POCO: Point Convolution for Surface Reconstruction

Alexandre Boulch[1] Renaud Marlet[1,2]

[1]Valeo.ai, Paris, France [2]LIGM, Ecole des Ponts, Univ Gustave Eiffel, CNRS, Marne-la-Vallée, France

Abstract

Implicit neural networks have been successfully used for surface reconstruction from point clouds. However, many of them face scalability issues as they encode the isosurface function of a whole object or scene into a single latent vector. To overcome this limitation, a few approaches infer latent vectors on a coarse regular 3D grid or on 3D patches, and interpolate them to answer occupancy queries. In doing so, they lose the direct connection with the input points sampled on the surface of objects, and they attach information uniformly in space rather than where it matters the most, i.e., near the surface. Besides, relying on fixed patch sizes may require discretization tuning. To address these issues, we propose to use point cloud convolutions and compute latent vectors at each input point. We then perform a learning-based interpolation on nearest neighbors using inferred weights. Experiments on both object and scene datasets show that our approach significantly outperforms other methods on most classical metrics, producing finer details and better reconstructing thinner volumes. The code is available at `https://github.com/valeoai/POCO`.

1. Introduction

Constructing a surface or volume representation from 3D points sampled at the surface of an object or scene has numerous applications, from digital twins processing to augmented and virtual reality. Cheaper sensors directly producing 3D points (depth cameras, low-cost lidars) and mature multi-view stereo techniques [88, 89] operating on images offer increasing opportunities for such reconstructions.

Traditional 3D reconstruction approaches [4] generally express the target surface as the solution to an optimization problem under some prior constraints. Possibly leveraging visibility or normal information, they are generally scalable to large scenes and offer a substantial robustness to noise and outliers [47, 51, 71, 81, 94, 103, 110, 123]. Although some try to cope with density variation [9, 42, 43], a common limitation of these approaches is their inability to properly complete parts of the scene that are less densely

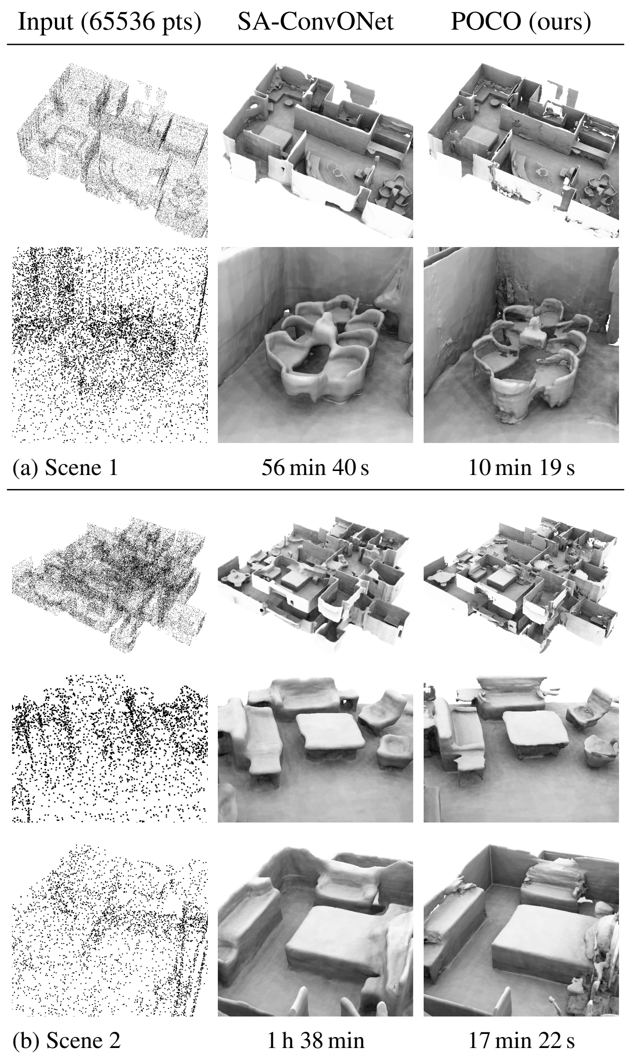

Figure 1. **MatterPort3D.** POCO trains on Synthetic Rooms 10k.

sampled or that are missing (typically due to occlusions). A variety of hand-crafted priors try to address this completeness issue: local or global smoothness [58], decomposition into geometric primitives [87] (in particular for piecewise-planar man-made environments [3, 6, 14, 28, 72]) and structural regularities [53, 79]. Data-driven priors have also been explored, based on shape retrieval [30], possibly with de-

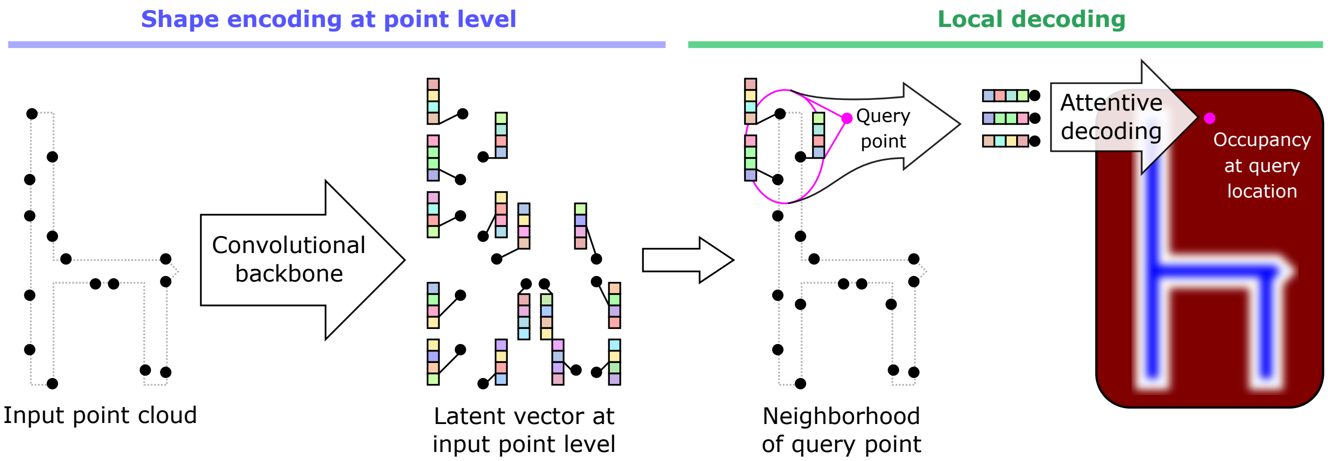

Figure 2. **Overview of our method (inference).** Given 3D points sampled on a surface, we construct latent vectors at each input point. Then, to estimate the occupancy of a given query point in space, we interpolate with inferred weights the relative occupancy scores in a neighborhood. Last, a mesh is reconstructed based on occupancy queries (white blur indicates uncertainty) using a form of Marching cubes.

formations [73]. But it remains limited in applicability.

To use richer priors, learning-based methods have been proposed, using explicit shape representations. Voxel-based approaches leverage a regular grid structure, extending 2D image-based techniques to 3D, but suffer from resolution limitations due to large memory consumption [20, 68, 111]. Directly generating a mesh with a neural network remains difficult [33] and is limited in practice to template deformation [35]. Some forms of implicit representations have been used for point cloud generation, but providing much weaker geometrical and topological information [29, 55, 119].

More success has been achieved with explicitly-designed implicit representations, where the network encodes a function $\mathbb{R}^3 \rightarrow \mathbb{R}$ expressing a volume occupancy [15, 69] or a distance to the surface [70, 77]. Such models require no discretization and can address arbitrary topologies. More precisely, discretization only occurs at mesh generation stage, using an algorithm such as the Marching cubes [63]. Yet, due to fully-connected architectures that lack translational equivariance, most existing approaches only operate on a single object and cannot apply to arbitrary scenes.

A few recent methods [17, 18, 22, 45, 80, 102], however, obtain a form of translational equivariance via Convolutional Neural Networks (CNNs). At least in theory, they can thus scale to larger scenes, possibly benefiting both from local and non-local information. But they operate on a voxelized discretization whose vertices may be far from the input point cloud. They thus lose the direct connection with points sampled on the surface of objects. They are also suboptimal in that the features or latent vectors holding the occupancy or distance information are more or less uniformly distributed in space rather than focused where difficult decisions have to be made, i.e., near the surface.

