PeRFception: Perception using Radiance Fields

Anonymous Author(s)
Affiliation
Address
email

Abstract

The recent progress in implicit 3D representation, i.e., Neural Radiance Fields (NeRFs), has made accurate and photorealistic 3D reconstruction possible in a differentiable manner. This new representation can effectively convey the information of hundreds of high-resolution images in one compact format and allows photorealistic synthesis of novel views. In this work, using the variant of NeRF called Plenoxels, we create the first large-scale implicit representation datasets for perception tasks, called the PeRFception dataset, which consists of two parts that incorporate both object-centric and scene-centric scans for classification and segmentation. It shows a significant memory compression rate (96.4%) from the original dataset, while containing both 2D and 3D information in a unified form. We construct the classification and segmentation models that directly take as input this implicit format and also propose a novel augmentation technique to avoid overfitting on backgrounds of images. The code and data will be publicly available.

1 Introduction

Over the last few years, advances in implicit representations have demonstrated great accuracy, versatility, and robustness in representing 3D scenes by mapping low dimensional coordinates to the local properties of the scene, such as occupancy [1, 2], signed distance fields [3, 4], or radiance fields [5, 6, 7]. They offer several benefits that explicit representations (e.g., voxels, meshes, and point clouds) could not represent: smoother geometry, less memory space for storage, novel view synthesis with high visual fidelity, to name a few. Thus, implicit representations have been used for 3D reconstruction [1, 2, 8, 9], novel view synthesis [5, 6, 7, 10, 11, 12, 13, 14, 15], pose estimation [16, 17, 18, 19], image generation [20, 21], and many more.

In particular, Neural Radiance Fields [5] (NeRF) and many follow-up works [10, 11, 12, 14, 15, 22] have shown that implicit networks can capture accurate geometry and render photorealistic images by representing a static scene as an implicit 5D function which outputs view-dependent radiance fields. They use differentiable volumetric rendering, a scene geometry, and the view-dependent radiance that can be encoded into an implicit network using only image supervisions. These components allow the networks to capture high fidelity photometric features, such as reflection and refraction in a differentiable manner unlike the conventional explicit 3D representations.

Given the success of the implicit representations, it is only natural to consider the implicit representation as one of the standard data representations for 3D and for perception. However, these novel representations, which can capture a scene with high fidelity, have not yet been used for perception tasks such as classification and segmentation. One of the main reasons is that there is no large-scale...
Figure 1: Overall illustration of PeRFception dataset with its applications. Our PeRFception dataset convey both visual (spherical harmonic coefficient) and geometric (density, sparse voxel grid) features in one compact format, it can be directly applied to various perception tasks, including 2D classification, 3D classification, and 3D segmentation.

Recent studies have resolved these limitations by adopting explicit sparse voxel grid geometry and basis functions for features. First, to tackle the slow speed, many works propose to use the explicit sparse voxel geometry, which reduces the number of samples along a ray by skipping empty space [10, 11, 12, 14, 15, 22]. Second, instead of using the implicit representations of weights of a network, directly optimizing features [14, 15, 22] assigned to explicit geometry reduces the time to extract features from a network. Lastly, for consistent features between scenes, which is crucial for perception or creating a scene with different objects in NeRF format, Yu et al. [14, 15] show that spherical harmonic coefficients can represent a scene as accurately as NeRFs while preserving consistent and structured features. In particular, Plenoxels [14] satisfy all criteria for data representation which supports fast learning and rendering while maintaining a consistent feature representation for perception and composition of scenes.

In this work, we adopt Plenoxels as the primary format for perception tasks and create both object-centric and scene-centric environments. We mainly use two image datasets and convert them into Plenoxels, the Common Object 3D dataset (Co3D) [23] and ScanNet [24], and name the converted datasets as PeRFception-Co3D and PeRFception-ScanNet, respectively. As the size of Plenoxels can be extremely large, we present a few techniques to compress the size of data and hyperparameters for each setup to maximize the accuracy while minimizing the data size.

