PRIF: Primary Ray-based Implicit Function

Brandon Y. Feng[1], Yinda Zhang[2], Danhang Tang[2], Ruofei Du[2], and Amitabh Varshney[1]

[1] University of Maryland, College Park
[2] Google Research

Abstract. We introduce a new implicit shape representation called Primary Ray-based Implicit Function (PRIF). In contrast to most existing approaches based on the signed distance function (SDF) which handles spatial locations, our representation operates on oriented rays. Specifically, PRIF is formulated to directly produce the surface hit point of a given input ray, without the expensive sphere-tracing operations, hence enabling efficient shape extraction and differentiable rendering. We demonstrate that neural networks trained to encode PRIF achieve successes in various tasks including single shape representation, category-wise shape generation, shape completion from sparse or noisy observations, inverse rendering for camera pose estimation, and neural rendering with color.

1 Introduction

Learning an accurate and efficient geometric representation of a 3D object is an important problem for computer graphics, computer vision, and robotics. Recent advances in machine learning have inspired a growing trend of implicit neural shape representations, where a neural network learns to predict the signed distance function (SDF) for an arbitrary location in the 3D space. Moreover, in addition to the 3D location (x, y, z), the neural SDF network may take in a latent vector that describes the object identity, thus enabling the generative modeling of multiple objects. Such an implicit neural representation (INR) not only produces fine-grained geometry, but also enables a plethora of applications [28], *e.g.*, shape completion, pose estimation, via a differentiable rendering based optimization.

However, rendering and extracting the shape from a trained neural SDF network are computationally expensive and often limited to watertight shapes. The direct approach to rendering from SDF requires sphere tracing, which needs access to the SDF values at multiple locations along each pixel ray [21]. The indirect approach computes and stores the SDF values at predefined 3D grid points, from which the shape can be rendered with sphere-tracing or extracted as a polygon mesh through Marching Cubes [30]. Both cases demand a large number of network evaluations since they require sampling SDF values at locations far away from the surface. Moreover, the shape quality is ultimately constrained by the converging criteria of sphere tracing or the resolution of the 3D grid.

In this work, we present a novel geometric representation that is efficient, accurate, and innately compatible for downstream tasks involving reconstruction and rendering. We break away from the conventional point-wise implicit

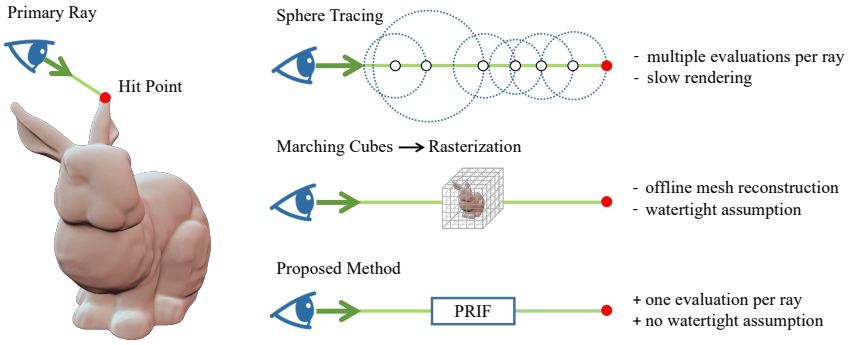


Fig. 1: **Overview.** Common neural shape representation methods learn the level-set functions implicitly describing the object geometry, such as SDF and OF. Rendering from these implicit networks requires either sphere tracing or rasterizing a separately extracted mesh. Sphere tracing is inefficient due to multiple network evaluations for a single ray. Rasterization induces a separate meshing step (often through Marching Cubes), which hinders end-to-end gradient propagation in differentiable applications, and the shape quality is ultimately restricted by the meshing algorithm's limitation, such as grid resolutions and shape watertightness. Our new representation, PRIF, directly maps each primary ray to its hit point. A network encoding PRIF is more efficient and convenient for rendering, since it requires only one evaluation for each ray, avoids the watertight constraint in conventional methods, and easily enables differentiable rendering.

functions and propose to encode 3D geometry into a novel ray-based implicit function. Specifically, our representation operates on the realm of oriented rays $r = (\mathbf{p}_r, \mathbf{d}_r)$, where $\mathbf{p}_r \in \mathbb{R}^3$ is the ray origin and $\mathbf{d}_r \in \mathbb{S}^2$ is the normalized ray direction. Unlike SDF that only outputs the distance to the nearest but undetermined surface point, we formulate our representation such that its output directly reveals the surface hit point of the input ray. The rays whose surface intersection we care about are specifically known in computer graphics as *primary* rays, whose origins are the rendering viewpoint, as opposed to *secondary* rays that originate from the object surface [1]. Therefore, we name our representation as Primary Ray-based Implicit Function (PRIF). In effect, a neural network trained to encode PRIF represents the manifestation of the object's geometry at any viewpoint that it is observed.

Modeling the object from the ray-based perspective, rather than 3D-point-based, has huge implications for efficient application in any task that involves rendering the object from a viewpoint. In Section 4, we show that PRIF outperforms common functions such as SDF and OF, and we further demonstrate successful applications of PRIF to various tasks using neural networks.

Properly formulating the rays is nontrivial. While it may be intuitive to let PRIF output the distance from the ray origin $\mathbf{p}_r$ to the hit point $\mathbf{h}_r$, this formulation leads to ray aliasing - *i.e.*, if we move $\mathbf{p}_r$ along the ray direction $\mathbf{d}_r$, the

distance to the hit point $\mathbf{h}_r$ needs to change. However, the actual surface intersection point would not change to translation along the view direction. Therefore, it is undesirable to have such a potential variance as it adds unnecessary complexity to the network output.

To avoid the aliasing, we reparametrize the ray r by replacing the view position $\mathbf{p}_r$ with perpendicular foot $\mathbf{f}_r$ from the coordinate origin $\boldsymbol{O}$ to r. Wherever we move $\mathbf{p}_r$ along the ray direction $\mathbf{d}_r$, the perpendicular foot $\mathbf{f}_r$ stays the same. Furthermore, we can easily define the surface hit point $\mathbf{h}_r$ by a single scalar value: its distance from $\mathbf{f}_r$. Thus, we formulate PRIF so that it outputs this distance for an input ray. More details are in Section 3.

