A Training and Model Architectures

In the experiments, during the training of 200K iterations, the loss L in Equation 4 is minimized by Adam with a linearly decaying learning rate (5e-4 to 8e-5). During training, Gaussian noise for density is also applied. The number of coarse and fine samplings is 64 and 128, respectively. The MLP of the neural radiance field consists of eight ReLU layers with 256 dimensions, followed by a linear layer for density, three layers for color, and three layers for feature, as shown in Fig. 1. Positional encoding of length 10 is used for the input coordinate and its skip connection, and that of length 4 is for viewing direction. If an independent MLP is prepared for the feature field, it consists of four layers (with a skip connection at the third layer if the positional encoding is used). The size of a training image is 320×240 for the Replica dataset and 1008×756 for the other datasets. The batchsize of training rays is 1024 for Replica and 2048 for the others. During finetuning of feature fields or radiance fields, Gaussian noise is removed, and the learning rate is set to 1e-4.

For segmentation, unless otherwise stated, we use thresholded cosine similarity to directly compute the probability of a query instead of softmax with negative queries in Eq. 5 and set $\mathbf{p} = 1$ if the similarity exceeds the threshold, and $\mathbf{p} = 0$ otherwise for hard decomposition. In Fig. 3 and 5, the results of the room scene used a label set, {whiteboard, ceiling, light, television, wall, bin, table, cabinet, cable, chair, box, floor}, for skipping tuning of thresholds.

B Editing Procedure

Editing for colorization, translation, and deletion proceed as follows:

(1) Sample points $[..., \mathbf{x}_i, ...]$ on a ray as usual in NeRF.
(2) For each point, we query the DFF. We can now calculate the probability of the coordinate being matched with a set of queries as $p \in [0, 1]$. We now define "the coordinate is selected by the query" if p is above a user-defined threshold, otherwise "not selected". We can also use the mixture of selected and not-selected results in the proportion of p without the threshold.
(3) If *not* selected, we calculate density $\sigma(\mathbf{x})$ and color $\mathbf{c}(\mathbf{x})$ at $\mathbf{x}$ via the vanilla NeRF.
(4) If selected, we may apply the following transforms:
(4-A) Deletion (Fig. 4): We set the density $\sigma(\mathbf{x})$ of the point to zero.
(4-B) Color editing: We query the NeRF for density $\sigma(\mathbf{x})$ and color $\mathbf{c}(\mathbf{x})$ by querying the NeRF. The color is then edited by a colorization function $\mathbf{b}$, i.e., it is transformed to $\mathbf{b}(\mathbf{c}(\mathbf{x}))$.
(4'-C) Translation / rescaling: Geometric transformation needs another step before performing (2). We first compute a deformed point coordinate $\mathbf{x}'$: $\mathbf{x}'$ is computed by applying the inverse of the editing transformation; that is, $\mathbf{x}' = \mathbf{g}^{-1}(\mathbf{x})$. For translation, $\mathbf{g}$ would be a simple addition with a vector. If $\mathbf{x}'$ is selected by the query, $\mathbf{x}'$ is used instead of $\mathbf{x}$ for calculating color and density. If both $\mathbf{x}$ and $\mathbf{x}'$ are selected and have non-zero density (e.g., the boundary between the deformed apple and others in Fig. 7), we mix their colors $\mathbf{c}(\mathbf{x})$ and $\mathbf{c}(\mathbf{x}')$ in the ratio of their alphas at the point for simplicity.
(5) Finally, as usual, we perform volume rendering with the series of (density, color) tuples.

C Feature Encoders

We investigate two teacher networks, LSeg and DINO, which are pre-trained and publicly available[10]. Each training image is encoded by the encoders of the networks and used as target feature maps, $\mathbf{f}_{\text{img}}(I, r)$, defined in Equation 4. Because the feature maps are of reduced sizes, we use them after resizing to the original image size via interpolation.

