ONeRF: Unsupervised 3D Object Segmentation from Multiple Views

Anonymous CVPR submission

Paper ID 1571

Abstract

We present ONeRF, a method that automatically segments and reconstructs object instances in 3D from multi-view RGB images without any additional manual annotations. The segmented 3D objects are represented using separate Neural Radiance Fields (NeRFs) which allow for various 3D scene editing and novel view rendering. At the core of our method is an unsupervised approach using the iterative Expectation-Maximization algorithm, which effectively aggregates 2D visual features and the corresponding 3D cues from multi-views for joint 3D object segmentation and reconstruction. Unlike existing approaches that can only handle simple objects, our method produces segmented full 3D NeRFs of individual objects with complex shapes, topologies and appearance. The segmented ONeRFs enable a range of 3D scene editing, such as object transformation, insertion and deletion.

1. Introduction

Neural Radiance Fields (NeRF) [28] have recently been becoming the mainstream approach for novel view synthesis, given its excellent performance for complex real-world scenes despite their simplicity. The key idea is to represent the entire scene as a radiance field parametrized using an MLP, taking in xyz coordinates and viewing directions as input and produces densities and view-dependent colors. Since the first contribution [28], NeRfs have been extended in various dimensions, for example, with improved efficiency [18, 48], improved quality [1, 52], deformation [29, 30], uncalibrated images [41, 47], sparse views [5, 13], better generalization [36, 49], material decomposition [3, 53], as well as many other applications [11, 24].

However, most of the existing work focuses on novel view rendering. Less effort has been made to leverage this representation for the fundamental computer vision problem of 3D object segmentation, which is the goal of this paper. Notably, by obtaining object-level NeRF representations of a 3D scene, various 3D scene editing tasks can be enabled, including object editing, insertion, removal and completion. Several recent works have attempted to learn category-specific object NeRFs from multi-view images and optional depth information, allowing for single-image inference [38, 50]. However, these methods mainly work with specific object categories, such as CLEVR [16] and ShapeNet [42] objects, and is not capable of segmenting arbitrary objects in general scenes. Others rely on additional information, such as 2D segmentation [44] and 2D bounding box annotations [14], and lift them to 3D. NeuralDiff [39] on the other hand leverages motion cues in egocentric videos to separate the foreground layer of moving objects and the static background layer, but cannot obtain full 3D segmentations of individual foreground objects. In contrast, our ONeRFs consist of an unsupervised approach that performs automatic 3D object segmentation of arbitrary scenes from multiple views, without relying on any external annotations, which can be regarded as a generalized version of some of the above category-specific methods.

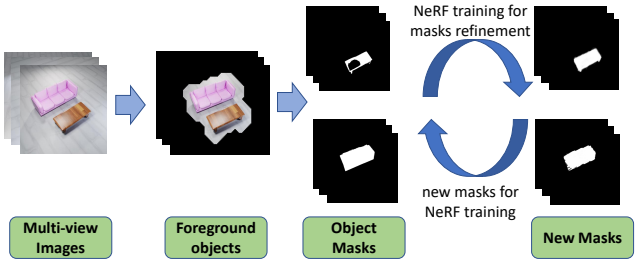

Figure 1. **3D slot attention.** Coarse 2D masks are iteratively refined with 3D object NeRFs in an unsupervised manner. The output are individual NeRFs for each object.

Given multi-view images of a scene, our goal is to detect individual objects and represent each one of them using a separate NeRF. In order to do this, we introduce the notion of *3D slot attention*, which is an unsupervised method for fusing 2D visual features and 3D geometrical information inherent from multiple views for segmenting the scene into individual 3D object NeRFs. Specifically, both the 2D features and 3D information extracted from multiple views are used to produce initial object and background masks, which will be iteratively refined via expectation-maximization to

segment individual NeRFs and thus 3D objects. Unlike previous works, our unsupervised ONeRFs segmentation can handle objects with complex textures, geometries and topologies, which support not only novel view synthesis but also scene editing such as object insertion, deletion, and modification with individual ONeRFs automatically learned from a given scene.

While the qualitative results have significant room for improvements as in other related state-of-the-arts, this paper makes the first attempt to contribute a significant baseline for unsupervised learning on segmenting a cluttered scene into individual ONeRFs from multiple views. Our qualitative and quantitative results have validated the soundness of our technical approach. Preliminary results on scene editing shows the great potential of ONeRFs.

2. Related Work

2.1. Multi-View 3D Reconstruction

Multi-view stereo (MVS) is a classical computer vision problem for 3D reconstruction from images captured from multiple viewpoints. Traditional methods include [2, 9, 21, 22]. More recent deep learning-based MVS methods are MVSNet [46], P-MVSNet [27], and CVP-MVSNet [45]. Once reconstructed, the 3D scene can be rendered at any viewpoints. However, the output of MVS consists of (quasi) dense point clouds and thus are not object aware. Combining deep MVS technique with differentiable volume rendering, MVSNeRF [4] reconstructs a neural radiance field from unstructured multi-view input images for novel view synthesis, which generalizes well across scenes using MVS. View synthesis from deep lightfield [17] has also been proposed. While novel views can be synthesized, the pertinent NeRF has little object awareness.

