

Nerflets: Local Radiance Fields for Efficient Structure-Aware 3D Scene Representation from 2D Supervision

Xiaoshuai Zhang^{1,3} Abhijit Kundu¹ Thomas Funkhouser¹
 Leonidas Guibas^{1,2} Hao Su³ Kyle Genova¹

¹ Google Research

² Stanford University

³ University of California, San Diego

Nerflets Out

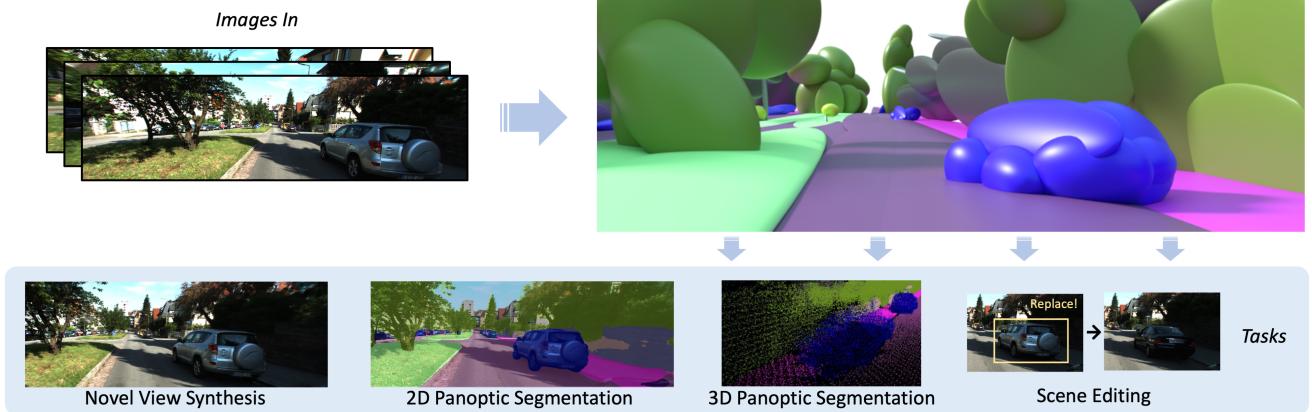


Figure 1. We propose to represent the scene with a set of local neural radiance fields, named nerflets, which are trained with only 2D supervision. Our representation is not only useful for 2D tasks such as novel view synthesis and panoptic segmentation, but also capable of solving 3D-oriented tasks such as 3D segmentation and scene editing. The key idea is our learned structured decomposition (top right).

Abstract

We address efficient and structure-aware 3D scene representation from images. Nerflets are our key contribution—a set of local neural radiance fields that together represent a scene. Each nerflet maintains its own spatial position, orientation, and extent, within which it contributes to panoptic, density, and radiance reconstructions. By leveraging only photometric and inferred panoptic image supervision, we can directly and jointly optimize the parameters of a set of nerflets so as to form a decomposed representation of the scene, where each object instance is represented by a group of nerflets. During experiments with indoor and outdoor environments, we find that nerflets: (1) fit and approximate the scene more efficiently than traditional global NeRFs, (2) allow the extraction of panoptic and photometric renderings from arbitrary views, and (3) enable tasks rare for NeRFs, such as 3D panoptic segmentation and interactive editing. *Our project page.*

1. Introduction

This paper aims to produce a compact, efficient, and comprehensive 3D scene representation from only 2D images. Ideally, the representation should reconstruct appear-

ances, infer semantics, and separate object instances, so that it can be used in a variety of computer vision and robotics tasks, including 2D and 3D panoptic segmentation, interactive scene editing, and novel view synthesis.

Many previous approaches have attempted to generate rich 3D scene representations from images. PanopticFusion [36] produces 3D panoptic labels from images, though it requires input depth measurements from specialized sensors. NeRF [34] and its descendants [3, 4, 35, 41] produce 3D density and radiance fields that are useful for novel view synthesis, surface reconstruction, semantic segmentation [52, 62], and panoptic segmentation [5, 23]. However, existing approaches require 3D ground truth supervision, are inefficient, or do not handle object instances.

We propose *nerflets*, a 3D scene representation with multiple local neural fields that are optimized jointly to describe the appearance, density, semantics, and object instances in a scene (Figure 1). Nerflets constitute a *structured* and *irregular* representation—each is parameterized by a 3D center, a 3D XYZ rotation, and 3 (per-axis) radii in a 9-DOF coordinate frame. The influence of every nerflet is modulated by a radial basis function (RBF) which falls off with increasing distance from the nerflet center according to its orientation

and radii, ensuring that each nerflet contributes to a local part of the scene. Within that region of influence, each nerflet has a miniature MLP to estimate density and radiance. It also stores one semantic logit vector describing the category (e.g., “car”) of the nerflet, and one instance label indicating which real-world object it belongs to (e.g., “the third car”). In Figure 1, each ellipsoid is a single nerflet, and they are colored according to their semantics.

A scene can contain any number of nerflets, they may be placed anywhere in space, and they may overlap, which provides the flexibility to model complex, sparse 3D scenes efficiently. Since multiple nerflets can have the same instance label, they can combine to represent the density and radiance distributions of complex object instances. Conversely, since each nerflet has only one instance label, the nerflets provide a complete decomposition of the scene into real-world objects. Nerflets therefore provide a 3D panoptic decomposition of a scene that can be rendered and edited.

Synthesizing images using nerflets proceeds with density-based volume rendering just as in NeRF [34]. However, instead of evaluating one large MLP at each point sample along a ray, we evaluate the small MLPs of only the nerflets near a sample. We average the results, weighting by the influence each nerflet has over that sample. The rendering is fully-differentiable with respect to all continuous parameters of the nerflets. Fitting the nerflet representation is performed from a set of posed RGB images with a single training stage. After training, instance labels are assigned based on the scene structure, completing the representation.

Experiments with indoor and outdoor datasets confirm the main benefits of nerflets. We find that: 1) Parsimony encourages the optimizer to decompose the scene into nerflets with consistent projections into novel panoptic images (Section 4.1); 2) Semantic supervision can be beneficial to novel view synthesis (Section 4.2); 3) Structure encourages efficiency, compactness, and scalability (Section 3.4); and 4) the explicit decomposition of a scene improves human interpretability for easy interactive editing, including adding and removing objects (Section 4.1). These benefits enable state-of-the-art performance on the KITTI360 [26] novel semantic view synthesis benchmark, competitive performance on ScanNet 3D panoptic segmentation tasks with more limited supervision, and an interactive 3D editing tool that leverages the efficiency and 3D decomposition of nerflets.

The following summarizes our main contributions:

- We propose a novel 3D scene representation made of small, posed, local neural fields named *nerflets*.
- The pose, shape, panoptic, and appearance information of nerflets are all fit jointly in a single training stage, resulting in a comprehensive learned 3D decomposition from real RGB images of indoor or outdoor scenes.
- We test nerflets on 4 tasks- novel view synthesis,

panoptic view synthesis, 3D panoptic segmentation and reconstruction, and interactive editing.

- We achieve 1st place on the KITTI-360 semantic novel view synthesis leaderboard.

2. Related Work

Recently, the success of deep learning approaches for both computer vision and graphics tasks has enabled researchers to reconstruct and reason about 3D scenes under various settings. We review related work on segmentation and neural field based scene representations.

Semantic, Instance, and Panoptic Segmentation: There are many methods designed for semantic, instance, and/or panoptic [20] segmentation. The most popular approaches are fully-supervised and operate within a single input data modality. For example, 2D approaches [2, 6, 16, 29, 31, 44, 57, 60, 61] are usually based on CNN or transformer backbones and associate each pixel in an image with certain semantic or instance labels. We leverage a trained 2D panoptic model, Panoptic Deeplab [7], in our framework.

