2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Spectral-GANs for High-Resolution 3D Point-cloud Generation

Sameera Ramasinghe[1,2], Salman Khan[1], Nick Barnes[1] and Stephen Gould[1]

***Abstract*— Point-clouds are a popular choice for robotics and computer vision tasks due to their accurate shape description and direct acquisition from range-scanners. This demands the ability to synthesize and reconstruct high-quality point-clouds. Current deep generative models for 3D data generally work on simplified representations (e.g., voxelized objects) and cannot deal with the inherent redundancy and irregularity in point-clouds. A few recent efforts on 3D point-cloud generation offer limited resolution and their complexity grows with the increase in output resolution. In this paper, we develop a principled approach to synthesize 3D point-clouds using a spectral-domain Generative Adversarial Network (GAN). Our spectral representation is highly structured and allows us to disentangle various frequency bands such that the learning task is simplified for a GAN model. As compared to spatial-domain generative approaches, our formulation allows us to generate high-resolution point-clouds with minimal computational overhead. Furthermore, we propose a fully differentiable block to transform from the spectral to the spatial domain and back, thereby allowing us to integrate knowledge from well-established spatial models. We demonstrate that Spectral-GAN performs well for point-cloud generation task. Additionally, it can learn a highly discriminative representation in an unsupervised fashion and can be used to accurately reconstruct 3D objects. Our codes are available at https://github.com/samgregoost/Spectral-GAN/.**

I. INTRODUCTION

Point-clouds are a popular 3D representation for real-world scenes and have attracted great interest in robotic vision [28, 30, 10, 21, 18, 23]. Particularly, efficient synthesis of 3D data is critical in cases where labeled real data is limited. In comparison to other representations such as voxels, mesh and truncated signed distance function (TSDF), point-clouds are often an attractive choice for 3D data because they capture shape details accurately, are computationally efficient to process and can be acquired as a default output from several 3D sensors (e.g., LiDAR). However, point-clouds pose a major challenge for deep networks, particularly the generative pipelines, due to their inherent redundancy and irregular nature (e.g., permutation-invariance).

Due to the complexity of point-clouds, most 3D synthesis approaches are inapplicable. For example, generative approaches designed for voxelized inputs [31, 12, 32, 34, 11], cannot work with the irregular point sets. To overcome this challenge, some recent generative approaches solely focus on point-cloud synthesis. For example, Achlioptas *et al.*[1] use a GAN framework for 3D point-cloud distribution modelling in the data and auto-encoder latent space, Yang *et al.*[35] sample 3D points from a prior spatial distribution and then transform them using an invertible parameterization while [24, 29] employ graph-structured networks for point-cloud generation. Further, [5] directly operates on meshes and [4] proposes an implicit field decoder to generate the 3D objects. Moreover, [3] use a differentiable rendering framework from 3D to 2D for 3D object generation.

All such efforts so far, operate in the 'spatial-domain' (3D Euclidean space) which makes the modelling task relatively difficult due to arbitrary point configurations in 3D space. This leads to a number of roadblocks towards a versatile generative model e.g., considering a fixed set of points [1] and limited scalability to arbitrary point resolutions [24, 29]. As opposed to previous works, we perform generative modelling in the spectral space using spherical harmonic moment vectors (SMVs), which inherently offers a solution to the above mentioned problems. Specifically, generating 3D shapes via spectral representations allows us to compactly represent redundant information in point-clouds, easily scale to high-dimensional point-cloud sets, remain invariant to the permutations in unordered point sets and generate high-fidelity shapes with relatively minimal outliers. Besides, our spectral representation allow us to develop an understanding about the frequency domain functional space of generic 3D objects. Our main contributions are:

- To handle the redundancy and irregularity of point-clouds, we propose the first spectral-domain GAN that synthesizes novel 3D shapes by using a spherical harmonics based representation.
- A fully differentiable transformation from the spectral to the spatial domain and back, thus allowing us to integrate knowledge from well-established spatial models.
- Through both quantitative and qualitative evaluations, we illustrate that Spectral-GAN can generate high-quality 3D shapes with minimal artifacts and can be easily scaled to high-dimensional outputs.
- Our proposed framework learns discriminative unsupervised features and can seamlessly perform 3D reconstruction from 2D inputs. Moreover, we show that Spectral-GAN is scalable to high-resolution outputs ($40\times$ resolution increase with just $4\times$ parameters).

II. RELATED WORK

Generative models in spectral-domain: Yang *et al.*[36] and Souza *et al.*[27] develop methods for MRI reconstruction using GANs, and use Fourier domain information to refine the output. In both these approaches, networks operate on both spatial and spectral domains and exchange information. A significant drawback of these approaches is that output resolution is tightly coupled to the network design and thus, they

[1] College of Engineering and Computer Science (CECS), Australian National University (ANU), Canberra ACT 0200, AU. email: sameera.ramasinghe@anu.edu.au
[2] Data61, CSIRO, Canberra ACT 2601, AU.

lack scalability to high dimensions. In a different application, Portilla *et al.* [17] present a method to synthesize textures as 2D images based on a complex wavelet transform. They parameterize this operation using a set of statistics computed on pairs of coefficients corresponding to basis functions at adjacent spatial locations, orientations, and scales. However, their approach is not a learning model, which offers less flexibility. Furthermore, Zhu *et al*. [39] recently proposed a model that initially processes undersampled input data in the frequency domain and then refines the result in the spatial domain using the inverse Fourier transform. They approximate the inverse Fourier transform using a sequence of connected layers, but one disadvantage is that their transformation has quadratic complexity with respect to the size of the input image. Furthermore, the above works are limited to 2D and do not study the 3D point-cloud generation problem in spectral domain.

3D GANs in spatial-domain: 3D GANs can be primarily categorized into two types: voxel outputs and point-cloud outputs. The latter typically entails more challenges as point-clouds are unordered and highly irregular in nature.

For voxelized 3D object modeling, several influential methods have been proposed in the literature. Wu *et al.* [31] extend the 2D GAN framework to 3D domain for the first time. Following their work, Smith *et al.* [26] use a novel GAN architecture for 3D shape generation by employing Wasserstein distance as the loss function. A recent work by Khan *et al.* [11] presents a factorized 3D generative model that sequentially generates shapes in a coarse-to-fine manner. Our approach also uses a two-step procedure–a forward pass and backward pass—to refine a coarse 3D shape, but a key difference here is that they use spatial information to refine the shape, while our method depends on frequency information.

