0.0001
CoherentGS [5]	**Q1 Win (%)**	**Q1 p-value**	**Q2 Win (%)**	**Q2 p-value**
vs. FreeNeRF [12]	71.53	<0.0001	75.69	<0.0001
vs. MVPGS [11]	29.17	<0.0001	25.69	<0.0001
vs. Ours	11.81	<0.0001	9.03	<0.0001
MVPGS [11]	**Q1 Win (%)**	**Q1 p-value**	**Q2 Win (%)**	**Q2 p-value**
vs. FreeNeRF [12]	92.36	<0.0001	98.61	<0.0001
vs. CoherentGS [5]	70.83	<0.0001	74.31	<0.0001
vs. Ours	12.50	<0.0001	10.42	<0.0001
Ours	**Q1 Win (%)**	**Q1 p-value**	**Q2 Win (%)**	**Q2 p-value**
vs. FreeNeRF [12]	98.61	<0.0001	100	<0.0001
vs. CoherentGS [5]	88.19	<0.0001	90.97	<0.0001
vs. MVPGS [11]	87.50	<0.0001	89.58	<0.0001

Table 2. Pairwise win rates of all gradient fusion alternatives.

(a) Sparse init.	**Q1 Win (%)**	**Q1 p-value**	**Q2 Win (%)**	**Q2 p-value**
vs. (b) k-NN color cons.	72.22	<0.0001	71.53	<0.0001
vs. (d) L1 loss on vir.	54.17	0.359	59.03	0.037
vs. (e) Poisson-blended	54.86	0.279	63.89	0.001
vs. (f) GF (ours)	11.11	<0.0001	11.11	<0.0001
(b) k-NN color cons	**Q1 Win (%)**	**Q1 p-value**	**Q2 Win (%)**	**Q2 p-value**
vs. (a) Sparse init.	27.78	<0.0001	28.47	<0.0001
vs. (d) L1 loss on vir.	42.36	0.08	52.08	0.677
vs. (e) Poisson-blended	38.19	0.006	47.22	0.56
vs. (f) GF (ours)	7.64	<0.0001	12.50	<0.0001
(d) L1 loss on vir.	**Q1 Win (%)**	**Q1 p-value**	**Q2 Win (%)**	**Q2 p-value**
vs. (a) Sparse init.	45.83	0.359	40.97	0.037
vs. (b) k-NN color cons.	57.64	0.08	47.92	0.677
vs. (e) Poisson-blended	49.31	0.934	52.78	0.56
vs. (f) GF (ours)	9.72	<0.0001	9.03	<0.0001
(e) Poisson-blended	**Q1 Win (%)**	**Q1 p-value**	**Q2 Win (%)**	**Q2 p-value**
vs. (a) Sparse init.	45.14	0.279	36.11	0.0001
vs. (b) k-NN color cons.	61.81	0.006	52.78	0.56
vs. (d) L1 loss on vir.	50.69	0.934	47.22	0.56
vs. (f) GF (ours)	10.42	<0.0001	7.64	<0.0001
(f) GF (ours)	**Q1 Win (%)**	**Q1 p-value**	**Q2 Win (%)**	**Q2 p-value**
vs. (a) Sparse init.	88.89	<0.0001	88.89	<0.0001
vs. (b) k-NN color cons.	92.36	<0.0001	87.50	<0.0001
vs. (d) L1 loss on vir.	90.28	<0.0001	90.97	<0.0001
vs. (e) Poisson-blended	89.58	<0.0001	92.36	<0.0001

Finally, Figure 9 displays the results of Poisson blending used in GF alternatives (e).

7. Comparison Details

To compare our results, we use the official implementations of FreeNeRF [12], FlipNeRF [8], SparseNeRF [9], MVSplat [3], DNGaussian [4], CoherentGS [5], and SCGaussian [6], and MVPGS [11]. The baseline methods use the default hyperparameters, including the number of iterations.

References

[1] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. 2022. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proc. IEEE/CVF CVPR 2022*. 5470–5479.

[2] Peter J Burt and Edward H Adelson. 1983. A multiresolution spline with application to image mosaics. *ACM Transactions on Graphics (TOG)* 2, 4 (1983), 217–236.