Similar frameworks have been proposed to solve 3D segmentation tasks for 3D point clouds [38–40, 46, 49], meshes [15, 18], voxel grids [14, 47], and octrees [43]. However, these methods typically require a large amount of annotated 3D data, which is expensive to obtain.

To avoid the need for 3D annotations, several multiview fusion approaches have explored aggregating 2D image semantic features onto a pointcloud or mesh using weighted averaging [1, 17, 25, 51], conditional random fields (CRFs) [24, 33], and bayesian fusions [32, 51, 56]. There have also been approaches like 2D3DNet [13] that combine both 2D mutiview fusion with a 3D model.

In contrast to these methods, ours builds a complete 3D representation including geometry, appearance, semantic, and instance information from only 2D inputs and without any input 3D substrate such as a mesh or a point cloud.

Scene Understanding with NeRF: NeRF [34] and subsequent work [21, 23, 37, 52, 55, 63] show the promise of neural radiance fields for tasks beyond novel view synthesis, including 3D reconstruction, semantic segmentation, and panoptic segmentation. For example, SemanticNeRF [63], and NeSF [52] are useful for semantic understanding, but do not consider object instances. DFF [21] leverages the power of large language models for semantic understanding, but similarly does not produce object instances. ObjectNeRF [55] and NSG [37] are useful for object editing, but do not produce a full panoptic decomposition or support efficient interactive editing. None of these methods produce a complete scene representation in the way that nerflets do.

Panoptic Neural Fields (PNF) [23] is very relevant to our paper as it supports both semantic scene understanding and

object-level scene editing. PNF first runs a 3D object detector and then a tracker to create an input set of object tracks. They then fit an individual MLP for each object track and another special “stuff” MLP for the remainder of the scene. This is a compelling and effective approach which supports moving objects, but it does not solve our target problem. It 1) requires expensive ground truth 3D supervision for the detector and tracker that are only available for some classes, and 2) has a fixed 3D scene decomposition that is provided by the input tracker results. This last point means that it can fail when the detector or tracker fails, even if a multi-view analysis-by-synthesis appearance loss, like in NeRF, would have been able to force a correct prediction by requiring all pixels to be described by some object instance. By comparison, nerflets require only 2D supervision, support any class for which 2D panoptic segmentations are available, and optimize most parameters jointly, improving efficiency and instance recall.

DM-NeRF [5] is highly related concurrent work. It learns an object decomposition of a scene, but does not provide the explicit structure, full panoptic decomposition, or easy interactive editing of nerflets. In particular, a large MLP decodes spatial positions to object identity vectors, and editing therefore requires careful consideration—an inverse query algorithm [5]. By contrast, nerflets can be edited directly as geometric primitives. We compare quantitatively to both PNF and DM-NeRF in Section 4.

Structured NeRF Representations: One of the key advantages of nerflets is their irregular structure, which has been investigated in other contexts. Many existing approaches exploit structure for efficiency [30, 35, 41, 48, 53], compactness [28], scalability [50, 59], human-interpretability [37], parsimony [11], or editability [23, 55]. For example, Kilo-NeRF [41] and DiVER [53] exploit regular grids of MLPs or features to improve the efficiency of NeRF novel view synthesis. MVP [30] builds an irregular primitive-based representation for real-time portrait rendering, but requires explicit scene geometry inputs to initialize the primitives and freezes their location after initialization. We take inspiration from these approaches, which achieve impressive performance through local structure, and apply their insights to panoptic segmentation and editing. Unlike these approaches, nerflets can conform to an object’s extent and then move as it is edited. In the future, more benefits of exploring irregular NeRF representations could include tracking a moving object or allowing for a consistent local coordinate frame for learning 3D priors.

3. Method

This section introduces our nerflet scene representation and our training and rendering method. As in NeRF [34], the input to our method is a set of posed 2D RGB im-

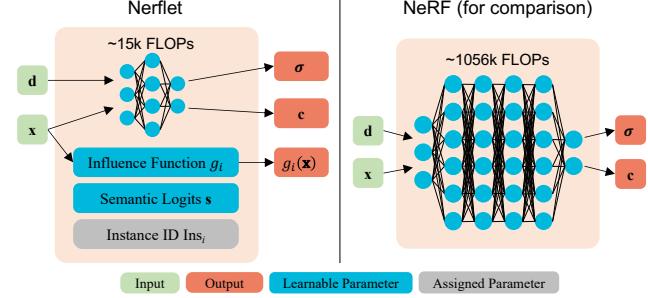


Figure 2. Information maintained by a nerflet and NeRF. Compared to NeRF, a nerflet focuses only on a small portion of the scene determined by its influence function g (Eq. 1), and thus it uses a miniature MLP to fit density σ and color c . Each nerflet also maintains a single semantic logit vector s_i and an assigned instance ID Ins_i . Together these parameters comprise a compact building block for our scene representation.

ages. We first run an off-the-shelf 2D panoptic segmentation model [7] to generate predicted 2D semantic and instance images, which we use as a target during the optimization. Next, we optimize our core nerflet representation (Section 3.1) to convergence on photometric, semantic, instance, and regularization losses applied to images rendered with volumetric ray-marching [34] (Section 3.2). Finally, we assign instance labels to the nerflets based on the learned decomposition (Section 3.3), at which point the representation is complete and ready for rendering or editing.

3.1. Scene Representation

The core novelty of our framework is the nerflet scene representation. Nerflets are a structured representation, where the output radiance and panoptic fields are defined by blending the values produced by N individual nerflets.

Nerflet definition: Each nerflet stores local geometry, appearance, semantic, and instance information. As shown in Figure 2, a nerflet i has 1) position and orientation parameters that define its influence function g_i over space, 2) its own tiny MLP f_i generating both density and radiance values, 3) a single semantic logit vector s_i storing its semantic information directly, and 4) an associated instance ID Ins_i . Compared to other semantics-aware NeRF methods [52, 62], we use a single logit vector to represent local semantic information instead of training an MLP to encode semantics. This aligns with our goal that a single nerflet should not span multiple classes or instances, and has the additional benefit of reducing the capacity burden of the MLP and providing a natural inductive bias towards 3D spatial consistency.

Pose and influence: Each nerflet has 9 pose parameters—a 3D center μ_i , 3 axis-aligned radii, and 3 rotation angles. We interpret these pose parameters in two ways. First, as a coordinate frame—each nerflet can be rasterized directly for

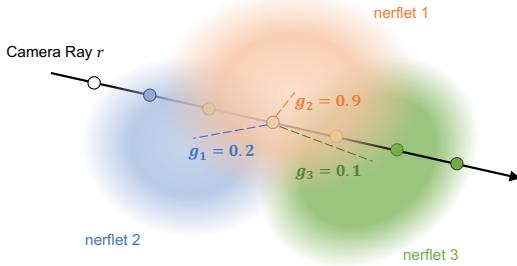


Figure 3. Sample and blend method illustration. Results from individual nerflets are mixed based on the influence values g_i determined by the distance-based weighting function. The mixing is smooth but most locations in space are dominated by a single nerflet, even when there is some overlap.

visualization by transforming an ellipsoid into the coordinate frame defined by the nerflet. This is useful for editing and understanding the scene structure (e.g., Figure 1). The second way, more critical for rendering, is via an influence function $g_i(\mathbf{x})$ defined by the same 9 pose parameters. g_i is an analytic radial basis function (RBF) based on scaled anisotropic multivariate Gaussians [11, 12]

$$g_i(\mathbf{x}) = \eta \exp\left(-\frac{1}{2\tau} (\mathbf{x} - \boldsymbol{\mu}_i)^T \Sigma_i^{-1} (\mathbf{x} - \boldsymbol{\mu}_i)\right). \quad (1)$$

$\boldsymbol{\mu}_i$ is the center of the basis function and Σ_i is a 6-DOF covariance matrix. The covariance matrix is determined by 3 Euler-angle rotation angles and 3 axis-aligned radii that are the reciprocal of variance along each principal axis. These 9 parameters provide a fast and compact way to evaluate a region of influence for each nerflet without evaluating any neural network. This property is crucial for our fast training and evaluation, which will be introduced later. η is a scaling hyper-parameter set to 5 for all experiments, and τ is a scheduled temperature hyper-parameter used to control the falloff hardness. τ is reduced after each training epoch to minimize overlap between nerflets gradually.