For LSeg, we use the official demo model, which has the ViT-L/16 image encoder and CLIP's ViT-B/32 text encoder. Inference follows the official script and uses multi-scale inference (scales = [0.75, 0.83, 0.92, 1.0, 1.08, 1.17, 1.25] for Replica, [0.75, 1.0, 1.25, 1.5, 1.75, 2.0, 2.25] for the others)[11]. The model is trained on seven different datasets [42], including ADE20K [115], BDD [108],

[10] https://github.com/isl-org/lang-seg https://github.com/facebookresearch/dino

[11] Multi-scale inference stabilizes features and alleviates artifacts due to discrete patch processing in ViT. Inference with large scale means inference with a zoomed and cropped image. Although it may increase the effective resolution of feature maps, it loses context information and may produce noisy features with wrong semantic understanding.

Cityscapes [19], COCO-Panoptic [46, 11], IDD [95], Mapillary Vistas [58], and SUN RGBD [84]. Possibly because the mixed dataset is biased, especially to traffic scenes, we experimentally confirmed that LSeg does not work well in out-of-distribution regions and queries. For example, small objects or parts are often not discriminative by the network. More careful training will improve its zero-shot ability in various domains and that of student neural feature fields.

For DINO, we use the extended implementation by Amir et al. [3][12]. It uses overlapping patches with stride 4 to enlarge the feature map's size. We use the feature at the 11th layer of the dino_vits8 model taking a 448x448-resized image as input. We average it with the features of the horizontally flipped image. While DINO cannot accept text queries, its feature captures more fine-grained information than LSeg's.

We visualize features of teacher networks, LSeg and DINO, in Fig. 9. We also visualize features of NeRF in Fig. 5. For visualizing feature spaces, we use scikit-learn's `sklearn.decomposition.PCA` [67]. We calculate 3-dimensional PCA components using the teacher's feature map of the first frame of the training view images, and visualize features by the components as RGB, normalized with min-max values with outlier removal. Although the 3-dimensional PCA cannot perfectly visualize the feature space, we show the results for reference and understanding.

As explained, although the features of both models are sufficiently view-consistent, their semantic resolution seems different. LSeg tends to show boundaries with coarse-grained categories, possibly due to the bias in the training datasets. The result will be changed if we train LSeg with other datasets focusing on fine-grained parts or regions. In contrast to LSeg, DINO produces fine-grained features where we can feel most of the original edges. This enables a wide range of decomposition based on various queries as we confirmed in the experiments. However, even with DINO, the finest-grained details in high resolution are still challenging to capture, e.g., the ribs of T-rex. The improvement of feature encoders will push the boundary of the decomposition quality of complex objects.

D Replica Dataset Experiment

We experiment distilled feature field on 3D semantic segmentation in Section 5.1 for demonstrating basic ability of segmentation of a 3D space and text queries. Because 3D zero-shot semantic segmentation is a novel task except for some studies Michele et al. [54], no standard benchmark datasets exist. Furthermore, even for 3D semantic segmentation with closed label sets, there is no dataset with high-quality images, accurate point clouds[13], and reasonable annotations. For proof-of-concept of our work, we created a dataset from the existing Replica dataset [86]. It is a moderate-quality room reconstruction dataset with semantic segmentation labels. We use four scenes, room_0, room_1, office_3, and office_4 with posed images rendered by Zhi et al. [112]. Because the pose trajectory was randomly generated, some images are unrealistically rendered (e.g., rendered from a camera inside furniture). We filtered such images from training and evaluation. Near and far of NeRF follow their setting, (0.1, 10.0), except for office_3 (0.1, 15.0). Although the quality of reconstructions is better than other datasets like ScanNet [20], it still suffers from collapsed geometry, less photo-realistic appearances, and defective annotations. Some objects are difficult to predict their labels due to bad appearance, label ambiguity, or both. Because this evaluation is not intended to measure the ability to overcome such unrealistic biases or artifacts in a dataset, we semi-automatically fix the label set of the dataset. We first apply LSeg to predict the semantic label map of each RGB image and evaluate the accuracy against Replica's original label map. In the evaluation, the accuracy of some labels is almost or exactly zero. We ignore points with such labels from the evaluation. In addition, noticeable label ambiguity is also fixed manually by ignoring the label or re-labeling (e.g., 'rug' and 'floor' are merged as they are intrinsically nonexclusive and often indistinguishable in the dataset). For reference, we show a visualization of predictions by LSeg and Replica's ground truth in Fig. 10, and the confusion matrix in Fig. 11. Note that it is not guaranteed that training images cover all the regions of the scene and its point cloud. Hence, the evaluation also measures the ability of generalization to unobserved regions via propagation.

