Occupancy Networks: Learning 3D Reconstruction in Function Space

Occupancy Networks: Learning 3D Reconstruction in Function Space

Lars Mescheder[1] Michael Oechsle[1,2] Michael Niemeyer[1] Sebastian Nowozin[3†] Andreas Geiger[1]
[1]Autonomous Vision Group, MPI for Intelligent Systems and University of Tübingen
[2]ETAS GmbH, Stuttgart
[3]Google AI Berlin
{firstname.lastname}@tue.mpg.de nowozin@gmail.com

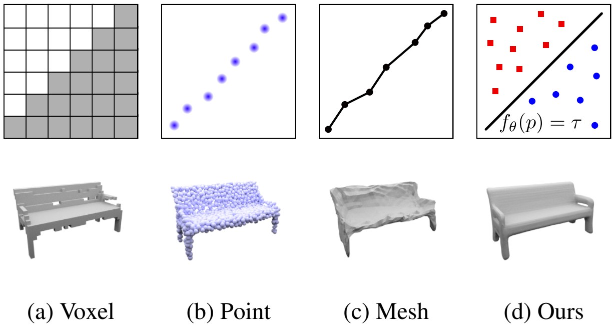
Abstract

With the advent of deep neural networks, learning-based approaches for 3D reconstruction have gained popularity. However, unlike for images, in 3D there is no canonical representation which is both computationally and memory efficient yet allows for representing high-resolution geometry of arbitrary topology. Many of the state-of-the-art learning-based 3D reconstruction approaches can hence only represent very coarse 3D geometry or are limited to a restricted domain. In this paper, we propose occupancy networks, a new representation for learning-based 3D reconstruction methods. Occupancy networks implicitly represent the 3D surface as the continuous decision boundary of a deep neural network classifier. In contrast to existing approaches, our representation encodes a description of the 3D output at infinite resolution without excessive memory footprint. We validate that our representation can efficiently encode 3D structure and can be inferred from various kinds of input. Our experiments demonstrate competitive results, both qualitatively and quantitatively, for the challenging tasks of 3D reconstruction from single images, noisy point clouds and coarse discrete voxel grids. We believe that occupancy networks will become a useful tool in a wide variety of learning-based 3D tasks.

1. Introduction

Recently, learning-based approaches for 3D reconstruction have gained popularity [4, 8, 19, 44, 58, 60]. In contrast to traditional multi-view stereo algorithms, learned models are able to encode rich prior information about the space of 3D shapes which helps to resolve ambiguities in the input.

While generative models have recently achieved remarkable successes in generating realistic high resolution images [28, 37, 56], this success has not yet been replicated in the 3D domain. In contrast to the 2D domain, the community has not yet agreed on a 3D output representation that is both memory efficient and can be efficiently inferred from data. Existing representations can be broadly categorized into three categories: voxel-based representations [4, 15, 34, 44, 49, 52, 58] , point-based representations [1, 14] and mesh representations [26, 43, 54], see Fig. 1.



(a) Voxel (b) Point (c) Mesh (d) Ours

Figure 1: **Overview:** Existing 3D representations discretize the output space differently: (a) spatially in voxel representations, (b) in terms of predicted points, and (c) in terms of vertices for mesh representations. In contrast, (d) we propose to consider the continuous decision boundary of a classifier f_θ (e.g., a deep neural network) as a 3D surface which allows to extract 3D meshes at any resolution.

Voxel representations are a straightforward generalization of pixels to the 3D case. Unfortunately, however, the memory footprint of voxel representations grows cubically with resolution, hence limiting naïve implementations to 32^3 or 64^3 voxels. While it is possible to reduce the memory footprint by using data adaptive representations such as octrees [47, 50], this approach leads to complex implementations and existing data-adaptive algorithms are still limited to relatively small 256^3 voxel grids. Point clouds [1, 14] and meshes [26, 43, 54] have been introduced as alternative representations for deep learning, using appropriate loss functions. However, point clouds lack the connectivity structure of the underlying mesh and hence require additional post-processing steps to extract 3D geometry from the model.

[†] Part of this work was done while at MSR Cambridge.

Existing mesh representations are typically based on deforming a template mesh and hence do not allow arbitrary topologies. Moreover, both approaches are limited in the number of points/vertices which can be reliably predicted using a standard feed-forward network.

In this paper, we propose a novel approach to 3D-reconstruction based on directly learning the *continuous* 3D occupancy function (Fig. 1d). Instead of predicting a voxelized representation at a fixed resolution, we predict the complete occupancy function with a neural network f_θ which can be evaluated at *arbitrary* resolution. This drastically reduces the memory footprint during training. At inference time, we extract the mesh from the learned model using a simple multi-resolution isosurface extraction algorithm which trivially parallelizes over 3D locations.

In summary, our **contributions** are as follows:

- We introduce a new representation for 3D geometry based on learning a continuous 3D mapping.
- We show how this representation can be used for reconstructing 3D geometry from various input types.
- We experimentally validate that our approach is able to generate high-quality meshes and demonstrate that it compares favorably to the state-of-the-art.

Besides our technical contributions, we also created a PyTorch package with high quality implementations of our method and several state-of-the-art baselines [8, 14, 34, 54] which we will make publicly available upon publication.

2. Related Work

Existing work on learning-based 3D reconstruction can be broadly categorized by the output representation they produce as either voxel-based, point-based or mesh-based.

Voxel Representations: Due to their simplicity, voxels are the most commonly used representation for discriminative [36, 41, 48] and generative [8, 19, 44, 49, 58, 60] 3D tasks.

Early works have considered the problem of reconstructing 3D geometry from a single image using 3D convolutional neural networks which operate on voxel grids [8, 51, 60]. Due to memory requirements, however, these approaches were limited to relatively small 32^3 voxel grids. While recent works [57, 59, 61] have applied 3D convolutional neural networks to resolutions up to 128^3, this is only possible with shallow architectures and small batch sizes, which leads to slow training.

