

# View-Consistent Hierarchical 3D Segmentation Using Ultrametric Feature Fields

Haodi He<sup>✉</sup>, Colton Stearns<sup>✉</sup>, Adam W. Harley<sup>✉</sup>, and Leonidas J. Guibas<sup>✉</sup>

Stanford University

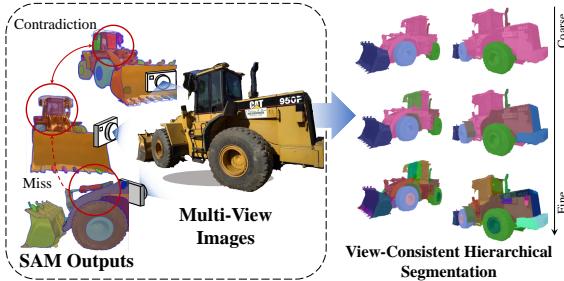
**Abstract.** Large-scale vision foundation models such as Segment Anything (SAM) demonstrate impressive performance in zero-shot image segmentation at multiple levels of granularity. However, these zero-shot predictions are rarely 3D-consistent. As the camera viewpoint changes in a scene, so do the segmentation predictions, as well as the characterizations of “coarse” or “fine” granularity. In this work, we address the challenging task of lifting multi-granular and view-inconsistent image segmentations into a hierarchical and 3D-consistent representation. We learn a novel feature field within a Neural Radiance Field (NeRF) representing a 3D scene, whose segmentation structure can be revealed at different scales by simply using different thresholds on feature distance. Our key idea is to learn an ultrametric feature space, which unlike a Euclidean space, exhibits transitivity in distance-based grouping, naturally leading to a hierarchical clustering. Put together, our method takes view-inconsistent multi-granularity 2D segmentations as input and produces a hierarchy of 3D-consistent segmentations as output. We evaluate our method and several baselines on synthetic datasets with multi-view images and multi-granular segmentation, showcasing improved accuracy and viewpoint-consistency. We additionally provide qualitative examples of our model’s 3D hierarchical segmentations in real world scenes.<sup>1</sup>

## 1 Introduction

Different applications often need different semantic understandings of a scene. This fact necessitates that segmentation methods offer a diverse set of predictions that span different modalities, showcase multiple levels of granularity, and offer hierarchical relationships. With the advent of the “Segment Anything Model” (SAM) [24], reliable multi-granular *single-view* segmentation might be described as accomplished. Yet, in a multi-view or moving-camera system, multi-granular segmentation of each image can produce an overwhelming total number of segments, many of which disagree with each other and most of which are not useful for the downstream application of interest. In this work, we attempt to distill these thousands of 2D segmentation options (which may be conflicting) into an organized 3D segmentation which is *view-consistent* and *hierarchical*.

---

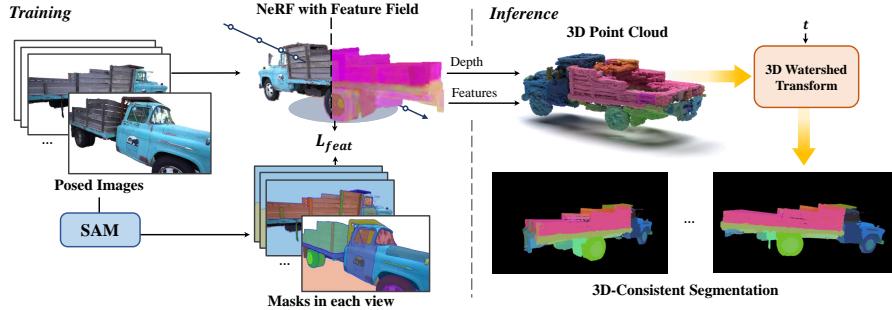
<sup>1</sup> The code and dataset are available at: [https://github.com/hardyho/ultrametric\\_feature\\_fields](https://github.com/hardyho/ultrametric_feature_fields)



**Fig. 1:** Our method takes as input multi-view posed images, paired with segmentation masks from the recent “Segment Anything Model” (SAM), and merges these into a coherent 3D representation where segmentation is view-consistent and hierarchical.

As demonstrated in many recent works, *view-consistency* can be achieved by distilling the segmentation information into a 3D implicit field, such as a Neural Radiance Field (NeRF) [33]. The first works in this area simply optimized a semantic labelling branch alongside the color branch of the NeRF [27, 32, 63]. This delivers view-consistent semantic segmentation, but does not generalize to unseen categories. More recent methods have proposed to distill generic image features into the implicit field [21, 26], allowing these features to be queried in 2D or 3D for segmentation or other tasks. However, these methods often struggle to capture precise segmentation boundaries, perhaps because language-aligned features are region-based and not pixel-based. Furthermore, most of these works do not directly account for multiple levels of granularity, and those that do [21] must re-render the feature field for every scale of interest. Nonetheless, the expressiveness and wide adoption of NeRFs makes them a natural choice as underlying 3D representation, and we adopt this same choice here.

Unlike prior work, we explicitly aim for our scene segmentation to be *hierarchical*. This means that the scene segmentation has a tree structure, where the root group is the full scene, and any group can be recursively divided into smaller groups, all the way down to the point level. A group is defined as a spatially connected neighborhood where all pairwise feature distances are within some threshold. At first glance, this is a familiar contrastive learning problem: within-segment feature pairs should have small distances, and cross-segment feature pairs should have large distances, so that thresholding yields groups that follow segmentation boundaries. However, we demonstrate this typical setup is not sufficient to create a consistent hierarchy, and we find it is crucial to use *ultrametric* distances, rather than standard Euclidean distances, in the contrastive loss. In an ultrametric space, for any three points  $x$ ,  $y$ , and  $z$ , distances satisfy a condition stronger than the standard triangle inequality  $d(x, z) \leq d(x, y) + d(y, z)$ , namely that  $d(x, z) \leq \max\{d(x, y), d(y, z)\}$ . Ultrametric spaces are ideally suited for hierarchical clustering because distance-based groupings are transitive: if  $d(x, y) \leq \epsilon$  and  $d(y, z) \leq \epsilon$ , then it follows that  $d(x, z) \leq \epsilon$ . In other words, if we group  $x$  and  $y$  together and  $y$  and  $z$  together, then we automatically also



**Fig. 2: Method Overview:** We train a NeRF with an ultrametric feature field using images and view-inconsistent segmentation masks from SAM [24]. After training, we use the depth estimation and feature maps from training views to construct a 3D point cloud. At inference, for a specified threshold  $t$  representing the granularity level, we apply a 3D watershed transform to segment the 3D point cloud. Then, we can query the point clouds in novel views and obtain view-consistent segmentation results.

group  $x$  and  $z$  together. Thus, as  $\epsilon$  grows, smaller clusters are naturally merged into larger clusters, automatically giving rise to a hierarchy.

We optimize our 3D ultrametric feature fields by rendering them to 2D, and using SAM as a noisy supervision signal for grouping. After optimization, we produce view-consistent hierarchical segmentations at arbitrary levels of granularity, by simply specifying a threshold  $\epsilon$ , and running a Watershed transform to retrieve all groups that exist under this threshold.

We evaluate our approach on models from the PartNet dataset, showing that our method recovers a view-consistent hierarchy of segmentations that captures the natural part decomposition of a 3D object. Furthermore, we introduce a synthetic dataset with hierarchical segmentation annotations based on the NeRF Blender Dataset. Unlike PartNet, our proposed dataset offers hierarchical decomposition of more complex scenes. In all evaluations, we compare our method to a set of competitive baselines, measuring IoU accuracy and 3D consistency, and we demonstrate that our method outperforms existing open-vocabulary 3D segmentation methods, such as DFF [26], LeRF [21], and SAM-3D [7]. Additionally, we introduce a metric for measuring the quality of a hierarchy, and demonstrate that the segmentations in prior work lack hierarchical structure, while our output is hierarchical by construction.

In sum, our key contribution is a novel formulation for 3D scene segmentation: *ultrametric feature fields*. Using this formulation, we are able to distill view-inconsistent 2D masks into a 3D representation which is not only view-consistent but also hierarchical, allowing arbitrary-granularity segmentation at test time. We also contribute a new synthetic dataset, and propose new evaluation metrics, to quantify our progress and facilitate future work in this direction. Finally we provide qualitative examples of our model’s 3D hierarchical segmentations in real world scenes.

## 2 Related Work

**Neural Radiance and Feature Fields** Neural Radiance Fields (NeRFs) [33] are a popular representation for novel view synthesis and 3D reconstruction, and in the last few years there has been an explosion of research surrounding NeRFs. To name a few research directions, NeRF has been extended to improve rendering quality [3, 15, 43], accelerate training and inference [9, 10, 17, 30, 37], improve geometry [40, 52, 57, 61], and learn with fewer viewpoints [39, 42, 50]. In this work we are most interested in related research that integrates *segmentation* into NeRF [7, 21, 26, 27, 31, 32, 35, 45, 55, 59, 63].