Naive extensions of traditional spatial GANs to 3D point-cloud generation do not produce satisfactory results, due to their inherent properties such as being an unordered, irregularly distributed collection (see Sec. III). Achlioptas *et al*.[1] were the first to use GANs to generate point-clouds. They first convert a point-cloud to a compact latent representation and then train a discriminator on it. Matsuzaki *et al*.[16] also use a latent representation and additionally employ parallel sub-decoders that can reconstruct the local regions of the input point-cloud more accurately. Although we also use a compact representation, i.e., the SMV to train the GAN, SMVs provide a richer representation compared to latent space approximations and theoretically guarantee accurate reconstruction of the 3D point-cloud. Moreover, Valsesia *et al*.[29] propose a graph convolution based network to extract localized features from 3D point-clouds, in order to reduce the effect of irregularity. A drawback of their method, however, is the rather high computational complexity of graph convolution, and less scalability with the resolution of the point-cloud. A recent work by Shu *et al*.[24] also propose a tree-structured graph convolution network, which is more computationally efficient. The model proposed by Li *et al*.[14] attempts to handle the irregularity of point-clouds using a separate inference model which captures a latent distribution, to deal with the irregularity of point-clouds. In contrast, we effectively reduce the problem to the standard GAN setting by using a fixed-dimensional representation for point-clouds.

III. Motivation

An *exchangeable* sequence is a sequence of random variables $\tilde{X} = \{x_i\}_{i=1}^n$, where the joint probability distribution of $\tilde{X}$ does not vary under position permutations. Formally,

Definition: *For a given finite set $\{x_i\}_{i=1}^n$ of random variables, let $\mu_{x_1,x_2,\ldots,x_n}$ be their joint distribution. This finite set is exchangeable if $\mu_{x_1,x_2,\ldots,x_n} = \mu_{x_{\pi(1)},x_{\pi(2)},\ldots,x_{\pi(n)}}$, for every permutation $\pi : \{1, 2, \ldots, n\} \rightarrow \{1, 2, \ldots, n\}$.*

The spatial representation X of a point-cloud corresponding to an object, is a *set* of d-dimensional vectors e.g., $d = 3$ in case of Euclidean geometry. A *set* is a collection of elements without any particular order and thus, $p(X)$ is fixed irrespective of the order of X. Therefore, X an exchangeable sequence. We argue that GANs, in their conventional form, are not well suited for modeling exchangeable sequences, since the discriminator learns to distinguish between fake and real distributions by observation. For example, let $\pi_m(X)$ be an arbitrary permutation m on the set $X = \{x_i\}_{i=1}^n$. Then, each element of the set $\phi = \{\pi_m(X)\}_{m=1}^M$ pretends to be a unique data instance to the GAN, although the elements of ϕ represent the same data instance. In other words, the GAN cannot learn $p(X)$ by just observing the marginal distributions of X. In contrast, GANs perform well on 2D image data, since they are an ordered representation on a 2D plane and hence not exchangeable. A possible solution is to learn a latent variable which determines the order of X while modeling the original data distribution. However, this can hinder the performance of the generator by putting an extra overhead on it. Another seemingly straightforward approach to resolve this problem is to model point-cloud data as ordered, fixed-dimensional vectors. However, this approach does not hold the integral probability metric (IPM) guarantees of a GAN [14].

On the contrary, we propose to model point-cloud data as SMVs, which effectively reduces the problem to the traditional case in two ways: 1) SMVs encode the corresponding shape information in a structured, fixed dimensional vector and 2) the vector elements are highly correlated with each other. The task of learning the distribution of elements of SMVs is theoretically similar to learning the pixel distribution of images, as in the latter case also, we only need to capture the joint probability distribution of pixels of each instance. In the case of image synthesis, however, GANs exploit the correlation of pixels using convolution kernels, which is not possible in the case of SMVs as correlation does not depend on proximity. Furthermore, different frequency portions of the SMVs show different characteristics. To handle these specific attributes, we propose a series of cascaded GANs, each consisting of only fully connected layers. Since each GAN only needs to generate a specific portion of the

SMV, they can be designed as shallow models with fewer floating point operations (FLOPs).

IV. Spectral GAN

We propose a 3D generative model that operates entirely in the spectral domain. Such a design offers unique advantages over spatial domain 3D generative models: (a) a compact representation of 3D shapes with an intuitive frequency-domain interpretation, (b) the flexibility to generate high-dimensional shapes with minimal changes to the model complexity, and (c) a permutation invariant representation which handles the irregularity of point-clouds. Below, we first introduce the spherical harmonics representations that serve as the basis for our proposed Spectral GAN model.

A. Spherical Harmonics for 3D Objects

Spherical harmonics are a set of complete and orthogonal basis functions, which can efficiently represent functions on the unit sphere $\mathbb{S}^2$ in $\mathbb{R}^3$. They are a higher dimensional analogy of the Fourier series, which forms a basis for functions on unit circle. The spherical harmonics are defined on $\mathbb{S}^2$ as,

$$Y_l^m(\theta,\phi) = N_l^m P_l^m(\cos\phi) e^{im\theta}, \tag{1}$$

where $\phi \in [0,\pi]$ is the polar angle, $\theta \in [0,2\pi]$ is the azimuth angle, $l \in \mathbb{Z}^+$ is a non-negative integer, $m \in \mathbb{Z}$ is an integer, $|m| < l$, $i = \sqrt{-1}$ is the imaginary unit, $N_l^m = (-1)^m \sqrt{\frac{2l+1}{4\pi}\frac{(l-m)!}{(l+m)!}}$ is the normalization coefficient and $P_l^m(x) = (-1)^m \frac{(1-x^2)^{\frac{m}{2}}}{2^l l!}\frac{d^{l+m}}{dx^{l+m}}(x^2-1)^l$ is the associated Legendre function. Since spherical harmonics are orthogonal and complete over the continuous functions on $\mathbb{S}^2$ with finite energy, such a function $f : \mathbb{S}^2 \to \mathbb{R}$ can be expanded as,

$$f(\theta,\phi) = \sum_{l=0}^{\infty}\sum_{m=-l}^{l} c_l^m Y_l^m(\theta,\phi), \tag{2}$$

where $c_l^m = \int_0^{\pi}\int_0^{2\pi} f(\theta,\phi) Y_l^m(\theta,\phi)^{\dagger} \sin\phi \, d\phi d\theta$. The sufficient conditions for the expansion in Eq. 2 are given in [8]. In practical cases, a bounded set of spherical harmonic basis functions $(M+1)^2$ is defined, where M is the maximum degree of harmonics series.

