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

Chung-Ho Wu[1]   Yang-Jung Chen[1]   Ying-Huan Chen[1]   Jie-Ying Lee[1]

Bo-Hsu Ke[1]   Chun-Wei Tuan Mu[1]   Yi-Chuan Huang[1]   Chin-Yang Lin[1]

Min-Hung Chen[2]   Yen-Yu Lin[1]   Yu-Lun Liu[1]

[1]National Yang Ming Chiao Tung University   [2]NVIDIA
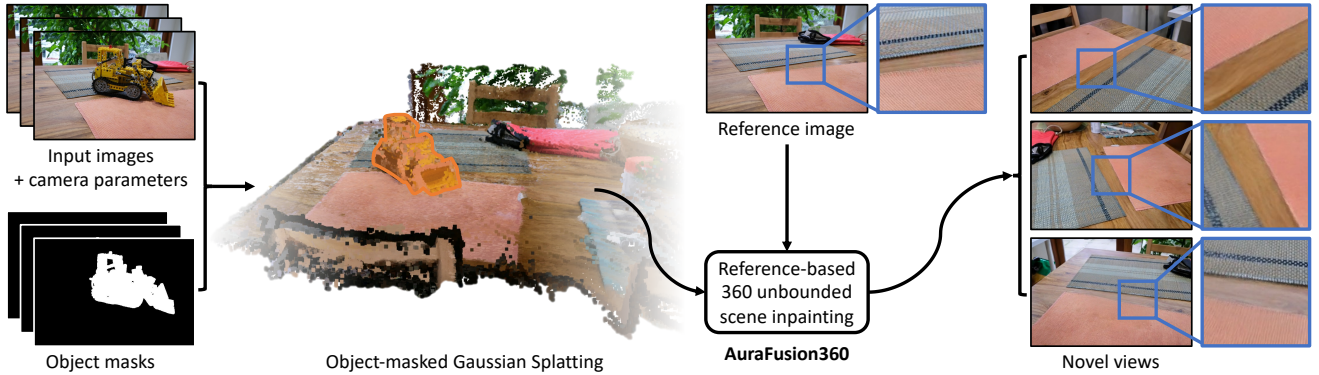
https://kkennethwu.github.io/aurafusion360/

Figure 1. **Overview of our reference-based 360° unbounded scene inpainting method.** Given input images with camera parameters, object masks, and a reference image, our AuraFusion360 approach generates an object-masked Gaussian Splatting representation. This representation can then render novel views of the inpainted scene, effectively removing the masked objects while maintaining consistency with the reference image.

Abstract

Three-dimensional scene inpainting is crucial for applications from virtual reality to architectural visualization, yet existing methods struggle with view consistency and geometric accuracy in 360° unbounded scenes. We present AuraFusion360, a novel reference-based method that enables high-quality object removal and hole filling in 3D scenes represented by Gaussian Splatting. Our approach introduces (1) depth-aware unseen mask generation for accurate occlusion identification, (2) Adaptive Guided Depth Diffusion, a zero-shot method for accurate initial point placement without requiring additional training, and (3) SDEdit-based detail enhancement for multi-view coherence. We also introduce 360-USID, the first comprehensive dataset for 360° unbounded scene inpainting with ground truth. Extensive experiments demonstrate that AuraFusion360 significantly outperforms existing methods, achieving superior perceptual quality while maintaining geometric accuracy across dramatic viewpoint changes.

1. Introduction

Three-dimensional scene reconstruction, driven by Neural Radiance Fields [33] and 3D Gaussian Splatting [20], is vital for VR/AR, robotics, and autonomous driving. A key challenge is realistic object removal and hole filling, which is essential for augmented reality and real estate visualization. Inpainting 360° unbounded scenes remains difficult due to the need for multi-view consistency, plausible unseen region extrapolation, and geometric coherence across views.

Fig. 1 shows our reference-based 360° unbounded scene inpainting approach. Given input images with camera parameters, object masks, and a reference image, our method generates an inpainted 3D scene using Gaussian Splatting [17, 20] for novel view rendering. We exploit multi-view information and generative models to fill unseen areas, ensuring coherent and plausible results across views. Integrating Gaussian Splatting's explicit representation with 2D generative inpainting, our method maintains multi-view consistency and geometric accuracy under significant viewpoint changes.

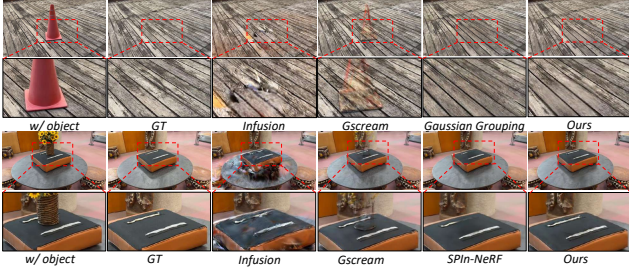Several critical challenges in 360° unbounded scene in-

1

Figure 2. **Comparison with different 3D inpainting approaches.** Existing methods such as SPin-NeRF [35] and GScream [60], designed for forward-facing scenes, perform poorly in 360° scenarios. Reference-based methods like Infusion [28] struggle with accurate depth projection, causing fine-tuning artifacts. Gaussian Grouping [66] frequently misidentifies unseen regions, reducing inpainting quality. Our AuraFusion360 achieves precise unseen masks and improved depth alignment via Adaptive Guided Depth Diffusion, employing SDEdit [31] for diffusion-guided, multi-view consistent RGB generation.

painting motivated our approach (Fig. 2). Existing methods [34–36, 60], effective for forward-facing scenes, struggle with extreme viewpoint changes in 360° scenes, resulting in inconsistencies and artifacts. Recent approaches like Gaussian Grouping [66] effectively propagate semantic information for object removal, but their reliance on a text-based tracker [8] often causes misidentified unseen regions, leading to inaccurate reconstructions.

To address these challenges, we propose a unified pipeline for 360° unbounded scene inpainting using Gaussian Splatting for object removal, depth-aware unseen region detection, and multi-view consistent inpainting. Inspired by Gaussian Grouping [66], our method integrates object-masked attributes into Gaussians for precise removal and reconstructs unseen regions before applying reference-guided inpainting. Unlike methods that directly apply inpainters, causing inconsistencies, we develop Adaptive Guided Depth Diffusion (AGDD) to unproject aligned points from the reference view into unseen regions. These points (1) initialize Gaussians and (2) guide inpainted RGB generation via SDEdit [31], ensuring coherent, high-quality 360° scene restoration.

Integrating these improvements, our framework achieves enhanced geometric accuracy and realism in 360° unbounded scenes. To advance 3D inpainting, we propose a method that improves consistency and provides a benchmark for future research. Our contributions include:

- A depth-aware method leveraging multi-view information to accurately generate unseen masks for 360° unbounded scene inpainting.
- Integration of reference view unprojection with SDEdit to produce consistent RGB guidance across views.
- A comprehensive framework with a new 360° dataset and capture protocol, supporting high-quality novel view synthesis and quantitative evaluation.

2. Related Work

NeRF. Neural Radiance Fields (NeRF)[33] revolutionized novel view synthesis via differentiable volume rendering[15, 55] and positional encoding [13, 56]. NeRF models improved in efficiency [7, 12, 26], rendering quality [2, 32, 72], handling dynamic scenes [27], and data efficiency [22, 52, 59, 68]. Despite excelling at view synthesis, NeRF's implicit representation complicates scene editing. Recent work on object manipulation [64], stylization [14, 57], and inpainting [24, 34, 35] struggles with 3D consistency and structural priors, especially in unbounded scenes.

3D Gaussian Splatting. 3D Gaussian Splatting (3DGS) [20] efficiently represents scenes with explicit 3D Gaussians, enabling faster rendering, easier training, and flexible editing[6]. Recent extensions like Scaffold-GS [29] enhance efficiency with dynamic anchors, while 2DGS [17] refines multi-view geometry. 3DGS has also expanded to dynamic scenes [11, 30, 63, 65] and semantic representations [42, 66], supporting advanced editing and novel view synthesis [17, 43]. Gaussian-based methods thus offer strong potential for explicit 3D inpainting.

Traditional and learning-based image inpainting. Early image inpainting techniques, including PDE-based [4], exemplar-based [9], and PatchMatch [1], were effective for small regions but struggled with complex textures and large gaps [18, 23]. Deep learning advanced the field significantly, starting with Context Encoders [39] and GAN-based methods like DeepFill [70, 71], improving content synthesis and coherence. Recent models such as LaMa [53] use Fourier convolutional networks to address large masks. Diffusion models [16], notably Stable Diffusion [45], introduced iterative refinement capabilities, providing more flexible and structurally consistent inpainting compared to GANs [10].