Rendering and blending: Given a scene represented by N nerflets, we can render 2D images with volume rendering, as in NeRF:

$$\hat{C}(\mathbf{r}) = \sum_{k=1}^K T_k \alpha_k \mathbf{c}_k, \quad (2)$$

$$\text{where } \alpha_k = 1 - \exp(-\sigma_k \delta_k), \quad (3)$$

$$T_k = \prod_{j=1}^{k-1} (1 - \alpha_j). \quad (4)$$

$\hat{C}(\mathbf{r})$ is the final color of ray \mathbf{r} , T_k is transmission at the k -th sample along the ray, α_k is the opacity of the sample, \mathbf{c}_k is the color at the sample, δ_k is the thickness of the current sample on the ray, and σ_k is the density at the sample. We denote k for the index of the sample along a ray, K total number of samples and reserve i for the index of the nerflet.

The biggest difference from NeRF is that instead of em-

ploying a single large MLP to produce \mathbf{c}_k and σ_k , we combine values produced by individual nerflets using their influence weights (Figure 3). We query individual nerflet MLPs f_i at the k -th input sample $(\text{pos}, \text{dir}) = (\mathbf{x}_k, \mathbf{d})$ along the ray, producing N values $\sigma_{k,i}$ and $\mathbf{c}_{k,i}$ for nerflets labeled $i \in [1, N]$. We then map the individual nerflet $\sigma_{k,i}$ values to $\alpha_{k,i}$ for rendering using δ_k and Equation 2. Finally, we take a weighted average of the N individual nerflet color and α values to produce \mathbf{c}_k and α_k :

$$\mathbf{c}_k = \sum_{i=1}^N \hat{g}_i(\mathbf{x}_k) \mathbf{c}_{k,i}, \quad (5)$$

$$\alpha_k = \sum_{i=1}^N \hat{g}_i(\mathbf{x}_k) \alpha_{k,i}, \quad (6)$$

$$\text{where } \hat{g}_i(\mathbf{x}_k) = \frac{g_i(\mathbf{x}_k)}{\sum_{j=1}^N g_j(\mathbf{x}_k) + \epsilon}. \quad (7)$$

ϵ is a factor allowing smooth decay to zero in empty space, with $\sum g_i(\mathbf{x}_k) \sim \epsilon$. After blending, the α_k and \mathbf{c}_k values are used directly for ray marching as in Eq. 2-4 to generate final pixel color values $\hat{C}(\mathbf{r})$, as in NeRF.

While in principle we should evaluate all nerflet MLPs in the scene in this step, as gaussian RBFs have infinite support, we do not do this. Typically, g_i is dominated by one or at most a handful of nerflets that are close to the sample. Therefore, we evaluate only the nearby MLPs, improving performance and scalability. This is implemented with our custom CUDA kernel, which will be introduced in Sec. 3.4. More distant nerflets are omitted from the average.

To generate semantic images, we average the per-nerflet semantic logit vector for each point sample in the same way described for color values $\mathbf{c}_{k,i}$ above.

To handle instances, we first compute a *nerflet influence activation* function $\mathbf{w} \in \mathbb{R}^n$ for each point sample:

$$\mathbf{w}(\mathbf{x}) = \text{SoftMax}([\sigma_1 g_1(\mathbf{x}), \dots, \sigma_N g_N(\mathbf{x})]). \quad (8)$$

σ_i is the density evaluation for \mathbf{x} on the i -th nerflet. This value intuitively represents how much influence each nerflet has over a given point sample, and can be accumulated by ray marching to generate a nerflet influence map $W(\mathbf{r})$ for each ray \mathbf{r} . $W(\mathbf{r})$ continuously captures which nerflet is dominant for each final pixel value, and is used by our influence loss described in Section 3.2. To assign a discrete instance label to a query position or ray, we take $\text{argmax}_{i \in [N]} w(\mathbf{r})$ or $\text{argmax}_{i \in [N]} W(r)$, respectively to get i , then output Ins_i , the instance of that nerflet.

Unbounded scenes: Nerflets support both indoor (bounded) and outdoor (unbounded) scenes. To handle unbounded scenes, we add a single MLP f_{far} to evaluate samples outside a large scene bounding box. We draw M additional samples for these points at the end of the ray, which we concatenate after our blended $\text{RGB}\alpha$ values and composite. We use the scheme proposed by Zhang *et al.* [58],

with an added semantics branch, though as the content is very distant and nearly directional, many other approaches would likely work well (e.g., an environment map).

3.2. Loss Function

During training, we jointly optimize all network parameters as well as the pose of the nerflets. In this way, each nerflet can be pushed by gradients to “move” across the scene, and focus on a specific portion of it. We expect a final decomposition mirroring the scene, with more nerflets on complex objects, and use multiple losses to that end.

The loss function is broken up into \mathcal{L}_{rgb} , semantic, instance, and regularization terms:

$$\mathcal{L} = \mathcal{L}_{\text{rgb}} + \mathcal{L}_{\text{sem}} + \mathcal{L}_{\text{ins}} + \mathcal{L}_{\text{reg}}. \quad (9)$$

\mathcal{L}_{rgb} : The RGB loss \mathcal{L}_{rgb} is the mean squared error between the synthesized color \hat{C} and the ground truth color C averaged over a batch of sampled rays, as in the original NeRF. The one change is that we weight this loss with a schedule parameter that is 0.0 at step 0. We gradually increase this value to 1.0 during training to prevent early overfitting to high frequency appearance information.

\mathcal{L}_{sem} : Our semantic loss \mathcal{L}_{sem} compares the volume-rendered semantic logits pixel with the Panoptic Deeplab prediction [7]. We use a per-pixel softmax-cross-entropy function for this loss.

\mathcal{L}_{ins} : The instance loss is defined as:

$$\mathcal{L}_{\text{ins}} = -\frac{1}{P} \sum_{(\mathbf{r}_1, \mathbf{r}_2)} \|W(\mathbf{r}_1) - W(\mathbf{r}_2)\|_1. \quad (10)$$

That is, we sample P ray pairs $(\mathbf{r}_1, \mathbf{r}_2)$ that are from the same class but different instances according to the instance segmentation model prediction, and enforce them to have different influence maps. While this approach is somewhat indirect, well separated nerflets can be easily assigned instance labels (Section 3.3), and it has the advantage of avoiding topology issues due to a variable number of instances in the scene while still achieving an analysis-by-synthesis loss targeting the instance decomposition. It is also compatible with the inconsistent instance ID labelings across different 2D panoptic image predictions. Ray pairs $(\mathbf{r}_1, \mathbf{r}_2)$ are chosen within in an $L \times L$ pixel window per batch for training efficiency.

