Received: 9 March 2024 | Revised: 4 July 2024 | Accepted: 10 July 2024

DOI: 10.1049/ipr2.13190

ORIGINAL RESEARCH

Point'n Move: Interactive scene object manipulation on Gaussian splatting radiance fields

Jiajun Huang | Hongchuan Yu | Jianjun Zhang | Hammadi Nait-Charif

National Centre for Computer Animation, Fern Barrow, Poole, Dorset, UK

Correspondence
Hongchuan Yu, National Centre for Computer Animation, Fern Barrow, Poole, BH12 5BB, Dorset, United Kingdom.
Email: hyu@bournemouth.ac.uk

Funding information
HORIZON EUROPE Marie Sklodowska-Curie Actions, Grant/Award Number: 101130271; EPSRC Centre for Doctoral Training in Digital Entertainment, Grant/Award Number: EP/L016540/1

Abstract
The authors propose Point'n Move, a method that achieves interactive scene object manipulation with exposed region inpainting. Interactivity here further comes from intuitive object selection and real-time editing. To achieve this, Gaussian Splatting Radiance Field is adopted as the scene representation and its explicit nature and speed advantage are fully leveraged. Its explicit representation formulation allows to devise a 2D prompt points to 3D masks dual-stage self-prompting segmentation algorithm, perform mask refinement and merging, minimize changes, and provide good initialization for scene inpainting and perform editing in real-time without per-editing training; all lead to superior quality and performance. The method was tested by editing both forward-facing and 360 scenes. The method is also compared against existing methods, showing superior quality despite being more capable and having a speed advantage.

1 | INTRODUCTION

Since the introduction of Neural Radiance Fields (NeRF) [1], it has garnered significant interest in various fields, such as computer vision, computer graphics, and AI, primarily as an image-based scene reconstruction technique. While the development of NeRF's potential for scene editing appears promising, current approaches tend to focus more on deformation and object-centric modifications rather than on the intuitive selection and manipulation of objects within scenes.

Editable representations produced by methods like Control-NeRF [2], NeuralEditor [3] and NeuMesh [4], despite their advancements, overlook the critical aspect of naturally selecting and manipulating objects captured within a scene. While techniques like OR-NeRF [5] and SPIn-NeRF [6] have targeted object selection, they fall short in facilitating comprehensive, unconstrained manipulation. Furthermore, these techniques typically do not address the challenge of inpainting newly exposed surfaces or filling holes created by the editing process, which can result in renderings that are perceptibly unrealistic.

State-of-the-art segmentation techniques, exemplified by SAGA, are adept at delivering high-quality segmentation masks. However, there often are distinct artefacts left due to manipulation (see our comparison in Figure 5).

To address these challenges, we introduce the 3D Gaussian Splatting Radiance Field (3DGS) [7] as the foundational scene representation, coupled with Point'n Move, our novel approach for real-time interactive manipulation of scene objects. Our method facilitates intuitive object selection, provides 3D semantic segmentation, and offers effective inpainting capabilities. 3DGS possesses considerable advantages. Compared to existing neural implicit representations such as NeRF [1], 3DGS concretely utilizes a multitude of 3D ellipsoids to represent the radiance field. This explicit representation leads to faster training and enables real-time rendering while achieving remarkable visual quality. The point cloud-like nature inherent in 3DGS allows us to approach segmentation and refinement from the perspective of point clouds. Furthermore, the rapid rendering of 3DGS presents a significant speed benefit, particularly evident in 3D scene inpainting tasks. Our goal is to facilitate real-time scene editing by directly manipulating selected primitives in the scene, as opposed to resorting to indirect and tedious fine-tuning processes.

We showcase the competence of our method through practical applications: selecting and editing objects within both 360-degree and forward-facing scenes. Additionally, we conduct comprehensive benchmarking of our approach against current object removal techniques, as well as against leading 3DGS

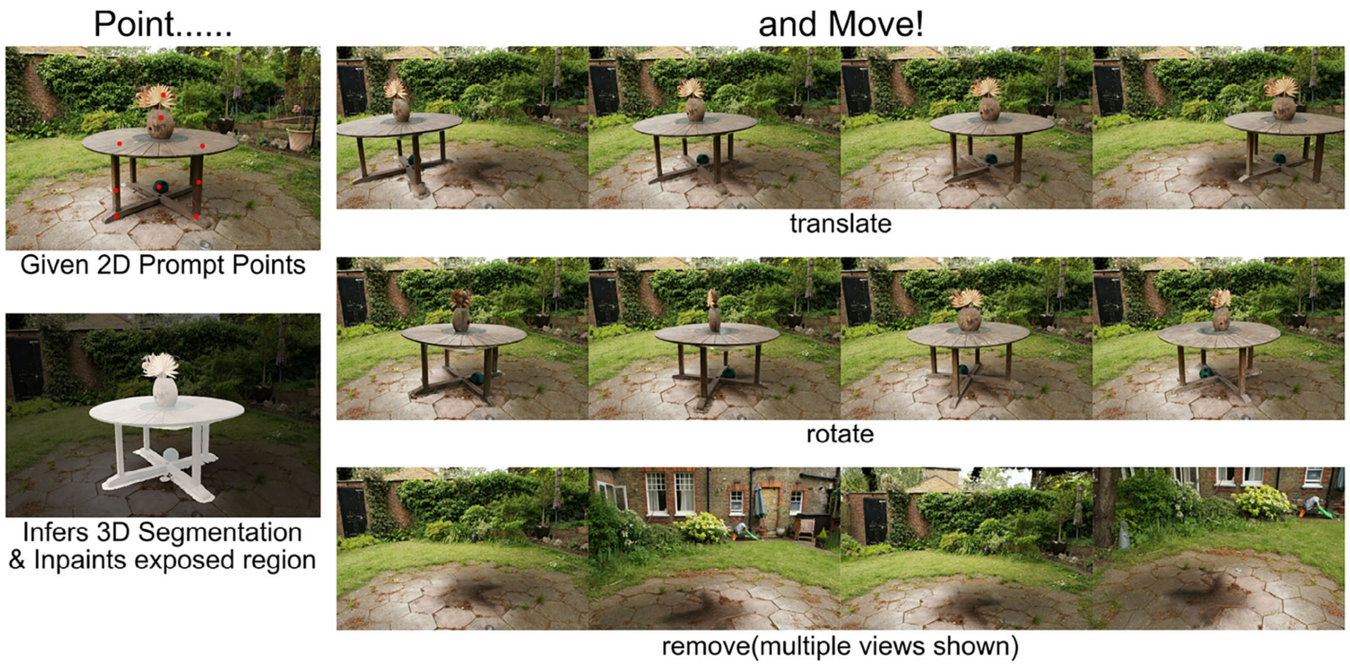
Point......
and Move!
Given 2D Prompt Points
translate
rotate
Infers 3D Segmentation & Inpaints exposed region
remove(multiple views shown)

FIGURE 1 Capability highlight of our method. The user selects an object in the scene via 2D point prompts, which our method uses to infer a 3D segmentation mask. After segmentation and exposed region inpainting, the user can freely manipulate the selected object in the scene in real-time, with all exposed regions or holes inpainted. Please refer to the supplementary material for a video demo on real-time editing.

segmentation and inpainting methods. Our results not only rival those in terms of quality but also exhibit enhanced capabilities such as full object manipulation–not merely removal. Moreover, our method holds a speed advantage when it comes to scene inpainting, further highlighting the effectiveness and efficiency of our approach. The code of our method will be published upon paper acceptance.

