ROAD: Learning an Implicit Recursive Octree Auto-Decoder to Efficiently Encode 3D Shapes

Sergey Zakharov, Rareş Ambruş, Katherine Liu, Adrien Gaidon

Abstract

Compact and accurate representations of 3D shapes are central to many perception and robotics tasks. State-of-the-art learning-based methods can reconstruct single objects but scale poorly to large datasets. We present a novel recursive implicit representation to efficiently and accurately encode large datasets of complex 3D shapes by recursively traversing an implicit octree in latent space. Our implicit Recursive Octree Auto-Decoder (ROAD) learns a hierarchically structured latent space enabling state-of-the-art reconstruction results at a compression ratio above 99%. We also propose an efficient curriculum learning scheme that naturally exploits the coarse-to-fine properties of the underlying octree spatial representation. We explore the scaling law relating latent space dimension, dataset size, and reconstruction accuracy, showing that increasing the latent space dimension is enough to scale to large shape datasets. Finally, we show that our learned latent space encodes a coarse-to-fine hierarchical structure yielding reusable latents across different levels of details, and we provide qualitative evidence of generalization to novel shapes outside the training set. Project website: https://zakharos. github.io/projects/road/

1. Introduction

Accurately and efficiently representing 3D geometry is a cornerstone capability in computer vision and computer graphics, with many practical applications in robotics and artificial intelligence. Decades of research in this area have produced a myriad of approaches, from traditional explicit methods [11, 16, 18, 41, 42, 45, 58] to learning-based implicit representations that encode shapes in the weights of neural networks and use various learning cues such as Signed Distance Fields [40, 47, 72], occupancy [30, 41], or radiance [4, 31, 50, 70]. Choosing one representation over another typically involves various tradeoffs between accuracy, scalability and generalization [3, 56, 62]. Related methods have either focused on modeling single shapes with increasing levels of accuracy at higher costs in terms of memory or time [31, 34, 53, 68] or on modeling classes of shapes,

typically with single MLPs, which results in the ability to generalize to novel shapes as well as adapt to test-time data via differentiability but at the expense of high-frequency details [40, 55, 59, 71].

In this paper we address these key challenges through a novel neural network capable of simultaneously encoding a large number of shapes to a higher level of accuracy than previously possible. We build on recent advances which use neural fields (i.e. neural network parameterizations of continuous functions defined on Euclidean space) to represent any topology with arbitrary precision [62] combined with continuous, generative latent spaces for 3D shape generation [9, 40]. We aim to recover a compact function which learns to map complex topologies to an implicit space of shapes, without sacrificing reconstruction accuracy or differentiability. Our key insight is based on the fact that the neural field paradigm maps spatial coordinates to an encoding of the surface and thus requires thousands of evaluations to extract the underlying surface; moreover, the underlying implicit function needs to accurately model space outside the target geometry [28, 29, 37]. Alternatively, some methods explicitly define the relationship between an underlying explicit data structure and the latent space [53, 55], thus partitioning the implicit function.

To achieve high compression while still being able to reconstruct high-frequency details, we propose an implicit Recursive Octree Auto-Decoder (ROAD) formulation that operates entirely in the latent space and is guided by an octree partitioning of the space. The octree data structure provides a simple yet elegant solution for increasing surface detail while traversing down the tree [53, 55, 69], as well as an intuitive setup for a curriculum learning schedule where learning progresses according to a coarse-to-fine approach. Our method uses a single neural network to map a latent vector to eight other latent vectors corresponding to its eight children in 3-dimensional space; in turn, each of the predicted latent vectors can be fed back to the network for further subdivision. To extract a surface, we simply traverse down the tree starting from a single root latent vector, expanding nodes as needed based on predicted occupancy, until the desired level of resolution is reached. The output of a forward pass of our network is the actual surface, and it
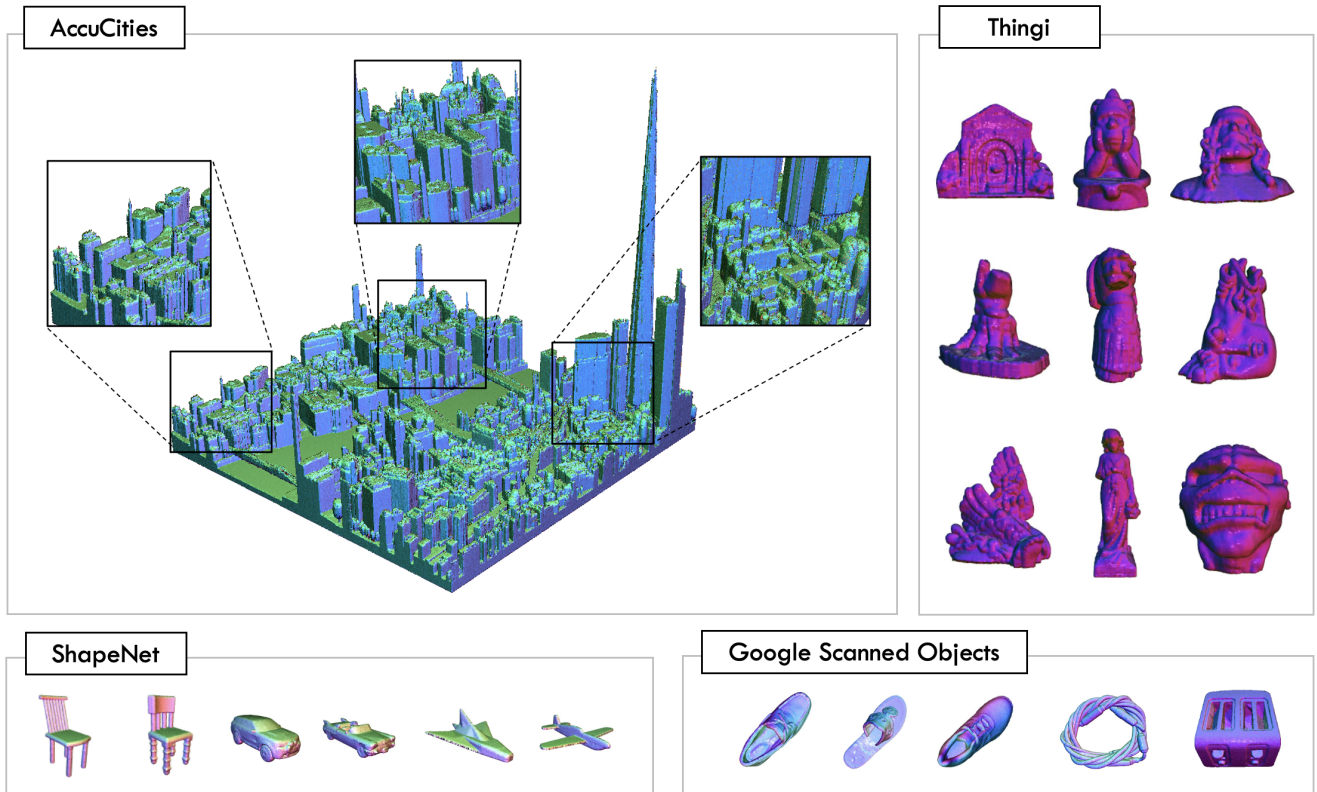
Figure 1. Our implicit Recursive Octree Auto-Decoder, **ROAD**, can represent many diverse shapes accurately and precisely with a small memory footprint thanks to recursive decoding of a hierarchically structured learned implicit shape representation.

