A. Summary

We provide details of our proposed refinement process in Appendix B. In Appendix C, additional details about our implementation are provided for reproducibility. Appendix D contains further qualitative experiments showing the visual performance of our multiview segmentation and inpainting methods. We also provide a supplementary video and a website with video renderings of the scenes with and without inpainting for better visualization. In Appendix E, we provide an ablation study measuring the impact of additional training stages to segmentation performance. Appendix F provides an overview of all of the scenes in our introduced dataset. For completeness, we provide an extended version of the background on NeRFs in Appendix G. A detailed version of the segmentation results can be found in Appendix H. In Appendix I, we discuss potential failure cases of our model. Finally, due to the generative nature of inpainting, we provide an ethics statement in Appendix J.

B. Refinement Details

For pixel values that are only visible in some of the views, we use mask refinement to project them to all of the input views, as introduced in § 4.2.4 in the main paper. This refinement reduces the masked area and leads to better inpaintings due to a decreased need for hallucination. Consider a source image, I_s, its corresponding depth, D_s, and mask, M_s. For each target image, depth, and mask tuple, (I_t, D_t, M_t), and for every masked pixel in the source view, u_s, we consider the ray passing through u_s: $r_{u_s} = o_{u_s} + td_{u_s}$. The same sampling approach used in the original NeRF paper [35] is performed to sample $\{t_i\}_{i=1}^N$ on ray r_{u_s}. At the i-th step, the point represented by t_i is projected into the world coordinate system as:

$$X_i = G_s K^{-1} t_i u_s, \tag{10}$$

where G_s is the source camera pose and K is the camera intrinsic matrix. Next, point X_i is unprojected into the target views to determine which pixel in the target view corresponds to u_s [66]:

$$u_{t,i} = \pi(K G_t^{-1} X_i), \tag{11}$$

where G_t is the camera pose of the target view, and π stands for the perspective projection operation. If $u_{t,i}$ is masked in the target view, t_i is ignored and we go to t_{i+1}. If it is not masked, we check if the depth, $D_t(u_{t,i})$, is consistent with the distance of X_i to the target camera. In case of depth inconsistency, again, t_i is discarded and we proceed to t_{i+1}. If the depths are consistent, the RGB color $I_s(u_s)$ is replaced with $I_t(u_{t,i})$ while unmasking u_s in the source view. Note that for refining $D_s(u_s)$, one cannot directly use $D_t(u_{t,i})$ because it is the distance to the source camera. The depth

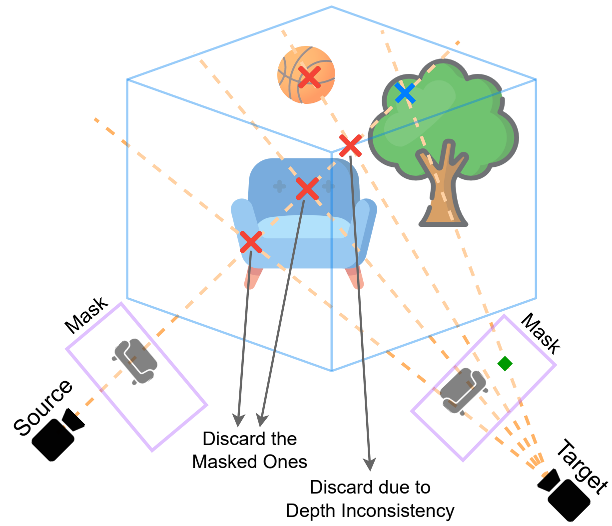

Figure 8. A visualization of our proposed mask refinement. The green pixel depicted in the target view is the final one that is used to transfer color and depth from the target view to the source view. Crosses represent the sampled points, and the blue cross is the final point used for the refinement in this example.

$D_t(u_{t,i})$ is first projected to the world coordinates similar to Eq. 10 as:

$$X_{\text{depth}} = G_t K^{-1} D_s(u_{t,i}) u_{t,i}. \tag{12}$$

The distance of X_{depth} to the source camera is then used to replace $D_s(u_s)$.

For each source image, we visit the target images one by one; if a pixel is able to be refined with respect to a new target image, the refinement is performed, and if a previously refined pixel is able to be refined with a point closer to the source camera, the refinement is updated. We iterate our refinement process multiple times, until no pixel is refined. This makes the process independent of the order of the target views.

Figure 8 shows a toy example to visualize the mask refinement process. The unwanted object is the sofa. For a source and target view, a masked pixel in the source view is considered, and a ray is passed through this pixel. The first two sampled points on this ray are still masked when unprojected into the target view since they fall on the sofa in the 3D world. The next sampled point is unprojected to an unmasked pixel on the target view, but the depth is inconsistent since the target camera sees the basketball from that pixel. Finally, the blue cross shows the fourth sampled point, where the depth is consistent, and the green pixel corresponding to the leaves of the tree is used to refine the source image. The distance of the blue cross to the source camera is used to replace the source depth. In practice, a source pixel is refined only if, after the refinement, the new depth is consistent with at least one of the eight neighbouring pixels in the source view. Figure 9 shows an example of an image from one of the scenes in our dataset, before and after refinement. We also provide corresponding masks to show the effect of our refinement process in reducing the masked area. Note that, following our other experiments in the main paper, the mask before refinement is dilated for five iterations, with a 5×5 kernel.