In summary, our contributions are as follows:

- We propose Point'n Move, the first end-to-end method that achieves interactive scene object manipulation with exposed region inpainting on Gaussian Splatting Radiance Fields (3DGS). More concretely, our method leverages the rapid and explicit nature of 3DGS to enable intuitive selection, high-quality inpainting and real-time editing.
- For intuitive selection, we propose a dual-stage self-prompting mask propagation process that produces high-quality 3D semantic segmentation masks from 2D image prompt points.
- For high-quality exposed region inpainting, we propose a rapid inpainting procedure that minimizes unnecessary inpainting and a reprojection-based initialization scheme. Both contribute to high-quality results (see Figure 1).

2 | RELATED WORK

2.1 | NeRF editing

There has been a lot of literature on scene editing based on radiance field representations, including geometry editing [8–10] and object-centric editing [11–13]. They adopt many sophisticated techniques such as fine-tuning [11, 12], rendering rays deformation [8–10], or editable representations [13].

However, most works prefer neural networks to represent radiance fields, and the implicit nature of this representation usually requires time-consuming fine-tuning for edit operations. As a result, these methods cannot support practical interactive editing use cases.

To tackle this challenge, another attempt is to represent radiance fields using explicit structures, including Control-NeRF [2] for volumetric grid, PAPR [14], NeuralEditor [3] and RIP-NeRF [15] for point clouds, NeuMesh [4] and Differentiable Blocks World [16] for meshes. The foundation of our method, Gaussian Splatting Radiance Field [7], also falls into this category.

The current body of work falls short in adequately supporting practical interactions, facing two challenges: intuitive 3D segmentation for object selection and inpainting of exposed regions or holes resulting from editing. Our method addresses these issues by leveraging the explicit representation of the Gaussian Splatting Radiance Field [7]. Building upon it, our method enables real-time, unrestricted editing.

2.2 | NeRF scene object removal

Scene object removal on NeRFs is another focus, aiming to select and remove objects from a trained neural radiance field. A pioneering work in object removal, SPIn-NeRF [6] achieves 3D segmentation by combining interactive segmentation methods, video segmentation methods, and NeRF-based semantic mask generation method that creates object removal masks. OR-NeRF [5] further simplifies this process by reprojecting the masks from the segmented 2D views onto new 2D views and then applying the Segment Anything Model [17] to perform segmentation on new views. Despite achieving good results, SPIn-NeRF and OR-NeRF both need to retrain NeRF after removal, which is not suitable for real-time unrestricted editing.

Inspired by SA3D [18], our approach incorporates a cross-view segmentation training process. Leveraging the explicit characteristics of the Gaussian Splatting Radiance Field [7], we realize explicit 3D segmentation through weighted point clouds, providing advantages for subsequent refinement and editing.

Both SPIn-NeRF and OR-NeRF introduced inpainting via 2D images for inpainting exposed areas. The scene NeRF associated with these images is updated by fine-tuning. A patch-wise perceptual loss in the masked regions is employed for multiview consistent effect from multiple inconsistent 2D inpaintings. Apart from these two approaches, [19] proposed an incremental scene inpainting and view-dependent effect estimation scheme for a multiview-consistent effect. [20] also proposed a novel confidence-score-based scheme to filter out view-inconsistent inpainted images, and a retraining process to update the confidence scores. Despite achieving impressive results, the complexity of these methods leads to a long training time.

Our method follows the overall paradigm employed by SPIn-NeRF and OR-NeRF, as we still perform 3D inpainting by training from inpainted 2D images. However, unlike existing works, we employ a simpler perceptual loss that compares the entire area inside the bounding box that bounds the mask rather than small patches inside, reducing the number of views to back-propagate through. We also make use of the explicit nature of Gaussian Splatting Radiance Fields, proposing a scene content-revealing pruning scheme and reprojection-based initialization process to ensure good quality.

3 | METHOD

Given a trained Gaussian Splatting Radiance Field [7] R, along with a set of cameras C in the scene and some 2D point annotations P that select an object in an image rendered from R at camera pose $C_0 \in C$, our method aims to achieve manipulation (translate/rotate/remove) of the object in real-time, with the newly exposed regions or holes properly inpainted.

Our proposed approach achieves this in three steps: segmentation, inpainting and recomposition (see Figure 2). Since our method heavily leverages its explicit formulation, we start by briefing Gaussian Splatting Radiance Fields.

3.1 | Background: Gaussian splatting radiance field

Gaussian Splatting Radiance Field [7], also referred to as 3D Gaussian Splatting (3DGS), is an explicit radiance field-based scene representation that represents a radiance field using a large number of 3D anisotropic balls, each modelled using a 3D gaussian distribution (Equation 1). More concretely, each anisotropic ball has mean $\mathcal{M} \in \mathbb{R}^3$, covariance Σ, opacity $\alpha \in \mathbb{R}$ and spherical harmonics parameters $C \in \mathbb{R}^k$ (k is the degrees of freedom) for modelling view-dependent color. For regularizing optimization, the covariance matrix is further decomposed into rotation matrix $\mathbf{R}$ and scaling matrix $\mathbf{S}$ by Equation 2. These matrices are further represented as quaternions $r \in \mathbb{R}^4$ and scaling factor $s \in \mathbb{R}^3$.

$$G(X) = e^{-\frac{1}{2}\mathcal{M}^T\Sigma^{-1}\mathcal{M}}, \tag{1}$$

$$\Sigma = \mathbf{RSS}^T\mathbf{R}^T \tag{2}$$

For this scene representation, view rendering is performed via point splatting [21]. Specifically, all gaussian balls in the scene are first projected onto the 2D image plane, and their color is computed from spherical harmonic parameters. Then, for every 16x16 pixel patch of the final image, the projected Gaussians that intersect with the patch are sorted by depth. For every pixel in the patch, its color is computed by alpha compositing the opacity and color of all the Gaussians covering this pixel, ordered by depth, as in Equation 3.

$$C = \sum_{i \in N_{cov}} c_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j), \tag{3}$$

where N_{cov} represents the splats that cover this pixel, α_i represents the opacity of this Gaussian splat multiplied by the density of the projected 2D Gaussian distribution at the location of the pixel, and c_i represents the computed color.

Gaussian Splatting Radiance Field achieves a significant advantage in training and inference speed due to its explicit approach and fast splatting rendering process. In addition to leveraging its superior speed, its point-cloud-like representation formulation enables us to perform processing from a point-cloud perspective, improving speed and quality.

3.2 | Object segmentation

While our method maintains the division of a scene into two parts, namely the object R_{obj} and the background R_{bg}, we devise a dual-stage segmentation process for implementation, as shown in Figure A1 in Appendix A.

Coarse stage segmentation. To achieve high-quality 3D segmentation, our method builds upon the cross-view self-prompting process proposed by SA3D [18]. We augment the scene representation for segmentation by adding and training a segmentation score attribute to all the Gaussian balls. Such scores could be used to render 2D segmentation maps by treating the score as color and rendering via the splatting process. Following SA3D, each training step starts by rendering images and segmentation masks from a chosen camera pose using the augmented representation, passing them through a heuristic algorithm to extract prompt points, and feed the prompts as well as the rendered image to Segment Anything Model (SAM) [17] to produce a more accurate pseudo ground truth mask. The pseudo-ground truth is then compared to the rendered mask for loss. The loss is presented in Equation 4, where M_{sam} is