can be obtained in milliseconds, unlike seconds or minutes for related methods requiring complex operations.

Our formulation leads to a latent space that captures shape similarity and hierarchy in a manner that is conducive to high compression as well generalization to shapes outside of the training set. Our efficient implementation leads to a reduction of up to 99% in terms of storage space compared to the original mesh models; this includes network weights as well as any other learned features needed for reconstructing the meshes. Finally, we explore the relationship between 3D modeling power (as measured by the Chamfer distance), latent space dimension and dataset size, and show how to tune our model's capacity with a single hyperparameter — the dimension of the latent space.

Our contributions are summarized as follows:

- A novel implicit representation parameterized by a recursive function that encodes an arbitrary number of 3D shapes in a shared latent space, while retaining high reconstruction fidelity and requiring up to 99% less storage space compared to the input mesh models;

- A curriculum learning method that naturally exploits the octree spatial data structure through a coarse-to-fine optimization scheme;

- An analysis of the scaling law that correlates latent space dimension, dataset size and 3D reconstruction accuracy and a qualitative analysis of the learned latent space indicating a coarse-to-fine hierarchical structure resulting in reusable latents.

2. Related Work

Neural Fields are continuous coordinate-based neural networks that encode an underlying property of a scene. The popularity of these representations has increased dramatically as recent results have shown that with enough modeling power coordinate-based networks can be used to encode underlying physical quantities with arbitrary levels of precision [31, 40, 62]. Applications of these techniques include modeling 3D shape [30, 40, 41, 47], appearance / radiance [4, 31, 50, 70], geometry [64], semantic information [25], material properties [5, 15], human shape and appearance [23], and robotics [2, 10, 38, 46, 66, 67]. Neural Fields have been used in robotics to represent 3D geometry and appearance with applications in grasping [6, 20], trajectory planning [2], object pose estimation and refinement [19, 21, 71, 72], object and surface reconstruction from sparse and noisy data [41, 60], multi-modal perception [14],
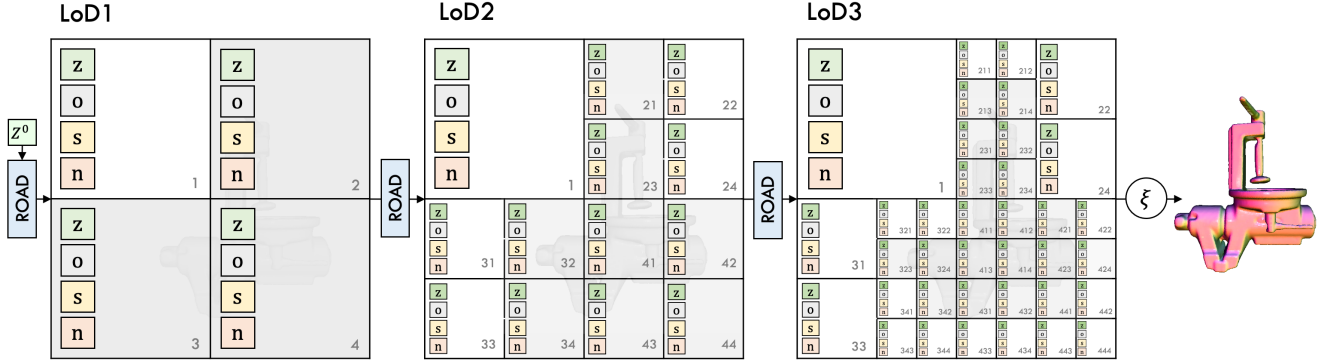
Figure 2. Our method extracts object surfaces by performing octree traversal. Starting from a latent vector of the parent cell ROAD extracts latent vectors for all children cells together with their occupancy, local SDF and surface normal estimates. It efficiently extends to large datasets while retaining high surface reconstruction quality.

localization [32] and SLAM [52, 74]. For an overview of recent methods and applications please consult [62].

Neural Fields for Shape Representation These methods represent shapes in the weights of the neural networks, and vary depending on the underlying signal used to encode the 3D space, e.g. occupancy [30, 41], Signed Distance Functions [40, 47, 72], density and radiance [4, 31]. A second distinction comes from the target domain, with some methods overfitting to a single scene/object [31, 34, 53], and other methods learning a generalizable prior over entire categories of shapes, e.g. as a generalizable latent space of Signed Distance Functions [40, 47, 72], as a convolutional prior over grid cells [30, 41] or image pixels [70], as weights of a kernel learned from data [26, 59, 61], as an object centric shape [71] and/or appearance prior [22]. The modeling power of these methods can be further improved by modulating the input coordinates with period functions [48, 54], while rendering speed, training time and networks size can be improved by factorizing the scene tensor into multiple low-rank components [8], by utilizing multiple small size MLPs [43], by training on Sparse Voxel Fields [27] or via multiresolution hash input encoding [34]. A number of methods employ an octree datastructure to guide learning towards occupied areas of space [53, 55, 69]. Closest to our method, NGLOD [53] also uses an octree to adaptively fit shapes to multiple Levels-of-Detail (LoDs), however unlike [53] our method can represent multiple shapes. Additionally, thanks to our recursive scheme, we only need to store root level latents and not the entire grid as is the case for [53]. Recursive parameterizations have also been employed in the context of radiance fields [65] or for 3D shape representation [55]. Unlike [55], we use a lightweight decoder-only recursive architecture to represent high quality shapes as dense oriented point clouds through which we can extract the encoded surface in real-time. Its capacity

can be easily extended by simply modifying the size of the input latent vector and it is capable of storing additional attributes (such as material information) at minimal cost.

