

Inverse Graphics GAN: Learning to Generate 3D Shapes from Unstructured 2D Data

Sebastian Lunz¹ Yingzhen Li² Andrew Fitzgibbon² Nate Kushman²

Abstract

Recent work has shown the ability to learn generative models for 3D shapes from only unstructured 2D images. However, training such models requires differentiating through the rasterization step of the rendering process, therefore past work has focused on developing bespoke rendering models which smooth over this non-differentiable process in various ways. Such models are thus unable to take advantage of the photo-realistic, fully featured, industrial renderers built by the gaming and graphics industry. In this paper we introduce the first scalable training technique for 3D generative models from 2D data which utilizes an off-the-shelf non-differentiable renderer. To account for the non-differentiability, we introduce a proxy neural renderer to match the output of the non-differentiable renderer. We further propose discriminator output matching to ensure that the neural renderer learns to smooth over the rasterization appropriately. We evaluate our model on images rendered from our generated 3D shapes, and show that our model can consistently learn to generate better shapes than existing models when trained with exclusively unstructured 2D images.

1. Introduction

Generative adversarial networks (GANs) have produced impressive results on 2D image data (Karras et al., 2019; Brock et al., 2019). Many visual applications, such as gaming, require 3D models as inputs instead of just images, however, and directly extending existing GAN models to 3d, requires access to 3D training data (Wu et al., 2016; Riegler et al., 2017). This data is expensive to generate and so exists in abundance only for only very common classes. Ideally we'd like to be able to learn to generate 3D models while training with only 2D image data which is much more widely available and much cheaper and easier to obtain.

¹Work done during an internship at Microsoft Research

²Microsoft Research. Correspondence to: Sebastian Lunz <sl767@cam.ac.uk>, Nate Kushman <nate@kushman.org>.

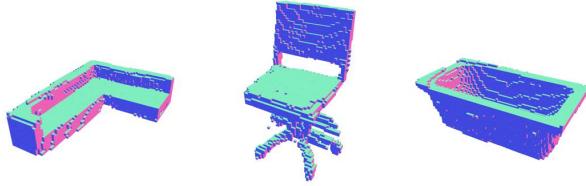


Figure 1. 3D shapes generated by training IG-GAN on unstructured 2D images rendered from three ShapeNet classes.

Training with only 2D data allows us to use any 3D representation. Our interest is in creating 3D models for gaming applications which typically rely on 3D meshes, but direct mesh generation is not ammenable to generating arbitrary topologies since most approaches are based on deforming a template mesh. So we instead choose to work with voxel representations because they can represent arbitrary topologies, can easily be converted to meshes using the marching cubes algorithm, and can be made differentiable by representing the occupancy of each voxel by a real number $\in [0, 1]$ which identifies the probability of voxel occupancy.

In order to learn with an end-to-end differentiable model, we need to differentiate through the process of rendering the 3D model to a 2D image, but the rasterization step in rendering is inherently non-differentiable. As a result, past work on 3D generation from 2D images has focused on differentiable renderers which are hand built from scratch to smooth over this non-differentiable step in various ways. However, standard photo realistic industrial renderers created by the gaming industry (e.g. Unreal Engine, Unity) are not differentiable, and so with such methods we cannot use these renderers, and must rely instead on the simple differentiable renderers build by the research community. In particular two aspects of the rendering process are non-differentiable: (1) the rasterization step inside of the renderer is inherently non-differentiable as a result of occlusion and (2) sampling the continuous voxel grid to generate a mesh is also not differentiable. This second step is required because typical industrial renderers take a mesh as input and we can easily convert a binary voxel grid to a mesh, but continuous voxel inputs do not have a meaningful mesh representation. So rendering a continuous voxel grid using an off-the-shelf

renderer requires first sampling a binary voxel grid from the distribution defined by the continuous voxel grid.

In this paper we introduce the first *scalable* training technique for 3D generative models from 2D data which utilises an off-the-shelf non-differentiable renderer. Examples of the result of our method can be seen in Figure 1. Key to our method is the introduction of a proxy neural renderer based on the recent successes of neural rendering (Nguyen-Phuoc et al., 2018) which directly renders the continuous voxel grid generated by the 3D generative model. It addresses the two challenges of the non-differentiability of the off-the-shelf render as follows:

Differentiate through the Neural Renderer: The proxy neural renderer is trained to match the rendering output of the off-the-shelf renderer given a 3D mesh input. This allows back-propagation of the gradient from the GAN discriminator through the neural renderer to the 3D generative model, enabling training using gradient descent.