The problem of reconstructing 3D geometry from multiple input views has been considered in [24, 27, 39]. Ji et al. [24] and Kar et al. [27] encode the camera parameters together with the input images in a 3D voxel representation and apply 3D convolutions to reconstruct 3D scenes from multiple views. Paschalidou et al. [39] introduced an architecture that predicts voxel occupancies from multiple images, exploiting multi-view geometry constraints [52].

Other works applied voxel representations to learn generative models of 3D shapes. Most of these methods are either based on variational auto-encoders [31, 45] or generative adversarial networks [20]. The former approach was taken by Brock et al. and Rezende et al. [4, 44], whereas the second approach was pursued by Wu et al. [58].

Due to the high memory requirements of voxel representations, recent works have proposed to reconstruct 3D objects in a multi-resolution fashion [22, 50]. However, the resulting methods are often complicated to implement and require multiple passes over the input to generate the final 3D model. Furthermore, they are still limited to comparably small 256^3 voxel grids. For achieving sub-voxel precision, several works [10, 33, 46] have proposed to predict truncated signed distance fields (TSDF) [9] where each point in a 3D grid stores the truncated signed distance to the closest 3D surface point. However, this representation is usually much harder to learn compared to occupancy representations as the network must reason about distance functions in 3D space instead of merely classifying a voxel as occupied or not. Moreover, this representation is still limited by the resolution of the underlying 3D grid.

Point Representations: An interesting alternative representation of 3D geometry is given by 3D point clouds which are widely used both in the robotics and in the computer graphics communities. Qi et al. [40, 42] pioneered point clouds as a representation for discriminative deep learning tasks. They achieved permutation invariance by applying a fully connected neural network to each point independently followed by a global pooling operation. Fan et al. [14] introduced point clouds as an output representation for 3D reconstruction. However, unlike other representations, this approachstat requires additional non-trivial post-processing steps [3, 6, 29, 30] to generate the final 3D mesh.

Mesh Representations: Meshes have first been considered for discriminative 3D classification or segmentation tasks by applying convolutions on the graph spanned by the mesh's vertices and edges [5, 21, 55].

More recently, meshes have also been considered as output representation for 3D reconstruction [25, 32, 53, 54]. Unfortunately, most of these approaches are prone to generating self-intersecting meshes. Moreover, they are only able to generate meshes with simple topology [54], require a reference template from the same object class [25, 32, 43] or cannot guarantee closed surfaces [53]. Liao et al. [34] proposed an end-to-end learnable version of the marching cubes algorithm [35]. However, their approach is still limited by the memory requirements of the underlying 3D grid and hence also restricted to 32^3 voxel resolution.

In contrast to the aforementioned approaches, our approach leads to high resolution closed surfaces without self-intersections and does not require template meshes from the same object class as input.

3. Method

In this section, we first introduce *Occupancy Networks* as a representation of 3D-geometry. We then describe how we can learn a model that infers this representation from various forms of input such as point clouds, single images and low-resolution voxel representations. Lastly, we describe a technique for extracting high-quality 3D meshes from our model at test time.

3.1. Occupancy Networks

As discussed in Section 2, voxel representations are often constrained to low-resolutions due to their cubic computational and memory requirements. As a result, they lack high-frequency details and show considerable discretization artifacts as illustrated in Fig. 1a.

Ideally, we would like to reason about the occupancy not only at fixed discrete 3D locations but instead at *every* possible 3D point $p \in \mathbb{R}^3$. We call the resulting function

$$o : \mathbb{R}^3 \to \{0, 1\} \tag{1}$$

the *occupancy function* of the 3D object. Our key insight is that we can approximate this 3D function with a neural network that assigns to every location $p \in \mathbb{R}^3$ an occupancy probability between 0 and 1. Note that this network is equivalent to a neural network for binary classification, except that we are interested in the decision boundary which implicitly represents the object's surface.

When using such a network for 3D reconstruction of an object based on observations of that object (e.g., image, point cloud, etc.), we must condition it on the input. Fortunately, we can make use of the following simple functional equivalence: a function that takes an observation $x \in \mathcal{X}$ as input and has a function from $p \in \mathbb{R}^3$ to $\mathbb{R}$ as output can be equivalently described by a function that takes a pair $(p, x) \in \mathbb{R}^3 \times \mathcal{X}$ as input and outputs a real number. The latter representation can be simply parameterized by a neural network f_θ that takes a pair (p, x) as input and outputs a real number which represents the probability of occupancy:

$$f_\theta : \mathbb{R}^3 \times \mathcal{X} \to [0, 1] \tag{2}$$

We call this network the *Occupancy Network*.

3.2. Training

To learn the parameters θ of the neural network $f_\theta(p, x)$, we randomly sample points in the 3D bounding volume of the object under consideration: for the i-th sample in a training batch we sample K points $p_{ij} \in \mathbb{R}^3$, $j = 1, \ldots, K$. We then evaluate the mini-batch loss $\mathcal{L}_\mathcal{B}$ at those locations:

$$\mathcal{L}_\mathcal{B}(\theta) = \frac{1}{|\mathcal{B}|} \sum_{i=1}^{|\mathcal{B}|} \sum_{j=1}^{K} \mathcal{L}(f_\theta(p_{ij}, x_i), o_{ij}) \tag{3}$$

Here, x_i is the i'th observation of batch $\mathcal{B}$, $o_{ij} \equiv o(p_{ij})$ denotes the true occupancy at point p_{ij}, and $\mathcal{L}(\cdot, \cdot)$ is a cross-entropy classification loss.

The performance of our method depends on the sampling scheme that we employ for drawing the locations p_{ij} that are used for training. In Section 4.6 we perform a detailed ablation study comparing different sampling schemes. In practice, we found that sampling uniformly inside the bounding box of the object with an additional small padding yields the best results.