In summary, our main contributions are the followings:

- We present PRIF, a novel formulation for geometric representation using ray-based neural networks.
- We show that PRIF outperforms common level-set-based methods in shape representation accuracy and extraction speed.
- We demonstrate that PRIF enables generative modeling, shape completion, and camera pose estimation.

2 Related Work

We discuss prior art on neural representations for 3D shapes and scene rays.

2.1 3D Shape Representations

Functional Representations. Traditional 3D shape representations include polygon meshes, point clouds, and voxels. In recent years, as deep neural networks achieve remarkable success on various vision-related tasks, there has been a growing interest in developing implicit neural representations (INRs) of 3D shapes. Following seminal works [9, 32, 36] showing successful applications of neural network to encode 3D shapes, many methods have been introduced to solve various vision and graphics tasks using INRs of 3D shapes [3, 4, 7, 10, 11, 15, 17, 24, 28, 29, 35, 38, 40, 42, 43, 49, 52, 53, 56].

INRs usually use the multilayer perceptron (MLP) architecture to encode geometric information of a 3D shape by learning the mapping from a given 3D spatial point and a scalar value. Typically, the output scalar value denotes either the occupancy at the given point, or the signed distance from the given point to the nearest point on the shape. On one hand, networks that are trained to encode the occupancy function [32] (OF) essentially learns the binary classification problem, where the output equals 0 if the point is empty, and equals 1 if occupied. Therefore, the decision boundary where the network predictions equal to 0.5 represents the surface of the encoded shape. On the other hand, for networks trained to encode the signed distance function [9,36] (SDF), the surface is represented by the decision boundary where the network predictions equal to 0. A 3D surface determined in such fashions is also known as an isosurface, which

is a level set of a continuous 3D function. In practice, however, obtaining 3D meshes from these isosurfaces extracted from INRs still requires an additional meshing step often through the Marching Cubes algorithm.

In this paper, we introduce a new shape representation which is not determined by an isosurface. Instead, our representation encodes a 3D shape by learning the PRIF associated with the shape. Outputs of such a function directly correspond to points on the surface, and the shape can be extracted without an additional meshing step that could inject inaccuracies to the final representation.

Global *v.s.* Local Representations The initial works of INRs for 3D shapes inspire many techniques to improve its the rendering efficiency [28, 55] and representation quality [44, 48]. Among many techniques, a common thread is the idea of spatial partitions which manifest in two main approaches. One approach divides the surfaces of shapes into different local patches, reducing the difficulty of globally fitting a complex surface with a single network [5, 19, 20, 23, 37, 50]. Another approach divides the 3D volume into small local regions (often based on the spatial octree structure), and then train INRs to encode the geometric information within each local region [8, 14, 27, 31, 34, 41, 47, 54].

In this paper, we only focus on global representations, where a shape is represented by a single network without any spatial partitions. While the aforementioned works largely focus on improving the performance of INRs based on SDF, our main contribution is a new functional representation in place of isosurface-based representations like SDF. Nonetheless, the idea of spatial partitions explored in previous works has the potential to improve the performance of our currently global representation, and we see local specializations as a promising future direction to explore.

2.2 Ray-based Neural Networks

Our method is closely related to an emerging concept called Ray-based Neural Networks. Rays are a common construct from computer graphics, and can be denoted by a tuple consisting of a 3D point (ray origin) and a 3D direction (ray direction). Due to their closeness to rendering 3D scenes, rays have become a central component in the problem formulation of many recent works using neural networks to model 3D scenes [33, 46]. Feng *et al.* [18] and Attal *et al.* [2] have further demonstrated that for front-facing light field scenes, MLPs can be trained to accurately map camera rays to their observed colors in highly detailed real-world scenes, but their networks only consider rays restricted within two parallel planes. Most recently, Sitzmann *et al.* [45] successfully train a MLP to encode the observed color of unrestricted rays with arbitrary origins and directions, and the key is to parametrize rays using the Plucker coordinates [22].

We similarly adopt rays as input to the neural network. However, instead of the Plucker parametrization which replaces the ray position by its moment vector about the origin, we replace the moment vector by the perpendicular foot between the ray and the origin. As discussed in Sec. 3, our formulation allows the network to simply produce an affine transformation to its input.

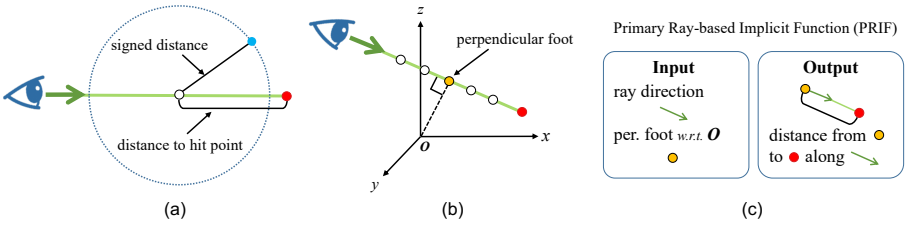


Fig. 2: **Representation.** (a) Signed distance at a sampling position (white) reveals the sphere (blue dots) where its nearest surface point (blue) exists, but that may be irrelevant when we really want to know the hit point (red) *along a specific direction*. Thus, multiple samples are often required. (b) Our new representation uses only one sample (yellow) along the ray to obtain its surface hit point. The sampling position is the perpendicular foot between the given ray and the coordinate system's origin $\boldsymbol{O}$. (c) The proposed function takes in the ray's direction and its sampling point, and returns the distance from that point to the actual surface hit point. We train neural networks to encode geometry by learning this new function, and we demonstrate the capability of our trained networks in various tasks involving shape representation and rendering.

3 Method

In this section, we introduce our new representation for 3D shapes. We first discuss the motivation behind ray parametrization used in prior works, and then we present our new formulation specifically designed for shape representations.

3.1 Background

Sitzmann *et al.* [45] encode light fields by training an MLP Φ_ϕ with parameters ϕ on a set of observed rays to learn the mapping of $r = (\mathbf{p}_r, \mathbf{d}_r) \rightarrow c_r$, where c_r denotes the color of the observed radiance. However, naively concatenating $\mathbf{p}_r$ and $\mathbf{d}_r$ as input to the network is not ideal due to ray aliasing. If we move the position of a ray r along the ray direction $\mathbf{d}_r$, we would obtain a ray $r' = (\mathbf{p}_{r'}, \mathbf{d}_r)$ that is really an aliased version of r. There is no guarantee that the trained network would produce the same output for different aliases r' of the ray r.