[12] https://github.com/ShirAmir/dino-vit-features

[13] If the input point clouds exist at geometrically wrong coordinates, in reality, their annotated labels could be wrong. It is problematic, especially for evaluating point-cloud-agnostic models like our model.

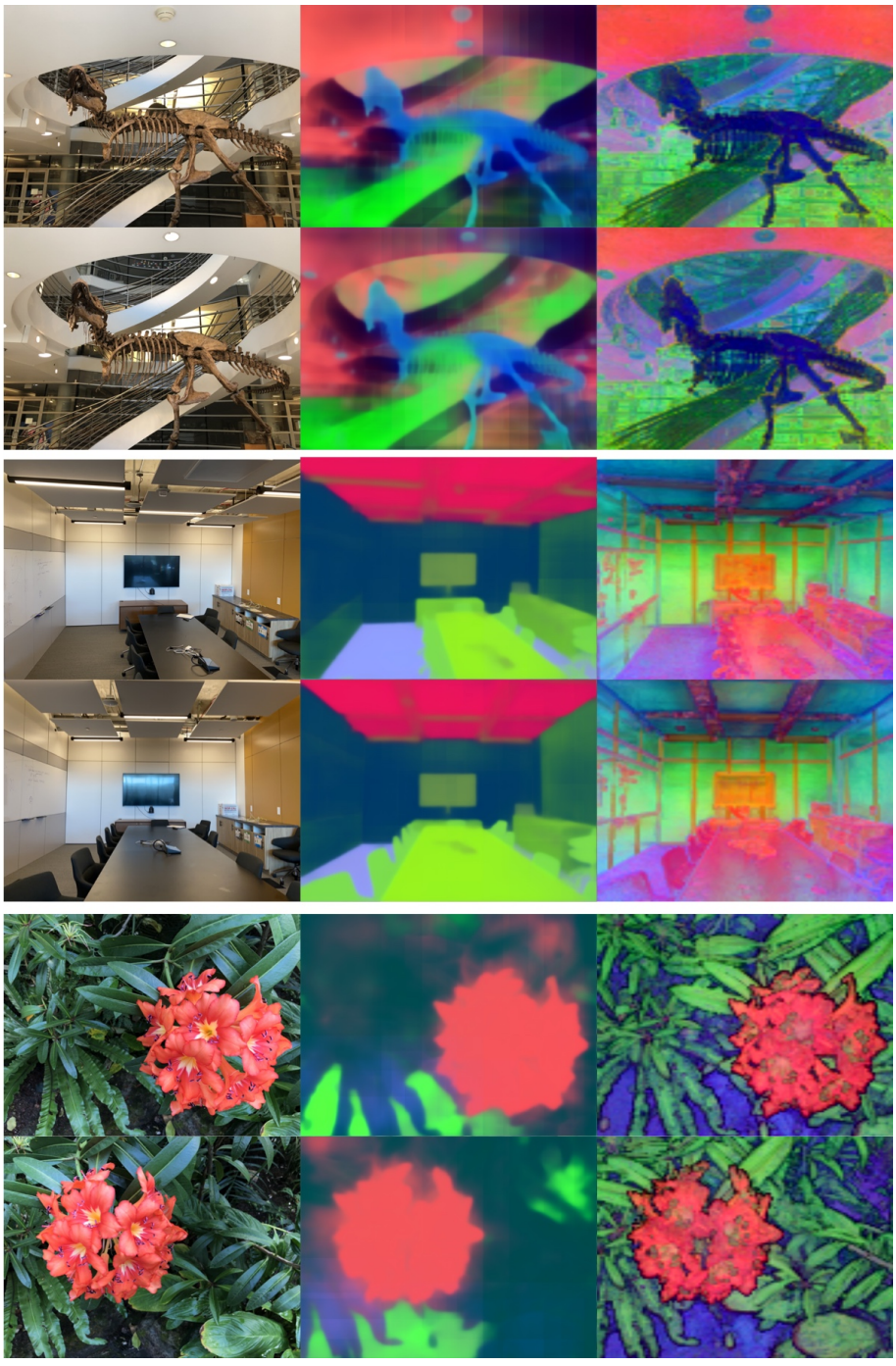

Figure 9: Visualization of features of LSeg (center) and DINO (right). Note that it is invalid to compare colors themselves between different scenes or models.