Diffusion models for image editing and inpainting. Beyond direct inpainting, diffusion models are widely used for image editing. SDEdit [31] injects Gaussian noise and iteratively denoises, enabling semantic edits while preserving global structure. Noise inversion techniques [37, 38], such as DDIM Inversion [51], further improve editing fidelity by enabling precise latent inference through deterministic reverse diffusion. Inpainting-specific diffusion models like SDXL-Inpainting [40] enhance image reconstruction by fine-tuning Stable Diffusion. Reference-based methods [54], such as LeftRefill [5], use diffusion models for reference-guided synthesis but struggle in regions distant from reference views. Despite advancements, Stable Diffusion-based inpainting [41] still suffers from inconsistent artifacts in scene-dependent contexts, causing multi-view inconsistencies problematic for 3D scenes [21]. This motivates our use of SDEdit and DDIM Inversion to preserve structural information and ensure multi-view coherence.

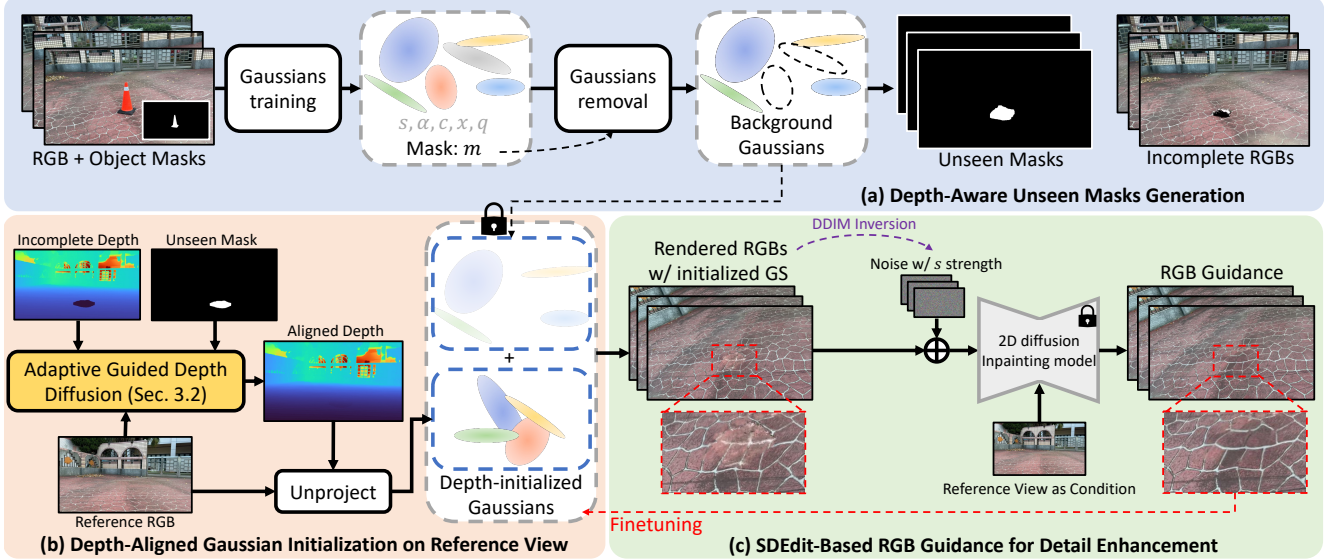**3D scene inpainting.** Existing 3D inpainting methods for

Figure 3. **Overview of our method.** Our approach takes multi-view RGB images and corresponding object masks as input and outputs a Gaussian representation with the masked objects removed. The pipeline consists of three main stages: (a) Depth-Aware Unseen Masks Generation to identify truly occluded areas, referred to as the "unseen region", (b) Depth-Aligned Gaussian Initialization on Reference View to fill unseen regions with initialized Gaussian containing reference RGB information after object removal, and (c) SDEdit-Based RGB Guidance for Detail Enhancement, which enhances fine details using an inpainting model while preserving reference view information. Instead of applying SDEdit with random noise, we use DDIM Inversion on the rendered initial Gaussians to generate noise that retains the structure of the reference view, ensuring multi-view consistency across all RGB Guidance.

NeRF [35, 50, 62, 67] typically adapt 2D models to NeRF's implicit representation. For instance, SPIn-NeRF [35] employs perceptual loss to improve multi-view consistency. Reference-based methods [34, 36, 60] enhance consistency using reference images but remain limited to small-angle view rendering, restricting their use in 360° scenes. NeR-Filler [61] iteratively refines consistency with grid prior but struggles with fine-grained textures due to image downsampling. InNeRF360 [58] handles 360° scenes via density hallucination but has limited scene utilization. Gaussian Splatting-based methods like Gaussian Grouping [66] inject semantic information, while InFusion [28] employs depth completion but requires manual view selection. GScream [29] integrates Scaffold-GS but faces difficulties in unbounded 360° scenes. Our method addresses these issues by enhancing multi-view consistency and depth-aware inpainting in 360° scenarios using Gaussian Splatting.

3. Method

Our method processes multi-view RGB images $\{I_n\}$ and object masks $\{M_n\}$, $n \in [1..N]$, to produce an inpainted Gaussian representation with removed objects. Occluded regions (unseen regions [66]) are consistently inpainted across views. As shown in Fig. 3, the process includes training a masked Gaussian using object masks, removing objects, and applying (a) Depth-Aware Unseen Mask Generation (Sec. 3.1), (b) Reference View Initial Gaussians Alignment

(Sec. 3.2), and (c) SDEdit for Detail Enhancement (Sec. 3.3). This pipeline ensures consistent texture propagation in unbounded scenes, achieving high-quality 3D inpainting.

3.1. Depth-Aware Unseen Mask Generation

Accurate identification of inpainting regions is critical for scene consistency and optimal use of background information. To generate the unseen mask for a view, it is necessary to differentiate between (1) the background visible across multiple views and (2) the unseen region occluded in all views, requiring inpainting.

A naive approach to detecting unseen masks with SAM2 [44] involves manually selecting the first view and propagating prompts across other views. However, SAM2 struggles to consistently detect unseen regions without refinement, often revealing parts of the background or inside objects. To address this, our method employs depth warping to generate bounding box prompts for each view (Fig. 4), ensuring accurate, fully automated unseen region detection.

Depth warping for generating bbox prompt to SAM2. To refine the unseen mask, we employ a depth-warping technique, as illustrated in Fig. 4. For each view n, we compute:

$$R_{i \to n} = \mathcal{W}_{\text{traverse}}(R_i, D_n^{\text{incomplete}}, T_{n \to i}), \qquad (1)$$

where $\mathcal{W}_{\text{traverse}}$ includes forward warping from view n to i and backward traversal to map the removal region back to n. R_i is the removal region mask for view i, derived from
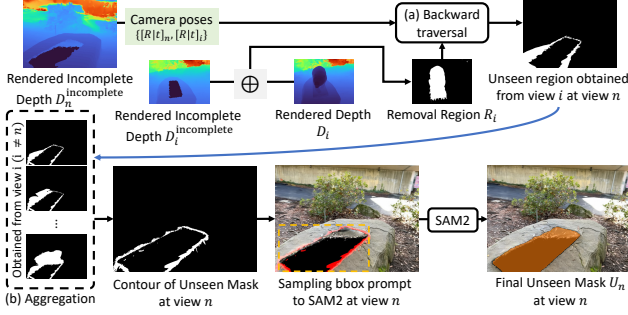
3

Figure 4. **Overview of the Unseen Mask Generation Process using Depth Warping.** To obtain the unseen mask for view n, we calculate the pixel correspondences between the view n and all other views i by using the rendered incomplete depth $D_n^{\text{incomplete}}$. For each view i, the removal region R_i is backward traversal to view n to align occlusions. We then aggregate the results from multiple views, averaging and applying a threshold to produce the initial contour of the unseen mask. This contour is subsequently converted into a bounding box prompt for SAM2 [44], which refines the unseen mask to its final version for view n.

depth differences. $D_n^{\text{incomplete}}$ is the incomplete depth map for view n, and $T_{n \to i}$ is the transformation from view n to i.