C. Additional Details

In practice, λ_{LPIPS} and λ_{depth} are set to 0.01 and 1, respectively. Our implementation is primarily in PyTorch [39], except for the encoders and MLP implementation, which use Tiny Cuda NN [37] for efficiency. The models are trained on a single Nvidia RTX A6000 GPU. We use the sparse depth supervision in the unmasked regions of the input views, as in DS-NeRF [10], to obtain more accurate scene geometries. Following Instant-NGP [3, 38], the multi-resolution hash encoder used in our NeRF has 16 levels, each returning two features. The base resolution is set to 16. The MLPs have 64-dimensional hidden layers. The first MLP, which calculates the density, σ (and "Objectness logit", s, for multiview segmentation), has two layers, while the color MLP has three layers. The training images used for our quantitative experiments have 567×1008 pixels (after being downsized four times to avoid memory issues), and all are captured by a Samsung Galaxy S20 FE. To calculate the perceptual loss, at each iteration, a random batch of four views is selected, and for each of them, a patch is rendered and compared to its inpainted counterpart in the perceptual space. Each patch is 16 times smaller than the original image in each direction, while the stride for sampling the patches is set to two to cover larger areas. This makes the perceptual loss more meaningful, without slowing down the training. As mentioned in the paper, FID and LPIPS are calculated only for the bounding box of the masked region. The mask for test views is rendered using our multiview segmentation model, because the test views do not contain the object and can not be manually masked. Since in the experiments, masks are sometimes dilated, we also expand each side of the bounding box containing the mask in every direction by 10% to make sure that in all of the experiments, the entire hallucinated region is being evaluated. Note that, for NeRF fitting, the object masks are slightly dilated (for five iterations with a 5×5 kernel) to reduce the effects of the shadow of the target objects in the inpainted scene and to make sure that the mask covers all of the object.

Dataset. All scenes in our dataset are forward-facing, and obtained by manually moving a camera using an unstructured trajectory mimicking the behavior of a non-expert user. We focus on forward-facing scenes due in part to the fact that the inpainting task is more challenging, due to a lower chance to see behind objects and thus a need for more hallucination compared to 360° scenes. All the $60 + 40$ images are jointly processed with Colmap to recover the camera parameters in a shared coordinate system. Each image is 2268×4032 pixels in size.

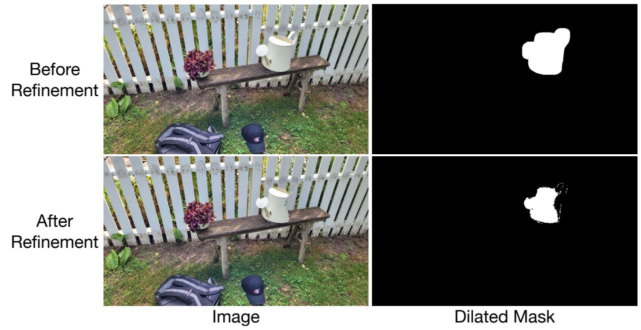

Figure 9. Qualitative example of how refinement can reduce the masked area by substituting pixel values from other views.

C.1. Approximate Timings

In Table 5, we provide the approximate times that each stage in our framework takes. We use a similar architecture to Instant-NGP [38], which yields fast convergence for our models. Note that the semantic NeRF typically converges to an acceptable geometry even half-way through the fitting iterations, and the remaining iterations are mostly for obtaining a sharp appearance. Since our segmentation and inpainting approaches only use the rendered masks and depths from the semantic NeRF, according to the application, one can trade off quality for speed, and early stop the semantic NeRF to further reduce the segmentation time. For fitting the inpainted NeRF, since we have to render multiple patches and calculate the perceptual loss for each of them, the entire process is slower than the segmentation part. However, according to the fitting times in the literature, this is still a fast NeRF manipulation model for realistic scenes. Note that all of these times can be reduced in the future with faster hardware and underlying models, e.g., better differentiable scene representations.

D. Additional Qualitative Results

Here, we provide additional qualitative examples to show the effectiveness of our multiview segmentation and multiview inpainting methods. Figure 10 is an extended version of Figure 6, and shows four additional qualitative examples of our view-consistent inpainting approach.