Our 3D representation can also be used for learning probabilistic latent variable models. Towards this goal, we introduce an encoder network $g_\psi(\cdot)$ that takes locations p_{ij} and occupancies o_{ij} as input and predicts mean μ_ψ and standard deviation σ_ψ of a Gaussian distribution $q_\psi(z|(p_{ij}, o_{ij})_{j=1:K})$ on latent $z \in \mathbb{R}^L$ as output. We optimize a lower bound [17, 31, 45] to the negative log-likelihood of the generative model $p((o_{ij})_{j=1:K}|(p_{ij})_{j=1:K})$:

$$\begin{aligned} \mathcal{L}_\mathcal{B}^{\text{gen}}(\theta, \psi) = \frac{1}{|\mathcal{B}|} \sum_{i=1}^{|\mathcal{B}|} \Big[\sum_{j=1}^{K} \mathcal{L}(f_\theta(p_{ij}, z_i), o_{ij}) \\ + \text{KL}\left(q_\psi(z|(p_{ij}, o_{ij})_{j=1:K}) \,\|\, p_0(z)\right) \Big] \end{aligned} \tag{4}$$

where KL denotes the KL-divergence, $p_0(z)$ is a prior distribution on the latent variable z_i (typically Gaussian) and z_i is sampled according to $q_\psi(z_i|(p_{ij}, o_{ij})_{j=1:K})$.

3.3. Inference

For extracting the isosurface corresponding to a new observation given a trained occupancy network, we introduce *Multiresolution IsoSurface Extraction (MISE)*, a hierarchical isosurface extraction algorithm (Fig. 2). MISE enables us to extract high resolution meshes from the occupancy network without densely evaluating all points of a high-dimensional occupancy grid.

We first discretize the volumetric space at an initial resolution and evaluate the occupancy network $f_\theta(p, x)$ for all p in this grid. We mark all grid points p as occupied for which $f_\theta(p, x)$ is bigger or equal to some threshold[1] τ. Next, we mark all voxels as active for which at least two adjacent grid points have differing occupancy predictions. These are the voxels which would intersect the mesh if we applied the marching cubes algorithm at the current resolution. We subdivide all active voxels into 8 subvoxels and evaluate all new grid points which are introduced to the occupancy grid through this subdivision. We repeat these steps until the desired final resolution is reached. At this final resolution,

[1] The threshold τ is the only hyperparameter of our occupancy network. It determines the "thickness" of the extracted 3D surface. In our experiments we cross-validate this threshold on a validation set.

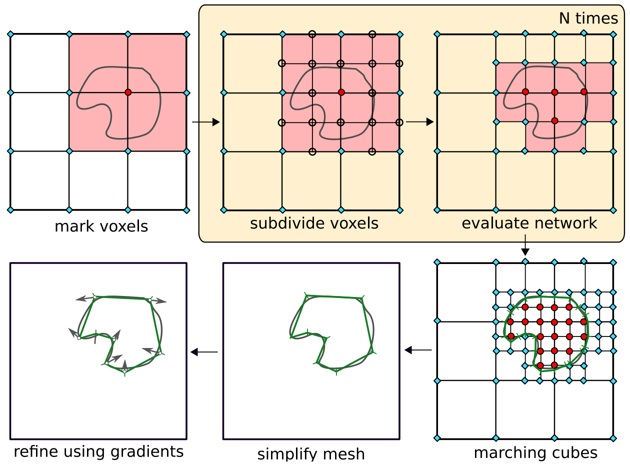


Figure 2: **Multiresolution IsoSurface Extraction:** We first mark all points at a given resolution which have already been evaluated as either occupied (red circles) or unoccupied (cyan diamonds). We then determine all voxels that have both occupied and unoccupied corners and mark them as active (light red) and subdivide them into 8 subvoxels each. Next, we evaluate all new grid points (empty circles) that have been introduced by the subdivision. The previous two steps are repeated until the desired output resolution is reached. Finally we extract the mesh using the marching cubes algorithm [35], simplify and refine the output mesh using first and second order gradient information.

we apply the Marching Cubes algorithm [35] to extract an approximate isosurface

$$\{p \in \mathbb{R}^3 \mid f_\theta(p, x) = \tau\}. \quad (5)$$

Our algorithm converges to the correct mesh if the occupancy grid at the initial resolution contains points from every connected component of both the interior and the exterior of the mesh. It is hence important to take an initial resolution which is high enough to satisfy this condition. In practice, we found that an initial resolution of 32^3 was sufficient in almost all cases.

The initial mesh extracted by the Marching Cubes algorithm can be further refined. In a first step, we simplify the mesh using the Fast-Quadric-Mesh-Simplification algorithm[2] [16]. Finally, we refine the output mesh using first and second order (i.e., gradient) information. Towards this goal, we sample random points p_k from each face of the output mesh and minimize the loss

$$\sum_{k=1}^{K}(f_\theta(p_k, x) - \tau)^2 + \lambda \left\| \frac{\nabla_p f_\theta(p_k, x)}{\|\nabla_p f_\theta(p_k, x)\|} - n(p_k) \right\|^2 \quad (6)$$

where $n(p_k)$ denotes the normal vector of the mesh at p_k. In practice, we set $\lambda = 0.01$. Minimization of the second term in (6) uses second order gradient information and can be efficiently implemented using Double-Backpropagation [12].

Note that this last step removes the discretization artifacts of the Marching Cubes approximation and would not be possible if we had directly predicted a voxel-based representation. In addition, our approach also allows to efficiently extract normals for all vertices of our output mesh by simply backpropagating through the occupancy network. In total, our inference algorithm requires 3s per mesh.

3.4. Implementation Details

We implemented our occupancy network using a fully-connected neural network with 5 ResNet blocks [23] and condition it on the input using conditional batch normalization [11, 13]. We exploit different encoder architectures depending on the type of input. For single view 3D reconstruction, we use a ResNet18 architecture. For point clouds we use the PointNet encoder [40]. For voxelized inputs, we use a 3D convolutional neural network [36]. For unconditional mesh generation, we use a PointNet [40] for the encoder network g_ψ. More details are provided in the supplementary material.