To resolve ray aliasing, Sitzmann *et al.* [45] reparametrize the ray $r = (\mathbf{p}_r, \mathbf{d}_r)$ into Plucker coordinates as $r = (\mathbf{m}_r, \mathbf{d}_r)$, where $\mathbf{m}_r = \mathbf{p}_r \times \mathbf{d}_r$ is also known as the moment vector of $\mathbf{p}_r$ about the origin $\boldsymbol{O}$. Plucker coordinates represent all oriented rays in space without singularity or special cases, and they are invariant to changes in the ray position along the ray direction. To better understand this property, consider moving the ray position to some other point $\mathbf{p}'_r$ at a fixed ray

direction $\mathbf{d}_r$. Then, for a certain $\lambda \in \mathbb{R}$, $\mathbf{p}'_r = \mathbf{p}_r - \lambda \mathbf{d}_r$, and

$$\begin{aligned} \mathbf{p}'_r \times \mathbf{d}_r &= (\mathbf{p}_r - \lambda \mathbf{d}_r) \times \mathbf{d}_r \\ &= \mathbf{p}_r \times \mathbf{d}_r - \lambda \mathbf{0} \\ &= \mathbf{p}_r \times \mathbf{d}_r. \end{aligned} \tag{1}$$

Therefore, $\mathbf{m}_r = \mathbf{p}_r \times \mathbf{d}_r$ is invariant to any change of λ along $\mathbf{d}_r$.

3.2 Describing Geometry with Perpendicular Foot

Our goal is to train a neural network to encode the mapping from a ray to its hit point on the 3D shape's surface. Although replacing ray position with the moment vector $\mathbf{m}_r$ allows Sitzmann et al. [45] to train networks to encode light fields, it is hard to geometrically relate a ray's moment vector to its surface hit point. We propose an alternative way to parameterize a ray as input to the network. which has an intuitive and intrinsic relationship to its hit point.

Specifically, we consider the perpendicular foot $\mathbf{f}_r$ between the ray r and the coordinate system's origin $\boldsymbol{O}$, which may be computed by

$$\mathbf{f}_r = \mathbf{d}_r \times (\mathbf{p}_r \times \mathbf{d}_r). \tag{2}$$

Similar to $\mathbf{m}_r$ in Plucker coordinates, $\mathbf{f}_r$ is also invariant to changing the ray position along the ray direction. Specifically, let $\mathbf{p}'_r$ be the translated ray position defined as before, we can then write $\mathbf{p}_{r'} = \mathbf{p}_r - \lambda \mathbf{d}_r$, and

$$\begin{aligned} \mathbf{f}_{r'} &= \mathbf{d}_r \times (\mathbf{p}_{r'} \times \mathbf{d}_r) \\ &= \mathbf{d}_r \times ((\mathbf{p}_r - \lambda \mathbf{d}_r) \times \mathbf{d}_r) \\ &= \mathbf{d}_r \times (\mathbf{p}_r \times \mathbf{d}_r - \lambda \mathbf{0}) \\ &= \mathbf{d}_r \times (\mathbf{p}_r \times \mathbf{d}_r) \\ &= \mathbf{f}_r \end{aligned} \tag{3}$$

In other words, $\mathbf{f}_r$ is invariant to moving $\mathbf{p}_r$ along the direction $\mathbf{d}_r$.

As a result, we can represent any ray $r = (\mathbf{f}_r, \mathbf{d}_r)$, and we can further establish the following relationship for each ray r:

$$\mathbf{h}_r = s_r \cdot \mathbf{d}_r + \mathbf{f}_r, \tag{4}$$

where $s_r \in \mathbb{R}$ denotes the signed displacement between the ray's hit point $\mathbf{h}_r$ and its perpendicular foot $\mathbf{f}_r$ *w.r.t.* the world origin $\boldsymbol{O}$.

To encode a 3D shape, we propose the mapping function $(\mathbf{f}_r, \mathbf{d}_r) \rightarrow s_r$, which we call Primary Ray-based Implicit Function (PRIF). Different than SDF or OF, which implicitly encodes the geometry through the distance from any given point to its nearest surface point *in any direction*, PRIF operates on oriented points with *a specific ray direction*.

In practice, we train an MLP to learn

$$\Phi(\mathbf{f}_r, \mathbf{d}_r) = s_r. \tag{5}$$

With this new representation, we are able to represent the surface hit point of a ray with a single value. In effect, our objective is equivalent to finding a simple affine transformation $f(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{b}$, with the input $\mathbf{x} = \mathbf{d}_r$, $\mathbf{A} = s_r I_3$, and $\mathbf{b} = \mathbf{f}_r$. We also avoid a major limitation in previous sphere-tracing-based methods, which is having to sample multiple points and perform multiple network evaluations to obtain a hit point.

3.3 Background Mask

For 3D functions like SDF or OF, every 3D point has a well-established scalar value denoting the distance to surface or the occupancy at that point. In contrast, our function would take in rays that never intersect with the shape and therefore do not even have a hit point.

To address this issue, we let the network additionally produce

$$\Phi(\mathbf{f}_r, \mathbf{d}_r) = a_r, \tag{6}$$

where $a_r \in [0, 1]$ denotes the probability in which the ray r hits the foreground. We compute the cross-entropy loss

$$\mathcal{L}_a = \sum_r -a_r^{gt} \log(a_r) - \left(1 - a_r^{gt}\right) \log(1 - a_r), \tag{7}$$

where $a_r^{gt} = 0$ for background rays and $a_r^{gt} = 1$ for foreground rays.

For the signed displacement s_r, we supervise the learning by computing its absolute difference to the ground truth as $\mathcal{L}_s = \sum_{r \in \mathcal{F}} \|s_r - s_r^{gt}\|$, given the set of foreground rays $\mathcal{F}$. As a result, the total loss function to train our network is $\mathcal{L} = \mathcal{L}_a + \mathcal{L}_s$ and is averaged among all rays in a training batch

3.4 Outlier Points Removal

In rare cases where sharp surface discontinuities exist between two neighboring rays, the network would likely produce continuous predictions when interpolating between those two rays, resulting in undesirable outlier points. Fortunately, since our network is fully differentiable, for each prediction s_r we can compute its gradient with respect to the changes in ray position. We discard all predictions that satisfy the threshold: $\left\| \frac{\partial s_r}{\partial \mathbf{p}_r} \right\| \geq \delta$. In our experiments, δ is set equal to 5.