Figure 11 shows an example of a single scene being inpainted twice, each time with a different part of the scene being masked. In the upper case, the statue without its concrete base is selected and the base is still in the scene after the inpainting. Notice that parts of the base as well as parts of the ground behind it were not visible in any of the

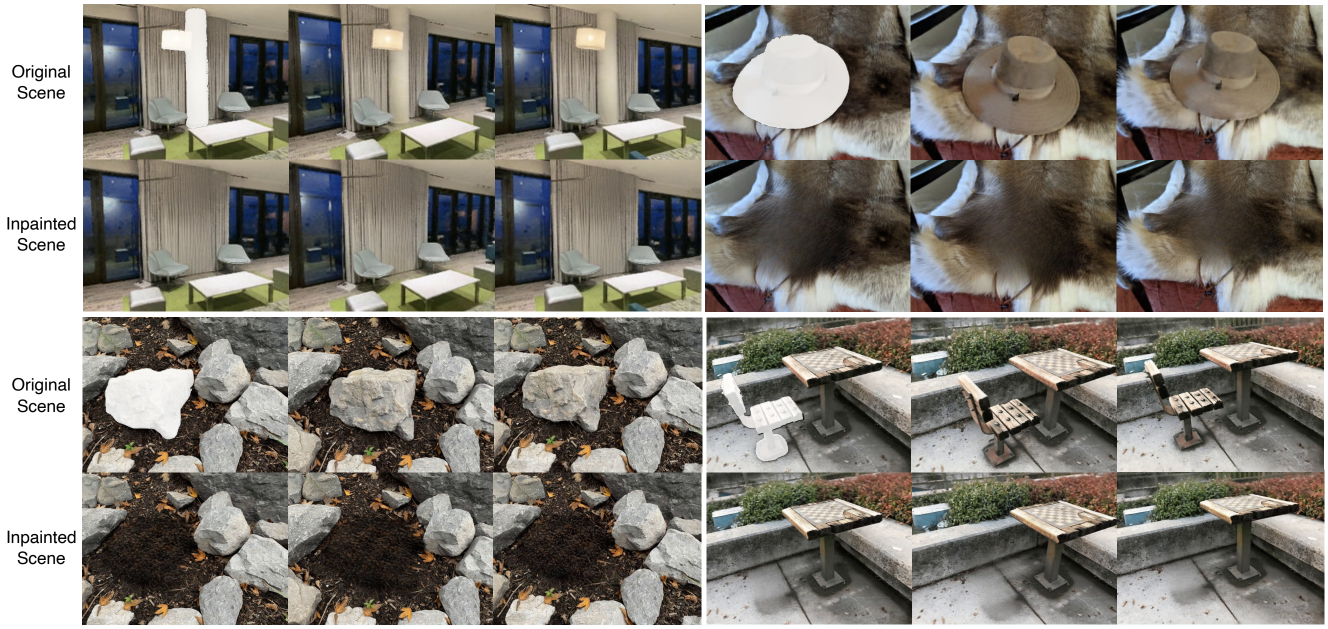

Figure 10. Additional qualitative visualizations of our view-consistent inpainting results, as in Figure 6 in the main paper. Upper rows per inset show NeRF renderings of the original scene from novel views, with the first image also displaying the associated mask. Lower rows show the corresponding inpainted view.

Table 5. Approximate times that each of the stages in our multiview segmentation and multiview inpainting framework take. These numbers do not include the time spent for human-annotations.

Stage Name	**Time**
Multiview Segmentation	
Interactive Segmentation	< 1 second
Video Segmentation	< 1 minute
Fitting the Semantic NeRF	$2 - 5$ minutes
Rendering Training Masks	1 minute
Multiview Inpainting	
Applying the Image Inpainter	< 1 minute
Fitting the Inpainted NeRF	$20 - 40$ minutes

training views. Our model shows consistent plausible hallucinations, which complete the cylinder shape of the base. The use of the perceptual loss leads to a sharp texture on the grass.

We further provide more qualitative results of our multiview segmentation model. Figure 12 is an extension of Figure 5, and shows target views from two scenes, the ground-truth mask in the target views, and the outputs of NVOS [44], video segmentation [4], and our model with or without the two-stage training. As evident in the results, our segmentation model consistently provides coherent masks with sharp accurate edges (zoom into boxes for examples).

Figure 13 shows additional qualitative comparisons of our model against NeRF-In [31] on three of the scenes of our dataset. As visible in the outputs, our models is able to produce sharper outputs.

E. Multi-Stage Multiview Segmentation

While it has been shown both qualitatively (Figure 5) and quantitatively (Table 1) that our multiview segmentation benefits from our proposed two-stage training, Figure 14 shows that additional training stages do not have a significant effect on the outputs, and thus, two training stages are sufficient. Quantitatively, Table 6 shows that our model with two or three stages of training has similar performance.

F. Our Multiview Inpainting Dataset

Figure 15 contains sample images from our introduced dataset used in our quantitative evaluations. This dataset contains 10 real-world scenes and includes different challenging 3D inpainting segmentation and inpainting scenarios. In the experiments, we use this dataset to provide a quantitative comparison of our inpainting method against the baselines, where our approach outperforms other methods.