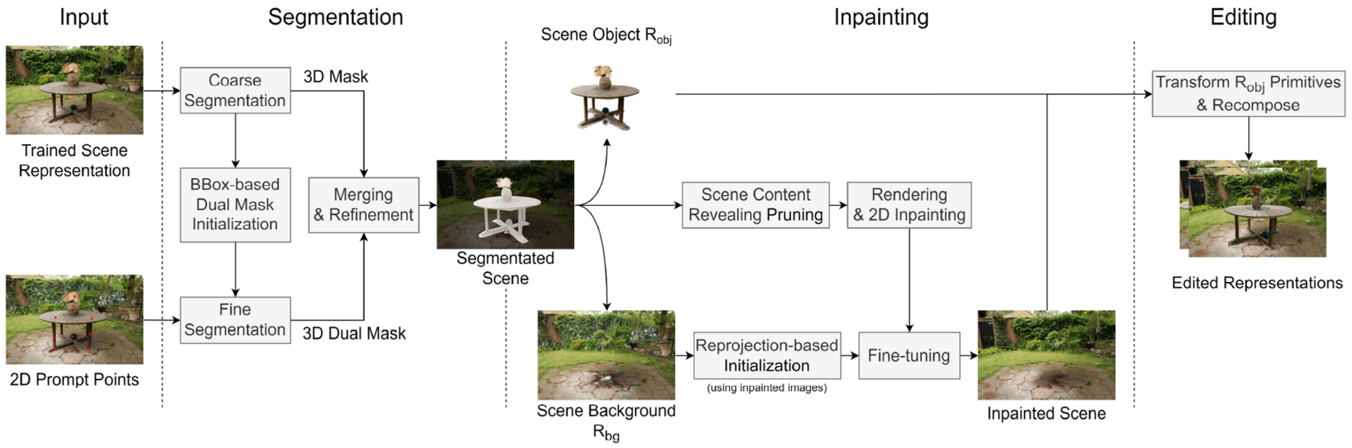

FIGURE 2 Overview of our method. Our method comprises three stages: segmentation, inpainting and editing. We start with a dual-stage self-prompting cross-view mask propagation process to produce a 3D segmentation of the scene, splitting it into R_{obj} and R_{bg}. The segmentation is then used to derive inpainted 2D scene images with a scene-content revealing pruning strategy. After using inpainted 2D images to fine-tune R_{bg}, R_{obj} can be edited and composited back into R_{bg} in real-time. Please refer to Figures A1 and A2 in Appendix A for more details about the segmentation and inpainting stage.

the pseudo ground truth mask, and M_r is the rendered mask. Herein, L_m is Equation 5 in [18].

$$L_{coarse} = L_m(M_{sam}, M_r). \tag{4}$$

Fine stage segmentation. To improve segmentation quality, we expand the process by adding a fine stage and a merging process. After iterating over all the provided camera poses C, we enter the fine stage of segmentation. In the same way as adding segmentation scores, we further augment the representation with dual segmentation scores, aiming to capture scene contents that should not be selected. We then initialize the dual scores by computing a 3D bounding box containing all Gaussians with high scores in the first stage. Every Gaussian outside the bounding box has its dual score set close to one, while Gaussians inside the bounding box have their dual score set to zero. We then continue the process for another full iteration, training the segmentation score and the newly added dual score, with the dual score attribute following the same self-prompting process and a new loss. The loss terms for the fine stage are presented in Equation 5. L_{mf} (Equation 6) is the loss for the dual mask, where M_{sd} is the pseudo ground truth dual mask, and M_{rd} is the rendered dual mask. We also added an additional term, L_{dual} (Equation 7), that encourages masks and dual masks to have no intersections.

$$L_{fine} = L_{coarse} + L_{mf} + L_{dual}, \tag{5}$$

$$L_{mf} = L_m(M_{sd}, M_{rd}), \tag{6}$$

$$L_{dual} = L_m(-M_r, M_{rd}) + \lambda_{dd} L_m(-M_{rd}, M_r). \tag{7}$$

Merging and refinement. Finally, we merge the scores and dual scores to create the final segmentation. We compute the 3D bounding box in the same way as in the initialization process. Then, in addition to the Gaussians with a high enough segmentation score, we also include every Gaussian in the bounding box whose dual score is below a threshold into the final segmentation, effectively accepting those "rejected" by dual mask. This rejection inclusion scheme aims to cover all Gaussians that are difficult to train, such as those hiding under the surface of the object. We also expand the selection by including Gaussians whose mean is close enough to the already selected Gaussians.

Remark. Thanks to the explicit representation of the Gaussian Splatting Radiance Field, we can efficiently implement the fine-stage initialization scheme, merging process, and expansion refinement. The resultant 3D segmentation mask, after merging and expansion, serves as our final output, effectively delineating the Gaussians and thereby segmenting the scene into R_{obj} and R_{bg}.

3.3 | Exposed surface inpainting

After splitting, R_{bg} will likely contain newly exposed surface regions or holes. For better rendering results after reintegrating the manipulated R_{obj}, these defects should be inpainted. We achieve this by inpainting rendered images and depth maps and then fine-tuning the scene representation with them.

2D inpainting. To minimize the amount of inpainting needed, we remove the masked Gaussians far away from the not masked Gaussians in R. This removes all selected Gaussians except those in close contact with the rest of the scene, which are very likely where the exposed regions are. These regions can be identified by rendering a segmentation mask using the pruned representation. We also add the exposed holes to the rendered mask, which is computed by including new pixels with low total opacity or color values close to the background color after performing the removal above. Finally, we refine the acquired inpainting masks before inpainting. (For details, please refer to Appendix B)

This scene content revealing pruning strategy allows us to minimize the amount of inpainting and preserve the existing scene contents as much as possible, as presented in the Pruned Scene image of Figure A2, in Appendix A.

We render RGB images and depth maps using this pruned representation and generate inpainting masks using the above-mentioned method. The RGB images and depth maps are then inpainted by a 2D inpainter.

3D rapid fine-tuning. With the inpainted images I_i and depth maps D_i, we then perform fine-tuning to inpaint R_{bg}. We initialize by reprojecting the masked region of an inpainted depth map and its associated image back into the representation as new Gaussians. As presented in the original 3DGS paper [7], good initialization is crucial for high-quality training results.

We then train using the losses below:

Outside mask color loss. We supervise the color of regions outside the mask via a weighted sum of pixel L1 and SSIM [22]:

$$L_{ocolor} = (1-\lambda_{ssim})\mathrm{L1}(I_i(1-M_i), I_r(1-M_i)) + \lambda_{ssim}\mathrm{SSIM}(I_i(1-M_i), I_r(1-M_i)), \quad (8)$$

where I_i and M_i are the inpainted image and mask, I_r is the rendered image of the representation, L1 stands for mean pixel L1 loss, and λ_{ssim} is the weight for SSIM.

Depth Loss where for scene geometry, we employ depth map L1:

$$L_{idepth} = \lambda_{depth}\mathrm{L1}(D_i, D_r), \quad (9)$$

where D_i is the inpainted depth map, D_r is the rendered depth map, L1 stands for mean pixel L1 loss, and λ_{depth} is the weighting factor.

Inside Mask Color Loss where for color inside the masked region, we adopt a perceptual color loss:

$$L_{icolor} = \lambda_{lpips}\mathrm{LPIPS}(\mathrm{Box}(I_i, M_i), \mathrm{Box}(I_r, M_i)), \quad (10)$$

where I_i and M_i are the inpainted image and mask, I_r is the rendered color image, Box stands for a function that computes the bounding box of the mask and only keeps parts of the image in the bounding box. LPIPS is the LPIPS [23] perceptual metric.

Here, we employ a perceptual color loss instead of directly comparing pixel values. This is because the 2D inpaintings are not guaranteed to be multiview consistent and a strict loss could harm quality. We also filter via the bounding box of the masked region instead of strictly in mask, this is to encourage better integration between the masked region and its surroundings.

3.4 | Editing and recomposition

With R_{bg} inpainted, R_{obj} can then be freely manipulated and composited back into R_{bg} in real-time for editing.

The selected object R_{obj} could be translated and rotated by transforming the position (or mean) $\mathcal{M}$ and rotation r of the underlying Gaussians. Recomposition is done by adding the Gaussians in R_{obj} back into R_{bg}. Note that as these transformations could be done by multiplying transform matrices, no further training is needed. All possible surfaces and holes that could be exposed by editing have already been inpainted in the previous step.