[3] Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. 2025. Mvsplat: Efficient 3d gaussian splatting from sparse multi-view images. In *Proc. ECCV 2025*. 370–386.

[4] Jiahe Li, Jiawei Zhang, Xiao Bai, Jin Zheng, Xin Ning, Jun Zhou, and Lin Gu. 2024. Dngaussian: Optimizing sparse-view 3d gaussian radiance fields with global-local depth normalization. In *Proc. IEEE/CVF CVPR 2024*. 20775–20785.

[5] Avinash Paliwal, Wei Ye, Jinhui Xiong, Dmytro Kotovenko, Rakesh Ranjan, Vikas Chandra, and Nima Khademi Kalantari. 2025. Coherentgs: Sparse novel view synthesis with coherent 3d gaussians. In *Proc. ECCV 2025*. Springer, 19–37.

[6] Rui Peng, Wangze Xu, Luyang Tang, Liwei Liao, Jianbo Jiao, and Ronggang Wang. 2024. Structure Consistent Gaussian Splatting with Matching Prior for Few-shot Novel View Synthesis. In *Proc. NeurIPS 2024*.

[7] Patrick Pérez, Michel Gangnet, and Andrew Blake. 2003. Poisson Image Editing. *ACM Transactions on Graphics* 22, 3 (2003), 313–318.

[8] Seunghyeon Seo, Yeonjin Chang, and Nojun Kwak. 2023. Flipnerf: Flipped reflection rays for few-shot novel view synthesis. In *Proc. IEEE/CVF CVPR 2023*. 22883–22893.

[9] Guangcong Wang, Zhaoxi Chen, Chen Change Loy, and Ziwei Liu. 2023. Sparsenerf: Distilling depth ranking for few-shot novel view synthesis. In *Proc. IEEE/CVF CVPR 2023*. 9065–9076.

[10] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, Fisher Yu, Dacheng Tao, and Andreas Geiger. 2023. Unifying flow, stereo and depth estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2023).

[11] Wangze Xu, Huachen Gao, Shihe Shen, Rui Peng, Jianbo Jiao, and Ronggang Wang. 2025. MVPGS: Excavating Multi-view Priors for Gaussian Splatting from Sparse Input Views. In *Proc. ECCV 2025*. Springer, 203–220.

[12] Jiawei Yang, Marco Pavone, and Yue Wang. 2023. Freenerf: Improving few-shot neural rendering with free frequency regularization. In *Proc. IEEE/CVF CVPR 2023*. 8254–8263.

