AuraFusion360: Augmented Unseen Region Alignment for Reference-based 360° Unbounded Scene Inpainting

Supplementary Material

A. Overview

This supplementary material provides additional details and results to support the main manuscript. We first describe the training process for masked Gaussians and object removal in Section B, followed by an explanation of depth warping for bounding box generation in SAM2 [9] and its role in identifying unseen region contours in Section C. Next, we present ablations on different depth inpainting methods in Section D and a comparison of captured and inpainted references in Section E. We then outline the experimental setup in Section F and discuss the limitations of our approach in Section G. Finally, we provide additional visual comparisons in Fig. 4 for the 360-UISD dataset and in Fig. 5 for the other collected 360 dataset [1].

B. Training Masked GS for Object Removal

During the training of masked Gaussians, we use 2DGS [4] as our codebase and introduce a masked attribute, ranging between 0 and 1, for each Gaussian. The L1 loss is computed between the object mask obtained via SAM2 [9] and the rasterized object mask for each training view. Additionally, we incorporate the Grouping Loss proposed by Gaussian Grouping [12], ensuring that neighboring Gaussians have similar masked attributes. This ensures that our Gaussian model retains accurate object mask information and is capable of rendering precise object masks for subsequent applications.

Thanks to the explicit nature of Gaussian Splatting, we can directly remove Gaussians with a masked attribute greater than a threshold τ during the removal stage, effectively achieving object removal. In our implementation, τ is set to 0.6.

C. Depth Warping for Unseen Contours

Following Sec. 3.2 and Fig. 4 of the main paper, we explain in detail how depth warping allows us to identify the contours of the unseen region, as illustrated in Fig. 1. Without loss of generality, to find the unseen region contour at view n, and for each pair of views n and i, we first compute the removal region for view i by identifying pixels that differ between the rendered depth and the incomplete depth of view i rather than using object masks. This approach better captures geometric changes and prevents misalignment artifacts, leading to improved SAM2[9] prompts and more precise unseen masks (Fig. 2).

Next, we establish pixel correspondences between view n and view i using the incomplete depth of view n. The removal region of view i is then backward-traversed to view n based on these correspondences. During this backward traversal, it is important to note that pixels outside the unseen region in view i will correspond to the background areas in view n, while pixels belonging to the unseen region remain in the unseen region. By aggregating contributions from all views i ($i \neq n$), we project non-unseen regions from each view i into different areas of view n, while consolidating the unseen regions. This allows us to identify the contours of the

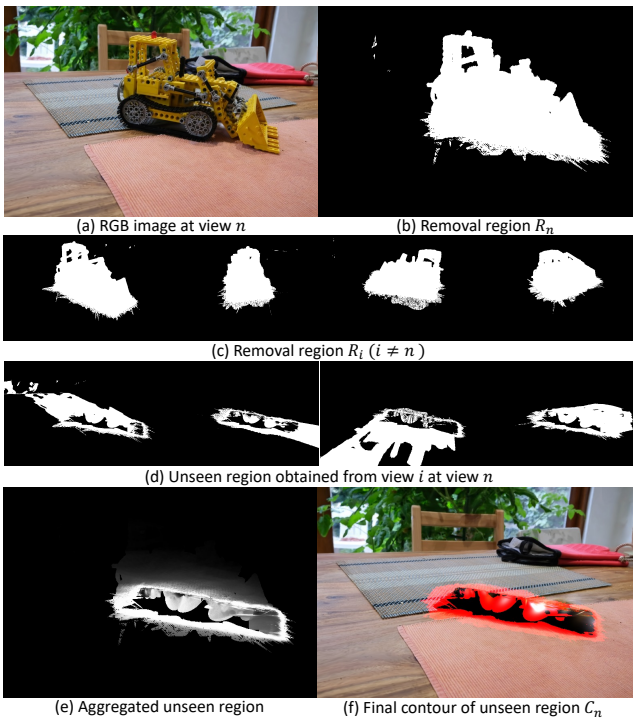

Figure 1. **Intermediate Results of Depth Warping for Unseen Region Detection.** This figure illustrates the intermediate results generated during the depth warping process. (a) and (b) show the RGB image and the corresponding removal region at view n, respectively. (c) displays the removal regions obtained from view i ($i \neq n$). (d) shows the unseen region obtained from view i through backward traversal. The intersections are concentrated near the unseen region. Note that the pixels within the unseen region, but with a value of zero, are due to the absence of Gaussians in that area, preventing depth rendering and thus making it impossible to establish pixel correspondences between view n and view i. (e) presents the aggregation of all unseen regions obtained from view i at view n. A threshold is applied to this result, and it is then intersected with the removal region at view n to obtain the final result in (f).