Discriminator Output Matching: In order to differentiate through the voxel sampling step we also train the proxy neural renderer using a novel loss function which we call *discriminator output matching*. This accounts for the fact that the neural renderer can only be trained to match the off-the-shelf renderer for binary inputs, which leaves it free to generate arbitrary outputs for the (typically) non-binary voxel grids created by the generator. We constrain this by computing the discriminator loss of an image rendered by the neural renderer when passed through the discriminator. This loss is matched to the average loss achieved by randomly thresholding the volume, rendering the now binary voxels with the off-the-shelf renderer, and passing the resulting image through the discriminator. This addresses the instance-level non-differentiability issue and instead targets the differentiable loss defined on the population of generated discrete 3D shapes, forcing the neural renderer to generate images which represent the continuous voxel grids as smoothly interpolation between the binary choice from the perspective of the discriminator.

We evaluate our model on a variety of synthetic image data sets generated from 3D models in ShapeNet (Chang et al., 2015) as well as the natural image dataset of Chanterelle mushrooms introduced in Henzler et al. (2019), and show that our model generates 3D meshes whose renders generate improved 2D FID scores compared to both Henzler et al. (2019) and a 2D GAN baseline.

2. Related Work

Geometry Based Approaches (or 3D Reconstruction): Reconstructing the underlying 3D scene from only 2D images has been one of the long-standing goals of computer vision. Classical work in this area has focused on geometry

based approaches in the single instanced setting where the goal was only to reconstruct a single 3D object or scene depicted in one or more 2D images (Bleyer et al., 2011; De Bonet & Viola, 1999; Broadhurst et al., 2001; Galliani et al., 2015; Kutulakos & Seitz, 2000; Prock & Dyer, 1998; Schönberger et al., 2016; Seitz et al., 2006; Seitz & Dyer, 1999). This early work was not learning based, however, and so was unable to reconstruct any surfaces which do not appear in the image(s).

Learning to Generate from 3D Supervision: Learning-based 3D reconstruction techniques use a training set of samples to learn a distribution over object shapes. Much past work has focused on the simplest learning setting in which we have access to full 3D supervision. This includes work on generating voxels (Brock et al., 2016; Choy et al., 2016; Riegler et al., 2017; Wu et al., 2016; Xie et al., 2019), generating point-clouds (Achlioptas et al., 2018; Fan et al., 2017; Jiang et al., 2018; Yang et al., 2019; Achlioptas et al., 2018; Li et al., 2018), generating meshes (Groueix et al., 2018; Pan et al., 2019; Wang et al., 2018) and generating implicit representations (Atzmon et al., 2019; Chen & Zhang, 2019; Genova et al., 2019; Huang et al., 2018; Mescheder et al., 2019; Michalkiewicz et al., 2019; Park et al., 2019; Saito et al., 2019; Xu et al., 2019). Creating 3D training data is much more expensive, however, because it requires either skilled artists or a specialized capture setup. So in contrast to all of this work we focus on learning only from unstructured 2D image data which is more readily available and cheaper to obtain.

Learning to Generate from 2D Supervision: Past work on learning to generate 3D shapes by training on only 2D images has mostly focused on differentiable renderers. We can categorize this work based on the representation used. Mesh techniques (Kanazawa et al., 2018; Chen & Zhang, 2019; Genova et al., 2018; Henderson & Ferrari, 2019) are based on deforming a single template mesh or a small number of pieces (Henderson & Ferrari, 2019), while Loper & Black (2014) and Palazzi et al. (2018) use only a low-dimensional pose representation, so neither is amenable to generating arbitrary topologies. Concurrent work on implicit models (Niemeyer et al., 2019; Liu et al., 2019) can directly learn an implicit model from 2D images without ever expanding to another representation, but these methods rely on having camera intrinsics for each image, which is usually unavailable with 2D image data. Our work instead focuses on working with unannotated 2D image data.