4. Experiments

We conduct three types of experiments to validate the proposed occupancy networks. First, we analyze the **representation power** of occupancy networks by examining how well the network can reconstruct complex 3D shapes from a learned latent embedding. This gives us an upper bound on the results we can achieve when conditioning our representation on additional input. Second, we **condition** our occupancy networks on images, noisy point clouds and low resolution voxel representations, and compare the performance of our method to several state-of-the-art baselines. Finally, we examine the **generative** capabilities of occupancy networks by adding an encoder to our model and generating unconditional samples from this model.[3]

Baselines: For the single image 3D reconstruction task, we compare our approach against several state-of-the-art baselines which leverage various 3D representations: we evaluate against 3D-R2N2 [8] as a voxel-based method, Point Set Generating Networks (PSGN) [14] as a point-based technique and Pixel2Mesh [54] as a mesh-based approach. For point cloud inputs, we adapted 3D-R2N2 and PSGN by changing the encoder. As Pixel2Mesh [54] is specialized to image inputs, we do not compare to Pixel2Mesh in this setting but instead use Deep Marching Cubes (DMC) [34] which has recently reported state-of-the-art results on this task. For the voxel super-resolution task we assess the improvements wrt. the input.

[2] https://github.com/sp4cerat/Fast-Quadric-Mesh-Simplification

[3] The code to reproduce all of our experiments will be made available upon publication.

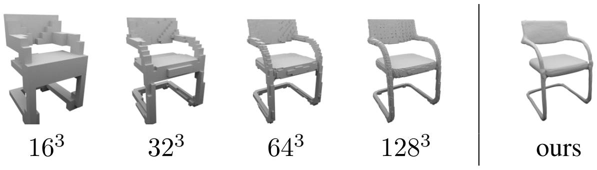


Figure 3: **Discrete vs. Continuous.** Qualitative comparison of our continuous representation (right) to voxelizations at various resolutions (left). Note how our representation encodes details which are lost in voxel-based representations.

Dataset: For all of our experiments we use the ShapeNet [7] subset of Choy et al. [8]. We also use the same voxelization, image renderings and train/test split as Choy et al. Moreover, we subdivide the training set into a training and a validation set on which we track the loss of our method and all baselines to determine when to stop training.

In order to generate watertight meshes and to determine if a point lies in the interior of a mesh (e.g., for measuring IoU) we use the code provided by Stutz et al. [49]. For a fair comparison, we sample points from the surface of the watertight mesh instead of the original model as ground truth for PSGN [14] and DMC [34]. All of our evaluations are conducted wrt. these watertight meshes.

Metrics: For evaluation we use the volumetric IoU, the Chamfer-L1 distance and a normal consistency score.

Volumetric IoU is defined as the quotient of the volume of the two meshes' union and the volume of their intersection. We obtain unbiased estimates of the volume of the intersection and the union by randomly sampling 100k points from the bounding volume and determining if the points lie inside our outside the ground truth / predicted mesh.

The Chamfer-L1 distance is defined as the mean of an accuracy and and a completeness metric. The accuracy metric is defined as the mean distance of points on the output mesh to their nearest neighbors on the ground truth mesh. The completeness metric is defined similarly, but in opposite direction. We estimate both distances efficiently by randomly sampling 100k points from both meshes and using a KD-tree to estimate the corresponding distances. Like Fan et al. [14] we use $1/10$ times the maximal edge length of the current object's bounding box as unit 1.

Finally, to measure how well the methods can capture higher order information, we define a normal consistency score as the mean absolute dot product of the normals in one mesh and the normals at the corresponding nearest neighbors in the other mesh.

4.1. Representation Power

In our first experiment, we investigate how well occupancy networks represent 3D geometry, independent of the inaccuracies of the input encoding. The question we try to answer in this experiment is whether our network can learn a memory efficient representation of 3D shapes while at the same time preserving as many details as possible. This gives us an estimate of the representational capacity of our model and an upper bound on the performance we may expect when conditioning our model on additional input. Similarly to [50], we embed each training sample in a 512 dimensional latent space and train our neural network to reconstruct the 3D shape from this embedding.

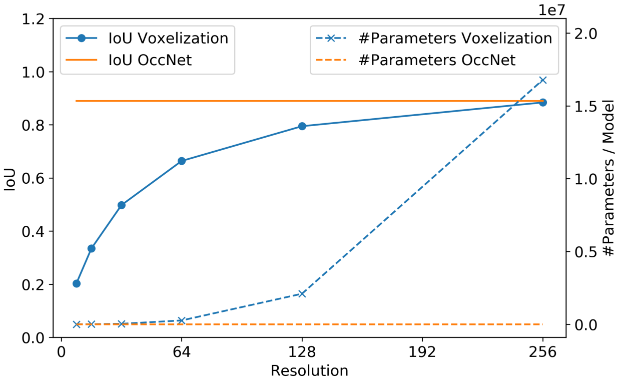


Figure 4: **IoU vs. Resolution.** This plot shows the IoU of a voxelization to the ground truth mesh (solid blue line) in comparison to our continuous representation (solid orange line) as well as the number of parameters per model needed for the two representations (dashed lines). Note how our representation leads to larger IoU wrt. the ground truth mesh compared to a low-resolution voxel representation. At the same time, the number of parameters of a voxel representation grows cubically with the resolution, whereas the number of parameters of occupancy networks is independent of the resolution.

We apply our method to the training split of the "chair" category of the ShapeNet dataset. This subset is challenging to represent as it is highly varied and many models contain high-frequency details. Since we are only interested in reconstructing the training data, we do not use separate validation and test sets for this experiment.

For evaluation, we measure the volumetric IoU to the ground truth mesh. Quantitative results and a comparison to voxel representations at various resolutions are shown in Fig. 4. We see that the Occupancy Network (OccNet) is able to faithfully represent the entire dataset with a high mean IoU of 0.89 while a low-resolution voxel representation is not able to represent the meshes accurately. At the same time, the occupancy network is able to encode all 4746 training samples with as little as 6M parameters, independently of the resolution. In contrast, the memory requirements of a voxel representation grow cubically with resolution. Qualitative results are shown in Fig. 3. We observe that the occupancy network enables us to represent details of the 3D geometry which are lost in a low-resolution voxelization.