Differentiable Rendering refers to the ability to render an image and back-propagate training signal from the image back to the underlying representation; for an overview of latest methods please refer to [24, 57]. This allows 3D representations of scenes to be learned using only 2D supervision [51], generative models of objects [35], compositionality [36, 39, 63], learning from data in the wild [17, 33, 44] or test-time adaptation [72]. However, in the context of 3D shape representation, extracting the underlying surface from the implicit field typically involves expensive operations such as volume rendering [29], sphere tracing [28] or ray marching [37]. Our method maintains differentiabilty with respect to the input and can thus be optimized given partial 3D data as well as 2D images and additionally we output the underlying surface by design, foregoing the need for expensive operations for surface extraction.

3. Methodology

Preliminaries Our approach takes as input a set of shapes $S = \{S_1, \ldots, S_K\}$ and learns a space of implicit surfaces that represents the input. Each shape S_K is represented by an oriented point cloud consisting of points $\{\boldsymbol{p}_i^{GT} \in \mathbb{R}^3\}_{i=1}^{|S_K|}$ and associated normals $\{\boldsymbol{n}_i^{GT} \in \mathbb{R}^3\}_{i=1}^{|S_K|}$, where $|S_K|$ denotes the number of points in S_K.

Formulation Our method represents each shape with a D-dimensional latent vector denoting the root node of an octree which is traversed recursively, up to a predefined Level-of-Detail (LoD) M. We seek to regress three functions to recursively reconstruct 3D geometry, parameterized by neural networks: $\phi : \mathbb{R}^D \rightarrow \mathbb{R}^{8D}$ for latent subdivision; $\psi : \mathbb{R}^D \rightarrow \mathbb{R}^5$ for mapping the latent space to surface geometry (occupancy, signed distance to surface, surface nor-

mal); and $\xi : \mathbb{R}^7 \to \mathbb{R}^3$ for zero iso-surface projection. The hyperparameter M, the dimension of the latent space, is linked to the capacity of our representation as a function of the dataset size. Given a latent vector $\boldsymbol{z}^m \in \mathbb{R}^D$, where $0 \leq m \leq M$ denotes the LoD of the latent vector, we define:

$$\phi(\boldsymbol{z}^m) \to \{\boldsymbol{z}_i^{m+1}\}_{i=1}^8 \qquad (1)$$

as the function that performs a traversal of the latent space given input latent $\boldsymbol{z}^m$ and outputs a latent vector $\boldsymbol{z}_i^{m+1} = \lfloor \phi(\boldsymbol{z}^m) \rfloor_i$ for each of the 8 possible children, where $\lfloor \phi \rfloor_i$ denotes the ith output of ϕ.

By definition, ϕ projects back to the latent space of dimension D, i.e. $\boldsymbol{z}_i^{m+1} \in \mathbb{R}^D$, forcing the resulting latent to encode both high and low level information. This formulation allows ROAD to simply pass back the resulting latent to our recursive function, i.e. through $\phi(\boldsymbol{z}^{m+1})$, while propagating coarser, higher-level information along the latent space. A latent $\boldsymbol{z}^0$ at the lowest LoD therefore encodes the geometry of an entire object. Unlike other octree-based methods [53] that store all the octree latents, our formulation allows us to record only the root level latents for each input shape, i.e. $\mathcal{Z}^0 = \{\boldsymbol{z}_k^0\}_{k=0}^K$. We provide ablations for other formulations of ϕ in Section 4.

We employ ψ to map any latent vector to underlying surface geometry as follows:

$$\psi(\boldsymbol{z}^m) = \{o^m, s^m, \boldsymbol{n}^m\} \qquad (2)$$

The output of ψ consists of $o^m \in (0, 1)$ — the occupancy estimate denoting whether to continue expanding this cell further; $s^m \in (-1, 1)$ — the constrained signed distance value from the center of the cell to the surface of the object; and $\boldsymbol{n}^m \in \mathbb{R}^3$ — the surface normal vector. To extract the surface information at a particular LoD m starting from a root latent $\boldsymbol{z}^0$ we perform a tree traversal as follows:

$$\psi(\underbrace{\phi(\dots(\phi(\boldsymbol{z}^0)))}_{m \text{ times}}) = \{o^m, s^m, \boldsymbol{n}^m\} \qquad (3)$$

LoD4   LoD6
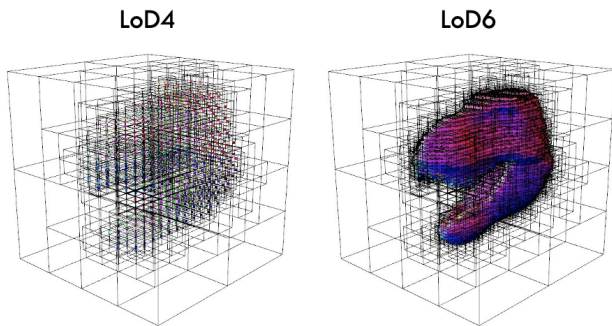

Figure 3. ROAD enables surface points to be directly extracted at different levels of detail.

Note that for clarity we omitted the notation $\lfloor \cdot \rfloor$ in Eq.3, however after each subdivision the appropriate child latent is selected, according to the desired branch of the octree to be expanded. We highlight that unlike other SDF based methods that implicitly encode surfaces [40], our method does not take as input Euclidean coordinates, and the mapping between different LoD levels is achieved entirely via the learned latents $\boldsymbol{z}^m \to \boldsymbol{z}^{m+1}$ which are forced to encode both local structure as well as global shape information. Moreover, unlike grid-based representations [30, 41] we fully exploit the sparse nature of the underlying octree representation, using the occupancy o_i to recursively expand only occupied cells.