4 Experiments

In this section, we first verify the efficacy of neural PRIF for shape representation. Then, we show verious applications achieved by using PRIF as the underlying neural shape representation. Note that the scope of our experiments is to compare these functional shape representations *encoded by neural networks*.

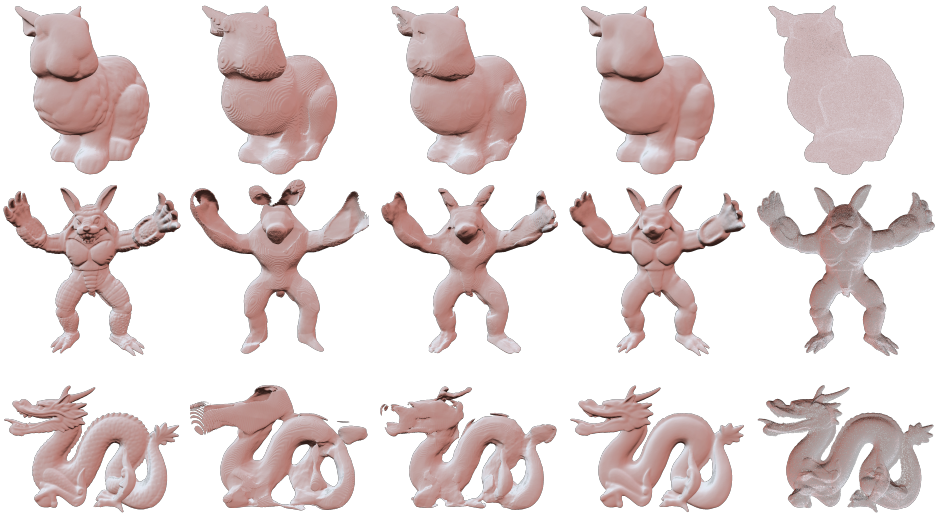


Fig. 3: **Single Shape.** To examine the representation capability of PRIF, we train networks with the same architecture to encode the occupancy function (OF), signed distance function (SDF), and our proposed function (PRIF). Here, we visualize the extracted shapes from the trained neural representations. For OF and SDF, we follow the convention and extract the shapes by Marching Cubes. Our method directly outputs hit points, and we also apply the point-based Screened Poisson algorithm and present the resulting mesh for comparison.

4.1 Single Shape Representation

We select five models (*Armadillo*, *Bunny*, *Buddha*, *Dragon*, *Lucy*) from the Stanford 3D Scanning Repository [13, 51] and train a neural network to fit the PRIF for each 3D model. We also fit the signed distance function (SDF) and the 3D occupancy function (OF) for these models. For a fair comparison between these functions, we adopt the same network architecture as Park et al. [36], containing eight layers with 512 hidden dimensions and ReLU activation.

Compared to SDF and OF which are trained on individual spatial points (x, y, z), PRIF requires an inherently different strategy to generate the training data since it takes in individual rays. In our experiments, for each 3D model, we select 50 virtual camera locations oriented towards the origin, and we capture 200×200 rays at each location. For SDF and OF, we follow Park et al. [36] and sample $500,000$ points with more aggressive sampling near the object surface.

For all three functions, we train the neural representation for 100 epochs with the learning rate initialized as 10^{-4} and decayed to 10^{-7} with a cosine annealing strategy. After training, the SDF- and OF-based shape representation are obtained by evaluating the neural network at uniform 256^3 volume grid

Method	Armadillo	Bunny	Buddha	Dragon	Lucy
SDF	1.905|1.260	1.717|1.147	6.119|2.258	5.184|1.946	3.387|1.417
OF	4.805|1.624	1.704|1.133	17.279|3.113	19.577|3.014	3.396|1.427
PRIF	**0.978|0.706**	**1.169|0.835**	**1.443|0.821**	**1.586|0.913**	**0.846|0.519**

Table 1: Quantitative results on single shape representation on 3D models from the Stanford 3D Scanning Repository. The left and right numbers represent the mean and median CD (multiplied by 10^{-4}). After extracting shapes from each representation, 30,000 points are sampled for evaluation.

and extracted using Marching Cubes. On the other hand, with our PRIF-based representation, we can evaluate the neural network at those virtual camera rays in the training set and directly obtain a dense set of surface points.

To evaluate the representation quality, for SDF- and OF-based representations, we first follow conventions [36] and sample $8,192$ points on the mesh extracted with Marching Cubes. For the point set produced by the PRIF-based representation, we apply the point-based meshing algorithm Screened Poisson [25] in MeshLab [12] and then sample $8,192$ points from the reconstructed mesh. Then, we obtain $8,192$ Poisson-disk samples of the ground truth surface points from the original 3D model. Finally, we compute the mean and median Chamfer Distance (CD) between the ground truth point set and point sets sampled from those three representations. In Table 1 and Fig. 3, we provide quantitative and qualitative comparisons among the three representations. PRIF significantly outperform SDF and OF in accurately preserving the fine details of the 3D shapes.

Method	Car	Chair	Table	Plane	Lamp	Sofa
SDF	2.315|0.495	2.649|0.407	7.213|0.441	2.728|0.170	32.571|3.475	6.427|0.218
OF	2.820|0.587	4.589|0.835	6.427|1.296	2.999|0.169	143.377|145.753	12.672|0.184
PRIF	**1.961|0.347**	**0.982|0.267**	**4.532|0.315**	**0.389|0.125**	**3.276|0.534**	**1.236|0.222**

Table 2: Quantitative results on generative representation on unseen 3D shapes of six categories from ShapeNetCore. The left and right numbers represent the mean and median CD ($\times 10^{-3}$) averaged over the test set. After extracting shapes from each representation, 30,000 points are sampled for evaluation.

4.2 Shape Generation

Having established the representation power of PRIF for single shapes, we now examine its capability for generative shape modeling. We adopt the strategy

Fig. 4: **Shape Generation.** We further examine the representation power on generative modeling over multiple shapes. After training the neural networks to learn shapes from a category, we use auto-decoding to find latent vectors that best represents novel shapes within this category. Here, we visualize the extracted shapes from networks trained to encode the three different functions. Note that *Points* are the immediate output of the PRIF networks.

from Park *et al.* [36] and enable multi-shape generation from a single network by concatenating a latent code for each object with the original network input.