The unseen mask contour for view n is obtained by aggregating warped removal regions and applying thresholding:

$$C_n = \theta \left(\frac{1}{K} \sum_{i=1}^{K} R_{i \to n} \right) \cap R_n, \qquad (2)$$

where C_n is the contour of the unseen mask, K is the number of views, and θ is a thresholding function. A bounding box $\text{bbox}(C_n)$ is created as a prompt for SAM2 [44] to generate the final unseen mask:

$$U_n = \text{SAM2}(\text{bbox}(C_n)). \qquad (3)$$

This mask U_n guides the inpainting process, focusing on areas needing reconstruction while preserving original scene information.

3.2. Reference View Initial Gaussians Alignment

After performing object removal and generating the unseen mask, similar to CorrFill [25], we select a reference view called V_{ref}, which can render an incomplete RGB image and depth. We then apply RGB inpainting to the incomplete RGB image of V_{ref} and denote it as I_{ref}. To maximize cross-view consistency, we project the reference RGB image into 3D space using depth estimates of I_{ref}, which is obtained through Adaptive Guided Depth Diffusion. This 3D projection serves two critical purposes: It guides the SDEdit-based RGB detail enhancement and initializes point positions for Gaussian fine-tuning. Accurate depth alignment is, therefore, fundamental to our pipeline, as it directly determines the precision of these initial point positions.

Adaptive Guided Depth Diffusion (AGDD). Aligning *estimated depth* with *existing depth* is challenging due to

monocular depth estimation [19]'s scale ambiguity and non-metric representation across coordinate systems. This challenge intensifies in 360° unbounded scenes, where large viewpoint changes hinder alignment. Traditional scale-shift optimization often yields suboptimal results, while depth-completion models demand costly fine-tuning. Our AGDD refines GDD [69] by addressing over-alignment issues, particularly where depth transitions from small to large values, which exaggerates disparities in distant regions and inflates loss values. To mitigate this, we introduce an adaptive loss L_{adaptive} that balances alignment, preventing distant regions from dominating and yielding more accurate depth estimates.

The framework is shown in Fig. 5. Following the standard denoising process of Marigold [19], we initialize with a latent representation perturbed by full-strength Gaussian noise, denoted as d_t, and generate aligned depth $D_{\text{aligned}} = \text{Decoder}(d_0)$ using a VAE decoder, where the latent d_0 is obtained by recursive denoising step $d_{t-1} = \text{Denoise}(d_t, t, \hat{\epsilon}_t)$. The $\hat{\epsilon}_t$ is derived by updating the original noise through the calculation of adaptive loss L_{adaptive} between the pre-decoded estimated depth D_{t-1} and the existing incomplete depth $D_{\text{incomplete}}$. Note that D_{t-1} is obtained by decoding d_0', which is the model's estimation of the fully denoised latent at timestep 0 when predicted from the noisy state at timestep $t-1$. This adaptive loss refines $\hat{\epsilon}_t$ to ensure that the estimated depth aligns with the existing incomplete depth during denoising. The optimization process is described as follows:

$$d_{t-1} = \text{Denoise}(d_t, t, \hat{\epsilon}_t) \qquad (4)$$

$$\hat{\epsilon}_t = \text{UNet}(d_t, I_{\text{scene}}, t) - \alpha \cdot \nabla \mathcal{L}_{\text{adpative}} \qquad (5)$$

where α is the learning rate for the optimization. We define a bounding box $\mathcal{B}$ around the unseen region and introduce a threshold δ to downweight errors for distant points. The adaptive loss $\mathcal{L}_{\text{adaptive}}$ between the pre-decoded estimated depth D_{t-1} and the incomplete depth $D_{\text{incomplete}}$ is computed as follows:

$$M_{\text{guide}}(x, y) = \begin{cases} 1 & \text{if } (x, y) \in \mathcal{B} \setminus U \\ 0 & \text{otherwise,} \end{cases} \qquad (6)$$

$$\mathcal{L}_{\text{adaptive}} = \sum_{(x,y)} M_{\text{guide}}(x, y) \cdot \mathcal{L}(D_{t-1}, D_{\text{incomplete}})(x, y), \qquad (7)$$

$$\mathcal{L}(d_1, d_2) = \begin{cases} \frac{1}{2}(d_1 - d_2)^2 & \text{if } |d_1 - d_2| < \delta \\ \delta \cdot |d_1 - d_2| - \frac{1}{2}\delta^2 & \text{otherwise,} \end{cases} \qquad (8)$$

where $M_{\text{guide}}(x, y)$ is a mask function indicating if a pixel (x, y) is within the bounding box $\mathcal{B}$ but not in the unseen mask U. At each denoising step, we update the noise over N iterations. Instead of directly optimizing the noise using L2 loss [69], this loss ensures that the updated noise input to the denoiser enables it to generate an estimated depth that aligns with the incomplete guided depth. This enables the AGDD output to achieve accurate alignment in regions adjacent to
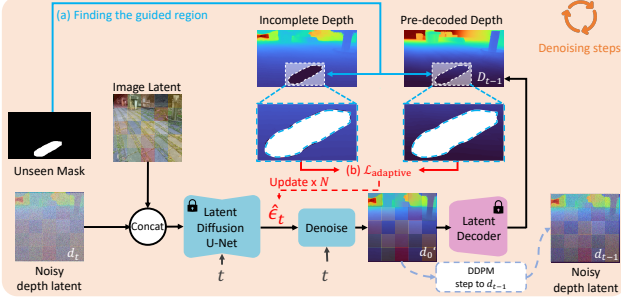
Figure 5. **Overview of Adaptive Guided Depth Diffusion (AGDD).** The framework takes image latent, incomplete depth, and unseen mask as inputs to generate aligned depth estimates. (a) The guided region is identified by dilating the unseen mask and subtracting the original mask. (b) At each timestep t, adaptive loss $\mathcal{L}_{\text{adaptive}}$ is computed between the pre-decoded and incomplete depth to update the noise input $\hat{\epsilon}_t$. This repeats N times before advancing to the next denoising step, ensuring the estimated depth aligns with the incomplete depth distribution in the guided region.

unseen areas, which is more appropriate for depth inpainting scenarios while also operating in a zero-shot manner.

Initializing Gaussians in unseen regions. With the aligned depth $D_{\text{aligned}}^{\text{ref}}$ of the reference view, we proceed to initialize new Gaussians in the unseen regions. First, we unproject the inpainted RGB of the reference view with $D_{\text{aligned}}^{\text{ref}}$ to 3D space, focusing on the unseen regions identified by the unseen mask. This unprojection takes into account the camera's intrinsic parameters. For each pixel (u, v) in the unseen region where $U_{\text{final}}(u, v) = 1$, we compute the 3D point $P = (X, Y, Z)$ as $Z = D_{\text{aligned}}^{\text{ref}}(u, v)$, $X = (u - c_x) \cdot Z/f_x$, $Y = (v - c_y) \cdot Z/f_y$, where (f_x, f_y) are the focal lengths in pixels and (c_x, c_y) are the principal point offsets. This process gives us a set of initial 3D points P. These points are then used to initialize new Gaussians in the unseen regions, inheriting color from the reference view. Existing background Gaussians, unaffected by object removal, remain fixed during initialization and optimization. These initialized Gaussians are crucial for the subsequent process of generating guided inpaint RGB images and optimization.

3.3. SDEdit for Detail Enhancement

After initializing Gaussians in unseen regions, we aim to obtain the inpainted RGB guidance with fine details while ensuring multi-view consistency, which further refines our initial Gaussians during fine-tuning. Inspired by SDEdit [31], we refine the rendered initial Gaussians by adding scaled noise proportional to a strength factor s, ensuring that the inpainting model retains structural information from the reference view while allowing for detail refinement across multiple perspectives. We further find that instead of injecting random Gaussian noise, applying DDIM Inversion [51] to the rendered initial Gaussians better preserves their structural

information during the denoising process. This approach allows the diffusion inpainting model to reconstruct missing details while maintaining alignment with the reference view, ensuring that inpainted regions integrate seamlessly into the scene (see Fig. 11).

Specifically, given a rendered training view I_{init}, we first obtain its corresponding noise representation via DDIM Inversion, capturing the essential structure of the reference view in the latent space. Instead of inverting fully to t_0, we compute an intermediate timestep t_{inv} based on the noise strength s:

$$t_{\text{inv}} = T(1 - s), \tag{9}$$

where T is the total number of timesteps in the diffusion process, and s controls the noise strength. We then perform DDIM Inversion to obtain the noise representation at t_{inv}:

$$\epsilon_{\text{inv}} = \text{DDIM-Invert}(I_{\text{init}}, t_{\text{inv}}). \tag{10}$$

Next, we denoise this noise using a 2D diffusion inpainting model, conditioned on the reference view I_{ref}, ensuring that the reconstructed details align with the global scene while maintaining consistency across views:

$$I_{\text{guided}} = \text{Denoise}(\epsilon_{\text{inv}}, \text{condition} = I_{\text{ref}}, t_{\text{inv} \to 0}). \tag{11}$$

By inverting to a noise level corresponding to strength s, this step ensures that the inpainting model refines details while maintaining geometric consistency with the reference view. Unlike traditional SDEdit, which applies random noise addition before denoising, our approach leverages DDIM Inversion to obtain structured noise that aligns with the scene, preventing hallucinated details that could disrupt multi-view coherence.