Our approach, based on point convolution, overcomes these issues. It is illustrated on Fig. 2. Our contributions are:

- We attach features representing the implicit function to input points. Not only does it preserve point positions until later processing stages, rather than abstract them away too soon, but it concentrates the information to learn where it matters the most: close to the surface.
- We compute features using point convolution, which yields a natural coverage and scalability to scenes of arbitrary size. (Rather than tailor yet another specific network architecture, we rely on a general point convolution backbone, which offers prospects for improvement when better point convolutions are designed.)
- Rather than relying on hand-designed forms of averaging, we extend prior learning to interpolation, which we apply to query-relative features rather than global features, as others do, as it leads to better results.
- We propose an efficient test-time augmentation to treat inputs of high density or large size.
- While simple, our approach outperforms other methods both on object and scene datasets, yielding finer details. It is robust to domain shift (training on objects, testing on scenes) and faster than methods that overfit to a scene or infer from scratch for each query.

2. Related work

2.1. 3D representations

Voxels have been a natural choice for learning to represent 3D volumes [20, 68, 111, 113–115]. However, they come with a cubic complexity in space, leading to coarse discretizations due to memory constraints. Multi-scale refinement [23, 39] and sparsity-based octrees [84, 85, 98] only partly reduce the impact of conforming to a 3D grid.

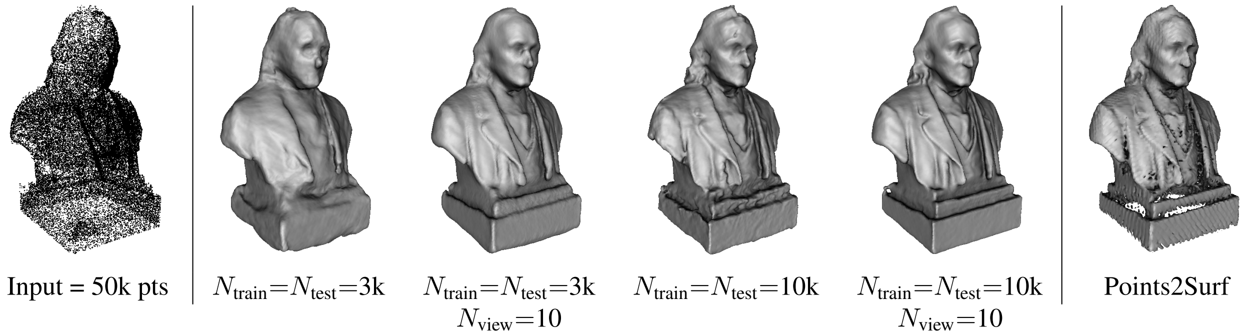

Figure 3. **Real World.** Model from Real World reconstructed by POCO in different settings and by Points2Surf

Points clouds are also produced as a sparse 3D representation, with various density and sampling distribution [1, 29, 55, 65, 108, 119, 120]. Point processing and generation do not suffer from the complexity and discretization induced by 3D grids; yet, the range of applications is limited regarding representing actual surfaces and volumes.

Meshes are a preferred representation for many uses, such as visualizations and simulations, but they are harder to directly produce from a neural network (vertex regression and face construction) [74]. Most existing approaches thus prefer to operate by deforming geometric primitives [35, 56, 104, 107], voxelized approximations [33, 54] or learned templates [36, 46]. Rather than actually inferring vertices, a mesh can also be extracted from labels inferred on a Delaunay tetrahedralization [64].

Implicit representations rely on a neural network to model a function expressing the occupancy of a given 3D point [15,69] or its distance to the surface, either signed [34, 70, 77], unsigned [18] or sign-agnostic [2, 8]. The signed or unsigned distance field (SDF, UDF) is often truncated (TSDF, TUDF) and estimated via a multi-layer perceptron (MLP). The isosurface can then be extracted from this occupancy or distance field with various methods such as Marching cubes [63]. Whereas voxels, points and mesh vertices are intrinsically discrete representations, implicit representations offer a virtually infinite resolution. Moreover, while mesh-based approaches struggle to enforce watertightness, to limit self-intersections and to address complex topologies (non genus-0), meshes reconstructed from implicit representations are guaranteed to be watertight and have no self-intersections. Besides, they can easily model arbitrary complex topologies. These advantages may explain the recent success of this representation, including to model 3D shapes from images without 3D supervision [60, 75, 93], with texturing [76] or specific rendering [61]. Departing from occupancy or distance fields, ShapeGF [10] models a shape by learning the gradient field of its log-density, then samples points on high likelihood regions of the shape and meshes them. Other work also study the decomposition of shapes and implicit surfaces into parts [26,31,32,44,78,101], possibly overfitting networks to generate or render a single object or scene [59, 67, 92, 95, 109, 118, 121].

Scalability, however, is an issue for all these methods. While they can encode reasonably well one object or a class of objects, they cannot cope with the variability and size of an arbitrary scene involving several objects. Even considering a single object and assuming a powerful decoder, the encoding of a single or a few latent vectors hardly can develop into detailed shape information. Using periodic activation functions [92, 96] or adding a 2D convolutional component on input images [86, 116] helps, but is not enough.

A solution is to split the input points on a regular 3D grid and to optimize one latent vector per voxel [11] (DeepLS), possibly from overlapping input patches. Patch splitting can also be irregular and optimization-driven to favor self-similarities, with a global post-optimization to flip inconsistent local signs [121] (SAIL-S3). But whether these methods optimize only the latent vectors or a whole network as well, for patch decoding, they make surface reconstruction significantly slower, leading to reduced test sets.

Besides, these methods rely on fully-connected architectures whereas, we believe, convolutions, and in particular point convolutions [5, 7, 40, 52, 62, 66, 100, 105, 112, 117], are the key to scalability and increased details.

2.2. Convolutions for implicit representations

LIG [45] divides the input point cloud along a regular 3D grid to create 3D patches and capture local geometric shapes shared by several objects at a medium scale. For each of these patches, a 3D CNN then computes a local feature vector, which goes through a reduced IM-NET [16] for SDF decoding. However, later on, only the learned decoder is exploited; no local embedding is inferred. Given an input point cloud, latent vectors on the grid are optimized from scratch to minimize an objective function similar to the loss used for training. LIG additionally requires to be provided with oriented normals to make use of points known to be inside or outside the shape. This, however, may introduce artificial back-faces, which can partly be addressed in a postprocessing stage. In contrast, we can work without normals, we directly operate with convolutions on surface points rather than on a regular grid, and we directly