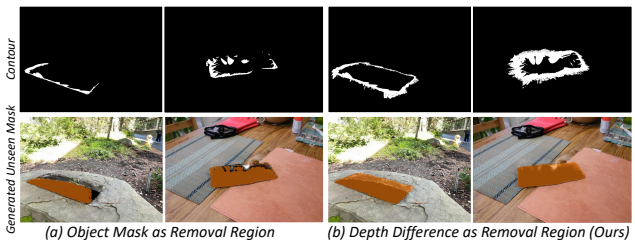

Figure 2. **Ablation Study on Removal Region Definition.** Comparison of (a) object masks vs. (b) depth difference for defining removal regions. Object masks fail to capture geometric changes, leading to less accurate unseen masks. Depth difference better preserves scene structure, improving SAM2 prompts and unseen region segmentation.

unseen region in view n. These contours can then be used as the bounding box prompt for SAM2, resulting in a more accurate unseen mask.

D. Comparison of Depth Completion Methods

In addition to Fig. 11 of the main paper, we compare scale–shift alignment, LaMa [10], InFusion [6], GDD [13], and AGDD for depth completion. As shown in Tab. 1, we evaluate the mean absolute difference (MAD) in object mask areas in 30 test views, using pseudo-GT depth from a 2DGS trained on 200 removal images, as mentioned in Sec. 4. Aligning scale-shift misaligns boundaries in 360° scenes, while LaMa provides reasonable depth completion but does not fully resolve alignment issues. AGDD achieves the lowest MAD and better handles complex geometry.

Table 1. MAD values for different depth completion methods.

Depth completion method	MAD ↓
Scale-shift align	0.063
LaMa depth inpainting	0.077
InFuion	0.047
GDD	0.065
AGDD	**0.045**

E. Reference Images in Real-World Use

Our 360-USID dataset provides real-world captured reference images. However, this does not mean that our method requires extra input. In practical scenarios, reference images can be captured post-removal for real-world use. We also ensure a fair evaluation by avoiding hallucinated textures, even if the inpainting is consistent. Additionally, reference guidance helps reduce multi-view inconsistency with minimal extra input. As shown in Tab. 2, while LaMa-based references slightly degrade the results, they still outperform other reference-based methods, such as GScream. Even when using an inpainted image as a reference, our approach still achieves good results.

Table 2. Comparison of Captured and Inpainted Reference.

Reference method	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓
GScream	14.758	0.955	0.514	152.295
LaMa-reference	17.102	0.960	0.407	69.874
Captured-reference	**17.661**	**0.961**	**0.388**	**62.173**

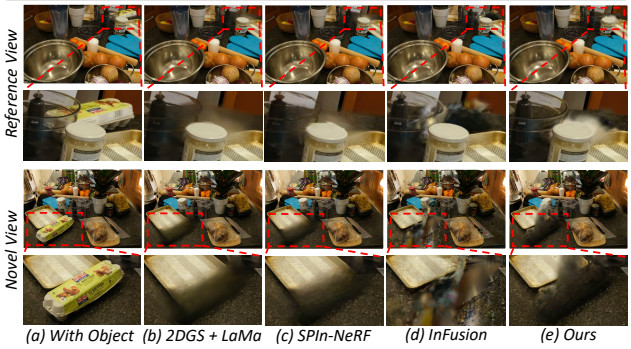

Figure 3. **Failure Cases.** The figure illustrates failure cases of inpainting results. These examples highlight the challenges of 3D inpainting when significant occlusions are present near the regions requiring inpainting. For instance, (b) and (c) demonstrate difficulties in achieving satisfactory guided inpainted RGB images in the training views, while (d) and (e) show errors resulting from incorrect pixel unprojections. These observations indicate that this issue is not effectively addressed by any of the compared methods, suggesting a potential avenue for further exploration and improvement.

F. Experimetal Setup

F.1. LeftRefill [2]

We use the same reference image as in our method, along with the rendered object masks of each novel testing view generated by our masked Gaussians, as input to LeftRefill and directly perform reference-based inpainting on each testing novel view.