One line of research aimed to learn a semantic or instance branch alongside the NeRF’s color and density, thus yielding view-consistent segmentation [27, 32, 45, 51, 63]. Semantic-NeRF [63] was a seminal work that learned a semantic field to propagate 2D segmentations into new views. Instance-NeRF [32] improved Semantic-NeRF by handling panoptic segmentation from Mask2Former [12]. Nevertheless, these methods are restricted to a closed vocabulary of instance categories, and additionally do not attempt a hierarchical understanding of the segmented instances.

Another recent direction has been to learn a generic volumetric feature field alongside NeRF. DFF [26], N3F [49], and LeRF [21] distilled 2D image features generated by off-the-shelf feature extractors such as CLIP [41], DINO [5], and LSeg [28] into feature fields that enable 3D segmentation and editing in NeRF. However, these methods often struggle to recover precise segmentation boundaries, and also do not establish a hierarchical structure on the features. In contrast, we learn a feature field that distills a hierarchy of segmentations from noisy multi-view SAM predictions.

Finally, other works [6, 7, 59] used NeRF or 3D Gaussians [20] to propagate a single SAM query into novel views and establish view-consistency. However, these methods handle one segmentation at a time, while our approach trains a feature field to jointly aggregate and reconcile *hundreds* of noisy 2D masks.

**Hierarchical Segmentation** Hierarchical segmentation, a specialized domain within image segmentation, partitions an image into regions that exhibit a hierarchical tree structure, (i.e., each segmentation can be recursively divided into smaller segmentations). In the pre-deep learning era, researchers utilized non-parametric approaches to generate contours and enable hierarchical segmentation [1, 2, 4, 38, 46]. Ultrametric distance and the watershed transform were employed to identify hierarchical clusters based on RGB values [1, 4, 38], and subsequent advancements by Yarkony et al. [58] and Xu et al. [56] improved the efficiency and flexibility of ultrametric hierarchical segmentation methods. More recently, Zhao et al. [62] and Li et al. [29] shifted their focus from hierarchical segmentation *within* an image toward estimating the hierarchy of segmentation *classes*. This class hierarchy has also been of particular interest in the field of human parsing, where segmentation is performed according to the hierarchical structure of the human body [14, 16, 53, 54]. In the 3D domain, Mo et al. [36] in-

troduced PartNet, a large scale 3D mesh dataset annotated with hierarchical segmentation, and researchers have subsequently explored hierarchical structure in 3D shapes, including multi-granularity segmentation on point clouds [47, 48, 65].

Most relevant to us, the recent Segment Anything Model [24] emphasized *multi-granular* segmentation in its open-vocabulary and zero-shot segmentation setting. Given its impressive performance, we use off-the-shelf SAM segmentations as supervision for our method. Furthermore, while concurrent works [23, 60] also attempt to distill SAM masks into NeRF, our ultrametric feature field constitutes a fundamentally different approach to the problem.

### 3 Preliminaries and Notation

In this section we describe the core building blocks of our approach: (1) implicit feature fields, (2) hierarchical segmentation via watershed transform and ultrametrics, and (3) an off-the-shelf image segmentation model.

**NeRF and Feature Fields** A Neural Radiance Field (NeRF) [33] is a volumetric representation that outputs a density  $\sigma$  and color  $\mathbf{c}$  given a 3D coordinate  $\mathbf{x} = (x, y, z)$  and 2D viewing direction  $\mathbf{d}$ . Given a pixel’s camera ray  $\mathbf{r}$ , the NeRF samples  $N$  points along the ray  $\mathbf{x}_1, \dots, \mathbf{x}_N$  at corresponding intervals  $\delta_1, \dots, \delta_N$  and performs approximate volume rendering to estimate the pixel’s color:  $\mathbf{C}(\mathbf{r}) = \sum_{k=1}^N T_k (1 - e^{(-\sigma_k \delta_k)}) \mathbf{c}_k$ , with  $T_k = e^{-\sum_{i=1}^k \sigma_i \delta_i}$ .

Recently, Distilled Feature Fields (dff) [26] proposed to learn a volumetric feature fields within a NeRF. Given 3D coordinate  $\mathbf{x}$ , DFF outputs a feature  $\mathbf{f}$  in addition to the original density  $\sigma$  and color  $\mathbf{c}$ . Using the same volume rendering equation, DFF renders features in addition to colors. We follow DFF and learn a NeRF with an accompanying feature field.

**Watershed Transform and Ultrametrics** The Watershed transform [4, 38] is a traditional hierarchical segmentation method. The method interprets edge energies as a heightmap, and initiates a flooding process, wherein energy basins beneath a given threshold are merged into regions, and higher thresholds cause regions to merge. Since the “water level” is uniform across the whole space, this yields a hierarchical segmentation.

Representing an image or scene as a graph, denoted as  $\mathcal{G} = (V, E)$ , where  $V$  includes all of the pixels/points and  $E$  connects points which are adjacent, the minimum water level that merges two points can be expressed as:

$$d(v_i, v_j) = \min_{p \in P} \max_{e \in p} |e|, \quad (1)$$

where  $P$  denotes all paths that connect  $v_i$  and  $v_j$  in the graph, and  $e$  is an edge on the path  $p$ . Computing the distance between  $v_i$  and  $v_j$  means first finding the shortest path between the nodes, where path length is determined by the maximum edge along the path, and then reporting that crucial edge length. This



**Fig. 3: Ultrametric Segmentation:** *Left:* We overlay a simple graph on the image, showing edge lengths corresponding to feature distances between points. *Right:* The hierarchical segmentation derived from the graph on the left. The numbers on the tree indicate the ultrametric distance between nodes on the two branches.

is sometimes called the minimax path problem. This distance is an *ultrametric* distance [18, 34], which satisfies a triangle inequality of the form

$$d(x, y) \leq \max\{d(x, z), d(y, z)\}. \quad (2)$$

Fig. 3 provides an illustration of paths and distances in a simple scene graph.

In our setup, we use Eq. (2) to define an ultrametric contrastive loss for our scene features. We use feature distances as edge lengths in Eq. (1), enabling us to obtain hierarchical segmentations at test time via the Watershed transform.

**Segment Anything Model (SAM)** SAM [24] is a state-of-the-art vision foundation model for image segmentation. Given a query in the form of points, a mask, a box, or a language prompt, SAM predicts a segmentation that best reflects the prompt. SAM generates segmentations at three levels of granularity: instance, part, and subpart.

While SAM produces excellent results on *single images*, it is nontrivial to lift its predictions into 3D. This is because SAM’s predictions are not consistent across viewpoints. For example, a pen may be segmented into multiple parts in a close-up view, then segmented as a single instance in a wider view, and then be completely missed when viewed from farther away. Furthermore, it is unclear how to best lift outputs from a “queryable” design into a well-organized 3D representation: given an abundance of queries, SAM will generate an abundance of masks, which overlap with one another unpredictably, and have no straightforward unification.

In our setup, we use SAM to provide a noisy signal of viable segmentations within each viewpoint, and rely on multi-view feature field optimization, using ultrametrics, to distill this knowledge into a 3D scene segmentation which is view-consistent and hierarchical.

## 4 Method

The previous section described the core components of our approach. In this section we describe how these pieces fit together, and explain the training and inference pipelines.



**Fig. 4: Hierarchical Segmentation:** Our method can hierarchically segment real world scenes at various levels of granularity.

#### 4.1 Learning an Ultrametric Feature Field

**Problem Formulation** We take as input a set of multi-view images, along with their camera parameters. We run SAM on every image, typically yielding 50-150 masks per image. Our goal is to learn an implicit feature field that encodes a hierarchical understanding of these masks.

**Contrastive Learning** The SAM masks are inconsistent across views, but carry a great deal of information within each view. We use contrastive learning [11, 22, 41] to distill this information into a 3D feature field.

We sample a pair of pixels in an image, and define it as a positive pair if both lie within a same mask, and a negative pair otherwise. We supervise the features of positive pairs to be more similar than those of negative pairs. Concretely, we follow Chen et al. [11] and minimize a binary cross entropy loss on distances between positive pairs and negative pairs. Given a positive pair  $s_p = \{v_{p1}, v_{p2}\}$  and a negative pair  $s_n = \{v_{n1}, v_{n2}\}$ , our contrastive loss is

$$\ell(s_p, s_n) = -\log\left(\frac{e^{d(v_{p1}, v_{p2})/\tau}}{e^{d(v_{p1}, v_{p2})/\tau} + e^{d(v_{n1}, v_{n2})/\tau}}\right) + \log\left(\frac{e^{d(v_{n1}, v_{n2})/\tau}}{e^{d(v_{p1}, v_{p2})/\tau} + e^{d(v_{n1}, v_{n2})/\tau}}\right), \quad (3)$$

where  $d$  is a distance metric, and  $\tau$  is the temperature.

