In this supplementary material, we introduce the model architecture in Sec. A, the in-painting details in Sec. B, more experimental results in Sec. C, and further elaboration on joint scene decomposition and composition in Sec. D

A. Model Architecture

The model architecture is shown in Fig. 11. Following NeRF [18] and ObjectNeRF [37], we utilize a 4-layer Multi-Layer Perceptron (MLP) as the shared backbone for learning generic implicit representations of scene points. The implicit point representations are then passed to specific heads for predicting object/background radiance fields. We use three MLPs for the density prediction and five MLPs for the color prediction. In the fine stage, we use three separate heads with the same structure for the background, object, and guidance prediction head. In the coarse stage, only the guidance prediction head is used. We set the positional embedding frequency to 10 for the input coordinates and 4 for the input ray direction.

B. Implementation of In-painting Process

To learn a better background-specific radiance field, especially the extrapolation on the occluded region, we involve a pre-trained in-painting model lama [31] to fill the uncertain occluded regions on the 2D images. Specifically, as shown in Fig. 12, we pass the original images with their corresponding foreground object masks to the pre-trained lama model. Then we use the in-painted pixel colors of the background region as pseudo color supervision. To mitigate imprecise mask contours, we apply an appropriate erosion for the foreground masks before passing the images to the lama model.

C. More Experimental Results

C.1. More Results on Scene Rendering

We show more qualitative scene rendering results on the ScanNet dataset in Fig. 13. Compared with other state-of-the-art methods ObjectNeRF [37] and ObjectSDF [35], our rendered images produce more fine-grained details on the overall-view quality and object details. Note that for the scene rendering comparison, the test-set images are directly provided by the authors of ObjectNeRF [37], which is the same as their paper, while different from the train-test split of their open-sourced code.

C.2. More Results on Scene Editing

We show more qualitative scene editing results on both ToyDesk and ScanNet datasets in Fig. 14. As we can see, by re-organizing the object radiance fields, we can generate images with object manipulations, including removal, duplication, and changing object position (*e.g.*, the movement and rotation).

ObjectSDF [35] focuses on implicit object geometry modeling. Although ObjectSDF mentioned that scene editing can be a potential application, it did not address the object editing problem in their paper or provide any open-sourced code for editing. In contrast to them, we focus on editable novel view synthesis, where novel view scene rendering and editing are both our important objectives. In Fig. 10, we conducted a decomposition comparison with ObjectSDF [35], using two editing operations, *i.e.*, object removal and object extraction. As highlighted by the red arrows, our model can achieve finer-grained object-level rendering results, with a clearer background decomposition (see black shadows on the table).

C.3. Video Demos on Scene Editing

We also upload one editing video demo, which can provide more qualitative results to directly compare with ObjectNeRF [37]. The video demos can clearly show that our model generates superior editing results compared to ObjectNeRF [37], with better video quality and background decomposition.

D. Further elaboration on joint scene decomposition and composition

Our framework targets joint scene editing and synthesis. The scene decomposition can provide the capability of learning disentangled representations of different background/objects, allowing for scene editing, while scene composition learns an entire scene representation for novel view synthesis. The decomposition and composition are jointly modeled and are technically correlated in our unified framework. As shown in Fig. 1 (paper), the forward pass of the network composes the background/objects into the whole scene, while the gradient backward pass can assist in the decomposition process. Thus, these two can be united to facilitate the consistency constraint in the unified optimization framework. We will add the discussion on this point in the revision.

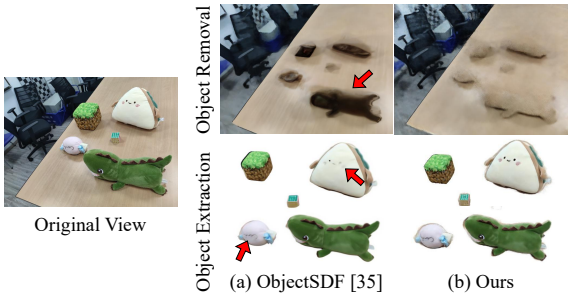


Figure 10. The scene editing comparison with ObjectSDF [35], using editing operations of both object removal and object extraction for the scene decomposition on ToyDesk2.