2.2. Object Segmentation

In our unsupervised approach, 2D object masks estimated from multiple views must be consistent with the underlying 3D voxel volume. 3D visual hulls [35] can be estimated from accurate 2D object masks given in the input calibrated images. Conversely, accurate segmentation of 3D point clouds/voxels can produce consistent 2D object masks across views. Segmentation of 3D point cloud/voxels is an active research area, including PointNet [31], DGCNN [40] and [33], and later on large-scale point cloud segmentation such as PointNet++ [32] and [12, 25, 43]. Recent works include weakly supervised segmentation [54, 56], semi-supervised segmentation [15], few shot segmentation [55], instance segmentation [51], and semantic segmetation [34]. While the above can be deployed for 3D NeRF segmentation on the pertinent voxel volume, ONeRFs is an end-to-end unsupervised approach for object NeRFs segmentation taking only images as input.

There is also another line of work on unsupervised object-centric learning, which is learning-based and decomposes a single image into different objects, mostly in 2D: e.g., GENISIS [8], SPACE [23], SPAIR [7], IODINE [10]), SlotAttention [26]. Most of these work with very simple CLEVR-like objects.

3. Method

ONeRFs is an end-to-end unsupervised approach for segmenting 3D object NeRFs from multiple views of the underlying scene. Borrowing attention typical of deep neural networks, our 3D attention slots operates by cooperating 2D visual and 3D geometrical cues, so that they can reinforce each other in alternating optimization where respective errors will be automatically corrected via resolving their inconsistencies. Figure 2 shows the overall framework, where starting coarse 2D masks are working in tandem with 3D reconstruction from NeRFs to iteratively achieve accurate object NeRFs segmentation.

3.1. Initial coarse masks generation

Clustering. Given the input images of the scene and camera poses from multiple viewpoints, we resize the images to 128×128 and extract VGG features [37] and reconstruct 3D positions of all the pixels, which are clustered into two classes using K-Means clustering: foreground and background. Let

$$s_i^f = ||\mathbf{f}_i - \boldsymbol{\mu}_f|| + w||P_i - \mathbf{p}_f||, \quad (1)$$

$$s_i^b = ||\mathbf{f}_i - \boldsymbol{\mu}_b||, \quad (2)$$

where s_i^f and s_i^b are respectively the foreground and background scores of pixel i, $\mathbf{f}_i$ is the VGG feature, P_i are the 3D coordinates of corresponding pixel i, which can be obtained from multi-view reconstruction from the input images. $\boldsymbol{\mu}_f$ and $\boldsymbol{\mu}_b$ are respectively the mean foreground and background clustered features, $\mathbf{p}_f$ are the 3D coordinates corresponding to the clustered foreground points, and w is the weight.

Assuming sufficiently far background, background pixels will have large variance in 3D locations. Thus foreground pixels tend to have smaller $||P_i - \mu_f||$. Then we update the three means by

$$\boldsymbol{\mu}_f = \frac{\sum_{i:s_i^f < s_i^b} \mathbf{f}_i}{\#}, \mathbf{p}_f = \frac{\sum_{i:s_i^f < s_i^b} P_i}{\#}, \quad (3)$$

$$\boldsymbol{\mu}_b = \frac{\sum_{i:s_i^f \geq s_i^b} \mathbf{f}_i}{\#}, \quad (4)$$

where #'s correspond to the relevant number of foreground and background pixels. Finally we get the foreground and background masks M_f^0, M_b^0, and upsample them into the original size.

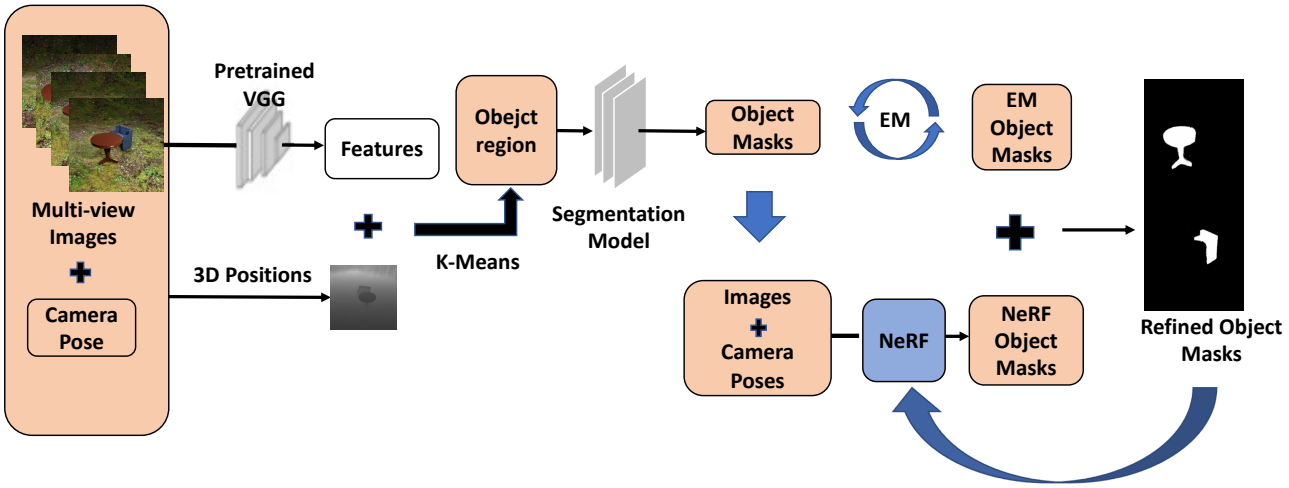

Figure 2. **Overall framework.** Given a set of images of a scene and their camera poses, we use K-Means to obtain coarse foreground object regions according to the VGG feature and 3D position of each pixel. Then we obtain the object masks by using an unsupervised object segmentation convolutional neural network, and then use these masks to train NeRF networks for individual objects. Combining the masks generated by NeRF and the masks improved by expectation maximization (EM), we get the refined objects masks, which will be used to train NeRF models in next iteration.

