Learning Implicit Fields for Generative Shape Modeling

Zhiqin Chen
Simon Fraser University
zhiqinc@sfu.ca

Hao Zhang
Simon Fraser University
haoz@sfu.ca

Abstract

We advocate the use of implicit fields *for learning generative models of shapes and introduce an* implicit field decoder*, called* IM-NET*, for shape generation, aimed at improving the visual quality of the generated shapes. An implicit field assigns a value to each point in 3D space, so that a shape can be extracted as an iso-surface. IM-NET is trained to perform this assignment by means of a* binary classifier*. Specifically, it takes a point coordinate, along with a feature vector encoding a shape, and outputs a value which indicates whether the point is outside the shape or not. By replacing conventional decoders by our implicit decoder for representation learning (via IM-AE) and shape generation (via IM-GAN), we demonstrate superior results for tasks such as generative shape modeling, interpolation, and single-view 3D reconstruction, particularly in terms of visual quality. Code and supplementary material are available at https://github.com/czq142857/implicit-decoder.*

1. Introduction

Unlike images and video, 3D shapes are not confined to one standard representation. Up to date, deep neural networks for 3D shape analysis and synthesis have been developed for voxel grids [18, 45], multi-view images [39], point clouds [1, 32], and integrated surface patches [16]. Specific to generative modeling of 3D shapes, despite the many progresses made, the shapes produced by state-of-the-art methods still fall far short in terms of visual quality. This is reflected by a combination of issues including low-resolution outputs, overly smoothed or discontinuous surfaces, as well as a variety of topological noise and irregularities.

In this paper, we explore the use of *implicit fields* for learning deep models of shapes and introduce an *implicit field decoder* for shape generation, aimed at improving the visual quality of the generated models, as shown in Figure 1. An implicit field assigns a value to each point (x, y, z). A shape is represented by all points assigned to a specific value and is typically rendered via iso-surface extraction such as Marching Cubes. Our implicit field decoder, or simply implicit encoder, is trained to perform this assignment task, by means of a *binary classifier*, and it has a very simple architecture; see Figure 2. Specifically, it takes a point coordinate (x, y, z), along with a feature vector encoding a shape, and outputs a value which indicates whether the point is outside the shape or not. In a typical application setup, our decoder, which is coined *IM-NET*, would follow an encoder which outputs the shape feature vectors and then return an implicit field to define an output shape.

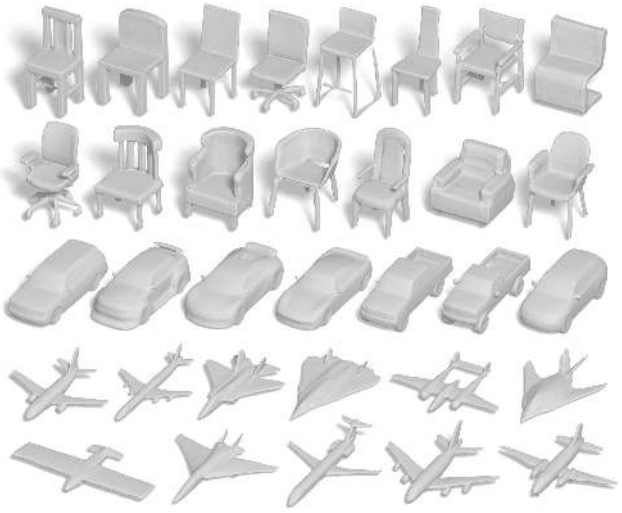

Figure 1: 3D shapes generated by IM-GAN, our *implicit field* generative adversarial network, which was trained on 64^3 or 128^3 voxelized shapes. The output shapes are sampled at 512^3 resolution and rendered after Marching Cubes.

Several novel features of IM-NET impact the visual quality of the generated shapes. First, the decoder output can be sampled at any resolution and is not limited by the resolution of the training shapes; see Figure 1. More importantly, we concatenate point coordinates with shape features, feeding both as input to our implicit decoder, which learns the inside/outside status of any point relative to a shape. In contrast, a classical convolution/deconvolution-based neural network (CNN) operating on voxelized shapes is typically trained to predict voxels relative to the extent of the bounding volume of a shape. Such a network learns

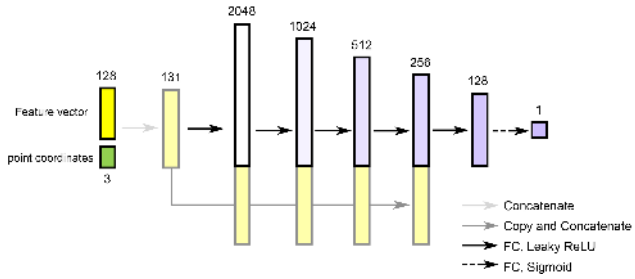

Figure 2: *Network structure of our implicit decoder, IM-NET.* The network takes as input a feature vector extracted by a shape encoder, as well as a 3D or 2D point coordinate, and it returns a value indicating the inside/outside status of the point relative to the shape. The encoder can be a CNN or use PointNET [32], depending on the application.

voxel distributions over the volume, while IM-NET learns *shape boundaries*; this is well exemplified in Figure 3 (top). Experiments show that shapes generated by our network possess higher surface quality than results from previous methods, as shown in Figure 1 and results in Section 4.

In addition, shape evolution is a direct result of changing the assignments of point coordinates to their inside/outside status and such assignments are precisely what our network, IM-NET, learns. In contrast, convolution kernels compute voxels as weighted averages, where the kernel windows are not "shape-aware". Thus a CNN-based decoder typically evolves shape geometries by means of intensity variations; see Figure 3 (bottom). As a result, our network produces cleaner interpolation results than previous works, even when there are topological changes; see Figure 5.