\mathcal{L}_{reg} : Our regularization loss has several terms to make the structure of the nerflets better mirror the structure of the scene—

$$\mathcal{L}_{\text{reg}} = \mathcal{L}_{\text{density}} + \mathcal{L}_{\text{radii}} + \mathcal{L}_{\ell_1} + \mathcal{L}_{\text{box}}. \quad (11)$$

In addition to the intuitions described below, each of these is validated in a knock-out ablation study (Sec. 4.2) and tested on multiple datasets, to reduce the risk of overfitting to one setting.

First, to minimize unnecessary nerflet overlap within objects and reduce scene clutter, we penalize the L_2 norm

of the radii of the nerflets ($\mathcal{L}_{\text{radii}}$). To encourage sparsity where possible, we penalize the L_1 norm of the nerflet influence values at the sample locations (\mathcal{L}_{ℓ_1}). We also require nerflets to stay within their scene bounding box, penalizing:

$$\sum_{d \in \{x, y, z\}} \max(x_d - \text{box}_{\text{max}}, \text{box}_{\text{min}} - x_d, 0). \quad (12)$$

This reduces risk of “dead” nerflets, where a nerflet is far from the scene content, so it does not contribute to the loss, and therefore would receive no gradient.

Finally, we incorporate a “density” regularization loss $\mathcal{L}_{\text{density}}$, which substantially improves the decomposition quality:

$$\mathcal{L}_{\text{density}} = -\frac{1}{D} \sum_{i=1}^n \sum_{\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_i, \Sigma_i)} \sigma(\mathbf{x}). \quad (13)$$

$\mathcal{N}(\boldsymbol{\mu}_i, \Sigma_i)$ represents the underlying multivariate gaussian distribution for the i -th nerflet and D is the number of samples drawn. This term rewards a nerflet for creating density near its center location. As a result, nerflets end up centered inside the objects they reconstruct.

3.3. Instance Label Assignment

Given an optimized scene representation, we use a greedy merge algorithm to group the nerflets and associate them with actual object instances. We first pick an arbitrary 2D instance image, and render the associated nerflet influence map $W(\mathbf{r})$ for the image. We then assign the nerflets most responsible for rendering each 2D instance to a 3D instance based on it. We proceed to the next image, assigning nerflets to new or existing 3D instances as needed. Because the nerflets have been optimized to project to only a single 2D instance in the training images, this stage is not prone to failure unless the 2D panoptic images disagree strongly. See the supplemental for additional details.

3.4. Efficient Nerflet Evaluation

Top- k Evaluation: Instead of evaluating all nerflet MLPs in a scene as in Eq. 5–8, we use the g_i influence values to filter out distant and irrelevant nerflets in two ways. First, we truncate all nerflet influences below some trivial threshold to 0—there is no need to evaluate the MLP at all if it has limited support. In free space, often all MLPs can be ignored this way. Next, we implement a “top- k ” MLP evaluation CUDA kernel that is compatible with the autodiff training and inference framework. This kernel evaluates only the highest influence nerflet MLPs associated with each sample. We use $k = 16$ for both training and visualizations in this paper, although even more aggressive pruning is quite similar in image quality (e.g., a difference of only ~ 0.05 PSNR between top-16 and top-3) and provides a substantial reduction in compute. A top- k ablation study is available in the supplemental.

Method	Appearance PSNR	Semantics mIOU	Worst Case kFLOPs
PBNR [22] + PSPNet [60]	19.91	65.07	-
FVS [42] + PSPNet [60]	20.00	67.08	-
NeRF [34] + PSPNet [60]	21.18	53.01	~ 1056
Mip-NeRF [3] + PSPNet [60]	21.54	51.15	~ 1056
PNF [23]	21.91	<u>74.28</u>	~ 1256
Ours	<u>21.69</u>	75.07	~ 244

Table 1. Results on novel view color and semantic synthesis tasks on KITTI-360 [26]. Nerflets achieve similar color synthesis quality and better semantic synthesis quality compared to PNF [23] without any 3D supervision. We also have the best efficiency in terms of worst case kFLOPs.

Interactive Visualization and Scene Editing: We develop an interactive visualizer combining CUDA and OpenGL for nerflets that takes advantage of their structure, efficiency, rendering quality, and panoptic decomposition of the scene. Details are available in the supplemental. The key insight enabling efficient evaluation is that nerflets have a good sparsity pattern for acceleration – they are sparse with consistent local structure. We greatly reduce computation with the following two pass approach. In the first pass, we determine where in the volumetric sample grid nerflets have high enough influence to contribute to the final image. In the second pass, we evaluate small spatially adjacent subgrids for a particular nerflet MLP, which is generally high-influence for all samples in the subgrid due to the low spatial frequency of the RBF function. This amortizes the memory bandwidth of loading the MLP layers into shared memory. This approach is not as fast as InstantNGP [35], but is still interactive and has the advantage of mirroring the scene structure.

4. Experiments

In this section, we evaluate our method using 512 nerflets on multiple tasks with two challenging real-world datasets. Please see the supplemental for other hyperparameters.

KITTI-360: For KITTI-360 experiments, we use the novel view synthesis split, and compare to Panoptic Neural Fields (PNF) [23], a recent state of the art method for panoptic novel view synthesis (1st on the KITTI-360 leaderboard). To generate 2D panoptic predictions for outdoor scenes, we use a Panoptic DeepLab [7] model trained on COCO [27].

ScanNet: For ScanNet experiments, we evaluate on 8 scenes as in DM-NeRF [5] and compare to recent baselines—DM-NeRF [5], which synthesizes both semantics and instance information, and Semantic-NeRF [62] which synthesizes semantics only. To generate panoptic images for indoor scenes, we use PSPNet [60] and Mask R-CNN [16]. Please see the supplemental for important subtleties regarding how 2D supervision on ScanNet is achieved.

4.1. Results

Novel View Synthesis: We evaluate the performance of nerflets for novel view image synthesis on both KITTI-

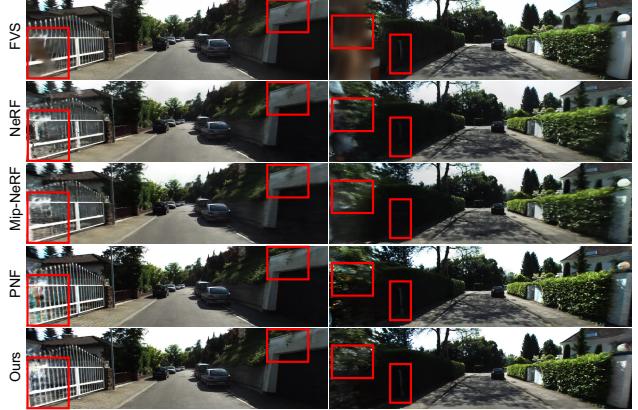


Figure 4. Novel view synthesis qualitative comparison. Nerflets outperform NeRF, Mip-NeRF, and FVS, and perform comparably to PNF with better performance in difficult areas (far left), possibly due to explicit spatial allocation of parameters.

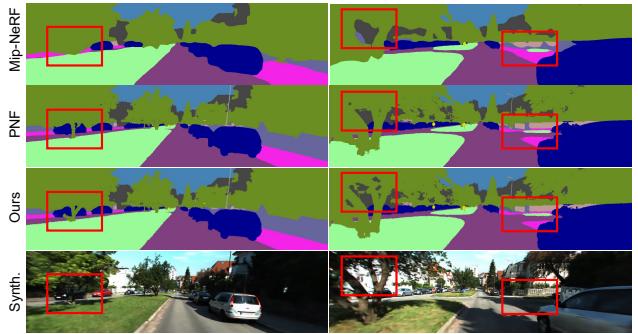


Figure 5. Novel view semantic synthesis qualitative comparison. Nerflets outperform other methods, particularly with respect to details and thin structures.