G. NeRF: Extended Background

Here, we provide an extended version of the background on Neural Radiance Fields (NeRFs) for complete-

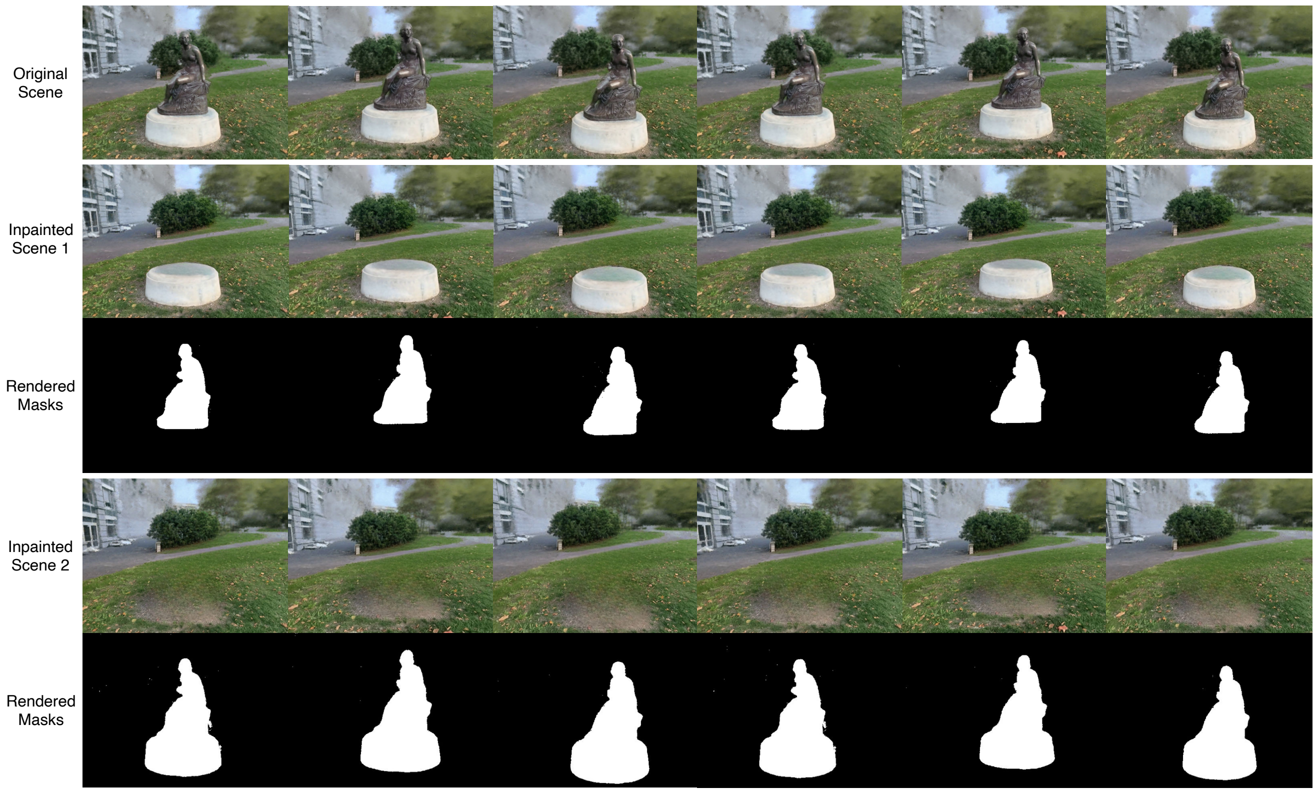

Figure 11. A single scene inpainted with two different masks using our multiview inpainting method.

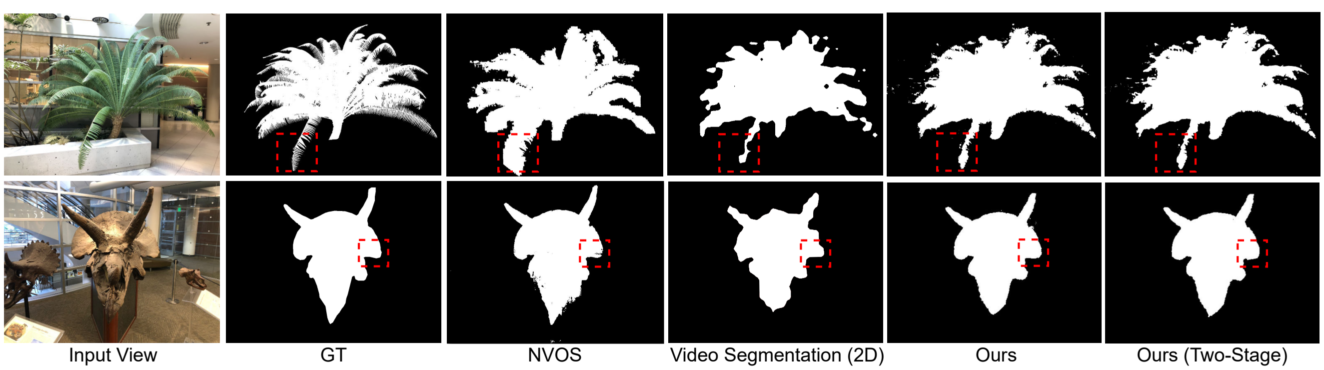

Figure 12. Qualitative comparison, as in Figure 5 in the main paper, of our multiview segmentation model against Neural Volumetric Object Selection (NVOS) [44], Video segmentation [4], and the human-annotated masks (GT).

Table 6. Quantitative evaluation of our proposed multiview segmentation with one, two, and three training stages.