We embed IM-NET into several contemporary analysis and synthesis frameworks, including autoencoders (AEs), variational autoencoders (VAEs), and generative adversarial networks (GANs), by replacing the decoders employed by current approaches with ours, leading to *IM-AEs* and *IM-GANs*. This allows assessing the capabilities of our novel decoder for tasks such as shape representation learning, 2D or 3D shape generation, shape interpolation, as well as single-view 3D shape reconstruction. Extensive experiments and comparative studies, both quantitative and qualitative, demonstrate the superiority of our network over previous works, particularly in terms of *visual quality*.

2. Related work

There have been a variety of 3D shape representations for deep learning of shapes, such as voxel grids [9, 14, 28, 45, 46], octrees [18, 35, 40, 43, 44], multi-view images [29, 39], point clouds [1, 12, 13, 32, 33, 47, 48], geometry images [37, 38], deformable mesh/patches [16, 38, 42, 47], and part-based structural graphs [27, 49]. To the best of our knowledge, our work is the first to introduce a deep network for learning implicit fields for generative shape modeling.

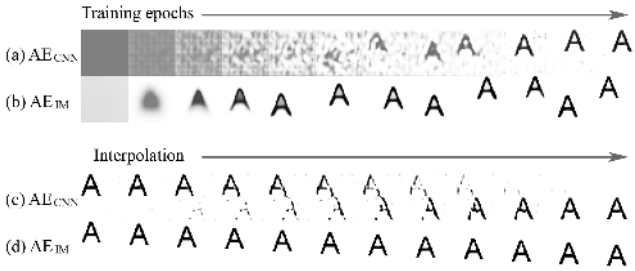

Figure 3: *CNN-based decoder vs. our implicit decoder.* We trained two autoencoders with CNN decoder (AE$_{CNN}$) and our implicit decoder (AE$_{IM}$), respectively, on a synthesized dataset of letter A's on white background. The two models have the same CNN encoder. (a) and (b) show the sampled images during AE training. (c) and (d) show interpolation sequences produced by the two trained AEs. See more comparisons in the supplementary material.

With remarkable progress made on generative modeling of images using VAEs [24], GANs [3, 15, 34], autoregressive networks [41], and flow-based models [23], there have been considerably fewer works on generative models of 3D shapes. Girdhar et al. [14] learned an embedding space of 3D voxel shapes for 3D shape inference from images and shape generation. Wu et al. [45] extended GANs from images to voxels and their 3DGAN was trained to generate 3D voxel shapes from latent vectors. Achlioptas et al. [1] proposed a latent-GAN workflow that first trains an autoencoder with a compact bottleneck layer to learn a latent representation of point clouds, then trains a plain GAN on the latent code. Common issues with these methods include limited model resolution, uneven and noisy shape surfaces, and inability to produce smooth shape interpolation.

Recently, Li et al. [27] introduced a part-based autoencoder for 3D shape structures, i.e., a hierarchical organization of part bounding boxes. The autoencoder is tuned with an adversarial loss to become generative. Then a separate network is trained to fill in part geometries within the confines of the part bounding boxes. Their method can produce cleaner 3D shapes and interpolation results, mainly owing to the decoupling of structure and geometry generation. However, their network has to be trained by segmented shapes with structural hierarchies. In contrast, our implicit encoder is trained on unstructured voxel shapes.

The output of our decoder IM-NET can be sampled at resolutions higher than that of the training shapes, however, it is not designed for the purpose of (voxel) super-resolution. There have been works on single image super-resolution using deep networks, *e.g.*, [11, 25], which are trained with low- and high-resolution image pairs. Progressive training [22] is another technique to improve image quality, and we adopt it in our work to reduce training times.

Most learning-based methods for single-view 3D reconstruction encode input images with deep convolutional net-

works, then use an appropriate decoder to reconstruct 3D shapes depending on the shape representations. The most commonly used representations are voxels [9, 14, 46] and point clouds [12, 13]. Voxels are natural extensions of image pixels, which allow migrating state-of-the-art techniques from image processing to shape processing. However, voxel representations are usually constrained by GPU memory size, resulting in low-resolution results. Octree representations attempt to fix the memory issues by predicting surfaces in a coarse-to-fine manner [18, 44].

The recent work by Huang et al. [21] also trains a network to perform binary classification, like IM-NET. However, the key distinction is that our network assigns inside/outside based on spatial point coordinates. Hence, it learns shape boundaries and effectively, an implicit function that is *Lipschitz continuous* over point coordinates, i.e., it maps close-by points to similar output values. Moreover, IM-NET can input an arbitrary 3D point and learn a continuous implicit field without discretization. In contrast, their network operates on convolutional features computed over *discretized* images and learns the inside/outside assignment based on multi-scale image features at a point x.

Point clouds can be lightweight on the decoder side by producing few thousand points. However, these points do not provide any surface or topological information and pose a reconstruction challenge. In Deep Marching Cubes, Liao et al. [28] proposed a differentiable formulation of marching cubes to train an end-to-end 3D CNN model for mesh reconstruction from point clouds. However, the resulting mesh still shares common issues with other CNN-based networks, *e.g.*, low resolution (32^3) and topological noise.

Some other works [16, 38] deform a surface template (*e.g.*, square patches or a sphere) onto a target shape. But many shapes cannot be well-represented by a single patch, while outputs from multi-patch integrations often contain visual artifacts due to gaps, foldovers, and overlaps. In Pixel2Mesh, Wang et al. [42] used a graph-based CNN [36] to progressively deform an ellipsoid template to fit an image. The end-to-end network directly generates meshes but the results tend to be overly smoothed, capturing only low-frequency features while restricted to sphere topology.