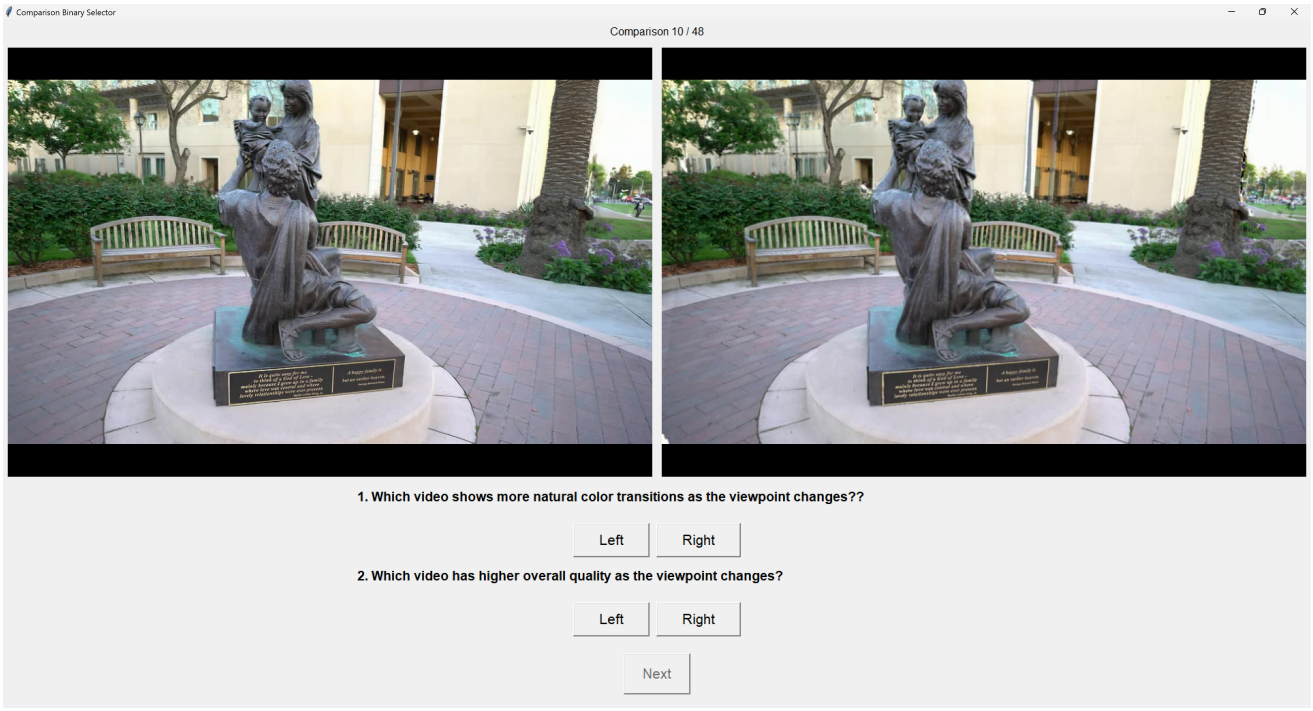


Figure 3. User study GUI screen. We conduct a user study in which participants choose between two anonymized videos in each trial. The videos are randomly assigned to the left and right displays, and both the scenes and methods are randomized across trials. The interface displays the total and current comparison counts at the top center. All participants complete the study on identical hardware under a controlled environment.

MVSplat Ours

Figure 4. Comparison of ours with MVSplat. When Gaussian primitives are predicted solely in MVSplat [3] using a pre-trained feed-forward model, severe distortions occur in novel views.

Figure 5. Qualitative comparison with NeRF-based methods on the DL3DV-10K dataset. Novel views reconstructed from three input images. While NeRF-based methods that rely on implicit representations often generate blurry and noisy outputs, our method produces sharp and high-quality results.

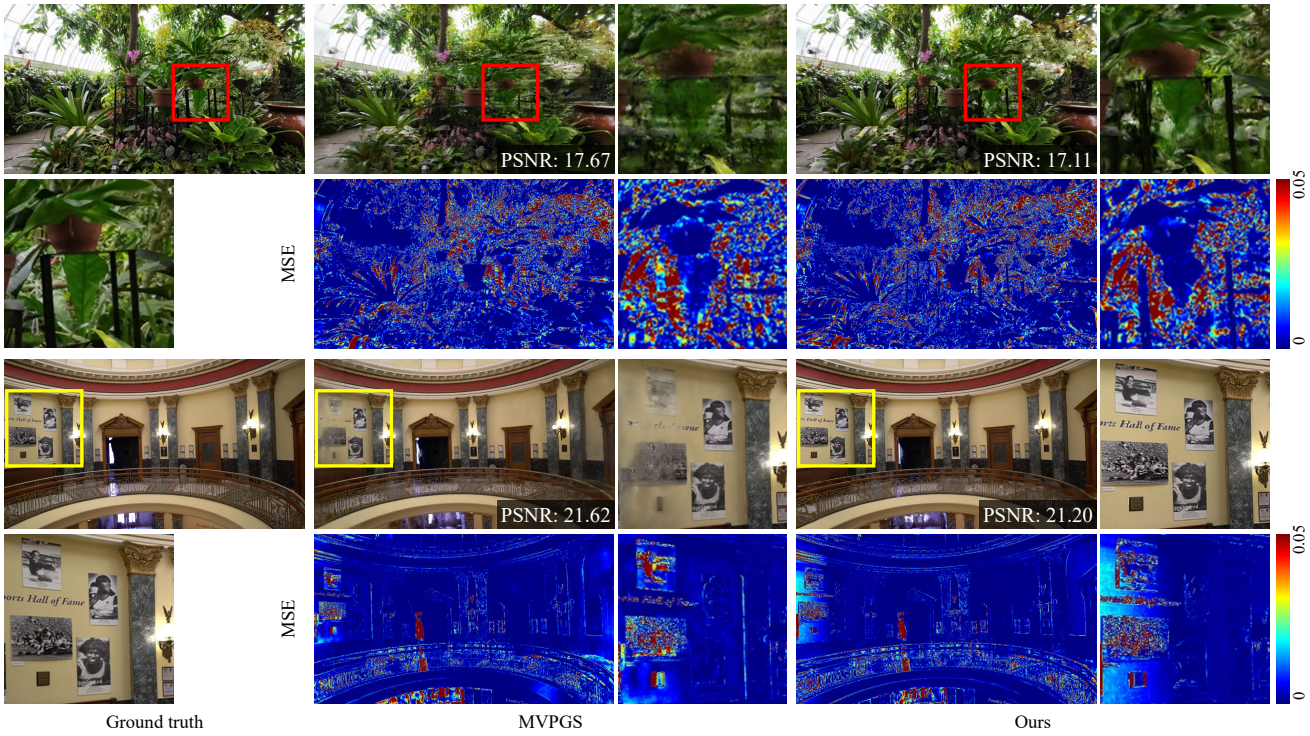
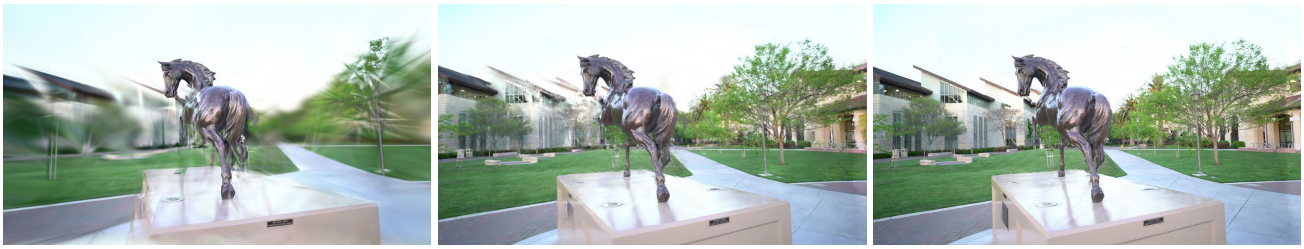


