3D Gaussian Inpainting with Depth-Guided Cross-View Consistency

Supplementary Material

A. Additional Details of 3DGIC

A.1. Details of Backbone 3D Gaussian Splatting Model

Given the multi-view images $I_{1:K}$ with corresponding camera poses $\xi_{1:K}$ of a 3D scene, the vanilla 3DGS [5] model parameterize each Gaussian G_i in $G_{1:N}$ with its 3-dimensional centroid $\mathbf{p}_i \in \mathbb{R}^3$, a 3-dimensional standard deviation $\mathbf{s}_i \in \mathbb{R}^3$, a 4-dimensional rotational quaternion $\mathbf{q}_i \in \mathbb{R}^4$, an opacity $\alpha_i \in [0, 1]$, and color coefficients $\mathbf{c}_i$ for spherical harmonics in degree of 3. Hence, G_i is represented with a set of the above parameters (i.e., $G_i = \{\mathbf{p}_i, \mathbf{s}_i, \mathbf{q}_i, \alpha_i, \mathbf{c}_i\}$). However, to make sure the 3DGS models in this paper are capable of removing Gaussians corresponding to any indicated object (e.g., "bear" in Figure 2) as described in Sect. 3.3, we incorporate the use of a semantic-aware 3DGS (i.e., Gaussian Grouping [18]) approach as the main backbone 3DGS model of our method. Also, since the rendered depth maps $D_{1:K}$ are utilized as important guidance in our 3DGIC, we additionally combine the use of Relightable Gaussian [4], which produces better depth estimations from 3DGS model as our final backbone for Sect. 3. We now briefly discuss both methods.

Incorporating Semantic Segmentation via Gaussian Grouping. To overcome the lack of fine-grained scene understanding in 3DGS, Gaussian Grouping [18] extends 3DGS by incorporating segmentation capabilities. Along with $I_{1:K}$, Gaussian Grouping additionally takes the Segment Anything Model (SAM) to produce 2D semantic segmentation masks $S_{1:K} = \{S_1, S_2, ..., S_K\}$ from multiple views as inputs, and an additional 16-dimensional parameter $\mathbf{e}_i \in \mathbb{R}^{16}$ is introduced to represent a 3D Identity Encoding for each Gaussian G_i. Therefore, each Gaussian G_i is extended as $G_i = \{\mathbf{p}_i, \mathbf{s}_i, \mathbf{q}_i, \alpha_i, \mathbf{c}_i, \mathbf{e}_i\}$. To make sure $G_{1:K}$ learns to segment each object represented by $S_{1:K}$ in the scene, a 2D identity loss $\mathcal{L}_{id}$ is applied by calculating cross-entropy between $\hat{S}_{1:K}$ and $S_{1:K}$, where $\hat{S}_{1:K} = \{\hat{S}_1, \hat{S}_2, ..., S_K\}$ denotes the rendered segmentation maps from $G_{1:K}$. Additionally, to further ensure that the Gaussians having the same identities are grouped together, a 3D regularization loss $\mathcal{L}_{3D}$ is applied to enforce each G_i's k-nearest 3D spatial neighbors to be close in their feature distance of Identity Encodings. Please refer to the original paper [18] for detailed formulations of segmentation map rendering and $\mathcal{L}_{3D}$. The design of Gaussian Grouping ensures that the segmentation results are coherent across multiple views, enabling the automatic generation of binary masks for any queried object in the scene.

Produce Reliable Depth Estimations with Relightable Gaussians. Different from Gaussian Grouping, Relightable Gaussians [4] extends the capabilities of Gaussian Splatting by incorporating Disney-BRDF [2] decomposition and ray tracing to achieve realistic point cloud relighting. Unlike traditional Gaussian Splatting, which primarily focuses on appearance and geometry modeling, Relightable Gaussians also aim to model the physical interaction of light with different surfaces in the scene. Specifically, for each Gaussian G_i, the original color coefficients $\mathbf{c}_i$ is decomposed into a 3-dimensional base color $\mathbf{b}_i \in [0, 1]^3$, a 1-dimensional roughness $r \in [0, 1]$, and incident light coefficients $\mathbf{l}_i$ for spherical harmonics in degree of 3. Subsequently, the Physical-Based Rendering (PBR) process and a point-based ray tracing are applied to obtain the colored PBR 2D images $\hat{I}^{PBR}_{1:K}$ and additionally supervised by $I_{1:K}$. Besides the above extensions on PBR for relighting, Relightable Gaussians also introduces a 3-dimensional normal $\mathbf{n}_i$ for G_i and leverages several techniques, including an unsupervised estimation of a depth map D_i from each input view ξ_i, to enhance the geometry accuracy and smoothness. By conducting this self-supervised estimation and regularization of normal maps and depth maps, the predicted depth map D_i is more reliable than the vanilla 3DGS. Please refer to the original paper of Relightable Gaussians [4] for detailed explanations.