The resulting guided inpainted RGBs are then used as supervision for Gaussian fine-tuning, updating only the unprojected Gaussians from Sec. 3.2. The final reconstruction is optimized using a combination of L1, SSIM, and LPIPS [73] losses:

$$\mathcal{L} = (1 - \lambda_{\text{SSIM}})\mathcal{L}_1 + \lambda_{\text{SSIM}}\mathcal{L}_{\text{SSIM}} + \lambda_{\text{LPIPS}}\mathcal{L}_{\text{LPIPS}}. \tag{12}$$

3.4. Implementation Details

We use the 2D Gaussian Splatting [17] codebase for Gaussian representation to obtain accurate rendered depth, with SAM2 generating object masks on the first frame for each training view. Masked Gaussians enable effective object removal due to their explicit representation. We set the aggregation threshold of θ to 0.6 in unseen mask generation. In AGDD, incomplete depth are normalized to match Marigold's [19] depth. With N set to 8, the denoised result is then unnormalized back to its original scale. The entire inference process takes approximately 1 minute on an RTX 4090 GPU. The noise strength of SDEdit $s = 0.85$ balances initial point retention, as shown in our ablation study. We condition the generation on the reference view using LeftRefill [5]. During Gaussian fine-tuning, we run 10,000 iterations with $\lambda_{\text{SSIM}} = 0.8$ and $\mathcal{L}_{\text{LPIPS}} = 0.5$.

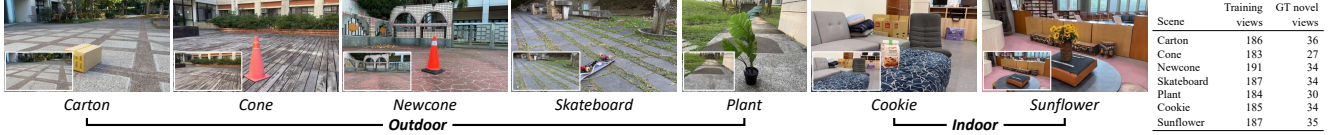| Scene	Training views	GT novel views
Carton	186	36
Cone	183	27
Newcone	191	34
Skateboard	187	34
Plant	184	30
Cookie	185	34
Sunflower	187	35

Figure 6. **Overview of the 360-USID dataset.** Sample images from each scene, including five outdoor scenes (Carton, Cone, Newcone, Skateboard, Plant) and two indoor scenes (Cookie, Sunflower). (*Bottom right*) The table shows statistics for each scene, including the number of training views and ground truth (GT) novel views. The dataset provides a diverse range of environments for evaluating 3D inpainting methods in both indoor and outdoor settings.
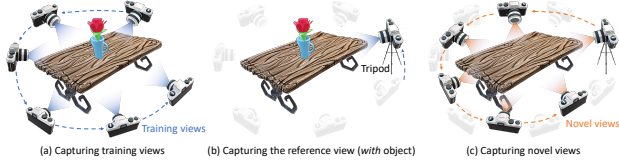

Figure 7. **Illustration of the data capture process for the 360-USID dataset.** (a) Capturing training views: Multiple images are taken around the object in the scene. (b) Capturing the reference view: A camera is mounted on a tripod to capture a fixed reference view (with an object). (c) Capturing novel views: After removing the object, additional images are taken from various viewpoints, including one from the same tripod position as the reference image.

4. 360° Unbounded Scenes Inpainting Dataset

To address the lack of reference-based 360° inpainting datasets, we introduce the 360° Unbounded Scenes Inpainting Dataset (360-USID), consisting of seven scenes with training views (RGB images and object masks), novel testing views (inpainting ground truth), and a reference view (without objects) for evaluating with other reference-based methods.

Dataset collection protocol. We developed a protocol using a standard camera to create this dataset, as simultaneously capturing multi-view photos with and without objects typically requires specialized equipment. Our protocol, illustrated in Fig. 7, consists of:

1. Positioning an object (*e.g.* a vase) on a textured surface within a 360° unbounded scene. Training views are captured in two complete circular trajectories around the object - the first focuses primarily on the object, while the second maximizes background coverage to ensure comprehensive scene capture.
2. Securing the camera on a tripod and capturing a reference view from a fixed position and orientation.
3. After object removal, capturing novel views from both the fixed tripod position and additional positions distinct from training trajectories for ground truth evaluation.

To ensure high-quality captures, we record video at 4K 60fps with stabilized camera settings and extract the sharpest frames using the variance of the Laplacian method. Each scene comprises 180∼200 training views and approximately 30 testing views for quantitative evaluations. Consistent light-

ing is maintained throughout to minimize shadow variations between reference and testing images

Data preprocessing and pose estimation. Our processing pipeline begins with using COLMAP [48, 49] or similar SfM pipelines like hloc [46, 47] to compute a shared 3D coordinate space for both training and novel views. We then generate object masks for training views using SAM2 [44] and mask out object regions in COLMAP reconstruction. After obtaining camera poses, we process the training images with NeRF/3DGS inpainting methods and render novel views for comparison against ground truth. Finally, we refine testing views by training a masked-3DGS model and selecting optimal frames based on PSNR scores computed outside object regions, yielding approximately 30 high-quality test views per scene. The resulting dataset provides a comprehensive benchmark for evaluating 360° inpainting methods across diverse scenes and viewpoints, with particular attention to view consistency and geometric accuracy.

Scene descriptions. Our 360-USID dataset, shown in Fig. 6, contains seven diverse scenes: five outdoor (Carton, Cone, Newcone, Plant, Skateboard) and two indoor (Cookie, Sunflower). Each scene includes 180-200 training images at 3840×2160 resolution (Plant at 1920×1440), 30 ground truth testing images, and one reference image without objects. Scenes are downscaled to 960×540 for evaluation, providing a comprehensive benchmark for testing 3D inpainting methods across varied real-world environments.

5. Experiments

5.1. Experimental setup

Datasets. We evaluate on two 360° unbounded scene datasets: (1) **360-USID (Ours)**: A new dataset of 7 scenes (3 indoor, 4 outdoor) for evaluating 360° inpainting, with 200-300 training views containing objects, around 30 test views without objects, and 1 reference. All images are processed at 960px width to preserve details for quantitative evaluation. (2) **Other-360 [3]** We collect additional 6 standard 360° unbounded scene datasets from NeRF[33], MipNeRF-360[3] and Instruct-NeRF2NeRF[14] for qualitative evaluation at 1/4 resolution, with frame 0 as reference for all methods.

Metrics. We evaluate our method using two complementary metrics: LPIPS (Learned Perceptual Image Patch Similar-

PSNR ↑ / LPIPS ↓	Carton	Cone	Cookie	Newcone	Plant	Skateboard	Sunflower	Average
SPIn-NeRF [35]	16.659 / 0.539	15.438 / 0.389	11.879 / 0.521	17.131 / 0.519	16.850 / 0.401	15.645 / 0.675	23.538 / 0.206	16.734 / 0.464
2DGS [17] + LaMa [53]	16.433 / 0.499	15.591 / 0.351	11.711 / 0.538	16.598 / 0.670	14.491 / 0.564	15.520 / 0.639	23.024 / 0.194	16.195 / 0.494
2DGS [17] + LeftRefill [5]	15.157 / 0.567	16.143 / 0.372	12.458 / 0.526	16.717 / 0.677	12.856 / 0.666	16.429 / 0.634	24.216 / 0.181	16.282 / 0.518
LeftRefill [5]	14.667 / 0.560	14.933 / 0.380	11.148 / 0.519	16.264 / 0.448	16.183 / 0.463	14.912 / 0.572	18.851 / 0.331	15.280 / 0.468
Gaussian Grouping [66]	16.695 / 0.502	14.549 / 0.366	11.564 / 0.731	16.745 / 0.533	16.175 / 0.440	16.002 / 0.577	20.787 / 0.209	16.074 / 0.480
GScream [60]	14.609 / 0.587	14.655 / 0.476	12.733 / 0.429	13.662 / 0.605	16.238 / 0.437	12.941 / 0.626	18.470 / 0.436	14.758 / 0.514
Infusion [28]	14.191 / 0.555	14.163 / 0.439	12.051 / 0.486	9.562 / 0.624	16.127 / 0.406	13.624 / 0.638	21.195 / 0.238	14.416 / 0.484
AuraFusion360 (Ours) w/o SDEdit	13.731 / 0.477	14.260 / 0.390	12.332 / 0.445	16.646 / 0.460	17.609 / 0.319	15.107 / 0.580	24.884 / 0.170	16.367 / 0.406
AuraFusion360 (Ours)	17.675 / 0.473	15.626 / 0.332	12.841 / 0.434	17.536 / 0.426	18.001 / 0.322	17.007 / 0.559	24.943 / 0.173	17.661 / 0.388