Figure 6. Error map comparison with MVPGS. Inset PSNR is calculated based on a single ground truth image and the rendered results. In the first and second rows, when the ground truth image is highly complex, our method successfully produces visually and perceptually natural results; however, the overall error is large, resulting in a lower PSNR compared to baseline methods. In the third and fourth rows, where view-dependent color variations are significant, our method captures smooth view-dependent colors while accurately representing fine details. Nevertheless, pixel-wise metrics like PSNR sometimes demonstrate better performance when the model overfits to the training views. These challenges underscore the difficulty in achieving high quantitative performance, despite our method excelling qualitatively in sparse-view scenarios.

Ground truth DNGaussian CoherentGS

SCGaussian MVPGS Ours

Ground truth DNGaussian CoherentGS

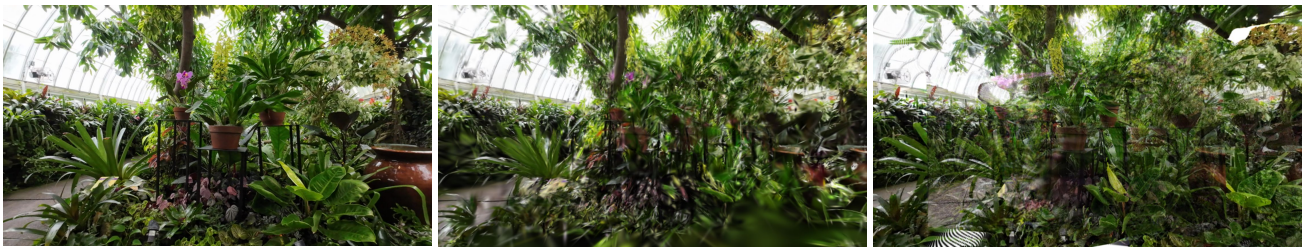


SCGaussian MVPGS Ours

Ground truth DNGaussian CoherentGS

SCGaussian MVPGS Ours

Figure 7. Additional comparison of GS-based methods on Tank and Temples dataset.



Ground truth DNGaussian CoherentGS

SCGaussian MVPGS Ours

Ground truth DNGaussian CoherentGS

SCGaussian MVPGS Ours

Ground truth DNGaussian CoherentGS

SCGaussian MVPGS Ours

Figure 8. Additional comparison of GS-based methods on DL3DV-10K dataset.