Image Segmentation and Instances. After labeling the pixels corresponding to the 3D foreground clustered above, we apply unsupervised segmentation [20] to produce $n+1$ parts $\mathcal{P}_0, \cdots, \mathcal{P}_n$. $\mathcal{P}_0$ is the background we separate in the first step, $\mathcal{P}_1$ is the background near the foreground objects which cannot be precisely clustered in the first step and the others correspond to the foreground objects, denoted as $\mathcal{P}_2, \cdots, \mathcal{P}_n$.

Suppose there are k objects in expectation. We then discard $n-(k+1)$ of $\mathcal{P}_1, \cdots \mathcal{P}_n$ occupying the smallest areas, i.e. those classes with fewest pixels, which are very likely due to incorrect clustering or noise.

We then put the full image into the model in image segmentation step to obtain the response vector map of the image [20], where the response vector at a given pixel represents the clustering scores for all the n classes (minus the weak ones rejected as above). The pixels in the same class tend to have similar response vectors. Here, we cluster the pixels by the highest entry in the response vector. The response vectors of pixels in $\mathcal{P}_0$ are similar to those in $\mathcal{P}_1$. We sample some pixels in $\mathcal{P}_i$ and find the mean of the response vectors of those pixels, denoted as $\mathbf{n}_i$. Then, $\mathcal{P}_i$ with $\mathbf{n}_i$ that is closest to $\mathbf{n}_0$ is the remaining background to be separated from the coarse foreground.

3.2. Train NeRFs and Refine Masks

Training. Given a 3D position $\mathbf{x}$ and view direction $\mathbf{d}$, the network will output the RGB color $\mathbf{c}$ and volume density σ of that point [28]. Following [44], for each segmented object a the loss is

$$\lambda_1 ||\hat{C}_a(\mathbf{r}) - C(\mathbf{r})|| \odot M_a + \lambda_2 ||\hat{A}_a(\mathbf{r}) - M_a|| \qquad (5)$$

where $\mathbf{r}$ is the ray, $\hat{C}_a(\mathbf{r})$ is the RGB triplet by the NeRF of object a, $C(\mathbf{r})$ is the ground truth RGB, M_a is the mask of object a, and $\hat{A}_a(\mathbf{r})$ is the accuracy map by NeRF of object a, and λ_1, λ_2 are weights. $C_a(\mathbf{r})$ and $A_a(\mathbf{r})$, which follows [28], are defined as

$$\hat{C}_a(\mathbf{r}) = \sum_{i=1}^{N} T_i^a \alpha_i^a \mathbf{c}_i^a \qquad (6)$$

$$\hat{A}_a(\mathbf{r}) = \sum_{i=1}^{N} T_i^a \alpha_i^a \qquad (7)$$

$$T_i^a = \sum_{j=1}^{i} \exp(-\sigma_j^a \delta_j) \qquad (8)$$

$$\alpha_i^a = 1 - \exp(-\sigma_j^a \delta_j) \qquad (9)$$

We sample N points along the ray $\mathbf{r}$ and the distance between the i-th point and the camera is δ_i, which will increase with the increase of the index. For point i in object a, we obtain the RGB color $\mathbf{c}_i^a$ and volume density σ_i^a at that point from the network. Then by Equation (6) and (7), we obtain the rendered color and accuracy masks along that ray.

The color within the in-mask region will converge to the ground-truth RGB color. For $\hat{A}_a$, in-mask region will converge to 1 while regions outside to 0. As NeRF needs to

guarantee 3D consistency, occlusion and incorrect masking in the last step can be handled automatically.

Refine Masks. The masks are iteratively refined by leveraging the object NeRFs for each individual object instance, in the presence of occlusion if any. We use the occlusion accuracy masks $\hat{A}'_a(\mathbf{r})$ as the occlusion masks from NeRF networks. Specifically, each new object mask is generated by

$$\hat{A}'_a(\mathbf{r}) = \sum_{i=1}^{N} T_i \alpha_i^a \tag{10}$$

$$T_i = \sum_{j=1}^{i} \exp(-\sigma_j \delta_j) \tag{11}$$

$$\sigma_j = \sum_{o} \sigma_j^o \tag{12}$$

where σ_j is the summation of volume densities from NeRF networks of all objects. We only consider the contribution of the other NeRFs on T_i, that is, decreasing values of α along the ray $\mathbf{r}$. Finally, expectation-maximization (EM) is employed to refine the input mask. We use the response vector map again. We initialize the μ as

$$\mu_i^{(0)} = \frac{\sum_{x_j \in M_i} \mathbf{v}(\mathbf{x}_j)}{\#} \tag{13}$$

where M_i is the mask, $\mathbf{v}_j$ is the response vector and #'s correspond to the number of pixels in M_i. Then EM is performed on the pixels of background. At the t-th iteration, for each response vector of background $\mathbf{v}_n$, we formulate the posterior probability of $\mathbf{v}_n$ given $\boldsymbol{\mu}_i$ as

$$p(\mathbf{v}_n|\boldsymbol{\mu}_i^{(t)}) = \exp(-||\mathbf{v}_n - \boldsymbol{\mu}||_2^2), \tag{14}$$