E Ablation Experiments of Variants

We experiment with some variants of DFF architectures and show the result in Table 3. While they could indicate different behavior for each scene, the average performance difference was marginal. In total, using MLP trained with volume rendering of coarse sampling performs better than MLP with fine sampling based on hierarchical sampling. The coarse sampling might implicitly regularize MLP

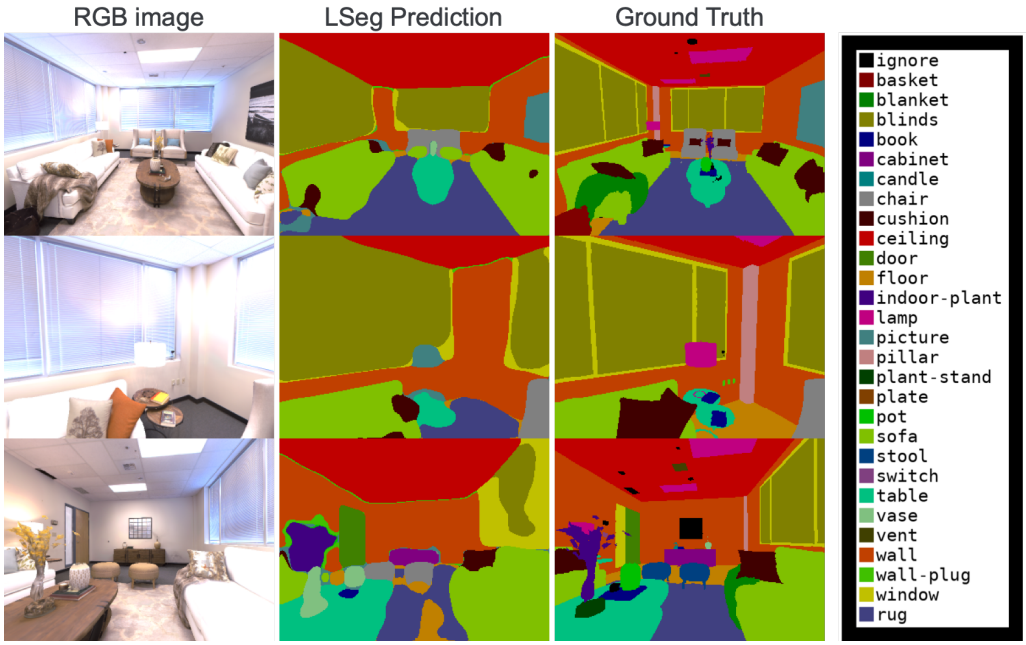

Figure 10: Visualization of prediction by the LSeg (teacher network) on the image semantic segmentation task on Replica's room_0.

Table 3: Performance of zero-shot semantic segmentation and novel view synthesis on Replica. mIoU is calculated on the 3D point cloud. @2 allows the matching of labels with the second-highest probability. 2D accuracy to teacher indicates the agreement ratio with the labels predicted by the LSeg teacher network. PSNR, SSIM, and LPIPS are metrics of image synthesis. δ<1.25 and absrel are metrics of geometry (depth estimation).

	mIoU	-@2	2D acc. to teacher	PSNR↑	SSIM↑	LPIPS↓	δ<1.25↑	absrel↓
basic NeRF	-	-	-	32.87	0.934	0.148	0.993	0.018
Coarse MLP								
indep. freq4	0.544	0.760	-	-	-	-	-	-
branch	0.562	0.779	-	-	-	-	-	-
branch ($\lambda \times 10$)	0.565	0.790	-	-	-	-	-	-
Fine MLP								
indep. freq4	0.553	0.774	0.941	32.87	0.934	0.148	0.993	0.018
branch	0.553	0.784	0.942	32.85	0.932	0.150	0.993	0.017
branch ($\lambda \times 10$)	0.543	0.770	0.942	32.68	0.927	0.162	0.993	0.018

and help to obtain smoothness properties. Even if increasing the weight of feature loss, λ, it did not improve the performance and hurt the quality of view synthesis.