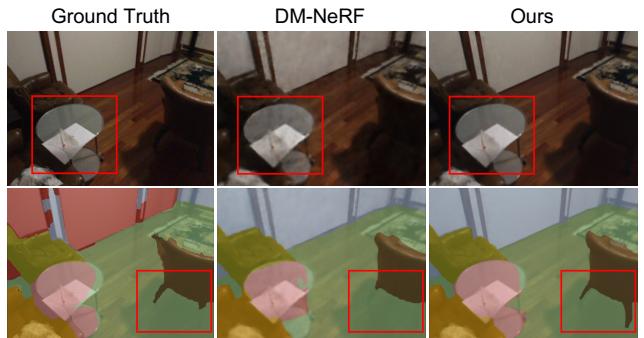


Figure 6. ScanNet qualitative result and comparison to DM-NeRF [5]. The comparison is on a ScanNet view synthesis example taken from the DM-NeRF paper. Our results improve in terms of both image and segmentation quality- notice the better image rendering for the glass table, and the better segmentation of the chair legs, which even exceeds the ground truth quality.

360 and ScanNet. As shown in Table 1, on KITTI-360, our method achieves better PSNR than all other 2D supervised methods on the leaderboard and is competitive with PNF [23], which utilizes 3D supervision. As shown in Fig-



Figure 7. KITTI-360 scene editing. We replace cars (top) or removing a sign instance (bottom).

Method	PSNR	Semantics mIoU	Instance mAP0.5
PSPNet [60] on GT Image	-	68.43	-
Mask R-CNN [16] on GT Image	-	-	23.53
PSPNet [60] on NeRF Im.	-	46.21	-
Mask R-CNN [16] on NeRF Im.	-	-	14.32
Semantic-NeRF [62]	28.43	71.34	-
DM-NeRF [5]	28.21	70.71	25.12
Ours	29.12	73.63	31.32

Table 2. ScanNet novel view synthesis quantitative results. Rendered color images and segmentation maps from nerflets have the best quality among all evaluated methods.

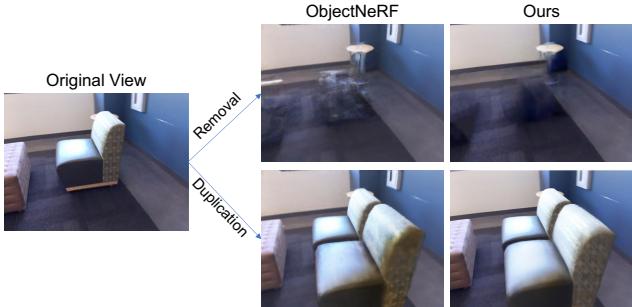


Figure 8. Scene editing comparison with ObjectNeRF on a pair of ScanNet images shown in the ObjectNeRF paper. Notice the improved handling of free space during removal and the more accurate texture during duplication, both attributable to the simple “copy-and-paste” nature of nerflets manipulation.



Figure 9. A 3D mesh extracted from a ScanNet RGB sequence with nerflets, colorized according to predicted panoptic labels. Ground truth RGBD mesh with human-annotated labels is shown on the left. Nerflets successfully reconstruct and label a chair instance missing from the ground truth mesh.

ure 4 our visual quality is approximately on par with PNF, and does particularly well in challenging areas. For complex indoor scenes in ScanNet (Table 2), we achieve the best performance for novel view synthesis under all settings (with or without instance supervision), including when

Supervision	Method	Semantics mIoU	Instance mAP0.5
Pointcloud	MinkowskiNet [8]	71.92	-
	3DMV [10]	49.22	-
	PointNet++ [40]	44.54	-
	Mask3D [45]	-	75.34
	3D-BoNet [54]	-	46.23
Images	Multiview Fusion [13]	55.23	-
	Ours	63.94	48.67

Table 3. Evaluation of nerflet panoptic performance on ScanNet 3D point cloud labeling task. Nerflets beat some less recent fully 3D-supervised methods with less supervision at both semantic and instance tasks, while also beating the similarly supervised multi-view fusion approach.

compared to DM-NeRF [5]. In particular, nerflets achieve better object details (Figure 6), likely due to their explicit allocation of parameters to individual object instances.

2D Panoptic Segmentation: Nerflets can render semantic and instance segmentations at novel views. We evaluate our 2D panoptic rendering performance quantitatively on both KITTI-360 (Table 1) and ScanNet (Table 2). On both datasets, nerflets outperform all baselines in terms of semantic mIoU, even compared to the 3D-supervised PNF [23]. On the ScanNet dataset, we also show that nerflets outperform PSPNet [60] and Mask R-CNN [16] in terms of both mIoU and instance mAP, although those methods were used to generate the 2D supervision for our method. This is an indication that nerflets are not only expressive enough to represent the input masks despite their much lower-dimensional semantic parameterization, but also that nerflets are effectively fusing 2D information from multiple views into a better more consistent 3D whole. We further explore this in the supplemental material. Qualitatively, nerflets achieve better, more detailed segmentations compared to baseline methods (Figure 5, Figure 6), particularly for thin structures.

Scene Editing: In Figure 7 and Figure 8, we use the instance labels on nerflets to select individual objects, and then manipulate the nerflet structure directly to edit scenes. No additional optimization is required, and editing can be done while rendering at interactive framerates (please see the video for a demonstration, the results here were rendered by the standard autodiff inference code for the paper). When compared to Object-NeRF on ScanNet (Figure 8), nerflets generate cleaner results with more detail, thanks to their explicit structure and alignment with object boundaries. Using nerflets, empty scene regions will not carry any density after deletion, as there is nothing there to evaluate. In Figure 7, we demonstrate additional edits on KITTI-360, with similar results. These visualizations help to confirm that indeed, nerflets learn a precise and useful 3D decomposition of the scene.

3D Panoptic Reconstruction: In Figure 9, we demonstrate the 3D capabilities of nerflets by extracting a panoptically labeled 3D mesh and comparing it to the ground truth. Surface extraction details are provided in the supplemental. We

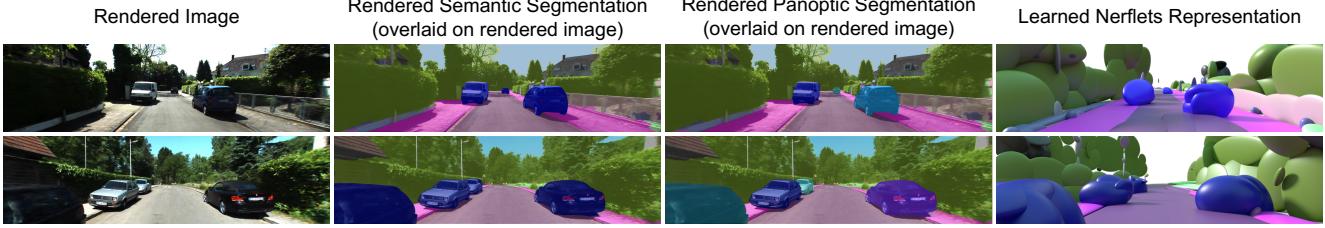


Figure 10. Visualization of nerflet outputs, trained on KITTI-360 images.