The process of 3D shape modeling via spherical harmonics can be decomposed into two major steps. First, sample points from the 3D shape surface and then computing spherical harmonic moments. Any polar 3D surface function can be represented as $r = f(\theta,\phi)$, where $f(\theta,\phi)$ is a single valued function on the unit sphere $\mathbb{S}^2$, r is the radial coordinate with respect to a predefined origin inside an object, and (θ,ϕ) is the direction vector. Thus, we can compute moments c_l^m of the corresponding 3D point-cloud. One seemingly possible drawback in using spherical harmonics to model 3D objects can be the smoothing effect that can occur when projecting the object function to the unit sphere ($\mathbb{S}^2$). However, we reduce this effect by casting two sets of rays, in slightly different angle settings, and obtaining the last and first ray hit locations as object points, respectively.

B. Cascaded GAN Structure

SMVs provide a highly structured representation of 3D objects, as explained in Sec. IV-A. Due to this structured nature, the margin for error is significantly lower in our setup, compared to GANs that try to produce spatial domain representations. Also, different frequency bands of the SMV typically entail different characteristics, which makes it highly challenging for a single GAN to generalize over the complete SMV. Therefore, to overcome this obstacle, we use multiple cascaded GANs, where each GAN specializes in generating a pre-defined frequency band of the SMV.

Our approach uses a combination of T GAN models to generate the SMV of 3D shapes. Among them, the first model is a regular GAN while the remaining $T-1$ models are conditional GANs (cGAN). The objective of initial GAN model is given by a two-player min-max game,

$$\min_{\mathcal{G}_1}\max_{\mathcal{D}_1} L_{GAN}(\mathcal{G}_1,\mathcal{D}_1) = \mathbb{E}_{\bar{\boldsymbol{g}}_1}[\log \mathcal{D}(\bar{\boldsymbol{g}}_1)] + \mathbb{E}_{z_1}[\log(1-\mathcal{D}(\mathcal{G}(z_1)))], \tag{3}$$

where $\bar{\boldsymbol{g}}_i \sim p_{\boldsymbol{g}}$ is the SMV band sampled from the spectral coefficient distribution and $z \sim p_z$ is the noise vector sampled from a Gaussian distribution. In a cGAN, synthetic data modes are controlled by forwarding conditioning variables (e.g., a class label) as additional information to the generator. In our case, we use a specific band of SMVs $\boldsymbol{g}_i$ predicted by the previous generator to condition the subsequent generator. Then, the cGAN objective becomes,

$$\min_{\mathcal{G}_i}\max_{\mathcal{D}_i} L_{cGAN}(\mathcal{G}_i,\mathcal{D}_i) = \mathbb{E}_{\bar{\boldsymbol{g}}_i}[\log \mathcal{D}(\bar{\boldsymbol{g}}_i)] + \mathbb{E}_{\boldsymbol{g}_{i-1},z_i}[\log(1-\mathcal{D}(\mathcal{G}_i(\boldsymbol{g}_{i-1},z_i)))] : i > 1. \tag{4}$$

Each GAN generates a portion of the complete spherical moment vector for the next GAN to be conditioned upon. The setup includes two major steps: (i) forward pass and (ii) backward pass. Accordingly, the overall architecture can be decomposed into two sets of generators $\mathcal{G}_f$ and $\mathcal{G}_b$, that implement the forward and backward functions, respectively. In the forward pass, the model tries to generate a coarse shape representation, and the backward pass refines the coarse representation to generate a refined representation. The basis of our design is the *Markovian assumption*, i.e., given the outputs from the neighbouring generators, a current generator is independent from the outputs of the rest. We describe the two generation steps in Sec. IV-B.1 and IV-B.2.

1) Forward pass: In the forward pass, we have a set of T' generative models $\mathcal{G}_f = \{\mathcal{G}_1,\ldots,\mathcal{G}_{T'}\}$, which work in unison to generate a coarse representation of a 3D shape. Each $\mathcal{G}_i \in \{\mathcal{G}_2,\ldots\mathcal{G}_{T'}\}$ is conditioned upon the outputs of $\mathcal{G}_{i-1}$, and generates a predefined frequency band ($\mathcal{S}_i$) of the complete spherical harmonic representation ($\mathcal{S}$) of the corresponding 3D shape. It is worthwhile to note that the forward pass is sufficient to generate the complete SMV without the aid of a backward pass. However, a critical limitation of this setup is that each GAN is only conditioned upon lower frequency bands of the SMV. In practice, this results in noisy outputs. Therefore, we also perform a backward pass, which

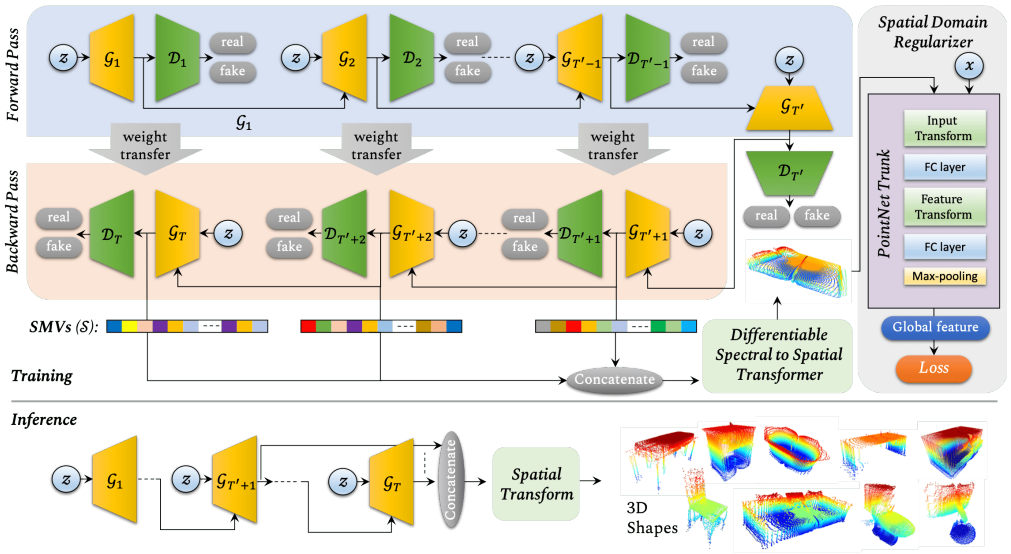


Fig. 1: Overview of the Spectral Generative Adversarial Network. An unrolled version (with an explicit forward and backward pass) of the training scheme is shown for clarity.

allows the GANs to refine the generation by observing the higher frequencies. This procedure is explained on Sec. IV-B.2.