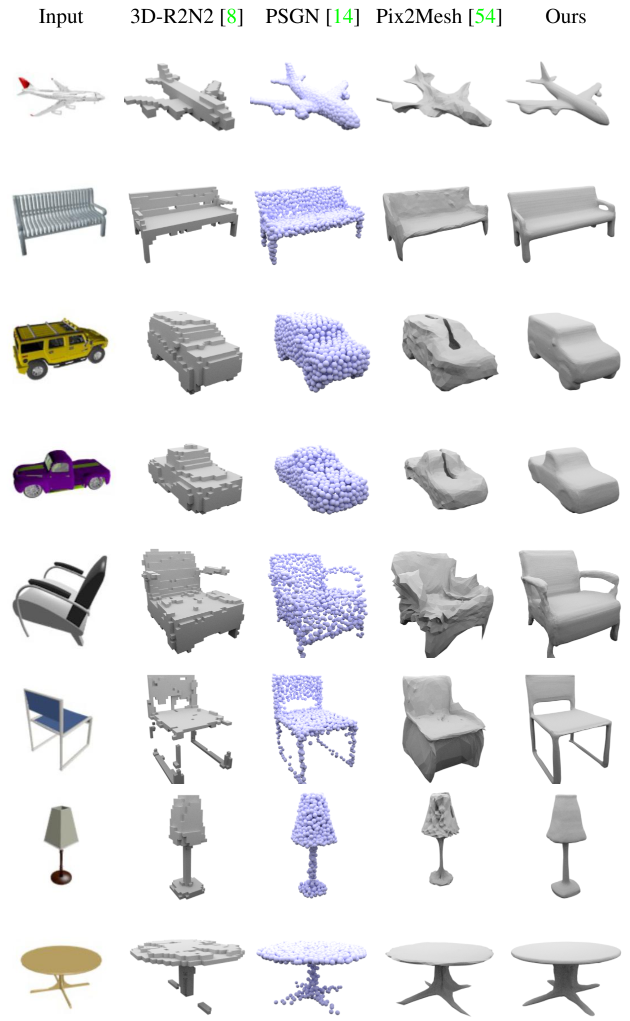


Figure 5: **Single Image 3D Reconstruction.** The input image is shown in the first column, the other columns show the results for our method compared to various baselines.

4.2. Single Image 3D Reconstruction

In our second experiment, we condition the occupancy network on an additional view of the object from a random camera location. The goal of this experiment is to evaluate how well occupancy functions can be inferred from complex input. While we train and test our method on the ShapeNet dataset, we also present qualitative results for the KITTI [18] and the Online Products dataset [38].

ShapeNet: In this experiment, we use a ResNet-18 image encoder, which was pretrained on the ImageNet dataset. For a fair comparison, we use the same image encoder for both 3D-R2N2 and PSGN[4]. For PSGN we use a fully connected decoder with 4 layers and 512 hidden units in each layer. The last layer projects the hidden representation to a 3072 dimensional vector which we reshape into 1024 3D points. As we use only a single input view, we remove the recurrent network in 3D-R2N2. We reimplemented the method of [54] in PyTorch, closely following the Tensorflow implementation provided by the authors.

For all methods, we track the loss and other metrics on the validation set and stop training as soon as the target metric reaches its optimum. For 3D-R2N2 and our method we use the IoU to the ground truth mesh as target metric, for PSGN and Pixel2Mesh we use the Chamfer distance to the ground truth mesh as target metric. To extract the final mesh, we use a threshold of 0.4 for 3D-R2N2 as suggested in the original publication [8]. To choose the threshold parameter τ for our method, we perform grid search on the validation set (see supplementary) and found that $\tau = 0.2$ yields a good trade-off between accuracy and completeness.

Qualitative results from our model and our baselines are shown in Fig. 5. We observe that all methods are able to capture the 3D geometry of the input image. However, 3D-R2N2 produces a very coarse representation and hence lacks details. In contrast, PSGN produces a high-fidelity output, but lacks connectivity. As a result, PSGN requires additional lossy post-processing steps to produce a final mesh[5]. Pixel2Mesh is able to create compelling meshes, but often misses holes in the presence of more complicated topologies. Such topologies are frequent, for example, for the "chairs" category in the ShapeNet dataset.

In contrast, our method is able to capture complex topologies, produces closed meshes and preserves most of the details. Please see the supplementary material for additional high resolution results and failure cases.

Quantitative results are shown in Table 1. We observe that our method achieves the highest IoU and normal consistency to the ground truth mesh. Surprisingly, while not trained wrt. Chamfer distance as PSGN or Pixel2Mesh, our method also achieves good results for this metric.

Real Data: To test how well our model generalizes to real data, we apply our network to the KITTI [18] and Online Products datasets [38]. To capture the variety in viewpoints of KITTI and Online Products, we rerendered all ShapeNet objects with random camera locations and retrained our network for this task.

For the KITTI dataset, we additionally use the instance masks provided in [2] to mask and crop car regions. We then feed these images into our neural network to predict the occupancy function. Some selected qualitative results are shown in Fig. 6a. Despite only trained on synthetic data, we observe that our method is also able to generate realistic reconstructions in this challenging setting.

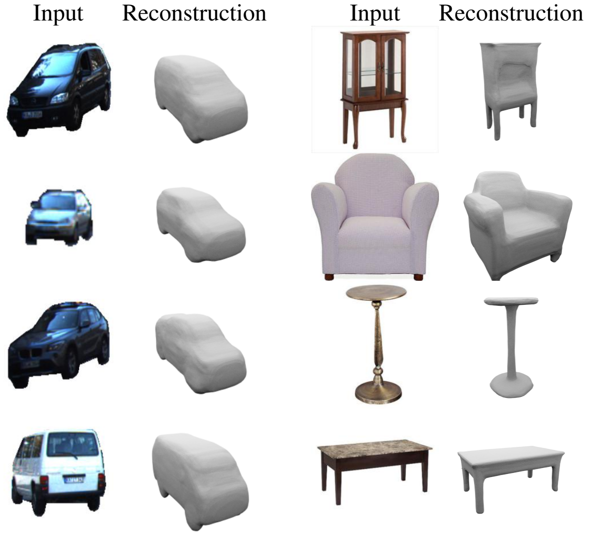
For the Online Products dataset, we apply the same pretrained model. Several qualitative results are shown in

[4] See supplementary for a comparison to the original architectures.