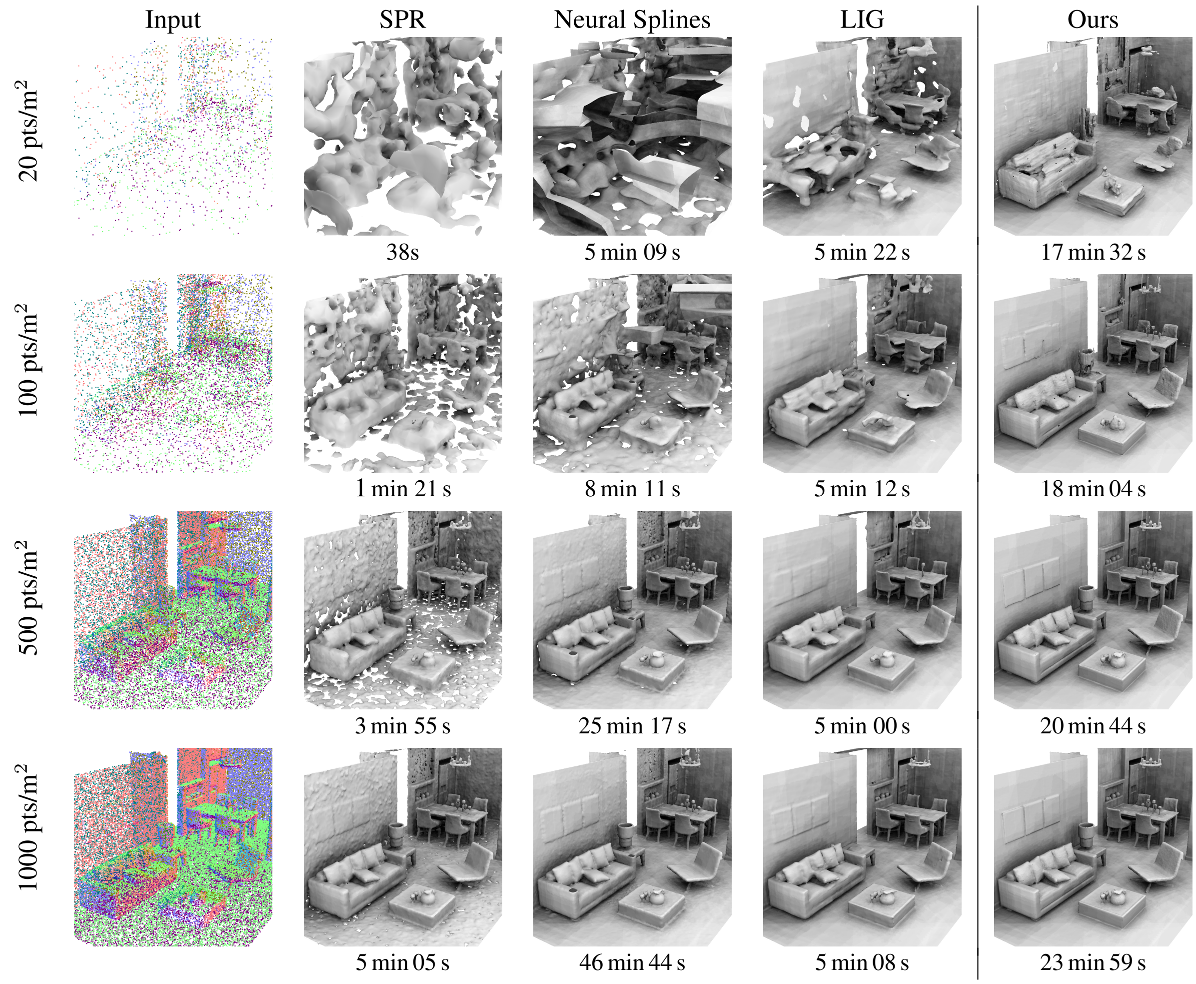

Figure 4. **SceneNet.** Partial view of a full scene. The color on point clouds indicates the orientation of normals.

use inferred embeddings without any heavy optimization.

IF-Net [17] introduces a multi-scale pyramid of 3D convolutional encoders aligned on a discrete voxel grid and trained on voxels at different scales. The occupancy of a query point is decided by a decoder taking as input the interpolated features extracted at this point for each pyramid level. In contrast, we do not discretize into voxels; we use point cloud convolution. Also, we learn how to interpolate the latent vectors rather than use a basic trilinear interpolation. Last, we provide results on scenes, not just on objects.

NDF [18] uses the same multi-scale encoding as IF-Net but relies on a UDF rather than occupancy for decoding. It allows the generation of very dense points clouds that can directly be meshed into possibly open surfaces.

SG-NN [22] uses a sparse 3D convolution [19] to learn a TSDF in a self-supervised setting, training for completion from partial scans. In contrast, we use point convolution and infer occupancy rather than SDF, which is easier to learn.

ConvONet [80] also uses a grid-based convolution, training an autoencoder that predicts occupancy. (It generalizes ONet [69], which only uses a single encoding and full connection.) For input point clouds, the encoder is a shallow PointNet [82] operating on points rather than on a voxelized discretization, and the decoder is a 3D U-Net [21]. The occupancy of a 3D point is inferred from a trilinear interpolation of grid features. Besides 3D convolution, variants based on a combination of 2D convolutions in a few spatial directions are proposed. DP-ConvONet [57] is a variant that considers a dynamic family of such directions. SA-ConvONet [97] overfits a pre-trained ConvONet model on the input using a sign-agnostic optimization of the implicit field. It improves accuracy at the cost of computation time.

As inference applies to grids, whose vertices or centers may be far from input points, the above methods lose the direct connection with the input surface samples. They are also suboptimal in that the latent vectors holding the information are uniformly distributed in space rather than concentrated where it matters the most, i.e., near the surface. To address these issues, we use point convolution and compute latent vectors at each input point. We then interpolate occupancy decisions of nearest neighbors using learned weights.

AdaConv [102] uses point convolution like us but ag-

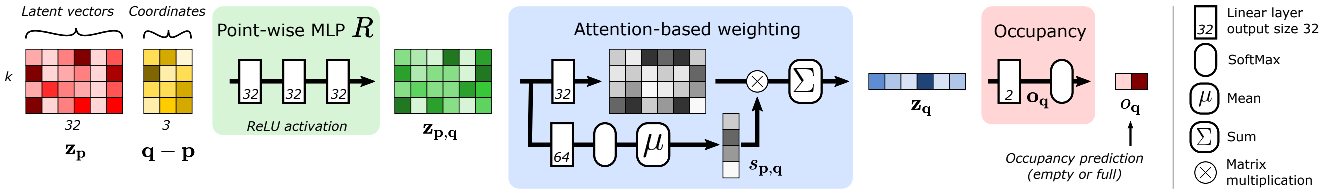

Figure 5. **Architecture.** The latent vectors $\mathbf{z_p}$ (red squares) produced by the convolution-based encoder E of k neighboring points $\mathbf{p}$ of a query point $\mathbf{q}$ are: (1) augmented with the relative query position $\mathbf{q} - \mathbf{p}$ (yellow squares), (2) re-encoded with a 3-layer point-wise MLP R (green frame) into relative latent vectors $\mathbf{z_{p,q}}$ (green squares), (3) combined (blue frame) with inferred weights $s_{\mathbf{p,q}}$ (gray squares) into a latent vector $\mathbf{z_q}$ (blue squares), (4) decoded with a linear layer D (pink frame) into occupancy logits $\mathbf{o_q}$ and probablities $o_\mathbf{q}$ (pink squares).

gregates multi-scale information on an adaptive voxel grid, while we attach features to points, closer to the surface. Besides, it requires oriented normals, contrary to us.

RetrievalFuse [91] splits a scene along a regular grid and encodes each 3D chunk as a latent vector via convolutional layers. But rather than using them for decoding, it retrieves similar chunks from the training set and combines their distance field to create a surface, enhancing the completion capability. In contrast, we are fully convolutional and the implicit function is directly obtained by interpolating inferred features, without the need to maintain the dataset samples used for training and with more generalization capacity.

Points2Surf [27] collects, for each query point, both a patch of neighbors (which gives a convolution flavor) and globally-sampled input points to help to provide a sign to the local distance field. The local patch and the global subsampling go through an MLP to create latent vectors that are concatenated and decoded into a signed distance. In contrast, we directly get non-local information as our receptive field is much larger. Besides, we are faster as we only compute a limited number of latent vectors (one per input point) that we later use for interpolation given a query point, while Points2Surf samples local+global points and goes through the whole encoder for each query point, i.e., a large number of times, that grows with the Marching-cubes resolution.

To infer occupancy or distance of a query point, methods that compute several latent vectors for a single object or scene either select the most appropriate latent vector to decode, typically in a multi-scale grid [102], or interpolate the latent vectors of query neighbors [17, 18, 45, 57, 80, 97]. We perform interpolation too, based on features computed on input points. However, given a query point, we do not interpolate the features themselves but the occupancy logits, as our experiments shows it leads to better results. Besides, we use a learned interpolation rather than the usual tri-linear interpolation [17, 18, 45, 57, 80, 97] or the inverse-distance distance weighting [83]. Although different in nature, learning has also been used in [91] to blend retrieved chunks.

3. Our method

Goal. Given as input a set of 3D points $\mathcal{P}$ sampled on a surface, possibly with noise, our goal is to construct a continuous function $\omega : \mathbb{R}^3 \to [0, 1]$ indicating the probability of occupancy $o_\mathbf{q} = \omega(\mathbf{q})$ at any given query point $\mathbf{q} \in \mathbb{R}^3$. We learn this function with a neural network using data consisting of point clouds sampled in the whole space and labeled with 0 (in empty space) or 1 (within the shape). The surface of the shape can then be extracted as the isosurface of the implicit function ω with occupancy level 0.5.