2) Backward pass: As explained in Sec. IV-B.1, the aim of the backward pass is to generate a more refined SMV, which produces a more refined 3D shape. Similar to forward pass, the backward pass is implemented using another set of generators $\mathcal{G}_b = \{\mathcal{G}_{T'+1}, \dots \mathcal{G}_T\}$, where $T = 2T' - 1$. Each $\mathcal{G}_i \in \mathcal{G}_b$ is conditioned upon the outputs of $\mathcal{G}_{i-1}$ and generates a specific portion of the complete SMV. In the training phase, we first transfer the trained weights from $\{\mathcal{G}_f \backslash \mathcal{G}_{T'}\}$ to $\mathcal{G}_b$, before training $\{\mathcal{G}_b\}$. Therefore, this can be intuitively considered as fine-tuning $\{\mathcal{G}_1 \dots \mathcal{G}_{T'-1}\}$ based on higher frequencies. The training procedure is explained in Sec. VI.

V. Spatial Domain Regularizer

Since SMVs are highly structured, each element of a particular SMV is crucial for accurate reconstruction of its corresponding 3D point-cloud. In other words, even slight variations of a particular SMV cause significant variations in the spatial domain. Therefore, it is cumbersome for a GAN to synthesize SMVs, corresponding to visually pleasing point-clouds, by solely observing a distribution of ground truth SMVs.

To surmount this barrier, we use a spatial domain regularizer that can refine the weights of our cascaded GAN architecture, in order to synthesize more plausible SMVs. The spatial domain regularizer provides feedback from the spatial domain to the GANs, depending on the quality of the spatial reconstruction. Firstly, we employ a pre-trained PointNet [19] model on the reconstructed synthetic point-cloud, and extract a global feature. Secondly, using the same procedure, we obtain another global feature from a ground truth point-cloud from the same class, and compute the L_2 distance between these two features. Finally, using back back-propagation, we update the weights of all the generators $\boldsymbol{G} = \{\mathcal{G}_f \cup \mathcal{G}_b\}$ to minimize the L_2 distance. The final architecture of the proposed model is shown in Fig. 1.

In order to back-propagate error signals from the spatial domain to the spectral domain, we require $\partial\mathcal{L}/\partial\boldsymbol{g}$, where $\boldsymbol{g}$ is the SMV and $\mathcal{L}$ is the loss. To this end, we derive the following formula: let $\boldsymbol{g} = (g_0^0, \dots, g_l^m, \dots g_K^K)^\top$ be the SMV of a particular instance and $\{r(\theta_0, \phi_0), \dots, r(\theta_n, \phi_n), \dots, r(\theta_N, \phi_N)\}$ be the corresponding reconstructed points on $\mathbb{S}^2$ for the same instance. Then, using the chain rule it can be shown that,

$$\frac{\partial \mathcal{L}}{\partial g_l^m} = \sum_\theta \sum_\phi \frac{\partial \mathcal{L}}{\partial r(\theta,\phi)} \frac{\partial r(\theta,\phi)}{\partial g_l^m}, \tag{5}$$

$$\text{where, } r(\theta,\phi) = \sum_{l=0}^{M} \sum_{m=-l}^{l} g_l^m Y_l^m(\theta,\phi). \tag{6}$$

Combining Eq. 5 and 6, we obtain,

$$\frac{\partial \mathcal{L}}{\partial g_l^m} = \sum_\theta \sum_\phi \frac{\partial \mathcal{L}}{\partial r(\theta,\phi)} Y_l^m(\theta,\phi). \tag{7}$$

The above expression can be written as a matrix-vector product to obtain derivatives $\partial\mathcal{L}/\partial\boldsymbol{g}$. This makes the transformer a fully differentiable and a network-agnostic module which can be used to communicate between spectral and spatial domains.

VI. Network Architecture and Training

Our aim is to generate a compact spectral representation, i.e., a vector, instead of a irregular point set. In the spatial domain, points are correlated across the spatial space, and convolutions can be adopted to capture these dependencies. In fact, convolution kernels extract local features, under the assumption that spatially closer data points form useful local features. In contrast, closer elements in spectral domain

Algorithm 1: Training procedure for the Spectral-GAN.

$\boldsymbol{G} = \{\mathcal{G}_f \cup \mathcal{G}_b\}$;
R_o = A set of samples from ground truth point-clouds;
for *i iterations* **do**
  for *each* $\mathcal{G}_k \in \mathcal{G}_f$ **do**
    for *j iterations* **do**
      Train $\mathcal{G}_k$;
  $\mathcal{G}_b \xleftarrow{\text{Weights}} \{\mathcal{G}_1, \ldots \mathcal{G}_{T'-1}\}$;
  for *each* $\mathcal{G}_k \in \mathcal{G}_b$ **do**
    for *j iterations* **do**
      Train $\mathcal{G}_k$;
  for *p iterations* **do**
    $\boldsymbol{g} \xleftarrow{\text{SYNTHESIZE}} \{\mathcal{G}_{T'} \cup \mathcal{G}_b\}$;
    $\boldsymbol{r}_g \leftarrow \text{RECONSTRUCT}(\boldsymbol{g})$;
    $f_g \leftarrow \text{POINTNET}(r_g)$;
    $f_o \leftarrow \text{POINTNET}(r_o \sim R_o)$;
    $L \leftarrow \|f_g - f_o\|_2$;
    $\boldsymbol{G} \leftarrow \text{UPDATE}(\boldsymbol{G}, L)$;

representations do not necessarily exhibit strong correlations. Therefore, convolutional layers fail to excel in this scenario and thus, we opt for fully connected (FC) layers in designing our GANs. Interestingly, however, our GANs learn to generate quality outputs with a low depth architecture.