# of Stages	Acc.↑	IoU↑
1	98.85	90.96
2	**98.91**	**91.66**
3	98.89	91.53

ness. NeRFs [35] encode a 3D scene as a function, $f : (x, d) \rightarrow (c, \sigma)$, that maps a 3D coordinate, x, and a view direction, d, to a color, c, and density, σ. The function f can be modelled in various ways, such as a multilayer perceptron (MLP) with positional encoding [35] or a discrete voxel grid with trilinear interpolation [47], depending on the application and desired properties. For a 3D ray, r, characterized as $r(t) = o + td$, where o denotes the ray's origin, d its direction, and t_n and t_f the near and far bounds, respectively, the expected color is:

$$C(r) = \int_{t_n}^{t_f} T(t)\sigma(r(t))c(r(t), d)\,\mathrm{d}t, \tag{13}$$

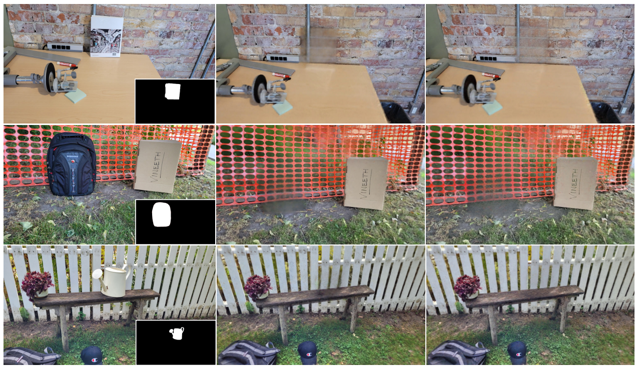

Figure 13. Additional qualitative comparisons of our model against NeRF-In [31].

where $T(t) = \exp(-\int_{t_n}^{t} \sigma(r(s))\,\mathrm{d}s)$ is the transmittance. The integral in Eq. 13 is estimated via quadrature by dividing the ray into N sections and sampling t_i from the i-th section:

$$\widehat{C}(r) = \sum_{i=1}^{N} T_i(1 - \exp(-\sigma_i \delta_i))c_i, \tag{14}$$

where $T_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j)$ and $\delta_i = t_{i+1} - t_i$ is the distance between two adjacent sampled points. For simplicity, $c(r(t_i), d)$ and $\sigma(r(t_i))$ are abbreviated as c_i and σ_i, respectively. For the rays passing through pixels of the training views, the ground-truth color, $C_{\text{GT}}(r)$, is available, and the representation is optimized using the reconstruction loss:

$$\mathcal{L}_{\text{rec}} = \sum_{r \in \mathcal{R}} \|\widehat{C}(r) - C_{\text{GT}}(r)\|^2, \tag{15}$$

where $\mathcal{R}$ is a ray batch sampled from the training views.

H. Detailed Segmentation Results

Table 7 shows a breakdown of Table 1 based on forward-facing and 360° scenes. The inputs to all of the models in this experiment is a single-view mask, which is to be transferred to other views. As a result, the task is more challenging for 360° scenes, due to the need to extrapolate the single-view mask to further views. Regardless of the differences in difficulty, our model consistently outperforms the baselines in both forward-facing and 360° scenarios (Table 7).

I. Failure Cases

Since SPIn-NeRF is based on an underlying NeRF and a 2D inpainter, it is prone to the failure cases of these models; e.g., the image inpainter failing results in the failure of SPIn-NeRF as well. Moreover, despite the effectiveness of the perceptual loss in handling texture-level inconsistencies between the image priors, potential semantic-level inconsistencies can result in failure. For instance, if some inpainted views contain novel inserted objects in the masked region (in contrast to simply extending the background to remove the unwanted object, as our method expects), the perceptual loss might fail to converge to a meaningful solution. In particular, as the resulting independently inpainted patches would not reside nearby in the perceptual metric space, the NeRF output (attempting to balance between them) in the masked area would likely be blurry or contain other artifacts. Due in part to this consideration, we utilize LaMa [48] as our underlying inpainter, as it reduces the likelihood of this scenario, since LaMa is not a "creative" inpainter and typically only removes objects. However, such problematic cases are likely with more creative inpainters, such as non-deterministic denoising diffusion-based inpainters.

Table 7. Quantitative multi-view segmentation evaluation for forward-facing and 360° scenes. See also Table 1.

	Forward-Facing		**360°**	
	Acc.↑	**IoU↑**	**Acc.↑**	**IoU↑**
Proj. + Grab Cut (2D)	92.19	59.84	89.54	28.09
Proj. + EdgeFlow (2D)	97.63	87.00	95.73	74.10
Semantic NeRF (only source mask)	98.72	90.96	88.90	52.98
Proj. + EdgeFlow + Semantic NeRF	98.74	91.53	95.20	73.35
Feature Field Distillation	98.20	85.61	96.19	79.51
Video Segmentation	98.87	91.38	97.81	84.08
Ours (two-stage)	**99.29**	**94.64**	**98.37**	**87.48**

J. Ethics Statement

There has been a constant debate about 2D generative models and image manipulation techniques, and the concerns regarding potential misuses. The majority of these concerns also apply to the new line of 3D generation and manipulation [41]. In the hands of an adversary, these models can be utilized to manipulate people's perception of reality and generate disinformation. Moreover, the fact that LaMa [48] is used in our implementation results in the inheritance of LaMa's potential undesirable biases in the outputs of our 3D inpainter.

