Sparse Convolutional Networks for Surface Reconstruction from Noisy Point Clouds

Tao Wang, Jing Wu, Ze Ji, and Yu-Kun Lai
Cardiff University
{wangt31,wuj11,jiz1,laiy4}@cardiff.ac.uk

Abstract

Reconstructing accurate 3D surfaces from noisy point clouds is a fundamental problem in computer vision. Among different approaches, neural implicit methods that map 3D coordinates to occupancy values benefit from the learning capabilities of deep neural networks and the flexible topology of implicit representations, achieving promising reconstruction results. However, existing methods utilize standard (dense) 3D convolutional neural networks for feature extraction and occupancy prediction, which significantly restricts their capability to reconstruct details. In this paper, we propose a neural implicit method based on sparse convolutions, where features and network calculations only focus on grid points close to the surface to be reconstructed. This allows us to build significantly higher resolution 3D grids and reconstruct high-fidelity details. We further build a 3D residual UNet to extract features which are robust to noise, while ensuring details are retained. A 3D position along with features extracted at the position are fed into the occupancy probability predictor network to obtain occupancy. As features at nearby grid points to the query position may not exist due to the sparse nature, we propose a normalized weight interpolation approach to obtain smooth interpolation with sparse data. Experimental results demonstrate that our method achieves promising results, both qualitatively and quantitatively, outperforming existing methods.

1. Introduction

With the increasing availability of 3D sensors, such as 3D scanners and LiDAR sensors (including those fitted on mobile phones), acquisition of 3D data becomes much easier. The raw data captured is often in the form of point clouds, with unavoidable sensor noise. To support downstream applications, it is essential to obtain accurate 3D surface reconstruction. However, this is a challenging task as the input may contain significant noise and can have flexible topology. To address this, classic methods perform 3D reconstruction using some implicit representations, such as radial basis functions [2] and Poisson reconstruction [20, 21]. Nevertheless, traditional implicit representations tend to lose fine geometric details, and have limited capability to handle noisy inputs. Deep neural networks have recently been utilized to predict implicit fields that depict 3D shapes [46, 48]. They typically use a neural network to map 3D coordinates of arbitrary locations in space to certain values, such as occupancy [31, 34] and signed (or unsigned) distance to the nearest surface point of an object [33, 38]. Thanks to their learning capabilities, such methods can achieve better reconstruction results, both in terms of quality of surface such as smoothness and details, and robustness to noise.

To learn an implicit field, often a dense grid of 3D features is extracted using convolutional layers. However, convolution in a 3D voxel grid brings in a huge computational overhead compared to 2D convolution. Meanwhile, it is wasteful as a large amount of empty space exists in the 3D voxel grid. Sparse tensors only store locations where the feature is non-zero which saves tremendous space without information loss. Sparse convolution [9, 15, 41] is a generalized convolution that directly conducts on sparse tensors, which is a feasible way to tackle intensive usage of computational resources and sparsity in high-dimension input data. Similar to how image resolution determines image quality in 2D, a higher resolution voxel grid can represent 3D objects better and provide abundant details. Sparse convolution mitigates the limit of memory cost, therefore, convolution can be conducted on a higher resolution voxel grid and output the latent feature in demand.

To determine the reconstructed 3D shapes, occupancy values are queried at arbitrary locations in the implicit field. To achieve this, features at the 3D grid are interpolated as in [14, 34], e.g. using tri-linear interpolation of features at the 8 grid cell corners for the grid cell that encloses the queried point. However, in our method because of the sparsity of sparse tensor, the queried point may be located in a grid cell where only part of grid cell corners contain fea-

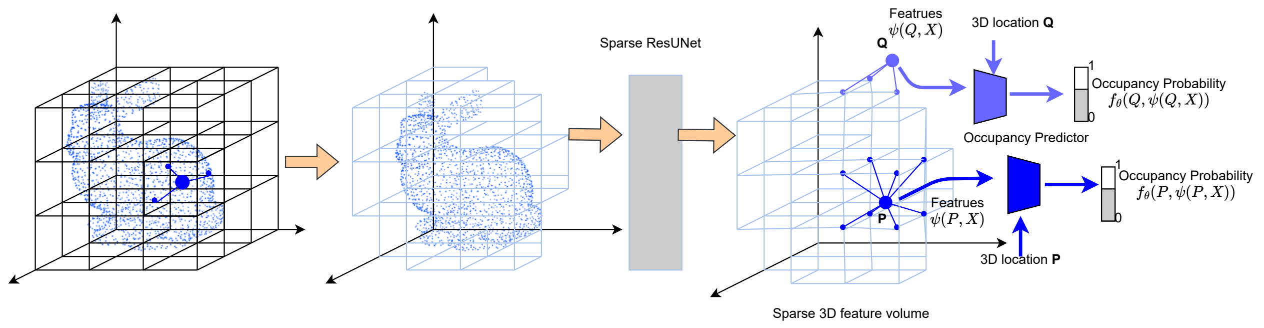

Figure 1. Pipeline of our method. Using higher resolution sparse grids provides high fidelity sampling space for randomly sampled points.

tures, or if the queried point is further away from the surface, it may happen that none of the grid cell corners of the enclosing grid cell has features. We thus propose a novel normalized linear interpolation for the sparse voxel grid, which only relies on sparse features while ensuring a smooth interpolation of them.

In summary, our contributions are:

- To the best of our knowledge, it is the first work that exploits sparse convolutions for surface reconstruction from noisy point clouds.
- To achieve this, neighboring voxels not originally occupied may be required to contain features for accurate prediction, so a generative sparse de-convolution is utilized to add appropriate voxels to the sparse grid for feature sampling. We also propose a novel normalized interpolation applicable to points where the enclosing grid cell does not have features at all grid corners.
- Experimental results show that our network architecture with higher-resolution, sparse voxel grids provides more accurate implicit field results, outperforming other methods for reconstruction at both scales: single objects and indoor scenes.