The closest work to ours uses voxel representations (Gadelha et al., 2017; Henzler et al., 2019). Voxels can represent arbitrary topologies and can easily be converted to a mesh using the marching cubes algorithm. Furthermore, although it is not a focus of this paper, past work has shown that the voxel representation can be scaled to relatively high resolutions

through the use of sparse representations (Riegler et al., 2017). Gadelha et al. (2017) employs a visual hull based differential renderer that only considers a smoothed version of the object silhouette, while Henzler et al. (2019) relies on a very simple emission-absorption based lighting model. As we show in the results, both of these models struggle to take advantage of lighting and shading information which reveals surface differences, and so they struggle to correctly represent concavities like bathtubs and sofas.

In contrast to all previous work, our goal is to be able to take advantage of fully-featured industrial renderers which included many advanced shading, lighting and texturing features. However these renderers are typically not built to be differentiable, which is challenging to work with in machine learning pipelines. The only work we are aware of which uses an off-the-shelf render for 3D generation with 2D supervision is Rezende et al. (2016). In order to differentiate through the rendering step they use the REINFORCE gradients (Williams, 1992). However, REINFORCE scales very poorly with number of input dimensions, allowing them to show results on simple meshes only. In contrast, our method scales *much* better since dense gradient information can flow through the proxy neural renderer.

Neural Rendering With the success of 2D generative models, it has recently become popular to skip the generation of an explicit 3D representation. *Neural Rendering* techniques focus only on simulating 3D by using a neural network to generate 2D images directly from a latent space with control over the camera angle (Eslami et al., 2018; Nguyen-Phuoc et al., 2019; Sitzmann et al., 2019) and properties of objects in the scene (Liao et al., 2019). In contrast, our goal is to generate the 3D shape itself, not merely controllable 2D renders of it. This is important in circumstances like gaming where the underlying rendering framework is may be fixed, or where we need direct access to the underlying 3D shape itself, such as in CAD/CAM applications. We do however build directly on RenderNet Nguyen-Phuoc et al. (2018) which is a neural network that can be trained to generate 2D images from 3D shapes by matching the output of an off-the-shelf renderer.

Differentiating Through Discrete Decisions Recent work has looked at the problem of differentiating through discrete decisions. Maddison et al. (2017) and Jang et al. (2017) consider smoothing over the discrete decision and Tucker et al. (2017) extends this with sampling to debias the gradient estimates. In section 3 we discuss why these methods cannot be applied in our setting. Liao et al. (2018) discusses why we cannot simply differentiate through the Marching Cubes algorithm, and also suggests using continuous voxel values to generate a probability distribution over 3D shapes. However in their setting they have ground truth 3D data so they directly use these probabilities to compute a loss

and do not have to differentiate through the voxel sampling process as we do when training from only 2D data.

3. IG-GAN

We wish to train a generative model for 3D shapes such that rendering these shapes with an off-the-shelf renderer generates images that match the distribution of 2D training image dataset. The generative model $G_\theta(\cdot)$ takes in a random input vector $\mathbf{z} \sim p(\mathbf{z})$ and generate a continuous voxel representation of the 3D object $\mathbf{x}_c = G_\theta(\mathbf{z})$. Then the voxels \mathbf{x}_c are fed to a *non-differentiable* rendering process, where the voxels first are thresholded to discrete values $\mathbf{x}_d \sim p(\mathbf{x}_d|\mathbf{x}_c)$, then the discrete-value voxels \mathbf{x}_d are rendered using the off-the-shelf renderer (e.g. OpenGL) $\mathbf{y} = R_d(\mathbf{x}_d)$. In summary, this generating process samples a 2D image $\mathbf{y} \sim p_G(\mathbf{y})$ as follows:

$$\begin{aligned} \mathbf{x}_c \sim p_G(\mathbf{x}_c) &\Leftrightarrow \mathbf{z} \sim p(\mathbf{z}), \mathbf{x}_c = G_\theta(\mathbf{z}), \\ \mathbf{y} \sim p_G(\mathbf{y}) &\Leftrightarrow \mathbf{x}_c \sim p_G(\mathbf{x}_c), \mathbf{x}_d \sim p(\mathbf{x}_d|\mathbf{x}_c), \mathbf{y} = R_d(\mathbf{x}_d). \end{aligned} \quad (1)$$