We use the PeRFception datasets to train networks for 2D image classification, 3D object classification, and 3D semantic segmentation. We successfully train networks for each perception task, indicating that our datasets effectively convey 2D and 3D information together in a unified format. Moreover,
we show that our representation allows more convenient background augmentation and sophisticated camera-level manipulation.

We summarize our contributions as follows:

- We introduce the first large-scale implicit datasets that can be readily used in downstream perception tasks, including 2D image classification, 3D object classification, and 3D scene semantic segmentation.
- We conduct the first comprehensive study of visual perception tasks that directly process the implicit representation. The extensive experiments show that our datasets effectively convey the information for 2D and 3D perception tasks.
- We provide the ready-to-use pipeline to generate the implicit datasets with fully automatic processes. We expect this automatic process allows generating a very large scale 3D dataset in future.

2 Related Work

2.1 Neural Implicit Representations

Representing a scene using an explicit representation such as voxels, meshes, or point clouds has been the most widely used format, but these are discrete and introduce discretization errors. Neural implicit representations, on the other hand, use a neural network to approximate the geometry or properties of a scene continuously \[1 \, 2 \, 3 \, 25\]. Mildenhall et al. \[5\] showed that neural radiance representation can generate high fidelity renderings with view-dependent illumination effects using multi-layer perceptrons. Many of recent studies extend such implicit representation to dynamic scenes. \[26 \, 27 \, 28 \, 29\], conditional generation \[30 \, 31 \, 32 \, 33\], pose-free \[16 \, 17 \, 19\], and many more. In particular, research on efficient rendering \[10 \, 12 \, 15 \, 22\] has been one of the major directions since volumetric rendering could take minutes. Hedman et al. \[10\] propose to create sparse voxel grids after training to accelerate rendering. Similarly, Plenoctree \[15\] uses an octree data structure instead of sparse voxels for fast rendering. DVGO \[22\] and Plenoxels \[14\] also adopt the sparse voxel structure, and improve both inference and training time. INGP \[12\] proposes a multi-level hash encoding which enables the fast convergence. Recently, TensoRF \[34\] boosts both training and inference time by factorizing 3D radiance fields into lower dimensional vectors or matrices. In this work, we use Plenoxels for our data format since they have explicit geometry and consistent features in form of spherical harmonic coefficients.

2.2 3D Perception Datasets

Over the last decade, many public large-scale datasets of real objects for 3D perception have been published thanks to the advances in commodity sensors. In this section, we cover such large-scale 3D datasets for objects and scenes.

ShapeNet \[35\] and ModelNet \[36\] provide class and part annotations that are from synthetic CAD models. Early object-centric 3D datasets augment image datasets with 3D CAD model annotations. Pascal3D+ \[37\] and Objectron \[38\] contain 3D shapes that are matched with real-world 2D images containing objects; however, 3D models are chosen from approximately aligned 3D models, not precisely reconstructed from the corresponding 2D images. Redwood \[39\] is a large-scale object-centric RGB-D scan video dataset, where only a few categories include 3D models and camera poses. GSO \[40\] holds clear 3D models of real objects with textures, but missing physically rendered images. Co3D \[23\] provides large-scale object-centric videos with camera poses and high-quality point cloud models. They assess quality of reconstructed 3D shapes using human-in-the-loop validation and marked 5625 point clouds as successfully reconstructed. Recently, ABO \[41\] offers a dataset consisting of household object images and high-quality 3D models with 4K texture maps and full-view coverage. Professional artists manually designed its high-quality spatially-varying Bidirectional Reflectance Distribution Functions(BRDFs), indicating that the data generation processes were not fully automatic. We summarize the details of the aforementioned datasets in Table 1.
Table 1: Specs for publicly available 3D datasets. “Real” denotes whether the objects are from real-world images, “Full 3D” for the availability of 3D geometries for all the objects, and “Multi-view” for multi-view images and corresponding real-world catalog images. ▲ is marked when the corresponding information is partially provided.