2. Related Work

3D reconstruction is a problem of widespread concern. Generally speaking, it involves the conversion from a partial observation of a 3D model (images or irregular 3D representation like point clouds) to a comprehensive and complete 3D structure. In this paper, we specifically address the problem of reconstructing smooth mesh surfaces from noisy point clouds.

2.1. Traditional Surface Reconstruction Methods

Poisson reconstruction [20, 21] and RBF (radial basis function) method [2] were representative methods for implicit surface reconstruction. However, RBF and Poisson methods both need the orientation of each point in the point cloud, which restricts the applicability of the method. Meanwhile, the computational overhead to solve the equation in the RBF method is substantial, especially when the input point cloud has a large number of points. Additionally, Moving Least Squares (MLS) method [1] reconstructs the surface by finding a suitable local area to constitute a polynomial. But it has limited capability of reconstructing details especially when the input point cloud contains much noise.

2.2. Learning 3D Shapes by Neural Networks

Deep neural networks work by learning from a large amount of data and have shown their capability in many 3D computer vision tasks. Besides, they can be regarded as a universal approximator to any function. For surface reconstruction, 3D shapes can be represented using either explicit or implicit representations. Explicit representation of 3D models directly represents the geometry such as coordinates, allowing its appearance to be directly rendered. However, they can either be expensive to represent details, or not sufficiently flexible with topological changes. In contrast, the implicit function is easier to be approximated by a deep neural network, although the output geometry cannot be directly rendered, relying on algorithms such as Marching Cubes for isosurface extraction to obtain an explicit representation for rendering.

Explicit representations. Point clouds, voxels and meshes are considered to be the most popular explicit representations of 3D models. A point cloud consists of a certain number of points whose coordinates can be output by a neural network. PointNet [35] based auto-encoders [28, 45, 49] have been applied to the completion task, and the output is often reshaped to a point cloud that has a fixed number of points. Every three channels in output are viewed as x, y, z coordinates to form a single point. Similarly, Fan et al. [13] use a point cloud as the output of the reconstruction task from a single image. However, limited by the number of points in the point cloud, the reconstruction results are relatively coarse with missing details. Point clouds also lack a regular structure, making learning more difficult. AtlasNet [17] shapes points to a patch by directly connecting 3D points that are originally connected in the 2D plane. Voxel-based 3D reconstruction has been attempted by [10, 26], but such methods have very high memory requirements. To solve the computation overhead in terms of the 3D voxel grid, several methods have been proposed using spatial partitioning such as octree-based methods [43] and sparse voxel grids [9, 12, 18]. Meshes have a complex topology, for the reconstruction task, the neural network is trying to convolve features at vertices and faces in [19]. However, mesh-based methods cannot easily model topological changes.

Implicit representations. Indicator/occupancy functions and signed distance functions are two classic types of implicit methods to represent 3D shapes. Related computer vision tasks like 3D reconstruction from a single image have been implemented in [31]. For the indicator/occupancy function, if the sampled point is located inside an object, the occupancy value will be 1. Otherwise, occupancy value of 0 indicates that the sampled point is located outside the object. The decision boundary defines surfaces of the object. This idea has also been explored by IM-NET [5], which generates shapes with this paradigm. The idea to approximate implicit fields by a neural network is also applied in signed distance functions (SDFs) [8, 23, 30, 33, 38–40] while the signed distance from a sampled point to its closest surface is recorded. The zero level set of signed distances depicts the surface. The implicit function maps any location in space to a real number, which is naturally differentiable. However, if the learning-based method overfocuses on global information of input, the reconstruction results tend to be close to model retrieval from the training set [44]. To solve this, Takikawa et al. [40] use Level of Detail (LOD) to better extract multi-scale latent features. Local information is also important, some methods [25,34] use convolution layers to better extract latent features fetched to the decoder. Both methods are based on interpolation over a certain plane or voxel grid, and the difference between them is how sampling planes are arranged. For [34], indoor scene reconstruction relies on voxel grid sampling, but the low resolution voxel grid lowers the representation capability of the 3D shape. Tang et al. [42] also use an occupancy network, but they propose to use test time augmentation that improves the capability of their model for specific inputs. However, the extra iteration time in the inference stage lowers the capability and flexibility of the model. Similarly, the methods [23, 33, 40] leverage the inference time optimization to acquire details of the geometry from coarse or partial input. When the model encounters unseen input without corresponding SDF values, the generated mesh may not be ideal.

Hybrid representations. It is popular and efficient to combine the implicit and explicit representations to acquire better results. Deep marching cubes algorithm [24] is proposed to make the marching cubes differentiable, allowing end-to-end training for mesh reconstruction from point cloud input. Neural marching cubes method [6] instead considers a differential solution for mesh reconstruction from general implicit fields. Octree has been used as a "scaffolding" in [29] and an implicit Moving Least Squares (MLS) representation is extracted from each octant within the octree. Similarly, Shen et al. [37] propose a deformable tetrahedral grid from which the surfaces are extracted. For the generation task, the methods [3,27,36] also build an implicit field on an explicit grid. Our method can also be regarded as a hybrid method, where the sparse voxel grid provides an adaptable sampling space that facilitates the input of the implicit function.

3. Approach

Our method uses a neural network to approximate an implicit function outputting occupancy values. This implicit function takes a randomly sampled point's coordinates and its corresponding features as input. The features of the point are interpolated in the latent space encoded from an input noisy point cloud. The ground-truth occupancy values of points to supervise the learning are obtained from the ground-truth meshes as in [31]. The whole pipeline as shown in Figure 1 is formed by an encoder-decoder component and an occupancy predictor. The details are described in the following subsections.