3. Implicit decoder and shape generation

An implicit field is defined by a continuous function over 2D/3D space. Then a mesh surface can be reconstructed by finding the zero-isosurface of the field with methods such as Marching Cubes [30]. In our work, we consider using neural networks to describe a shape in such an implicit way. For a closed shape, we define the inside/outside field $\mathcal{F}$ of the shape by taking the sign of its signed distance field:

$$\mathcal{F}(p) = \begin{cases} 0 & \text{if point } p \text{ is outside the shape,} \\ 1 & \text{otherwise.} \end{cases} \tag{1}$$

Assume the in-out field is restricted in a unit 3D space, we attempt to find a parameterization $f_\theta(p)$ with parameters θ that maps a point $p \in [0,1]^3$ to $\mathcal{F}(p)$. This is essentially a binary classification problem, which has been studied very well. Multi-layer perceptrons (MLPs) with rectified linear unit (ReLU) nonlinearities are ideal candidates for such tasks. With sufficient hidden units, MLP family is able to approximate the field $\mathcal{F}$ within any precision. This is a direct consequence of the universal approximation theorem [20]. Notice that using MLPs also gives us a representation which is continuous across space, so that the mesh can be recovered by taking the k-isosurface of the approximated field, where k is an appropriate threshold.

3.1. Data preparation

The training of such implicit model needs point-value pairs. It is natural to first voxelize or rasterize the shape for the sake of convenience and uniform sampling. For 3D shapes, we use the same technique as in Hierarchical Surface Prediction (HSP) [18] to get the voxel models in different resolutions (16^3, 32^3, 64^3, 128^3). We sample points on each resolution in order to train the model progressively.

A naive sampling would take the center of each voxel and produce n^3 points. A more efficient approach when dealing with shapes is to sample more points near shape surfaces and neglect most points far away, leading to roughly $O(n^2)$ points. To compensate for the density change, we assign a weight w_p to each sampled point p, representing the inverse of the sampling density near p. The implementation of such a sampling method is flexible and varies according to resolution and shape category; more details can be found in the supplementary material. Most 2D shapes are already rasterized into images. For simplicity, we apply the naive sampling approach for images with an appropriate threshold to determine whether a pixel belongs to the shape.

3.2. Network structure of IM-NET

Our model is illustrated in Figure 2. The skip connections (copy and concatenate) in the model can make the learning progress faster in the experiments. They can be removed when the feature vector is long, so as to prevent the model from becoming too large. The loss function is a weighted mean squared error between ground truth labels and predicted labels for each point. Let S be a set of points sampled from the target shape, we have:

$$\mathcal{L}(\theta) = \frac{\sum_{p \in S} |f_\theta(p) - \mathcal{F}(p)|^2 \cdot w_p}{\sum_{p \in S} w_p} \tag{2}$$

3.3. Shape generation and other applications

Our implicit field decoder, IM-NET, can be embedded into different shape analysis and synthesis frameworks to

support various applications. In this paper, we demonstrate shape autoencoding, 2D and 3D shape generation, and single-view 3D reconstruction. Due to page limit, the models are briefly introduced here. Detailed structures and hyperparameters can be found in the supplementary material.

For auto-encoding 3D shapes, we used a 3D CNN as encoder to extract 128-dimensional features from 64^3 voxel models. We adopt progressive training techniques, to first train our model on 16^3 resolution data, then increase the resolution gradually. Notice that the structure of the model does not change when switching between training data at different resolutions, thus higher-resolution models can be trained with pre-trained weights on low-resolution data. In the experiments, progressive training can stabilize training process and significantly reduce training time.

For 3D shape generation, we employed latent-GANs [1, 2] on feature vectors learned by a 3D autoencoder. We did not apply traditional GANs trained on voxel grids since the training set is considerably smaller compared to the size of the output. Therefore, the pre-trained AE would serve as a means for dimensionality reduction, and the latent-GAN was trained on high-level features of the original shapes. We used two hidden fully-connected layers for both the generator and the discriminator, and the Wasserstein GAN loss with gradient penalty [3, 17]. In generative models for 2D shapes, we used the same structure as in the 3D case, except the encoder was 2D CNN and the decoder took a 2D point as input. We did not apply progressive training for 2D shapes since it is unnecessary when the images are small.

For single-view 3D reconstruction (SVR), we used the ResNET [19] encoder to obtain 128-D features from 128^2 images. We followed the idea from AtlasNET [16] to first train an autoencoder, then fix the parameters of the implicit decoder when training SVR. In our experiments, we adopted a more radical approach by only training the ResNET encoder to minimize the mean squared loss between the predicted feature vectors and the ground truth. This performed better than training the image-to-shape translator directly, since one shape can have many different views, leading to ambiguity. Pre-trained decoders provide strong priors that can not only reduce such ambiguity, but also shorten training time, since the decoder was trained on unambiguous data in the autoencoder phase and encoder training was independently from the decoder in SVR phase.

We reconstructed the 3D meshes by Marching Cubes, and 2D images by sampling a grid of points then optionally applying thresholding to obtain binarized shapes.

4. Results and evaluation

In this section, we show qualitative and quantitative results on various tasks using our implicit decoder, IM-NET, and compare them with state-of-the-art approaches. We used the dataset provided by [18], which contains 256^3-voxelized and flood-filled 3D models from ShapeNet Core dataset (v1) [7], and the corresponding rendered views. To compare with other methods that output point clouds, we first used Marching Cubes to obtain meshes from the 256^3-voxelized models, then used Poisson-disk Sampling [10] to obtain 10000 points. This gave us point clouds with only points on the surfaces of the meshes. We evaluated our method, and others, on five representative categories: plane, car, chair, rifle, and table. These categories contain 4,045, 7,497, 6,778, 2,373, and 8,509 3D shapes, respectively.