In conclusion, each Gaussian of our 3DGIC is parameterized as $G_i = \{\mathbf{p}_i, \mathbf{s}_i, \mathbf{q}_i, \alpha_i, \mathbf{c}_i, \mathbf{e}_i, \mathbf{b}_i, r, \mathbf{l}_i, \mathbf{n}_i\}$. By combining these methods, we are able to perform reliable depth estimations and effective removal of the Gaussians corresponding to any object in the scene for our 3DGIC.

A.2. Additional Details of Inferring Depth-Guided Inpainting Masks

In Sect. 2.2 in our main paper, we introduce infer proper inpainting masks $M'_{1:K}$ to determine the region to be inpaint by realizing visible background regions across different views. In our implementation, after updating the inpainting masks $M'_{1:K}$ with the process described in Sect. 3.2, we further conduct a refinement for each mask as a post-processing to prevent noisy mask. Taking M'_1 as an example, this process updates M'_1 as:

$$M'_1 \leftarrow Open(M'_1), \tag{1}$$

where $Open(\cdot)$ represents a morphological opening process to reduce noises. This refinement process ensures that small noisy pixels are suppressed in our Depth-Guided Inpainting Masks.

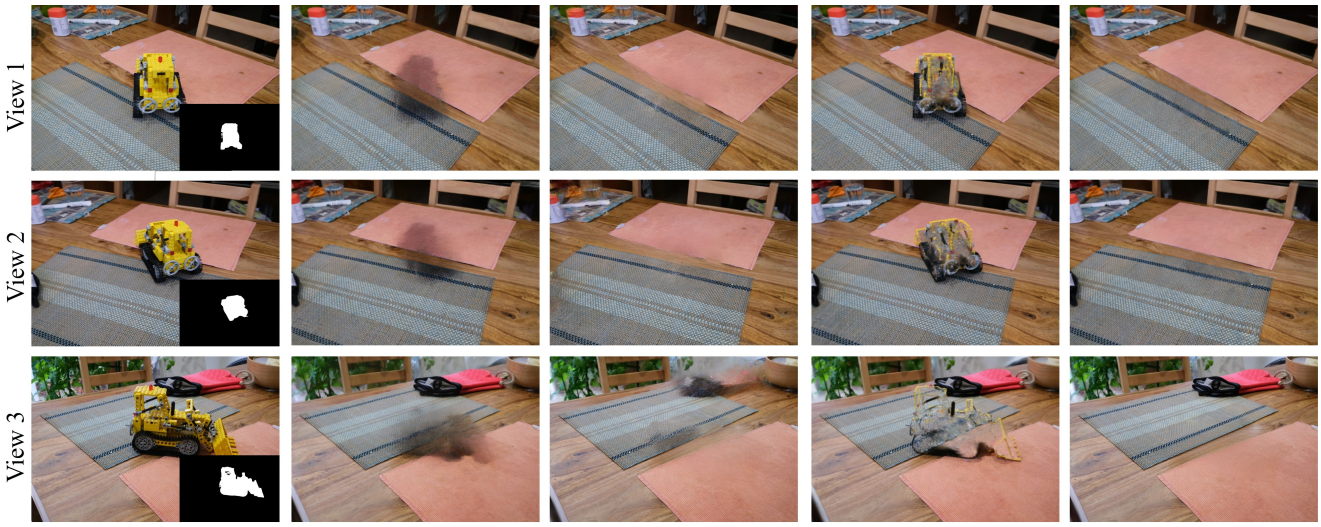

Figure A1. **Qualitative results on the *Kitchen* scene from the MipNeRF360 [1] dataset.** We compare the rendering results with SPIn-NeRF [9], Gaussian Grouping [18], and GScream [16]. The three rows show different views of the scene. We can see that our 3DGIC inpaint a smooth kitchen table, while other approaches produce blurry results.

A.3. Additional Details of Initializing Inpainted Gaussian

In Sect. 3.3, we introduce to remove the Gaussians with semantic labels corresponding to the "*bear*" object in $G_{1:N}$ and replace by the same amount of randomly initialized Gaussians in the masked region as the initialization of $G'_{1:N'}$. We now detail this initialization process for $G'_{1:N'}$.