Shape Recovery   To recover the final shape, we perform the zero-isosurface projection (see Fig. 4), using distance s_i, surface normal estimates $\boldsymbol{n}_i$, and voxel cell center coordinates $\boldsymbol{x}_i$ at the desired LoD. The voxel centers are determined and tracked whenever we subdivide a cell. To extract the object surface at LoD m, we use the following equation: $\boldsymbol{p}_i^m = \xi(\boldsymbol{x}_i^m, s_i^m, \boldsymbol{n}_i^m) = \boldsymbol{x}_i^m - \alpha^m \boldsymbol{n}_i^m s_i^m$, where α^m is a value that scales s^m. In our experiments we use a scaling factor equal to the voxel size at LoD m, i.e. $\alpha^m = 2/(2^m)$. Once the projection is performed for all the query points at a particular LoD, we get a dense surface point cloud that is differentiable back to the input latent vector $\boldsymbol{z}^0$. This property allows us to perform optimization to complete partial shapes based on the prior encapsulated in the network. An overview of our complete pipeline is provided in Algorithm 1.

Architecture and Training   We parameterize the functions ϕ, ψ with a single MLP with parameters θ, choosing a SIREN-based [49] network as periodic activation functions have been shown to be more capable at representing fine details. Our MLP uses a single layer encoder and multiple 2-layer decoder heads to output occupancy o, SDF s, surface normals $\boldsymbol{n}$. All hidden layers are 512-dimensional. We train our method by optimizing both the latent vectors $\mathcal{Z}$ and the parameters θ of the MLP using the Adam solver
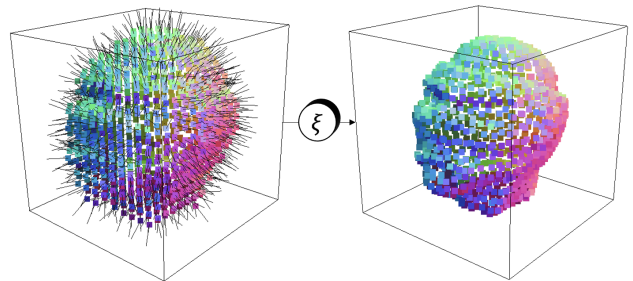

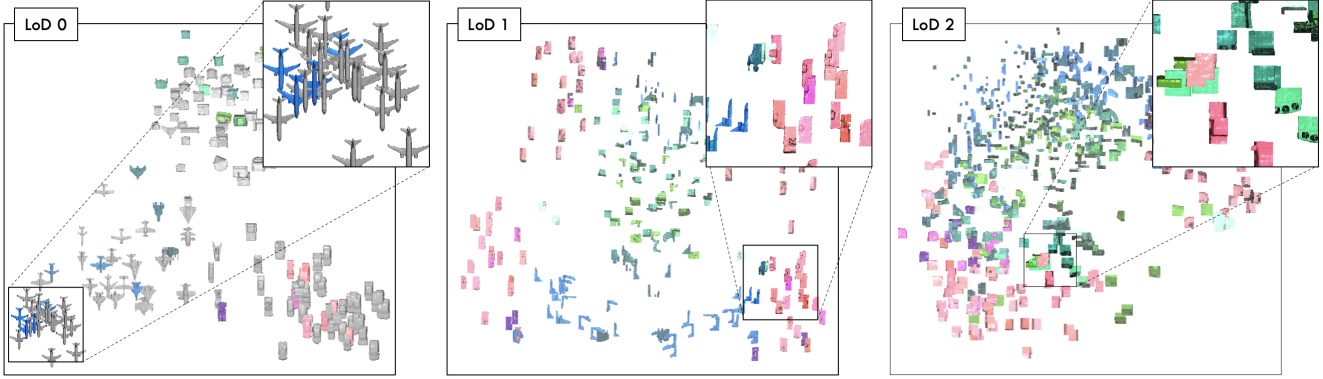Figure 4. Zero-isosurface projection.

Figure 5. **Latent space structure.** We use principle component analysis to visualize the encoded geometries of the ShapeNet150 dataset in two dimensions. Object color is related to object instance, with objects of the same class having similar colors, and is carried through each LoD. For visualization purposes, grey objects in LoD 0 are not propagated to the higher LoDs. Similar latent vectors encode similar geometries resulting in a clear class separation at LoD 0. Similar areas of the projected latent space become increasingly shared by the different classes at higher LoDs, suggesting that our approach efficiently encodes object geometry by learning common geometric primitives.

with a learning rate of 6×10^{-5}.

To supervise training, we define losses for each of the decoder levels at every LoD. Occupancy loss $\mathcal{L}_o$ as a binary cross entropy, whereas SDF $\mathcal{L}_s$ and surface normals $\mathcal{L}_n$ losses minimize the l_2 distance between respective predictions and ground truth values. The final loss is formulated as:

$$\mathcal{L} = \sum_{m \in M} w_o \mathcal{L}_o^m + w_s^m \mathcal{L}_s^m + w_n \mathcal{L}_n^m, \qquad (4)$$

Algorithm 1: Octree-based recursive surface extraction

Input: M maximum recursion depth, $\mathcal{Z}^0 = \{z^0\}$ object latent vector

Output: $\mathbf{p} \in \mathbb{R}^3$ surface points,

```
/* Recursively subdivide voxels
   until desired LoD is reached  */
```
1 **for** $m \in \{1, \ldots, M\}$ **do**
2    $\mathcal{Z}^m \leftarrow \{\}$
3   **for** z^{m-1} *in* $\mathcal{Z}^{m-1}$ **do**
4       $\{z_i^m, o_i^m, s_i^m, \boldsymbol{n}_i^m\}_{i=1}^8 \leftarrow \text{ROAD}(z^{m-1})$;
```
        // recursive subdivision
```
5       $\mathcal{Z}^m \leftarrow \mathcal{Z}^m \cup \{z_i^m | o_i^m \geq \theta\}$;   // add occupied latents
6   **end**
7 **end**
```
/* Extract object shape          */
```
8 **for** z^m *in* $\mathcal{Z}^m$ **do**
9    $\mathbf{p} \leftarrow \mathbf{p} \cup \xi(\text{ROAD}(z^m))$
10 **end**
11 **return** p

---

where $w_o = 1$, w_s^d is a function returning an inverse voxel radius for level m, and $w_v = 0.1$.