Overview. Our method consist of the following steps:

1. We encode input points $\mathbf{p} \in \mathcal{P}$ into latent vectors $\mathbf{z_p}$.
2. Given an arbitrary query point $\mathbf{q}$, we consider a neighborhood $\mathcal{N}_\mathbf{q}$ of input points in $\mathcal{P}$ to interpolate from.
3. For each neighbor $\mathbf{p} \in \mathcal{N}_\mathbf{q}$, we construct a relative latent vector $\mathbf{z_{p,q}}$ from $\mathbf{z_p}$ and local coordinates $\mathbf{q} - \mathbf{p}$.
4. We extract significance weights $s_{\mathbf{p,q}}$ to sum the relative latent vectors $\mathbf{z_{p,q}}$: $\mathbf{z_q} = \sum_{\mathbf{p} \in \mathcal{N}_\mathbf{q}} s_{\mathbf{p,q}}\, \mathbf{z_{p,q}}$.
5. We decode the resulting feature vector $\mathbf{z_q}$ as two full-empty logits $\mathbf{o_q}$, and turn them into probabilities $o_\mathbf{q}$.

These steps, illustrated on Figure 5, are detailed below.

Absolute encoding. A point convolution first produces a latent vector $\mathbf{z_p} = E(\mathbf{p})$ for each input point $\mathbf{p} \in \mathcal{P}$. The encoder E can be implemented by any point cloud segmentation backbone, only changing the last layer to yield a vector of some chosen dimension n as the size of vectors $\mathbf{z_p}$. (In our experiments, the convolution backbone is FKAConv [7] and $n = 32$.) To also use normals (optionally), the input points are just augmented with the 3 normal coordinates.

Query neighborhood. Given an arbitrary query point $\mathbf{q}$ (when training or to predict occupancy at test time), we construct a set of neighbors $\mathcal{N}_\mathbf{q}$ from input points $\mathcal{P}$. (In our experiments, $\mathcal{N}_\mathbf{q}$ is the k nearest neighbors of $\mathbf{q}$, with $k = 64$.)

Relative encoding. We augment the latent vector $\mathbf{z_p}$ of each neighbor $\mathbf{p} \in \mathcal{N}_\mathbf{q}$ with the local coordinates $\mathbf{q} - \mathbf{p}$ of query point $\mathbf{q}$ relatively to $\mathbf{p}$. These augmented latent vectors are then processed by an MLP R to produce relative latent vectors $\mathbf{z_{p,q}} = R(\mathbf{z_p} \,\|\, \mathbf{q} - \mathbf{p})$, where $\|$ is the concatenation. (In our experiments, $\mathbf{z_p}$ and $\mathbf{z_{p,q}}$ have size $n = 32$.)

Feature weighting. As PRNet [106], we observe that the norm of embeddings $\mathbf{z_{p,q}}$ tends to correlate with their significance, hinting how much an input point $\mathbf{p}$ matters for deciding the occupancy of query point $\mathbf{q}$, given $\mathbf{p}$'s neighbors and the position of $\mathbf{q}$ w.r.t. $\mathbf{p}$. We use it to infer significance weights for relative latents vectors $\mathbf{z_{p,q}}$. Concretely, we use an attention mechanism (blue frame in Fig. 5): The relative embeddings $\mathbf{z_{p,q}}$ go through a linear layer parameterized by a weight vector $\mathbf{w}$, also of size n, producing relative weights $w_{\mathbf{p,q}} = \mathbf{w} \odot \mathbf{z_{p,q}}$, that are normalized by softmax over $\mathcal{N}_\mathbf{q}$ into positive interpolation weights $s_{\mathbf{p,q}}$ summing to 1. We actually use a multi-head strategy to obtain a form of ensembling. We learn h independent linear layers, parameterized by h corresponding weight vectors $(\mathbf{w}_i)_{i=1..h}$, producing h relative weights $w_{\mathbf{p,q},i} = \mathbf{w}_i \odot \mathbf{z_{p,q}}$, that are finally softmaxed as $s_{\mathbf{p,q},i}$ and averaged as $s_{\mathbf{p,q}} = \frac{1}{n}\sum_i s_{\mathbf{p,q},i}$. (In our experiments, we use $h = 64$.)

Interpolation. The feature vector $\mathbf{z_q}$ at query point $\mathbf{q}$ is interpolated from the relative latent vectors $\mathbf{z_{p,q}}$ of neighbors $\mathbf{p}$, as the weighted sum $\mathbf{z_q} = \sum_{\mathbf{p}\in\mathcal{N}_\mathbf{q}} s_{\mathbf{p,q}}\, \mathbf{z_{p,q}}$.

Decoding. A linear layer D decodes the feature vector $\mathbf{z_q}$ into occupancy scores $\mathbf{o_q} = D(\mathbf{z_q})$, which is a two-logit vector classifying position $\mathbf{q}$ as occupied or not, that is then turned via softmax into occupancy probabilities $o_\mathbf{q}$.

Loss function. To train the network, we use a cross-entropy loss that penalizes wrong occupancy predictions. Please note that using a binary cross-entropy, like in IF-Net [17] or ConvONet [80], leads to identical results.

4. Refinements

Adapting to high density. We train our network with a fixed number N_train of input points for easy mini-batching. (In our experiments, $N_\text{train} = 3\text{k}$ or 10k.) At test time, if the surface is more densely sampled, the receptive field of the backbone may lack enough global context to decide which side of the surface is full or empty, unless oriented normals are also provided with points. A way to broaden enough the receptive field is to downsample the input point cloud, but it then naturally leads to a loss of details.

To reduce this effect, we rely on test-time augmentation (TTA) [50], which can be seen as a form of ensembling: we average several runs on different subsamples. However, aggregating final results, as often done in TTA [90], would be very time consuming in our case as we would have to do it to answer the occupancy of each query, basically multiplying the inference running time by the number of subsamples.

Instead, we perform TTA at latent vector level, thus running several times only the first step of our approach (absolute encoding), before query decoding. It depends on the number of input points (to attach a latent vector on), rather than on the number of query points, which is much larger. Concretely, we randomly create enough subsamples so that each point $\mathbf{p} \in \mathcal{P}$ is seen at least N_view times, and average each $\mathbf{z_p}$ over all samples. (In experiments, $N_\text{view} = 10$.) The subsamples are randomly generated by sequentially picking a point $\mathbf{p} \in \mathcal{P}$ with a priority that is the opposite of the number of times $\mathbf{p}$ appears in previous subsamples.

Adapting to large size. As our method is convolutional, it naturally adapts to input point clouds $\mathcal{P}$ of arbitrary size. Yet, while $\mathcal{P}$ may contain millions of points, GPU memory limits in practice the number of points N_test that can be treated together by the backbone. (We use $N_\text{test} = 100\text{k}$.)

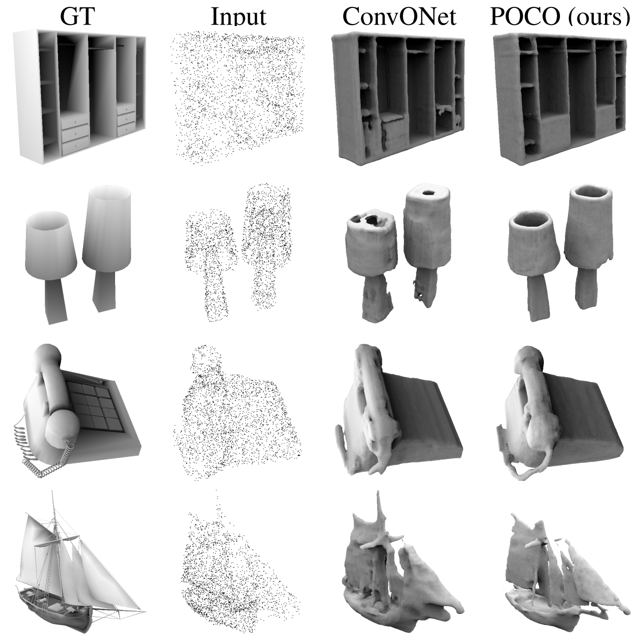

Figure 6. **ShapeNet.** The methods train and test on 3k noisy pts.

As with semantic segmentation [7], we can use a sliding-window with overlapping chunks of $\mathcal{P}$ of maximum size N_test. Alternatively, as above, we can make subsamples of $\mathcal{P}$ by iteratively picking a low-priority point $\mathbf{p} \in \mathcal{P}$ and its $N_\text{test}-1$ nearest neighbors. (In our experiments, $N_\text{view} = 3$.)