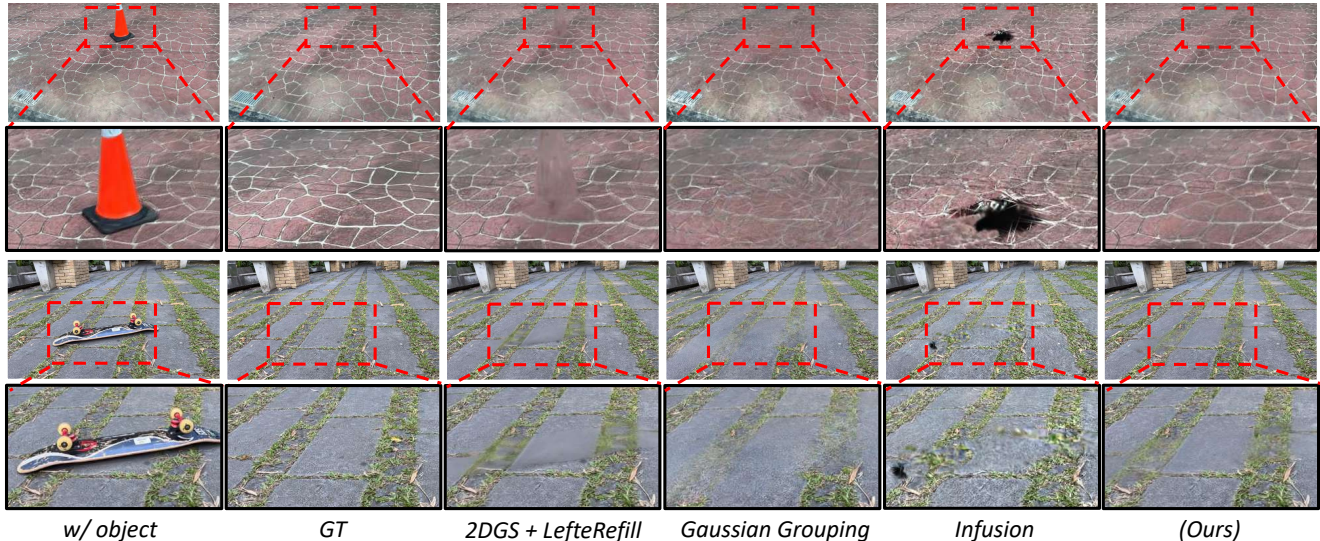| w/ object | GT | 2DGS + LefteRefill | Gaussian Grouping | Infusion | (Ours) |

Figure 8. **Visual Comparison on our 360-USID dataset.** We compare our method against state-of-the-art approaches including Gaussian Grouping [66], 2DGS + LeftRefill, and Infusion [28]. While Gaussian Grouping struggles with misidentifying unseen regions, leading to floating artifacts, and 2DGS + LeftRefill faces view consistency issues, our method successfully maintains geometric consistency and preserves fine details across different viewpoints. Ground truth (GT) is shown for reference, and the original scene with an object is provided in the first row for comparison.

ity) [73] for perceptual quality and PSNR (Peak Signal-to-Noise Ratio) for reconstruction accuracy. Following SPIn-NeRF [35], we compute these metrics only within object masks to focus on inpainting quality. While both metrics are used for 360-USID, which has ground truth, only qualitative assessment is possible for Other-360. Additional evaluation results are provided in supplementary materials.

5.2. Comparisons with State-of-the-Art Methods

Quantitative comparisons. We evaluate AuraFusion360 against state-of-the-art approaches on the 360-USID dataset. Tab. 1 shows PSNR and LPIPS scores across different scenes. Our method consistently outperforms existing approaches. SPIn-NeRF [35][1] and Infusion [28] struggle with 360° consistency, while Gaussian Grouping [66] misidentifies the unseen region, causing significant floating artifacts. GScream [60] fails to properly remove objects, and LeftRefill [5] improves but still falls short in 360° environments. 2DGS + LaMa [53] and 2DGS + LeftRefill outperform 2D

methods but face view consistency challenges. Our method achieves the highest PSNR score and the lowest average LPIPS, indicating superior perceptual quality and better similarity to the ground truth. The performance gap is especially noticeable in scenes with complex geometry or large removed objects, demonstrating our method's ability to leverage multi-view information and maintain 360° consistency. The code for InNeRF360 [58] could not be successfully executed, and [34] did not provide code, so we were unable to compare our method with theirs.

Qualitative visual comparisons. Fig. 8 compares our AuraFusion360 method against state-of-the-art approaches on challenging scenes from the 360-USID dataset. Our method excels in maintaining view consistency and preserving fine details in 360° unbounded environments. Additional qualitative results on other 360 datasets and failure cases are

---

[1]We implement SPin-NeRF's method on the 2D Gaussian Splatting codebase to extend its capabilities to 360° unbounded scenes.

Table 2. **Ablation study of our AuraFusion360.**

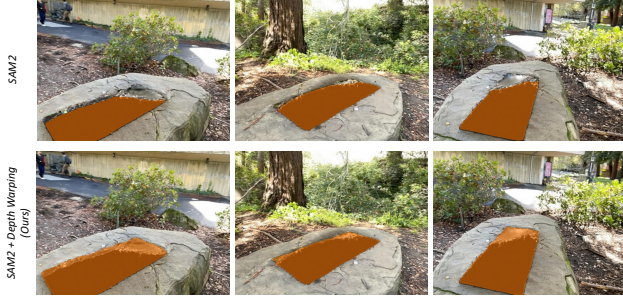| Depth init. (Sec. 3.2)	SDEdit strength (Sec. 3.3)	PSNR ↑	LPIPS ↓
	0.85	16.638	0.456
✓	0.5	17.646	0.393
✓	1.0	17.512	0.391
✓	0.85	**17.661**	**0.388**

Figure 9. **Visual comparison of unseen mask generation method.** Our method enables SAM2 [44] to generate more accurate predictions for each view without the need for manually provided prompts, as the bounding box prompts are automatically generated through depth warping.

provided in the supplementary material.

5.3. Ablation Studies

To evaluate the effectiveness of each component in our AuraFusion360 method, we conduct a series of ablation studies. Tab. 2 presents the quantitative results of these studies.

Unseen mask generation. We compared our unseen mask generation method with SAM2 [44] and Gaussian Grouping [66] tracker in Fig. 9 and Fig. 10. Our approach significantly improves inpainting quality, particularly in areas occluded from multiple views. The unseen masks identify truly occluded regions, leading to more accurate and consistent inpainting results. This is especially noticeable in scenes with complex geometries, where object masks alone may not capture all necessary information for effective inpainting.

Effect of reference view initial Gaussians alignment. Tab. 2 and Fig. 11 show that our depth-aware 3DGS initialization accurately estimates aligned depth while maintaining geometric consistency in the inpainted regions. Compared to random initialization, our method produces more structurally coherent results, particularly in areas with significant depth variations. This is especially evident in scenes where the inpainted geometry needs to blend seamlessly with the existing scene structure.

6. Conclusion

We presented AuraFusion360, a novel reference-based 360° inpainting method for 3D scenes in unbounded environments.