<table>
<thead>
<tr>
<th>Dataset</th>
<th># Classes</th>
<th># Objects</th>
<th>Real</th>
<th>Full 3D</th>
<th>Multi-view</th>
</tr>
</thead>
<tbody>
<tr>
<td>ShapeNet</td>
<td>55</td>
<td>51K</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ModelNet</td>
<td>40</td>
<td>128K</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>Pascal3D</td>
<td>12</td>
<td>36K</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>Redwood</td>
<td>9</td>
<td>2K</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Objectron</td>
<td>9</td>
<td>15K</td>
<td>✓</td>
<td>✗</td>
<td>▲</td>
</tr>
<tr>
<td>GSO</td>
<td>✓</td>
<td>2K</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>3D-Future</td>
<td>8</td>
<td>2K</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>ABO</td>
<td>98</td>
<td>8K</td>
<td>▲</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Co3D</td>
<td>51</td>
<td>19K</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PeRFception – Co3D</td>
<td>51</td>
<td>19K</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Many scene-centric 3D datasets use depth sensors to scan a section or an entire room and create dense annotations. NYUv2 [43] initially sparked interest for 3D scene understanding, with 464 indoor scans, 1,449 frames of which are annotated with 2D polygons for semantic segmentation. Similarly, SUN RGB-D [44] collected 13,355 RGB-D images, which are densely annotated with 2D polygons and 3D bounding boxes. Both datasets do not include camera parameters, which is essential information for surface reconstruction. SUN3D [45] is comprised of 415 RGB-D indoor video sequences in 254 different spaces; only eight sequences are annotated. Each sequence was captured densely, with a large number of frames collected. 2D-3D-S [46, 47] is an instance-level annotated large-scale indoor scene dataset. It offers diverse modalities of six indoor scenes in RGB images, depth maps, surface normals, 3D meshes, and point clouds. Currently, ScanNet [24] is the most popular large-scale indoor scene dataset that collected instance-level annotated 1,513 scans of RGB-D images and 3D data. Matterport3D [48] contains large-scale RGB-D images annotated with surface and semantic information. In particular, it covers a wide area of 90 building-scale scenes by capturing panoramic views, but it does not provide annotations. Replica [49] is a small but high-quality surface-annotated indoor scene reconstruction database. In this paper, we use two datasets, Co3D and ScanNet, to cover both object- and scene-centric dataset respectively.

We select Co3D for object-centric dataset since it consists of camera-annotated real-world images and has sufficient number of classes. In addition, we use ScanNet since it is the one of the most popular 3D indoor dataset providing adequate number of data with rich annotations.

2.3 3D Perception Models

Unlike perceptions in the 2D domain, where the image is the de facto standard representation, there is no canonical representation for spatial 3D data. Existing explicit representations, such as voxels, meshes, and point clouds, target different aspects of data and have pros and cons. We categorize methods into two groups based on the input representation. Point-based methods [50, 51] directly consume the continuous 3D coordinates of point clouds or meshes using MLP and continuous/graph convolutions. Recent studies have tried to define custom convolution layers [52, 53, 54, 55, 56] upon the continuous coordinate space, or non-local operations [57, 58, 59]. Overall, these methods exhibit fast and simple processing, but it often requires a large computational cost due to neighbor search.

On the other hand, voxel-based methods discretize input coordinates into voxels, which introduces small quantization errors but allows fast neighbor lookup using a data structure. Specifically, recent advances in spatial sparse convolutions [60, 61, 62] that operates on sparse voxels utilize an efficient GPU hash table and require small memory footprint for neighbor search. It has shown successful adoption in many perception tasks, including semantic segmentation [61, 62, 63], object detection [64], representation learning [65], and registration [66, 67, 68, 69]. We use the spatial sparse convnets to create the first perception network on our PeRFception datasets due to its scalability in terms of memory footprint and computational cost.
Table 2: Specs of Co3D and ScanNet, and our PeRFception-Co3D and PeRFception-ScanNet. SH denotes spherical harmonic coefficients, D for densities, and C for diffused color. 3D-BG marks whether the 3D representation includes backgrounds of scenes. We note the number of frames in our proposed datasets as ∞ since our data representation is feasible to render frames from infinitely many camera intrinsics and extrinsics.