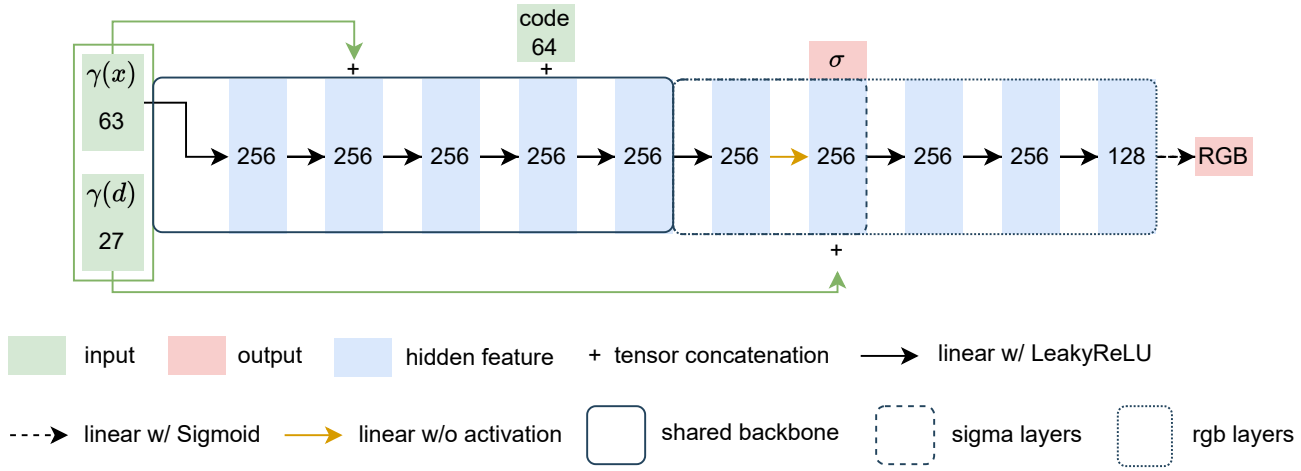
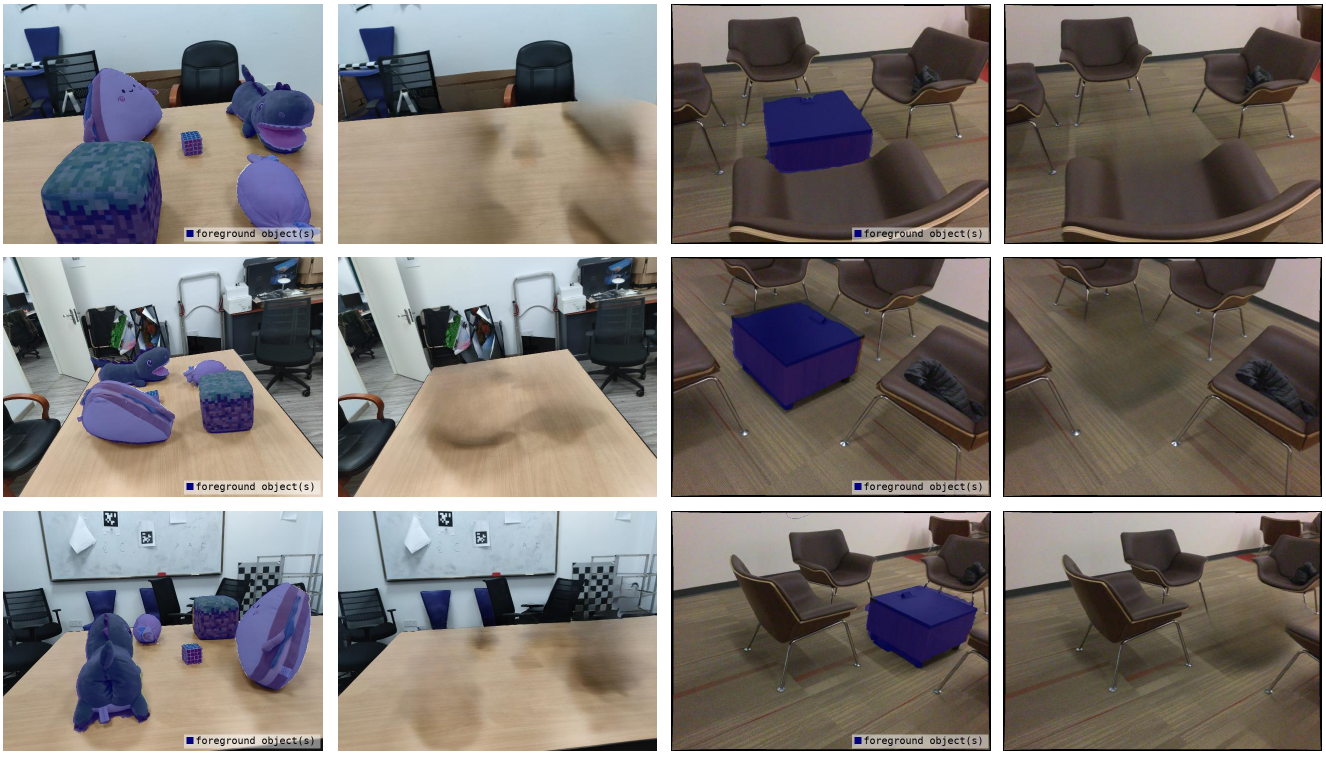