Minimizing the loss in Eq. (3) across many pairs  $s_p$  and  $s_n$  yields a feature space where segmentations can be recovered by querying a point and finding the neighborhood where pairwise feature distances are all below some threshold. As discussed in earlier sections, whether or not the resulting segmentation will be hierarchical depends on whether the distance metric  $d$  is an ultrametric or a standard Euclidean metric. Our main goal is to reduce ultrametric distances, but to improve optimization, we apply loss in both ultrametric space and Euclidean space:

$$L_{feat} = \sum_{(s_p, s_n) \in \mathcal{S}} \ell_{ultra}(s_p, s_n) + \alpha \ell_{Euclid}(s_p, s_n). \quad (4)$$

The ultrametric term forces the optimization to find a hierarchical decomposition of the scene, while the Euclidean term serves as regularization. Note that the Euclidean term has the advantage of directly providing a gradient to all

considered feature pairs, whereas the ultrametric loss only provides a loss to the active (maximal) edge along the path between two segments.

As described in Eqs. (1) and (2), ultrametric distances are defined on a graph. At each training step, we form an approximate graph by sampling 4096 pixels within the image, connect each pixel to its 10 nearest neighbors, and use feature distances as edge weights. We use the binary partition tree algorithm [13] to efficiently compute ultrametric distances during training.

**Hierarchical Sampling** Because our input segmentations overlap within and across views, it is ambiguous whether a pair of pixels “lie within the same mask” and therefore represent a positive pair in our contrastive formulation. For example, two pixels may be within the same mask at a coarse granularity but not for a finer granularity. We address this ambiguity with a simple tree-based strategy for sampling positive and negative pairs of pixels.

For each image, we organize the segmentation masks into a hierarchical structure determined by the inclusion ratio between them. We additionally include an all-positive mask as the root of the tree. Then, for each view, we sample positive and negative pairs starting from the leaf nodes of the hierarchical tree, *i.e.* masks at the finest granularity. For a leaf mask  $A$  with parent mask  $B$ , we randomly select two pixels in  $A$  and designate these as a positive pair. We then randomly select one pixel in  $A$  and one pixel in  $\bar{A} \cap B$  and designate these as a negative pair. We then move to the parent mask and repeat. Additional details are provided in the supplementary.

We note that masks grow in size as we rise up the hierarchy, and pairs which were declared negative at finer granularity will be declared positive in coarser segmentations. We mitigate this conflict by computing the contrastive loss *within each level*. This asks that the positive distances be smaller than the negative distances within a single level, and avoids the possibility of a pair being simultaneously positive and negative in Eq. (3). Taken together, our sampling strategy reflects an ultrametric structure and helps the segmentation hierarchy propagate across the scene.

**Improving Depth with Segmentation** Our approach shows that the 3D structure of a scene can resolve conflicts in segmentation cues. Conversely, can segmentation cues help resolve ambiguity in 3D structure? To explore this possibility, we add an assumption that regions belonging to the same semantic mask have smoothly changing depth. We propose a regularization that penalizes changes in curvature (*i.e.* the third derivative of depth) within a segment. Concretely, at each training iteration, we sample  $k_{depth}$  local patches of  $4 \times 4$  pixels, ensuring that each sampled patch lies within one of the SAM’s finest-grained mask predictions. The depth continuity loss is defined as

$$L_{dc} = \sum_{p_0 \in P_D} \max\left(\frac{(d_{p_0} - 3d_{p_1} + 3d_{p_2} - d_{p_3})}{(\max(d_{p_0}, d_{p_1}, d_{p_2}, d_{p_3}) \Delta \theta)^3} - t, 0\right) \quad (5)$$

where  $\Delta\theta$  represents the ray angle difference between adjacent pixels,  $t$  denotes a threshold, and  $P_D$  contains the sampled pixels, with  $p_0, \dots, p_3$  denoting adjacent pixels in a row or column. For training stability, we start using this loss halfway through training. Fig. 5 presents the benefits of the depth continuity loss.

#### 4.2 Segmentation from Ultrametric Features

After training our ultrametric feature field, we can perform 2D or 3D hierarchical segmentation by applying the Watershed transform on either rendered feature maps or the 3D feature field itself.

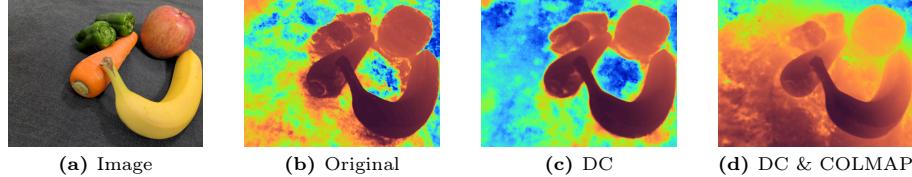
**Segmenting in 2D** To segment a 2D image from our feature field, we begin by rendering our feature field to the viewpoint of the image. We then construct a graph from the feature map, denoted as  $\mathcal{G} = (V, E)$ , where  $V$  contains the pixel features, and  $E$  contains edges connecting each pixel to its 4 spatial neighbors. Edge lengths are defined as feature distances. With a given threshold  $t$  as the indicator of granularity, we remove all edges longer than this threshold, resulting in a new graph denoted as  $\mathcal{G}_t$ . We then identify all connected components within this graph, and return these components as our segmentation. We use this approach to compare against 2D segmentation methods, but we note that 2D-based segmentation is not view-consistent.

**Segmenting in 3D** To achieve 3D-consistent segmentation, we create a featurized 3D point cloud by unprojecting feature maps and depth maps rendered from the optimized implicit field. Following related NeRF works, we remove outliers and downsample the pointcloud using Open3D [64]. Then, we construct a k-nearest-neighbor (KNN) graph on the 3D point cloud, and we set each edge weight in the KNN graph to the feature distance between the connected vertices. For a threshold  $t$  indicating the level of granularity, we remove all edges longer than that threshold, and the remaining connected components represent segments at the specified level of granularity. In practice, we keep the  $N$  largest components as our final segmentation. Using different values of  $t$  allows for segmentation at varying levels of granularity.

To propagate the 3D segmentation into a novel view, we first render a depth map of the novel view and unproject the render into 3D points. Then, for each unprojected pixel, we find its  $k$  nearest neighbors to the previously estimated KNN graph and assign a segmentation label using the mode of the neighborhood.

### 5 Experiments

We report qualitative and quantitative evaluations for our method and a set of relevant baselines. In Sec. 5.1 we give an overview of the datasets we evaluate on, and we define our evaluation metrics. In Sec. 5.2 and Sec. 5.3, we present quantitative comparisons and ablation studies. We present implementation details in Sec. 5.4.



**Fig. 5: Depth Continuity:** Our depth continuity loss (labeled as DC above) leads to smoother and more plausible depth estimation, and can be seamlessly combined with additional depth cues such as COLMAP [44], resulting in even greater accuracy.

### 5.1 Datasets and Metrics

*PartNet Dataset* The official PartNet dataset [8,36] contains 26,671 3D models with professionally verified hierarchical part decomposition. Furthermore, each object model is rendered into 24 viewpoints that uniformly cover the viewing sphere. PartNet offers us the unique ability to evaluate on hierarchical segmentation that verifiably aligns with human perception of structural hierarchy. We evaluate on five objects from each of the *chair*, *table*, and *storage furniture* categories, totalling to 15 objects models.

*Blender with Hierarchical Segmentation* While PartNet contains 3D objects with hierarchical part annotations, each model is visually simplistic and lacks the photometric complexity often associated with NeRFs. To the best of our knowledge, no publicly available dataset exists that displays both complex photometric structure as well as 3D-consistent hierarchical segmentation labels. Thus, we create a new synthetic dataset based on the Blender Dataset [33] which we call Blender with Hierarchical Segmentation (Blender-HS). We choose three scenes from the Blender Dataset that exhibit clear hierarchical structure: Lego, Hotdog, and Drums, and for each we define three granularity levels, which we denote as “scene”, “collection”, and “object”. The object level consists of each distinct 3D asset; the “collection” level contains 3D asset groupings that were decided by the original artist; the “scene” level consists of the entire scene. While many potential hierarchies exist in general, the artist-defined groupings naturally represent a hierarchy that makes sense for humans. We present visualizations of the hierarchical ground truth in the supplementary. We use 100 views in the training set for training and 10 views from the validation set for evaluation.