<table>
<thead>
<tr>
<th>Dataset</th>
<th># Scenes</th>
<th># Frames</th>
<th>3D Shape</th>
<th>Features</th>
<th>3D-BG</th>
<th>Memory</th>
<th>Memory(Rel)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Co3D</td>
<td>18.6K</td>
<td>1.5M</td>
<td>pcl</td>
<td>C</td>
<td>✗</td>
<td>1.44TB</td>
<td>±0.00%</td>
</tr>
<tr>
<td>PeRFception-Co3D</td>
<td>18.6K</td>
<td>∞</td>
<td>voxel</td>
<td>SH</td>
<td>✓</td>
<td>1.33TB</td>
<td>−6.94%</td>
</tr>
<tr>
<td>ScanNet</td>
<td>1.5K</td>
<td>2.5M</td>
<td>pcl</td>
<td>C</td>
<td>✗</td>
<td>966GB</td>
<td>±0.00%</td>
</tr>
<tr>
<td>PeRFception-ScanNet</td>
<td>1.5K</td>
<td>∞</td>
<td>voxel</td>
<td>SH</td>
<td>✓</td>
<td>35GB</td>
<td>−96.4%</td>
</tr>
</tbody>
</table>

Figure 2: Visualization of a few example data of original datasets and our PeRFception datasets. From the source images and corresponding parameters, we successfully construct PeRFception datasets with both accurate geometry and photorealistic rendering.

3 Preliminary

Yu et al. [14] proposed a novel geometry representation called Plenoxels that combines a sparse voxel grid for coarse geometry with spherical harmonic coefficients for radiance fields. Unlike conventional MLP-based NeRFs [5, 6, 10, 11], which use a single neural network to represent an entire scene, Plenoxels optimize coefficients of spherical harmonic in each non-empty voxel independently, which uses the same differentiable model for volumetric rendering described in NeRF [5] as follows:

\[ \hat{C}(r) = \sum_{i=1}^{N} T_i (1 - \exp(-\sigma_i \delta_i)) c_i, \quad \text{where} \quad T_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j), \]

where \( T_i \) denotes light transmittance of a \( i \)-th sample, \( \sigma \) is opacity, \( c \) is color, and \( \delta \) is distance to the next sample on a ray \( r \). Plenoxels lookup the stored densities and spherical harmonic coefficients in the sparse voxel grids. For scenes with background, Plenoxels also use the lumisphere background representation to render the backgrounds.

We modified the official Plenoxels implementation to set the initial grid properly and hyperparameters. As the size of Plenoxels can be extremely large, we present a few techniques to compress the size of data and hyperparameters. More implementation details are in Section 4.1, Section 4.2, and the appendix.

4 Generating PeRFception dataset

We generate two datasets PeRFception-Co3D and PeRFception-ScanNet to train perception networks.
4.1 PeRFception-Co3D

Co3D [23] is a large-scale object-centric dataset that contains multi-view observations of objects. It contains 18,669 annotated videos with a total 1.5 million of camera-annotated frames and 50 classes from MS-COCO [70], and images crowd-sourced from Amazon Mechanical Turk (AMT). It also provides reconstructed pointclouds, generated by pretrained instance segmentation algorithm [71] and COLMAP [72]. Although reconstructing depth and pointclouds is automatic, its generation step still requires human-in-the-loop validation. When the amount of data increases, this human-in-the-loop validation becomes unsuitable. On the other hand, our 3D dataset generation does not require manual verification since image reconstruction qualities on unseen views are used as a proxy for reconstruction quality. We compare specs of the original Co3D and PeRFception-Co3D in Table 2.