4.1. Quality metrics

In our experiments, both qualitative (via visual examination) and quantitative evaluations are provided. Specific to shapes, most evaluation metrics for encoding and reconstruction are based on point-wise distances, *e.g.*, chamfer distance (CD), or global alignment, *e.g.*, mean squared error (MSE) and intersection over union (IoU) on voxels. However, these may not be the best *visual* similarity or quality metrics. For example, slightly adjusting the angles between the legs of a chair and its seat may be barely perceptible, compared to removing one shape part, yet the latter could lead to a lower CD or IoU. Past works, *e.g.*, [31], have shown that low-frequency displacements (*e.g.*, bending a leg) in shapes are less noticeable than high-frequency errors over local surface characteristics such as normals and curvatures. Metrics such as MSE, CD, and IoU do not account for visual quality of the object surfaces.

A less frequently used visual similarity metric in the computer vision community, the light field descriptor (LFD) [8], has been widely adopted in computer graphics. Inspired by human vision system, LFD considers a set of rendered views of a 3D shape from various camera angles. Each projected image is then encoded using Zernike moments and Fourier descriptors for similarity comparisons.

4.2. Auto-encoding 3D shapes

We first compare IM-NET with CNN decoders. For each category, we sorted the shapes by name and used the first 80% as training set and the rest for testing. We trained one model with our implicit decoder (IM-AE) and another with 3D CNN decoder (CNN-AE) on each category. The 3D CNN decoder is symmetric to the 3D CNN encoder (details can be found in the supplementary material). Both models had the same encoder structure and were trained on 64^3 resolution for the same number of epochs between 200 and 400, depending on the size of the dataset.

Table 1 evaluates reconstruction results using several common evaluation metrics: MSE, IoU, symmetric Chamfer distance (CD), and LFD. MSE and IoU were computed against the ground truth 64^3 voxel models. For CD and LFD, we obtained meshes from the output voxel models by Marching Cubes with threshold 0.5. We sampled 2,048

	Plane	Car	Chair	Rifle	Table
CNN64-MSE	**1.47**	**4.37**	**7.76**	**1.62**	**5.80**
IM64-MSE	2.14	4.99	11.43	1.91	10.67
CNN64-IoU	**86.07**	**90.73**	**74.22**	**78.37**	**84.67**
IM64-IoU	78.77	89.26	65.65	72.88	71.44
CNN64-CD	**3.51**	5.31	**7.34**	**3.48**	**7.45**
IM64-CD	4.22	**5.28**	8.96	3.78	12.05
IM256-CD	4.23	5.44	9.05	3.77	11.54
CNN64-LFD	3,375	1,323	2,555	**3,515**	**1,824**
IM64-LFD	3,371	1,190	2,515	3,714	2,370
IM256-LFD	**3,236**	**1,147**	**2,453**	3,602	2,201

Table 1: *3D reconstruction errors.* CNN and IM represent CNN-AE and IM-AE, respectively, with 64 and 256 indicating sampling resolutions. The mean is taken over the first 100 shapes in each tested category. MSE is multiplied by 10^3, IoU by 10^2, and CD by 10^4. LFD is rounded to integers. Better-performing numbers are shown in boldface.

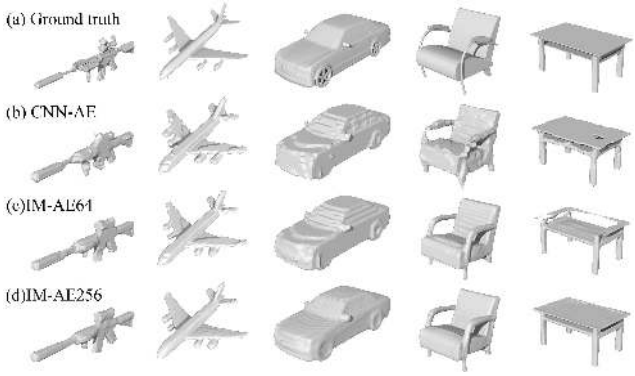

Figure 4: *Visual results for 3D reconstruction.* Each column presents one example from one category. IM-AE64 is sampled on 64^3 resolution and IM-AE256 on 256^3. All results are rendered using the same Marching Cubes setup.

points from the vertices for each output mesh and compare against ground truth point clouds to compute CD. Note that CNN-AE has fixed output size (64^3), yet our implicit model can be up-sampled to arbitrarily high resolution by adjusting the sampling grid size. In Table 1, IM256 is the same IM-AE model but sampled at 256^3.

Although CNN-AE beats IM-AE in nearly all five categories in terms of MSE, IOU, and CD, visual examination clearly reveals that IM-AE produces better results, as shown in Figure 4; more such results are available in the supplementary material. This validates that LFD is a better visual similarity metric for 3D shapes. On one hand, the movement of some parts, for example, table boards, may cause significant MSE, IOU and CD changes, but bring little visual changes; on the other hand, legs are usually thin, so that missing one leg may cause minor MSE, IOU and CD changes, but can bring significant visual changes. As remarked above, MSE, IOU and CD do not capture well

Figure 5: *3D shape interpolation results.* 3DGAN, CNN-GAN, and IM-GAN are sampled at 64^3 resolution to show the smoothness of the surface is not just a matter of sampling resolution. Notice that the morphing sequence of IM-GAN not only consists of smooth part movements (legs, board), but also handles topology changes.

surface quality: a smooth but not exactly aligned surface might have poorer evaluation results than an aligned jagged surface. The situation is better when using LFD. However, since LFD only renders the silhouette of the shape without lighting, it can only capture surface condition on the edge of the silhouette. We expect better evaluation metrics to be proposed in the future, and for the following experiments, we use LFD as our primary evaluation metric.

Note that in the table example in Figure 4, a 64^3 resolution represents an under-sampling, while going up to 256^3 reveals more details. This indicates that our generative model is able to generate a table board thinner than the resolution of the training data, which shows that the model learned the implicit field from the whole shape in the space rather than merely learning voxel distributions.