3.5 | Implementation details

Both stages of the segmentation training are done using the SGD optimizer, with a learning rate of 1.0. λ_{dd} is 0.1. For image inpainting, we employ state-of-the-art 2D image inpainting model Lama [24] for inpainting the color image and depth map. The scene inpainting training process also uses the SGD optimizer with learning rates equal to what is specified in the original 3D Gaussian Splatting paper [7]. λ_{ssim} is 0.2, λ_{depth} is 1.0, λ_{lpips} is 1.0. All experiments are conducted using a single A5000.

4 | EXPERIMENTS

We demonstrate the effectiveness of our method by testing it on both 360 and forward-facing scene datasets. Our method is also compared to existing scene object removal methods, reporting competitive performance in segmentation and inpainting despite being able to do more than just removal and having a speed advantage. We also compare our method against state-of-the-art 3DGS segmentation and inpainting methods. Finally, we conclude with ablation studies to highlight our key contributions.

4.1 | Experiment setups

Datasets. For diversity in evaluation, we test our method on the MipNeRF360 [25] dataset for 360 scenes and the SPIn-NeRF [6] dataset for forward-facing scenes. Both datasets consist of captured images of a scene with their associated camera parameters, which can be used to train the model to edit. As a dataset for evaluating object removal methods, the SPIn-NeRF dataset also contains scene images without the objects to remove and object segmentation masks for all captured images. For input prompt points, we use the point annotations from OR-NeRF [5].

Metrics. For segmentation, we report the mean accuracy and IoU (Intersection over Union) score of the rendered 2D segmentation masks across all images. For the image quality of inpainted scenes, we report the mean PSNR and LPIPS [23] score between the rendered image of the inpainted scene representation and the ground truth scene image where the object is removed. We also calculate the Frechet inception distance (FID) [26] between all rendered images and ground truth images taken without the target object for image quality.

Baselines. To evaluate the segmentation and inpainting ability of our method, we compare our approach against the state-of-the-art methods in 3D segmentation and scene object removal: SPIn-NeRF [6] and OR-NeRF [5]. More concretely, we compare against the TensoRF variant of OR-NeRF, which is the fastest and the best overall result quality variant presented in the paper. We also compare against recent state-of-the-art 3DGS segmentation and inpainting methods, namely SAGA [27] for segmentation and InFusion [28] for inpainting.

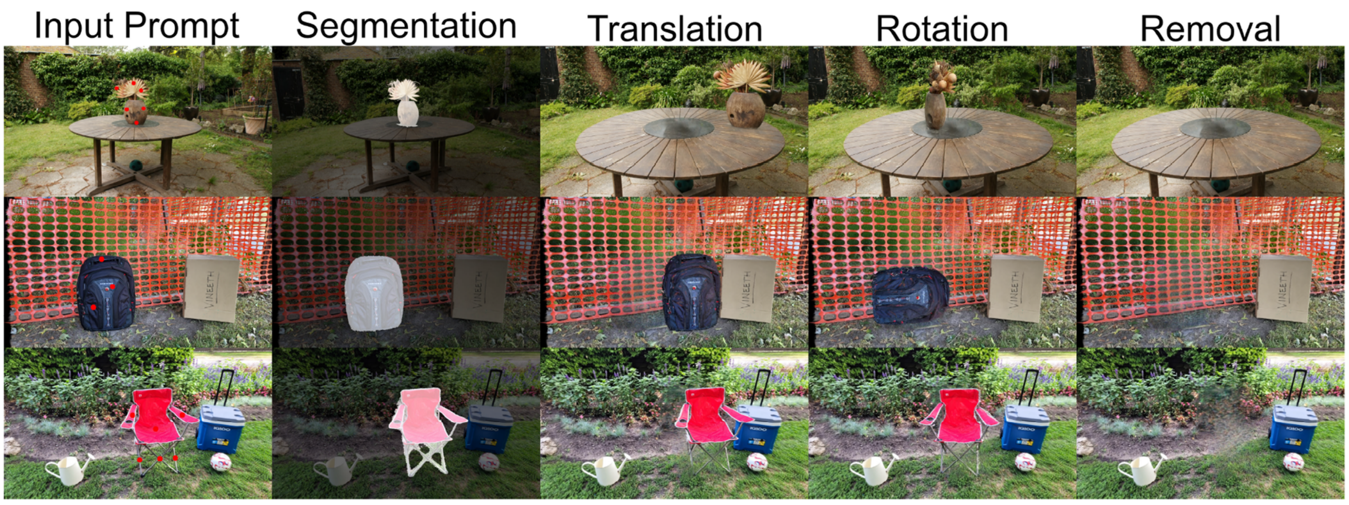

FIGURE 3 Input prompts, rendered segmentation mask and editing results of our method on 360 and forward-facing scenes. Note that our method gives sensible results even on scenes with extreme prompts (see last row). Please refer to the supplementary material for a video demo on real-time editing.

TABLE 1 Quantitative results on segmentation. The best value of each column is bolded for ease of reading. The table shows our method is superior to SAGA, on par with SPIn-NeRF and slightly inferior compared to OR-NeRF. However, our method produces 3D segmentation, which is more useful than the 2D segmentation masks produced by OR-NeRF.

	Segmentation metrics	
Method	**Acc (↑)**	**IoU (↑)**
Ours	99.51	92.71
OR-NeRF	**99.71**	**95.42**
SPIn-NeRF	98.91	91.66
SAGA	90.46	57.32

TABLE 2 Quantitative results on inpainting. The best value of each column is bolded for ease of reading. As presented in the table, our method achieves superior quality compared to all existing methods and has a significant speed advantage compared to NeRF-based methods.

	Inpainting metrics			
Method	**PSNR (↑)**	**FID (↓)**	**LPIPS (↓)**	**Inpainting time (↓)**
Ours	**19.83**	**40.33**	**0.2684**	20.41 min
InFusion	19.08	80.61	0.3424	**<1 min**
OR-NeRF	13.94	49.91	0.6162	168.97 min
SPIn-NeRF	14.83	67.26	0.6506	58.25 min

4.2 | Object manipulation on 360 and forward-facing scenes

To demonstrate the effectiveness of our approach, we perform manipulation with our method on scenes from the forward-facing SPIn-NeRF dataset and the 360 MipNeRF360 dataset. The qualitative results are presented in Figure 3. It can be noted that our method can perform high-quality selection and editing on various scenes, with exposed regions properly inpainted. Our method produces sensible results even for cases with extreme input, such as the scene in the last row of Figure 3.

4.3 | Evaluating segmentation quality

We compare our method against existing object removal methods: OR-NeRF [5], SPIn-NeRF [6] and the recent state-of-the-art 3DGS segmentation method: Segment Any 3D Gaussians (SAGA) [27]. Comparisons are done on the SPIn-NeRF [6] dataset.

The qualitative results are presented in Figure 4, and the quantitative results are presented in Table 1. The table shows that our method only achieves competitive performance against OR-NeRF. We attribute this to the fact that OR-NeRF directly operates on 2D images for segmentation and creates 2D masks instead of 3D segmentation. On the other hand, our method achieves true 3D segmentation and is rendered to 2D masks for comparison. The performance of our method is on par with SPIn-NeRF [6], a 3D segmentation method, and is superior to SAGA [27], the 3DGS segmentation method.

Our method also has the advantage of leaving almost no Gaussians behind. We compare our method against SAGA [27] on the MipNeRF360 [25] garden scene and segment the vase. Results are shown in Figure 5. Despite producing reasonable masks when rendered, SAGA has failed to include large portions of the vase, which is undesirable for object manipulation. In comparison, this is not the case for our method.