Generator architecture: For our main experiments, we choose the maximum degree of SMVs and the number of GANs as M=100 and T=7, respectively, where $\mathcal{G}_f = \{\mathcal{G}_1, .., \mathcal{G}_4\}$ and $\mathcal{G}_b = \{\mathcal{G}_5, \mathcal{G}_6, \mathcal{G}_7\}$. We observed that the output quality decreases when T is too low, as the generator has to predict a wider band. Similarly, if T is too high, the propagated error becomes larger. Therefore, we found T=7 to be a empirically good number. Each generator in $\mathcal{G}_f$ respectively generates frequency bands $(0 \leq l \leq 50, -l \leq m \leq 0)$, $(0 \leq l \leq 50, 0 < m \leq l)$, $(50 < l \leq 100, -l \leq m \leq 0)$ and $(50 < l \leq 100, 0 < m \leq l)$. Since $\mathcal{G}_5, \mathcal{G}_6, \mathcal{G}_7$ are used to fine tune $\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_3$, they generate the same frequency portions as the latter set. For all the generators, we use the same architecture, except for the last FC layer. Each generator consists of three FC layers, first two layers with 512 neurons each, and the number of neurons in the last layer depends on the output size. For the first two layers, we use ReLU activation and the final layer has a $tanh$ activation.

Training: The input to each of our generators, except to $\mathcal{G}_1$, is a 300-d vector: a 200-d noise vector concatenated with a 100-d vector sampled in equal intervals from the previous generator output. For $\mathcal{G}_1$, we use a 200-d noise input. We use RMSprop as the optimization algorithm with ρ=0.9, momentum=$0, \epsilon$=10^{-7}, where symbols refer to usual notation. Surprisingly, we observed that more commonly used optimizers such as Adam show inferior performance. For $\mathcal{G}_f$ and $\mathcal{G}_b$, we use learning rates 0.001 and 0.0001 respectively, and for discriminators, we use a learning rate 10^{-5}. While training, we use three discriminator updates per each generator update. Our training scheme is illustrated in Algorithm 1.

VII. 3D RECONSTRUCTION FROM SINGLE IMAGE

As a different application, we propose a generative model which can reconstruct 3D objects by observing a single RGB image. The model follows the network architecture proposed in Sec. VI, with a few alterations. Instead of randomly choosing the latent vector z, we use a set of image encoders to obtain an object representative vector $\hat{z}$, by taking a 2D image as the input. We use the same image encoder proposed in [32], which consists of five spatial convolution layers with kernel size $\{11, 5, 5, 5, 8\}$ with strides $\{4, 2, 2, 2, 1\}$. We use batch normalization after each layer, and ReLu activation as the non-linearity.

We use T' such image encoders for each $\mathcal{G}_i \in \mathcal{G}_f$, and use the same $\hat{z}$ vectors generated for $\{\mathcal{G}_1, \ldots, \mathcal{G}_{T'-1}\}$ when training $\mathcal{G}_i \in \mathcal{G}_b$. Each image encoder is trained end-to-end with $\mathcal{G}_i \in \mathcal{G}_f$. The training procedure is similar to Algorithm 1, although we use different loss functions in this case. To optimize the GANs in spectral domain, we use two loss components: an adversarial loss $\mathcal{L}_{ad}$ and a spectral reconstruction loss $\mathcal{L}_{sr}$. Thus, the final spectral domain loss is,

$$L_{spectral} = L_{ad} + \alpha L_{sr}, \tag{8}$$

where L_{sr} is the L_2 distance between the ground-truth SMV and the generated SMV from $\mathcal{G}'_T \cup \mathcal{G}_b$ and $\mathcal{L}_{ad} = \log \mathcal{D}(x) + \log(1 - \mathcal{D}(\mathcal{G}(\mathcal{E}(y))))$. Here, $\mathcal{E}(\cdot)$ is the encoder function, $\mathcal{D}(\cdot)$, $\mathcal{G}(\cdot)$ and y are discriminator function, generator function and image input, respectively. we choose $\alpha = 0.1$ empirically. For the spatial domain optimization, we replace spatial regularization loss with the Chamfer distance as follows:

$$L_{spatial} = \sum_{u \in S_1} \min_{v \in S_2} \|u - v\|_2^2 + \sum_{v \in S_2} \min_{u \in S_1} \|u - v\|_2^2, \tag{9}$$

where S_1 and S_2 are ground-truth and generated point sets, respectively. First, we obtain S_2 by converting the SMV to a point-cloud using Eq. 2 and then compute the loss (Eq. 9).

VIII. EXPERIMENTS

In this section, we evaluate our model both qualitatively and quantitatively, and develop useful insights.

A. 3D shape generation

Qualitative results: We train our model for each category in ModelNet10 and show samples of generated 3D point-clouds in Fig. 2. As expected, the reconstruction from SMV adds some noise to the ground truth point-clouds. An interesting observation, however, is that the quality of generated point-clouds are not far from from the reconstructed point-clouds from the ground-truth. Since the network only consumes the reconstructed ground-truth, this observation highlights the ability of our network in accurate modeling of input data distributions.

Quantitative analysis: To assess the proposed approach quantitatively, we compare the Inception Score (IS) of our network with other voxel-based generative models in Tab. II. In this experiment, we use [20] as the reference network. IS

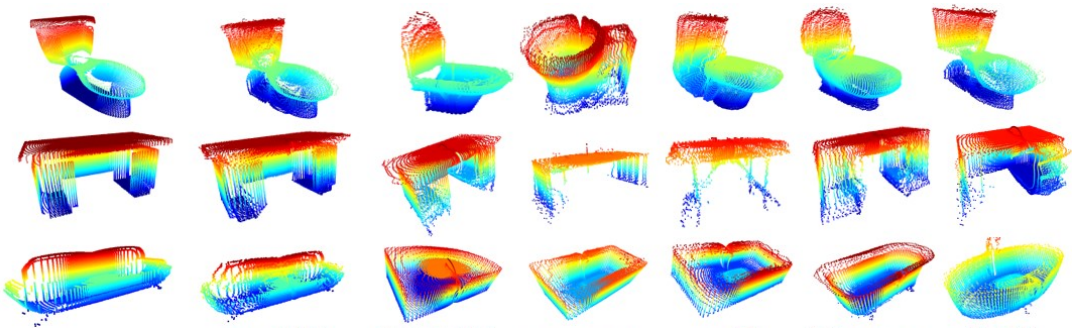


Fig. 2: Qualitative analysis of the results. From the left, 1^{st} column: Ground truth, 2^{nd} column: ground truth point-clouds reconstructed by SMV, $3^{rd} - 7^{th}$ columns: generated samples using spectral GAN.