In line with Park *et al.* [36], we select five categories of 3D objects from ShapeNetCore [6]. Within each category, we select the first 180 objects as the training set and the next 20 as the testing set. We maintain the same network architecture and training schedule when training neural networks to fit OF, SDF, and PRIF. We evaluate the trained networks on unseen shapes from the test set using auto-decoding [36] and provide quantitative and qualitative comparisons in Table 2 and Fig. 4. The PRIF-based representation extends its success from representing single shapes and remains effective in the generative task.

4.3 Shape Denoising and Completion

As depth sensors have become available on mobile and AR/VR devices, there are an array of applications for persistent geometric reconstructions [16]. Given real-world data of sparse and noisy observations, the capability of generative modeling could enable shape recovery from noisy and incomplete point clouds.

In Fig. 5, we demonstrate the denoising and completion ability of a trained generative PRIF network. Specifically, after training a PRIF network on an

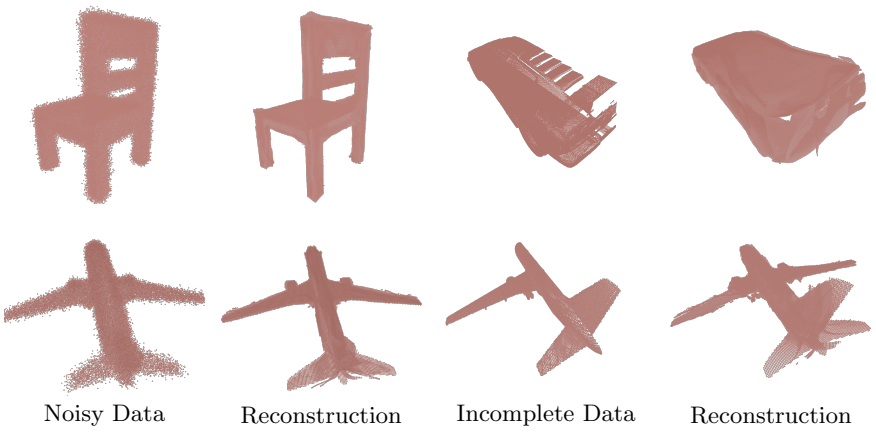


Fig. 5: **Noisy and Incomplete Observations.** Networks trained for shape generation can reconstruct unseen shapes in challenging scenarios where only noisy or incomplete observations are available. Here we visualize the observations and reconstructions as raw point cloud data.

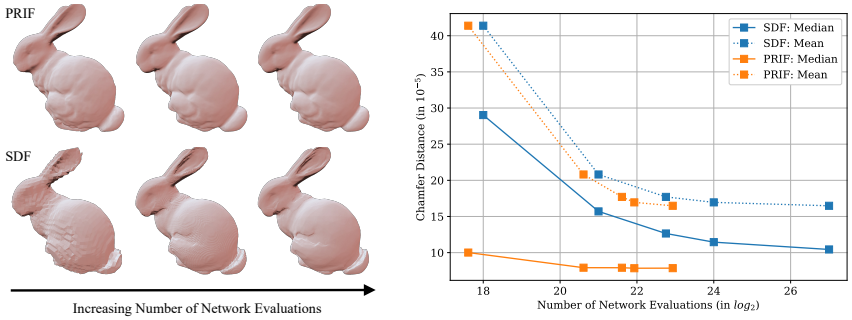


Fig. 6: **Complexity Analysis.** We evaluate the trained SDF network at grid resolutions of 64^3, 128^3, 192^3, 256^3, 512^3 and obtain final meshes by Marching Cubes. We evaluate the trained PRIF network at resolutions of 5×200^2, 10×400^2, 20×400^2, 25×400^2, 50×400^2 (# of Cameras $\times$ Resolution), and we apply Screened Poisson [25] to the output hit points to obtain the final mesh. The reconstructed mesh quality is measured by mean and median CD ($\times10^{-5}$). The PRIF network achieves better quality with fewer evaluations.

object category as described before, we provide the network with an unseen object's point cloud observations, which are either incomplete or contain noise.

4.4 Analysis and Ablations

Complexity Analysis. Shapes are commonly extracted from networks encoding SDF or OF using meshing algorithms like Marching Cubes. This meshing

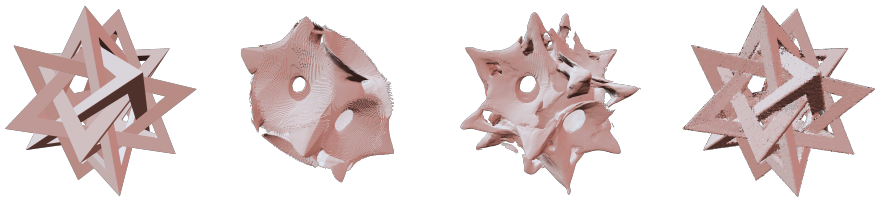
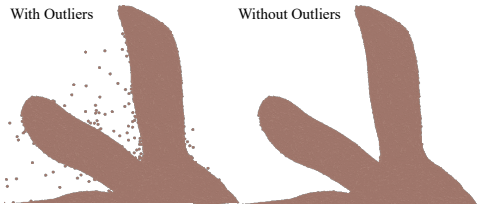


Fig. 7: **Stress Testing.** We test on a Tetrahedron grid that is self-intersecting and non-watertight. We obtain the SDF and OF values with the scanning method [26], and extract the mesh by Marching Cubes. While the level-set representations fail as expected, our method reliably preserves the shape.



Setup	Trained	Novel
Proposed	**12.07**|**7.95**	**20.81**|**12.05**
$\mathbf{f}_r \Rightarrow \mathbf{m}_r$	12.15|8.05	28.38|13.53
$\mathbf{f}_r \Rightarrow \mathbf{p}_r$	1.16|0.77	1924.4|1067.7
w/ Outliers	12.14|8.06	21.32|12.05

Fig. 8 & Table 3: **Ablations.** *Left*: Impact of outlier removal described in Sec. 3. While the outliers have little impact on the quantitative metrics since they are scant, removing them improves visual consistency. *Right*: Mean and median CD ($\times 10^{-5}$) of network prediction on trained rays and novel rays under different setup. Notice that changing $\mathbf{f}_r \Rightarrow \mathbf{p}_r$ causes ray aliasing discussed in Sec. 3, leading the network to overfit on trained rays and collapse on novel rays.

step requires evaluating the network at many 3D grid sampling points, and the grid resolution affects the final quality of the mesh. In our case, we can extract the shapes by rendering the scene from multiple positions and change the number of evaluations by varying the number and resolution of virtual cameras. For a more objective comparison against prior arts that require meshing, we further apply Screened Poisson to mesh our output hit points. In Fig. 6, we analyze the trade-off between the number of network evaluations and the reconstructed mesh quality measured in mean and median CD. Evidently, PRIF produces reconstruction with a better quality with fewer network evaluations.