*Normalized Covering Score* To assess the quality of hierarchical segmentation, we use the Normalized Covering (NC) score [19]. This metric averages the Intersection over Union (IoU) between each ground truth mask and the *best-matching* (*i.e.* most-overlapping) predicted mask. We calculate a separate NC score for each granularity and report the mean of these scores. We also adapt the metric to evaluate the accuracy of point clouds segmentation on PartNet [36].

*Segmentation Injectivity Score* In hierarchical segmentation, it is important that each pixel belongs to only one mask for each level of granularity. To measure this, we propose a metric which we call Segmentation Injectivity (SI). For each ground truth segmentation, we retrieve its best-matching predicted mask and sample two random points  $p_1$  and  $p_2$  from that mask. We then query the model for a new mask at each of these points, at the same granularity. We compute the Intersection over Union (IoU) between the two resulting masks. Note that a perfect model will return the same mask at both locations, since these points belong to the same ground-truth segment, whereas a worse model will return masks which may not even overlap. Since the random point sampling introduces randomness, we run this 100 times per ground truth mask and average the scores across all ground truth masks and all viewpoints. We find this is more tractable than estimating masks densely, and in practice exhibits low variance. Note there is a trade-off between the SI score and the NC score: predicting an excessive number of overlapping masks could inflate the NC score but would significantly lower the SI score.

*View Consistency Score* We use a View Consistency (VC) score to measure the 3D consistency of image segmentations. The key idea of this score is to estimate segmentations in two nearby viewpoints independently, and then warp these estimates onto one another, and measure their agreement. To compute this score, we begin by defining a pixel transformation  $T$  from a source viewpoint to another one shifted 10 degrees, using ground truth depth and camera pose. Next, for each ground truth segmentation, we retrieve its best-matching predicted mask, sample a random point  $p_1$  from that mask, and compute the point’s location in the shifted viewpoint  $p_2 = T(p_1)$ . We then obtain segmentation estimates to compare, centered on  $p_1$  in the first view and centered on  $p_2$  in the second. Finally we warp  $p_2$ ’s mask into the  $p_1$  viewpoint, and measure IoU with the  $p_1$  mask. To ensure that the IoU does not merely reveal occlusion/disocclusion differences, we use ground truth visibility masks to remove pixels which are invisible in either view, before computing the IoU. Similar to the other scores, we compute this score 100 times per ground truth mask and average the scores across all ground truth masks and all viewpoints.

## 5.2 View-Consistent Hierarchical Segmentation

*Baselines* Because the exact task of 3D-consistent hierarchical segmentation from images collections is less explored, we adapt state-of-the-art NeRF baselines to this setting. We adapt language-based methods LSeg [28], DFF [26], and LeRF [21] to predict a segmentation mask based on a 2D pixel query, by querying the rendered feature map and estimating a segmentation mask via feature similarity. We generate multi-granular predictions by using different thresholds on the feature similarity. We also adapt SAM3D [7] as a baseline, which we modify to produce multiple segmentations instead of only one. We propagate SAM outputs from 20 training views into the novel view to generate a variety of masks.

**Table 1: Results on Blender-HS.** We report the Normalized Covering (NC), Segmentation Injectivity (SI), and View Consistency (VC) as a percentage.  $NC_{obj}$ ,  $NC_{coll}$ , and  $NC_{scene}$  refer to the NC score on objects, collections, and scenes, respectively.

Method	$NC_{obj} \uparrow$	$NC_{coll} \uparrow$	$NC_{scene} \uparrow$	$NC_{mean} \uparrow$	$SI \uparrow$	$VC \uparrow$
LSeg [28]	21.8	40.6	67.0	43.2	69.8	53.6
LSeg + DFF [26]	16.4	42.7	83.9	47.7	84.8	<b>82.7</b>
LeRF [21]	26.6	41.3	85.7	51.2	59.8	64.1
SAM [24]	44.3	65.2	78.7	62.7	80.8	46.2
SAM3D [7]	46.0	59.3	53.2	52.8	-	72.4
Ours, 2D	<b>50.0</b>	63.0	94.0	69.0	<b>100.0</b>	67.8
Ours	48.0	<b>67.7</b>	<b>97.1</b>	<b>70.9</b>	<b>100.0</b>	78.9

*Quantitative Evaluation* In Table 1, we report segmentation metrics for our 2D inference mode, our 3D inference mode, and our baselines. Our method significantly outperforms all other methods in Normalized Covering score (*i.e.* segmentation accuracy). We even outperform SAM, suggesting that our ultrametric feature field not only distills thousands of SAM predictions into a compact set of hierarchical and view-consistent masks, but that our masks are more accurate than *any* of the original SAM predictions. Due to the Watershed transform, we also achieve a perfect score on segmentation injectivity (SI), while the other methods (which are non-hierarchical) tend to predict overlapping masks. Finally, our 3D inference shows significant improvements in view consistency over all methods except DFF. We also point out that DFF’s object-level NC score suggests that it does not segment small objects, which are significantly more challenging to segment in a view-consistent manner, and this fact may inflate the VC score for the method.

We additionally visualize our ultrametric feature field on the Lego scene in the supplementary, showing sharper features than DFF.

*Qualitative Analysis* We provide qualitative validation of our method on two real-world scenes from the Tanks and Temples Dataset [25]. In Fig. 4, we visualize three segmentation granularities on the truck scene. In Fig. 1, we show our segmentations on the Bobcat tractor scene, which resolves inconsistencies from the source SAM masks. Our method’s segmentations tend to have sharp boundaries, and the hierarchical structure decomposes the truck and tractor into intuitive parts and subparts.

*Partnet Experiments* Each object in PartNet has a unique hierarchical part decomposition that may vary in both the number of parts and number of relationships between parts. To evaluate the part hierarchy using our NC metric, we categorize the part hierarchy of each object into three levels: finest, middle, and coarse. The finest level corresponds to the leaf level, while the coarse level represents the root level (*i.e.*, full object), and the rest are the middle level. In Table 2, we compare our approach with LSeg+DFF [26] and SAM3D [7], two other methods that produce 3D-consistent segmentation. Our method surpasses

**Table 2: Evaluation in PartNet.** We report the Normalized Covering (NC) as a percentage on PartNet dataset. NC<sub>fine</sub>, NC<sub>mid</sub>, and NC<sub>coar</sub> refer to the NC score on three different hierarchy levels. <sup>†</sup> means the method is evaluated based on point clouds.

Method	NC <sub>fine</sub> ↑	NC <sub>mid</sub> ↑	NC <sub>coar</sub> ↑	NC <sub>mean</sub> ↑
LSeg + DFF [26]	22.4	48.5	70.6	47.2
SAM3D [7]	<b>51.0</b>	60.8	66.8	59.6
Ours	49.9	<b>63.3</b>	<b>82.8</b>	<b>65.3</b>
Ours <sup>†</sup>	48.9	60.4	-	-
3D-PIS <sup>†</sup> [48]	<b>52.2</b>	<b>75.0</b>	-	-

both in NC score, while also being the only method to produce hierarchical segmentation. We also report the NC Score based on point clouds and compare our method with 3D-PIS [48], a supervised point clouds segmentation network. We can see that our method can achieve comparable NC Score on the finest level. Our method falls behind on the middle level, as the ground truth hierarchical tree is constructed semantically, 3D-PIS learns these semantics from supervision while ours does not.

### 5.3 Ablation Experiments

*Hierarchical Data Sampling* In Tab. 3 we evaluate the influence of hierarchical data sampling for contrastive learning. In the baseline approach, positive pairs are randomly sampled within each mask, and negative pairs are sampled with one point inside and one point outside. Results indicate that using the hierarchical sampling strategy improves the overall performance by 5.7 points.

*Ultrametric training* In Tab. 3 we report the impact of the ultrametric loss,  $\ell_{\text{ultra}}$ . In the baseline method, we train the feature space using only the Euclidean loss,  $\ell_{\text{Euclid}}$ . Results show that the ultrametric training improves the segmentation performance by 2.1 points.

*Depth Continuity Loss* Tab. 4 shows the influence of using the depth continuity loss during training. In addition to evaluating the NC score, we evaluate depth estimation accuracy, using mean  $\ell_2$  error. Incorporating the depth continuity loss improves both segmentation accuracy and depth accuracy. Fig. 5 qualitatively illustrates the benefit of the depth continuity loss in real data.