(a) Gaussian Grouping Incomplete RGB   (b) Gaussian Grouping Unseen Mask   (c) Ours Incomplete RGB   (d) Ours Unseen Mask
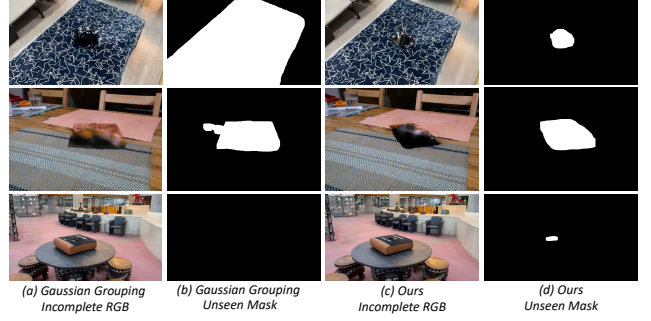
Figure 10. **Compared Unseen Mask w/ Gaussian Grouping.** Gaussian Grouping [66] uses a video tracker [8] and the "black blurry hole" prompt for DEVA [8] to track the unseen region. However, this can result in tracking errors, affecting inpainting. In contrast, our geometry-based approach uses depth warping to estimate the unseen region's contour, reducing segmentation errors.

(a) Infusion   (b) Align Scale & Shift   (c) Poisson Depth Blending   (d) Guided Depth Diffusion   (e) AGDD (Ours)
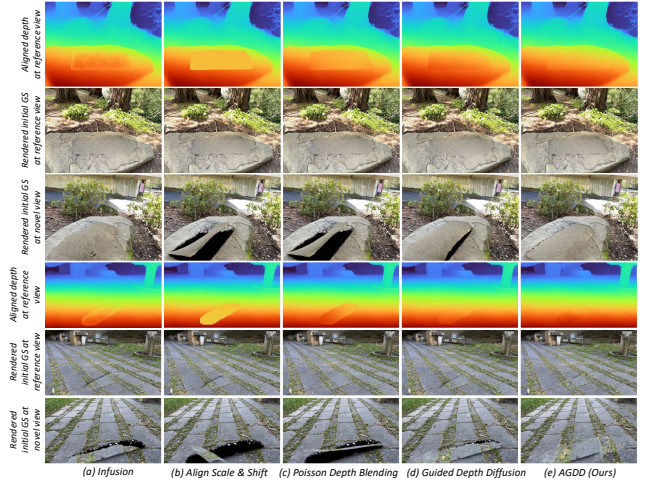
Figure 11. **Compared to other depth completion methods.** The depth completion model in Infusion [28] (a) performs better at depth alignment compared to traditional methods (b) and (c), but it produces noisy depth in unseen regions. Similarly, (d) Guided Depth Diffusion [69] struggles to achieve precise alignment, as the background regions amplify the loss, leading to misalignment. In contrast, (e) Our AGDD effectively addresses these issues.

Our approach effectively addresses the challenges of object removal and hole filling in complex 3D scenes. Key contributions include leveraging multi-view information through improved unseen mask generation, integrating reference-guided 3D inpainting with diffusion priors, and introducing the 360-USID dataset for comprehensive evaluation. Experimental results demonstrate AuraFusion360's superior performance over existing methods, particularly in complex geometries and large view variations. While this work represents a significant advancement in 3D scene editing, future work will focus on computational efficiency, dynamic scenes, and language-guided editing capabilities.

References

[1] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM TOG*, 2009. 2

[2] Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. Mip-NeRF: A multiscale representation for anti-aliasing neural radiance fields. In *ICCV*, 2021. 2

[3] 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. 6, 12, 14

[4] M Bertalmio. Image inpainting, 2000. 2

[5] 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, 5, 7, 13

[6] Yiwen Chen, Zilong Chen, Chi Zhang, Feng Wang, Xiaofeng Yang, Yikai Wang, Zhongang Cai, Lei Yang, Huaping Liu, and Guosheng Lin. Gaussianeditor: Swift and controllable 3d editing with gaussian splatting. In *CVPR*, 2024. 2

[7] Bo-Yu Cheng, Wei-Chen Chiu, and Yu-Lun Liu. Improving robustness for joint optimization of camera pose and decomposed low-rank tensorial radiance fields. In *AAAI*, 2024. 2

[8] Ho Kei Cheng, Seoung Wug Oh, Brian Price, Alexander Schwing, and Joon-Young Lee. Tracking anything with decoupled video segmentation. In *ICCV*, 2023. 2, 8, 14

[9] Antonio Criminisi, Patrick Pérez, and Kentaro Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE TIP*, 2004. 2

[10] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In *NeurIPS*, 2021. 2

[11] Cheng-De Fan, Chen-Wei Chang, Yi-Ruei Liu, Jie-Ying Lee, Jiun-Long Huang, Yu-Chee Tseng, and Yu-Lun Liu. Spectromotion: Dynamic 3d reconstruction of specular scenes. In *CVPR*, 2025. 2

[12] Stephan J. Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. FastNeRF: High-fidelity neural rendering at 200FPS. In *ICCV*, 2021. 2

[13] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutional sequence to sequence learning. In *ICML*, 2017. 2

[14] Ayaan Haque, Matthew Tancik, Alexei A. Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. In *ICCV*, 2023. 2, 6

[15] Philipp Henzler, Niloy J. Mitra, and Tobias Ritschel. Escaping Plato's cave: 3D shape from adversarial rendering. In *ICCV*, 2019. 2

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020. 2

[17] 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, 7, 12, 13

[18] Jireh Jam, Connah Kendrick, Kevin Walker, Vincent Drouard, Jison Gee-Sern Hsu, and Moi Hoon Yap. A comprehensive review of past and present image inpainting methods. *CVIU*, 203:103147, 2021. 2

[19] 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. 4, 5, 14

[20] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM TOG*, 2023. 1, 2

[21] Xin Li, Yulin Ren, Xin Jin, Cuiling Lan, Xingrui Wang, Wenjun Zeng, Xinchao Wang, and Zhibo Chen. Diffusion models for image restoration and enhancement–a comprehensive survey. *arXiv preprint arXiv:2308.09388*, 2023. 2

[22] Chin-Yang Lin, Chung-Ho Wu, Chang-Han Yeh, Shih-Han Yen, Cheng Sun, and Yu-Lun Liu. Frugalnerf: Fast convergence for few-shot novel view synthesis without learned priors. In *CVPR*, 2025. 2

[23] Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *ECCV*, 2018. 2

[24] Hao-Kang Liu, I-Chao Shen, and Bing-Yu Chen. NeRF-In: Free-form NeRF inpainting with RGB-D priors. In *arXiv*, 2022. 2

[25] Kuan-Hung Liu, Cheng-Kun Yang, Min-Hung Chen, Yu-Lun Liu, and Yen-Yu Lin. Corrfill: Enhancing faithfulness in reference-based inpainting with correspondence guidance in diffusion models. In *WACV*, 2025. 4

[26] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *NeurIPS*, 2020. 2

[27] Yu-Lun Liu, Chen Gao, Andreas Meuleman, Hung-Yu Tseng, Ayush Saraf, Changil Kim, Yung-Yu Chuang, Johannes Kopf, and Jia-Bin Huang. Robust dynamic radiance fields. In *CVPR*, 2023. 2

[28] 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, 8, 12, 14

[29] Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *CVPR*, 2024. 2, 3

[30] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In *3DV*, 2024. 2

[31] 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. 2, 5, 14

[32] Andreas Meuleman, Yu-Lun Liu, Chen Gao, Jia-Bin Huang, Changil Kim, Min H Kim, and Johannes Kopf. Progressively optimized local radiance fields for robust view synthesis. In *CVPR*, 2023. 2

[33] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 2, 6

[34] Ashkan Mirzaei, Tristan Aumentado-Armstrong, Marcus A. Brubaker, Jonathan Kelly, Alex Levinshtein, Konstantinos G. Derpanis, and Igor Gilitschenski. Reference-guided controllable inpainting of neural radiance fields. In *ICCV*, 2023. 2, 3, 7

[35] 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, 3, 7, 13

[36] Ashkan Mirzaei, Riccardo De Lutio, Seung Wook Kim, David Acuna, Jonathan Kelly, Sanja Fidler, Igor Gilitschenski, and Zan Gojcic. Reffusion: Reference adapted diffusion models for 3d scene inpainting. *arXiv preprint arXiv:2404.10765*, 2024. 2, 3

[37] Daiki Miyake, Akihiro Iohara, Yu Saito, and Toshiyuki Tanaka. Negative-prompt inversion: Fast image inversion for editing with text-guided diffusion models. *arXiv preprint arXiv:2305.16807*, 2023. 2

[38] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *CVPR*, 2023. 2

[39] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016. 2

[40] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023. 2

[41] Kira Prabhu, Jane Wu, Lynn Tsai, Peter Hedman, Dan B Goldman, Ben Poole, and Michael Broxton. Inpaint3d: 3d scene content generation using 2d inpainting diffusion. *arXiv preprint arXiv:2312.03869*, 2023. 2

[42] Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. Langsplat: 3d language gaussian splatting. In *CVPR*, 2024. 2