3.1. Sparse Convolution based Encoder-Decoder

Encoder-decoder is a widely used structure to deal with computer vision tasks. Our model also takes this fashion to convert the noisy point cloud into latent code by an encoder. The encoder is inspired by the PointNet [35] which can directly process the point cloud. It extracts the feature of each point in the noisy point cloud. Then, the point cloud with its per-point feature is voxelized to a voxel grid to facilitate convolution. It is arguable that the higher resolution of the voxel grid will bring higher fidelity towards the original input. However, the curse of dimensionality limits the upper bound of resolution in many related works using voxelization [10, 34]. As surfaces are being modeled, there is a reasonable sparsity of the input noisy point cloud, thus plenty of voxels among the dense voxel grid are redundant as no features are assigned to them. Therefore, we use sparse convolution to solve this problem. Sparse convolution conducts convolution on sparse tensors where redundant voxels are skipped. A denser but sparse voxel grid can be acquired, which also has the capability of recovering more details of the 3D model. Meanwhile, the storage of tensors will also be reduced since locations with zero features will be ignored. The parameter for the voxelization is set by $s \times s \times s$ ($s = 80$ in our experiments, but could be higher). Subsequently, a sparse 3D-UNet module with

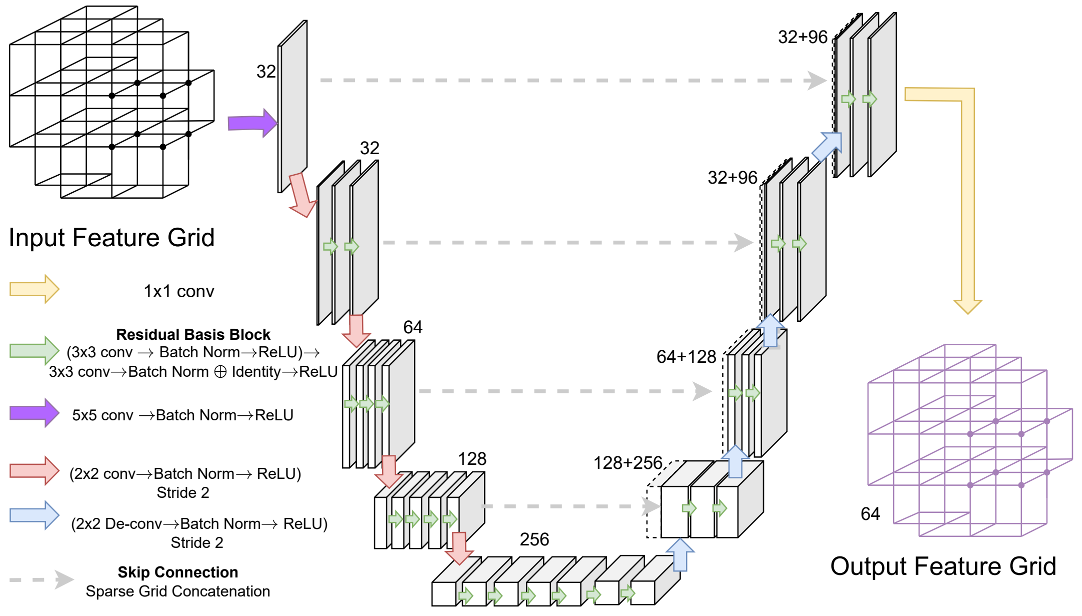
Input Feature Grid
1x1 conv
Residual Basis Block
(3x3 conv → Batch Norm→ReLU)→
3x3 conv→Batch Norm ⊕ Identity→ReLU
5x5 conv →Batch Norm→ReLU
(2x2 conv→Batch Norm→ ReLU)
Stride 2
(2x2 De-conv→Batch Norm→ ReLU)
Stride 2
Skip Connection
Sparse Grid Concatenation
32
32
64
128
256
128+256
64+128
32+96
32+96
64
Output Feature Grid

Figure 2. Our sparse Res-UNet network architecture where convolutions are conducted on the sparse grid.

residual blocks as shown in Figure 2 is attached to extract local to global information from this voxelized point cloud. This module will output a sparse voxel grid with feature latent codes at (selected) grid points, which is crucial to the reconstruction task.

3.2. Occupancy Predictor

Given an arbitrary sample point $\mathbf{p}$ and its corresponding feature obtained from the output of the encoder-decoder by interpolation $\psi(\mathbf{p}, \mathbf{x})$, an occupancy predictor that is essentially a fully connected network $f_\theta(\mathbf{p}, \psi(\mathbf{p}, \mathbf{x}))$ predicts the occupancy value of $\mathbf{p}$. After training with randomly sampled points and their occupancy values, our network approximates an implicit function outputting occupancy of any point in space. For a 3D shape, occupancy values $0, 1$ mean outside and inside of the 3D object respectively, and the decision boundary ($occupany = \tau$) will be used to describe the surface of 3D models (through Marching Cubes based isosurface extraction).