Scene scaling. At inference time, the scale of the input point cloud may differ from the scales in the training set. As point-based backbones can be sensitive to variations of scale and density, we rescale the input such that the average distance between a point and its nearest neighbor is the same both in the training set and in the test point cloud.

5. Experiments

We experiment both on objects and scenes, in different point density regimes, with or without normal information depending on the baseline methods we compare with.

Because existing methods often perform well in some setting but not in others, most published papers tend to evaluate on different datasets or in specific configurations: number of train/test points, added noise, normals, generalization, etc. Some methods are also too slow to be evaluated on full datasets and report results only on dataset fractions. To be fair with these methods, we evaluate in their setting (when enough information is provided to do so) rather than impose them specific settings. It also illustrates the ability of our method to adapt to various configurations.

5.1. Datasets, baselines and metrics

ShapeNet [13], as pre-processed by [20], contains watertight meshes of shapes in 13 classes, with train/val splits and 8500 objects for testing. As [80], we sample 3000 points from each mesh (at each epoch) and apply a Gaussian noise with zero mean and standard deviation 0.05.

Synthetic Rooms [80] has 5000 synthetic scenes with random walls and populated with ShapeNet objects. We use [80]'s protocol for sampling 10k pts on the meshes to create train/val/test data, with noise as for ShapeNet. Shapes are scenes in terms of complexity, objects in terms of size.

ABC [48] is a set of CAD models, mainly mechanical parts. We use splits and point preprocessing from [27]: 4950 shapes for training, 100 for validation and 100 for testing.

Famous [27] contains 22 shapes of various origins, e.g., from the Stanford 3D Scanning Repository [49].

Thingi10k [122], as prepared by [27], has 100 shapes.

SceneNet [37, 38] is a synthetic dataset of indoor scenes. Data prepared in the same way as [41] yield 34 scenes.

MatterPort3D [12] has indoor scenes too. We use the same 2 scenes as prepared and used by [97]: with 65k pts.

Baselines are drawn among the state-of-the-art methods presented in Section 2.2. We also compare to SPR [47], a popular, non-learning-based reconstruction method that requires oriented normals (which is a strong hypothesis) and, possibly, a trimming parameter tuning (factor 6 in Tab. 4).

Our method, unless otherwise stated, uses the FKAConv backbone [7], feature size $n = 32$ as in ConvONet [80] or LIG [45], $k = 64$ neighbors, $h = 64$ interpolation heads, and does not use normals nor TTA.

Mesh Generation, for implicit functions, is done with the Marching cubes [63] with resolution 256^3 for objects, 1 cm for SceneNet, 2 cm for MatterPort3D.

Metrics. We use the following common metrics: volumetric **IoU**, symmetric Chamfer L1-distance $\times 10^2$ (**CD**), normal consistency (**NC**), i.e., mean absolute cosine of normals in one mesh and normals at nearest neighbors in the other mesh, and F-Score [99] with threshold value 1% (**FS**). Surface metrics are approximated by point sampling.

5.2. Alternative and ablation studies

To justify our algorithmic choices, we experiment on ShapeNet in generalization mode, training on chairs but evaluating on all the classes. We use the same train/test split as [69, 80], evaluating on 130 shapes (10 per class).

As can be seen in Table 1(a), the convolutional backbone FKAConv [7] is more efficient by a large margin than the PointNet-based segmentation network with residual connections [69, 82], which loses small scale information [83].

Though interpolating from $k = 64$ neighbors rather than $k = 128$ has a slightly worse CD and NC (cf. Tab. 1(b)), it has a better IoU and it is faster; we use this setting in the following. We note we get better results with a multi-head attention (using $h = 64$ rather than $h = 1$) and when interpolating relative rather than global features (cf. Tab. 1(c)).

(a) Point backbone			IoU↑	CD↓	NC↑
Residual PointNet			0.661	10.583	0.817
FKAConv			0.882	4.069	0.929
(b) No. interpolation neighbors			IoU↑	CD↓	NC↑
$k = 1$			0.799	6.951	0.867
$k = 8$			0.819	6.723	0.912
$k =$ **64**			0.882	4.069	0.929
$k = 128$			0.876	3.611	0.930
(c) Interpol. features	*glob.*	*rel.*	IoU↑	CD↓	NC↑
Max		✓	0.882	4.069	0.929
Mean		✓	0.883	3.703	0.933
Mean	✓		0.854	5.331	0.902
Inverse distance		✓	0.877	3.947	0.935
Inverse distance	✓		0.851	4.724	0.912
Single-head attention		✓	0.879	3.686	0.934
Multi-head attention		✓	0.895	3.702	0.938

Table 1. **Alternative study.** We train on ShapeNet chairs, without normals, 3k input points, with noise, and unless otherwise stated, FKAConv backbone, $k = 64$ neighbors, and max interpolation. We test on 10 models from each of the 13 ShapeNet classes. We interpolate either global features $\mathbf{z_p}$ or relative features $\mathbf{z_{p,q}}$.

Last, Tab. 2 and Fig. 3 show the benefits of the TTA strategy with models trained with 3k and 10k points on ABC.

5.3. Reconstruction

Reconstruction without normals. Because of long running times, only a few published methods evaluate on the whole ShapeNet dataset. We outperform them on all metrics with a significant margin (Table 3). We reconstruct finer details (Figure 6) and we do not have the same tendency as ConvONet to fill volumes; we can instead generate more easily thin surfaces, which explain our superior IoU. We outperform other methods as well on Synthetic Rooms (Table 4), where also we capture much finer details.

Generalization. LIG is specifically designed for scalability and generality. It learns to reconstruct small shape patches from a given dataset, and then applies it to any new object or scene. Points2Surf is a patch-learning method too, although its requirement for a global view of the input and its running time make it less suited for scene reconstruction.

We compare to LIG, training both methods on ShapeNet objects (with normals as LIG requires them) and testing on SceneNet. We generalize better (Tab. 5) at all densities, capturing finer details and not erasing thin objects (Fig. 4).

We compare to Points2Surf, training on ABC in the same setting. We outperform Points2Surf on most of their settings (Tab. 2), both on ABC and when generalizing to Famous and Thingi10k. Points2Surf outperforms POCO only

Method	Test set / Noise setting	ABC (100 shapes) no-n.	var-n.	max-n.	Famous (22 shapes) no-n.	med-n.	max-n.	sparse	dense	Thingi10k (100 shapes) no-n.	med-n.	max-n.	sparse	dense
DeepSDF	[77]	8.41	12.51	11.34	10.08	9.89	13.17	10.41	9.49	9.16	8.83	12.28	9.56	8.35
AtlasNet	[35]	4.69	4.04	4.47	4.69	4.54	4.14	4.91	4.35	5.29	5.19	4.90	5.64	5.02
SPR	[47]	2.49	3.29	3.89	1.67	1.80	3.41	2.17	1.60	1.78	1.81	3.23	2.35	1.57
Points2Surf	[27]	1.80	2.14	2.76	1.41	1.51	2.52	1.93	1.33	1.41	1.47	2.62	2.11	1.35
POCO	$N_{\text{train}}=N_{\text{test}}=3\text{k}$	1.87	2.26	2.90	1.56	1.75	2.99	1.99	1.70	1.47	1.64	3.21	2.00	1.55
POCO	$N_{\text{train}}=N_{\text{test}}=3\text{k}$, $N_{\text{view}}=10$	1.77	2.10	2.68	1.40	1.54	2.93	1.78	1.50	1.39	1.46	2.55	1.83	1.40
POCO	$N_{\text{train}}=N_{\text{test}}=10\text{k}$	1.72	2.15	2.72	1.57	1.61	3.04	1.92	1.57	1.50	1.57	2.82	2.08	1.51
POCO	$N_{\text{train}}=N_{\text{test}}=10\text{k}$, $N_{\text{view}}=10$	1.70	2.01	2.50	1.34	1.50	2.75	1.89	1.50	1.35	1.44	2.34	1.95	1.38

Table 2. **ABC, Famous, Thingi10k.** Training on ABC shapes with 10 scans, variable Gaussian noise (σ uniformly picked in $[0, 0.05L]$, L largest box length). Chamfer distance $\times$ 100 on ABC, Famous and Thingi10k test sets, as prepared by [27]: 'no-n.' (no noise), 'var-n.' (variable noise, as training), 'max-n.' ($\sigma = 0.05L$), 'med-n.' ($\sigma = 0.01L$), 'sparse' (5 scans), 'dense' (30 scans). Only SPR uses normals.