where $\boldsymbol{\mu}_i^{(t)}$ is the $\boldsymbol{\mu}$ value at the t-th iteration. The latent variable $\mathbf{z}_n^{(t)}$ for $\mathbf{v}_n$ is

$$z_{n,i}^{(t)} = \frac{p(\mathbf{v}_n|\boldsymbol{\mu}_i)}{\sum_{k=1}^{K} p(\mathbf{v}_n|\boldsymbol{\mu}_k)}, \tag{15}$$

where $z_{n,i}^{(t)}$ is the i-th entry of $\mathbf{z}_n^{(t)}$ and K is total number of classes, which is the number of objects plus one (the background). Then we update the $\boldsymbol{\mu}$ value of the next iteration as

$$\boldsymbol{\mu}_i^{(t+1)} = \frac{\sum_{n=1}^{N} z_{n,i} \mathbf{v}_n}{\sum_{n=1}^{N} z_{n,i}} \tag{16}$$

Then we output the softmax results upon the above EM convergence. Let M_{re} be the resulting mask, then the final mask is given by

$$w_1 M_{re} + w_2 \hat{A}'_a > 0.5 \tag{17}$$

for some weights w_1, w_2.

4. Experiments

4.1. Setup

4.1.1 Datasets

We first test our method on CLEVR dataset [16] and then move to more complex scenes. Our dataset is built upon ClevrTex [19], which wraps some realistic textures to the objects and background. Instead of regular geometric models, we use the some objects with more realistic shapes, such as animals and furniture items, and use Blender [6] to render 400×400 images from 30 different viewpoints for each scene, keeping the camera-to-world matrices for all such rendered images. Additionally, we render the ground truth depth map for evaluating the predicted 3D position. Each scene contains 2 to 4 objects with different colors. We use 25 images from segmentation and training and 5 for testing randomly. We test on 8 different scenes in total.

4.1.2 Implementation Details

To obtain the groundtruth 3D positions, we train a NeRF model [28] for each scene and generate the depth map from each training viewpoints. By combining camera rays and the depth maps, we obtain the corresponding 3D positions of the pixels.

We use the features of the first 16 layers of pretrained VGG models and upsample the feature maps to the original image size for clustering. After clustering, we dilate the object masks to reduce the mis-clustering of object pixels. For object segmentation, we train a segmentation network for each scene with batch size 10.

4.2. Qualitative Results

4.2.1 Qualitative Comparisons

Figure 3 shows qualitative results of some objects from unseen novel views. Specifically, to visualize the 3D segmentation and each object space, we render the segmented object NeRFs at novel viewpoints and obtain the final 2D masks. Notably, all novel views are rendered *without* using any masks, which validates the converged masks output by our method.

Our method is scene-specific and unsupervised for 3D object segmentation. Notably, this approach is not learning-based and does not need large sets of data on similar scene for training. To our knowledge there is no non-learning-based methods for 3D object segmentation on NeRFs similar to ours. With the 3D information from NeRFs, we can obtain two kinds of masks that will be useful for further applications: 1) the mask for a certain object in original space; 2) the mask for certain object in individual object space. The results show that our method could handle complex textures (e.g., wood coffee table), and complex shapes

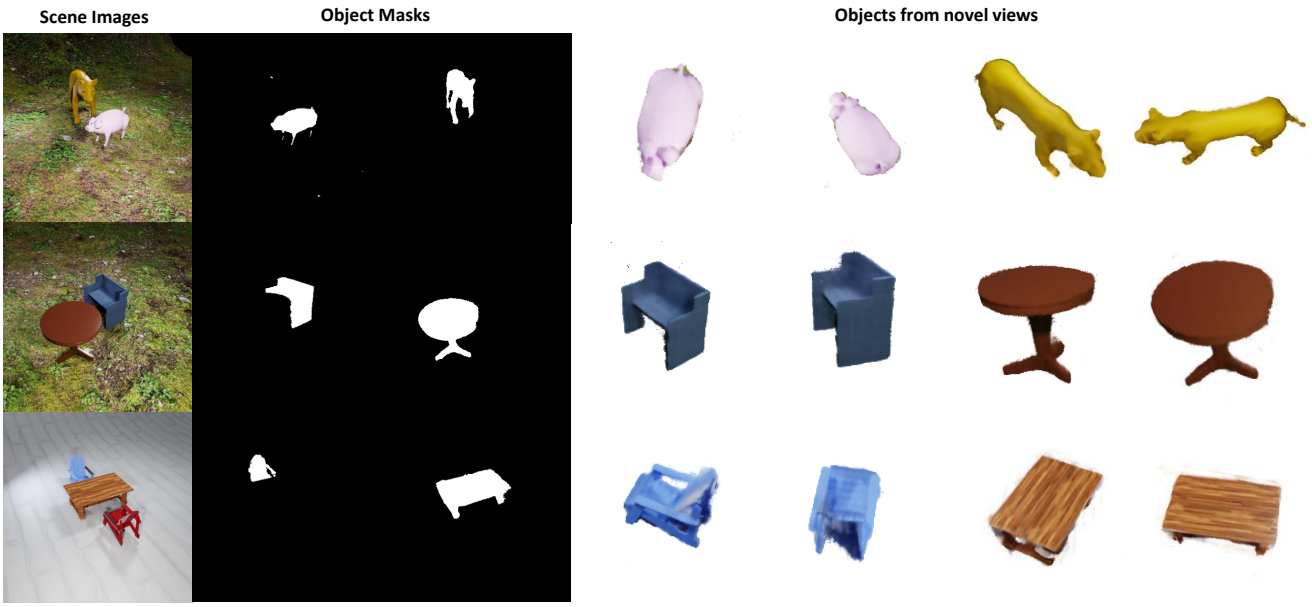

Figure 3. **Qualitative results.** This figure shows a sample input image for each of the three examples, refined masks of the 3D objects in the scene, and output images of different objects from two testing views of each scene. The NeRFs are trained with 25 images.