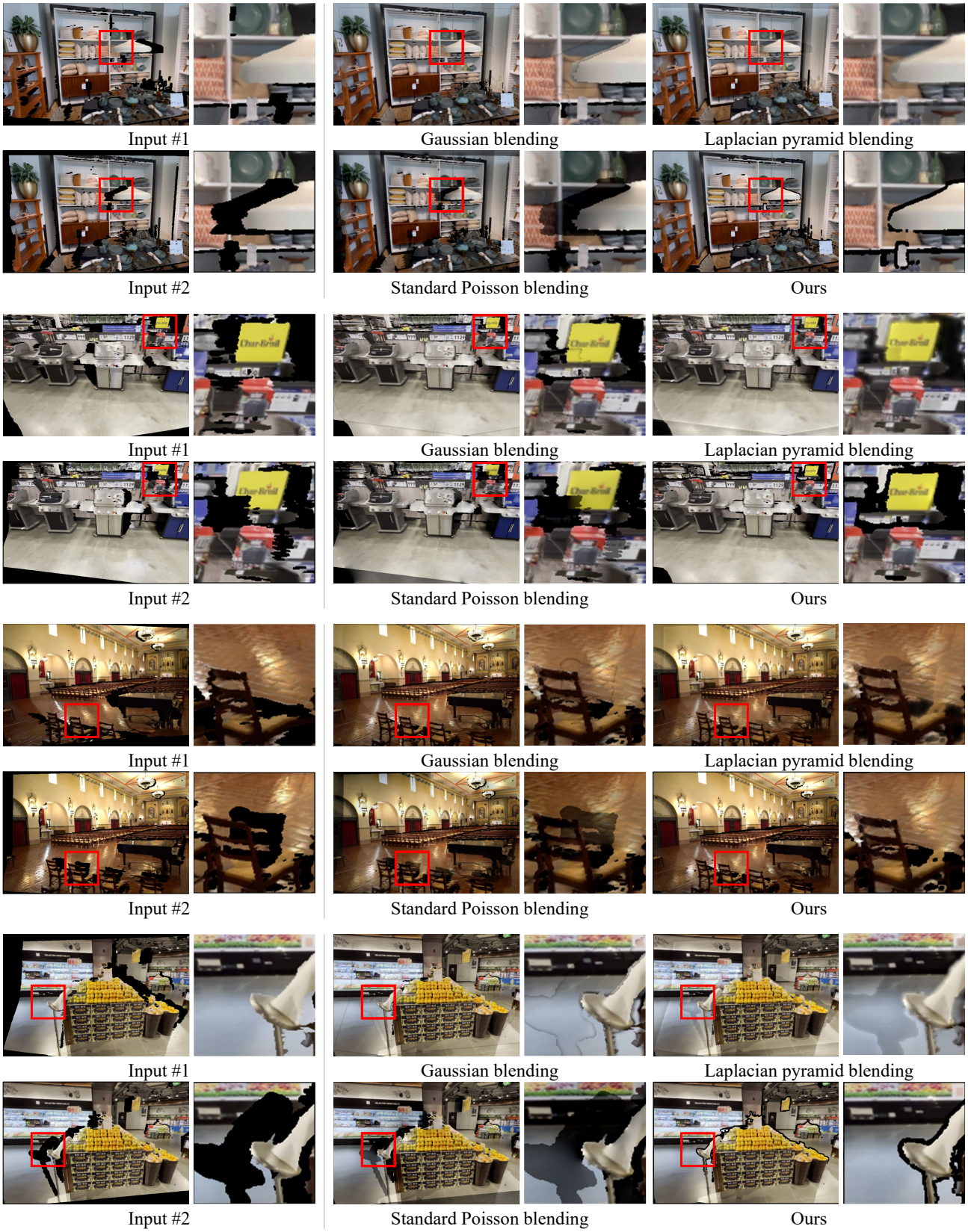


Figure 9. Additional comparison results of blending methods within the GS framework: standard Gaussian blending, Laplacian pyramid blending [2], standard Poisson blending [7], and our occlusion-aware Poisson blending used in Section 5.3 (e).