TABLE I: 3D shape classification results on ModelNet10.

Method	Type	Accuracy
3D-ShapeNet (CVPR'15) [32]	Supervised	**93.5%**
EC-CNNs (CVPR'17) [25]	Supervised	90.0%
Kd-Network (ICCV'17) [13]	Supervised	**93.5%**
LightNet (3DOR'17) [38]	Supervised	93.4%
SO-Net (CVPR'18) [15]	Supervised	**95.5%**
Light Filed Descriptor [2]	Unsupervised	79.9%
Vconv-DAE (ECCV'16) [22]	Unsupervised	80.5%
3D-GAN (NIPS'16) [31]	Unsupervised	91.0%
3D-DesNet (CVPR'18) [34]	Unsupervised	92.4%
3D-WINN (AAAI'19) [9]	Unsupervised	91.9%
PrimtiveGAN (CVPR'19) [11]	Unsupervised	92.2%
Spectral-GAN (ours)	Unsupervised	93.1%

Method	3D Data	Accuracy
3D-ShapeNet [32] (CVPR'15)	voxel	4.13 $\pm$ 0.19
3D-VAE [12] (ICLR'15)	voxel	11.02 $\pm$ 0.42
3D-GAN [31] (NIPS'16)	voxel	8.66 $\pm$ 0.45
3D-DesNet [34] (CVPR'18)	voxel	**11.77 $\pm$ 0.42**
3D-WINN [9] (AAAI'19)	voxel	8.81 $\pm$ 0.18
PrimitiveGAN [11] (CVPR'19)	voxel	11.52 $\pm$ 0.33
Spectral-GAN (ours)	p-cloud	**11.58 $\pm$ 0.08**

II: Inception scores (IS) for 3D shape generation. We only compare with voxel based methods since no point-cloud (p-cloud) based method reports IS.

evaluates a model in terms of both quality and diversity of the generated shapes. Overall, our model demonstrates the second highest performance with a score of 11.58. To the best of our knowledge, our work is the first 3D point-cloud GAN to report IS.

We further evaluate our model using Frechet Inception Distance (FID) proposed by Heusel *et al.* [7], and compare with state-of-the-art. IS does not always coincide with human judgement regarding the quality of the generated shapes, as it does not directly capture the similarity between the synthetic and generated shapes. Therefore, FID is used as a complementary measure to evaluate GAN performance. Huang *et al.* [9] were the first to incorporate FID to 3D GANs, and following them, we also use [20] as the reference network. As evident from Table III, our results are on-par with state-of-the-art, getting highest scores in three

TABLE III: FID scores for 3D shape generation. (*lower is better*) All the methods except ours are voxel based.

Method	Dresser	Toilet	Stand	Chair	Table	Sofa	Monitor	Bed	Bathtub	Desk
3D-GAN [31] (NIPS'16)	-	-	-	**469**	-	517	-	-	-	651
3D-DesNet [34] (CVPR'18)	**414**	662	517	490	538	494	511	574	-	-
3D-WINN [9] (AAAI'19)	**305**	**474**	**456**	**225**	**220**	**151**	**181**	**222**	**305**	**322**
Spectral-GAN (ours)	462	**195**	**452**	472	**522**	**180**	**192**	**230**	**208**	**354**

TABLE IV: Comparison with point-cloud generative models. We randomly hand picked three classes to better illustrate the results.

Method	Class	MMD-CD	MMD-EMD
r-GAN (dense) [1]	Chair	0.0029	0.136
r-GAN (conv) [1]		0.0030	0.223
Valsesia *et al.* (no up.) [29]		0.0033	0.104
Valsesia *et al.* (up.) [29]		0.0029	0.097
TreeGAN [24]		**0.0016**	**0.101**
Spectral-GAN (ours)		**0.0012**	**0.080**
r-GAN (dense) [1]	Airplane	0.0009	0.094
r-GAN (conv) [1]		0.0008	0.101
Valsesia *et al.* (no up.) [29]		0.0010	0.102
Valsesia *et al.* (up.) [29]		0.0008	0.071
TreeGAN [24]		**0.0004**	**0.068**
Spectral-GAN (ours)		**0.0002**	**0.057**
r-GAN (dense) [1]	Sofa	**0.0020**	0.146
r-GAN (conv) [1]		0.0025	0.110
Valsesia *et al.* (no up.) [29]		**0.0024**	0.094
Valsesia *et al.* (up.) [29]		**0.0020**	**0.083**
Spectral-GAN (ours)		**0.0020**	**0.080**
r-GAN (dense) [1]	All classes	0.0021	0.155
TreeGAN [24]		**0.0018**	**0.107**
Spectral-GAN (w/o backward pass)		0.0020	0.127
Spectral-GAN (ours)		**0.0015**	**0.097**

categories: toilet, night stand and bath tub. Interestingly, our Spectral-GAN generally performs better with objects that have curved boundaries, which is a favorable characteristic, as curved boundaries are generally difficult to generate in Euclidean spaces. Note that we convert the point-clouds to meshes before evaluating with both IS and FID.

Comparison with point-cloud generation approaches: We use two metrics proposed in Achlioptas *et al.*[1] (i.e., MMD-CD, MMD-ED) to compare the performance of the proposed architecture with other point-cloud generation methods, and display the results in Table IV. In this experiment, we use 16 classes of ShapeNet [37]. As shown, our network gives best results. Intuitively, this suggests that shapes generated by our network have high fidelity compared to the test set.

In Table V we compare our method against the recently proposed work of Matsuzaki *et al.*[16] on the ModelNet10. In this comparison, we use MMD-EMD and Coverage (COV) as the evaluation metrics. As illustrated, our method achieves superior performance in terms of both COV and MMD-EMD.