4.4 | Evaluating inpainting quality

We also evaluate the inpainting ability of our method on the SPIn-NeRF dataset. In addition to comparing with object removal methods such as SPIn-NeRF [6] and OR-NeRF [5], we also compare against Infusion [28], a state-of-the-art 3DGS inpainting method.

Qualitative results are presented in Figure 6, and the quantitative results are in Table 2. As can be seen, our method

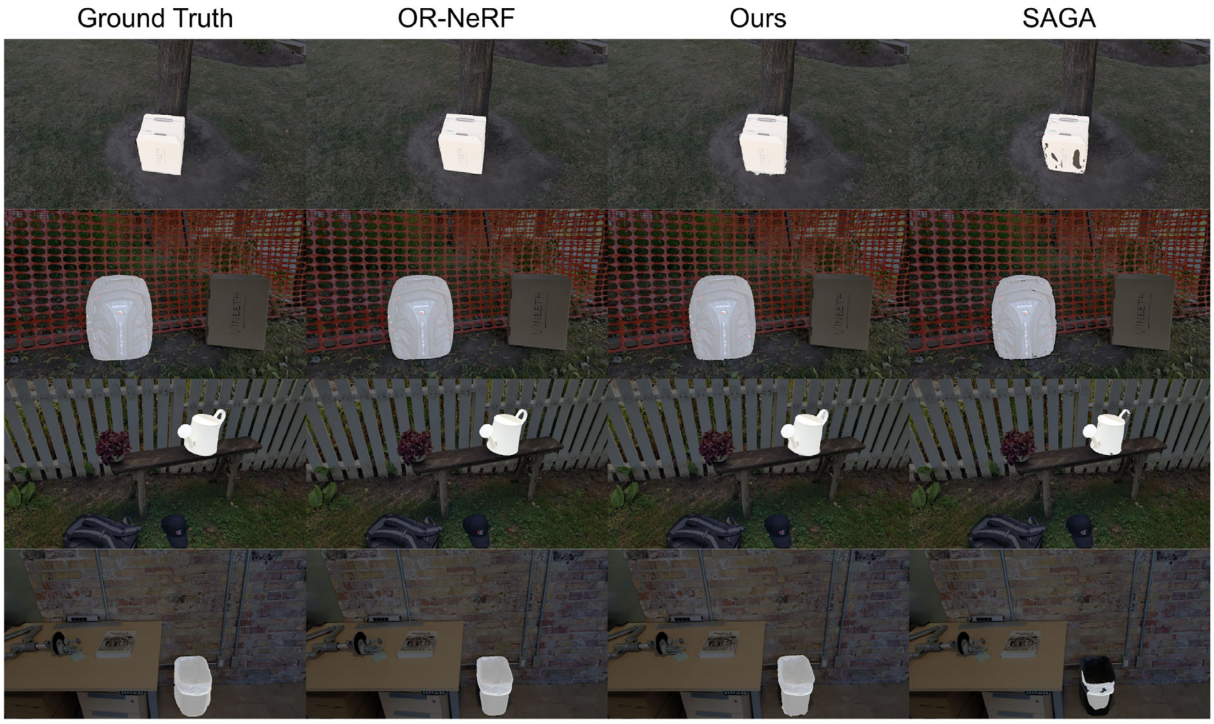

FIGURE 4 Qualitative results on segmentation. Our method achieves superior quality compared to SAGA. This is especially obvious from the box scene in the first row and the trash bin scene in the last row. Our method is also comparable to OR-NeRF (which is superior to SPIn-NeRF) despite producing 3D segmentation masks instead of 2D segmentation masks for every view.

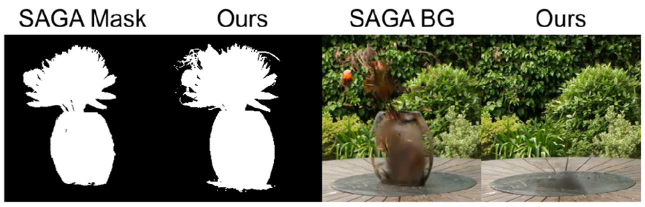

FIGURE 5 Comparing the not-selected parts of segmentation. Please note that SAGA, despite producing reasonable 2D masks when rendered, left out a large portion of Gaussians in its 3D segmentation, which is undesirable for manipulation.

produces more plausible results than existing methods. This is most significant in the second row, where our method correctly and sharply reproduces the structure of the plastic net, while InFusion fails, and OR-NeRF and SPIn-NeRF produce blurrier results. We attribute this to our adopted representation's superior representation capability and our proposed initialization and fine-tuning scheme. Good initialization and representation give clearer results. Extensive fine-tuning allows our method to achieve quality superior to InFusion at the cost of speed, as InFusion relies on reprojection using predicted depth.

Our method also has a significant speed advantage when compared against NeRF-based methods. As shown in the last column of Table 2, our method is about three times faster than the fastest NeRF-based method despite achieving superior result quality. Our method is slower than InFusion, but this is compensated by better quality and only needs to be done once before all editing operations.

4.5 | Ablation studies

Dual stage segmentation. To validate the importance of dual-stage segmentation, we disable the fine stage in our segmentation process and compare by evaluating images rendered from R_{bg}. As shown in Figure 7, disabling the fine stage could leave floaters in the scene that would be fixed by the fine stage and merging process.

Content-revealing scene pruning. We validate the importance of scene content-revealing pruning by running our algorithm with and without this process and comparing the final result against a pseudo-ground truth. The pseudo-ground truth is created by just removing the segmented object from the scene without any additional inpainting or fine-tuning.

We test this by removing the table in the MipNeRF360 [25] garden scene. The results are presented in Figure 8. As can be seen from the region highlighted by red rectangles, not performing pruning leads to blurry results, especially in places that do not require inpainting. We also present the quantitative results in Table 3, and shows that introducing pruning leads to better quality.

Inpainting optimization initialization. We analyze the effect of initialization by disabling it and directly training the pruned scene. As shown in Figure 9, without proper initialization, the fine-tuning process would leave a cloud of tiny floating Gaussians, instead of completely optimizing them away.

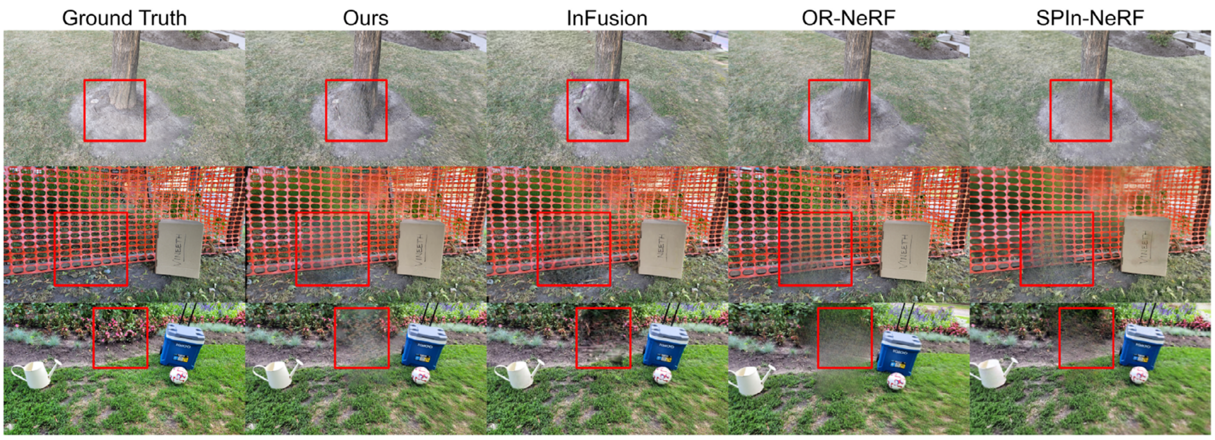

FIGURE 6 Qualitative results on inpainting. We present the ground truth and the output of all the methods, with the inpainted region marked with red rectangles. As can be seen from the third row, our method produces results closest to ground truth, while SAGA and SPIn-NeRF fill it with a dark hole, and OR-NeRF produces a blurry patch. In the second row, our method is the only one that correctly and sharply reproduced the structure of the plastic net.