While existing methods tend to use simple trilinear interpolation, the use of sparse convolutions means that for a given point, its enclosing unit grid cell may have some or even all cell corners without features. We consider the following two cases:

1) If none of 8 cell corners have learned features in the sparse grid, the sampled point is likely to be further away from the surface to be reconstructed. We use a Gaussian kernel weighting to combine features of all grid points, where the contributing weight for a grid point with center coordinates $\mathbf{x}_i$ is defined as

$$w_i = \exp_{\mathbf{x}_i \in \mathbf{x}} \frac{-\|\mathbf{p} - \mathbf{x}_i\|_2^2}{2\sigma^2} \tag{1}$$

where $\mathbf{x}$ is the set of all the grid cells with features, and $\mathbf{p}$ indicates a sampled point, and σ is the Gaussian kernel size. The interpolated feature at $\mathbf{p}$ is calculated as

$$\gamma(\mathbf{p}, \mathbf{x}) = \frac{\sum_{i=1}^{n} V_{\mathbf{x_i}} w_i}{\sum_{i=1}^{n} w_i}. \tag{2}$$

where $V_{\mathbf{x_i}}$ indicates the feature of grid cell $\mathbf{x_i}$. Note that although in principle all grid points with features are involved, the local support nature of Gaussian kernels means that only relatively nearby grid points have reasonable contributions.

2) If at least one of the 8 cell corners of the unit grid cell that encloses $\mathbf{p}$ has features, then the sampled point is close to the reconstructed surface. Trilinear interpolation can give smooth interpolation results. However, standard trilinear interpolation can involve grid points without calculated features, and using a zero feature vector for these would lead to unsmooth features, resulting in bumpy surfaces. We thus propose a normalized trilinear interpolation scheme, which is conceptually equivalent to propagating features from existing cell corners to missing cell corners, but without the need to modify sparse tensors. Such interpolation operation is denoted as $\lambda(\mathbf{p}, \mathbf{x})$ and the details will be given in the next subsection.

Let $c_\mathbf{p}$ be the enclosing unit grid cell (including its 8 corners) for an arbitrary position $\mathbf{p}$, our feature interpolation is

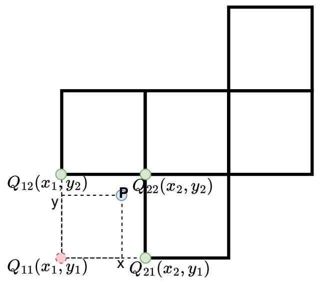

Figure 3. Normalized sparse grid interpolation, using a 2D sparse grid for illustration. Point P is located outside the sparse grid, but its enclosing grid cell contains corners with features, and its corresponding feature is obtained via normalized interpolation with Q_{11}'s weight excluded.

formally defined as

$$\psi(\mathbf{p}, \mathbf{x}) = \begin{cases} \lambda(\mathbf{p}, \mathbf{x}) & \text{If} \quad c_{\mathbf{p}} \cap \mathbf{x} = \emptyset \\ \gamma(\mathbf{p}, \mathbf{x}) & \text{otherwise} \end{cases} \tag{3}$$

3.3. Normalized Trilinear Interpolation

As mentioned above, we modify the traditional trilinear interpolation to provide smooth interpolation for sparse grids. The outcome of the trilinear interpolation is normalized by the sum of all valid weights. However, as shown in Figure 3, a sampled point's enclosing grid cell may not be fully inside the voxel grid due to the sparsity of our sparse voxel grid.

In 3D, the grid cell of a randomly sampled point has 8 vertices. Let V_{ijk} denote the feature value of the vertex point with index ijk, and x, y, z the sampled point's relative coordinates with respect to its grid cell. The output of the trilinear interpolation is calculated as:

$$\begin{aligned} P_{xyz} &= V_{000} * (1-x)(1-y)(1-z) \\ &+ V_{100} * x(1-y)(1-z) + V_{010} * (1-x)y(1-z) \\ &+ V_{001} * (1-x)(1-y)z + V_{101} * x(1-y)z \\ &+ V_{011} * (1-x)yz + V_{110} * xy(1-z) + V_{111} * xyz \end{aligned} \tag{4}$$

However, only those vertices inside the sparse voxel grid have non-zero features and should be considered in the calculation of the weight response to the sampled point. In this situation, the interpolated outcome will be normalized by the sum of these valid weights (for a dense voxel grid, this sum is guaranteed to be one). This normalization enables the interpolation of sampled points outside of the sparse voxel grid, and thus ensures coherent point latent codes of these points. It is helpful to get high-fidelity results. For a unit grid cell, if a feature value V_{ijk} is zero, its corresponding indicator function $\mathbb{1}$ will be zero and the weight will be ignored. Therefore, the sum of valid weights is calculated as in Eq. (5). The normalized interpolation outcome $P_{xyz\prime}$ is calculated as in Eq. (6).

$$\begin{aligned} W_{valid} &= \mathbb{1}(1-x)(1-y)(1-z) + \mathbb{1}x(1-y)(1-z) + \\ &\mathbb{1}(1-x)y(1-z) + \mathbb{1}(1-x)(1-y)z + \mathbb{1}x(1-y)z + \\ &\mathbb{1}(1-x)yz + \mathbb{1}xy(1-z) + \mathbb{1}xyz \end{aligned} \tag{5}$$

$$P_{xyz\prime} = \frac{P_{xyz}}{W_{valid}} \tag{6}$$

Furthermore, spectral normalization [32] is effective to make a fully connected network achieve Lipschtz continuity, which is beneficial to get a smooth surface. Lipschitz-continuity is a uniform continuity for functions. For any function, Lipschtz continuity guarantees it will not change too fast as input changes. Similarly for our neural network, its output occupancy value is not supposed to change too fast for two adjacent sample points. As later shown in the ablation study, its application brings considerable effect in terms of surface smoothness.

4. Implementation and Experiments

The key idea of our method is using a sparse voxel grid to extend the resolution of sampling space to have a better isosurface and acquire abundant detail. We take advantage of the sparse tensor auto-differentiation library [9] to implement our pipeline. Additionally, the multiresolution isosurface extraction algorithm proposed in [31] has been used to extract the explicit surface from the implicit field. Experiments are conducted on two scales: single-object reconstruction and scene-level reconstruction.