Like many GAN algorithms, a discriminator D_ϕ is then trained on both images sampled from the 2D data distribution $p_D(\mathbf{y})$ and generated images sampled from $p_G(\mathbf{y})$. We consider maximising e.g. the classification-based GAN cross-entropy objective when training the discriminator

$$\max_\phi \mathcal{L}_{\text{dis}}(\phi) := \mathbb{E}_{p_D(\mathbf{y})} [\log D_\phi(\mathbf{y})] + \mathbb{E}_{p_G(\mathbf{y})} [\log(1 - D_\phi(\mathbf{y}))]. \quad (2)$$

A typical GAN algorithm trains the generator $G_\theta(\cdot)$ by maximising a loss defined by the discriminator, e.g.

$$\max_\theta \mathcal{L}_{\text{gen}}(\theta) := \mathbb{E}_{p_G(\mathbf{y})} [\log D_\phi(\mathbf{y})]. \quad (3)$$

For 2D image GAN training, optimisation is usually done by gradient descent, where the gradient is computed via the reparameterisation trick (Salimans et al., 2013; Kingma & Welling, 2014; Rezende et al., 2014). Unfortunately, the reparameterisation trick is not applicable in our setting since the generation process (1) involves sampling discrete variable \mathbf{x}_d thus non-differentiable. An initial idea to address this issue would be to use the REINFORCE gradient estimator (Williams, 1992) for the generative model loss (3):

$$\nabla_\theta \mathcal{L}_{\text{gen}}(\theta) = \mathbb{E}_{p_G(\mathbf{x}_c)} \left[\mathbb{E}_{p(\mathbf{x}_d|\mathbf{x}_c)} [\log D_\phi(R_d(\mathbf{x}_d))] S_G(\mathbf{x}_c) \right], \quad (4)$$

with $S_G(\mathbf{x}_c) = \nabla_\theta \log p_G(\mathbf{x}_c)$ short-hands the score function of $p_G(\mathbf{x}_c)$. Here the expectation term $\mathbb{E}_{p(\mathbf{x}_d|\mathbf{x}_c)} [\log D_\phi(R_d(\mathbf{x}_d))]$ is often called the “reward” of generating $\mathbf{x}_c \sim p_G(\mathbf{x}_c)$. Intuitively, the gradient ascent update using (4) would encourage the generator to generate \mathbf{x}_c with high reward, thus fooling the discriminator. But again REINFORCE is not directly applicable as the score function $S_G(\mathbf{x}_c)$ is intractable (the distribution $p_G(\mathbf{x}_c)$ is implicitly

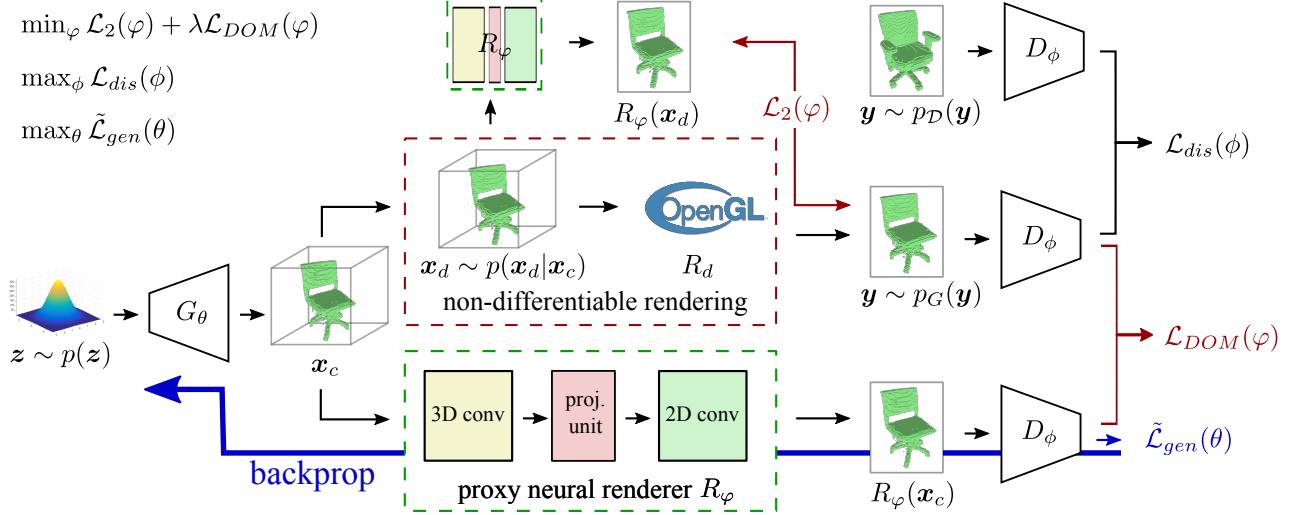


Figure 2. The architecture and training setup for IG-GAN.