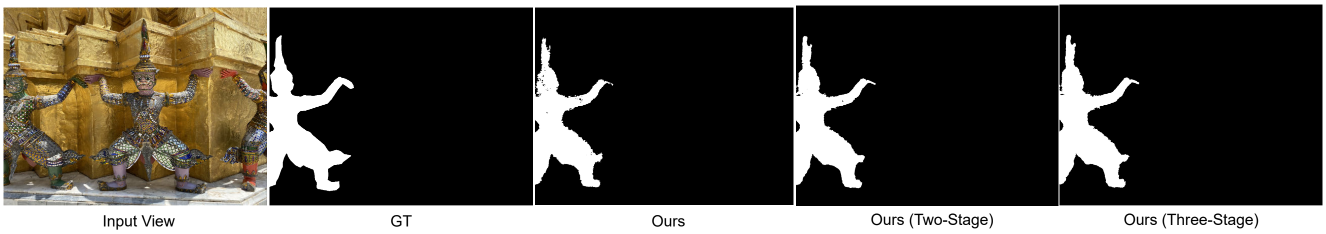

Figure 14. Qualitative comparison of our multiview segmentation model with two-stage and three-stage optimizations. As evident in the results, three-stage optimization does not lead to a significant improvement over the two-stage fitting.

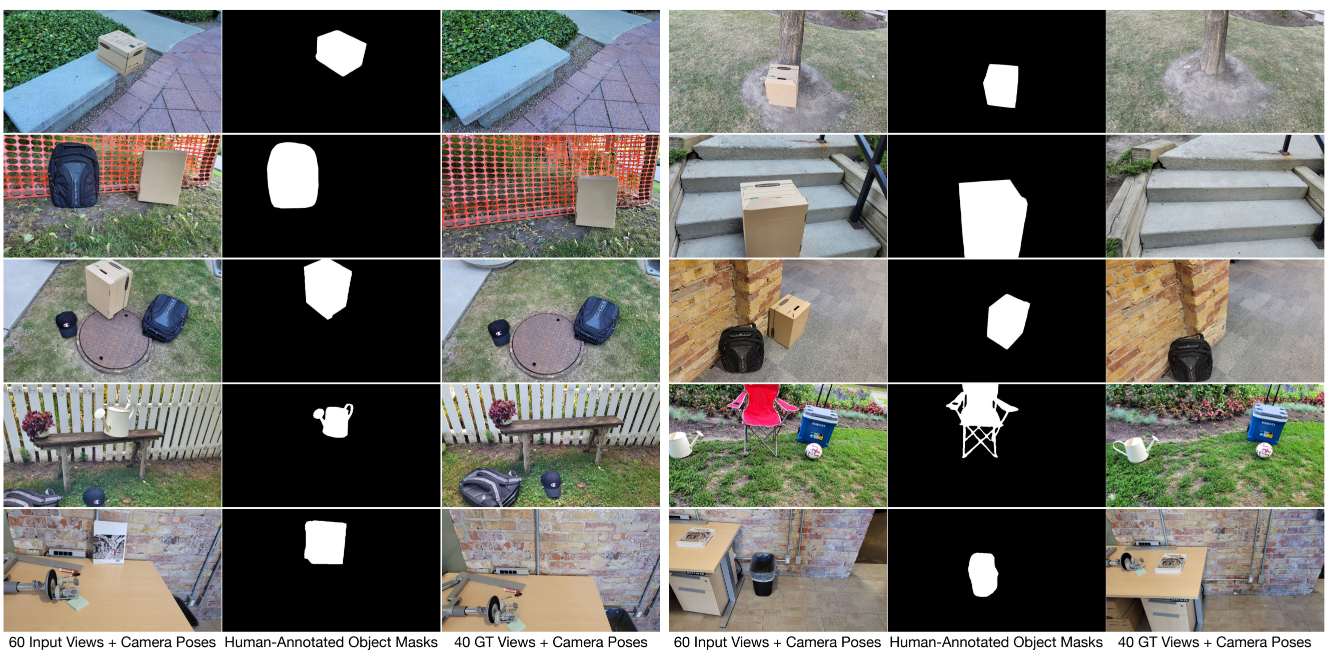

Figure 15. Overview of the 10 different scenes in our introduced dataset for multiview inpainting.