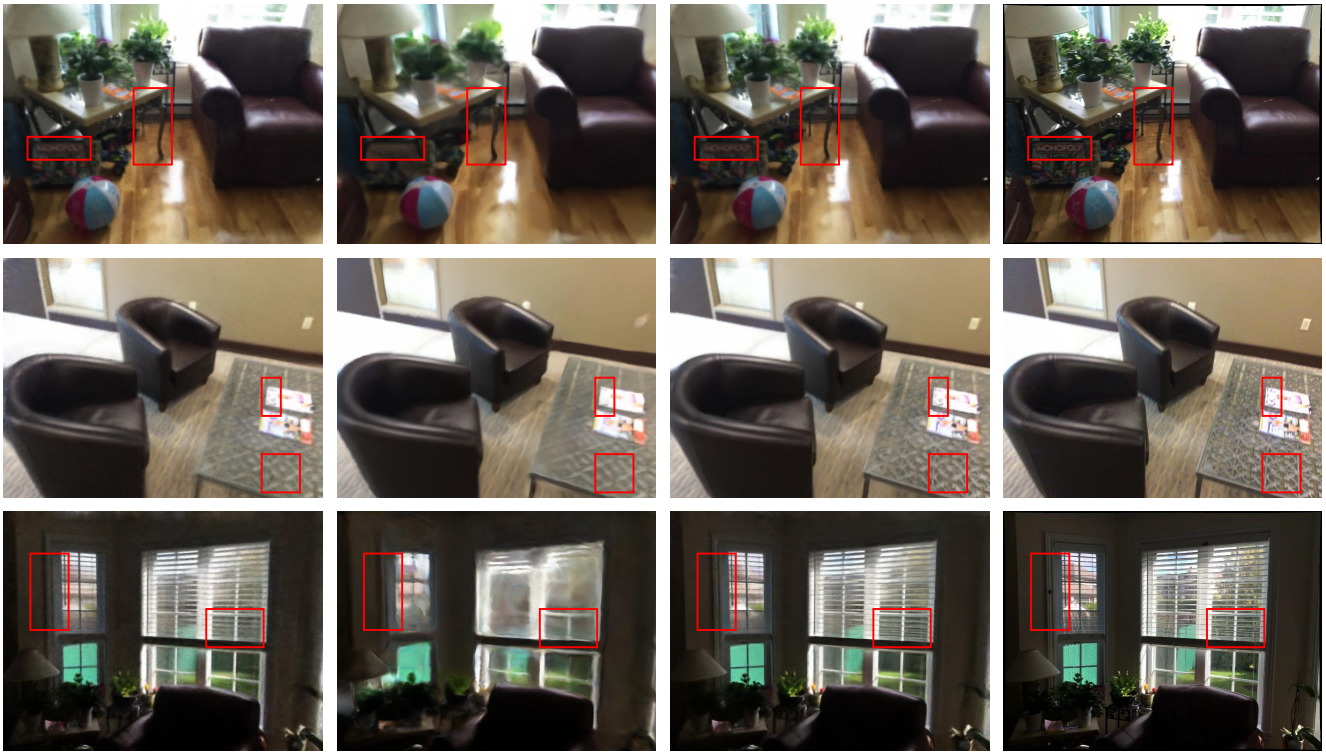