Curriculum Learning When it comes to training on large datasets or on datasets with high resolution models requiring high LoDs, the vanilla training procedure requires much more time to converge as opposed to training on simpler datasets. To alleviate this problem we introduce a curriculum training procedure. Instead of initiating training from the desired final LoD, we change the first LoD to be lower (we use LoD 3 in our experiments) and keep track of the mean of predicted occupancy confidences at a given level m. Lower LoDs are faster to train on due to a smaller number of latents to optimize. Moreover, it makes it faster to sample random training points at each iteration, thus further accelerating the training procedure. To compute our confidence score, we first take a softmax over two occupancy values for all estimated voxels at a given LoD, and then take a max value between all the pairs. If our average confidence θ is high enough in either occupancy or non occupancy, we jump to the next LoD and repeat this procedure until the final LoD is reached. In our experiments, we show that this simple technique helps to accelerate the training process especially when training on large and high resolution datasets. We set $\theta = 0.95$ in our experiments.

4. Experiments

To demonstrate the 3D reconstruction and compression capabilities of our approach we run a number of experiments on the ShapeNet [7], Thingi10K [73], Google Scanned Objects [13], and the AccuCities [1] datasets. We report reconstruction metrics as measured by the Chamfer distance (multiplied by 10^3) as well as interesection over

Method	ShapeNet150			Thingi32		
	Storage (MB) ($\downarrow$)	gIoU ($\uparrow$)	Chamfer ($\downarrow$)	Storage (MB) ($\downarrow$)	gIoU ($\uparrow$)	Chamfer ($\downarrow$)
DeepSDF [40]	1052.6	86.9	0.316	224.6	96.8	0.053
FFN [54]	301.6	88.5	0.077	64.3	97.7	0.033
SIREN [49]	151.3	78.4	0.381	32.3	95.1	0.077
Neural Implicits [12]	4.4	82.2	0.500	**0.9**	96.0	0.092
NGLOD [53]	185.4	91.7	0.062	39.6	**99.4**	0.027
Ours / LoD6		86.3	0.175		96.4	0.138
Ours / LoD7	**3.8**	94.2	0.067	3.2	98.4	0.045
Ours / LoD8		94.9	0.041		98.7	0.022
Ours / LoD9		**94.9**	**0.036**		98.7	**0.017**

Table 1. **Shape Reconstruction**. This table shows reconstruction and compression comparisons against two datasets. As opposed to the baselines, our method trains a single model for the entire dataset, while still outperforming them in terms of reconstruction quality.

union over points uniformly sampled in the bounding volume of the ground truth shape.

Reconstruction We follow the protocol of [53] and train on a subset of 150 objects from ShapeNet [7] and a subset of 32 objects from Thingi10K [73]. We report results in Table 1. We note that the baselines we compare against train one model for each shape in the dataset (i.e. [53] trains 32 networks for each object in Thingi32). Owing to the efficient recursive decoding scheme implemented by our method, we can train a single model for each dataset and still be competitive in terms of network size and reconstruction accuracy. We set the latent size D to 64 for Thingi32 and to 96 for ShapeNet150 respectively. Indeed, our network achieves state-of-the-art reconstruction results of 0.036 Chamfer distance and an IoU of more than 94% on ShapeNet150 with a network of size 3.8MB: this amounts to a compression of more than 99%, as the original mesh datas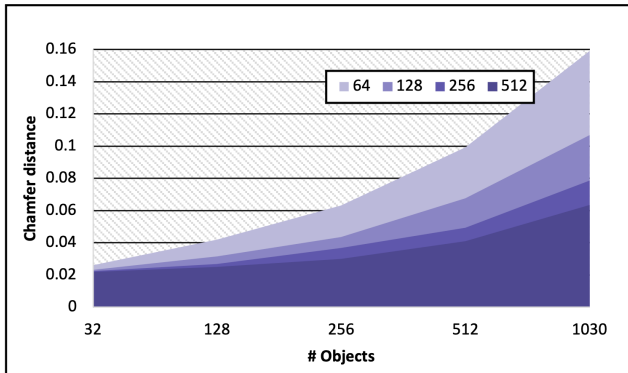et measures 630MB. We report a similar compression ratio on Thingi32 (473MB) while achieving state-of-the-art reconstruction accuracy of 0.017 Chamfer distance and a competitive IoU of 98.7%.

We further explore the reconstruction capabilities of our method by encoding a model from the AccuCities [1] dataset: a neighborhood from London consisting of 1.9 million triangles and requiring 252MB of disk space. We set the latent size D to 512. Qualitative results are shown in Fig.1 with additional images in the supplementary. Quantitatively we achieve a Chamfer distance of 0.04 when comparing against the ground truth model, while using a network of size 11 MB.

Scaling to larger datasets For this experiment we use the entire Google Scanned Objects [13] dataset consisting of a

Figure 6. Chamfer distance vs data quantity for different latent vector sizes.
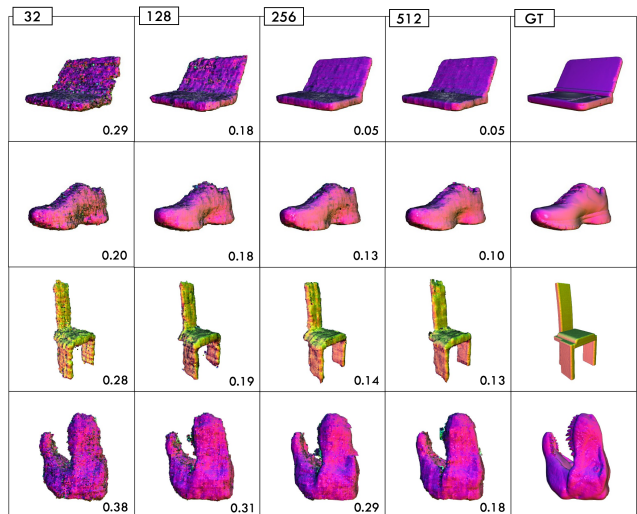

Figure 7. **Generalization.** We demonstrate our method's increasing generalization ability when trained on many objects.

Superposition	Storage (MB)	gIoU	Chamfer
Direct	3.2	98.7	0.017
Addition	3.2	96.9	0.039
Concatenation	17.9	99.4	0.013

Table 2. **Latent Fusion.** Here we compare different ways to propagate latent vectors to the next LoDs.