When first removing the Gaussians corresponding to the "bear" object, we directly use the remaining Gaussian to render the image I'_1 and depth map D'_1. Following the 2D inpainting process described in Sect. 3.3, the inpainted image I_1^{In} and depth map D_1^{In} are produced and projected into 3D space as colored point clouds P_1. We then use the 3D coordinates of P_1 as the initialized 3D position for the newly introduced Gaussians for $G'_{1:N'}$, since P_1 represents the ideal surface of the inpainted 3D Gaussian provided by I_1^{In} after removing the bear. Note that if the number of points in P_1 does not match the number of newly initialized Gaussians in $G'_{1:N'}$ (also the number of removed Gaussians in $G_{1:N}$), we apply random selection to the coordinates of P_1 to match the number of the newly introduced Gaussians. As for the other parameters of the newly introduced Gaussians in $G'_{1:N'}$, we follow Gaussian Grouping [18] to average the parameters of each Gaussian's 5-nearest neighbors (in 3D space) from the remaining Gaussians as initialization. By this process, $G'_{1:N'}$ is properly initialized.

A.4. Implementation Details

In all our experiments, we train one model for each object category, using a single NVIDIA RTX 3090 GPU (24G) for training with the PyTorch [10] libraries. For each scene, 5000 iterations of optimization are applied to obtain the inpainted 3DGS model. We also use the official implementation of [3, 9, 16, 18] for comparison. When applying 2D inpainting models to the image and depth map to be inpaint, if we use non-diffusion-based LAMA [14] as inpainter, the RGB image and depth map are inpainted separately. However, if LDM [12] is applied as our 2D inpainters, we follow the suggestion in NeRFiller [17] to stack the RGB image and the depth map in the same image for inpainting to ensure the inpainted RGB image and the depth map are consistent in terms of the geometry details. Specifically, we crop a 512×512 patch for the RGB image and the depth map to be inpainted center at the pixel coordinate of the inpainting mask's center, and paste the cropped RGB patch to a 1024×1024-resolution black image at the upper right corner with the cropped depth map at the lower left corner as the input image for the LDM. Similarly, we also crop a 512×512 patch for the inpaint masks and put them to the upper right and lower left corner of another 1024×1024-resolution black image as the input binary inpainting mask for the LDM. We then use the prompt "*an RGB image and a depth image of the same scene*" to inpaint the input image. Finally, the inpainted RGB patch and the depth map patch are pasted back to the original image and depth map, respectively, as the 2D inpainting result. It is worth noting that we apply the 2D inpainting process for every 500 iterations. Following MALD-NeRF [8], we use the technique of partial DDIM [13], to start from latter step of the denoising process as optimization iteration grows. Specifically, for a 50-step

DDIM process, we start from step 0 of the LDM denoising process for step 0 of our optimization. After 500 iteration steps, the second time of the LDM inpainting starts from step 5 of the DDIM process and so on. When our optimization reaches the last 500 iterations, the 2D inpainting process only denoises using the last five steps of DDIM. This prevents inpainting results that are too different from the current scene and provides more stability for our optimization process.

A.5. Dataset Details

For the "figurines" scene from LeRF [6] dataset, we have 260 training frames and 40 testing frames, each with a resolution of 986×728. For the "bear" dataset from InNeRF360 [15], we have 90 training frames and 6 testing frames, each with a resolution of 985×729. As for "counter" and "kitchen" scenes from MipNeRF360 [1], 240 (230 for training and 10 for testing) and 279 (270 for training and 9 for testing) frames are available in total, respectively. Both scenes are in the resolution of 779.

Table A1. **Ablation studies on SPIn-NeRF.**

	FID↓	m-FID↓	LPIPS↓	m-LPIPS↓
Baseline	48.2	125.5	0.32	0.046
w/o Cross-view consistency	43.7	119.4	0.29	0.036
w/o Depth-guided mask	38.8	102.1	0.28	0.035
3DGIC (Ours)	**36.4**	**96.3**	**0.26**	**0.028**