4.3. 3D shape generation and interpolation

Next, we assess and evaluate improvements made by our implicit decoder for generative modeling of 3D shapes. We trained latent-GANs on both CNN-AE and IM-AE to obtain CNN-GAN and IM-GAN. We also compared our results with two state-of-the-art methods, 3DGAN [45] and the generative model for point clouds in [1] (PC-GAN). For 3DGAN, we used the trained models made available online by the authors. PC-GAN was trained using latent WGAN [1]. The autoencoder of PC-GAN was trained on our aforementioned point cloud data for 400 epochs for each category. PC-GAN, CNN-GAN, and IM-GAN were trained on the training split of the dataset for 10,000 epochs. 3DGAN was not trained using train/test split [45].

To compare the generative schemes, we adopted the evaluation metrics from [1], but replaced CD or EMD by LFD. Suppose that we have a testing set G and a sample set A, for each shape in A, we find its closest neighbor in G using LFD, say g, and mark g as "matched". In the end, we

		Plane	Car	Chair	Rifle	Table	Average w/o planes	Average
COV-LFD (%)	3DGAN [45]		12.13	25.07	62.32	18.80	29.58	
	PC-GAN [1]	**73.55**	61.40	70.06	61.47	77.50	67.61	68.80
	CNN-GAN	69.22	**73.00**	**77.73**	61.26	83.73	73.93	72.99
	IM-GAN	70.33	69.33	75.44	**65.26**	**86.43**	**74.12**	**73.36**
MMD-LFD	3DGAN [45]		1,993	4,365	4,476	5,208	4,010	
	PC-GAN [1]	3,737	1,360	3,143	3,891	2,822	2,804	2,991
	CNN-GAN	3,745	1,288	3,012	3,819	2,594	2,678	2,892
	IM-GAN	**3,689**	**1,287**	**2,893**	**3,760**	**2,527**	**2,617**	**2,831**

Table 2: *Quantitative evaluation of 3D shape generation.* LFD is rounded to integers. See texts for explanation of the metrics.

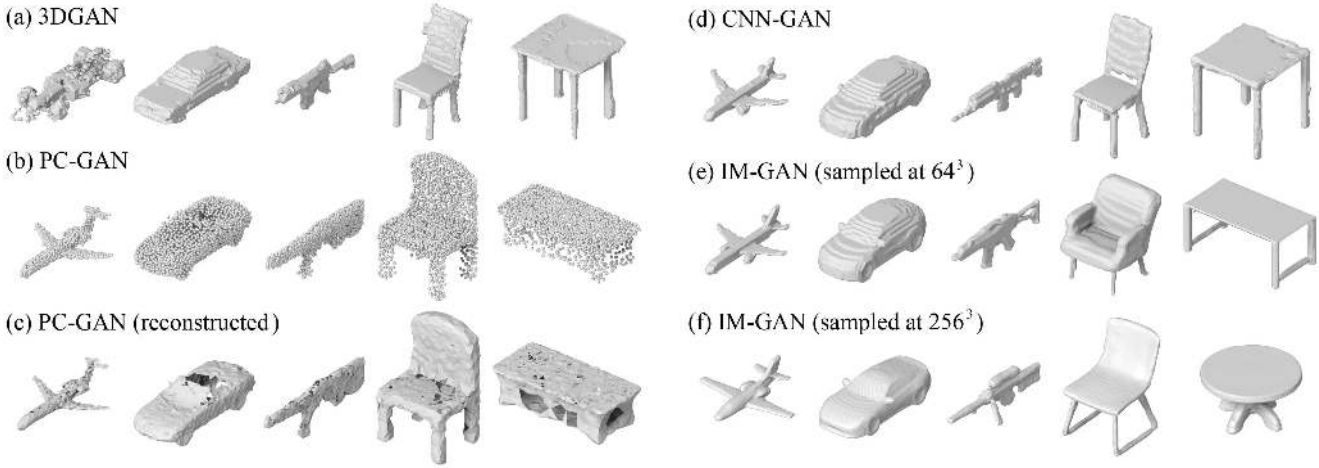

Figure 6: *3D shape generation results.* One generated shape from each category is shown for each model; more results are available in the supplementary material. Since the trained models of 3DGAN did not include plane category, we fill the blank with another car. The ball-pivoting method [6] was used for mesh reconstruction (c) from PC-GAN results (b).

calculate the percentage of G marked as "matched" to obtain the coverage score (COV-LFD) that roughly represents the diversity of the generated shapes. However, a random set may have a high coverage, since matched shapes need not be close. Therefore, we match every shape in G to the one in A with the minimum distance and compute the mean distances in the matching as Minimum Matching Distance (MMD-LFD). Ideally, a good generative model would have higher COV-LFD and lower MMD-LFD values.

We first sampled shapes using the subject generative model to obtain A, where the number of sampled shapes is five times the number of shapes in the testing split (G) of that category. For PC-GAN, we employed the ball-pivoting method [6] to reconstruct the shape surface, while for all the other generative models, we used Marching Cubes. IM-GAN was sampled at 64^3 in quantitative evaluation.

Quantitative and qualitative evaluations are shown in Table 2 and Figure 6, respectively. Overall, IM-GAN performs better on both COV-LFD and MMD-LFD. More importantly, IM-GAN generates shapes with better visual quality compared to other methods, in particular, with smoother and more coherent surfaces. 3DGAN appears to suffer from mode collapse on several categories, leading to lower coverage. Point clouds generated by PC-GAN are recognizable but lack detailed features; high-quality reconstruction from only 2048 generated points would be challenging. In addition, as shown in Figure 5, IM-GAN exhibits superior capability in 3D shape interpolation. As usual for the latent generative models, the interpolation is carried out by a linear interpolation between two latent codes; in-between 3D shapes are then generated from the intermediate codes.

We trained the IM-GAN further with 128^3 resolution data on category plane, car and rifle. Figure 1 shows some results sampled at 512^3. We also include, in the supplementary material, videos showing interpolation results of IM-GAN sampled at 256^3, and comparisons between interpolations in IM-AE and IM-GAN latent spaces.