Surface density	Low	Medium	High
Sphere tracing	5 min	6 min	10 min
Marching cubes	0.1 s	0.9 s	6 s
Ours	**11 ms**	**13 ms**	**17 ms**

Table 3. **Inference time.** Our method extracts object surfaces in real time, significantly outperforming the state of the art.

total of 1030 object models. We introduce training splits of different sizes (32, 128, 256, 512, 1030) to study the relation between the dataset complexity and the latent vector size, a novel hyperparameter specific to our formulation. We restrict our analysis into the scaling properties of our network to the dimension of the latent space, and mention that other methods that are generally applicable to machine learning models (i.e. number of layers, training schedule, etc.) can be used to further tweak the performance of our method. Our results are recorded in Fig. 6: for each split we train networks with increasing latent vector sizes (64, 128, 256, 512) and record the resulting Chamfer distance by comparing the reconstructed models with the ground truth models of that specific split. We note a strong correlation between dataset complexity and latent size. Specifically, our results indicate that our method can efficiently scale to an increasing number of shapes by only modifying the latent vector size while keeping the network parameterization intact.

Latent Space Analysis To qualitatively analyze the properties of the learned latent space, we project the latent space at specific LoDs into two dimensions via principle component analysis to visualize the encoded geometries of the ShapeNet dataset, as shown in Fig. 5.

We observe that objects of the same class are spatially close in the projected space at the top level LoD, demonstrating that similar latent vectors encode similar geometries. Furthermore, at higher LoDs, similar areas of the projected latent space are increasingly shared by the different classes, suggesting that our approach efficiently encodes object geometry by learning geometric primitives common in the dataset.

We also visualize the nearest neighbors of specific latent vectors from the network trained on Thingi32 to LoD 9 in Fig. 8. We show that similar latent vectors can represent the local geometry at different object coordinates, without explicit positional encoding. Increasing the LoD also intuitively reduces the geometric complexity represented by the latent, as seen by the 3D edge feature from LoD 2 and the oriented patch feature from LoD 4.

Generalization In this experiment we demonstrate the generalization capabilities of our method. We take four networks from the data compression experiment each trained

on a different split of the Google Scanned Objects dataset (32, 128, 256, and 512 objects) and optimize latent vectors to fit unseen models from the same as well as other datasets (Thingi32, ShapeNet150) while keeping network weights frozen. The results are shown in Fig. 7. Reconstruction quality plotted in terms of Chamfer distance shows increasing generalization capability for networks trained on more models.

Surface Extraction By design, our method differentiably extracts object surfaces in real time with minimal memory overheads. In comparison, NGLOD [53] or other SDF-based [40] methods require either an expensive sphere ray tracing or non-differentiable marching cubes to extract surface. Table 3 compares inference times when extracting the object surface at different levels of density. For our method we perform inference up to LoDs 6, 7 and 8 respectively, which corresponds to approximately 20000, 80000 and 300000 surface points, respectively for the Thingi32 models. We compare against a marching cubes baseline from [40] and a sphere tracing baseline from [53], and we iterate until the desired number of surface points is sampled. For a fair comparison with our method we extract 20000, 80000 and 300000 using both sphere tracing and marching cubes. We note that our method extracts the object surface up to 3 orders of magnitude faster that the sphere tracing baseline; our method requires *a total of 1-2 seconds* to extract object surfaces for the *entire* ShapeNet150 dataset. The marching cubes baseline, while faster than sphere tracing, is still not real-time capable and not differentiable. Finally, we observe that the results for this experiment were obtained on a single A6000 GPU, without any optimization.

Latent Vector Fusion As described in Section 3 and in Fig. 2, our method propagates information through the latent space via the recursive function ϕ (see Eq.1). Here we explore different forms of latent subdivision: addition and concatenation, with the complete definitions provided in the supplementary. While addition does not change the dimension of the latent vector D, it explicitly defines how information is propagated from parent to child latent (i.e. via addition). Conversely, concatenation makes D increase with each recursion level, and information is directly copied as we traverse the latent space. This introduces significant
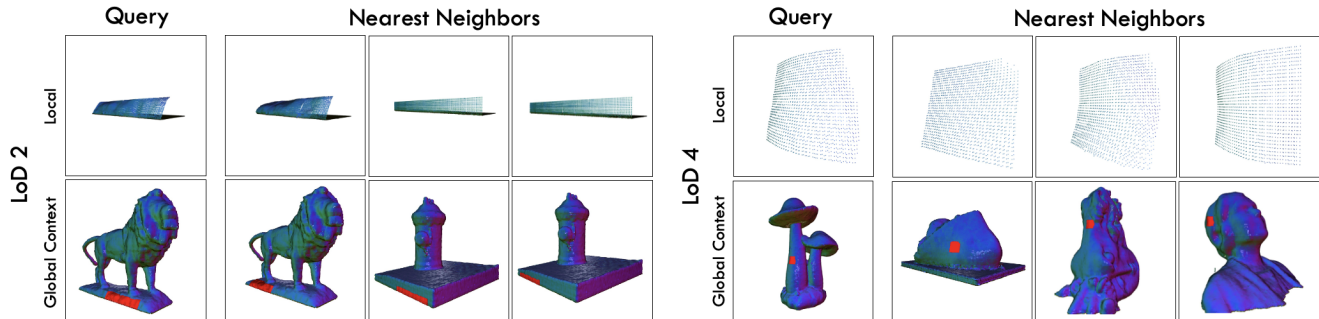
Figure 8. **Nearest neighbors.** We visualize the geometry encoded by example latents, as well as the geometry encoded by the nearest neighbors within the same LoD across all objects in the Thingi32 dataset, determined by the Euclidean distance metric on the latent vector space (top row). We color the points in the octree cell corresponding to the latent, showing that our approach enables latents with similar shapes to be used at different global positions (bottom row).

modifications to the underlying neural network architecture, requiring specialized networks at each LoD. The results of this ablative analysis are summarized in Table 2. As expected, contcatenation serves as an upper bound for performance and it achieves the highest performance but requires $5\times$ more storage space, and a more complicated formulation. Although similar in formuation to direct regression, the addition version of our method achieves poor results, which we attribute to the artificial constraint imposed on how information is propagated through the latent space.