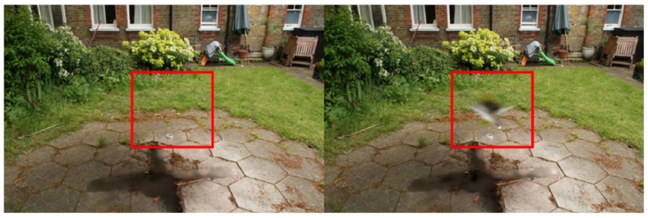

FIGURE 7 R_{bg} part of segmentation, performed with (left) or without (right) the fine stage. As could be seen, disabling the fine stage introduces floaters (highlighted using red rectangles) that should be included in R_{obj}.

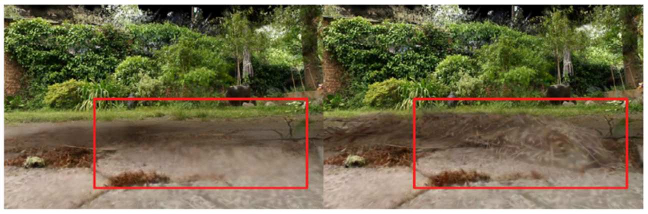

FIGURE 9 Close-up view of the output of scene fine-tuning, performed with (left) or without (right) reprojection-based initialization. Please note the small residual grains present on the right, highlighted using red rectangles.

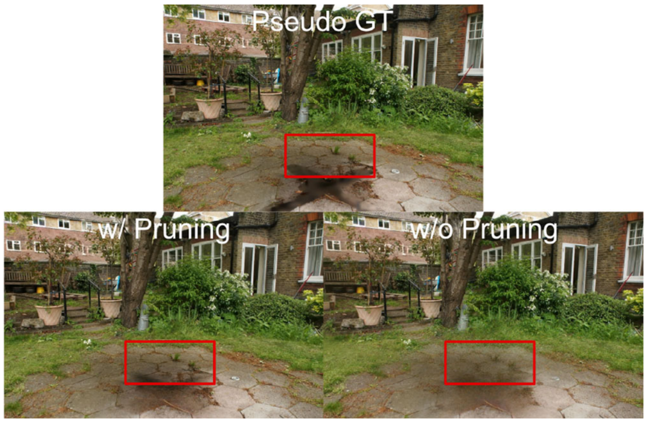

FIGURE 8 Results of our method, ran with (bottom left) or without (bottom right) content-revealing pruning, along with the pseudo ground truth (top). Note that pruning removes the blur caused by excessive inpainting in the red rectangle region.

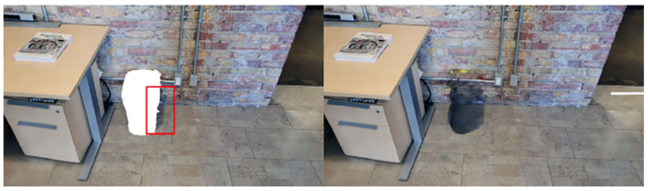

FIGURE 10 Artifacts caused by residual shadows misled the inpainter. The shadows of the object (shadows in the red box on the left) could mislead the 2D inpainter, producing dark results.

TABLE 3 Quantitative analysis on the effect of scene pruning. It can be seen that pruning makes the result much more similar to the pseudo-ground truth.

	Image quality metrics		
Method	**PSNR(↑)**	**FID(↓)**	**LPIPS(↓)**
w/ pruning	**27.90**	**75.58**	**0.0911**
w/o pruning	25.70	97.00	0.1561

5 | CONCLUSION

This paper introduces Point'n Move, a methodology enabling interactive manipulation of scene objects coupled with exposed region inpainting. Leveraging the explicit, point cloud-like formulation and speed advantages offered by the Gaussian Splatting Radiance Field [7] as the fundamental framework of our design, our approach achieves intuitive object selection, high-fidelity exposed region inpainting, and real-time editing. These facets collectively deliver a user-friendly interactive editing experience characterized by high-quality results.

Limitations. Currently, our method does not handle lighting or texture and focuses solely on geometry editing. Also, for some scenes, the inpaintings produced by our method are darker than expected, as presented in Figure 10. We attribute this to the precision of our segmentation as it does not include the shadow of the object, which misleads the 2D inpainter to inpaint with the shadow, making the output darker.

AUTHOR CONTRIBUTIONS

Jiajun Huang: Conceptualization; formal analysis; investigation; methodology; software; visualization; writing—original draft. **Hongchuan Yu**: Funding acquisition; project administration; resources; supervision; validation; writing—review and editing. **Jianjun Zhang**: Funding acquisition; project administration; resources; supervision. **Hammadi Nait-Charif**: Funding acquisition; project administration; resources; supervision.

CONFLICT OF INTEREST STATEMENT

The authors declare no conflicts of interest.

DATA AVAILABILITY STATEMENT

Data sharing is not applicable to this article as no new data were created in this study.