[43] Ri-Zhao Qiu, Ge Yang, Weijia Zeng, and Xiaolong Wang. Language-driven physics-based scene synthesis and editing via feature splatting. In *ECCV*, 2024. 2

[44] 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. 3, 4, 6, 8, 12

[45] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 2

[46] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *CVPR*, 2019. 6

[47] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *CVPR*, 2020. 6

[48] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 6

[49] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*, 2016. 6

[50] I-Chao Shen, Hao-Kang Liu, and Bing-Yu Chen. Nerf-in: Free-form nerf inpainting with rgb-d priors. *Computer Graphics and Applications (CG&A)*, 2024. 3

[51] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021. 2, 5

[52] Chih-Hai Su, Chih-Yao Hu, Shr-Ruei Tsai, Jie-Ying Lee, Chin-Yang Lin, and Yu-Lun Liu. Boostmvsnerfs: Boosting mvs-based nerfs to generalizable view synthesis in large-scale scenes. In *ACM SIGGRAPH 2024 Conference Papers*, 2024. 2

[53] 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, 7, 12, 13

[54] Luming Tang, Nataniel Ruiz, Qinghao Chu, Yuanzhen Li, Aleksander Holynski, David E Jacobs, Bharath Hariharan, Yael Pritch, Neal Wadhwa, Kfir Aberman, et al. Realfill: Reference-driven generation for authentic image completion. *ACM TOG*, 2024. 2

[55] Shubham Tulsiani, Tinghui Zhou, Alexei A. Efros, and Jitendra Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *CVPR*, 2017. 2

[56] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2

[57] Can Wang, Ruixiang Jiang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Nerf-art: Text-driven neural radiance fields stylization. *IEEE TVCG*, 2023. 2

[58] Dongqing Wang, Tong Zhang, Alaa Abboud, and Sabine Süsstrunk. Inpaintnerf360: Text-guided 3d inpainting on unbounded neural radiance fields. In *CVPR*, 2024. 3, 7

[59] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul Srinivasan, Howard Zhou, Jonathan T. Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. IBRNet: Learning multi-view image-based rendering. In *CVPR*, 2021. 2

[60] Yuxin Wang, Qianyi Wu, Guofeng Zhang, and Dan Xu. Gscream: Learning 3d geometry and feature consistent gaussian splatting for object removal. In *ECCV*, 2024. 2, 3, 7, 14

[61] Ethan Weber, Aleksander Holynski, Varun Jampani, Saurabh Saxena, Noah Snavely, Abhishek Kar, and Angjoo Kanazawa. Nerfiller: Completing scenes via generative 3d inpainting. In *CVPR*, 2024. 3

[62] Silvan Weder, Guillermo Garcia-Hernando, Aron Monszpart, Marc Pollefeys, Gabriel Brostow, Michael Firman, and Sara Vicente. Removing objects from neural radiance fields. In *CVPR*, 2023. 3

[63] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *CVPR*, 2024. 2

[64] Bangbang Yang, Yinda Zhang, Yinghao Xu, Yijin Li, Han Zhou, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. Learning object-compositional neural radiance field for editable scene rendering. In *ICCV*, 2021. 2

[65] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *CVPR*, 2024. 2

[66] Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian grouping: Segment and edit anything in 3d scenes. In *ECCV*, 2024. 2, 3, 7, 8, 12, 14

[67] Youtan Yin, Zhoujie Fu, Fan Yang, and Guosheng Lin. Ornerf: Object removing from 3d scenes guided by multiview segmentation with neural radiance fields. *arXiv preprint arXiv:2305.10503*, 2023. 3

[68] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *CVPR*, 2021. 2

[69] 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. 4, 8, 12

[70] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *CVPR*, 2018. 2

[71] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. In *ICCV*, 2019. 2

[72] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020. 2

[73] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 5, 7

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 [44] 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. 15 for the 360-UISD dataset and in Fig. 16 for the other collected 360 dataset [3].

B. Training Masked GS for Object Removal

During the training of masked Gaussians, we use 2DGS [17] 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 [44] and the rasterized object mask for each training view. Additionally, we incorporate the Grouping Loss proposed by Gaussian Grouping [66], 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. 12. 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[44] prompts and more precise unseen masks (Fig. 13).

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

(a) RGB image at view n (b) Removal region R_n

(c) Removal region R_i ($i \neq n$)

(d) Unseen region obtained from view i at view n

(e) Aggregated unseen region   (f) Final contour of unseen region C_n
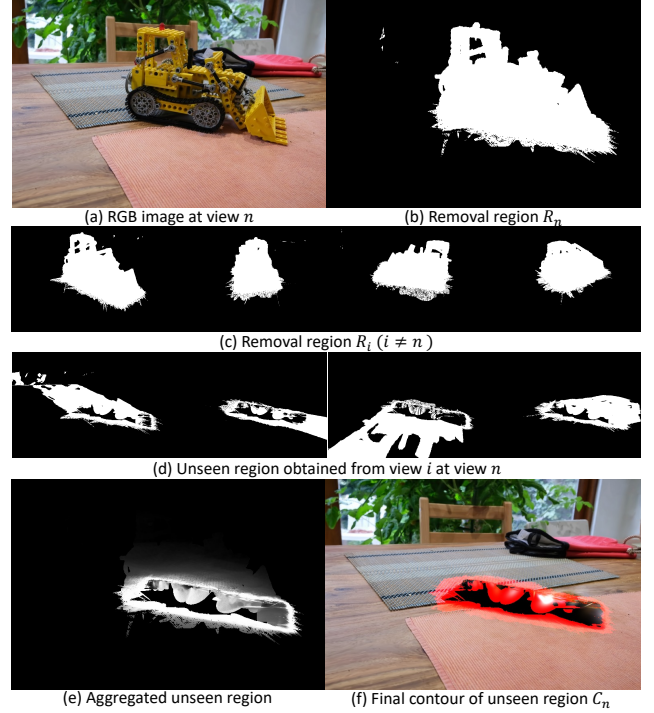
Figure 12. **Intermediate Results of Depth Warping for Unseen Region Detection.** This figure illustrates the intermediate results generated during the depth warping process. (a) and (b) show the RGB image and the corresponding removal region at view n, respectively. (c) displays the removal regions 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).

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 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 [53], InFusion [28], GDD [69], and AGDD for depth completion. As shown in Tab. 3, 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.
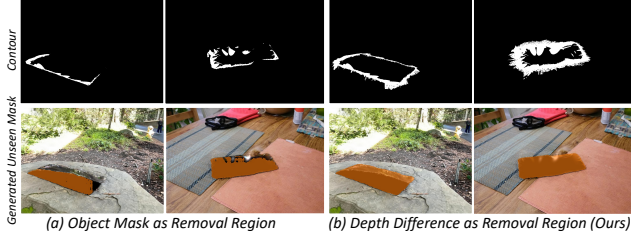
Figure 13. **Ablation Study on Removal Region Definition.** Comparison of (a) object masks vs. (b) depth difference for defining removal regions. Object masks fail to capture geometric changes, leading to less accurate unseen masks. Depth difference better preserves scene structure, improving SAM2 prompts and unseen region segmentation.

(a) With Object (b) 2DGS + LaMa (c) SPIn-NeRF (d) InFusion (e) Ours
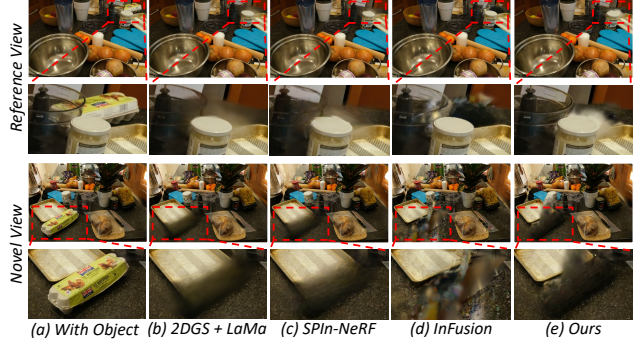
Figure 14. **Failure Cases.** The figure illustrates failure cases of inpainting results. These examples highlight the challenges of 3D inpainting when significant occlusions are present near the regions requiring inpainting. For instance, (b) and (c) demonstrate difficulties in achieving satisfactory 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.

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 3. 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. 4, 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 4. 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**

F. Experimetal Setup

F.1. LeftRefill [5]

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 [17] + LaMa [53]

We provide the same reference image and training view object masks as in our method and use LaMa [53] to obtain per-frame inpainting results for each training view to train the 2DGS.