Method		IoU↑	CD↓	NC↑	FS↑
ONet	[69]	0.761	0.87	0.891	0.785
ConvONet	[80]	0.884	0.44	0.938	0.942
DP-ConvONet	[57]	0.895	0.42	0.941	0.952
POCO	(ours)	0.926	0.30	0.950	0.984

Table 3. **ShapeNet.** The methods train and test on 3k noisy pts.

Method		IoU↑	CD↓	NC↑	FS↑
ONet	[69]	0.475	2.03	0.783	0.541
SPR	[47]	-	2.23	0.866	0.810
SPR trimmed	[47]	-	0.69	0.890	0.892
ConvONet	[80]	0.849	0.42	0.915	0.964
DP-ConvONet	[57]	0.800	0.42	0.912	0.960
POCO	(ours)	0.884	0.36	0.919	0.980

Table 4. **Synthetic Rooms.** Learning-based methods train and test on 10k noisy pts. Only SPR uses normals. Numbers from [57, 80].

pts/m^2	Method		CD↓	NC↑	FS↑
20	SPR	[47]	5.27	0.772	0.4392
	Neural Splines	[110]	3.76	0.815	0.6563
	LIG	[45]	1.52	0.923	0.8757
	POCO	(ours)	0.84	0.960	0.9600
100	SPR	[47]	1.96	0.853	0.7709
	Neural Splines	[110]	1.15	0.931	0.9228
	LIG	[45]	0.97	0.961	0.9643
	POCO	(ours)	0.57	0.984	0.9941
500	SPR	[47]	0.86	0.936	0.9787
	Neural Splines	[110]	0.60	0.982	0.9958
	LIG	[45]	0.87	0.975	0.9773
	POCO	(ours)	0.53	0.992	0.9987
1000	SPR	[47]	0.73	0.967	0.9957
	LIG	[45]	0.84	0.978	0.9750
	POCO	(ours)	0.53	0.993	0.9987
	Oracle	(4M pts)	0.50	0.995	0.9998

Table 5. **SceneNet.** LIG and POCO train on ShapeNet with 10k pts with normals (no noise). Test is on SceneNet with normals (no noise). Neural Splines uses a grid size of 1024, 10k Nyström samples, 8×8×8 chunks. Numbers differ from [45] as we had to regenerate the unavailable watertight meshes: we used [41] with resolution 500k, higher than in [45], getting finer and thinner details where CAD models have no volume; as [45], we ignore scenes with volume-to-area ratio > 0.13, getting 34 scenes. 'Oracle' is the ground truth evaluated against itself (two different samplings).

on very noisy or dense inputs, and only with a small margin.

Scene reconstruction without normals. We compare to SA-ConvONet on MatterPort3D scenes (Fig. 1) in their same actual setting (downsampling to 65536 pts). Our reconstruction is less smooth than SA-ConvONet but has finer details. As SA-ConvONet overfits many networks at inference time on top of ConvONet, it is notably slower too.

5.4. Discussion and limitations

Our approach is suited both for single-object and whole-scene reconstruction. However, although it can cope with a substantial variation of point density, it cannot complete shapes when large parts are missing. Apart from a few methods like [22, 24, 25, 91], only object-targeted methods can presently do it, for classes known at training time, but they cannot reconstruct scenes at all.

Inferring surface orientation, when normals are not provided, requires wide context information. But a high density may reduce the receptive field, yielding orientation failures and artifacts. Our TTA only partly addresses the issue; handling it directly at backbone level would be better.

Nevertheless, POCO reaches the state of the art for both object and scene reconstruction, with or without oriented normals. It shows good generalization capabilities to shapes and scenes that are very different from the training set.

More details on the method and on the experiments are in the supplementary material.

Acknowledgments to Gilles Puy for fruitful discussions.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3D point clouds. In *International Conference on Machine Learning (ICML)*, 2018. 3

[2] Matan Atzmon and Yaron Lipman. SAL: Sign agnostic learning of shapes from raw data. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[3] J.P. Bauchet and F. Lafarge. Kinetic shape reconstruction. *ACM Transactions on Graphics (TOG)*, 39(5), 2020. 1

[4] Matthew Berger, Andrea Tagliasacchi, Lee M. Seversky, Pierre Alliez, Joshua A. Levine, Andrei Sharf, and Claudio T. Silva. State of the art in surface reconstruction from point clouds. In *Eurographics Conference (EG)*, 2014. 1

[5] Alexandre Boulch. ConvPoint: Continuous convolutions for point cloud processing. *Computers & Graphics (CG)*, 88:24–34, 2020. 3

[6] Alexandre Boulch, Martin de La Gorce, and Renaud Marlet. Piecewise-planar 3D reconstruction with edge and corner regularization. *Computer Graphics Forum (CGF)*, 33(5):55–64, 2014. 1

[7] Alexandre Boulch, Gilles Puy, and Renaud Marlet. FKAConv: Feature-kernel alignment for point cloud convolution. In *Asian Conference on Computer Vision (ACCV)*, 2020. 3, 5, 6, 7

[8] Alexandre Boulch, Gilles Puy, and Renaud Marlet. NeeDrop: Self-supervised shape representation from sparse point clouds using needle dropping. In *International Conference on 3D Vision (3DV)*, 2021. 3

[9] A. Bódis-Szomorú, H. Riemenschneider, and L. Van Gool. Efficient volumetric fusion of airborne and street-side data for urban reconstruction. In *International Conference on Pattern Recognition (ICPR)*, 2016. 1

[10] Ruojin Cai, Guandao Yang, Hadar Averbuch-Elor, Zekun Hao, Serge Belongie, Noah Snavely, and Bharath Hariharan. Learning gradient fields for shape generation. In *European Conference on Computer Vision (ECCV)*, 2020. 3

[11] Rohan Chabra, Jan Eric Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, S. Lovegrove, and Richard A. Newcombe. Deep local shapes: Learning local SDF priors for detailed 3D reconstruction. In *European Conference on Computer Vision (ECCV)*, 2020. 3

[12] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niebner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from RGB-D data in indoor environments. In *International Conference on 3D Vision (3DV)*, pages 667–676. IEEE, 2017. 7

[13] A.X. Chang, T.A. Funkhouser, L.J. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. ShapeNet: An information-rich 3D model repository, 2015. arXiv preprint arXiv:1512.03012. 7

[14] Anne-Laure Chauve, Patrick Labatut, and Jean-Philippe Pons. Robust piecewise-planar 3D reconstruction and completion from large-scale unstructured point data. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1261–1268, 2010. 1

[15] Z. Chen and H. Zhang. Learning implicit fields for generative shape modeling. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2, 3

[16] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[17] J. Chibane, T. Alldieck, and G. Pons-Moll. Implicit functions in feature space for 3D shape reconstruction and completion. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 4, 5, 6

[18] Julian Chibane, Aymen Mir, and Gerard Pons-Moll. Neural unsigned distance fields for implicit function learning. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020. 2, 3, 4, 5

[19] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4D spatio-temporal ConvNets: Minkowski convolutional neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 4

[20] Christopher B. Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3D-R2N2: A unified approach for single and multi-view 3D object reconstruction. In *European Conference on Computer Vision (ECCV)*, 2016. 2, 7

[21] Ö. Çiçek, A. Abdulkadir, S.S. Lienkamp, T. Brox, and O. Ronneberger. 3D U-Net: Learning dense volumetric segmentation from sparse annotation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2016. 4

[22] Angela Dai, Christian Diller, and Matthias Nießner. SG-NN: Sparse generative neural networks for self-supervised scene completion of RGB-D scans. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 4, 8

[23] A. Dai, C.R. Qi, and M. Nießner. Shape completion using 3D-encoder-predictor CNNs and shape synthesis. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[24] Angela Dai, Charles Ruizhongtai Qi, and Matthias Nießner. Shape completion using 3D-encoder-predictor CNNs and shape synthesis. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 8

[25] Angela Dai, Daniel Ritchie, Martin Bokeloh, Scott Reed, Jürgen Sturm, and Matthias Nießner. ScanComplete: Large-scale scene completion and semantic segmentation for 3D scans. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 8