(e.g., animals) and non-trivial topologies (e.g., genus-n objects such as the chairs).

We compare our results with both 2D unsupervised segmentation methods and 3D learning-based unsupervised segmentation approaches. We cannot obtain reasonable results by directly applying learning-based methods such as uORF [50] on our dataset. In our tests, we use the random camera poses but all the objects are included in the background slot. We also use the training camera poses of uORF to re-render our scenes in blender. Then uORF can obtain some foreground objects for some scenes, shown in Figure 4, which can roughly separate the blue chair from the whole scene while the brown desk still fails to be separated. But the segmented chair is unclear and blurry in the uORF results. For some other scenes, uORF still cannot properly segment the scene even after supplying the given camera poses.

For 2D unsupervised model, we compare with slot attention [26]. We adopt their model which is trained on CLEVR dataset [16], and then fine-tune the pretrained model on each of our scenes respectively. In most of the scenes, slot attention can segment some colors. However, without adequate 3D consideration or supervision, typically [26] cannot cleanly separate the background, especially those scenes with textured background. Moreover, as our dataset is quite diverse, slot attention cannot achieve a very good reconstruction on par with the results produced by our model.

4.2.2 3D Scene Editing

After obtaining the masks and the individual object NeRFs, we can perform 3D scene editing. Generally, we can implement any rigid transformation with transformation matrices on the respective object NeRFs. Moreover, we can perform object insertion and removal by operating on the NeRFs separated by our method. While rendering the composite scene, by integrating the colors of all objects with the density contribution of each object, we can coherently handle occlusion after adding more objects to the scene. Figure 5 shows the results of multiple objects insertion. Given the NeRFs of the objects and a scene, we use the camera pose in the scene space to verify our editing results. Specifically, we use the RGB loss of the out-masked region of the images to get the NeRF of the scene after removing some objects. To translate an object in the scene, we translate the camera position for rendering. For example, suppose we want to move the object to $\mathbf{p}$. Let the pinhole camera be at $\mathbf{o}$ and the object is at the origin. Then we move the camera position $\mathbf{o}' = \mathbf{o} - \mathbf{p}$, where $\mathbf{o}'$ is the new camera position. We sample points from $\mathbf{o}'$ along the original directions to obtain the results after moving objects.

4.2.3 Results on Real Scenes

Figure 6 shows the result tested on a real scene. Although the original images from real scene contain more noise due to complex real shading and non-homogeneous color dis-