Scalability to high resolutions: A favorable attribute of our network design is the ability to scale to higher resolutions with minimal changes to the architecture. To verify this, we vary the degree of SMV, and train our model separately for each case. Since the number of points n is tied to

TABLE V: Comparison with the state-of-the-art on ModelNet10. Values scaled by 10 for clarity.

Method		Dresser	Toilet	Stand	Chair	Table	Sofa	Monitor	Bed	Bathtub	Desk
Matsuzaki *et al.*[16]	MMD	0.514	0.587	0.692	**0.629**	0.560	**0.413**	0.514	0.482	0.510	0.768
Spectral-GAN (ours)		**0.420**	**0.314**	**0.518**	0.726	**0.524**	0.553	**0.396**	**0.466**	**0.492**	**0.553**
Matsuzaki *et al.*[16]	COV	0.475	**0.517**	0.435	0.472	0.507	0.451	0.447	0.487	0.464	0.450
Spectral-GAN (ours)		**0.570**	0.578	**0.539**	**0.558**	**0.574**	**0.568**	**0.541**	**0.559**	**0.482**	**0.528**

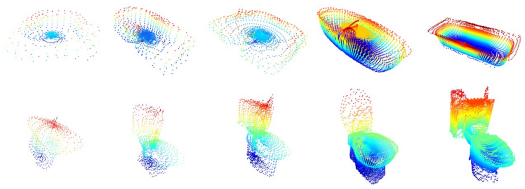


Fig. 3: Scalability of the proposed network with resolution. We obtain increasingly dense resolution by only changing the output layer size in each training phase. Number of points from the left: $30^2, 60^2, 100^2, 150^2$ and 200^2

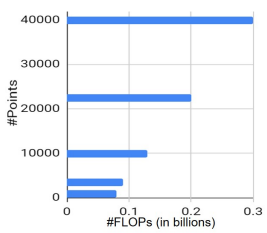
.

the maximum degree M of SMVs as $n{=}4M^2$, we obtain samples with different resolutions for each case (see Fig. 3). A key point here is that we only change the output layer size of the generator (according to the length of SMV) to generate point-clouds with different resolutions. Fig. 4 illustrates the variation of resolution with the number of FLOPs. Remarkably, we are able to generate high-resolution outputs up to $40,000$ points with only $0.3B$ FLOPs. Another intriguing observation is that our network is able to increase the output resolution by a factor of 40, while the number of FLOPs is only increased by a factor around 4.

Usefulness of backward pass: Fig. 5 illustrates the effect of performing a backward pass. As shown, the forward pass only generates a coarse representation of the shapes without fine details. This is anticipated, since cascaded GANs can only observe the lower frequency portions of SMV in the forward pass. In contrast, the backward pass observes the higher frequency portions, and fine tunes the coarse representation by adding complementary details.

B. Unsupervised 3D Representation Learning

In this section, we evaluate the representation learning capacity of our discriminator. To this end, we pass relevant SMV frequency bands of 3D point-clouds through trained discriminators, extract the features from the third FC layer, and finally concatenate them to create a feature vector. This feature vector is then fed through a binary SVM classifier and the classification results are obtained as one-against-the-rest. The classification results on ModelNet10 are depicted in Table I. As evident, we achieve the highest result with a value of 93.1%, which highlights the excellent representation learning capacity of our discriminators.



4: Spectral GAN can generate high-resolution outputs with minimal computational overhead. We increase resolution approximately by $40\times$ with only an increase of $4\times$ in the total number of FLOPs.

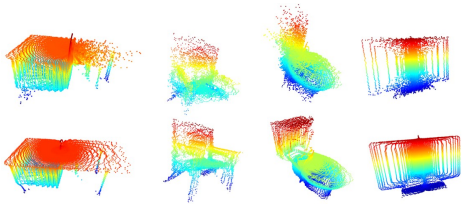


Fig. 5: Effect of backward pass. Top row: samples generated using only forward pass. Bottom row: same samples after passing through both forward and backward pass. Backward pass refines the image by adding more fine details.

Fig. 6: Qualitative results for 3D point-cloud reconstruction from a single image.

C. 3D reconstruction results

In this section, we evaluate the performance of the 3D reconstruction network proposed in Sec. VII. First, we randomly apply a rotation $R = (R_x, R_y, R_z)$ to each 3D model from the IKEA dataset 15 times, and render the rotated model in front of background images obtained from [33]. Afterwards, we save the rendered images and the corresponding 3D models to create ground-truth image-3D model pairs. The ground truth 3D-models are manually aligned using the Iterative closest point (ICP) algorithm. While applying rotations, we set the constraints $-\frac{\pi}{6} < R_x, R_y < \frac{\pi}{6}$ and $-\pi < R_z < \pi$ and crop the rendered images for the object to be in the center of the images. For the test set, we use the original images provided in the IKEA dataset and test our network on four object classes: chair, sofa, table and bed. We train our model separately for each category and use mean average precision (mAP) to evaluate the performance. In evaluation, we voxelize the generated and ground truth point-clouds using a $20{\times}20{\times}20$ voxel grid, and obtain average precision for voxel prediction. The results and illustrative examples are shown in Table VI and Fig. 6, respectively. As depicted, we surpass state-of-the-art results in sofa and bed categories, while achieving second best results in the table category.

Method	Chair	Sofa	Bed	Table
AlexNet-fc8 [6]	20.4	38.8	29.5	16.0
AlexNet-conv4 [6]	31.4	69.3	38.2	19.1
T-L network [6]	32.9	71.7	56.3	23.3
3D-VAE-GAN [31]	47.2	**78.8**	63.2	42.3
VAE-IWGAN [26]	**49.3**	68.0	65.7	52.2
PrimtiveGAN [11]	**47.5**	77.1	**68.4**	**60.0**
Spectral-GAN (ours)	42.3	**81.2**	**71.4**	**48.3**

TABLE VI: Average precision for 3D point-cloud reconstruction from single image. The point-clouds are voxelized before obtaining the score.

IX. Conclusion

We propose a generative model for 3D point-clouds that operates in the spectral-domain. In contrast to previous methods that operate in the spatial-domain, our approach provides a structured way to deal with the inherent redundancy and irregularity of point-clouds. We demonstrate that our model generates sound 3D outputs, can scale to high-dimensional outputs and learns discriminative features in an unsupervised manner. Further, it can be used for 3D reconstruction task.