4.4. 2D shape generation and interpolation

To evaluate IM-GAN for 2D shape generation, we conducted our experiments on the MNIST dataset since handwritten digits are naturally 2D shapes. We compared our results against DCGAN [34], VAE [24], and WGAN with gradient penalty [3, 17]. We also included the 2D version of CNN-GAN. In addition, we substituted the CNN decoders of VAE and WGAN with our implicit decoders, to obtain VAE_{IM} and WGAN_{IM}. We trained all models on 5,000 binarized images from the training split of

	DCGAN [34]	CNN-GAN	IM-GAN	VAE [24]	VAE$_{IM}$	WGAN [17]	WGAN$_{IM}$	Oracle
COV-CD (%)	3.9	**82.7**	75.2	72.1	**74.9**	**86.5**	84.7	88.4
MMD-CD	0.846	0.155	**0.151**	0.145	**0.14**	0.158	**0.149**	0.137
PWE (nat)	-282.83	-8.07	**-6.16**	17.39	**30.6**	-24.54	**-4.17**	18.99
PWE-nb (nat)	-230.47	**130.93**	128.38	304.57	**318.07**	**97.32**	93.1	241.19
IS [26]	3.26	8.79	**9.36**	9.09	**9.42**	8.9	**9.22**	9.8
IS-nb	3.26	8.8	**9.39**	7.58	**8.28**	8.95	**9.22**	9.88

Table 3: *Quantitative evaluation for 2D shape generation.* Oracle is the results obtained by using the training set as the sample set. The metrics without suffix "-nb" are evaluated using binarized images. PWE and IS are the higher the better. The better results in each subgroup (latent-GAN, VAE, WGAN) are bolded, and the best results over all models are underlined.

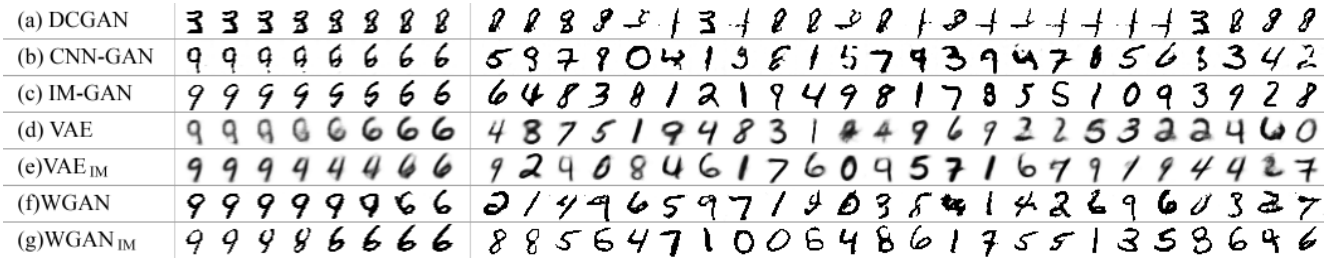

Figure 7: *Visual results for 2D shape generation.* The first part of each row presents an interpolation between 9 and 6, except for DCGAN since it failed to generate number 6 or 9. The second part of each row shows some generated samples. The sampled images are not binarized. More samples and binarized images can be found in the supplementary material.

MNIST dataset for 1,000 epochs. The training set contains a smaller-than-usual amount of images, so that we can better observe the different features learned by CNN models and implicit models. The autoencoders of IM-GAN and CNN-GAN were pre-trained for 200 epochs. We replace LFD with chamfer distance in 2D images to obtain COV-CD and MMD-CD for evaluation. We also report the inception score for MNIST (IS) [26] and the log-likelihood produced by Parzen-window estimate (PWE) [4, 5, 15]. For COV-CD and MMD-CD, we sampled 5,000 images from the subject models and compared against 1,000 ground truth images from the testing split. For IS and PWE, we sampled 10,000 images and used the entire testing split.

Quantitative and qualitative evaluations are shown in Table 3 and Figure 7, respectively. Models equipped with our implicit decoders generally perform better. Due to inadequate training samples, DCGAN suffers from mode collapse, suggesting that the WGAN loss is preferred to extract true features with smaller training sets. VAEs have better performance over GANs when the output images are binarized, since VAEs tend to produce blurry results. For interpolation, CNN based methods tend to make old parts disappear and then new parts appear. This phenomenon is especially apparent in CNN-GAN and VAE. The implicit model usually warps the shape, but can also carry the "disappear and appear" trick. In visual comparison, IM-GAN and WGAN$_{IM}$ output cleaner and more recognizable "in-between" digits. One can find missing or redundant parts in the samples produced by CNN based methods, which are vestiges of the "disappear and appear" phenomenon.

	Plane	Car	Chair	Rifle	Table
HSP	6,307	2,009	4,255	6,360	3,765
AtlasNet25	4,877	1,667	**3,244**	6,507	**2,725**
AtlasNetO	5,208	1,751	4,124	6,117	3,909
IM-SVR	**4,743**	**1,658**	3,321	**5,067**	2,918

Table 4: *Quantitative evaluation for SVR using LFD.* The means are taken over the first 100 shapes in the testing set for each category and rounded to integers. AtlasNet25 is AtlasNet with 25 patches ($28,900$ mesh vertices in total) and AtlasNetO ($7,446$ vertices) is with one sphere. IM-SVR and HSP were both reconstructed at 256^3 resolution.

4.5. Single-view 3D reconstruction (SVR)

We compare our approach with two state-of-the-art SVR methods, HSP [18], an octree-based method that uses 3D CNN decoders to generate 256^3 voxels, and AtlasNet [16], which warps surface patches onto target shapes. For AtlasNet, we tested two setups for the initial surfaces, 25 patches or a sphere, and denote them by AtlasNet25 and AtlasNetO, respectively. For all methods, we trained individual models for each category, and used grayscale images as inputs.