	PSNR	mIOU	mAP0.5
w/o $\mathcal{L}_{\text{density}}$	20.85	63.31	11.20
w/o $\mathcal{L}_{\text{radii}}$	27.23	72.43	26.32
w/o \mathcal{L}_{ℓ_1}	28.83	68.23	21.74
w/o \mathcal{L}_{box}	28.93	72.14	29.88
full model	29.12	73.63	31.32

Table 4. Ablation experiment on ScanNet for the effectiveness of our regularization terms— density loss $\mathcal{L}_{\text{density}}$, radii penalty $\mathcal{L}_{\text{radii}}$, influence sparsity loss \mathcal{L}_{ℓ_1} and scene box loss. \mathcal{L}_{box}

observe that the resulting mesh has both good reconstruction and panoptic quality compared to the ground truth. For example, nerflets even reveal a chair instance that is entirely absent in the ground truth mesh. We demonstrate this quantitatively by transferring nerflet representations to a set of ground truth 3D ScanNet meshes, comparing to existing 3D-labeling approaches in Table 3. We observe that nerflets outperform the similarly supervised multi-view fusion baseline, while adding instance capabilities. State of the art directly-3D supervised baselines are still more effective than nerflets when input geometry and a large 3D training corpus are available, but even so nerflets outperform some older 3D semantic and instance segmentation methods.

4.2. Analysis & Ablations

Scene Decomposition Quality: Our insight was to create an irregular representation that mirrors the structure of the scene. Do nerflets succeed at achieving this scene decomposition? In Fig. 10, we show RGB and panoptic images alongside the underlying nerflet decomposition that generated them. We see that indeed, the nerflets do not cross object boundaries, do join together to represent large or complex objects, and do cover the scene content.

Semantics Help Appearance: One key insight about our approach is that the semantic structure of the nerflets decomposition is beneficial even for lower level tasks, like novel view synthesis. We perform an experiment on the KITTI-360 validation set and observe that when training without a semantic or instance loss (i.e., photometric and regularization losses only), nerflets achieve a PSNR of 20.95. But when adding the semantics loss, PSNR *increases* to 22.43, because the nerflets end up more accurately positioned where the content of the scene is. This is also why we train with a higher semantic loss early in training, to encourage better nerflet positioning.

Ablation Study: Here we run a knock-out ablation study to validate the effectiveness of each of our regularization terms. Table 4 shows that all regularization terms contribute to final performance quantitatively. $\mathcal{L}_{\text{density}}$ is the most crucial term for learning a nice representation. It affects both image synthesis and segmentation performance, as it encourages nerflets to focus around actual scene content. $\mathcal{L}_{\text{radii}}$, \mathcal{L}_{ℓ_1} and \mathcal{L}_{box} all also improve performance, due to their effect of forcing a more well-separated and active decomposition of the scene where all nerflets contribute to the final result.

Performance: Nerflets have good performance due to their local structure. Our editor renders 320×240 top-1 editable volume images with 192 samples/pixel at 31 FPS with 4 A100 GPUs and 64 nerflets— 457 million sample evaluations per second.

5. Conclusion and Limitations

In this work, we present nerflets, a novel 3D scene representation which decomposes the scene into a set of local neural fields. Past work demonstrated structure is useful for parsimony in MLP-based shape representation [11], and we have found similar evidence extending that to scenes in this paper. Thanks to the locality of each nerflet, our model is compact, efficient, and multi-view-consistent. Results of experiments on two challenging real-world datasets KITTI-360 and ScanNet demonstrate state-of-the-art performance for panoptic novel view synthesis, as well as competitive novel view synthesis and support for downstream tasks such as scene editing and 3D segmentation.

Despite these positives, nerflets have several limitations. For example, we do not model dynamic content. Even though the representation is well-suited for handling rigid motions (as demonstrated in scene editing), that feature has not been investigated. Also, while individual nerflet radiance fields are capable of handling participating media, the overall representation may struggle to fit scenes where those effects cross semantic boundaries (e.g., foggy outdoor sequences). Finally, we currently assume a fixed number of nerflets for each scene, regardless of the scene complexity. However, it may be advantageous to prune, add, or otherwise dynamically adjust the number based on where they are needed (e.g. where the loss is highest). Investigating these novel features is interesting for future work.

Acknowledgements. We sincerely thank Yueyu Hu, Nilesh Kulkarni, Songyou Peng, Mikaela Angelina Uy, Boxin Wang, and Guandao Yang for useful discussion and help on experiments. We also thank Avneesh Sud for feedback on the manuscript.

Appendix

A. More Details

A.1. Model Architecture

For all nerflets MLPs f_i , we follow the NeRF architecture [34] but reduce the number of hidden layers from 8 to 4, and reduce the number of hidden dimensions from 256 to 32. We also removed the shortcut connection in the original network. All other architecture details are as in [34]. The background neural field uses NeRF++ [58] style encoding, and its MLP f_{far} is with 6 hidden layers and 128 hidden dimensions. One distinction is that we do perform coarse-to-fine sampling as in [34], but both coarse and fine samples are drawn from a single MLP, not two distinct ones.

A.2. Hyper-parameters

We use $N = 512$ nerflets for all experiments in the main paper. The scaling parameter η is set to 5 for all experiments. We initialize the nerflets temperature parameter τ to 1 and multiply τ by 0.9 across the epochs. The smooth decay factor ϵ is set to 10^{-7} for all experiments. We draw 64 samples for coarse level and 128 samples for fine level within the bounding box. For unbounded scenes, we draw 16 coarse samples and 16 fine samples from the background MLP. We increase the weight for \mathcal{L}_{rgb} from 0.0 to a maximum of 1.0 by the step of 0.2 across epochs to prevent early overfitting to high frequency information. Contrastive ray pairs are sampled within an 32×32 pixel window. The weight for regularization loss \mathcal{L}_{reg} is set to 0.1. All other losses are with weight 1.0.

A.3. Dataset Details

For training on each ScanNet scene, we uniformly sample 20% of the RGB frames for training and 10% of the RGB frames for evaluation— about 200 frames for training and 100 frames for evaluation. For both ScanNet and KITTI-360 scenes, we estimate the scene bounding box with camera extrinsics and normalize the coordinate inputs to $[-0.5, 0.5]$ for all experiments.

One important note about the ScanNet [9] experiments is that 2D ScanNet supervision indirectly comes from 3D. That is because the 2D ScanNet dataset was made by rendering the labeled mesh into images. We do not use this 2D ground truth directly, but PSPNet [60] is trained on it. Here, this is primarily a limitation of the evaluation rather than the

method— there are many 2D models that can predict reasonable semantics and instances on ScanNet images, but we want to be able to evaluate against the exact classes present in the 3D ground truth. This does not affect the comparison to other 2D supervised methods, as all receive their supervision from the same 2D model. By comparison KITTI-360 results are purely 2D only, but all quantitative evaluations must be done in image-space.

A.4. Paper Visualization Details

For ScanNet mesh extraction, we create point samples on a grid and evaluate their density, semantic and instance information from nerflets. We then estimate point normals using the 5 nearest neighbors and create a mesh with screened Poisson surface reconstruction [19]. The mesh triangles are colored according to the semantic and instance labels of their vertices. For the teaser and KITTI-360 visualizations of our learned nerflets representation, we visualize nerflets according to its influence function. We draw ellipsoids at influence value $e^{-\frac{1}{2}} \approx 0.607$.