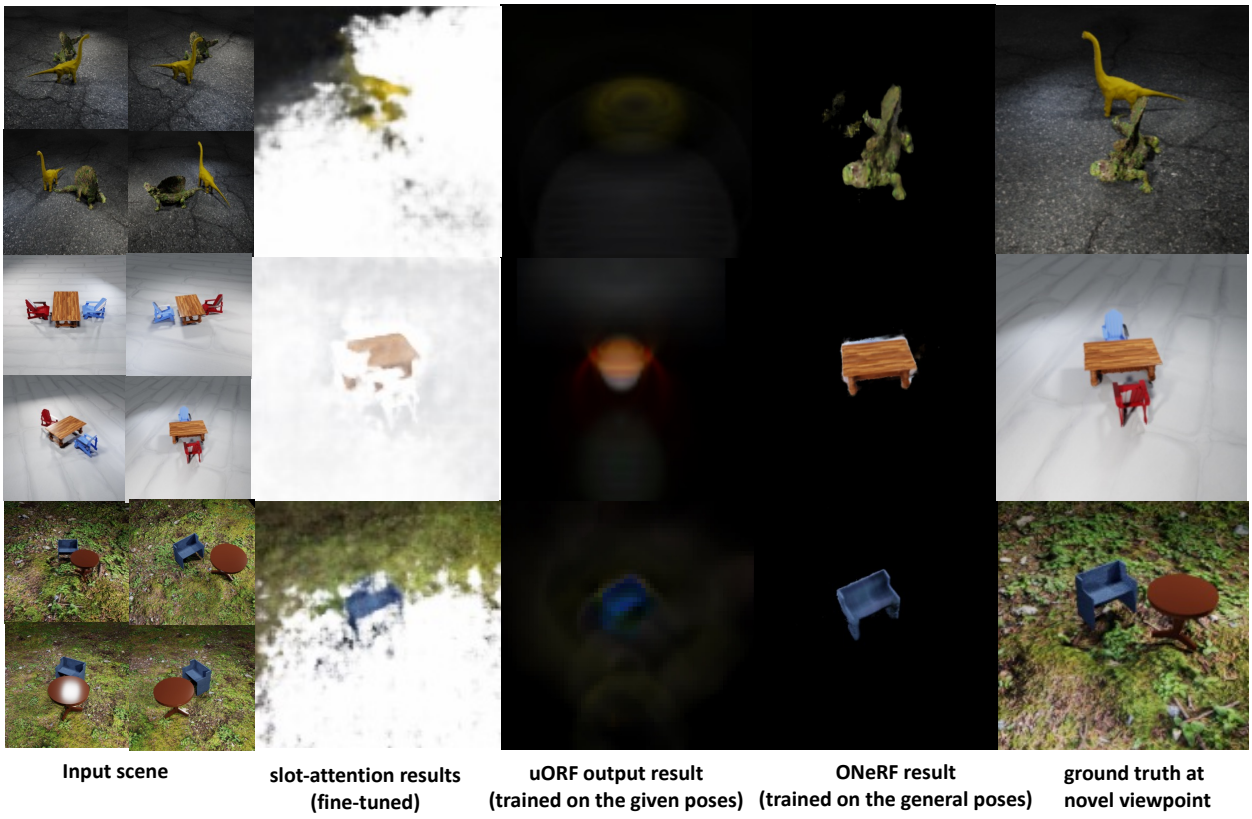

Figure 4. **Qualitative comparison.** Comparison with uORF [50] and slot attention [26]. The results show that our model is the most robust on the scenes with complex shape and color. Learning based methods which rely on large number of scenes with similar color and shape distribution is difficult to generalize on our dataset.

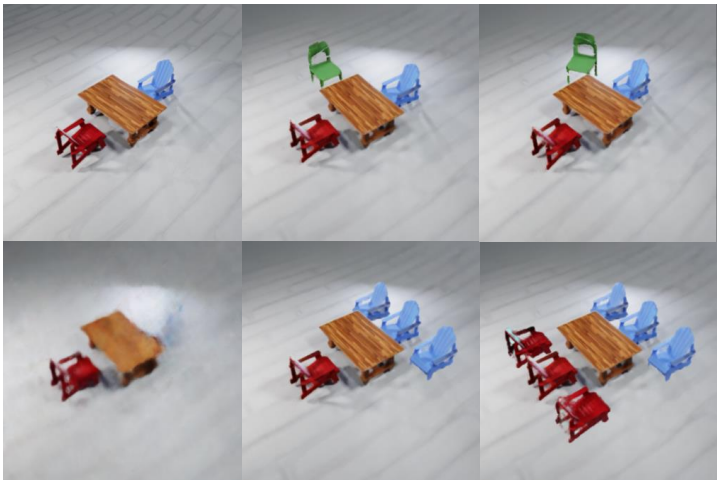

Figure 5. **Results of object insertion, deletion and transformation.** The figure in the top left corner is the original scene image. The other two subfigures in the top row show insertions of a green chair. The bottom left corner shows deletion of the chairs. The remaining two show the results of adding more objects.

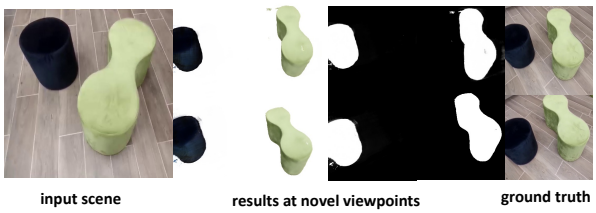

Figure 6. **ONeRF of real scene.** Here we show the result tested on a real scene. We visualize the result rendered at two novel viewpoints.

Method	Mask IoU ↑	Depth Error ↓	PSNR ↑
Ours	**0.86** ±0.0	**0.002** ±0.0	**30.23** ±0.0
uORF [50]	**0.50** ±0.0	-	**23.80** ±0.0
Slot attention [26]	**0.40** ±0.0	-	**18.99** ±0.0

Table 1. **Quantitative comparison.** We evaluate the segmented 3D objects on various metrics by comparing to the ground-truth renderings on from novel unseen viewpoints. For depth error, we normalize the depth between 0 and 1 and find the average mean square error. For uORF, it can only segment 2 objects out of all the 8 so we take the average on these two objects. The depth is used to estimate the 3D position, which is not applicable for [50] and [26].

Method	before refinement	before refinement
Mask IoU ↑	**0.86**	**0.76**

Table 2. **Masks comparison.** We evaluate the IoU errors of all objects over the 8 scenes before and after the refinement step.

tribution on a single object, our method is quite robust to such noise, and achieves a reasonable separation in this real-world scenario.

4.3. Quantitative Evaluation

Tab. 1 shows the quantitative comparison between our method and the two baseline methods, where we can see that in our moderately complex and diverse scenes scenario, our method produces the most stable performance.

4.4. Ablation Studies

Effect of mask refinement. We compare the masks before and after our refinement using intersection over union. The errors and figures for comparison are shown in Table 2 and Figure 7. The refinement step can complete both tables.

Effect of object segmentation network. As shown in the second column of Figure 8, just simple K-means clustering cannot replace our more advanced segmentation model, since the former cannot precisely distinguish coherent object boundary and is very vulnerable to noise.

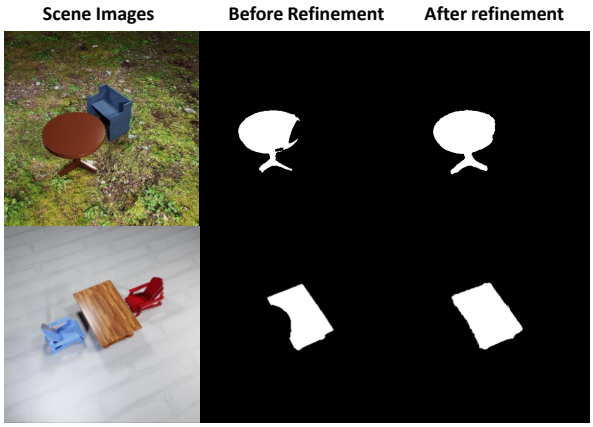

Figure 7. **Masks Comparison.** Here are the masks of the two tables before and after refinement.