[5] See supplementary material for meshing results.

	IoU				Chamfer-L1				Normal Consistency			
category	3D-R2N2	PSGN	Pix2Mesh	OccNet	3D-R2N2	PSGN	Pix2Mesh	OccNet	3D-R2N2	PSGN	Pix2Mesh	OccNet
airplane	0.426	-	0.363	**0.547**	0.227	**0.137**	0.197	0.164	0.629	-	0.673	**0.828**
bench	0.373	-	0.288	**0.452**	0.194	**0.181**	0.217	0.220	0.678	-	0.684	**0.800**
cabinet	0.667	-	0.637	**0.732**	0.217	0.215	0.218	**0.168**	0.782	-	0.802	**0.879**
car	0.661	-	0.527	**0.731**	0.213	0.169	0.189	**0.166**	0.714	-	0.733	**0.850**
chair	0.439	-	0.362	**0.502**	0.270	0.247	0.283	**0.240**	0.663	-	0.696	**0.822**
display	0.440	-	0.439	**0.479**	0.314	0.284	**0.261**	0.289	0.719	-	0.775	**0.853**
lamp	0.281	-	0.296	**0.370**	0.778	**0.314**	0.316	0.497	0.560	-	0.624	**0.732**
loudspeaker	0.611	-	0.582	**0.653**	0.318	0.316	**0.294**	0.303	0.711	-	0.753	**0.836**
rifle	0.375	-	0.325	**0.458**	0.183	**0.134**	0.169	0.150	0.670	-	0.668	**0.763**
sofa	0.626	-	0.575	**0.671**	0.229	0.224	0.225	**0.208**	0.731	-	0.771	**0.863**
table	0.420	-	0.354	**0.506**	0.239	0.222	0.237	**0.194**	0.732	-	0.755	**0.857**
telephone	0.611	-	0.613	**0.709**	0.195	0.161	0.169	**0.148**	0.818	-	0.867	**0.935**
vessel	0.482	-	0.368	**0.521**	0.238	**0.188**	0.213	0.230	0.629	-	0.653	**0.794**
mean	0.493	-	0.441	**0.564**	0.278	**0.215**	0.230	0.229	0.695	-	0.727	**0.832**

Table 1: **Single Image 3D Reconstruction.** This table shows a numerical comparison of our approach and the baselines for single image 3D reconstruction on the ShapeNet dataset. We measure the IoU, Chamfer-L1 distance and Normal Consistency for various methods wrt. the ground truth mesh. Note that in contrast to prior work, we compute the IoU wrt. the high-resolution mesh and not a coarse voxel representation.



(a) KITTI (b) Online Products

Figure 6: **Qualitative results for real data.** We applied our trained model to the KITTI and Online Products datasets. Despite only trained on synthetic data, our model generalizes reasonably well to real data.

Fig. 6b. Again, we observe that our method generalizes reasonably well to real images despite being trained solely on synthetic data.

4.3. Point Cloud Completion

As a second conditional task, we apply our method to the problem of reconstructing the mesh from noisy point clouds. Towards this goal, we subsample 300 points from the surface of each of the (watertight) ShapeNet models and apply noise using a Gaussian distribution with zero mean and standard deviation 0.05 to the point cloud.

	IoU	Chamfer-L1	Normal Consistency
3D-R2N2	0.565	0.169	0.719
PSGN	-	0.202	-
DMC	0.645	0.126	0.835
OccNet	**0.762**	**0.087**	**0.891**

Table 2: **3D Reconstruction from Point Clouds.** This table shows a numerical comparison of our approach wrt. the baselines for 3D reconstruction from point clouds on the ShapeNet dataset. We measure IoU, Chamfer-L1 distance and Normal Consistency wrt. the ground truth mesh.

Again, we measure both the IoU and Chamfer-L1 distance wrt. the ground truth mesh. The results are shown in Table 2. We observe that our method achieves the highest IoU and normal consistency as well as the lowest Chamfer-L1 distance. Note that all numbers are significantly better than for the single image 3D reconstruction task. This can be explained by the fact that this task is much easier for the recognition model, as there is less ambiguity and the model only has to fill in the gaps.

4.4. Voxel Super-Resolution

As a final conditional task, we apply occupancy networks to 3D super-resolution. Here, the task is to reconstruct a high-resolution mesh from a coarse 32^3 voxelization of this mesh.

The results are shown in Table 3. We observe that our model improves IoU, Chamfer-L1 distance and normal consistency considerably compared to the coarse input mesh. Please see the supplementary for qualitative results.

	IoU	Chamfer-L1	Normal Consistency
Input	0.631	0.136	0.810
OccNet	**0.701**	**0.111**	**0.879**

Table 3: **Voxel Super-Resolution.** This table shows a numerical comparison of the output of our approach in comparison to the input on the ShapeNet dataset.

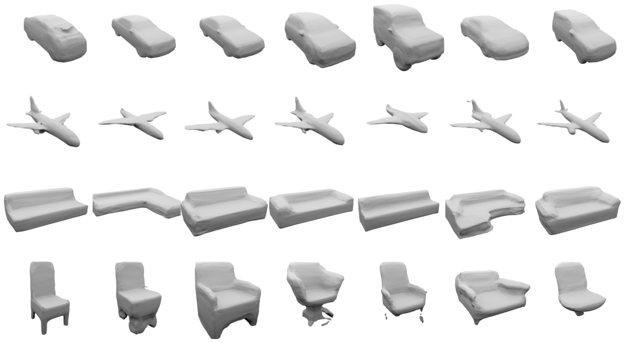


Figure 7: **Unconditional 3D Samples.** Random samples of our unsupervised models trained on the categories “car“, “airplane“, “sofa“ and “chair“ of the ShapeNet dataset. We see that our models are able to capture the distribution of 3D objects and produce compelling new samples.