defined by neural network transformations of noise variables). A second attempt for gradient approximation would replace the discrete sampling step $\mathbf{x}_d \sim p(\mathbf{x}_d|\mathbf{x}_c)$ with continuous value approximations (Jang et al., 2017; Maddison et al., 2017), with the hope that this enables the usage of the reparameterisation trick. However, the off-the-shelf renderer is treated as black-box so we do not assume the back-propagation mechanism is implemented there. Even worse, the discrete variable sampling step is necessary as off-the-shelf renderer can only work with discrete voxel maps. Therefore the instance-level gradient is not defined on the discrete voxel grid, and gradient approximation methods based on continuous relaxations cannot be applied either.

To address the non-differentiability issues, we introduce a proxy neural renderer $R_\varphi(\cdot)$ as a pathway for back-propagation in generator training. In detail, we define a neural renderer $\tilde{y} = R_\varphi(\mathbf{x}_c)$ which directly renders the continuous voxel representation \mathbf{x}_c into the 2D image \tilde{y} . To encourage realistic renderings that are closed to the results from the off-the-shelf renderer, the neural renderer is trained to minimise the ℓ_2 error of rendering on discrete voxels:

$$\mathcal{L}_2(\varphi) = \mathbb{E}_{p_G(\mathbf{x}_c)p(\mathbf{x}_d|\mathbf{x}_c)} [\|R_\varphi(\mathbf{x}_d) - R_d(\mathbf{x}_d)\|_2^2]. \quad (5)$$

If the neural renderer matches closely with the off-the-shelf renderer on rendering discrete voxel grids, then we can replace the non-differentiable renderer $R_d(\cdot)$ in (4) with the neural renderer $R_\varphi(\cdot)$:

$$\begin{aligned} \nabla_\theta \mathcal{L}_{gen}(\theta) &= \mathbb{E}_{p_G(\mathbf{x}_c)} [\mathbb{E}_{p(\mathbf{x}_d|\mathbf{x}_c)} [\log D_\phi(R_d(\mathbf{x}_d))] S_G(\mathbf{x}_c)] \\ &\approx \mathbb{E}_{p_G(\mathbf{x}_c)} [\mathbb{E}_{p(\mathbf{x}_d|\mathbf{x}_c)} [\log D_\phi(R_\varphi(\mathbf{x}_d))] S_G(\mathbf{x}_c)]. \end{aligned} \quad (6)$$

The intractability of $S_G(\mathbf{x}_c)$ remains to be addressed. Notice that the neural renderer can take in both dis-

crete and continuous voxel grids as inputs, therefore the instance-level gradient $\nabla_{\mathbf{x}} \log D_\phi(R_\varphi(\mathbf{x}))$ is well-defined and computable for both $\mathbf{x} = \mathbf{x}_d$ and $\mathbf{x} = \mathbf{x}_c$. This motivates the “reward approximation” approach $\mathbb{E}_{p(\mathbf{x}_d|\mathbf{x}_c)} [\log D_\phi(R_\varphi(\mathbf{x}_d))] \approx \log D_\phi(R_\varphi(\mathbf{x}_c))$ which sidesteps the intractability of $S_G(\mathbf{x}_c)$ via the reparameterisation trick:

$$\begin{aligned} \nabla_\theta \mathcal{L}_{gen}(\theta) &\approx \mathbb{E}_{p_G(\mathbf{x}_c)} [\mathbb{E}_{p(\mathbf{x}_d|\mathbf{x}_c)} [\log D_\phi(R_\varphi(\mathbf{x}_d))] S_G(\mathbf{x}_c)] \\ &\approx \mathbb{E}_{p_G(\mathbf{x}_c)} [\log D_\phi(R_\varphi(\mathbf{x}_c)) \nabla_\theta \log p_G(\mathbf{x}_c)] \\ &= \mathbb{E}_{p(\mathbf{z})} [\nabla_\theta G_\theta(\mathbf{z}) \nabla_{\mathbf{x}_c} \log D_\phi(R_\varphi(\mathbf{x}_c))|_{\mathbf{x}_c=G_\theta(\mathbf{z})}] \\ &= \nabla_\theta \mathbb{E}_{p_G(\mathbf{x}_c)} [\log D_\phi(R_\varphi(\mathbf{x}_c))] := \nabla_\theta \tilde{\mathcal{L}}_{gen}(\theta). \end{aligned} \quad (7)$$