### 5.4 Implementation Details

We implement our method with help from the publicly available codebase from DFF [26]. We model our feature branch after the RGB branch of Instant-NGP [37] – we employ a multi-resolution grid hash encoder to transform 3D coordinates into features, followed by an MLP. For the grid hash encoder, we

**Table 3:** Ablation on Hierarchical Sampling (HS) and Ultrametric Training (UT).

Method	NC <sub>obj</sub>	NC <sub>coll</sub>	NC <sub>scene</sub>	NC <sub>mean</sub>
Ours	<b>48.0</b>	<b>67.7</b>	<b>97.1</b>	<b>70.9</b>
w/o UT	47.1	62.8	96.4	68.8
w/o HS	42.6	64.0	89.0	65.2

**Table 4:** Ablation on Depth Continuity (DC) loss.

Method	NC <sub>mean</sub>	Depth Error
Ours	<b>70.9</b>	<b>0.059</b>
w/o DC	67.1	0.089

configure the number of levels to 17, features per level to 4, and the hash map size to  $2^{20}$ . The MLP comprises three hidden layers with 128 dimensions each, producing a final output feature of 256 dimensions. We provide additional implementation details in the supplementary.

For our quantitative evaluation on Blender-HS, we compute the NC score of LSeg, DFF, LeRF, and our method across 50 distance thresholds ranging from 0.01 to 0.50. We follow DFF [26] and perform all evaluation on 4× downsampled images. For all methods, we exclude masks containing fewer than 20 pixels.

## 6 Discussion and Limitations

A key limitation of our approach is its dependence on high-quality point clouds produced by the NeRF. While the depth-smoothing loss outlined in Section 5.3 improves point cloud quality, there is much room for further improvement. Regarding our evaluation, we find there is a scarcity of high-quality datasets for hierarchical 3D segmentation that exhibit complex scene structure. Additionally, it is ambiguous *which* hierarchies are the most meaningful and appropriate in complex environments, without defining end-tasks that rely on these hierarchies. While our Blender-HS Dataset is a first step in providing ground truth on hierarchical 3D segmentation in complex scenes, it is limited to only three scenes. We hope that future efforts can develop better and larger hierarchical 3D datasets, to enable more comprehensive evaluations.

## 7 Conclusion

Consistent hierarchical 3D segmentation is essential for many applications involving mobile agents interacting with the real world at scale, such as robotics and augmented reality. In this work we have demonstrated significant progress towards achieving consistent hierarchical segmentation, building on state-of-the-art systems that perform multi-granularity segmentation in images, whose output predictions are neither hierarchical nor consistent across different views. Our ultrametric feature field distills this inconsistent 2D information into a representation that can be queried at will for arbitrary-granularity segmentations that are consistent across views.

*Acknowledgements* This work was supported by a Vannevar Bush Faculty Fellowship and ARL grant W911NF-21-2-0104.

## References

1. Arbelaez, P.: Boundary extraction in natural images using ultrametric contour maps. In: 2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06). pp. 182–182 (2006). <https://doi.org/10.1109/CVPRW.2006.48>
2. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. IEEE transactions on pattern analysis and machine intelligence **33**(5), 898–916 (2010)
3. Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.P.: Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5855–5864 (2021)
4. Beucher, S.: Watershed, hierarchical segmentation and waterfall algorithm. Mathematical morphology and its applications to image processing pp. 69–76 (1994)
5. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers (2021)
6. Cen, J., Fang, J., Yang, C., Xie, L., Zhang, X., Shen, W., Tian, Q.: Segment any 3d gaussians (2024), <https://arxiv.org/abs/2312.00860>
7. Cen, J., Zhou, Z., Fang, J., Shen, W., Xie, L., Jiang, D., Zhang, X., Tian, Q.: Segment anything in 3d with nerfs (2023)
8. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015)
9. Chen, A., Xu, Z., Geiger, A., Yu, J., Su, H.: Tensorrf: Tensorial radiance fields. In: European Conference on Computer Vision. pp. 333–350. Springer (2022)
10. Chen, A., Xu, Z., Wei, X., Tang, S., Su, H., Geiger, A.: Factor fields: A unified framework for neural fields and beyond. ArXiv **abs/2302.01226** (2023), <https://api.semanticscholar.org/CorpusID:256503583>
11. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International conference on machine learning. pp. 1597–1607. PMLR (2020)
12. Cheng, B., Misra, I., Schwing, A.G., Kirillov, A., Girdhar, R.: Masked-attention mask transformer for universal image segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 1290–1299 (2022)
13. Cousty, J., Najman, L., Kenmochi, Y., Guimarães, S.: Hierarchical segmentations with graphs: quasi-flat zones, minimum spanning trees, and saliency maps. Journal of Mathematical Imaging and Vision **60**(4), 479–502 (2018)
14. De Goes, F., Goldenstein, S., Velho, L.: A hierarchical segmentation of articulated bodies. Computer graphics forum **27**(5), 1349–1356 (2008)
15. Deng, K., Liu, A., Zhu, J., Ramanan, D.: Depth-supervised nerf: Fewer views and faster training for free. CoRR **abs/2107.02791** (2021), <https://arxiv.org/abs/2107.02791>
16. Gong, K., Gao, Y., Liang, X., Shen, X., Wang, M., Lin, L.: Graphonomy: Universal human parsing via graph transfer learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7450–7459 (2019)
17. Hedman, P., Srinivasan, P.P., Mildenhall, B., Barron, J.T., Debevec, P.E.: Baking neural radiance fields for real-time view synthesis. CoRR **abs/2103.14645** (2021), <https://arxiv.org/abs/2103.14645>

18. Johnson, S.C.: Hierarchical clustering schemes. *Psychometrika* **32**(3), 241–254 (1967)
19. Ke, T.W., Hwang, J.J., Guo, Y., Wang, X., Yu, S.X.: Unsupervised hierarchical semantic segmentation with multiview cosegmentation and clustering transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2571–2581 (2022)
20. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics* **42**(4) (July 2023), <https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/>
21. Kerr, J., Kim, C.M., Goldberg, K., Kanazawa, A., Tancik, M.: Lorf: Language embedded radiance fields (2023)
22. Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., Krishnan, D.: Supervised contrastive learning. *Advances in neural information processing systems* **33**, 18661–18673 (2020)
23. Kim, C.M., Wu, M., Kerr, J., Tancik, M., Goldberg, K., Kanazawa, A.: Garfield: Group anything with radiance fields. In: arXiv (2024)
24. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., Dollár, P., Girshick, R.: Segment anything (2023)
25. Knapitsch, A., Park, J., Zhou, Q.Y., Koltun, V.: Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics* **36**(4) (2017)
26. Kobayashi, S., Matsumoto, E., Sitzmann, V.: Decomposing nerf for editing via feature field distillation. In: Advances in Neural Information Processing Systems. vol. 35 (2022), <https://arxiv.org/pdf/2205.15585.pdf>
27. Kundu, A., Genova, K., Yin, X., Fathi, A., Pantofaru, C., Guibas, L., Tagliasacchi, A., Dellaert, F., Funkhouser, T.: Panoptic neural fields: A semantic object-aware neural scene representation. In: CVPR (2022)
28. Li, B., Weinberger, K.Q., Belongie, S., Koltun, V., Ranftl, R.: Language-driven semantic segmentation (2022)
29. Li, L., Zhou, T., Wang, W., Li, J., Yang, Y.: Deep hierarchical semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1246–1257 (2022)
30. Liu, L., Gu, J., Lin, K.Z., Chua, T.S., Theobalt, C.: Neural sparse voxel fields. NeurIPS (2020)
31. Liu, X., Chen, J., Yu, H., Tai, Y.W., Tang, C.K.: Unsupervised multi-view object segmentation using radiance field propagation (2022)
32. Liu, Y., Hu, B., Huang, J., Tai, Y.W., Tang, C.K.: Instance neural radiance field (2023)
33. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis (2020)
34. Milligan, G.W.: Ultrametric hierarchical clustering algorithms. *Psychometrika* **44**(3), 343–346 (1979)
35. Mirzaei, A., Aumentado-Armstrong, T., Derpanis, K.G., Kelly, J., Brubaker, M.A., Gilitschenski, I., Levinshtein, A.: Spin-nerf: Multiview segmentation and perceptual inpainting with neural radiance fields (2023)
36. Mo, K., Zhu, S., Chang, A.X., Yi, L., Tripathi, S., Guibas, L.J., Su, H.: PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)

37. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.* **41**(4), 102:1–102:15 (Jul 2022). <https://doi.org/10.1145/3528223.3530127>, <https://doi.org/10.1145/3528223.3530127>
38. Najman, L., Schmitt, M.: Geodesic saliency of watershed contours and hierarchical segmentation. *IEEE Transactions on pattern analysis and machine intelligence* **18**(12), 1163–1173 (1996)
39. Niemeyer, M., Barron, J.T., Mildenhall, B., Sajjadi, M.S.M., Geiger, A., Radwan, N.: Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 5470–5480 (2021), <https://api.semanticscholar.org/CorpusID:244773517>
40. Oechsle, M., Peng, S., Geiger, A.: Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In: International Conference on Computer Vision (ICCV) (2021)
41. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision (2021)
42. Roessle, B., Barron, J.T., Mildenhall, B., Srinivasan, P.P., Nießner, M.: Dense depth priors for neural radiance fields from sparse input views. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 12882–12891 (2021), <https://api.semanticscholar.org/CorpusID:244921004>
43. Roessle, B., Barron, J.T., Mildenhall, B., Srinivasan, P.P., Nießner, M.: Dense depth priors for neural radiance fields from sparse input views (2022)
44. Schönberger, J.L., Zheng, E., Pollefeys, M., Frahm, J.M.: Pixelwise view selection for unstructured multi-view stereo. In: European Conference on Computer Vision (ECCV) (2016)
45. Siddiqui, Y., Porzi, L., Bul'ó, S.R., Muller, N., Nießner, M., Dai, A., Kortschieder, P.: Panoptic lifting for 3d scene understanding with neural fields. 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 9043–9052 (2022), <https://api.semanticscholar.org/CorpusID:254877618>
46. Sudderth, E., Torralba, A., Freeman, W., Willsky, A.: Learning hierarchical models of scenes, objects, and parts. In: Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1. vol. 2, pp. 1331–1338 Vol. 2 (2005). <https://doi.org/10.1109/ICCV.2005.137>
47. Sun, C.Y., Yang, Y.Q., Guo, H.X., Wang, P.S., Tong, X., Liu, Y., Shum, H.Y.: Semi-supervised 3d shape segmentation with multilevel consistency and part substitution (2022)
48. Sun, C., Tong, X., Liu, Y.: Semantic segmentation-assisted instance feature fusion for multi-level 3d part instance segmentation (2022)
49. Tschernezki, V., Laina, I., Larlus, D., Vedaldi, A.: Neural feature fusion fields: 3d distillation of self-supervised 2d image representations. In: 2022 International Conference on 3D Vision (3DV). pp. 443–453. IEEE (2022)
50. Uy, M.A., Martin-Brualla, R., Guibas, L., Li, K.: Scade: Nerfs from space carving with ambiguity-aware depth estimates. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2023)
51. Vora, S., Radwan, N., Greff, K., Meyer, H., Genova, K., Sajjadi, M.S.M., Pot, E., Tagliasacchi, A., Duckworth, D.: Nesf: Neural semantic fields for generalizable semantic segmentation of 3d scenes (2021)

52. Wang, P., Liu, L., Liu, Y., Theobalt, C., Komura, T., Wang, W.: Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. ArXiv **abs/2106.10689** (2021), <https://api.semanticscholar.org/CorpusID:235490453>
53. Wang, W., Zhang, Z., Qi, S., Shen, J., Pang, Y., Shao, L.: Learning compositional neural information fusion for human parsing. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 5703–5713 (2019)
54. Wang, W., Zhu, H., Dai, J., Pang, Y., Shen, J., Shao, L.: Hierarchical human parsing with typed part-relation reasoning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 8929–8939 (2020)
55. Xu, X., Yang, Y., Mo, K., Pan, B., Yi, L., Guibas, L.: Jacobinerf: Nerf shaping with mutual information gradients (2023)
56. Xu, Y., Carlinet, E., Géraud, T., Najman, L.: Hierarchical segmentation using tree-based shape spaces. IEEE transactions on pattern analysis and machine intelligence **39**(3), 457–469 (2016)
57. Yariv, L., Gu, J., Kasten, Y., Lipman, Y.: Volume rendering of neural implicit surfaces. In: Thirty-Fifth Conference on Neural Information Processing Systems (2021)
58. Yarkony, J.E., Fowlkes, C.: Planar ultrametrics for image segmentation. In: Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 28. Curran Associates, Inc. (2015), [https://proceedings.neurips.cc/paper\\_files/paper/2015/file/3416a75f4cea9109507cacd8e2f2aefc-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2015/file/3416a75f4cea9109507cacd8e2f2aefc-Paper.pdf)
59. Yin, Y., Fu, Z., Yang, F., Lin, G.: Or-nerf: Object removing from 3d scenes guided by multiview segmentation with neural radiance fields (2023)
60. Ying, H., Yin, Y., Zhang, J., Wang, F., Yu, T., Huang, R., Fang, L.: Omniseg3d: Omniversal 3d segmentation via hierarchical contrastive learning (2023)
61. Yu, Z., Chen, A., Antic, B., Peng, S.P., Bhattacharyya, A., Niemeyer, M., Tang, S., Sattler, T., Geiger, A.: Sdfstudio: A unified framework for surface reconstruction (2022), <https://github.com/autonomousvision/sdfstudio>
62. Zhao, H., Puig, X., Zhou, B., Fidler, S., Torralba, A.: Open vocabulary scene parsing (2017)
63. Zhi, S., Laidlow, T., Leutenegger, S., Davison, A.J.: In-place scene labelling and understanding with implicit scene representation (2021)
64. Zhou, Q.Y., Park, J., Koltun, V.: Open3D: A modern library for 3D data processing. arXiv:1801.09847 (2018)
65. Zhou, Y., Gu, J., Li, X., Liu, M., Fang, Y., Su, H.: Partslip++: Enhancing low-shot 3d part segmentation via multi-view instance segmentation and maximum likelihood estimation (2023), <https://arxiv.org/abs/2312.03015>

## 8 Supplementary Material

We provide additional implementation details in Sec. A. We present additional qualitative results on Blender-HS dataset, PartNet dataset, and LLFF dataset in Sec. B. Additionally, we include a video attachment with visualizations of the view-consistent hierarchical segmentation results.

### A. Additional Implementation Details

**A.1. Hyperparameters** On Blender-HS and PartNet, we train our model for 20,000 iterations with a batch size of 4096 and use the same optimization parameters as DDF. In contrastive learning (see Eq. (3)), we set the temperature  $\tau$  to 0.1 and sample 64 positive and negative pairs from each mask. We set the loss weight  $\alpha$  of the Euclidean loss in Eq. (4) to 1. For depth continuity loss, we sample 16 patches per mask, and begin using the depth continuity loss after 5000 iterations. During 3D inference, we extract a point cloud from training-view depth maps, apply voxel downsampling with voxel size  $2 \times 10^{-3}$ , and run outlier removal with distance threshold  $4 \times 10^{-3}$  and number threshold of 1. We build the graph of points using  $k_{graph} = 16$  nearest neighbors, and we transfer point segmentation labels into a novel view using the mode of  $k_{query} = 5$  nearest neighbors. We retain  $N = 200$  graph components and set the distance threshold  $d$  to be  $5 \times 10^{-3}$ . Please refer to Sec. 4 for definitions of the above hyperparameters.

**A.2. Hierarchical Sampling** We first organize the segmentation masks into a hierarchical structure determined by the inclusion ratio between them. One mask  $A$  is designated as a *child* of another mask  $B$  when  $\frac{|A \cap B|}{|A|} > p_{in}$  and  $\frac{|A \cap B|}{|A \cup B|} < p_{IoU}$ . We empirically set  $p_{in} = 0.95$  and  $p_{IoU} = 0.85$ . We present the hierarchical sampling algorithm we introduced in Sec. 4.1 in Algorithm 1.

We sample same number of positive pairs and negative pairs for training for training efficiency. Implementing a 4-1 ratio (4× more negatives than positives) instead of 1-1, the normalized covering (NC) score increases from 0.709 to **0.720**. However, this slow down our training by 43%, primarily due to the time-consuming computation of ultrametric distances and the associated minimum spanning tree.

### A.3. Additional Details on Evaluation Metrics

*Normalized Covering Score* As described in Sec. 5.1, we measure the *quality* of hierarchical segmentation with the Normalized Covering (NC) score [19]. This metric averages the Intersection over Union (IoU) between each ground truth mask and the *best-matching* (*i.e.* most-overlapping) predicted mask. The metric is defined as

$$\text{NC}(S' \rightarrow S) = \frac{1}{|S|} \sum_{A \in S} \max_{A' \in S'} \frac{|A \cap A'|}{|A \cup A'|} \quad (6)$$

**Algorithm 1** Data Sampling

---

```

pos_samples ← []
neg_samples ← []
for all  $A \in \text{leaf\_masks}$  do
    # Sampling positive pairs from the leaf node
    sample ← (Random( $A$ ), Random( $A$ ))
    while  $A$  has Parent do
         $B \leftarrow A.\text{Parent}$ 
        pos_samples += sample
        # Sampling negative pairs for the current level
        sample ← (Random( $A$ ), Random( $\bar{A} \cap B$ ))
        neg_samples += sample
         $A \leftarrow B$ 
    end while
end for
return pos_samples, neg_samples

```

---

Where  $S$  denotes all segmentation masks and  $S'$  denotes all predicted masks. For LSeg, DFF, and LeRF, which only output feature fields without segmentation results ( $S'$ ), we adopt a similar approach as our method, and we extract segmentations by thresholding feature distances.