F Implementation of CLIPNeRF Experiment

Because the official implementation of CLIPNeRF [97] is not available, we implement it from the description by ourselves. In addition, the experiment for Figure 14 in their paper (i.e., editing of the LLFF scene) unfortunately lacks significant descriptions to be required for reproduction. Thus, there might be nuanced differences while we confirmed that our result shows similar behaviors. We follow the description in the paper as much as possible. For 500 iterations, we optimize the parameters of the NeRF while freezing the sub-parameters related to density. It minimizes negative cosine similarity between a text prompt and a rendered image from a randomly sampled pose in the training dataset, calculated by CLIP ViT-B/32, with Adam of learning rate 0.0001. Because a naive differentiable rendering of high-resolution images is intractable due to memory constraints, we use an efficient

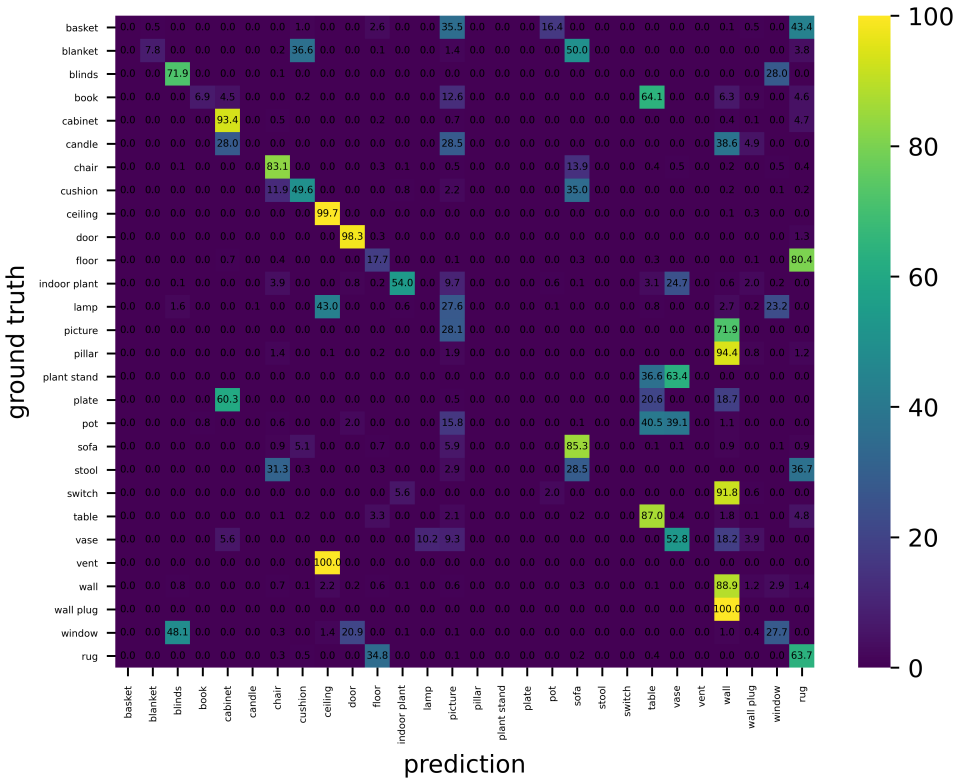

Figure 11: Confusion matrix of the LSeg on the image semantic segmentation task on Replica room_0 (full label setting).

method. We render a patch of 128×128 size, followed by bilinear interpolation for resizing and feed it to CLIP. For making the patch cover the view widely, rays in 128×128 patch are sampled with stride interval 5; thus, CLIP's receptive field is 640×640 of the original image. For memory efficiency, during the rendering of a patch, we reduce the number of importance sampling to 16; fine MLP uses $64 + 16 = 80$ in total. While it is out of the scope of this paper, it is possible to further increase the patch resolution if we use a recomputation technique, where NeRF first computes depth without the need for gradient, and then, renders a patch with a few samples nearby the depth surface per ray in a differentiable manner.