A.5. Interactive Visualizer Details

Our interactive visualizer allows real-time previewing of nerflet editing results while adjusting the bounding boxes of objects in the scene. The visualizer draws the following components. First, a volume-rendered RGB or depth image at an interactive resolution of up to 320x240. This enables viewing the changes being made to the scene in real time. Second, the nerflets directly, by rendering an ellipsoid per nerflet at a configurable influence threshold. This enables seeing the scene decomposition produced by the nerflets. Third, a dynamic isosurface mesh extracted via marching cubes that updates as the scene is edited, giving some sense of where the nerflets are in relation to the content of the scene. Fourth, a set of boxing box manipulators, one per object instance, with draggable translation and rotation handles. These boxes are instantiated by taking the bounding box of the ellipsoid outline meshes for all nerflets associated with a single instance ID. A transformation matrix that varies per instance is stored and pushed to the nerflets on each edit.

Most of the editor is implemented in OpenGL, with the volume rendering implemented as a sequence of CUDA kernels that execute asynchronously and are transferred to the preview window when ready. In the main paper we report performance numbers for top-1 evaluation, which is often the right compromise for maximizing perceived quality in a given budget (e.g., pixel count can be more important), though interactive framerates with top-16 or top-3 evaluation are possible at somewhat lower resolutions.

	PSNR	mIOU
$n = 64$	26.34	53.23
$n = 128$	28.34	62.41
$n = 256$	28.81	69.97
$n = 512$	29.12	73.63
$n = 1024$	29.19	74.09

Table 5. Ablation experiment on ScanNet for different number of nerflets.

	PSNR	mIOU
$k = 32$	29.13	73.72
$k = 16$	29.12	73.63
$k = 3$	29.05	72.95
$k = 1$	28.35	70.73

Table 6. Ablation experiment on ScanNet for evaluating only nerflets with top- k influence weights during training and testing.

A.6. Instance Label Assignment

To assign instance labels for each nerflet, we render the nerflet influence map W_i for each view and compare with corresponding 2D semantic and instance segmentation maps to match each l_i^j (2D object instance or stuff with local id j in view i) to a set of nerflets $M(l_i^j)$. Here $M(\cdot)$ maps an instance ID to its set of associated nerflets. We then create a set of 3D instances $G = \{g_k\}$ according to the segmentation result of the first view – we create a 3D global instance for each detected 2D object instance and also each disjoint stuff labels in the semantic maps. For each new view i , we match l_i^j to the 3D instance g_k if $|M(l_i^j) \cap M(g_k)| / |M(l_i^j)| \geq \delta$, and then update $M(g_k)$ to $M(g_k) \cup M(l_i^j)$. If no match is found in $\{g_k\}$, we create a new 3D instance and insert it into G . Before inserting any new global instance g , we remove all nerflets that are already covered by the global set G from g . By using this first-come-first-serve greedy strategy we always guarantee no nerflet is associated with 2 different global instances. After this step, each nerflet is associated with a global instance ID, and our representation can be used to reason at an instance level effectively.

B. More Results

The following experiments are done on the subset of ScanNet from [5].

B.1. Number of Nerflets

We perform ablation study on the number of nerflets on ScanNet. The results are in Tab. 5. We find that increasing the number of nerflets could improve the performance on both photometric and semantic metrics. However, the benefit saturates when adding more nerflets than 512. To balance performance and efficiency, we use 512 nerflets in all experiments in the paper.

B.2. Effect of Top- k Evaluation

We perform ablation study on the performance impact of k when we only evaluate nerflets with top- k influence weights for each point sample. The results are in Tab. 6. We find that evaluating 32, 16 or 3 nerflets have little influence on the model performance, since each nerflet is only contributing locally. However we see a moderate performance drop when only evaluating one nerflets with the highest influence weight. We choose to evaluate $k = 16$ nerflets for all experiments in the paper to balance the computational cost and performance.

B.3. Inactive Nerflets

One known problem [12] with training using RBFs that have learned extent is that when an RBF gets too small or too far from the scene, it does not contribute to the construction results. The radii loss $\mathcal{L}_{\text{radii}}$ and box loss \mathcal{L}_{box} are proposed to alleviate this issue. To estimate the actual number of inactive nerflets, we utilize nerflet influence map W and count nerflets that do not appear on any of these maps in any views. In KITTI-360 experiments, we estimate to have 10.6 inactive nerflets on average per scene, making up $\sim 2.07\%$ of all available nerflets. In ScanNet experiments in the main paper, we estimate to have 30.6 inactive nerflets on average per scene, making up $\sim 5.98\%$ of all available nerflets.

B.4. Robustness against Input 2D Segmentation

In Figure 11, we visualize more examples on ScanNet comparing our panoptic predictions with reference annotations from the dataset. It can be seen that our representation learned from 2D supervision contains rich information and can produce more accurate segmentation results than reference maps in some cases. Our method produces clearer boundaries, fewer holes and discovers missing objects in the reference results, thanks to its ability to fuse segmentations from multiple views with a 3D sparsity prior from the structure of the representation.

References

- [1] Iro Armeni, Zhi-Yang He, JunYoung Gwak, Amir R Zamir, Martin Fischer, Jitendra Malik, and Silvio Savarese. 3d scene graph: A structure for unified semantics, 3d space, and camera. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5664–5673, 2019. 2
- [2] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017. 2
- [3] 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 *Proceedings of the IEEE/CVF Inter-*

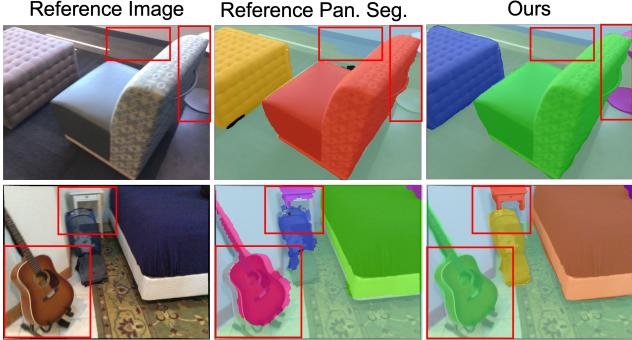


Figure 11. Comparison of ScanNet reference panoptic segmentation maps and our panoptic segmentation predictions overlaid on reference images.

- national Conference on Computer Vision*, pages 5855–5864, 2021. 1, 6
- [4] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. *CVPR*, 2022. 1
 - [5] Wang Bing, Lu Chen, and Bo Yang. Dm-nerf: 3d scene geometry decomposition and manipulation from 2d images. *arXiv preprint arXiv:2208.07227*, 2022. 1, 3, 6, 7, 10
 - [6] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017. 2
 - [7] Bowen Cheng, Maxwell D Collins, Yukun Zhu, Ting Liu, Thomas S Huang, Hartwig Adam, and Liang-Chieh Chen. Panoptic-deeplab: A simple, strong, and fast baseline for bottom-up panoptic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12475–12485, 2020. 2, 3, 5, 6
 - [8] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019. 7
 - [9] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017. 9
 - [10] Angela Dai and Matthias Nießner. 3dmv: Joint 3d-multi-view prediction for 3d semantic scene segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 7
 - [11] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. Local deep implicit functions for 3d shape. In *CVPR*, pages 4857–4866, 2020. 3, 4, 8
 - [12] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. Learning shape templates with structured implicit functions. In *ICCV*, pages 7154–7164, 2019. 4, 10