Stress Testing. Fig. 7 shows an example of encoding a self-intersecting and non-watertight shape, which is expected to be challenging for neural networks trained to encode conventional functional representations like SDF and OF. In contrast, the shape is well preserved by the network trained to encode our proposed functional representation PRIF.

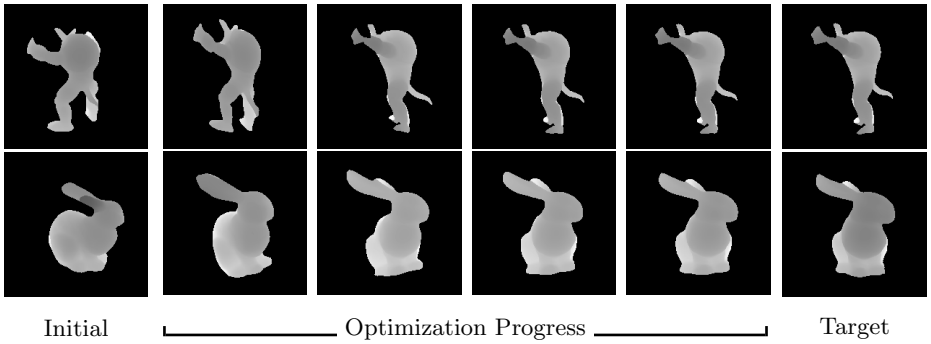


Fig. 9: **Learning Camera Poses.** Starting from the *Initial* camera pose, we optimize the learnable pose parameters based on the difference between PRIF output rendered at the current pose and the PRIF output rendered at an unknown *Target* pose. Here, we render the PRIF output as depth images. PRIF-based rendering successfully facilitates the differentiable optimization progress as the camera pose gradually converges to the correct *Target* pose.

Ablations. We present ablation studies validating the effectiveness of our proposed techniques in Fig. 8 and Table 3. The results verify that: 1) not reparametrizing $r = (\mathbf{p}_r, \mathbf{d}_r)$ leads to severe overfitting on the training data, and 2) replacing the Plucker moment vector $\mathbf{m}_r$ with our proposed $\mathbf{f}_r$ improves the encoding quality, possibly because the network only has to learn a simple affine transform matrix as mentioned in Sec. 3.2.

4.5 Further Applications

Learning Camera Poses. Evaluating the PRIF network on some input camera rays is essentially performing differentiable rendering of the underlying scene represented by the network. With a PRIF network trained on a 3D shape, given a silhouette image at an unknown camera pose, our task is to recover that camera pose based on the silhouette image. In this task, the weights of the PRIF network are frozen, and the only learnable parameters are the camera pose matrix. At each iteration, we render the scene through the PRIF network with camera rays defined by the current estimate of the camera pose, and we adjust the estimate based on the silhouette difference between the observed and rendered images. In Fig. 9, we present the rendered image during the optimization steps. The estimated camera pose gradually converges to the correct solution as the rendered image becomes more similar to the observed image.

Neural Rendering with Color. The PRIF network behaves as a geometry renderer since it effectively returns the surface hit point given a viewing ray. Here we show that the PRIF network can be further extended to rendering color at the surface. We select a 3D model of human from the Renderpeople dataset [39]

Fig. 10: **Incorporating Color Rendering.** After training the PRIF network to encode the geometry (rendered as depth images in the first row), we can further train a second-stage network that takes in the hit point produced by PRIF and produces its corresponding color (rendered in the second row).

and virtually capture the hit points and RGB colors at 25 locations surrounding the model. We then train the PRIF network to fit the geometric shape, similar to the single shape experiments. Finally, we extend the network to produce the observed RGB color given the input ray while fixing its hit point prediction. Fig. 10 shows the rendered results of geometry and appearance of the 3D model.

Limitation. This paper is focused on 3D shape encoding and decoding through hit points of rays from known views. PRIF can render at novel views by sampling new rays, but to guarantee consistent novel view results would likely require further modifications such as multi-view consistency loss or denser training views.

5 Conclusion

We propose PRIF, a new 3D shape representation based on the relationship between a ray and its perpendicular foot with the origin. We demonstrate that neural networks can successfully encode PRIF to achieve accurate shape representations. With this new representation, we avoid multi-sample sphere tracing and obtain the hit point with a single network evaluation. Neural networks trained to encode PRIF inherit such advantages and can represent shapes more accurately than common neural shape representations using the same network architecture. We further extend the neural PRIF networks to enable various downstream tasks including generative shape modeling, shape denoising and completion, camera pose optimization, and color rendering. Promising future directions include using spatial partitions to improve the network accuracy, jointly learning the geometry and view-dependent color directly from images, speeding up network training, and real-time inference for robotic applications.