F.2. 2DGS [4] + LaMa [10]

We provide the same reference image and training view object masks as in our method and use LaMa [10] to obtain per-frame inpainting results for each training view to train the 2DGS.

F.3. 2DGS [4] + LeftRefill [2]

We provide the same reference image and training view object masks as in our method and use LeftRefill to obtain per-frame inpainting results for each training view to train the 2DGS.

F.4. SPIn-NeRF [8]

The original SPIn-NeRF [8] codebase is designed for forward-facing scenes; however, we adapt it for comparison on 360° scenes by implementing its approach on 2DGS [4].

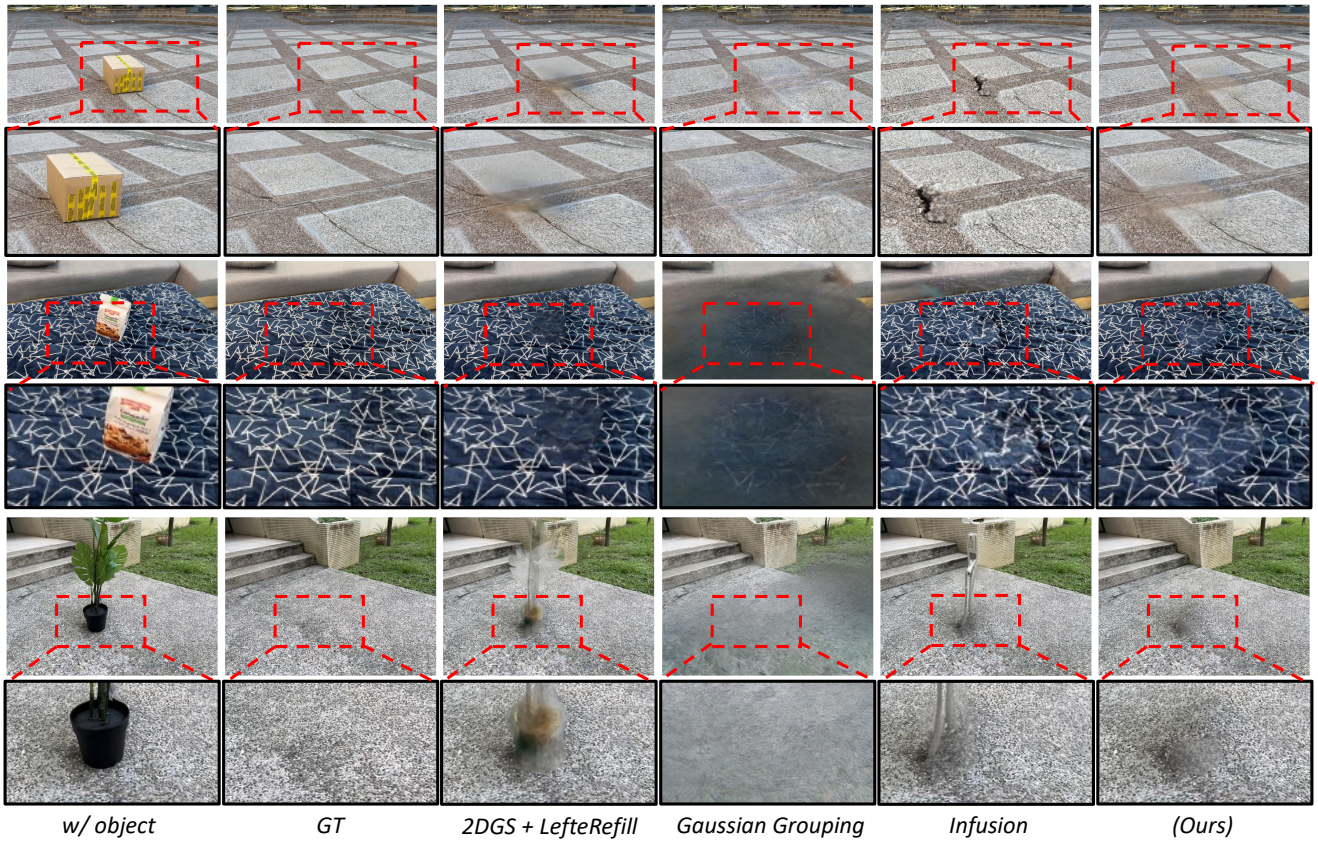

Figure 4. **Visual Comparison on our 360-USID dataset.**

We first obtain the depth for each training view by training a 2DGS model. Next, we generate inpainted RGB and depth maps using LaMa [10], which are then used to train the inpainted 2DGS model. During training, we follow SPIn-NeRF's methodology by incorporating patch-based RGB-LPIPS loss and using the Pearson correlation coefficient to compute a scale- and shift-invariant depth loss.