B. Additional Experiments

B.1. Quantitative Ablation Studies

To verify our designed components, we conduct ablation studies on the SPIn-NeRF dataset in Table A1. Note that for the baseline model in this table, the original object mask (provided by SAM) is used and 2D images from all views are inpainted as input to fine-tune a 3DGS model. For the model "w/o Cross-view consistency", we use our Depth-guided mask described in Sect. 3.2 to locate regions to be inpainted but still inpaint all views separately. As for the model "w/o Depth-guided mask", we conduct the cross-view consistent refinement in Sect. 3.3 but take the original mask as inpainting mask. We can see that without our cross-view consistent refinement, both FID and LPIPS are much worse compared to our 3DGIC, especially for FID. This is because the inpainted results would be blurry and noisy in all views. On the other hand, without our Depth-guided mask introduces non-ideal LPIPS score although the FID score in close to ours. This is because the background regions that are originally observed from other views are not considered during the inpainting and therefore the inpainted results are not consistent to those backgrounds (although looks good visually). To this end, we have verified the design of our proposed components.

B.2. Additional Qualitative Results

We additionally show the results on the "*kitchen*" scene from the MipNeRF360 [1] dataset in Figure A1. We can see that SPIn-NeRF produces blurry result, while GScream fail to handle camera views with a wide range and not able to remove the excavator clearly. Although Gaussian Grouping also produces plausible results at the excavator-removed regions, it incorrectly detects the glove behind the excavator as region to be inpaint by using the "black blurry hole" as the prompt for Gounded-SAM [11] to find inpainting masks and therefore changes the background that should not be changed (shown in the third view). On the other hand, our 3DGIC locates the proper region to inpaint and produces smooth and high fidelity results.

C. Limitations

We now discuss the potential limitations of our 3DGIC. Since our 3DGIC uses the rendered depth map as guidance for the 3D inpainting process, the reliability of the rendered depth map becomes an important issue. As detailed in Sect. A.1, we combine the optimization technique introduced in Relightable Gaussians [4] to conduct a self-supervised loss for the predicted normal map and the rendered depth map to enhance the accuracy of the rendered depth map. However, if the input views are too sparse, the rendered depth map would not be guaranteed to be accurate, which hinders the inferring of Depth-Guided Mask and the achievement of cross-view consistency. Another potential limitation of our 3DGIC lies in the capability of the SAM [7] model. As detailed in Sect. A.1, we use SAM to produce 2D segmentation masks and use these masks as supervision for our backbone 3DGS model so that we don't have to manually annotate the 2D object mask of the object to be removed like SPIn-NeRF [9]. However, if the object to be removed is too small, the SAM model would confuse it with other objects and not produce the correct segmentation mask for the object. To overcome the above limitations, studies on the production of reliable depth maps for 3DGS models with only sparse input views and producing a more accurate segmentation mask for any object would be possible directions to improve the quality of 3D Gaussian inpainting.

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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 3

[2] Brent Burley and Walt Disney Animation Studios. Physically-based shading at disney. In *Acm Siggraph*, 2012. 1

[3] Honghua Chen, Chen Change Loy, and Xingang Pan. Mvip-nerf: Multi-view 3d inpainting on nerf scenes via diffusion prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 2

[4] Jian Gao, Chun Gu, Youtian Lin, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. Relightable 3d gaussian: Real-time point cloud relighting with brdf decomposition and ray tracing. *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 1, 3

[5] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (TOG)*, 2023. 1

[6] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2023. 3

[7] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2023. 3

[8] Chieh Hubert Lin, Changil Kim, Jia-Bin Huang, Qinbo Li, Chih-Yao Ma, Johannes Kopf, Ming-Hsuan Yang, and Hung-Yu Tseng. Taming latent diffusion model for neural radiance field inpainting. *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 2

[9] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2, 3

[10] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 2

[11] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, et al. Grounded sam: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024. 3

[12] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[13] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021. 2

[14] 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 *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2022. 2

[15] Dongqing Wang, Tong Zhang, Alaa Abboud, and Sabine Süsstrunk. Innerf360: Text-guided 3d-consistent object inpainting on 360-degree neural radiance fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 3

[16] Yuxin Wang, Qianyi Wu, Guofeng Zhang, and Dan Xu. Gscream: Learning 3d geometry and feature consistent gaussian splatting for object removal. *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 2

[17] Ethan Weber, Aleksander Holynski, Varun Jampani, Saurabh Saxena, Noah Snavely, Abhishek Kar, and Angjoo Kanazawa. Nerfiller: Completing scenes via generative 3d inpainting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20731–20741, 2024. 2

[18] Mingqiao Ye, Martin Danelljan, Fisher Yu, and Lei Ke. Gaussian grouping: Segment and edit anything in 3d scenes. *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 1, 2