References

1. Akenine-Moller, T., Haines, E., Hoffman, N.: Real-Time Rendering. AK Peters/CRC Press (2019)
2. Attal, B., Huang, J.B., Zollhoefer, M., Kopf, J., Kim, C.: Learning Neural Light Fields With Ray-Space Embedding Networks. arXiv Preprint arXiv:2112.01523 (2021). https://doi.org/10.48550/arXiv.2112.01523
3. Atzmon, M., Lipman, Y.: SAL: Sign Agnostic Learning of Shapes From Raw Data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2565–2574 (2020). https://doi.org/10.1109/CVPR42600.2020.00264
4. Bhatnagar, B.L., Sminchisescu, C., Theobalt, C., Pons-Moll, G.: Combining Implicit Function Learning and Parametric Models for 3D Human Reconstruction. In: European Conference on Computer Vision. pp. 311–329. Springer, Springer (2020). https://doi.org/10.1007/978-3-030-58536-5_19
5. Chabra, R., Lenssen, J.E., Ilg, E., Schmidt, T., Straub, J., Lovegrove, S., Newcombe, R.: Deep Local Shapes: Learning Local SDF Priors for Detailed 3D Reconstruction. In: European Conference on Computer Vision. pp. 608–625. Springer, Springer (2020). https://doi.org/10.1007/978-3-030-58526-6_36
6. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., Yu, F.: ShapeNet: an Information-Rich 3D Model Repository. Tech. Rep. arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago (2015)
7. Chen, Z., Zhang, Y., Genova, K., Fanello, S., Bouaziz, S., Häne, C., Du, R., Keskin, C., Funkhouser, T., Tang, D.: Multiresolution Deep Implicit Functions for 3D Shape Representation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 13087–13096 (2021). https://doi.org/10.1109/ICCV48922.2021.01284
8. Chen, Z., Tagliasacchi, A., Zhang, H.: Bsp-Net: Generating Compact Meshes Via Binary Space Partitioning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 45–54 (2020). https://doi.org/10.1109/CVPR42600.2020.00012
9. Chen, Z., Zhang, H.: Learning Implicit Fields for Generative Shape Modeling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5939–5948 (2019). https://doi.org/10.1109/CVPR.2019.00609
10. Chibane, J., Alldieck, T., Pons-Moll, G.: Implicit Functions in Feature Space for 3D Shape Reconstruction and Completion. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 6968–6979 (2020). https://doi.org/10.1109/CVPR42600.2020.00700
11. Chibane, J., Alldieck, T., Pons-Moll, G.: Implicit Functions in Feature Space for 3D Shape Reconstruction and Completion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6970–6981 (2020)
12. Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., Ranzuglia, G.: Meshlab: an open-source mesh processing tool. In: Eurographics Italian Chapter Conference. vol. 2008, pp. 129–136. Salerno, Italy (2008)
13. Curless, B., Levoy, M.: A Volumetric Method for Building Complex Models From Range Images. In: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques. pp. 303–312 (1996). https://doi.org/10.1145/237170.237269

14. Deng, B., Genova, K., Yazdani, S., Bouaziz, S., Hinton, G., Tagliasacchi, A.: CvxNet: Learnable Convex Decomposition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 31–44 (2020). https://doi.org/10.1109/CVPR42600.2020.00011
15. Driess, D., Ha, J.S., Toussaint, M., Tedrake, R.: Learning Models As Functionals of Signed-Distance Fields for Manipulation Planning. In: Conference on Robot Learning. pp. 245–255. PMLR (2022). https://doi.org/10.48550/arXiv.2110.00792
16. Du, R., Turner, E., Dzitsiuk, M., Prasso, L., Duarte, I., Dourgarian, J., Afonso, J., Pascoal, J., Gladstone, J., Cruces, N., Izadi, S., Kowdle, A., Tsotsos, K., Kim, D.: DepthLab: Real-Time 3D Interaction With Depth Maps for Mobile Augmented Reality. In: Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology. pp. 829–843. UIST, ACM (Oct 2020). https://doi.org/10.1145/3379337.3415881
17. Duan, Y., Zhu, H., Wang, H., Yi, L., Nevatia, R., Guibas, L.J.: Curriculum DeepSDF. In: European Conference on Computer Vision. pp. 51–67. Springer, Springer (2020). https://doi.org/10.1007/978-3-030-58598-3_4
18. Feng, B.Y., Varshney, A.: SIGNET: Efficient Neural Representation for Light Fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 14224–14233 (2021). https://doi.org/10.1109/ICCV48922.2021.01396
19. Genova, K., Cole, F., Sud, A., Sarna, A., Funkhouser, T.: Local Deep Implicit Functions for 3D Shape. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4857–4866 (2020). https://doi.org/10.1109/CVPR42600.2020.00491
20. Genova, K., Cole, F., Vlasic, D., Sarna, A., Freeman, W.T., Funkhouser, T.: Learning Shape Templates With Structured Implicit Functions. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7154–7164 (2019). https://doi.org/10.1109/ICCV.2019.00725
21. Hart, J.C.: Sphere Tracing: a Geometric Method for the Antialiased Ray Tracing of Implicit Surfaces. The Visual Computer **12**(10), 527–545 (1996)
22. Jia, Y.B.: Plucker Coordinates for Lines in the Space. https://faculty.sites.iastate.edu/jia/files/inline-files/plucker-coordinates.pdf (2020)
23. Jiang, C., Sud, A., Makadia, A., Huang, J., Niessner, M., Funkhouser, T., et al.: Local Implicit Grid Representations for 3D Scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6001–6010 (2020). https://doi.org/10.1109/CVPR42600.2020.00604
24. Jiang, Y., Ji, D., Han, Z., Zwicker, M.: SDFDiff: Differentiable Rendering of Signed Distance Fields for 3D Shape Optimization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1251–1261 (2020). https://doi.org/10.1109/CVPR42600.2020.00133
25. Kazhdan, M., Hoppe, H.: Screened Poisson Surface Reconstruction. ACM Transactions on Graphics (ToG) **32**(3), 1–13 (2013). https://doi.org/10.1145/2487228.2487237
26. Kleineberg, M., Fey, M., Weichert, F.: Adversarial generation of continuous implicit shape representations. In: Wilkie, A., Banterle, F. (eds.) 41st Annual Conference of the European Association for Computer Graphics, Eurographics 2020 - Short Papers, Norrköping, Sweden, May 25-29, 2020 [online only]. pp. 41–44. Eurographics Association (2020). https://doi.org/10.2312/egs.20201013
27. Lindell, D.B., Van Veen, D., Park, J.J., Wetzstein, G.: BACON: Band-Limited Coordinate Networks for Multiscale Scene Representation. arXiv Preprint arXiv:2112.04645 (2021). https://doi.org/10.48550/arXiv.2112.04645