5. Discussion

Limitations and Future Work Our representation currently only supports 3D geometry. For future work, we would like to explore its extension to other modalities (object color and material properties) as well as representations (images and radiance fields [31]). Another interesting direction that could be explored is a combination of our pipeline with different downstream tasks (object detection and pose estimation). Our representaton is fully differentiable and thus allows the propagation of useful 3D gradients for shape optimization given partial information.

Conclusion We presented a novel recursive implicit representation to effectively represent and compress 3D geometry by framing it as the traversal of an implicit octree in a learned latent space. It extracts geometry in real-time and scales to large datasets while retaining high reconstruction quality. As a result, we outperform state-of-the-art reconstruction results on the ShapeNet150 and Thingi32 datasets, even when compared to methods training a single network per model. Our analysis of the representation explores the structure of the latent space and presents a scaling law defining a relationship between latent space dimension, dataset size and reconstruction accuracy.

References

[1] 3d model of london and city models. https://www.accucities.com/. Accessed: 2022-05-18. 5, 6

[2] Michal Adamkiewicz, Timothy Chen, Adam Caccavale, Rachel Gardner, Preston Culbertson, Jeannette Bohg, and Mac Schwager. Vision-only robot navigation in a neural radiance world. *RA-L*, 2022. 2

[3] Eman Ahmed, Alexandre Saint, Abd El Rahman Shabayek, Kseniya Cherenkova, Rig Das, Gleb Gusev, Djamila Aouada, and Bjorn Ottersten. A survey on deep learning advances on different 3d data representations. *arXiv*, 2018. 1

[4] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *ICCV*, 2021. 1, 2, 3

[5] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. Nerd: Neural reflectance decomposition from image collections. In *ICCV*, 2021. 2

[6] Michel Breyer, Jen Jen Chung, Lionel Ott, Siegwart Roland, and Nieto Juan. Volumetric grasping network: Real-time 6 dof grasp detection in clutter. In *CoRL*, 2020. 2

[7] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. ShapeNet: An information-rich 3d model repository. *arXiv*, 2015. 5, 6

[8] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. *arXiv*, 2022. 3

[9] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *CVPR*, 2019. 1

[10] Kashyap Chitta, Aditya Prakash, and Andreas Geiger. Neat: Neural attention fields for end-to-end autonomous driving. In *ICCV*, 2021. 2

[11] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH*, 1996. 1

[12] Thomas Davies, Derek Nowrouzezahrai, and Alec Jacobson. Overfit neural networks as a compact shape representation. *arXiv*, 2020. 6

[13] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items. *arXiv*, 2022. 5, 6

[14] Ruohan Gao, Yen-Yu Chang, Shivani Mall, Li Fei-Fei, and Jiajun Wu. Objectfolder: A dataset of objects with implicit visual, auditory, and tactile representations. In *CoRL*, 2021. 2

[15] Saeed Hadadan, Shuhong Chen, and Matthias Zwicker. Neural radiosity. *TOG*, 2021. 2

[16] Rana Hanocka, Amir Hertz, Noa Fish, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. Meshcnn: a network with an edge. *TOG*, 2019. 1

[17] Philipp Henzler, Jeremy Reizenstein, Patrick Labatut, Roman Shapovalov, Tobias Ritschel, Andrea Vedaldi, and David Novotny. Unsupervised learning of 3d object categories from videos in the wild. In *CVPR*, 2021. 3

[18] Armin Hornung, Kai M Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. Octomap: An efficient probabilistic 3d mapping framework based on octrees. *Autonomous robots*, 2013. 1

[19] Lin Huang, Tomas Hodan, Lingni Ma, Linguang Zhang, Luan Tran, Christopher Twigg, Po-Chen Wu, Junsong Yuan, Cem Keskin, and Robert Wang. Neural correspondence field for object pose estimation. *ECCV*, 2022. 2

[20] Jeffrey Ichnowski, Yahav Avigal, Justin Kerr, and Ken Goldberg. Dex-nerf: Using a neural radiance field to grasp transparent objects. In *CoRL*, 2021. 2

[21] Muhammad Zubair Irshad, Sergey Zakharov, Rares Ambrus, Thomas Kollar, Zsolt Kira, and Adrien Gaidon. Shapo: Implicit representations for multi-object shape, appearance, and pose optimization. In *ECCV*, 2022. 2

[22] Wonbong Jang and Lourdes Agapito. Codenerf: Disentangled neural radiance fields for object categories. In *ICCV*, 2021. 3

[23] Wei Jiang, Kwang Moo Yi, Golnoosh Samei, Oncel Tuzel, and Anurag Ranjan. Neuman: Neural human radiance field from a single video. *arXiv*, 2022. 2

[24] Hiroharu Kato, Deniz Beker, Mihai Morariu, Takahiro Ando, Toru Matsuoka, Wadim Kehl, and Adrien Gaidon. Differentiable rendering: A survey. *arXiv*, 2020. 3

[25] Abhijit Kundu, Kyle Genova, Xiaoqi Yin, Alireza Fathi, Caroline Pantofaru, Leonidas Guibas, Andrea Tagliasacchi, Frank Dellaert, and Thomas Funkhouser. Panoptic neural fields: A semantic object-aware neural scene representation. *arXiv*, 2022. 2

[26] Hsueh-Ti Derek Liu, Francis Williams, Alec Jacobson, Sanja Fidler, and Or Litany. Learning smooth neural functions via lipschitz regularization. *arXiv*, 2022. 3

[27] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *NeurIPS*, 2020. 3

[28] Shaohui Liu, Yinda Zhang, Songyou Peng, Boxin Shi, Marc Pollefeys, and Zhaopeng Cui. Dist: Rendering deep implicit signed distance function with differentiable sphere tracing. In *CVPR*, 2020. 1, 3

[29] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *arXiv*, 2019. 1, 3

[30] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*, 2019. 1, 2, 3, 4

[31] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 2, 3, 8

[32] Arthur Moreau, Nathan Piasco, Dzmitry Tsishkou, Bogdan Stanciulescu, and Arnaud de La Fortelle. Lens: Localization enhanced by nerf synthesis. In *CoRL*, 2022. 3

[33] Norman Müller, Andrea Simonelli, Lorenzo Porzi, Samuel Rota Bulò, Matthias Nießner, and Peter Kontschieder. Autorf: Learning 3d object radiance fields from single view observations. *arXiv*, 2022. 3