References

[1] 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. *ICCV*, 2021. 2

[2] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. *CVPR*, 2022. 2

[3] Yash Bhalgat. Hashnerf-pytorch. https://github.com/yashbhalgat/HashNeRF-pytorch/, 2022. 4, 13

[4] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021. 2, 3, 6, 7, 14, 15

[5] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *ECCV*, 2022. 2

[6] Zhiqin Chen, Thomas Funkhouser, Peter Hedman, and Andrea Tagliasacchi. MobileNeRF: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures. In *arXiv*, 2022. 1, 2

[7] Lu Chi, Borui Jiang, and Yadong Mu. Fast Fourier convolution. In *NeurIPS*, 2020. 2

[8] PaddlePaddle Contributors. PaddleSeg, end-to-end image segmentation kit based on PaddlePaddle. https://github.com/PaddlePaddle/PaddleSeg, 2019. 2

[9] Antonio Criminisi, Patrick Pérez, and Kentaro Toyama. Object removal by exemplar-based inpainting. In *CVPR*, 2003. 2

[10] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised NeRF: Fewer views and faster training for free. In *CVPR*, June 2022. 2, 13

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 2

[12] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, 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] Niv Granot, Ben Feinstein, Assaf Shocher, Shai Bagon, and Michal Irani. Drop the GAN: In defense of patches nearest neighbors as single image generative models. In *CVPR*, 2022. 5

[15] Yuying Hao, Yi Liu, Zewu Wu, Lin Han, Yizhou Chen, Guowei Chen, Lutao Chu, Shiyu Tang, Zhiliang Yu, Zeyu Chen, and Baohua Lai. Edgeflow: Achieving practical interactive segmentation with edge-guided flow. In *ICCV Workshops*, 2021. 2, 3, 6

[16] James Hays and Alexei A Efros. Scene completion using millions of photographs. In *SIGGRAPH*, 2007. 2

[17] Peter Hedman, Pratul P. Srinivasan, Ben Mildenhall, Jonathan T. Barron, and Paul Debevec. Baking neural radiance fields for real-time view synthesis. In *ICCV*, 2021. 2

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

[19] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. *NeurIPS*, 2017. 5

[20] Tao Hu, Shu Liu, Yilun Chen, Tiancheng Shen, and Jiaya Jia. EfficientNeRF: Efficient neural radiance fields. In *CVPR*, 2022. 1

[21] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. *ToG*, 2017. 2

[22] Jitesh Jain, Yuqian Zhou, Ning Yu, and Humphrey Shi. Keys to better image inpainting: Structure and texture go hand in hand. In *WACV*, 2023. 2

[23] 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

[24] Varun Jampani, Huiwen Chang, Kyle Sargent, Abhishek Kar, Richard Tucker, Michael Krainin, Dominik Kaeser, William T Freeman, David Salesin, Brian Curless, and Ce Liu. Slide: Single image 3D photography with soft layering and depth-aware inpainting. In *ICCV*, 2021. 2

[25] Kacper Kania, Kwang Moo Yi, Marek Kowalski, Tomasz Trzciński, and Andrea Tagliasacchi. CoNeRF: Controllable neural radiance fields. In *CVPR*, 2022. 2

[26] Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann. Decomposing NeRF for editing via feature field distillation. In *NeurIPS*, 2022. 6

[27] Wenbo Li, Zhe Lin, Kun Zhou, Lu Qi, Yi Wang, and Jiaya Jia. MAT: Mask-aware transformer for large hole image inpainting. In *CVPR*, 2022. 2

[28] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. BARF: Bundle-adjusting neural radiance fields. In *ICCV*, 2021. 2

[29] David B Lindell, Dave Van Veen, Jeong Joon Park, and Gordon Wetzstein. Bacon: Band-limited coordinate networks for multiscale scene representation. In *CVPR*, 2022. 2

[30] Hongyu Liu, Bin Jiang, Yi Xiao, and Chao Yang. Coherent semantic attention for image inpainting. In *ICCV*, 2019. 2

[31] Hao-Kang Liu, I-Chao Shen, and Bing-Yu Chen. NeRF-In: Free-form NeRF inpainting with RGB-D priors. In *arXiv*, 2022. 3, 6, 7, 14, 16

[32] Steven Liu, Xiuming Zhang, Zhoutong Zhang, Richard Zhang, Jun-Yan Zhu, and Bryan Russell. Editing conditional radiance fields. In *ICCV*, 2021. 2

[33] Yi Liu, Lutao Chu, Guowei Chen, Zewu Wu, Zeyu Chen, Baohua Lai, and Yuying Hao. PaddleSeg: A high-efficient development toolkit for image segmentation. In *arXiv*, 2021. 2

[34] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ToG*, 2019. 5

[35] 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, 3, 4, 5, 6, 7, 12, 15

[36] Ashkan Mirzaei, Yash Kant, Jonathan Kelly, and Igor Gilitschenski. LaTeRF: Label and text driven object radiance fields. In *ECCV*, 2022. 2, 3, 4

[37] Thomas Müller. Tiny CUDA neural network framework, 2021. https://github.com/nvlabs/tiny-cuda-nn. 4, 13

[38] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *TOG*, 2022. 2, 4, 13

[39] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 13

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

[41] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3D using 2D diffusion. In *arXiv*, 2022. 16

[42] Hiba Ramadan, Chaymae Lachqar, and Hamid Tairi. A survey of recent interactive image segmentation methods. *CVM*, 2020. 2

[43] Chaolin Rao, Huangjie Yu, Haochuan Wan, Jindong Zhou, Yueyang Zheng, Yu Ma, Anpei Chen, Minye Wu, Binzhe Yuan, Pingqiang Zhou, Xin Lou, and Jingyi Yu. Icarus: A specialized architecture for neural radiance fields rendering. In *arXiv*, 2022. 1