28. Liu, S., Zhang, Y., Peng, S., Shi, B., Pollefeys, M., Cui, Z.: DIST: Rendering Deep Implicit Signed Distance Function With Differentiable Sphere Tracing. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE (Jun 2020). https://doi.org/10.1109/CVPR42600.2020.00209
29. Liu, S., Saito, S., Chen, W., Li, H.: Learning to Infer Implicit Surfaces Without 3D Supervision. Advances in Neural Information Processing Systems **32** (2019)
30. Lorensen, W.E., Cline, H.E.: Marching Cubes: a High Resolution 3D Surface Construction Algorithm. ACM SIGGRAPH Computer Graphics **21**(4), 163–169 (1987). https://doi.org/10.1007/978-3-030-58452-8_24
31. Martel, J.N., Lindell, D.B., Lin, C.Z., Chan, E.R., Monteiro, M., Wetzstein, G.: Acorn: Adaptive Coordinate Networks for Neural Scene Representation. arXiv Preprint arXiv:2105.02788 (2021), https://arxiv.org/pdf/2105.02788
32. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy Networks: Learning 3D Reconstruction in Function Space. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4460–4470 (2019). https://doi.org/10.1109/CVPR.2019.00459
33. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J., Ramamoorthi, R., Ng, R.: NeRF: Representing Scenes As Neural Radiance Fields for View Synthesis. In: ECCV 2020: Computer Vision – ECCV 2020, pp. 405–421. Springer International Publishing (2020). https://doi.org/10.1007/978-3-030-58452-_24
34. Müller, T., Evans, A., Schied, C., Keller, A.: Instant Neural Graphics Primitives With a Multiresolution Hash Encoding. arXiv Preprint arXiv:2201.05989 (2022)
35. Niemeyer, M., Mescheder, L., Oechsle, M., Geiger, A.: Differentiable Volumetric Rendering: Learning Implicit 3D Representations Without 3D Supervision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3504–3515 (2020). https://doi.org/10.1109/CVPR42600.2020.00356
36. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2019). https://doi.org/10.1109/CVPR.2019.00025
37. Paschalidou, D., Katharopoulos, A., Geiger, A., Fidler, S.: Neural Parts: Learning Expressive 3D Shape Abstractions With Invertible Neural Networks. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 3203–3214 (2021). https://doi.org/10.1109/CVPR46437.2021.00322
38. Peng, S., Niemeyer, M., Mescheder, L., Pollefeys, M., Geiger, A.: Convolutional Occupancy Networks. In: European Conference on Computer Vision. pp. 523–540. Springer, Springer (2020). https://doi.org/10.1007/978-3-030-58580-8_31
39. Renderpeople: renderpeople.com (2022)
40. Saito, S., Huang, Z., Natsume, R., Morishima, S., Kanazawa, A., Li, H.: PIFu: Pixel-Aligned Implicit Function for High-Resolution Clothed Human Digitization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2304–2314 (2019). https://doi.org/10.1109/ICCV.2019.00239
41. Saragadam, V., Tan, J., Balakrishnan, G., Baraniuk, R.G., Veeraraghavan, A.: MINER: Multiscale Implicit Neural Representations (2022). https://doi.org/10.48550/arXiv.2202.03532
42. Simeonov, A., Du, Y., Tagliasacchi, A., Tenenbaum, J.B., Rodriguez, A., Agrawal, P., Sitzmann, V.: Neural Descriptor Fields: SE(3)-Equivariant Object Representations for Manipulation. arXiv Preprint arXiv:2112.05124 (2021). https://doi.org/10.48550/arXiv.2112.05124

43. Sitzmann, V., Chan, E., Tucker, R., Snavely, N., Wetzstein, G.: MetaSDF: Meta-Learning Signed Distance Functions. Advances in Neural Information Processing Systems **33**, 10136–10147 (2020). https://doi.org/10.5555/3495724.3496574
44. Sitzmann, V., Martel, J., Bergman, A., Lindell, D., Wetzstein, G.: Implicit Neural Representations With Periodic Activation Functions. Advances in Neural Information Processing Systems **33**, 7462–7473 (2020). https://doi.org/10.1109/WACV51458.2022.00234
45. Sitzmann, V., Rezchikov, S., Freeman, B., Tenenbaum, J., Durand, F.: Light Field Networks: Neural Scene Representations With Single-Evaluation Rendering. Advances in Neural Information Processing Systems **34** (2021). https://doi.org/10.48550/arXiv.2106.02634
46. Sitzmann, V., Zollhöfer, M., Wetzstein, G.: Scene Representation Networks: Continuous 3D-Structure-Aware Neural Scene Representations. Advances in Neural Information Processing Systems **32** (2019). https://doi.org/10.48550/arXiv.1906.01618
47. Takikawa, T., Litalien, J., Yin, K., Kreis, K., Loop, C., Nowrouzezahrai, D., Jacobson, A., McGuire, M., Fidler, S.: Neural Geometric Level of Detail: Real-Time Rendering With Implicit 3D Shapes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11358–11367 (2021). https://doi.org/10.1109/CVPR46437.2021.01120
48. Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., Ng, R.: Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. Advances in Neural Information Processing Systems **33**, 7537–7547 (2020). https://doi.org/10.48550/arXiv.2006.10739
49. Tang, D., Singh, S., Chou, P.A., Hane, C., Dou, M., Fanello, S., Taylor, J., Davidson, P., Guleryuz, O.G., Zhang, Y., et al.: Deep Implicit Volume Compression. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1293–1303 (2020). https://doi.org/10.1109/CVPR42600.2020.00137
50. Tretschk, E., Tewari, A., Golyanik, V., Zollhöfer, M., Stoll, C., Theobalt, C.: Patchnets: Patch-Based Generalizable Deep Implicit 3D Shape Representations. In: European Conference on Computer Vision. pp. 293–309. Springer, Springer (2020)
51. Turk, G., Levoy, M.: Zippered Polygon Meshes From Range Images. In: Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques. pp. 311–318 (1994). https://doi.org/10.1145/192161.192241
52. Xu, Q., Wang, W., Ceylan, D., Mech, R., Neumann, U.: DISN: Deep Implicit Surface Network for High-Quality Single-View 3D Reconstruction. Advances in Neural Information Processing Systems **32** (2019). https://doi.org/10.5555/3454287.3454332
53. Yang, G., Belongie, S., Hariharan, B., Koltun, V.: Geometry Processing With Neural Fields. In: Thirty-Fifth Conference on Neural Information Processing Systems (2021), https://papers.nips.cc/paper/2021/file/bd686fd640be98efaae0091fa301e613-Paper.pdf
54. Yao, S., Yang, F., Cheng, Y., Mozerov, M.G.: 3D Shapes Local Geometry Codes Learning With SDF. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2110–2117 (2021). https://doi.org/10.1109/ICCVW54120.2021.00239
55. Yariv, L., Gu, J., Kasten, Y., Lipman, Y.: Volume Rendering of Neural Implicit Surfaces. Advances in Neural Information Processing Systems **34** (2021). https://doi.org/10.1109/CVPR46437.2021.01120

56. Zakharov, S., Kehl, W., Bhargava, A., Gaidon, A.: Autolabeling 3D Objects With Differentiable Rendering of SDF Shape Priors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12224–12233 (2020). https://doi.org/10.1109/CVPR42600.2020.01224