4.1. Datasets

A subset of ShapeNet dataset [4] has been used for single object reconstruction. As in [31, 34], 13 categories of objects have been chosen. In total, there are 50,000 objects split into training/validation/test lists in $7:2:1$ ratio. For scene-level reconstruction, we focus on indoor scene reconstruction, the synthesized dataset proposed by [34] has been used for training. To evaluate our method's capability on the unseen indoor scenes, another real-world scanned dataset ScanNet [11] is used.

4.2. Loss Function

The occupancy predictor accomplishes the mapping between any location's coordinates to an occupancy value. The interval of occupancy value is between 0 and 1. It is reasonable to use the binary cross-entropy (BCE) loss as our loss function (7).

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)] \tag{7}$$

where $\hat{y}_i$ indicates the sample point's occupancy value output from the decoder, y_i indicates the target occupancy value processed from ground truth mesh. N is the total number of sampled points. In our case, there are 2048 sampled points for a single object and indoor scene reconstruction.

4.3. Network Training and Inference

To optimize our model, we choose the Adam optimizer [22] with the initial learning rate $1e{-}4$. A scheduler has also been implemented to reduce the learning rate over training iterations. This scheduler reads the average IoU of the validation set in our implementation, and when this metric is not numerically improved, the learning rate will be decreased by a factor of 0.1. The input noisy point cloud is sampled from the ground truth mesh surface. In our experiments, there are $10,000$ points in the point cloud with 0.005 Gaussian noise as input to the encoder both for the single object and indoor scene reconstruction. The training batch size is 4 and the Gaussian weighted sampling $\sigma = 0.25$. In the stage of inference, the threshold τ is set to 0.5 to decide the level set for isosurface. The whole training is conducted on a single Nvidia RTX3090 GPU.

5. Results and Evaluation

5.1. Metrics

Our pipeline achieves superior results both qualitatively and quantitatively. We comprehensively compared our method with others in four metrics, which are intersection over union (IoU), L_1 Chamfer distance (CD), normal consistency (NC) and F-score. IoU is a classic method to compare the similarity of two meshes. It is the ratio between the area of intersection and the area of the union of two shapes. Chamfer distance is a metric widely used in the evaluation of point clouds. The output and ground truth meshes have been processed by point sampling 100k points on their surfaces. Chamfer distance does not require the numbers of points in the output S_1 and ground truth S_2 to be the same. It measures the distance between each point in one set to its nearest neighbor in the other set. Normal consistency is calculated between the estimated normal of each triangle polygon and the ground truth normal. Another metric we use is F-score which is less sensitive to the outlier. F-score is set with threshold $\eta = 0.01$ in our experiment. The precision is measured by the percentage of distances between the reconstruction result and ground truth that are less than the threshold; recall vice versa.

5.2. Results

Single object reconstruction ShapeNet dataset has been tested in our experiment, and 13 different categories of objects have been chosen (aeroplane, bench, cabinet, car, chair, display, lamp, loudspeaker, rifle, sofa, table, telephone, vessel). From Table 1, except Chamfer distance, it is clearly indicated that our method outperforms other methods in other three metrics by considerable margins. Additionally, qualitative results of single object reconstruction are shown in Figure 4, which also demonstrate the superiority of our results. It should be noted that in this evaluation, unlike [23, 33, 40], our model only takes noisy point cloud as input, there is no need for latent code optimization with corresponding occupancy values.

Method	CD ↓	IoU ↑	NC ↑	F-score ↑
PointConv [47]	0.126	0.689	0.858	0.644
DCC-DIF [23]	0.233	0.171	0.613	0.563
IF-Nets [7]	0.084	0.196	0.823	-
ONet [31]	0.087	0.761	0.891	0.785
Conv-ONet (32^3) [34]	0.057	0.863	0.912	0.900
Conv-ONet (3×64^2) [34]	**0.040**	0.897	0.941	0.952
DP-CovONet [25]	0.042	0.896	0.941	0.951
Ours (80^3)	0.046	**0.922**	**0.950**	**0.953**

Table 1. Single objects reconstruction result, quantitative comparison between our method and others.

Method	CD ↓	IoU ↑	NC ↑	F-score ↑
ONet [31]	0.203	0.475	0.783	0.541
PointConv [47]	0.165	0.523	0.811	0.790
SPSR [21]	0.223	-	0.866	0.810
SPSR (trimmed) [21]	0.069	-	0.890	0.892
ConvONet (64^3) [34]	0.042	0.849	<u>0.915</u>	0.964
DP-ConvONet [25]	0.042	0.800	0.912	0.960
Ours (64^3)	<u>0.039</u>	<u>0.861</u>	<u>0.915</u>	<u>0.973</u>
Ours (80^3)	**0.038**	**0.874**	**0.916**	**0.976**

Table 2. Quantitative comparison on the synthesized indoor scene between our method and others.

Indoor scene reconstruction The evaluation is conducted on the dataset created by [34] and the real-world scanned dataset ScanNet V2 [11]. Our method prevails in all metrics when reconstructing complex indoor scenes, which is shown in Table 2. Additionally, qualitative comparisons are displayed in Figure 5, where the surface of the table can be successfully reconstructed using our method where other methods fail. To validate the generalization ability of our model, the evaluation has also been conducted on ScanNet V2 dataset [11]. Even the model is trained on a synthesized dataset, it can still handle complex scenes in the real-world scanned dataset. From the perspective of quantitative analysis, Table 3 shows that our model achieves state-of-the-art performance on the real world scanned dataset.