*Segmentation Injectivity Score* We propose the Segmentation Injectivity (SI) score to measure if each pixel belongs to only one mask for each level of granularity. As described in Sec. 5.1, given a ground truth mask, we first randomly sample  $p_1$  and  $p_2$  from that mask, and then query the model at these points and granularity for a new mask prediction. Then, we measure the IoU between the two resulting masks. We iterate this process  $N = 100$  for each ground truth mask, calculating scores for each run. The final SI score is obtained by averaging the scores across all ground truth masks and viewpoints.

We represent the segmentation model as  $F(v, p, t) \rightarrow A'$  where  $v$  denotes the viewpoint,  $p$  represents the pixel query,  $t$  corresponds to the granularity level, and  $A'$  is the resulting segmentation mask. The SI score is defined as

$$\text{SI}(S' \rightarrow S) = \frac{1}{|N||S|} \sum_{A \in S} \sum_{i=1}^N \frac{|F(v, p_1^i, t) \cap F(v, p_2^i, t)|}{|(F(v, p_1^i, t) \cup F(v, p_2^i, t))|}$$

$$\text{where } t = \arg \max_t \frac{|A \cap |F(v, p_1^i, t)|}{|A \cup |F(v, p_1^i, t)||}$$

where  $v$  represents the view corresponding to the ground truth mask  $A$ .

*View Consistency Score* We use the View Consistency (VC) score to measure the 3D consistency of image segmentations. Starting with the source view, we rotate the camera by 10 degrees, rendering both a new image and the corresponding ground truth visibility mask in the shifted view – Fig. 6 provides an example of two viewpoints and their visibility mask on the Blender Hotdog scene.

For a given point query  $p_1$  and a granularity  $t$  and its mask prediction  $A_1 = F(v, p_1, t)$  in the source view, we leverage the ground truth camera parameters to warp the point to  $p_2 = T(p_1)$  and the mask prediction to  $T(A_1)$  in the shifted view where  $T$  denotes the pixel transformation. Following this, we query the model in the shifted view with  $p_2$  using the same threshold  $t$ , resulting in  $A_2 = F(v', p_2, t)$ .

Utilizing the visibility mask  $V$ , we eliminate pixels that are occluded in either view from  $A_2$  and  $T(A_1)$ . The Intersection over Union (IoU) between the remaining masks is computed as the VC score for this sample. We reduce the noise induced by random sampling by computing this score for  $N = 100$  times per ground truth mask.

Taken together, the VC score is defined as

$$\text{VC}(S' \rightarrow S) = \frac{1}{|N||S|} \sum_{A \in S} \sum_{i=1}^N \frac{|T(A_1^i) \cap A_2^i \cap V|}{|(T(A_1^i) \cup A_2^i) \cap V|}$$

where  $A_1^i = F(v, p_1^i, \arg \max_t \frac{|A \cap |F(v, p_1^i, t)|}{|A \cup |F(v, p_1^i, t)|})$

$$A_2^i = F(v', p_2^i, \arg \max_t \frac{|A \cap |F(v', p_2^i, t)|}{|A \cup |F(v', p_2^i, t)|})$$

For additional details, please refer to Sec. 5.1.

We also evaluate the View Consistency across multiple angles, in Tab. 5. The ranking of the methods is the same.

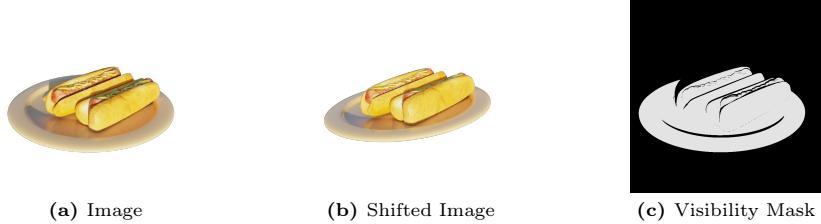
**Table 5:** View Consistency score with different view angles.

Method	VC <sub>10°</sub> ↑	VC <sub>45°</sub> ↑	VC <sub>90°</sub> ↑	VC <sub>135°</sub> ↑
LSeg [26]	0.536	0.522	0.510	0.498
LSeg + DFF [24]	<b>0.827</b>	<b>0.813</b>	<b>0.810</b>	<b>0.808</b>
SAM3D [6]	0.724	0.601	0.578	0.539
Ours	0.789	0.763	0.742	0.712

*Depth Error* We leverage the ground truth depth map rendered in blender to compute the depth error of our method. The scale of the depth error adheres to the normalized NeRF scene.

#### A.4. Additional Details on Baselines

*DFF* We configure DFF to use a white background and employ uniform ray sampling on the BlenderHS dataset. All other hyperparameters directly adhere



**Fig. 6: View Consistency:** We evaluate the view consistency between the source viewpoint (a) and another one shifted 10 degrees (b). We render the ground truth visibility mask (c) with ray casting to avoid the occlusion/disocclusion between views.

**Table 6: Distilled Feature Fields:** We present the NC score of DFF with volume rendering (VR) and the NC score of the official codebase on our BlenderHS dataset.

Method	NC <sub>obj</sub> ↑	NC <sub>coll</sub> ↑	NC <sub>scene</sub> ↑	NC <sub>mean</sub> ↑
DFF (VR)	<b>0.164</b>	<b>0.427</b>	<b>0.839</b>	<b>0.477</b>
DFF (Official)	0.082	0.286	0.666	0.345

to the official implementation. Furthermore, DFF’s official code<sup>2</sup> does not apply volume rendering to the feature branch. Instead, it generates a feature map by directly querying the volumetric features at the 3D locations of the predicted surface points. We extended their code to perform volume rendering, and we show that using volume rendering leads to improved performance on the BlenderHS dataset (see Tab. 6).

*LeRF* We use LeRF’s reported NSVF hyperparameters for the Blender synthetic dataset. This includes configuring the background to white, selecting uniform sampling as the ray sampling strategy, disabling space distortion, and setting average appearance embedding to off. We train the model for 20000 steps. For the Normalized Covering Score, we report the highest result among all 30 semantic scales available in the LeRF feature field for each ground truth granularity. For the Segmentation Injectivity score and View Consistency scores, we evaluate LeRF at the semantic scale corresponding to the ground truth granularity which yields the highest NC score.

*SAM3D* Given a pretrained NeRF and a segmentation mask from a single view, SAM3D optimizes a binary voxel grid using mask inverse rendering and cross-view self-prompting to propagate the mask into 3D. In our experiments, we propagate the SAM masks from the segment-everything mode using 20 training views (while still using all 100 training images to pretrain the NeRF). We observed saturation in SAM3D’s NC score after 20 views, and, on an A6000 GPU, it takes approximately a day per scene to propagate the segmentation maps from 20 views. In contrast, our method takes around 2 hours.

<sup>2</sup> <https://github.com/pfnet-research/distilled-feature-fields>

*SAM* We employ the ViT-H model from the official SAM GitHub repository<sup>3</sup> to generate mask predictions. To generate the training data of our model, we use the segment-everything mode to generate our supervision.

In the evaluation process, when querying segmentation models with a randomly sampled point, we employ the point as a prompt for SAM to generate the segmentation prediction. This approach, compared to evaluating based on the output of the segment-everything mode, yields a higher NC score and provides a clearer granularity level.

**A.5. Training and Inference Time** We train and perform inference on a Titan RTX GPU. Training typically takes  $\sim$ 70 minutes, while inference takes 5 seconds per granularity for 10 views. The main expense in inference is the watershed algorithm running on 3D point clouds, which is executed once per granularity and is view-independent.

## B. Additional Qualitative Results

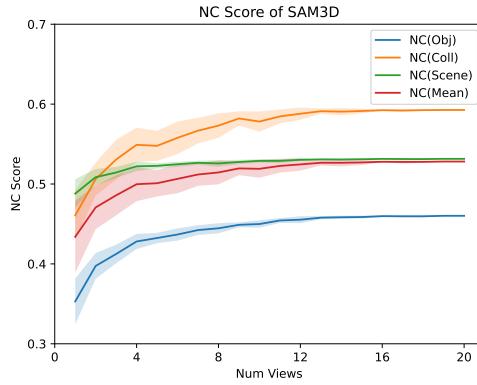
**B.1. BlenderHS Dataset** We first visualize the ground truth segmentations for the Drums scene in the BlenderHS Dataset [33] in Fig. 8. We then present qualitative results on the BlenderHS dataset [33] in Fig. 9. Our segmentations exhibit a hierarchical structure and maintain consistency across different views. We also visualize our ultrametric feature field on the Lego scene in Fig. 11, showing sharper features than DFF.