Data Generation. We use the official implementation of Plenoxels [14] with a slight modification to the default configuration. We reduce the resolution of the background lumisphere from 1,024 to 512 and the number of background layers from 64 to 16. For sharper surface, we set the lambda sparsity value to $10^{-10}$, 10 times larger than the default configuration. A voxel grid is initialized with $128^3$ resolution and trained for 25,600 iterations. Then, it is upsampled once to $256^3$ resolution and trained for further 51,200 iterations. Before saving the data, we quantize the trained parameters to unsigned 8-bit integers to minimize for storage except for density values. For each scene, we first filter out defective images and uniformly sample 10% of the images as the test set to assess the rendering quality. The quantitative and qualitative results of the rendering quality are reported in Table 3 and Figure 2. More details are in the appendix.

4.2 PeRFception-ScanNet

ScanNet is a 3D-scanned indoor dataset that captures more than 1.5K indoor scenes with the commercial RGB-D sensors. It provides 3D reconstructed point clouds of scenes with semantic labels containing 20 common object classes, as well as the raw RGB-D frames with corresponding semantic masks and camera parameters. In our experiment, we follow the official data split and report the numbers on the validation split since the test set annotations are not publicly available.

Data Generation. ScanNet videos are captured using handheld cameras where auto-exposure option is held. So, a fair number of frames contain motion blur which could lead to poor scene geometries. We filter out these blury frames by thresholding images with a low variance of Laplacian [73]. Then we uniformly sample a number of frames per scene for training to saturate the CPU memory. Another characteristic of ScanNet is that, unlike object-centric datasets where cameras face inward, the images are captured from inside a room facing outward. These results in fewer images observing the same part of the space, which results in poor reconstruction of Plenoxel’s geometry on ScanNet dataset. Specifically, the Plenoxel reconstruction artificially creates an excessive number of voxels in the empty space (i.e. floaters) to minimize the image reconstruction loss.

Instead, to supply an additional geometric prior to Plenoxel training, we initialize the voxel grid using the unprojected depth maps provided in ScanNet rather than starting from the dense voxel grid. However, since the provided depth maps of ScanNet are contaminated with noisy observations, we incorporate the connected component analysis to filter out the disconnected outlier points in the unprojected point clouds. This leads to stable and more accurate reconstruction and does not excessively generate floaters to minimize the rendering loss. The resulting PeRFception-ScanNet dataset occupies only 35GB in disk whereas the original video streams of ScanNet requires about 966GB disk space. This is a significant compression rate (96.4%), which emphasizes the accessibility of our representation as a dataset. Detailed dataset specs of the original ScanNet and PeRFception-ScanNet are reported in Table 2. We report the rendering quality on Table 3 and visualize the qualitative novel view renderings on Figure 2.

---

1The LPIPS metric sometimes generates “nan” although their visual qualities are great enough. Only 0.01% of scenes are noted as failure.
Table 3: Overall rendering qualities of PeRFception-Co3D and PeRFception-ScanNet on test set. For class-wise rendering scores are reported in the appendix.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>PSNR(↑)</th>
<th>SSIM(↑)</th>
<th>LPIPS(↓)</th>
<th>PSNR &gt; 15 (%)</th>
<th>PSNR &gt; 20 (%)</th>
<th>PSNR &gt; 25 (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeRFception-Co3D</td>
<td>28.82</td>
<td>0.8564</td>
<td>0.3451</td>
<td>99.8%</td>
<td>98.2%</td>
<td>87.3%</td>
</tr>
<tr>
<td>PeRFception-ScanNet</td>
<td>23.42</td>
<td>0.7470</td>
<td>0.4637</td>
<td>99.7%</td>
<td>73.9%</td>
<td>40.4%</td>
</tr>
</tbody>
</table>

5 Experiment

We benchmark popular 2D image classification, 3D object classification, and 3D segmentation networks to demonstrate that our unified data format can be used for various perception tasks.