References

[1] Panos Achlioptas et al. "Learning representations and generative models for 3d point clouds". In: *arXiv preprint arXiv:1707.02392* (2017).

[2] Ding-Yun Chen et al. "On visual similarity based 3D model retrieval". In: *Computer graphics forum*. Vol. 22. 3. Wiley Online Library. 2003, pp. 223–232.

[3] Wenzheng Chen et al. "Learning to predict 3d objects with an interpolation-based differentiable renderer". In: *Advances in Neural Information Processing Systems*. 2019, pp. 9609–9619.

[4] Zhiqin Chen and Hao Zhang. "Learning implicit fields for generative shape modeling". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 5939–5948.

[5] Shiyang Cheng et al. "Meshgan: Non-linear 3d morphable models of faces". In: *arXiv preprint arXiv:1903.10384* (2019).

[6] Rohit Girdhar et al. "Learning a predictable and generative vector representation for objects". In: *ECCV*. Springer. 2016, pp. 484–499.

[7] Martin Heusel et al. "Gans trained by a two time-scale update rule converge to a local nash equilibrium". In: *NeurIPS*. 2017, pp. 6626–6637.

[8] Ernest W Hobson. *The theory of spherical and ellipsoidal harmonics*. CUP Archive, 1931.

[9] Wenlong Huang et al. "3d volumetric modeling with introspective neural networks". In: *AAAI*. 2019.

[10] Robin Kerstens et al. "3D point cloud data acquisition using a synchronized in-air imaging sonar sensor network". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (2019, to be published)*. 2019.

[11] Salman H Khan et al. "Unsupervised Primitive Discovery for Improved 3D Generative Modeling". In: *CVPR*. 2019, pp. 9739–9748.

[12] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[13] Roman Klokov and Victor Lempitsky. "Escape from cells: Deep kd-networks for the recognition of 3d point cloud models". In: *ICCV*. 2017, pp. 863–872.

[14] Chun-Liang Li et al. "Point cloud gan". In: *arXiv preprint arXiv:1810.05795* (2018).

[15] Jiaxin Li, Ben M Chen, and Gim Hee Lee. "So-net: Self-organizing network for point cloud analysis". In: *CVPR*. 2018, pp. 9397–9406.

[16] Kohei Matsuzaki and Kazuyuki Tasaka. "Representation Learning via Parallel Subset Reconstruction for 3D Point Cloud Generation". In: *IROS*. IEEE. 2019, pp. 289–296.

[17] Javier Portilla and Eero P. Simoncelli. "A parametric texture model based on joint statistics of complex wavelet coefficients." In: *IJCV* (2000).

[18] Charles Ruizhongtai Qi et al. "Pointnet++: Deep hierarchical feature learning on point sets in a metric space". In: *NeurIPS*. 2017, pp. 5099–5108.

[19] Charles R Qi et al. "Pointnet: Deep learning on point sets for 3d classification and segmentation". In: *CVPR*. 2017, pp. 652–660.

[20] Charles R Qi et al. "Volumetric and multi-view cnns for object classification on 3d data". In: *CVPR*. 2016, pp. 5648–5656.

[21] Sameera Ramasinghe et al. "Representation Learning on Unit Ball with 3D Roto-translational Equivariance". In: *IJCV* (2019), pp. 1–23.

[22] Abhishek Sharma, Oliver Grau, and Mario Fritz. "Vconv-dae: Deep volumetric shape learning without object labels". In: *ECCV*. Springer. 2016, pp. 236–250.

[23] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. "Pointrcnn: 3d object proposal generation and detection from point cloud". In: *CVPR*. 2019, pp. 770–779.

[24] Dong Wook Shu, Sung Woo Park, and Junseok Kwon. "3D Point Cloud Generative Adversarial Network Based on Tree Structured Graph Convolutions". In: *arXiv preprint arXiv:1905.06292* (2019).

[25] Martin Simonovsky and Nikos Komodakis. "Dynamic edge-conditioned filters in convolutional neural networks on graphs". In: *CVPR*. 2017, pp. 3693–3702.

[26] Edward Smith and David Meger. "Improved adversarial systems for 3d object generation and reconstruction". In: *arXiv preprint arXiv:1707.09557* (2017).

[27] Roberto Souza and Richard Frayne. "A hybrid frequency-domain/image-domain deep network for magnetic resonance image reconstruction". In: *arXiv preprint arXiv:1810.12473* (2018).

[28] Bo Sun and Philippos Mordohai. "Oriented Point Sampling for Plane Detection in Unorganized Point Clouds". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 2917–2923.

[29] Diego Valsesia, Giulia Fracastoro, and Enrico Magli. "Learning Localized Generative Models for 3D Point Clouds via Graph Convolution". In: (2018).

[30] Thumeera R Wanasinghe et al. "Automated Seedling Height Assessment for Tree Nurseries Using Point Cloud Processing". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 3686–3691.

[31] Jiajun Wu et al. "Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling". In: *NeurIPS*. 2016, pp. 82–90.

[32] Zhirong Wu et al. "3d shapenets: A deep representation for volumetric shapes". In: *CVPR*. 2015, pp. 1912–1920.

[33] Jianxiong Xiao et al. "Sun database: Large-scale scene recognition from abbey to zoo". In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE. 2010, pp. 3485–3492.

[34] Jianwen Xie et al. "Learning descriptor networks for 3d shape synthesis and analysis". In: *CVPR*. 2018, pp. 8629–8638.

[35] Guandao Yang et al. "PointFlow: 3D Point Cloud Generation With Continuous Normalizing Flows". In: *The IEEE International Conference on Computer Vision (ICCV)*. Oct. 2019.

[36] Guang Yang et al. "DAGAN: deep de-aliasing generative adversarial networks for fast compressed sensing MRI reconstruction". In: *IEEE transactions on medical imaging* 37.6 (2017), pp. 1310–1321.

[37] Li Yi et al. "A scalable active framework for region annotation in 3d shape collections". In: *ACM Transactions on Graphics (TOG)* 35.6 (2016), p. 210.

[38] Shuaifeng Zhi et al. "LightNet: A Lightweight 3D Convolutional Neural Network for Real-Time 3D Object Recognition." In: *3DOR*. 2017.

[39] Bo Zhu et al. "Image reconstruction by domain-transform manifold learning". In: *Nature* 555.7697 (2018), p. 487.