4.5. Unconditional Mesh Generation

Finally, we apply our occupancy network to unconditional mesh generation, training it separately on four categories of the ShapeNet dataset in an unsupervised fashion. Our goal is to explore how well our model can represent the latent space of 3D models. Some samples are shown in Figure 7. Indeed, we find that our model can generate compelling new models. In the supplementary material we show interpolations in latent space for our model.

4.6. Ablation Study

In this section, we test how the various components of our model affect its performance on the single-image 3D-reconstruction task.

	IoU	Chamfer-L1	Normal Consistency
Uniform	**0.564**	**0.229**	0.832
Uniform (64)	0.554	0.256	0.829
Equal	0.475	0.291	**0.835**
Surface	0.536	0.254	0.822

(a) Influence of Sampling Strategy

	IoU	Chamfer-L1	Normal Consistency
Full model	**0.564**	**0.229**	**0.832**
No ResNet	0.559	0.243	0.831
No CBN	0.522	0.301	0.806

(b) Influence of Occupancy Network Architecture

Table 4: **Ablation Study.** When we vary the sampling strategy, we observe that uniform sampling in the bounding volume performs best. Similarly, when we vary the architecture, we find that our ResNet architecture with conditional batch normalization yields the best results.

Effect of sampling strategy First, we examine how the sampling strategy affects the performance of our final model. We try three different sampling strategies: (i) sampling 2048 points uniformly in the bounding volume of the ground truth mesh (uniform sampling), (ii) sampling 1024 points inside and 1024 points outside mesh (equal sampling) and (iii) sampling 1024 points uniformly and 1024 points on the surface of the mesh plus some Gaussian noise with standard deviation 0.1 (surface sampling). We also examine the effect of the number of sampling points by decreasing this number from 2048 to 64.

The results are shown in Table 4a. To our surprise, we find that uniform, the simplest sampling strategy, works best. We explain this by the fact that other sampling strategies introduce bias to the model: for example, when sampling an equal number of points inside and outside the mesh, we implicitly tell the model that every object has a volume of 0.5. Indeed, when using this sampling strategy, we observe thickening artifacts in the model’s output. Moreover, we find that reducing the number of sampling points from 2048 to 64 still leads to good performance, although the model does not perform as well as a model trained with 2048 sampling points.

Effect of architecture To test the effect of the various components of our architecture, we test two variations: (i) we remove the conditional batch normalization and replace it with a linear layer in the beginning of the network that projects the encoding of the input to the required hidden dimension and (ii) we remove all ResNet blocks in the decoder and replace them with linear blocks.

The results are presented in Table 4b. We find that both components are helpful to achieve good performance.

5. Conclusion

In this paper, we introduced occupancy networks, a new representation for 3D geometry. In contrast to existing representations, occupancy networks are not constrained by the discretization of the 3D space and can hence be used to represent realistic high-resolution meshes.

Our experiments demonstrate that occupancy networks are very expressive and can be used effectively both for supervised and unsupervised learning. We hence believe that occupancy networks are a useful tool which can be applied to a wide variety of 3D tasks.

Acknowledgements

This work was supported by Microsoft Research through its PhD Scholarship Programme.

References

[1] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. J. Guibas. Learning representations and generative models for 3D point clouds. In *Proc. of the International Conf. on Machine learning (ICML)*, 2018. 1

[2] H. A. Alhaija, S. K. Mustikovela, L. Mescheder, A. Geiger, and C. Rother. Augmented reality meets deep learning for car instance segmentation in urban scenes. In *Proc. of the British Machine Vision Conf. (BMVC)*, 2017. 6

[3] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE Trans. on Visualization and Computer Graphics (VCG)*, 5(4):349–359, 1999. 2

[4] A. Brock, T. Lim, J. M. Ritchie, and N. Weston. Generative and discriminative voxel modeling with convolutional neural networks. *arXiv.org*, 1608.04236, 2016. 1, 2

[5] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. Geometric deep learning: Going beyond euclidean data. *Signal Processing Magazine*, 34(4):18–42, 2017. 2

[6] F. Calakli and G. Taubin. SSD: smooth signed distance surface reconstruction. *Computer Graphics Forum*, 30(7):1993–2002, 2011. 2

[7] 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. *arXiv.org*, 1512.03012, 2015. 5

[8] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3D-R2N2: A unified approach for single and multi-view 3D object reconstruction. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2016. 1, 2, 4, 5, 6

[9] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *ACM Trans. on Graphics (SIGGRAPH)*, 1996. 2

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

[11] H. de Vries, F. Strub, J. Mary, H. Larochelle, O. Pietquin, and A. C. Courville. Modulating early visual processing by language. In *Advances in Neural Information Processing Systems (NIPS)*, 2017. 4

[12] H. Drucker and Y. Le Cun. Improving generalization performance using double backpropagation. *IEEE Trans. on Neural Networks*, 3(6):991–997, 1992. 4

[13] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville. Adversarially learned inference. In *Proc. of the International Conf. on Learning Representations (ICLR)*, 2017. 4

[14] H. Fan, H. Su, and L. J. Guibas. A point set generation network for 3D object reconstruction from a single image. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2, 4, 5, 6

[15] M. Gadelha, S. Maji, and R. Wang. 3D shape induction from 2d views of multiple objects. In *Proc. of the International Conf. on 3D Vision (3DV)*, 2017. 1

[16] M. Garland and P. S. Heckbert. Simplifying surfaces with color and texture using quadric error metrics. In *Visualization'98. Proceedings*, pages 263–269. IEEE, 1998. 4

[17] M. Garnelo, J. Schwarz, D. Rosenbaum, F. Viola, D. J. Rezende, S. M. A. Eslami, and Y. W. Teh. Neural processes. *arXiv.org*, 1807.01622, 2018. 3