5.1 Classification on PeRFception-Co3D dataset

Co3D provides multi-view images of objects and 51 class labels for classification. We use the same class labels for classification of PeRFception-Co3D dataset. We adopt a few classification models for our dataset for both 2D [74] and 3D classification [62]. We split the dataset into the train, validation, and test set by scenes since the original Co3D does not provide such splits. We use 10% of the scenes for validation set and 10% for test set in each class. We use the same splits for 2D and 3D classification.

5.1.1 Implementation Details

All the 2D classification models are trained with the cross-entropy loss with the weight decay factor $10^{-4}$. Following the recommendations from [75], we utilize the label smoothing with $\epsilon = 0.005$, remove bias decay, and initialize weights of the batch normalization layers on the residual connections to 0. We use the SGD optimizer with momentum 0.9 and trained for 500 epochs with a batch size 64. For 50 epochs, we linearly warmed up the learning rate from 0 to 0.1 and decayed it using the cosine annealing scheduler. It takes up to a day for training using a single RTX 3090 GPU. We have benchmarked variants of ResNet (ResNet18, ResNet34, ResNet50, ResNet101, ResNet152), ResNext [76] (ResNext50, ResNext101), and WideResNet [77] (WideResNet50, WideResNet101) networks.

For 3D classification, we train 3D version of ResNets [74] with varying depths that are implemented with spatially sparse convolutional layers [61, 62]. These networks directly take sparse voxels from Plenoxels as input. The Plenoxels consist of two components: coordinates of sparse voxels and their features (spherical harmonic coefficients and density values). To demonstrate the efficacy of such features in perception task, we train the networks by providing different input features. We use the SGD optimizer and set the initial learning rate as 0.1 for all experiments and decay it with the cosine annealing scheduler for 100K iterations with batch size 16 on a single RTX 3090 GPU. We augment the input data with both geometric augmentation (random rotation, coordinate dropout, random horizontal flip, coordinate uniform translation, and random scaling) and feature-level augmentation, random feature jittering. Further implementation details are in the appendix.

Background Augmentation. Plenoxels use both sparse voxels and lumispheres to render foregrounds and backgrounds respectively. In other words, we can render each of them separately or manipulate them to create various augmentations. Specifically, we create a novel augmentation that substitutes the background in a scene with backgrounds from other scenes while preserving the foreground object. We describe the composition of foregrounds and randomly selected backgrounds in the appendix. In addition, we visualize several background augmentation examples in the appendix.

5.1.2 2D Classification on PeRFception-Co3D

We train both the scratch and ImageNet [78] pretrained version of ResNet [74] variants on the original Co3D and PeRFception-Co3D to show that PeRFception-Co3D contains the same information as the

Acc@1 (%)  
82  84  86  88  90

Train: Co3D / Test: Co3D  
Train: Co3D / Test: PeRFception-Co3D  
Train: PeRFception-Co3D (w/o bkgd) / Test: Co3D  
Train: PeRFception-Co3D (w/o bkgd) / Test: PeRFception-Co3D  
Train: PeRFception-Co3D (w/ bkgd) / Test: Co3D  
Train: PeRFception-Co3D (w/ bkgd) / Test: PeRFception-Co3D

Figure 3: 2D classification accuracies (Acc@1) of the ResNet models trained either on Co3D or PeRFception–Co3D and evaluated either on Co3D or PeRFception–Co3D. * denotes the ImageNet[78] pretrained network. The models trained on PeRFception-Co3D dataset perform well on both Co3D test dataset and PeRFception-Co3D test dataset. Furthermore, the background augmentation of PeRFception-Co3D dataset improves the 2D classification performance. The score table is in the appendix.

3DResNet14  3DResNet18  3DResNet34  3DResNet50  3DResNet101

Acc@1 (%)  
75  80  85  90  95

None  D  SH  D + SH

Figure 4: 3D classification performance of 3D ResNet [62] models on our PeRFception-Co3D. We visualize Acc@1 (Left) and Acc@5 (Right) score for each model and input features. "None" denotes the case where 3D classification D denotes the density and SH denotes spherical harmonic coefficients.