ORCID

Jiajun Huang https://orcid.org/0009-0001-3512-6119

REFERENCES

1. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: NeRF: Representing scenes as neural radiance fields for view synthesis. In: Computer Vision – ECCV 2020: 16th European Conference, Proceedings, Part I, pp. 405–421. Springer, Berlin (2020). https://doi.org/10.1007/978-3-030-58452-824
2. Lazova, V., Guzov, V., Olszewski, K., Tulyakov, S., Pons-Moll, G.: Controlnerf: Editable feature volumes for scene rendering and manipulation. In: 2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), pp. 4329–4339. IEEE, Piscataway (2022)
3. Chen, J.-K., Lyu, J., Wang, Y.-X.: NeuralEditor: Editing neural radiance fields via manipulating point clouds. In: Computer Vision and Pattern Recognition Conference (CVPR). IEEE, Piscataway (2023)
4. Bao, C., Yang, B., Junyi, Z., Hujun, B., Yinda, Z., Zhaopeng, C., Guofeng, Z.: Neumesh: Learning disentangled neural mesh-based implicit field for geometry and texture editing. In: European Conference on Computer Vision (ECCV). Springer, Berlin (2022)
5. Yin, Y., Fu, Z., Yang, F., Lin, G.: Or-nerf: Object removing from 3d scenes guided by multiview segmentation with neural radiance fields. arXiv:2305.10503 (2023)
6. Mirzaei, A., Aumentado-Armstrong, T., Derpanis, K.G., Kelly, J., Brubaker, M.A., Gilitschenski, I., Levinshtein, A.: SPIn-NeRF: Multiview segmentation and perceptual inpainting with neural radiance fields. In: Computer Vision and Pattern Recognition Conference (CVPR). IEEE, Piscataway (2023)
7. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d Gaussian splatting for real-time radiance field rendering. ACM Trans. Graph. 42(4), (2023)
8. Peng, Y., Yan, Y., Liu, S., Cheng, Y., Guan, S., Pan, B., Zhai, G., Yang, X.: CageNeRF: Cage-based neural radiance field for generalized 3D deformation and animation. In: Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A. (eds.) Advances in Neural Information Processing Systems, vol. 35, pp. 31402–31415. Curran Associates, Red Hook, NY (2022)
9. Xu, T., Harada, T.: Deforming radiance fields with cages. In: Computer Vision – ECCV 2022: 17th European Conference, Proceedings, Part XXXIII, pp. 159–175. Springer, Berlin (2022). https://doi.org/10.1007/978-3-031-19827-4_10
10. Yuan, Y.-J., Sun, Y.-T., Lai, Y.-K., Ma, Y., Jia, R., Gao, L.: NeRF-Editing: Geometry editing of neural radiance fields. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 18332–18343. IEEE, Piscataway (2022). https://doi.org/10.1109/CVPR52688.2022.01781
11. Rematas, K., Martin-Brualla, R., Ferrari, V.: Sharf: Shape-conditioned radiance fields from a single view. In: Proceedings of the 38th International Conference on Machine Learning, pp. 8948–8958. International Machine Learning Society, Madison, WI (2021)
12. Liu, S., Zhang, X., Zhang, Z., Zhang, R., Zhu, J.-Y., Russell, B.: Editing conditional radiance fields. arXiv:2105.06466 (2021)
13. Yang, B., Zhang, Y., Xu, Y., Li, Y., Zhou, H., Bao, H., Zhang, G., Cui, Z.: Learning object-compositional neural radiance field for editable scene rendering. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 13759–13768. IEEE, Piscataway (2021). https://doi.org/10.1109/ICCV48922.2021.01352
14. Zhang, Y., Peng, S., Moazeni, A., Li, K.: Papr: Proximity attention point rendering. In: Advances in Neural Information Processing Systems. Curran Associates, Red Hook, NY (2023)
15. Wang, Y., Wang, J., Qu, Y., Qi, Y.: Rip-nerf: Learning rotation-invariant point-based neural radiance field for fine-grained editing and compositing. In: Proceedings of the 2023 ACM International Conference on Multimedia Retrieval. ACM, New York (2023)
16. Monnier, T., Austin, J., Kanazawa, A., Efros, A.A., Aubry, M.: Differentiable blocks world: Qualitative 3d decomposition by rendering primitives. arXiv abs/2307.05473 (2023)
17. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.-Y., Dollár, P., Girshick, R.: Segment anything. arXiv:2304.02643 (2023). http://arxiv.org/abs/2304.02643
18. Cen, J., Zhou, Z., Fang, J., Shen, W., Xie, L., Zhang, X., Tian, Q.: Segment anything in 3d with nerfs. arXiv:230412308 (2023)
19. Mirzaei, A., Aumentado-Armstrong, T., Brubaker, M.A., Kelly, J., Levinshtein, A., Derpanis, K.G., Gilitschenski, I.: Reference-guided controllable inpainting of neural radiance fields. In: International Conference on Computer Vision (ICCV). IEEE, Piscataway (2023)
20. Weder, S., Garcia-Hernando, G., Monszpart, A., Pollefeys, M., Brostow, G.J., Firman, M., Vicente, S.: Removing objects from neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 16528–16538. IEEE, Piscataway (2023)
21. Yifan, W., Serena, F., Wu, S., Öztireli, C., Sorkine-Hornung, O.: Differentiable surface splatting for point-based geometry processing. ACM Trans. Graph. 38(6), 1–14 (2019)
22. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. 13, 600–612 (2004)
23. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR. IEEE, Piscataway (2018)
24. Suvorov, R., Logacheva, E., Mashikhin, A., Remizova, A., Ashukha, A., Silvestrov, A., Kong, N., Goka, H., Park, K., Lempitsky, V.: Resolution-robust large mask inpainting with Fourier convolutions. In: 2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), pp. 3172–3182. IEEE, Piscataway (2022). https://doi.org/10.1109/WACV51458.2022.00323
25. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-nerf 360: Unbounded anti-aliased neural radiance fields. CVPR (2022)
26. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: GANs trained by a two time-scale update rule converge to a local nash equilibrium. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. NIPS'17, pp. 6629–6640 (2017)
27. Cen, J., Fang, J., Yang, C., Xie, L., Zhang, X., Shen, W., Tian, Q.: Segment any 3d Gaussians. arXiv preprint arXiv:231200860 (2023)
28. Liu, Z., Ouyang, H., Wang, Q., Cheng, K.L., Xiao, J., Zhu, K., Xue, N., Liu, Y., Shen, Y., Cao, Y.: Infusion: Inpainting 3d Gaussians via learning depth completion from diffusion prior. arXiv preprint arXiv:240411613 (2024)

How to cite this article: Huang, J., Yu, H., Zhang, J., Nait-Charif, H.: Point'n Move: Interactive scene object manipulation on Gaussian splatting radiance fields. IET Image Process. 18, 3507–3517 (2024). https://doi.org/10.1049/ipr2.13190

APPENDIX A: DETAILED METHOD DIAGRAMS

Here we present the detailed diagrams of our method. Please refer to Figure A1 for the segmentation stage and Figure A2 for the inpainting stage.

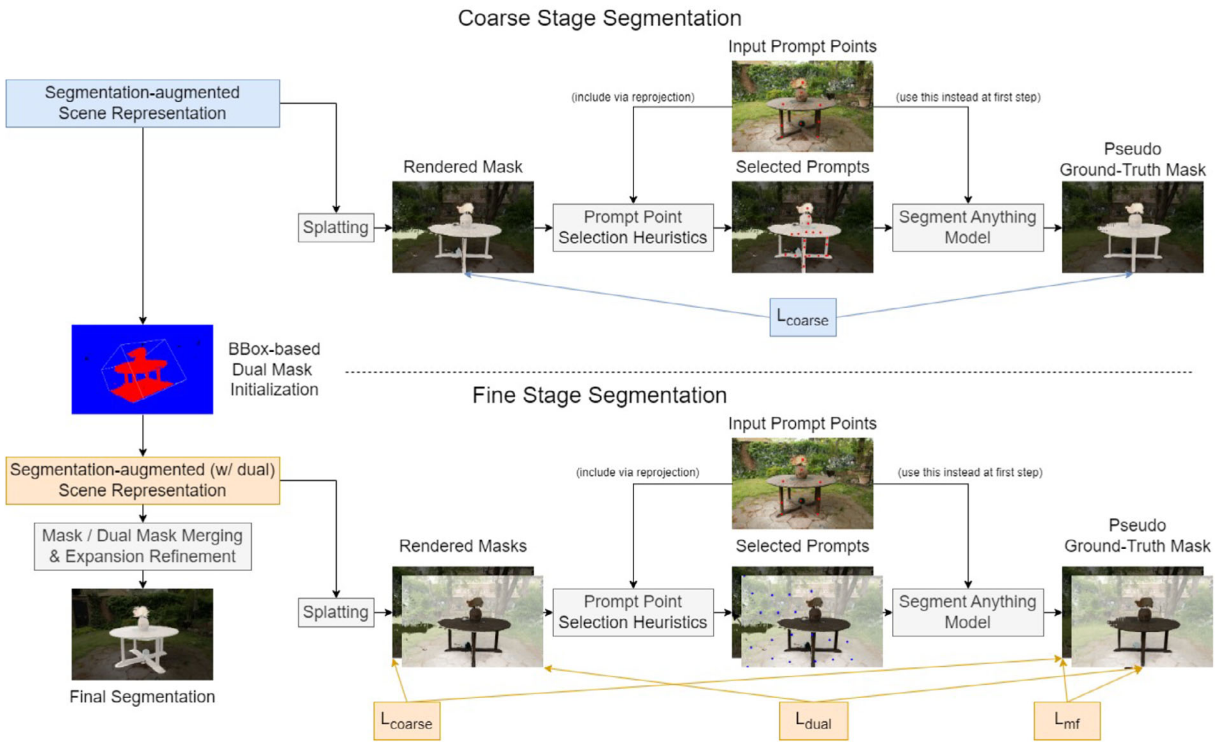
Coarse Stage Segmentation
Segmentation-augmented Scene Representation
Input Prompt Points
(include via reprojection)
(use this instead at first step)
Rendered Mask
Selected Prompts
Pseudo Ground-Truth Mask
Splatting
Prompt Point Selection Heuristics
Segment Anything Model
L_{coarse}
BBox-based Dual Mask Initialization
Fine Stage Segmentation
Segmentation-augmented (w/ dual) Scene Representation
Mask / Dual Mask Merging & Expansion Refinement
Final Segmentation
Rendered Masks
L_{dual}
L_{mf}

FIGURE A1 Detailed view of the segmentation stage. The cross-view transfer process involves selecting prompt points from rendered masks, which are used to produce pseudo-ground truths to supervise the rendering mask. The process is started using the input prompt points. After the coarse stage, a dual mask for content outside the selection and additional loss terms for training are added. Finally, the scores and dual scores are merged into the final segmentation.