[18] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The KITTI dataset. *International Journal of Robotics Research (IJRR)*, 32(11):1231–1237, 2013. 6

[19] R. Girdhar, D. F. Fouhey, M. Rodriguez, and A. Gupta. Learning a predictable and generative vector representation for objects. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2016. 1, 2

[20] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*, 2014. 2

[21] K. Guo, D. Zou, and X. Chen. 3D mesh labeling via deep convolutional neural networks. In *ACM Trans. on Graphics (SIGGRAPH)*, 2015. 2

[22] C. Häne, S. Tulsiani, and J. Malik. Hierarchical surface prediction for 3D object reconstruction. In *Proc. of the International Conf. on 3D Vision (3DV)*, 2017. 2

[23] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016. 4

[24] M. Ji, J. Gall, H. Zheng, Y. Liu, and L. Fang. SurfaceNet: an end-to-end 3D neural network for multiview stereopsis. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2017. 2

[25] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik. End-to-end recovery of human shape and pose. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[26] A. Kanazawa, S. Tulsiani, A. A. Efros, and J. Malik. Learning category-specific mesh reconstruction from image collections. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2018. 1

[27] A. Kar, C. Häne, and J. Malik. Learning a multi-view stereo machine. In *Advances in Neural Information Processing Systems (NIPS)*, 2017. 2

[28] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *Proc. of the International Conf. on Learning Representations (ICLR)*, 2018. 1

[29] M. M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Eurographics Symposium on Geometry Processing (SGP)*, 2006. 2

[30] M. M. Kazhdan and H. Hoppe. Screened poisson surface reconstruction. *ACM Trans. on Graphics (SIGGRAPH)*, 32(3):29, 2013. 2

[31] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *Proc. of the International Conf. on Learning Representations (ICLR)*, 2014. 2, 3

[32] C. Kong, C.-H. Lin, and S. Lucey. Using locally corresponding CAD models for dense 3D reconstructions from a single image. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[33] L. Ladicky, O. Saurer, S. Jeong, F. Maninchedda, and M. Pollefeys. From point clouds to mesh using regression. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2017. 2

[34] Y. Liao, S. Donne, and A. Geiger. Deep marching cubes: Learning explicit surface representations. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2, 4, 5

[35] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *ACM Trans. on Graphics (SIGGRAPH)*, 1987. 2, 3, 4

[36] D. Maturana and S. Scherer. Voxnet: A 3D convolutional neural network for real-time object recognition. In *Proc. IEEE International Conf. on Intelligent Robots and Systems (IROS)*, 2015. 2, 4

[37] L. Mescheder, A. Geiger, and S. Nowozin. Which training methods for GANs do actually converge? In *Proc. of the International Conf. on Machine learning (ICML)*, 2018. 1

[38] H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese. Deep metric learning via lifted structured feature embedding. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016. 6

[39] D. Paschalidou, A. O. Ulusoy, C. Schmitt, L. van Gool, and A. Geiger. Raynet: Learning volumetric 3D reconstruction with ray potentials. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[40] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 4

[41] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. Guibas. Volumetric and multi-view CNNs for object classification on 3D data. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[42] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems (NIPS)*, 2017. 2

[43] A. Ranjan, T. Bolkart, S. Sanyal, and M. J. Black. Generating 3D faces using convolutional mesh autoencoders. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2018. 1, 2

[44] D. J. Rezende, S. M. A. Eslami, S. Mohamed, P. Battaglia, M. Jaderberg, and N. Heess. Unsupervised learning of 3D structure from images. In *Advances in Neural Information Processing Systems (NIPS)*, 2016. 1, 2

[45] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proc. of the International Conf. on Machine learning (ICML)*, 2014. 2, 3

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

[47] G. Riegler, A. O. Ulusoy, and A. Geiger. OctNet: Learning deep 3D representations at high resolutions. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1

[48] S. Song and J. Xiao. Deep sliding shapes for amodal 3D object detection in RGB-D images. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 2

[49] D. Stutz and A. Geiger. Learning 3D shape completion from laser scan data with weak supervision. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2, 5

[50] M. Tatarchenko, A. Dosovitskiy, and T. Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3D outputs. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2017. 1, 2, 5

[51] S. Tulsiani, T. Zhou, A. A. Efros, and J. Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[52] A. O. Ulusoy, A. Geiger, and M. J. Black. Towards probabilistic volumetric reconstruction using ray potentials. In *Proc. of the International Conf. on 3D Vision (3DV)*, 2015. 1, 2

[53] M. Vakalopoulou, G. Chassagnon, N. Bus, R. Marini, E. I. Zacharaki, M. Revel, and N. Paragios. AtlasNet: Multi-atlas non-linear deep networks for medical image segmentation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[54] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang. Pixel2Mesh: Generating 3D mesh models from single RGB images. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2018. 1, 2, 4, 6

[55] P. Wang, Y. Gan, Y. Zhang, and P. Shui. 3D shape segmentation via shape fully convolutional networks. *Computers & Graphics*, 1702.08675, 2017. 2

[56] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional GANs. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1

[57] J. Wu, Y. Wang, T. Xue, X. Sun, B. Freeman, and J. Tenenbaum. MarrNet: 3D shape reconstruction via 2.5D sketches. In *Advances in Neural Information Processing Systems (NIPS)*, 2017. 2

[58] 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 *Advances in Neural Information Processing Systems (NIPS)*, 2016. 1, 2

[59] J. Wu, C. Zhang, X. Zhang, Z. Zhang, W. T. Freeman, and J. B. Tenenbaum. Learning shape priors for single-view 3D completion and reconstruction. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2018. 2

[60] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3D ShapeNets: A deep representation for volumetric

shapes. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015. 1, 2

[61] X. Zhang, Z. Zhang, C. Zhang, J. B. Tenenbaum, W. T. Freeman, and J. Wu. Learning to reconstruct shapes from unseen classes. In *Advances in Neural Information Processing Systems (NIPS)*, 2018. 2