F.3. 2DGS [17] + LeftRefill [5]

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 [35]

The original SPIn-NeRF [35] codebase is designed for forward-facing scenes; however, we adapt it for comparison on 360° scenes by implementing its approach on 2DGS [17]. We first obtain the depth for each training view by training a 2DGS model. Next, we generate inpainted RGB and depth maps using LaMa [53], 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
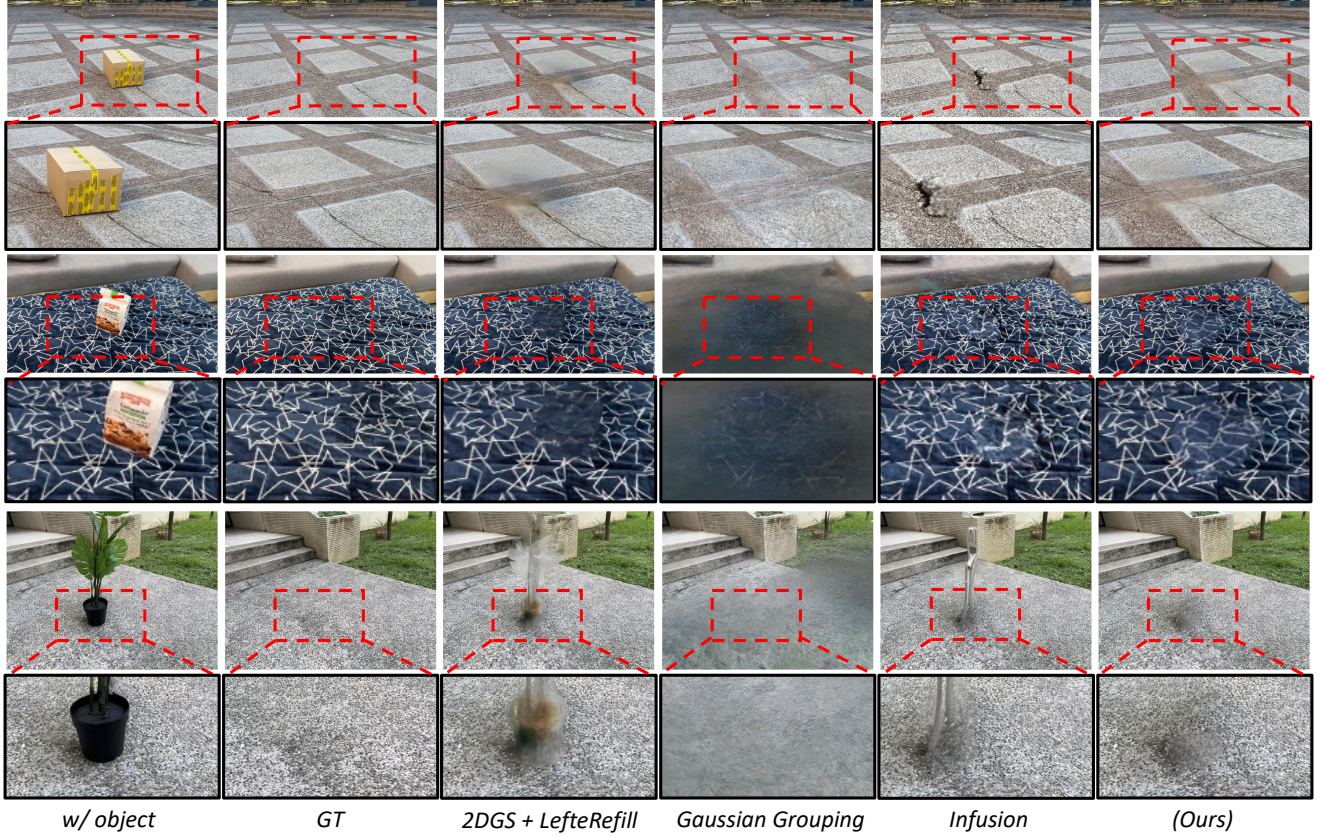
Figure 15. **Visual Comparison on our 360-USID dataset.**

| w/ object | GT | 2DGS + LefteRefill | Gaussian Grouping | Infusion | (Ours) |

compute a scale- and shift-invariant depth loss.

F.5. Gscream [60]

We follow the original GScream [60] 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 [19] to generate estimated depths for all training images, meeting GScream's input data requirements.

F.6. Gaussian Grouping [60]

We utilize the original Gaussian Grouping [66] 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 [8] 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 [3] 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 [28]

We use the original InFusion [28] 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 [31] to enhance the 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. 14 shows that the method may produce incorrect pixel unprojections in cases with significant occlusions near the object requiring inpainting, resulting in
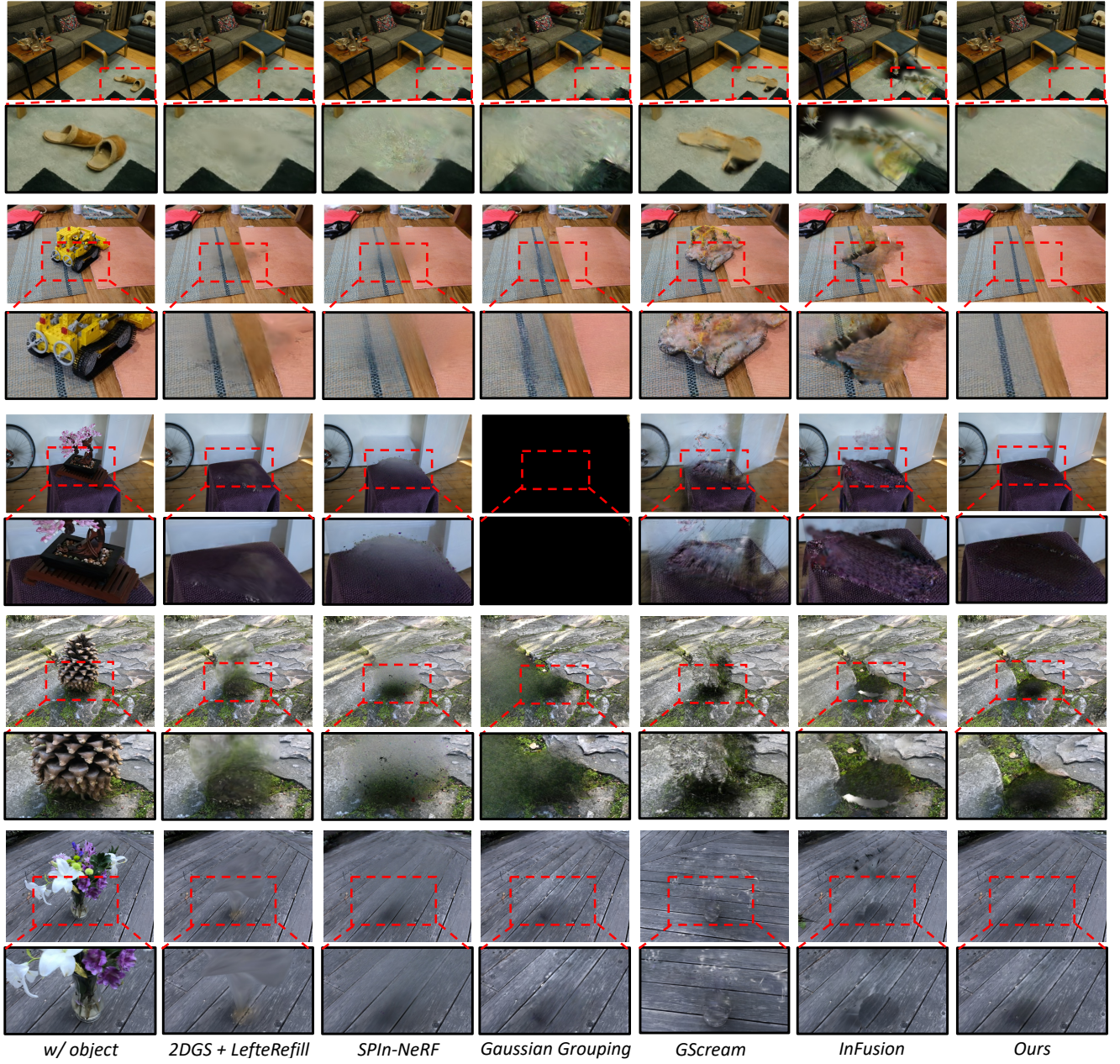
14

Figure 16. **Visual Comparison on Other-360 dataset.**

floaters in the final inpainted outputs. This limitation is similarly observed across all compared methods, underscoring a valuable direction for future research and improvement.