[44] Zhongzheng Ren, Aseem Agarwala, Bryan Russell, Alexander G. Schwing, and Oliver Wang. Neural volumetric object selection. In *CVPR*, 2022. 2, 6, 7, 14, 15

[45] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 2012. 6

[46] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. SinGAN: Learning a generative model from a single natural image. In *ICCV*, 2019. 5

[47] Sara Fridovich-Keil and Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *CVPR*, 2022. 2, 3, 15

[48] 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*, 2022. 2, 3, 4, 5, 6, 7, 16

[49] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *NeuRIPS*, 2020. 2

[50] Ayush Tewari, Justus Thies, Ben Mildenhall, Pratul Srinivasan, Edgar Tretschk, Yifan Wang, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, Tomas Simon, Christian Theobalt, Matthias Niessner, Jonathan T. Barron, Gordon Wetzstein, Michael Zollhoefer, and Vladislav Golyanik. Advances in neural rendering. In *SIGGRAPH*, 2021. 1, 2

[51] Vadim Tschernezki, Iro Laina, Diane Larlus, and Andrea Vedaldi. Neural Feature Fusion Fields: 3D distillation of self-supervised 2D image representations. In *3DV*, 2022. 6

[52] 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

[53] 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

[54] Suhani Vora, Noha Radwan, Klaus Greff, Henning Meyer, Kyle Genova, Mehdi S. M. Sajjadi, Etienne Pot, Andrea Tagliasacchi, and Daniel Duckworth. NeSF: Neural semantic fields for generalizable semantic segmentation of 3D scenes. In *TMLR*, 2022. 7

[55] Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. CLIP-NeRF: Text-and-image driven manipulation of neural radiance fields. *CVPR*, 2022. 2

[56] 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, 4, 5

[57] Wenguan Wang, Tianfei Zhou, Fatih Porikli, David Crandall, and Luc Van Gool. A survey on deep learning technique for video segmentation. In *arXiv*, 2021. 3, 6

[58] Yi Wang, Xin Tao, Xiaojuan Qi, Xiaoyong Shen, and Jiaya Jia. Image inpainting via generative multi-column convolutional neural networks. In *NeurIPS*, 2018. 2

[59] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. NeRF–: Neural radiance fields without known camera parameters. In *arXiv*, 2021. 2

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

[61] Suttisak Wizadwongsa, Pakkapon Phongthawee, Jiraphon Yenphraphai, and Supasorn Suwajanakorn. NeX: Real-time view synthesis with neural basis expansion. In *CVPR*, 2021. 5

[62] Chaohao Xie, Shaohui Liu, Chao Li, Ming-Ming Cheng, Wangmeng Zuo, Xiao Liu, Shilei Wen, and Errui Ding. Image inpainting with learnable bidirectional attention maps. In *ICCV*, 2019. 2

[63] Zhaoyi Yan, Xiaoming Li, Mu Li, Wangmeng Zuo, and Shiguang Shan. Shift-net: Image inpainting via deep feature rearrangement. In *ECCV*, 2018. 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, 6, 7

[65] Shunyu Yao, Tzu Ming Hsu, Jun-Yan Zhu, Jiajun Wu, Antonio Torralba, Bill Freeman, and Josh Tenenbaum. 3D-aware scene manipulation via inverse graphics. *NeuRIPS*, 2018. 2

[66] Lin Yen-Chen, Pete Florence, Jonathan T. Barron, Tsung-Yi Lin, Alberto Rodriguez, and Phillip Isola. NeRF-Supervision: Learning dense object descriptors from neural radiance fields. In *ICRA*, 2022. 5, 12

[67] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. PlenOctrees for real-time rendering of neural radiance fields. In *ICCV*, 2021. 2

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

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

[70] Yu-Jie Yuan, Yang-Tian Sun, Yu-Kun Lai, Yuewen Ma, Rongfei Jia, and Lin Gao. NeRF-editing: geometry editing of neural radiance fields. In *CVPR*, 2022. 2

[71] Kai Zhang, Nick Kolkin, Sai Bi, Fujun Luan, Zexiang Xu, Eli Shechtman, and Noah Snavely. ARF: Artistic radiance fields. In *ECCV*, 2022. 5

[72] 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. 2, 3, 4, 5

[73] Shengyu Zhao, Jonathan Cui, Yilun Sheng, Yue Dong, Xiao Liang, Eric I Chang, and Yan Xu. Large scale image completion via co-modulated generative adversarial networks. In *ICLR*, 2021. 2

[74] Yunhan Zhao, Connelly Barnes, Yuqian Zhou, Eli Shechtman, Sohrab Amirghodsi, and Charless Fowlkes. Geofill: Reference-based image inpainting of scenes with complex geometry. In *arXiv*, 2022. 2

[75] Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. Pluralistic image completion. In *CVPR*, 2019. 2

[76] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew Davison. In-place scene labelling and understanding with implicit scene representation. In *ICCV*, 2021. 2, 3, 6

[77] Shuaifeng Zhi, Edgar Sucar, Andre Mouton, Iain Haughton, Tristan Laidlow, and Andrew J. Davison. iLabel: Interactive neural scene labelling. In *arXiv*, 2021. 2, 3