Figure 11. Illustration of the detailed model architecture. We set the positional embedding frequency to 10 for the input coordinates x and 4 for the input ray direction d, so the input embedding dimensions of these two are $3+3\times2\times10=63$ and $3+3\times2\times4=27$, respectively. The volume density σ and RGB color are predicted by specific layers on the top of the shared backbone. We use three separate heads with same structure for the background, object and guidance prediction.



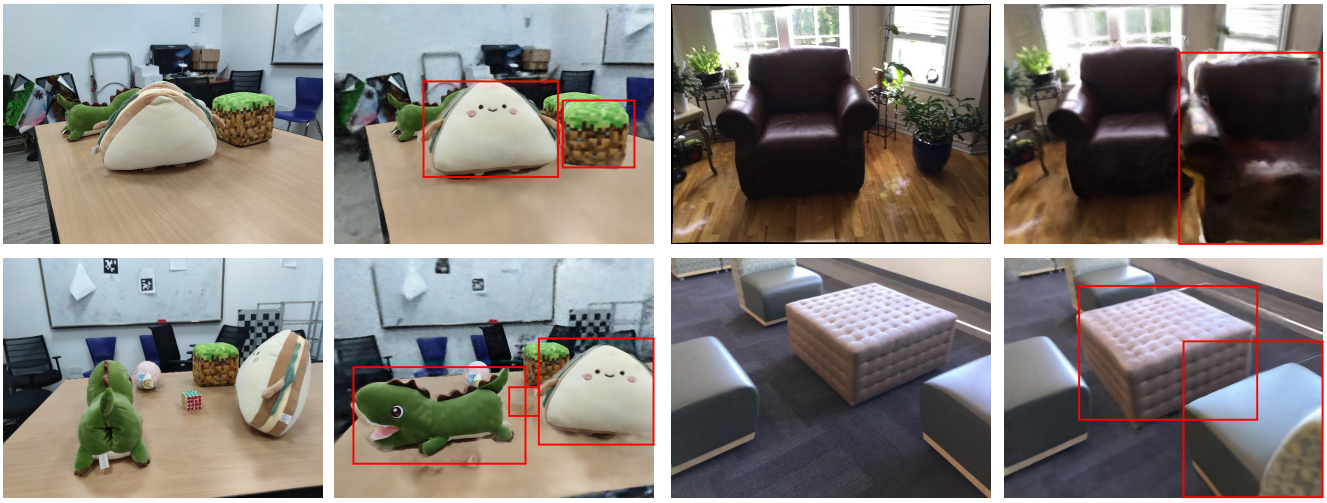
(a) Masked Region (b) In-painted Image (a) Masked Region (b) In-painted Image

Figure 12. Illustration of the In-painting process. (a) shows the original image and foreground object masks, while (b) shows the in-painted images from the pre-trained lama model. The uncertain occluded regions are filled with pseudo colors. Note that the 2D in-painting may bring new ambiguity for the regions seen in other views. For example, the lama model does not correctly in-paint the armrest (in row 1, column 2) and the chair leg (in row 3, column 4). However, supervision from other views can mitigate most of the ambiguities.



(a) ObjectNeRF (b) ObjectSDF (c) Our Method (d) Ground Truth

Figure 13. More qualitative scene rendering results on the ScanNet dataset. We highlight the details in the red boxes. Our rendering results outperform other SoTA methods ObjectNeRF [37] and ObjectSDF [35] with higher overall-view quality and more object details.



(a) Original View (b) Edited Image (a) Original View (b) Edited Image

Figure 14. More qualitative scene editing results on both ToyDesk and ScanNet datasets. We highlight the editings in the red boxes, including duplication (the green cube, triangle toy, and the sofa), removal (the Rubik's cube), rotation (the dinosaur and the triangle toy), and movement (the sofa and the mattress).