For demonstrating that distilled feature field can be used with other 3D scene representations, we additionally experimented with the InstantNGP [57] and the LSeg-DFF, followed by CLIPNeRF editing. The implementation is derived from https://github.com/kwea123/ngp_pl. For computational efficiency, we do not perform the complete process of volume rendering for feature rendering. We first compute the pseudo depth of the surface as a weighted average of the depth of each sampled point in volume rendering. We simply compute the feature in five points around the pseudo surface point by adding small perturbations during training and inference. This method is very efficient but works for training the distilled feature field. When optimizing the scene with CLIPNeRF, instead of backpropagating gradients to all the rays, we backpropagate the gradients to only the rays selected by "apple". The result of the "rainbow apple"-edit is shown in Fig. 13.

G CLIP-inspired Segmentation Models

We use 2D vision(-and-language) models as teacher networks for distillation. In particular, for processing text queries, we use LSeg [44] as a teacher network. We can use other models for teacher

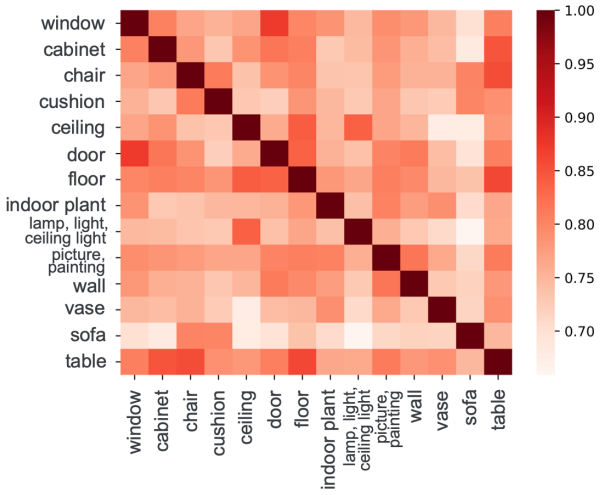

Figure 12: Similarity matrix of text label features by LSeg of Replica room_0.

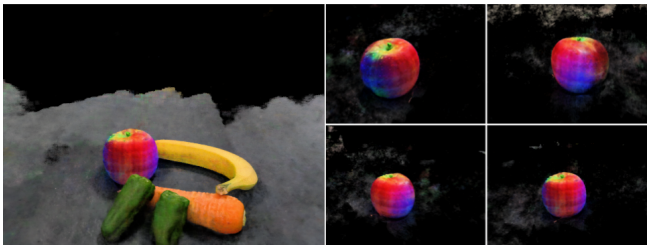

Figure 13: Editing of an InstantNGP scene with CLIPNeRF and extraction.

networks. As introduced in Section 2, within a few months[14], many studies [44, 52, 99, 103, 72] concurrently tackled image semantic segmentation tasks with similar approaches using CLIP [69]. They use the pretrained CLIP models as a text encoder and an image feature encoder in freezing or finetuning manners and train zero-shot image semantic segmentation models. Because they were concurrent, differences in various choices and performances are missing so far. While we use LSeg by Li et al. [44] for simplicity of implementation, it is possible to use other models, especially text-unconditional encoders, where the image encoder produces a feature map without using text labels and calculates a score map by pixel-level calculations with text labels. Although the publicly available LSeg model can accept any text queries thanks to the CLIP's text encoder, it is not so robust to out-of-distribution data (e.g., "fossils", "T-Rex", "Triceratops") due to their training strategy overfitting to the bias in its training datasets (e.g., traffic scenes). Using other concurrent models or the ongoing improvement of training and architectures [50] will alleviate such issues and improve distilled feature field.

[14] Li et al. [44] appeared anonymously at OpenReview on September 29th, 2021. The others appeared at arXiv on November or December, 2021.