5.1.3 3D classification on PeRFception-Co3D

PeRFception-Co3D provides a novel 3D data representation which we can directly feed into a network without explicit rendering. We train spatially sparse 3D networks on PeRFception-Co3D and visualize the 3D classification accuracy on Figure 4. For each classification model, we utilize four types of input features that are in our Plenoxels representation: ones(None), densities(D), spherical harmonic coefficients(SH), and concatenation of spherical harmonic coefficients and densities (SH + D). One interesting observation is that using either density values or spherical harmonic coefficients as features improves performance much better. We conjecture this is because the density values provide information about where the model should focus more, and the spherical harmonic coefficients explicitly encode visual features.
5.2 Semantic Segmentation on PeRFception-ScanNet dataset

To further verify the fine-grained perception on the large-scale implicit data, we create and evaluate 3D semantic segmentation networks on our scene-centric PeRFception-ScanNet dataset. We assign semantic labels to each voxel by aligning the reconstructed PeRFception-ScanNet data with the provided ground truth point cloud data. Then, for each voxel of PeRFception-ScanNet data, we find the nearest point in the point cloud and when the distance is smaller than the predefined threshold (5cm for our experiments), we assign the class label of the nearest point to the voxel. Otherwise, we set the voxel label to IGNORE_CLASS.

Similar to 3D classification experiments, we use spatially sparse convolutional networks for prediction, but we use U-shaped convnets with varying depth and width for segmentation. For all networks, we trained for 60K iterations, with batch size 8, SGD optimizer with initial learning rate 0.1, and cosine annealing scheduler. We train each network with different input features as same with the 3D classification in Sec 5.1 to analyze the effect of the plenoptic features to the 3D semantic segmentation task. We apply geometric augmentation (random rotation, random crop, random affine transform, coordinate dropout, random horizontal flip, random translation, elastic distortion), and feature-level augmentation (random feature jittering).

We use the standard experimental settings following [62], and report mean Intersection over Union (mIoU), mean per-point accuracy (mAcc) on the validation split in Figure 5 and scenewise statistics are in the appendix. We achieve up to 62.77% mIoU and 72.36% mAcc on the validation split of PeRFception-ScanNet, which shows that our PeRFception-ScanNet dataset has accurate geometry for networks to learn semantic of each object class. Consistent with 3D classification, the networks trained with spherical harmonic coefficients and density as input feature exhibit higher segmentation accuracy. This results indicate that the spherical harmonic coefficients and density values provide additional cues for fine-grained geometric perception as well. Qualitative results of the segmentation results are provided in the appendix.

6 Conclusion

In this work, we present the first perception networks for an implicit representation and conduct the comprehensive study of various visual perception tasks. To this end, we created two large-scale implicit datasets, namely PeRFception-Co3D and PeRFception-ScanNet, that cover object-centric and scene-centric environments, respectively. Extensive experiments with diverse perception scenarios, including 2D image classification, 3D object classification, and 3D scene semantic segmentation, show that our datasets effectively convey the same information for both 2D and 3D in a unified and compressed data format. This data format allows eliminating the need to separately store different data formats, 2D images and 3D shapes. Consequently, the required disk space for storage is reduced and the unified data format includes richer features. Furthermore, we propose a novel image augmentation method that was infeasible in image datasets. We expect our fully automatic pipeline should be a great candidate for establishing equally large datasets on 3D to tremendously large 2D image datasets, potentially enabling larger models to be trained.
Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] Refer to the appendix on the supplementary materials.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] Refer to the appendix on the supplementary materials.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]

3. If you ran experiments (e.g. for benchmarks)...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The code and data will be publicly available and all the instructions will be provided.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section4 and appendix for details.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Full score tables are in the appendix.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See the appendix.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] Our source datasets (Co3D and ScanNet) and perception models are all cited.
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you’re using/curating? [Yes]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes]

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]
References


[40] GoogleResearch. Google scanned objects, August.