We used the train/test split in [18] to utilize the pre-trained models of HSP, since HSP requires quite a long time to converge. For HSP, we used the trained models made available online by the authors, and continued training for at most 2 days for each category and used the ones with the lowest testing loss. For AtlasNet, we trained the autoencoder part for 400 epochs and SVR part for 400 epochs

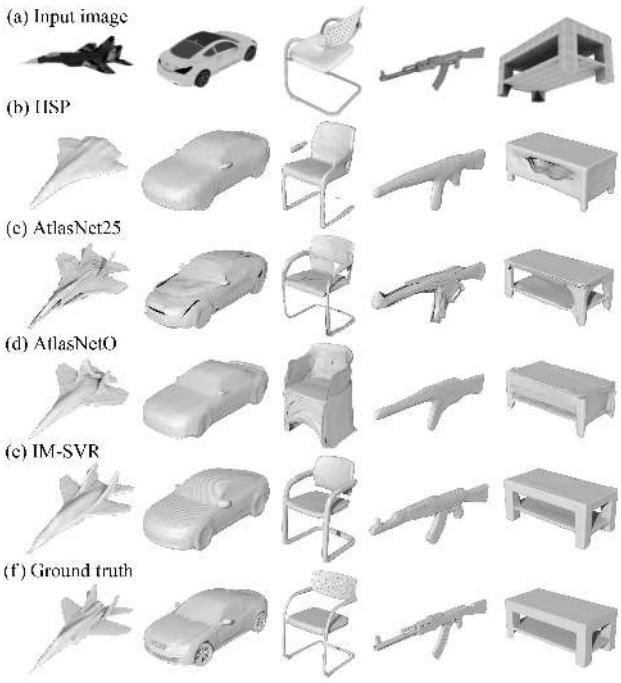

Figure 8: *Visual results for single-view 3D reconstruction.* See caption of Table 4 for output model settings.

and used the ones with the lowest testing loss. For our method, we trained IM-AE for 200-400 epochs on 64^3 resolution, and IM-SVR for 1,000-2,000 epochs. The number of epochs depends on the size of the dataset. We did not train AtlasNet with such number of epochs since its testing loss had stopped dropping. Since IM-SVR was trained to map an image into a latent code, we did not have a good evaluation metric for testing errors. Therefore, we tested the last five saved checkpoints and report the best results.

Quantitative and qualitative evaluations are shown in Table 4 and Figure 8, respectively. IM-SVR outputs were sampled at 256^3 resolution, same as HSP. Output mesh settings for AtlasNet are from the authors' code. Though IM-SVR seems to have similar quantitative results with AtlasNet25, please keep in mind that LFD only captures the silhouette of the shape. AtlasNet25 represents shapes well yet clear artifacts can be observed since the shapes are made of patches and there is no measure to prevent slits, foldovers or overlapped surfaces. AtlasNetO can generate cleaner shapes than AtlasNet25, but the topology is pre-assigned to be equivalent to a sphere, thus AtlasNetO can hardly reconstruct shapes with holes. HSP can produce smooth surfaces but failed to recover most details. We show the comparisons of the first 16 shapes in the testing set for each category in the supplementary material, and also provide the evaluation results by chamfer distance to further verify that CD may not be an ideal evaluation metric for visual quality.

5. Conclusion, limitation, and future work

We introduced a simple and generic implicit field decoder to learn shape boundaries. The new decoder IM-NET can be easily plugged into contemporary deep neural networks for a variety of applications including shape autoencoding, generation, interpolation, and single-view reconstruction. Extensive experiments demonstrate that IM-NET leads to cleaner closed meshes with superior visual quality and better handling of shape topology during interpolation.

A key merit of our implicit encoder is the inclusion of point coordinates as part of the input feature, but this comes as the cost of longer training time, since the decoder needs to be applied on each point in the training set. In practice, CNN-AE is typically 30 times faster than IM-AE on 64^3 data without progressive training. Even with progressive training, IM-AE training took about a day or two and CNN-AE is still 15 times faster. When retrieving generated shapes, CNN only needs one shot to obtain the voxel model, while our method needs to pass every point in the voxel grid to the network to obtain its value, therefore the time required to generate a sample depends on the sampling resolution. While AtlasNet also employed MLP as decoder, AtlasNet25 is 5 times faster than ours in training, since AtlasNet only needs to generate points on the surface of a shape yet ours need to generate points in the whole field.

Our implicit decoder does lead to cleaner surface boundaries, allowing both part movement and topology changes during interpolation. However, we do not yet know how to regulate such topological evolutions to ensure a meaningful morph between highly dissimilar shapes, e.g., those from different categories. We reiterate that currently, our network is only trained per shape category; we leave multi-category generalization for future work. At last, while our method is able to generate shapes with greater visual quality than existing alternatives, it does appear to introduce more low-frequency errors (e.g., global thinning/thickening).

In future work, we also plan to generalize IM-NET. First, using MLPs to decode may be too simple and inefficient; it is possible to improve the decoder structure to make the model size smaller for faster inference. Second, besides inside/outside signs, it is possible to train our decoder to output other attributes, e.g., color, texture, surface normal, signed distance, or deformation fields, for new applications. Finally, IM-NET has shown signs of understanding shape parts, suggesting its potential utility for learning part segmentation and correspondence.

Acknowledgment. We thank Matt Fisher, Daniel Cohen-Or, and the anonymous reviewers for their comments, and Kangxue Yin and Ali Mahdavi-Amiri for proofreading. The research is supported by NSERC and an Adobe gift fund.