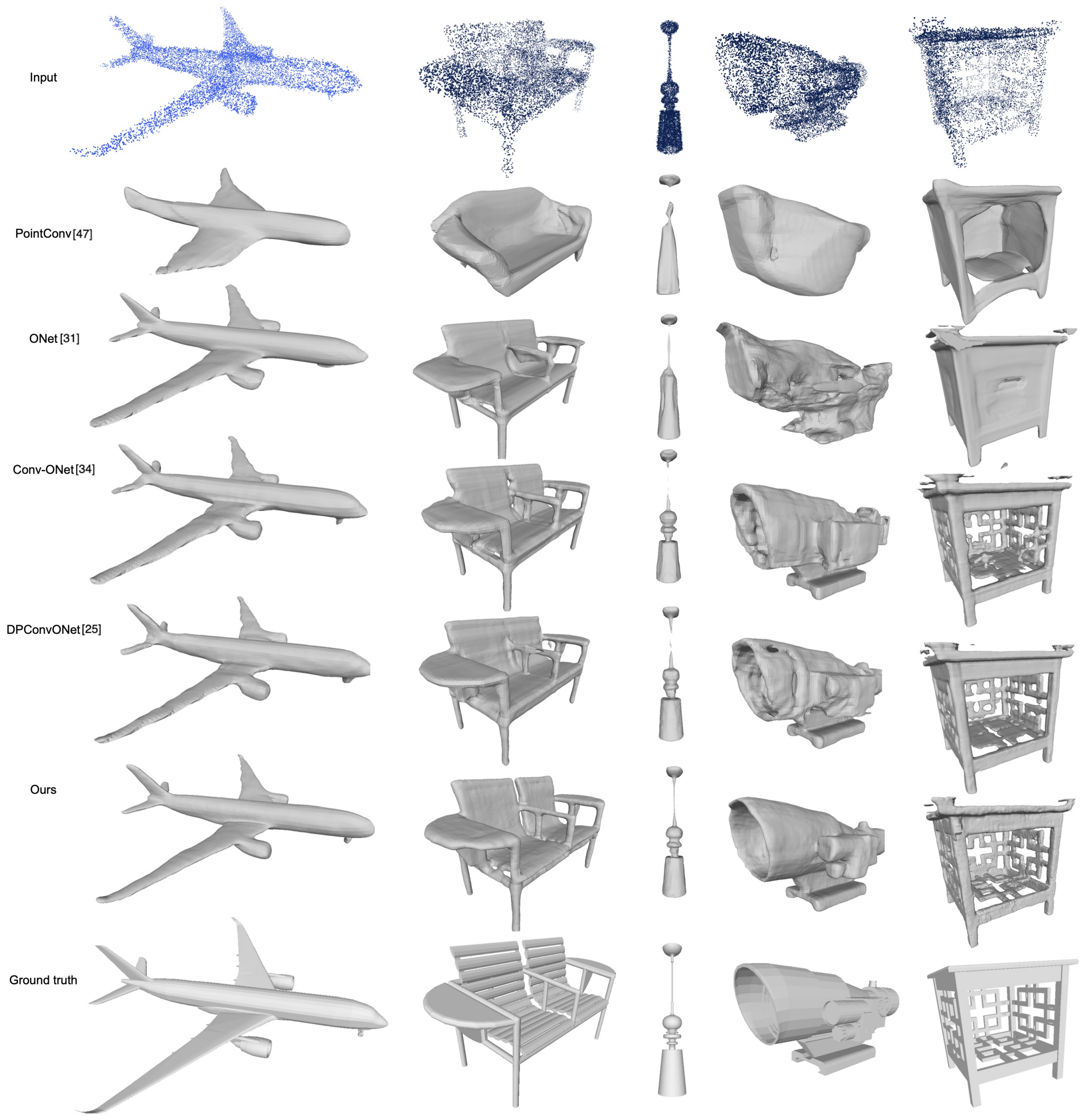

Figure 4. Qualitative comparison between different methods for 3D object reconstruction on ShapeNet.

Method	CD ↓	F-score ↑
ONet [31]	0.389	0.390
PointConv [47]	0.316	0.439
SPSR [21]	0.293	0.731
SPSR (trimmed) [21]	0.086	0.847
Conv-ONet (64^3) [34]	0.077	0.886
DP-ConvONet [25]	0.079	0.876
Ours	**0.052**	**0.890**

Table 3. Quantitative comparison on real-world scanned dataset ScanNet V2 [11].

Handling different levels of noise To demonstrate the model's generalizability w.r.t. noise, we test input with different levels of noise and results are shown in Figure 6. Our method is still able to deliver better results with smoother and more detailed surfaces. Quantitatively, our method achieves 0.2% and 1.6% better IoU than Conv-ONet [34] under 0.002 and 0.007 Gaussian noise respectively.

Memory and efficiency analysis Table 4 indicates that our method costs the least GPU memory even with a higher voxel grid resolution. The saved GPU memory could be used for a deeper network or more complex network structure like attention modules. Our method is much more efficient than test time optimization-based methods