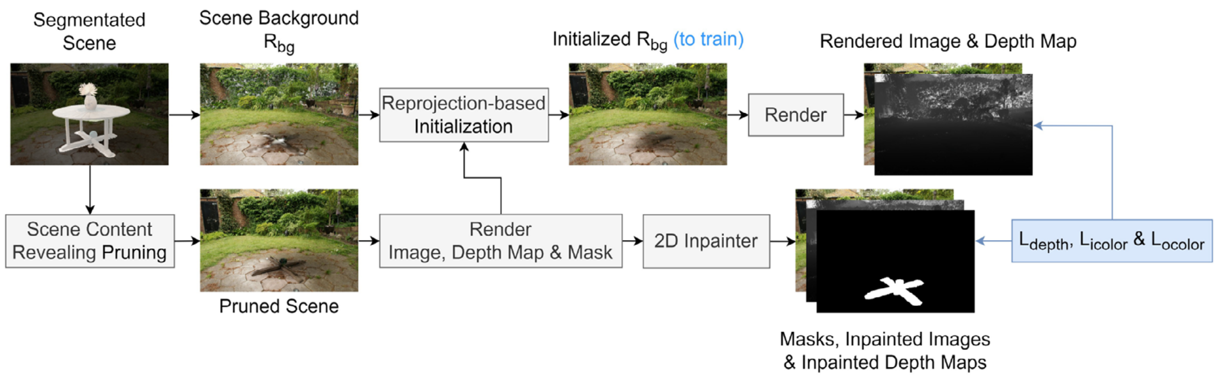
Segmentated Scene
Scene Background R_{bg}
Initialized R_{bg} (to train)
Rendered Image & Depth Map
Reprojection-based Initialization
Render
Scene Content Revealing Pruning
Pruned Scene
Render Image, Depth Map & Mask
2D Inpainter
L_{depth}, L_{icolor} & L_{ocolor}
Masks, Inpainted Images & Inpainted Depth Maps

FIGURE A2 Detailed view of the inpainting stage. We start by performing scene content revealing pruning to reveal already captured backgrounds, minimizing the areas to inpaint. We then render segmentation masks, depth maps and images of the pruned scene and inpaint the images and depth maps using 2D inpainting models and the rendered masks. With the inpainted results, we use one pair of the inpainted depth map and color images to initialize R_{bg}, and then fine-tune it using the inpainted depth map and images, supervised using the proposed losses.

APPENDIX B: 2D MASK REFINEMENT ALGORITHM

Our 2D mask refinement algorithm (see Algorithm B1) is inspired by the refinement procedure present in the implementation of OR-NeRF [5].

In essence, our method dilates the mask, finds the contour with the largest area and takes the region inside it as the refined mask. This is to eliminate potential holes in the mask. All functions invoked in Algorithm B1 can be implemented using functions from OpenCV.

ALGORITHM B1 2D Mask Refinement Algorithm.

```
Require: 2D Inpainting Mask $m$
Ensure: Refined Inpainting Mask $ret$
    $m \leftarrow \text{dilate}(m, kernel\_size = 3, iteration = 3)$
    $contours \leftarrow \text{findContours}(m)$
    $contour \leftarrow \text{contourWithMaxArea}(contours)$
    $ret \leftarrow \text{filledContour}(contour)$
    return $ret$
```

APPENDIX C: ADDITIONAL PER-SCENE METRICS

Quantitative results presented in Table 1 and Table 2 are the means across all scenes in the SPIn-NeRF [6] dataset. We present the quantitative result for each individual scene here. Please see Table C1 for segmentation metrics and Table C2 for inpainting metrics.

TABLE C1 Quantitative per-scene result on segmentation quality. Results for SPIn-NeRF and OR-NeRF are taken from their original papers, while results for SAGA are reproduced from their published code. SPIn-NeRF did not provide per-scene metrics; hence, they are not shown. The best value of each column is bolded for ease of reading. Our method is slightly inferior to OR-NeRF but has an advantage over SPIn-NeRF. However, we produce 3D segmentation, while OR-NeRF performs segmentation on 2D images, which is not usable for unrestricted manipulations.

Metric	Method	2	3	4	7	10	12	Book	Trash	Mean
Acc (↑)	Ours	99.80	**99.79**	99.74	99.59	99.84	98.72	99.01	**99.56**	99.51
	OR-NeRF	**99.82**	99.73	**99.79**	**99.81**	**99.87**	**99.30**	**99.51**	99.51	**99.71**
	SPIn-NeRF	–	–	–	–	–	–	–	–	98.91
	SAGA	97.47	99.43	50.28	86.50	99.68	95.63	97.51	97.19	90.46
IoU (↑)	Ours	96.16	**98.09**	98.15	94.68	94.67	86.51	83.45	**89.99**	92.71
	OR-NeRF	**96.47**	97.48	**98.50**	**97.43**	**95.47**	**91.73**	**88.68**	88.68	**95.42**
	SPIn-NeRF	–	–	–	–	–	–	–	–	91.66
	SAGA	65.79	94.78	20.85	29.11	89.44	66.03	58.27	34.25	57.32

TABLE C2 Quantitative per-scene result on scene inpainting quality. Results for SPIn-NeRF and OR-NeRF are taken from their original papers, while results for InFusion are reproduced from their published code. The best value of each column is bolded for ease of reading. Please note our method's overall best performance across most scenes.

Metric	Method	2	3	4	7	10	12	Book	Trash	Mean
PSNR (↑)	Ours	**18.48**	**18.04**	**20.88**	**21.40**	**19.75**	**16.62**	**22.28**	21.18	**19.83**
	OR-NeRF	15.94	11.42	13.02	14.37	12.89	11.40	15.88	16.63	13.94
	SPIn-NeRF	16.69	12.08	14.90	15.34	12.73	12.39	17.84	16.70	14.83
	InFusion	18.28	16.98	19.68	19.27	19.04	16.04	21.59	**21.79**	19.08
FID (↓)	Ours	**53.60**	36.39	**51.78**	**22.48**	**21.67**	**26.23**	81.68	**28.84**	**40.33**
	OR-NeRF	72.10	**34.72**	74.04	38.66	43.89	38.02	**64.91**	32.96	49.91
	SPIn-NeRF	71.75	68.35	61.10	43.95	91.73	50.52	102.71	47.98	67.26
	InFusion	223.79	62.00	67.84	60.77	28.32	32.20	114.02	55.92	80.61
LPIPS (↓)	Ours	**0.4544**	**0.2217**	**0.3229**	**0.2858**	**0.2264**	**0.3352**	**0.2147**	**0.2301**	**0.2864**
	OR-NeRF	0.7909	0.4937	0.6684	0.6445	0.6165	0.7179	0.5094	0.4882	0.6162
	SPIn-NeRF	0.8489	0.5472	0.6815	0.6552	0.7003	0.7518	0.4226	0.5972	0.6506
	InFusion	0.5377	0.2829	0.3851	0.3308	0.2765	0.3610	0.2729	0.2921	0.3424