References

[1] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. J. Guibas. Learning representations and generative models for 3d point clouds. In *International Conference on Machine Learning (ICML)*, 2018. 1, 2, 4, 5, 6

[2] M. Arjovsky and L. Bottou. Towards principled methods for training generative adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2017. 4

[3] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning (ICML)*, pages 214–223, 2017. 2, 4, 6

[4] Y. Bengio, E. Laufer, G. Alain, and J. Yosinski. Deep generative stochastic networks trainable by backprop. In *International Conference on Machine Learning (ICML)*, pages 226–234, 2014. 7

[5] Y. Bengio, G. Mesnil, Y. Dauphin, and S. Rifai. Better mixing via deep representations. In *International Conference on Machine Learning (ICML)*, pages 552–560, 2013. 7

[6] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 5(4):349–359, 1999. 6

[7] A. X. Chang, T. Funkhouser, L. 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. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 4

[8] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung. On visual similarity based 3d model retrieval. In *Computer graphics forum*, volume 22, pages 223–232. Wiley Online Library, 2003. 4

[9] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. 2, 3

[10] M. Corsini, P. Cignoni, and R. Scopigno. Efficient and flexible sampling with blue noise properties of triangular meshes. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 18(6):914–924, 2012. 4

[11] C. Dong, C. C. Loy, K. He, and X. Tang. Learning a deep convolutional network for image super-resolution. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2014. 2

[12] H. Fan, H. Su, and L. J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, page 6, 2017. 2, 3

[13] M. Gadelha, R. Wang, and S. Maji. Multiresolution tree networks for 3d point cloud processing. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 2, 3

[14] R. Girdhar, D. F. Fouhey, M. Rodriguez, and A. Gupta. Learning a predictable and generative vector representation for objects. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 484–499. Springer, 2016. 2, 3

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2672–2680, 2014. 2, 7

[16] T. Groueix, M. Fisher, V. G. Kim, B. Russell, and M. Aubry. Atlasnet: A papier-mâché approach to learning 3d surface generation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2, 3, 4, 7

[17] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5767–5777, 2017. 4, 6, 7

[18] C. Häne, S. Tulsiani, and J. Malik. Hierarchical surface prediction for 3d object reconstruction. In *Proceedings of the International Conference on 3D Vision (3DV)*. 2017. 1, 2, 3, 4, 7

[19] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 4

[20] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, Mar. 1991. 3

[21] Z. Huang, T. Li, W. Chen, Y. Zhao, J. Xing, C. LeGendre, L. Luo, C. Ma, and H. Li. Deep volumetric video from very sparse multi-view performance capture. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 3

[22] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *International Conference on Learning Representations (ICLR)*, 2018. 2

[23] D. P. Kingma and P. Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *arXiv preprint arXiv:1807.03039*, 2018. 2

[24] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *International Conference on Learning Representations (ICLR)*, 2014. 2, 6, 7

[25] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[26] C. Li, H. Liu, C. Chen, Y. Pu, L. Chen, R. Henao, and L. Carin. Alice: Towards understanding adversarial learning for joint distribution matching. *Advances in Neural Information Processing Systems (NIPS)*, 2017. 7

[27] J. Li, K. Xu, S. Chaudhuri, E. Yumer, H. Zhang, and L. Guibas. Grass: Generative recursive autoencoders for shape structures. *ACM Transactions on Graphics (TOG)*, 36(4):52, 2017. 2

[28] Y. Liao, S. Donné, and A. Geiger. Deep marching cubes: Learning explicit surface representations. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 3

[29] C.-H. 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

[30] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH Computer Graphics*, 21(4):163–169, Aug. 1987. 3

[31] D. C.-O. Olga Sorkine and S. Toledo. High-pass quantization for mesh encoding. In *Eurographics Sym. on Geometry Processing*, 2015. 4

[32] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2

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

[34] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *International Conference on Learning Representations (ICLR)*, 2016. 2, 6, 7

[35] G. Riegler, A. O. Ulusoy, H. Bischof, and A. Geiger. Octnetfusion: Learning depth fusion from data. In *Proceedings of the International Conference on 3D Vision (3DV)*, pages 57–66. IEEE, 2017. 2

[36] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. *Neural Networks*, 20(1):61–80, 2009. 3

[37] A. Sinha, J. Bai, and K. Ramani. Deep learning 3d shape surfaces using geometry images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 223–240. Springer, 2016. 2

[38] A. Sinha, A. Unmesh, Q. Huang, and K. Ramani. Surfnet: Generating 3d shape surfaces using deep residual networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 791–800, 2017. 2, 3

[39] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015. 1, 2

[40] M. Tatarchenko, A. Dosovitskiy, and T. Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, volume 2, page 8, 2017. 2

[41] A. van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, et al. Conditional image generation with pixelcnn decoders. In *Advances in Neural Information Processing Systems (NIPS)*, pages 4790–4798, 2016. 2

[42] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 2, 3

[43] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong. O-CNN: Octree-based Convolutional Neural Networks for 3D Shape Analysis. *ACM Transactions on Graphics (SIGGRAPH)*, 36(4), 2017. 2

[44] P.-S. Wang, C.-Y. Sun, Y. Liu, and X. Tong. Adaptive O-CNN: A Patch-based Deep Representation of 3D Shapes. *ACM Transactions on Graphics (SIGGRAPH Asia)*, 37(6), 2018. 2, 3

[45] 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)*, pages 82–90, 2016. 1, 2, 5, 6

[46] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 2, 3

[47] Y. Yang, C. Feng, Y. Shen, and D. Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 3, 2018. 2

[48] K. Yin, H. Huang, D. Cohen-Or, and H. Zhang. P2P-NET: Bidirectional point displacement net for shape transform. *ACM Trans. on Graph.*, 37(4):Article 152, 2018. 2

[49] C. Zhu, K. Xu, S. Chaudhuri, R. Yi, and H. Zhang. SCORES: Shape composition with recursive substructure priors. *ACM Trans. on Graph.*, 37(6), 2018. 2