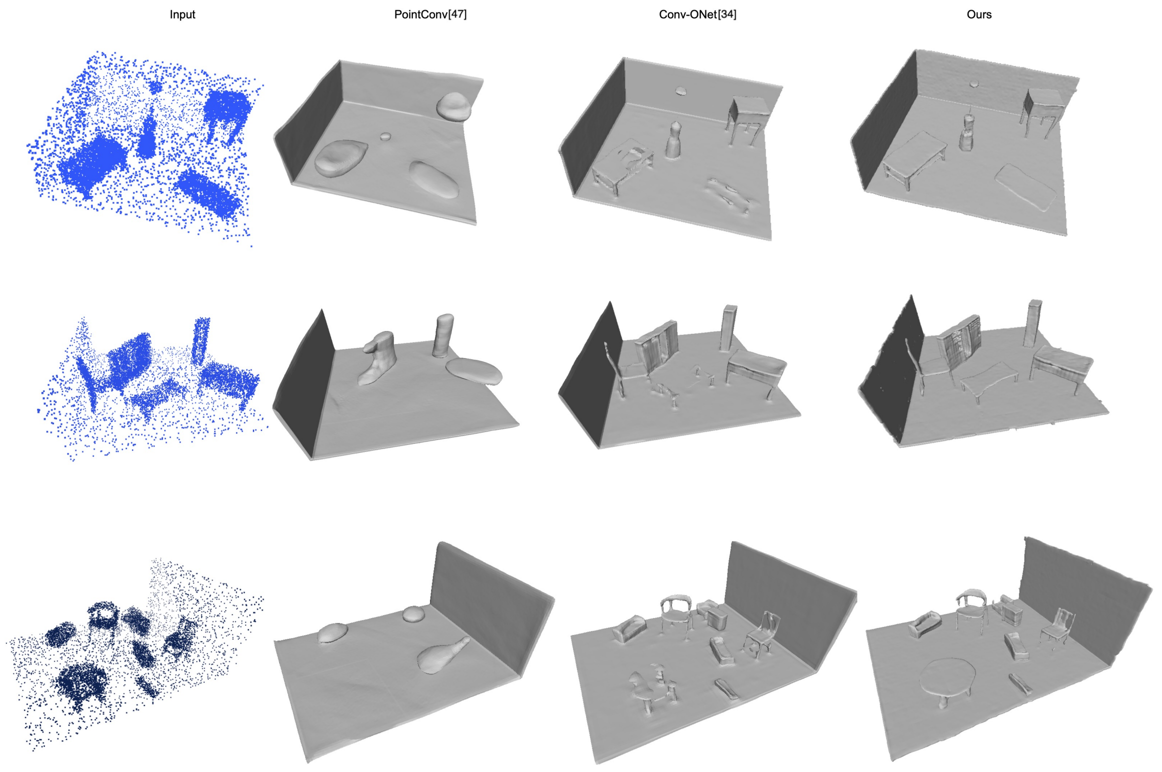

Figure 5. Synthetic room reconstruction comparing our method with existing methods.

Method	GPU Memory	Inference Time
IGR [16]	5.03G	45.19s
IF-Nets [7]	5.76G	2.12s
Conv-ONet (64^3) [34]	9.71G	0.90s
Ours (64^3)	3.39G	5.09s
Ours (80^3)	3.60G	5.66s

Table 4. Sparse convolution significantly decreases the GPU memory overhead on 3D grid convolution, even with a higher resolution.

like IGR [16]. Although Conv-ONet [34] takes less inference time, it costs significantly more GPU memory.

6. Conclusion

We propose a novel method to tackle 3D reconstruction which is usually restricted by the resolution of sampling grids. By only storing features at meaningful grid points with sparse convolutions, our method is able to extract better features at different scales for shape reconstruction. We further develop a normalized interpolation scheme along with Gaussian weighted approach for feature interpolation for sampled points near to and far away from the reconstructed surface. The qualitative and quantitative results demonstrate our method achieves state-of-the-art performance. Further work will focus on the generative pattern of the sparse grid and outdoor street-level reconstruction.

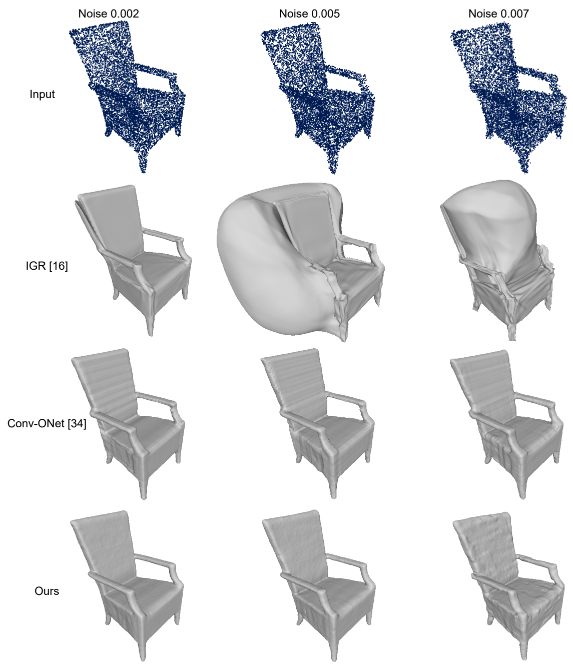

Figure 6. Experiments show that under different levels of Gaussian noise, our method still could reconstruct promising surfaces.

References

[1] Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T Silva. Point set surfaces. In *Proceedings Visualization, 2001. VIS'01.*, pages 21–29. IEEE, 2001. 2

[2] Jonathan C Carr, Richard K Beatson, Jon B Cherrie, Tim J Mitchell, W Richard Fright, Bruce C McCallum, and Tim R Evans. Reconstruction and representation of 3D objects with radial basis functions. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 67–76, 2001. 1, 2

[3] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *arXiv*, 2021. 3

[4] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An information-rich 3D model repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 5

[5] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 3

[6] Zhiqin Chen and Hao Zhang. Neural marching cubes. *ACM Transactions on Graphics (Special Issue of SIGGRAPH Asia)*, 40(6), 2021. 3

[7] Julian Chibane, Thiemo Alldieck, and Gerard Pons-Moll. Implicit functions in feature space for 3d shape reconstruction and completion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6970–6981, 2020. 6, 8