- [13] Kyle Genova, Xiaoqi Yin, Abhijit Kundu, Caroline Pantofaru, Forrester Cole, Avneesh Sud, Brian Brewington, Brian Shucker, and Thomas Funkhouser. Learning 3d semantic segmentation with only 2d image supervision. In *2021 International Conference on 3D Vision (3DV)*, pages 361–372. IEEE, 2021. 2, 7
- [14] Ben Graham. Sparse 3d convolutional neural networks. *arXiv preprint arXiv:1505.02890*, 2015. 2
- [15] Rana Hanocka, Amir Hertz, Noa Fish, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. Meshcnn: a network with an edge. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019. 2
- [16] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 2, 6, 7
- [17] Alexander Hermans, Georgios Floros, and Bastian Leibe. Dense 3d semantic mapping of indoor scenes from rgbd images. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2631–2638. IEEE, 2014. 2
- [18] Jingwei Huang, Haotian Zhang, Li Yi, Thomas Funkhouser, Matthias Nießner, and Leonidas J Guibas. Texturenet: Consistent local parametrizations for learning from high-resolution signals on meshes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4440–4449, 2019. 2
- [19] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):1–13, 2013. 9
- [20] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9404–9413, 2019. 2
- [21] Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann. Decomposing nerf for editing via feature field distillation. *NIPS* 2022, 2022. 2
- [22] Georgios Kopanas, Julien Philip, Thomas Leimkühler, and George Drettakis. Point-based neural rendering with per-view optimization. *Computer Graphics Forum (Proceedings of the Eurographics Symposium on Rendering)*, 40(4):29–43, 2021. 6
- [23] Abhijit Kundu, Kyle Genova, Xiaoqi Yin, Alireza Fathi, Caroline Pantofaru, Leonidas J Guibas, Andrea Tagliasacchi, Frank Dellaert, and Thomas Funkhouser. Panoptic neural fields: A semantic object-aware neural scene representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12871–12881, 2022. 1, 2, 3, 6, 7
- [24] Abhijit Kundu, Yin Li, Frank Dellaert, Fuxin Li, and James M Rehg. Joint semantic segmentation and 3d reconstruction from monocular video. In *European Conference on Computer Vision*, pages 703–718. Springer, 2014. 2
- [25] Abhijit Kundu, Xiaoqi Yin, Alireza Fathi, David Ross, Brian Brewington, Thomas Funkhouser, and Caroline Pantofaru. Virtual multi-view fusion for 3d semantic segmentation. In *European Conference on Computer Vision*, pages 518–535. Springer, 2020. 2

- [26] Yiyi Liao, Jun Xie, and Andreas Geiger. KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *arXiv preprint arXiv:2109.13410*, 2021. [2](#), [6](#)
- [27] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. [6](#)
- [28] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33:15651–15663, 2020. [3](#)
- [29] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. [2](#)
- [30] Stephen Lombardi, Tomas Simon, Gabriel Schwartz, Michael Zollhoefer, Yaser Sheikh, and Jason Saragih. Mixture of volumetric primitives for efficient neural rendering. *ACM Transactions on Graphics (TOG)*, 40(4):1–13, 2021. [3](#)
- [31] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. [2](#)
- [32] Lingni Ma, Jörg Stückler, Christian Kerl, and Daniel Cremers. Multi-view deep learning for consistent semantic mapping with rgb-d cameras. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 598–605. IEEE, 2017. [2](#)
- [33] John McCormac, Ankur Handa, Andrew Davison, and Stefan Leutenegger. Semanticfusion: Dense 3d semantic mapping with convolutional neural networks. In *2017 IEEE International Conference on Robotics and automation (ICRA)*, pages 4628–4635. IEEE, 2017. [2](#)
- [34] 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. *Communications of the ACM*, 65(1):99–106, 2021. [1](#), [2](#), [3](#), [6](#), [9](#)
- [35] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022. [1](#), [3](#), [6](#)
- [36] Gaku Narita, Takashi Seno, Tomoya Ishikawa, and Yohsuke Kaji. Panopticfusion: Online volumetric semantic mapping at the level of stuff and things. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4205–4212. IEEE, 2019. [1](#)
- [37] Julian Ost, Fahim Mannan, Nils Thuerey, Julian Knodt, and Felix Heide. Neural scene graphs for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2856–2865, 2021. [2](#), [3](#)
- [38] Quang-Hieu Pham, Thanh Nguyen, Binh-Son Hua, Gemma Roig, and Sai-Kit Yeung. Jsis3d: joint semantic-instance segmentation of 3d point clouds with multi-task pointwise networks and multi-value conditional random fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8827–8836, 2019. [2](#)
- [39] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. [2](#)
- [40] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017. [2](#), [7](#)
- [41] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14335–14345, 2021. [1](#), [3](#)
- [42] Gernot Riegler and Vladlen Koltun. Free view synthesis. In *ECCV*, 2020. [6](#)
- [43] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3577–3586, 2017. [2](#)
- [44] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. [2](#)
- [45] Jonas Schult, Francis Engelmann, Alexander Hermans, Or Litany, Siyu Tang, and Bastian Leibe. Mask3d for 3d semantic instance segmentation. *arXiv preprint arXiv:2210.03105*, 2022. [7](#)
- [46] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. *arXiv preprint arXiv:1912.13192*, 2019. [2](#)
- [47] Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1746–1754, 2017. [2](#)
- [48] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *CVPR*, 2022. [3](#)
- [49] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6411–6420, 2019. [2](#)
- [50] Haithem Turki, Deva Ramanan, and Mahadev Satyanarayanan. Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12922–12931, 2022. [3](#)
- [51] Vibhav Vineet, Ondrej Miksik, Morten Lidegaard, Matthias Nießner, Stuart Golodetz, Victor A Prisacariu, Olaf Kähler, David W Murray, Shahram Izadi, Patrick Pérez, et al. Incre-

- mental dense semantic stereo fusion for large-scale semantic scene reconstruction. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 75–82. IEEE, 2015. 2
- [52] Suhani Vora, Noha Radwan, Klaus Greff, Henning Meyer, Kyle Genova, Mehdi SM Sajjadi, Etienne Pot, Andrea Tagliasacchi, and Daniel Duckworth. NeSF: Neural semantic fields for generalizable semantic segmentation of 3d scenes. *arXiv preprint arXiv:2111.13260*, 2021. 1, 2, 3
- [53] Liwen Wu, Jae Yong Lee, Anand Bhattad, Yuxiong Wang, and David Forsyth. Diver: Real-time and accurate neural radiance fields with deterministic integration for volume rendering, 2021. 3
- [54] Bo Yang, Jianan Wang, Ronald Clark, Qingyong Hu, Sen Wang, Andrew Markham, and Niki Trigoni. Learning object bounding boxes for 3d instance segmentation on point clouds. In *Advances in Neural Information Processing Systems*, pages 6737–6746, 2019. 7
- [55] 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 *International Conference on Computer Vision (ICCV)*, October 2021. 2, 3
- [56] Cheng Zhang, Zhi Liu, Guangwen Liu, and Dandan Huang. Large-scale 3d semantic mapping using monocular vision. In *2019 IEEE 4th International Conference on Image, Vision and Computing (ICIVC)*, pages 71–76. IEEE, 2019. 2
- [57] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Haibin Lin, Zhi Zhang, Yue Sun, Tong He, Jonas Mueller, R Manmatha, et al. Resnest: Split-attention networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2736–2746, 2022. 2
- [58] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020. 4, 9
- [59] Xiaoshuai Zhang, Sai Bi, Kalyan Sunkavalli, Hao Su, and Zexiang Xu. Nerfusion: Fusing radiance fields for large-scale scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5449–5458, 2022. 3
- [60] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017. 2, 6, 7, 9
- [61] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6881–6890, 2021. 2
- [62] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew Davison. In-Place Scene Labelling and Understanding with Implicit Scene Representation. In *ICCV*, 2021. 1, 3, 6, 7
- [63] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew Davison. In-place scene labelling and understanding with implicit scene representation. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.