The reward approximation quality is key to the performance of generative model, as gradient ascent using (7) would encourage the generator to create continuous voxel grids \mathbf{x}_c for which the neural renderer would return realistic rendering results. Notice the neural renderer is free to render arbitrary outcomes for continuous voxel grids if it is only trained by minimising $\mathcal{L}_2(\varphi)$ on discrete voxel grids. Therefore \mathbf{x}_c is not required to resemble the desired 3D shape in order to produce satisfactory neural rendering results. Since the 3D generative model $G_\theta(\cdot)$ typically creates non-discrete voxel grids, the generated 2D images using the off-the-shelf renderer will match poorly to the training images. We address this issue by training the neural renderer with a novel loss function which we call *discriminator output matching* (DOM). Define $F(\cdot) = \log D_\phi(\cdot)$, the DOM loss is

$$\mathcal{L}_{DOM}(\varphi) = \mathbb{E}_{p_G(\mathbf{x}_c)p(\mathbf{x}_d|\mathbf{x}_c)} [(F(R_d(\mathbf{x}_d)) - F(R_\varphi(\mathbf{x}_c)))^2]. \quad (8)$$

Using neural networks of enough capacity, the optimal neural renderer achieves perfect reward approximation, i.e. $\log D_\phi(R_{\varphi^*}(\mathbf{x}_c)) = \mathbb{E}_{p(\mathbf{x}_d|\mathbf{x}_c)} [\log D_\phi(R_d(\mathbf{x}_d))]$. This

approach forces the neural renderer to preserve the population statistics of the discrete rendered images defined by the discriminator. Therefore to fool the discriminator, the 3D generative model must generate continuous voxel grids which correspond to meaningful representations of the underlying 3D shapes. In practice the neural renderer is trained using a combination of the two loss functions

$$\min_{\varphi} \mathcal{L}_{\text{render}}(\varphi) := \mathcal{L}_2(\varphi) + \lambda \mathcal{L}_{\text{DOM}}(\varphi). \quad (9)$$

We name the proposed method *inverse graphics GAN* (IG-GAN), as the neural renderer in back-propagation time “inverts” the off-the-shelf renderer and provide useful gradients for the 3D generative model training. The model is also visualised in Figure 2, which is trained in an end-to-end fashion. To speed up the generative model training, the neural renderer can be pretrained on a generic data set, like tables or cubes, that can differ significantly from the 2D images that the generative model is eventually trained on.

4. Experimental Setup

4.1. Implementation Details

Off-the-shelf Renderer Our rendering engine is based on the Pyrender ([Matl](#)) which is built on top of OpenGL.

Architecture We employ a 3D convolutional GAN architecture for the generator ([Wu et al., 2016](#)) with a 64^3 voxel resolution. To incorporate the viewpoint, the rigid body transformation embeds the 64^3 grid into a 128^3 resolution. We render the volumes with a RenderNet ([Nguyen-Phuoc et al., 2018](#)) architecture, with the modification of using 2 residual blocks in 3D and 4 residual blocks in 2D only. The discriminator architecture follows the design in DCGAN ([Radford et al., 2016](#)), taking images of 128^2 resolution. Additionally, we add spectral normalization to the discriminator ([Miyato et al., 2018](#)) to stabilize training.

Hyperparameters We employ a 1:1:1 updating scheme for generator, discriminator and the neural renderer, using learning rates of $2e-5$ for the neural renderer and $2e-4$ for both generator and discriminator. The Discriminator Output Matching loss is weighted by $\lambda = 100$ over the \mathcal{L}_2 loss, as in (9). We found that training was stable against changes in λ and extensive tuning was not necessary. The binerization distribution $p(\mathbf{x}_d|\mathbf{x}_c)$ was chosen as a global thresholding, with the threshold being distributed uniformly in $[0, 1]$.