Effect of coarse clustering. We use the VGG feature to distinguish the background and foreground objects, and 3D positions to select foreground objects. This method can roughly choose foreground object automatically while removing a large portion of background regions in the images, which helps the subsequent object segmentation.

During our testing, we find that with the same initial K-Means clustering parameters, adding 3D positions can cluster the foreground objects to the specific class where we use 3D information. Additionally, we remove the coarse clustering step, i.e., directly apply the unsupervised segmentation method [20] to the same input images in our testing. Then we replace the predicted 3D positions with random noise in the first step to check whether the positions effects. Both results are shown in Figure 8.

5. Conclusions

This paper presents the first significant step toward segmenting multiple objects NeRFs from multiple views of the underlying scene. Compared with single-image methods, with the 3D from NeRFs, ONeRFs can alternately optimize 2D masks and 3D geometry so that the former can be used to delineate cluttered scenes into coherent object NeRFs, which can then be used to enable a range of 3D scene editing tasks, allowing users to readily insert 3D objects, delete 3D objects, and modify object geometry in addition to novel view synthesis. In the future, we will investigate NeRF completion when the pertinent occluded parts of the object were not photographed by any of the input views. This is achieved by detecting the underlying object symmetry, which would not have been possible without segmenting the scene into individual object NeRFs.

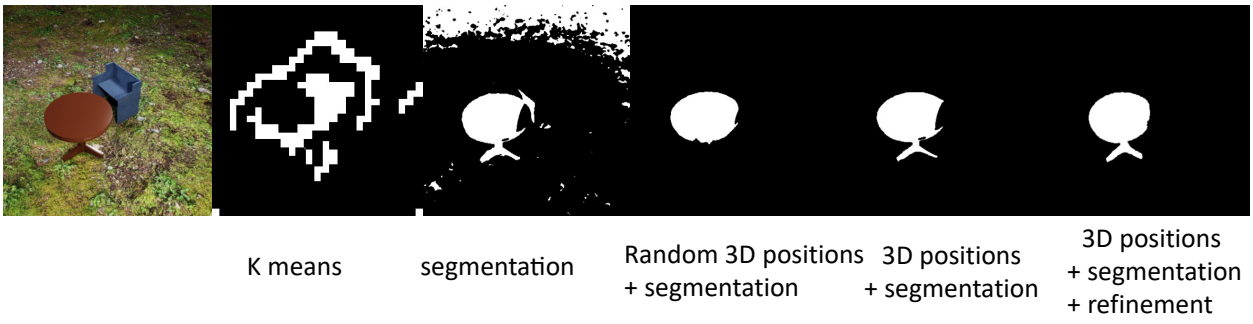

Figure 8. **Ablation study.** Here we visualize the results of the ablation study. From left to right: the ground truth image at novel view, result without segmentation model, result without foreground-background separator, result without correct 3D information, result without refinement and result produced by our full model.

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. In *ICCV*, 2021. 1

[2] Jeremy S. De Bonet and Paul Viola. Poxels: Probabilistic voxelized volume reconstruction. In *ICCV*, 1999. 2

[3] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T. Barron, Ce Liu, and Hendrik P.A. Lensch. Nerd: Neural reflectance decomposition from image collections. In *ICCV*, 2021. 1

[4] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *ICCV*, 2021. 2

[5] Julian Chibane, Aayush Bansal, Verica Lazova, and Gerard Pons-Moll. Stereo radiance fields (srf): Learning view synthesis for sparse views of novel scenes. In *CVPR*, 2021. 1

[6] Blender Online Community. *Blender – a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 4

[7] Eric Crawford and Joelle Pineau. Spatiial invariant unsupervised object detection with convolutional neural networks. In *AAAI*, 2019. 2

[8] Martin Engelcke, Adam R. Kosiorek, Oiwi Parker Jones, and Ingmar Posner. Genesis: Generative scene inference and sampling with object-centric latent representations. In *ICLR*, 2020. 2

[9] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE TPAMI*, 32(8):1362–1376, 2010. 2

[10] Klaus Greff, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Chris Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. 2019. 2

[11] Yudong Guo, Keyu Chen, Sen Liang, Yongjin Liu, Hujun Bao, and Juyong Zhang. Ad-nerf: Audio driven neural radiance fields for talking head synthesis. In *ICCV*, 2021. 1

[12] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *CVPR*, 2020. 2

[13] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *ICCV*, 2021. 1

[14] Zhang Jiakai, Liu Xinhang, Ye Xinyi, Zhao Fuqiang, Zhang Yanshun, Wu Minye, Zhang Yingliang, Xu Lan, and Yu Jingyi. Editable free-viewpoint video using a layered neural representation. In *ACM SIGGRAPH*, 2021. 1

[15] Li Jiang, Shaoshuai Shi, Zhuotao Tian, Xin Lai, Shu Liu, Chi-Wing Fu, and Jiaya Jia. Guided point contrastive learning for semi-supervised point cloud semantic segmentation. In *ICCV*, 2021. 2