F.5. Gscream [11]

We follow the original GScream [11] pipeline as a baseline for comparison. We provide the same reference image and training view object masks as our method to ensure consistency. Following their pipeline, we use Marigold [5] to generate estimated depths for all training images, meeting GScream's input data requirements.

F.6. Gaussian Grouping [11]

We utilize the original Gaussian Grouping [12] codebase as a baseline for comparison. First, it generates segmentation IDs, from which we select the IDs corresponding to objects that require inpainting. These selected IDs are then used in the removal process. Following the original workflow, the unseen regions are identified, subsequently inpainted, and used for their fine-tuning process.

Notably, after removing objects from the scene, Gaussian Grouping relies on TrackingAnything-DEVA [3] to identify unseen regions requiring further inpainting through the "black blurry hole" prompt. However, DEVA occasionally fails to accurately identify unseen regions in certain scenes, leading to incorrect inpainting and suboptimal results. Additionally, in some scenes, such as the *bonsai* scene from the Mip-NeRF-360 [1] dataset and the *plant* scene from the 360-UISD dataset, the object tracker misidentifies objects, resulting in incorrect object removal and further degrading the inpainting quality.

F.7. InFusion [6]

We use the original InFusion [6] codebase as a baseline for comparison. We provide the same reference image used in our method as the input RGB for its depth completion model. This reference image is also used in its fine-tuning process.

G. Limitations

Our method successfully addresses complex, unbounded 360° scene inpainting. However, rendering the unprojected initial Gaussians and applying SDEdit [7] to enhance the

Figure 5. **Visual Comparison on Other-360 dataset.**

guided inpainted RGB images can be time-consuming, particularly for high-resolution or large-scale scenes, which poses challenges for real-time applications. Furthermore, our analysis Fig. 3 shows that the method may produce incorrect pixel unprojections in cases with significant occlusions near the object requiring inpainting, resulting in floaters in the final inpainted outputs. This limitation is similarly observed across all compared methods, underscoring a valuable direction for future research and improvement.

References

[1] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *CVPR*, 2022. 1, 3

[2] Chenjie Cao, Yunuo Cai, Qiaole Dong, Yikai Wang, and Yanwei Fu. Leftrefill: Filling right canvas based on left reference through generalized text-to-image diffusion model. In *CVPR*, 2024. 2

[3] Ho Kei Cheng, Seoung Wug Oh, Brian Price, Alexander Schwing, and Joon-Young Lee. Tracking anything with de-

coupled video segmentation. In *ICCV*, 2023. 3

[4] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 Conference Papers*, 2024. 1, 2

[5] Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation. In *CVPR*, 2024. 3

[6] Zhiheng Liu, Hao Ouyang, Qiuyu Wang, Ka Leong Cheng, Jie Xiao, Kai Zhu, Nan Xue, Yu Liu, Yujun Shen, and Yang Cao. Infusion: Inpainting 3d gaussians via learning depth completion from diffusion prior. *arXiv preprint arXiv:2404.11613*, 2024. 2, 3

[7] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *ICLR*, 2022. 3

[8] Ashkan Mirzaei, Tristan Aumentado-Armstrong, Konstantinos G. Derpanis, Jonathan Kelly, Marcus A. Brubaker, Igor Gilitschenski, and Alex Levinshtein. SPIn-NeRF: Multiview segmentation and perceptual inpainting with neural radiance fields. In *CVPR*, 2023. 2

[9] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. In *ICLR*, 2025. 1

[10] Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor Lempitsky. Resolution-robust large mask inpainting with Fourier convolutions. In *WACV*, pages 2149–2159, 2022. 2, 3

[11] Yuxin Wang, Qianyi Wu, Guofeng Zhang, and Dan Xu. Gscream: Learning 3d geometry and feature consistent gaussian splatting for object removal. In *ECCV*, 2024. 3

[12] Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian grouping: Segment and edit anything in 3d scenes. In *ECCV*, 2024. 1, 3

[13] Hong-Xing Yu, Haoyi Duan, Charles Herrmann, William T. Freeman, and Jiajun Wu. Wonderworld: Interactive 3d scene generation from a single image. In *CVPR*, 2025. 2