[26] Boyang Deng, Kyle Genova, Soroosh Yazdani, Sofien Bouaziz, Geoffrey Hinton, and Andrea Tagliasacchi. CvxNet: Learnable convex decomposition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[27] Philipp Erler, Paul Guerrero, Stefan Ohrhallinger, Niloy J. Mitra, and Michael Wimmer. Points2Surf: Learning implicit surfaces from point clouds. In *European Conference on Computer Vision (ECCV)*, 2020. 5, 7, 8

[28] Lafarge F. and Alliez P. Surface reconstruction through point set structuring. In *Eurographics Conference (EG)*, 2013. 1

[29] H. Fan, H. Su, and L.J. Guibas. A point set generation network for 3D object reconstruction from a single image. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 3

[30] Ran Gal, Ariel Shamir, Tal Hassner, Mark Pauly, and Daniel Cohen-Or. Surface reconstruction using local shape priors. In *Eurographics Symposium on Geometry Processing (SGP)*, page 253–262, 2007. 1

[31] K. Genova, F. Cole, A. Sud, A. Sarna, and T.A. Funkhouser. Local deep implicit functions for 3D shape. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[32] K. Genova, F. Cole, D. Vlasic, A. Sarna, W.T. Freeman, and T. Funkhouser. Learning shape templates with structured implicit functions. In *International Conference on Computer Vision (ICCV)*, 2019. 3

[33] G. Gkioxari, J. Malik, and J. Johnson. Mesh R-CNN. In *International Conference on Computer Vision (ICCV)*, 2019. 2, 3

[34] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *International Conference on Machine Learning (ICML)*, 2020. 3

[35] T. Groueix, M. Fisher, V.G. Kim, B.C. Russell, and M. Aubry. AtlasNet: A papier-mâché approach to learning 3D surface generation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 3, 8

[36] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. 3D-CODED: 3D correspondences by deep deformation. In *European Conference on Computer Vision (ECCV)*, 2018. 3

[37] Ankur Handa, Viorica Patraucean, Vijay Badrinarayanan, Simon Stent, and Roberto Cipolla. Understanding real world indoor scenes with synthetic data. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4077–4085, 2016. 7

[38] Ankur Handa, Viorica Patraucean, Simon Stent, and Roberto Cipolla. SceneNet: An annotated model generator for indoor scene understanding. In *International Conference on Robotics and Automation (ICRA)*, 2016. 7

[39] C. Hane, S. Tulsiani, and J. Malik. Hierarchical surface prediction for 3D object reconstruction. In *International Conference on 3D Vision (3DV)*, 2017. 2

[40] Binh-Son Hua, Minh-Khoi Tran, and Sai-Kit Yeung. Pointwise convolutional neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[41] Jingwei Huang, Hao Su, and Leonidas Guibas. Robust watertight manifold surface generation method for ShapeNet models. *arXiv preprint arXiv:1802.01698*, 2018. 7, 8

[42] Michal Jancosek and Tomas Pajdla. Multi-view reconstruction preserving weakly-supported surfaces. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. 1

[43] Michal Jancosek and Tomas Pajdla. Exploiting visibility information in surface reconstruction to preserve weakly supported surfaces. *International Scholarly Research Notices*, 2014. 1

[44] T. Jeruzalski, B. Deng, M. Norouzi, J.P. Lewis, G.E. Hinton, and A. Tagliasacchi. NASA: neural articulated shape approximation. In *European Conference on Computer Vision (ECCV)*, 2020. 3

[45] C. Jiang, A. Sud, A. Makadia, J. Huang, M. Nießner, and T. Funkhouser. Local implicit grid representations for 3D scenes. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 3, 5, 7, 8

[46] A. Kanazawa, S. Tulsiani, A.A. Efros, and J. Malik. Learning category-specific mesh reconstruction from image collections. In *European Conference on Computer Vision (ECCV)*, 2018. 3

[47] M.M. Kazhdan and H. Hoppe. Screened Poisson surface reconstruction. *ACM Transactions on Graphics (TOG)*, 32(3), 2013. 1, 7, 8

[48] Sebastian Koch, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, and Daniele Panozzo. Abc: A big cad model dataset for geometric deep learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 7

[49] Venkat Krishnamurthy and Marc Levoy. Fitting smooth surfaces to dense polygon meshes. In *ACM Conference on Computer Graphics and Interactive Techniques (PACM CGIT)*, pages 313–324, 1996. 7

[50] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2012. 6

[51] P. Labatut, J. P. Pons, and R. Keriven. Robust and efficient surface reconstruction from range data. *Computer Graphics Forum (CGF)*, 2009. 1

[52] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. PointCNN: Convolution on X-transformed points. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2018. 3

[53] Yangyan Li, Xiaokun Wu, Yiorgos Chrysathou, Andrei Sharf, Daniel Cohen-Or, and Niloy J. Mitra. GlobFit: Consistently fitting primitives by discovering global relations. *ACM Transactions on Graphics (TOG)*, 30(4), 2011. 1

[54] Y. Liao, S. Donne, and A. Geiger. Deep marching cubes: Learning explicit surface representations. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[55] C. Lin, C. Kong, and S. Lucey. Learning efficient point cloud generation for dense 3D object reconstruction. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2018. 2, 3

[56] C. Lin, O. Wang, B.C. Russell, E. Shechtman, V.G. Kim, M. Fisher, and S. Lucey. Photometric mesh optimization for video-aligned 3D object reconstruction. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[57] Stefan Lionar, Daniil Emtsev, Dusan Svilarkovic, and Songyou Peng. Dynamic plane convolutional occupancy

networks. In *Winter Conference on Applications of Computer Vision (WACV)*, 2021. 4, 5, 8

[58] Yaron Lipman, Daniel Cohen-Or, and David Levin. Data-dependent MLS for faithful surface approximation. In *Eurographics Symposium on Geometry Processing (SGP)*, 2007. 1

[59] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020. 3

[60] S. Liu, S. Saito, W. Chen, and H. Li. Learning to infer implicit surfaces without 3D supervision. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2019. 3

[61] S. Liu, Y. Zhang, S. Peng, B. Shi, M. Pollefeys, and Z. Cui. DIST: rendering deep implicit signed distance function with differentiable sphere tracing. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[62] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[63] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *ACM SIGGRAPH Computer Graphics (CG)*, 21(4):163–169, 1987. 2, 3, 7

[64] Yiming Luo, Zhenxing Mi, and Wenbing Tao. DeepDT: Learning geometry from Delaunay triangulation for surface reconstruction. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2021. 3

[65] Priyanka Mandikal, K L Navaneet, Mayank Agarwal, and R Venkatesh Babu. 3D-LMNet: Latent embedding matching for accurate and diverse 3D point cloud reconstruction from a single image. In *British Machine Vision Conference (BMVC)*, 2018. 3

[66] Jiageng Mao, Xiaogang Wang, and Hongsheng Li. Interpolated convolutional networks for 3D point cloud understanding. In *International Conference on Computer Vision (ICCV)*, 2019. 3

[67] Julien N. P. Martel, David B. Lindell, Connor Z. Lin, Eric R. Chan, Marco Monteiro, and Gordon Wetzstein. ACORN: Adaptive coordinate networks for neural scene representation. *ACM Transactions on Graphics (TOG)*, 40(4), 2021. 3

[68] D. Maturana and S. Scherer. VoxNet: A 3D convolutional neural network for real-time object recognition. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2015. 2

[69] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. Occupancy networks: Learning 3D reconstruction in function space. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2, 3, 4, 7, 8

[70] M. Michalkiewicz, J.K. Pontes, D. Jack, M. Baktashmotlagh, and A. Eriksson. Implicit surface representations as layers in neural networks. In *International Conference on Computer Vision (ICCV)*, 2019. 2, 3

[71] Patrick Mullen, Fernando De Goes, Mathieu Desbrun, David Cohen-Steiner, and Pierre Alliez. Signing the unsigned: Robust surface reconstruction from raw pointsets. *Computer Graphics Forum (CGF)*, 29(5):1733–1741, 2010. 1

[72] Liangliang Nan and Peter Wonka. Polyfit: Polygonal surface reconstruction from point clouds. In *International Conference on Computer Vision (ICCV)*, 2017. 1

[73] Liangliang Nan, Ke Xie, Andrei Sharf, and Shenzhen Visuca. A search-classify approach for cluttered indoor scene understanding. *ACM Transactions on Graphics (TOG)*, 2012. 2