[8] Gene Chou, Ilya Chugunov, and Felix Heide. GenSDF: Two-stage learning of generalizable signed distance functions. In *Proc. of Neural Information Processing Systems (NeurIPS)*, 2022. 3

[9] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4D spatio-temporal ConvNets: Minkowski convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019. 1, 3, 5

[10] Christopher B. Choy, Danfei Xu, Jun Young Gwak, Kevin Chen, and Silvio Savarese. 3D-R2N2: A unified approach for single and multi-view 3D object reconstruction. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9912 LNCS, pages 628–644. Springer Verlag, apr 2016. 3

[11] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017. 5, 6, 7

[12] Angela Dai, Christian Diller, and Matthias Nießner. SG-NN: Sparse generative neural networks for self-supervised scene completion of RGB-D scans. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2020. 3

[13] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3D object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017. 2

[14] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510, 2022. 1

[15] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3D semantic segmentation with submanifold sparse convolutional networks. *CVPR*, 2018. 1

[16] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. *arXiv preprint arXiv:2002.10099*, 2020. 8

[17] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3D surface generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 216–224, 2018. 2

[18] JunYoung Gwak, Christopher Choy, and Silvio Savarese. Generative sparse detection networks for 3D single-shot object detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*, pages 297–313. Springer, 2020. 3

[19] Shi-Min Hu, Zheng-Ning Liu, Meng-Hao Guo, Jun-Xiong Cai, Jiahui Huang, Tai-Jiang Mu, and Ralph R Martin. Subdivision-based mesh convolution networks. *ACM Transactions on Graphics (TOG)*, 41(3):1–16, 2022. 3

[20] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, page 0, 2006. 1, 2

[21] Michael Kazhdan and Hugues Hoppe. Screened Poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):1–13, 2013. 1, 2, 6, 7

[22] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[23] Tianyang Li, Xin Wen, Yu-Shen Liu, Hua Su, and Zhizhong Han. Learning deep implicit functions for 3D shapes with dynamic code clouds. In *CVPR*, pages 12840–12850, 2022. 3, 6

[24] Yiyi Liao, Simon Donné, and Andreas Geiger. Deep marching cubes: Learning explicit surface representations. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[25] Stefan Lionar, Daniil Emtsev, Dusan Svilarkovic, and Songyou Peng. Dynamic plane convolutional occupancy networks. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1829–1838, 2021. 3, 6, 7

[26] Feng Liu and Xiaoming Liu. Voxel-based 3D detection and reconstruction of multiple objects from a single image. *Advances in Neural Information Processing Systems*, 34:2413–2426, 2021. 3

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

[28] Minghua Liu, Lu Sheng, Sheng Yang, Jing Shao, and Shi-Min Hu. Morphing and sampling network for dense point cloud completion. (Cd), 2019. 2

[29] Shi-Lin Liu, Hao-Xiang Guo, Hao Pan, Peng-Shuai Wang, Xin Tong, and Yang Liu. Deep implicit moving least-squares functions for 3D reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1788–1797, 2021. 3

[30] Xiaoxiao Long, Cheng Lin, Peng Wang, Taku Komura, and Wenping Wang. Sparseneus: Fast generalizable neural surface reconstruction from sparse views. In *European Conference on Computer Vision*, pages 210–227. Springer, 2022. 3

[31] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3D reconstruction in function space. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019-June:4455–4465, dec 2018. 1, 3, 5, 6, 7

[32] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018. 5

[33] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 3, 6

[34] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 12348 LNCS:523–540, mar 2020. 1, 3, 5, 6, 7, 8

[35] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-Janua:77–85, dec 2017. 2, 3

[36] Katja Schwarz, Axel Sauer, Michael Niemeyer, Yiyi Liao, and Andreas Geiger. VoxGRAF: Fast 3D-aware image synthesis with sparse voxel grids. *ARXIV*, 2022. 3

[37] Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. Deep marching tetrahedra: a hybrid representation for high-resolution 3D shape synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 3

[38] Vincent Sitzmann, Eric Chan, Richard Tucker, Noah Snavely, and Gordon Wetzstein. MetaSDF: Meta-learning signed distance functions. *Advances in Neural Information Processing Systems*, 33:10136–10147, 2020. 1, 3

[39] Jiaming Sun, Yiming Xie, Linghao Chen, Xiaowei Zhou, and Hujun Bao. Neuralrecon: Real-time coherent 3d reconstruction from monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15598–15607, 2021. 3

[40] 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*, pages 11358–11367, 2021. 3, 6

[41] Haotian Tang, Zhijian Liu, Xiuyu Li, Yujun Lin, and Song Han. Torchsparse: Efficient point cloud inference engine. *Proceedings of Machine Learning and Systems*, 4:302–315, 2022. 1

[42] Jiapeng Tang, Jiabao Lei, Dan Xu, Feiying Ma, Kui Jia, and Lei Zhang. Sa-convonet: Sign-agnostic optimization of convolutional occupancy networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6504–6513, 2021. 3

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

[44] M. Tatarchenko, S. R. Richter, R. Ranftl, Z. Li, V. Koltun, and T. Brox. What do single-view 3D reconstruction networks learn? In *CVPR*, 2019. 3

[45] Lyne P Tchapmi, Vineet Kosaraju, S. Hamid Rezatofighi, Ian Reid, and Silvio Savarese. TopNet: Structural point cloud decoder. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[46] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. NeuS: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021. 1

[47] Wenxuan Wu, Zhongang Qi, and Li Fuxin. PointConv: Deep convolutional networks on 3D point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 6, 7

[48] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems*, 34:4805–4815, 2021. 1

[49] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. PCN: Point completion network. *Proceedings - 2018 International Conference on 3D Vision, 3DV 2018*, pages 728–737, 2018. 2