**B.2. PartNet Dataset** We present qualitative results on the PartNet dataset [36] in Fig. 10. Our method is able to generate hierarchical segmentation results of different objects. Leveraging the 2D masks predicted with SAM as guidance, our method proficiently segments various surfaces of sub-parts within the object, while those are not included in the PartNet ground truth annotations.

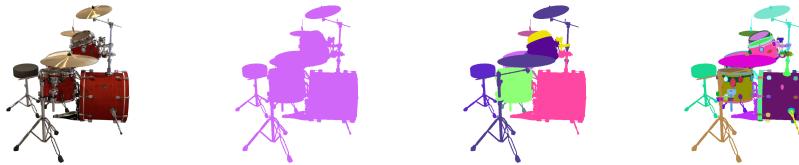
**B.3. LLFF Dataset** We present qualitative results on the the LLFF dataset [33] in Fig. 12. Our approach is able to generate view-consistent hierarchical segmentation results for real-world scenes.

---

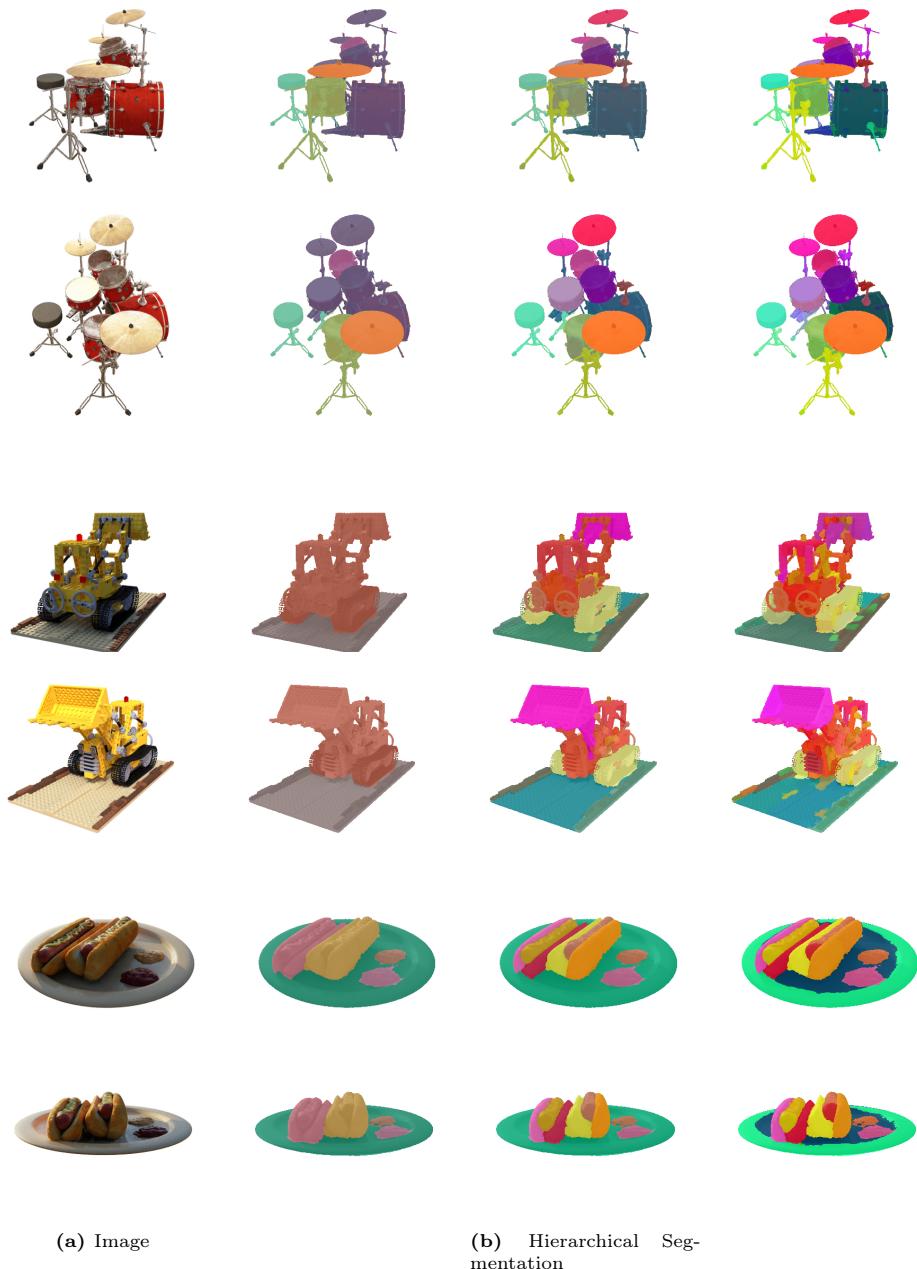
<sup>3</sup> <https://github.com/facebookresearch/segment-anything>



**Fig. 7: NC Score of SAM3D:** We present the Normalized Covering (NC) score (y-axis) of SAM3D, correlating it with the number of views (x-axis) from which we propagate the SAM segmentation masks.



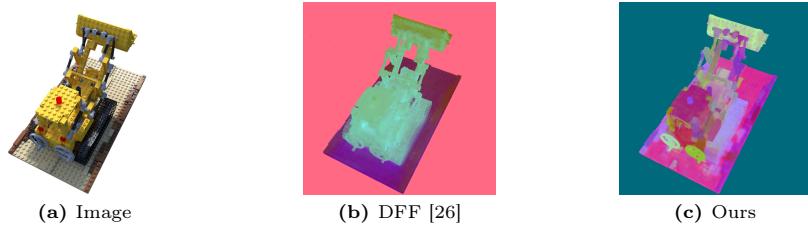
**Fig. 8: Blender with Hierarchical Segmentation (Blender-HS):** We render hierarchical segmentation maps at three levels of granularity, namely *Scene*, *Collection*, and *Object*, using information saved into the blender file by the scene artist.



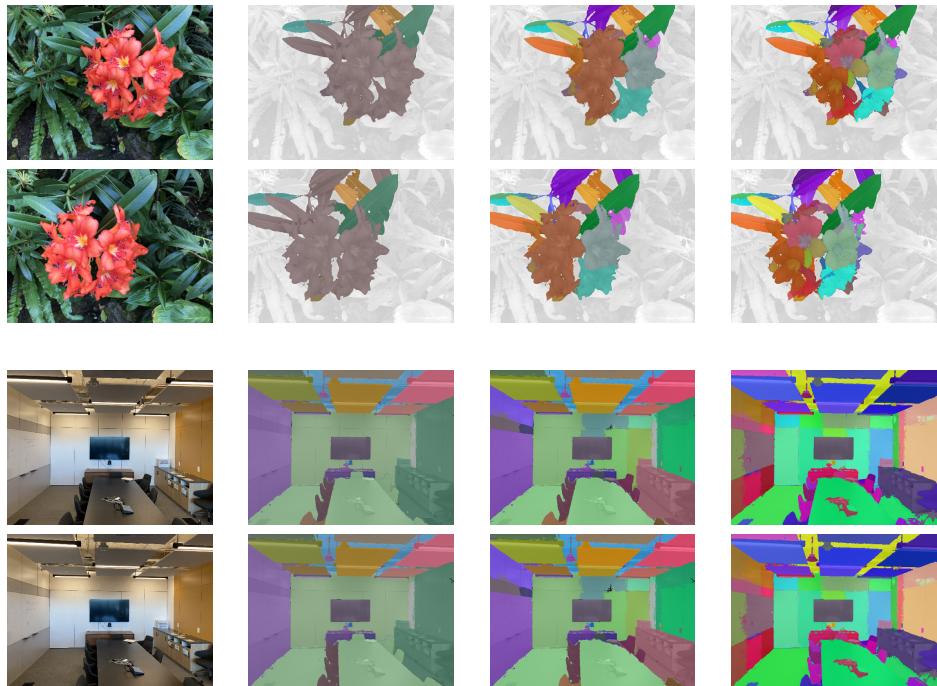
**Fig. 9: BlenderHS Dataset:** We present the qualitative results obtained from our BlenderHS dataset. The segmentation results demonstrate both view consistency and hierarchical structure.



**Fig. 10: PartNet Dataset:** We showcase the qualitative results on the PartNet dataset.



**Fig. 11: Feature Visualization:** We visualize rendered feature maps using PCA. The feature map generated by DFF [26] fails to distinguish between different parts of the Lego. In contrast, our method learns features that can distinguish various Lego bricks.



**Fig. 12: LLFF Dataset:** We showcase the qualitative results on the LLFF dataset.