[74] Charlie Nash, Yaroslav Ganin, S. M. Ali Eslami, and Peter W. Battaglia. PolyGen: an autoregressive generative model of 3D meshes. In *International Conference on Machine Learning (ICML)*, 2020. 3

[75] M. Niemeyer, L.M. Mescheder, M. Oechsle, and A. Geiger. Differentiable volumetric rendering: Learning implicit 3D representations without 3D supervision. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[76] M. Oechsle, L. Mescheder, M. Niemeyer, T. Strauss, and A. Geiger. Texture fields: Learning texture representations in function space. In *International Conference on Computer Vision (ICCV)*, 2019. 3

[77] J.J. Park, P. Florence, J. Straub, R.A. Newcombe, and S. Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2, 3, 8

[78] D. Paschalidou, L. van Gool, and A. Geiger. Learning unsupervised hierarchical part decomposition of 3D objects from a single RGB image. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[79] Mark Pauly, Niloy J. Mitra, Johannes Wallner, Helmut Pottmann, and Leonidas J. Guibas. Discovering structural regularity in 3D geometry. *ACM Transactions on Graphics (TOG)*, 27(3), 2008. 1

[80] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *European Conference on Computer Vision (ECCV)*, 2020. 2, 4, 5, 6, 7, 8

[81] Nikolai Poliarnyi. Out-of-core surface reconstruction via global TGV minimization. In *International Conference on Computer Vision (ICCV)*, 2021. 1

[82] C.R. Qi, H. Su, K. Mo, and L.J. Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 4, 7

[83] C.R. Qi, L. Yi, H. Su, and L.J. Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2017. 5, 7

[84] G. Riegler, A.O. Ulusoy, H. Bischof, and A. Geiger. OctNetFusion: Learning depth fusion from data. In *International Conference on 3D Vision (3DV)*, 2017. 2

[85] G. Riegler, A.O. Ulusoy, and A. Geiger. OctNet: Learning deep 3D representations at high resolutions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[86] S. Saito, Z. Huang, R. Natsume, S. Morishima, A. Kanazawa, and H. Li. PIFu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *International Conference on Computer Vision (ICCV)*, 2019. 3

[87] Ruwen Schnabel, Patrick Degener, and Reinhard Klein. Completion and reconstruction with primitive shapes. *Computer Graphics Forum (CGF)*, 28(2):503–512, 2009. 1

[88] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1

[89] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 1

[90] Divya Shanmugam, Davis Blalock, Guha Balakrishnan, and John Guttag. Better aggregation in test-time augmentation. In *International Conference on Computer Vision (ICCV)*, 2021. 6

[91] Yawar Siddiqui, Justus Thies, Fangchang Ma, Qi Shan, Matthias Nießner, and Angela Dai. RetrievalFuse: Neural 3D scene reconstruction with a database. In *International Conference on Computer Vision (ICCV)*, 2021. 5, 8

[92] Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020. 3

[93] V. Sitzmann, M. Zollhöfer, and G. Wetzstein. Scene representation networks: Continuous 3D-structure-aware neural scene representations. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2019. 3

[94] Raphael Sulzer, Loïc Landrieu, Renaud Marlet, and Bruno Vallet. Scalable surface reconstruction with Delaunay-graph neural networks. *Computer Graphics Forum (CGF)*, 40(5):157–167, 2021. 1

[95] 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 *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 3

[96] 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. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020. 3

[97] Jiapeng Tang, Jiabao Lei, Dan Xu, Feiying Ma, Kui Jia, and Lei Zhang. SA-ConvONet: Sign-agnostic optimization of convolutional occupancy networks. In *International Conference on Computer Vision (ICCV)*, 2021. 4, 5, 7

[98] M. Tatarchenko, A. Dosovitskiy, and T. Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3D outputs. In *International Conference on Computer Vision (ICCV)*, 2017. 2

[99] M. Tatarchenko, S.R. Richter, R. Ranftl, Z. Li, V. Koltun, and T. Brox. What do single-view 3D reconstruction networks learn? In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 7

[100] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J. Guibas. KPConv: Flexible and deformable convolution for point clouds. In *International Conference on Computer Vision (ICCV)*, 2019. 3

[101] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Carsten Stoll, and Christian Theobalt. PatchNets: Patch-based generalizable deep implicit 3D shape representations. In *European Conference on Computer Vision (ECCV)*, 2020. 3

[102] Benjamin Ummenhofer and Vladlen Koltun. Adaptive surface reconstruction with multiscale convolutional kernels. In *International Conference on Computer Vision (ICCV)*, 2021. 2, 4, 5

[103] Hoang Hiep Vu, Patrick Labatut, Jean Philippe Pons, and Renaud Keriven. High accuracy and visibility-consistent dense multiview stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 34(5), 2012. 1

[104] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.G. Jiang. Pixel2Mesh: Generating 3D mesh models from single RGB images. In *European Conference on Computer Vision (ECCV)*, 2018. 3

[105] Shenlong Wang, Simon Suo, Wei-Chiu Ma, Andrei Pokrovsky, and Raquel Urtasun. Deep parametric continuous convolutional neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[106] Yue Wang and Justin M. Solomon. PRNet: Self-supervised learning for partial-to-partial registration. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2019. 5

[107] C. Wen, Y. Zhang, Z. Li, and Y. Fu. Pixel2Mesh++: Multi-view 3D mesh generation via deformation. In *International Conference on Computer Vision (ICCV)*, 2019. 3

[108] Xin Wen, Tianyang Li, Z. Han, and Yu-Shen Liu. Point cloud completion by skip-attention network with hierarchical folding. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[109] Francis Williams, Teseo Schneider, Claudio Silva, Denis Zorin, Joan Bruna, and Daniele Panozzo. Deep geometric prior for surface reconstruction. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[110] Francis Williams, Matthew Trager, Joan Bruna, and Denis Zorin. Neural splines: Fitting 3D surfaces with infinitely-wide neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1, 8

[111] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum. Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2016. 2

[112] Wenxuan Wu, Zhongang Qi, and Li Fuxin. PointConv: Deep convolutional networks on 3D point clouds. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[113] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3D shapenets: A deep representation for volumetric shapes. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2

[114] Haozhe Xie, Hongxun Yao, Xiaoshuai Sun, Shangchen Zhou, and Shengping Zhang. Pix2Vox: Context-aware 3D reconstruction from single and multi-view images. In *International Conference on Computer Vision (ICCV)*, 2019. 2

[115] Haozhe Xie, Hongxun Yao, Shengping Zhang, Shangchen Zhou, and Wenxiu Sun. Pix2Vox++: Multi-scale context-aware 3D object reconstruction from single and multiple images. *International Journal on Computer Vision (IJCV)*, 128, 2020. 2

[116] Q. Xu, W. Wang, D. Ceylan, R. Mech, and U. Neumann. DISN: deep implicit surface network for high-quality single-view 3D reconstruction. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2019. 3

[117] Yifan Xu, Tianqi Fan, Mingye Xu, Long Zeng, and Yu Qiao. SpiderCNN: Deep learning on point sets with parameterized convolutional filters. In *European Conference on Computer Vision (ECCV)*, 2018. 3

[118] Zike Yan, Yuxin Tian, Xuesong Shi, Ping Guo, Peng Wang, and Hongbin Zha. Continual neural mapping: Learning an implicit scene representation from sequential observations. In *International Conference on Computer Vision (ICCV)*, 2021. 3

[119] G. Yang, X. Huang, Z. Hao, M. Liu, S.J. Belongie, and B. Hariharan. PointFlow: 3D point cloud generation with continuous normalizing flows. In *International Conference on Computer Vision (ICCV)*, 2019. 2, 3

[120] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. PCN: Point completion network. In *International Conference on 3D Vision (3DV)*, 2018. 3

[121] Wenbin Zhao, Jiabao Lei, Yuxin Wen, Jianguo Zhang, and Kui Jia. Sign-agnostic implicit learning of surface self-similarities for shape modeling and reconstruction from raw point clouds. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 3

[122] Qingnan Zhou and Alec Jacobson. Thingi10k: A dataset of 10,000 3D-printing models. *arXiv preprint arXiv:1605.04797*, 2016. 7

[123] Yang Zhou, Shuhan Shen, and Zhanyi Hu. Detail preserved surface reconstruction from point cloud. *Sensors*, 19(6), 2019. 1