[16] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, 2017. 1, 4, 5

[17] Nima Khademi Kalantari, Ting-Chun Wang, and Ravi Ramamoorthi. Learning-based view synthesis for light field cameras. *ACM TOG*, 35(6), 2016. 2

[18] Jun-Yan Zhu Kangle Deng, Andrew Liu and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. *arXiv preprint arXiv:2107.02791*, 2021. 1

[19] Laurynas Karazija, Iro Laina, and Christian Rupprecht. Clevrtex: A texture-rich benchmark for unsupervised multi-object segmentation. In *NeurIPS*. 4

[20] Wonjik Kim, Asako Kanezaki, and Masayuki Tanaka. Unsupervised learning of image segmentation based on differentiable feature clustering. *IEEE TIP*, 29, 2020. 3, 7

[21] Vladimir Kolmogorov and Ramin Zabih. Multi-camera scene reconstruction via graph cuts. In *ECCV*, 2002. 2

[22] Kiriakos N. Kutulakos and Steven M. Seitz. A theory of shape by space carving. *IJCV*, 38:307–314, 2000. 2

[23] Zhixuan Lin, Yi-Fu Wu, Skand Vishwanath Peri, Weihao Sun, Gautam Singh, Fei Deng, Jindong Jiang, and Sungjin Ahn. Space: Unsupervised object-oriented scene representa-

tion via spatial attention and decomposition. In *ICLR*, 2020. 2

[24] Steven Liu, Xiuming Zhang, Zhoutong Zhang, Richard Zhang, Jun-Yan Zhu, and Bryan Russell. Editing conditional radiance fields. In *ICCV*, 2021. 1

[25] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *CVPR*, 2019. 2

[26] Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. In *NeurIPS*, 2020. 2, 5, 6, 7

[27] Keyang Luo, Tao Guan, Lili Ju, Haipeng Huang, and Yawei Luo. P-mvsnet: Learning patch-wise matching confidence aggregation for multi-view stereo. In *ICCV*, 2019. 2

[28] 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, 3, 4

[29] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. *ICCV*, 2021. 1

[30] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *CVPR*, 2021. 1

[31] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 2

[32] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *NeurIPS*, 2017. 2

[33] Shi Qiu, Saeed Anwar, and Nick Barnes. Geometric back-projection network for point cloud classification. *IEEE Transactions on Multimedia*, 2021. 2

[34] Shi Qiu, Saeed Anwar, and Nick Barnes. Semantic segmentation for real point cloud scenes via bilateral augmentation and adaptive fusion. In *CVPR*, 2021. 2

[35] Ramesh Raskar, Wojciech Matusik, Chris Buehler, Steven Gortler, and Leonard Mcmillan. Image-based visual hulls. *SIGGRAPH 2000*. 2

[36] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *NeurIPS*, 2020. 1

[37] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 2

[38] Karl Stelzner, Kristian Kersting, and Adam R. Kosiorek. Decomposing 3D Scenes into Objects via Unsupervised Volume Segmentation. *arXiv preprint arXiv:2104.01148*, 2021. 1

[39] Vadim Tschernezki, Diane Larlus, and Andrea Vedaldi. NeuralDiff: Segmenting 3D objects that move in egocentric videos. In *3DV*, 2021. 1

[40] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds. *ACM TOG*, 2019. 2

[41] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. NeRF−−: Neural radiance fields without known camera parameters. *arXiv preprint arXiv:2102.07064*, 2021. 1

[42] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015. 1

[43] Xu Yan, Chaoda Zheng, Zhen Li, Sheng Wang, and Shuguang Cui. Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling. In *CVPR*, 2020. 2

[44] 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. 1, 3

[45] Jiayu Yang, Wei Mao, Jose M. Alvarez, and Miaomiao Liu. Cost volume pyramid based depth inference for multi-view stereo. In *CVPR*, 2020. 2

[46] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. *ECCV*, 2018. 2

[47] Christopehr Choy Animashree Anandkumar Minsu Cho Yoonwoo Jeong, Seokjun Ahn and Jaesik Park. Self-calibrating neural radiance fields. In *ICCV*, 2021. 1

[48] 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. 1

[49] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *CVPR*, 2021. 1

[50] Hong-Xing Yu, Leonidas J. Guibas, and Jiajun Wu. Unsupervised Discovery of Object Radiance Fields. *arXiv preprint arXiv:2107.07905*, 2021. 1, 5, 6, 7

[51] Biao Zhang and Peter Wonka. Point cloud instance segmentation using probabilistic embeddings. In *CVPR*, 2021. 2

[52] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv:2010.07492*, 2020. 1

[53] Xiuming Zhang, Pratul P Srinivasan, Boyang Deng, Paul Debevec, William T Freeman, and Jonathan T Barron. Nerfactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM TOG*, 2021. 1

[54] Yachao Zhang, Yanyun Qu, Yuan Xie, Zonghao Li, Shanshan Zheng, and Cuihua Li. Perturbed self-distillation: Weakly supervised large-scale point cloud semantic segmentation. In *ICCV*, 2021. 2

[55] Na Zhao, Tat-Seng Chua, and Gim Hee Lee. Few-shot 3d point cloud semantic segmentation. In *CVPR*, 2021. 2

[56] Yunsong Zhou, Hongzi Zhu, Chunqin Li, Tiankai Cui, Shan Chang, and Minyi Guo. Tempnet: Online semantic segmentation on large-scale point cloud series. In *ICCV*, 2021. 2