[34] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *arXiv*, 2022. 1, 3

[35] Siva Karthik Mustikovela, Shalini De Mello, Aayush Prakash, Umar Iqbal, Sifei Liu, Thu Nguyen-Phuoc, Carsten Rother, and Jan Kautz. Self-supervised object detection via generative image synthesis. In *ICCV*, 2021. 3

[36] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *CVPR*, 2021. 3

[37] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *CVPR*, 2020. 1, 3

[38] Joseph Ortiz, Alexander Clegg, Jing Dong, Edgar Sucar, David Novotny, Michael Zollhoefer, and Mustafa Mukadam. isdf: Real-time neural signed distance fields for robot perception. *arXiv*, 2022. 2

[39] Julian Ost, Fahim Mannan, Nils Thuerey, Julian Knodt, and Felix Heide. Neural scene graphs for dynamic scenes. In *CVPR*, 2021. 3

[40] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. 1, 2, 3, 4, 6, 7

[41] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *ECCV*, 2020. 1, 2, 3, 4

[42] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. PointNet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 1

[43] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *ICCV*, 2021. 3

[44] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *ICCV*, 2021. 3

[45] Radu Bogdan Rusu and Steve Cousins. 3d is here: Point cloud library (pcl). In *ICRA*, 2011. 1

[46] Anthony Simeonov, Yilun Du, Andrea Tagliasacchi, Joshua B Tenenbaum, Alberto Rodriguez, Pulkit Agrawal, and Vincent Sitzmann. Neural descriptor fields: Se (3)-equivariant object representations for manipulation. In *ICRA*, 2022. 2

[47] Vincent Sitzmann, Eric Chan, Richard Tucker, Noah Snavely, and Gordon Wetzstein. Metasdf: Meta-learning signed distance functions. *NeurIPS*, 2020. 1, 2, 3

[48] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *NeurIPS*, 2020. 3

[49] Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Proc. NeurIPS*, 2020. 4, 6

[50] Vincent Sitzmann, Semon Rezchikov, Bill Freeman, Josh Tenenbaum, and Fredo Durand. Light field networks: Neural scene representations with single-evaluation rendering. *NeurIPS*, 2021. 1, 2

[51] Vincent Sitzmann, Michael Zollhoefer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *NeurIPS*, 2019. 3

[52] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J Davison. imap: Implicit mapping and positioning in real-time. In *ICCV*, 2021. 3

[53] Towaki Takikawa, Joey Litalien, Kangxue Yin, Karsten Kreis, Charles Loop, Derek Nowrouzezahrai, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In *CVPR*, 2021. 1, 3, 4, 6, 7

[54] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *NeurIPS*, 2020. 3, 6

[55] Jia-Heng Tang, Weikai Chen, Jie Yang, Bo Wang, Songrun Liu, Bo Yang, and Lin Gao. Octfield: Hierarchical implicit functions for 3d modeling. *arXiv*, 2021. 1, 3

[56] Johan WH Tangelder and Remco C Veltkamp. A survey of content based 3d shape retrieval methods. *Multimedia tools and applications*, 2008. 1

[57] Ayush Tewari, Justus Thies, Ben Mildenhall, Pratul Srinivasan, Edgar Tretschk, Yifan Wang, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, et al. Advances in neural rendering. *arXiv*, 2021. 3

[58] Thomas Whelan, Stefan Leutenegger, Renato Salas-Moreno, Ben Glocker, and Andrew Davison. Elasticfusion: Dense slam without a pose graph. Robotics: Science and Systems, 2015. 1

[59] Francis Williams, Zan Gojcic, Sameh Khamis, Denis Zorin, Joan Bruna, Sanja Fidler, and Or Litany. Neural fields as learnable kernels for 3d reconstruction. *arXiv*, 2021. 1, 3

[60] Francis Williams, Zan Gojcic, Sameh Khamis, Denis Zorin, Joan Bruna, Sanja Fidler, and Or Litany. Neural fields as learnable kernels for 3d reconstruction. In *CVPR*, 2022. 2

[61] Francis Williams, Matthew Trager, Joan Bruna, and Denis Zorin. Neural splines: Fitting 3d surfaces with infinitely-wide neural networks. In *CVPR*, 2021. 3

[62] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond. *arXiv*, 2021. 1, 2, 3

[63] Bangbang Yang, Yinda Zhang, Yinghao Xu, Yijin Li, Han Zhou, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. Learning object-compositional neural radiance field for editable scene rendering. In *ICCV*, 2021. 3

[64] Guandao Yang, Serge Belongie, Bharath Hariharan, and Vladlen Koltun. Geometry processing with neural fields. *NeurIPS*, 2021. 2

[65] Guo-Wei Yang, Wen-Yang Zhou, Hao-Yang Peng, Dun Liang, Tai-Jiang Mu, and Shi-Min Hu. Recursive-nerf: An efficient and dynamically growing nerf. *arXiv*, 2021. 3

[66] Lin Yen-Chen, Pete Florence, Jonathan T Barron, Tsung-Yi Lin, Alberto Rodriguez, and Phillip Isola. Nerf-supervision: Learning dense object descriptors from neural radiance fields. *arXiv*, 2022. 2

[67] Lin Yen-Chen, Pete Florence, Jonathan T Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. inerf: Inverting neural radiance fields for pose estimation. In *IROS*, 2021. 2

[68] Wang Yifan, Lukas Rahmann, and Olga Sorkine-Hornung. Geometry-consistent neural shape representation with implicit displacement fields. *arXiv*, 2021. 1

[69] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *ICCV*, 2021. 1, 3

[70] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *CVPR*, 2021. 1, 2, 3

[71] Sergey Zakharov, Rares Andrei Ambrus, Vitor Campagnolo Guizilini, Dennis Park, Wadim Kehl, Fredo Durand, Joshua B Tenenbaum, Vincent Sitzmann, Jiajun Wu, and Adrien Gaidon. Single-shot scene reconstruction. In *CoRL*, 2021. 1, 2, 3

[72] Sergey Zakharov, Wadim Kehl, Arjun Bhargava, and Adrien Gaidon. Autolabeling 3d objects with differentiable rendering of sdf shape priors. In *CVPR*, 2020. 1, 2, 3

[73] Qingnan Zhou and Alec Jacobson. Thingi10k: A dataset of 10,000 3d-printing models. *arXiv*, 2016. 5, 6

[74] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *CVPR*, 2022. 3