4.2. Datasets

We evaluate our model on a variety of synthetic datasets generated from 3D models in the ShapeNet ([Chang et al., 2015](#)) database as well as on a real data set consisting of chanterelle mushrooms, introduced in ([Henzler et al., 2019](#)).

We synthesize images from three different categories of ShapeNet objects, *Chairs*, *Couches* and *Bathtubs*. For each of these categories, we generate three different datasets:

500: A very small dataset consisting of 500 images taken from 500 different objects in the corresponding data set, each rendered from a single viewpoint.

One per Model: A more extensive one where we sample each object in the 3D data set once from a single viewpoint for the *Chairs* and *Couches* data, and from four viewpoints for *Bathtubs*, due to the small number of objects in this category. This results in data sets containing 6777 chairs, 3173 couches and 3424 images of bathtubs.

Unlimited: Finally, we render each object from a different viewpoint throughout each training epoch. This leads to a theoretically unlimited data set of images. However, all images are generated from the limited number of objects contained in the corresponding ShapeNet category.

In order to render the ShapeNet volumes, we light them from two beam light sources that illuminate the object from 45° below the camera and 45° to its left and right. The camera always points directly at the object, and we uniformly sample from all 360° of rotation, with an elevation between 15 and 60 degrees for bathtubs and couches and -30 and 60 degrees for chairs. The limitation on elevation angle is chosen to generate a dataset that is as realistic as possible, given the object category.

LVP: We also consider a limited viewpoint (LVP) setting for the chairs dataset where the azimuth rotation is limited to 60° in each direction from center, and elevation is limited to be between 15 ad 60 degrees. This setting is meant to further simulate the viewpoint bias observed in natural photographs.

Chanterelle Mushrooms: We prepare the *Chanterelle* data by cropping and resizing the images to 128^2 resolution and by unifying the mean intensity in an additive way. Note that the background of the natural images has been masked out in the original open-sourced dataset.

4.3. Baselines

We compare to the following state-of-the-art methods for learning 3D voxel generative models from 2D data:

Visual Hull: This is the model from [Gadelha et al. \(2017\)](#) which learned from a smoothed version of object silhouettes.

Absorbtion Only: This is the model from [Henzler et al. \(2019\)](#) which assumes voxels absorb light based on their fraction of occupancy.

2D-DCGAN: For comparison we also show results from a DCGAN ([Radford et al., 2016](#)) trained on 2D images only. While this baseline is solving a different task and is not able to produce any 3D objects, it allows a comparison of the

Table 1. FID scores computed on ShapeNet objects (bathtubs, couches and chairs).

| # of Images Dataset | 500 | | | One Per Model (≈ 3000) | | | Unlimited | | | |
|------------------------|--------------|--------------|--------------|----------------------------------|-------------|-------------|-------------|-------------|-------------|--------------------|
| | Tubs | Couches | Chairs | Tubs | Couches | Chairs | Tubs | Couches | Chairs | LVP |
| 2D-DCGAN | 737.7 | 540.5 | 672.8 | 461.8 | 354.3 | 362.3 | 226.7 | 210.9 | 133.2 | 237.9 ² |
| Visual Hull | 305.8 | 279.3 | 183.4 | 184.6 | 106.2 | 37.1 | 90.1 | 35.1 | 15.7 | 34.5 |
| Absorbtion Only | 336.9 | 282.9 | 218.2 | 275.8 | 78.0 | 32.8 | 104.5 | 25.5 | 23.8 | 38.6 |
| IG-GAN (Ours) | 187.8 | 114.1 | 119.9 | 67.5 | 35.8 | 20.7 | 44.0 | 17.8 | 13.6 | 20.6 |

generated image quality.

We show results from our reimplementation of these models because we found [Gadelha et al. \(2017\)](#) performed better by using recently developed GAN stabalization techniques, i.e. spectral normalization ([Miyato et al., 2018](#)), and the code for [Henzler et al. \(2019\)](#) was not available at the time we ran our original results. A discussion of the Emission-Absorption model also proposed in [Henzler et al. \(2019\)](#) is provided in the supplemental material ¹. To provide the most favorable comparison for the baselines, each baseline is trained on a dataset of images synthesized from ShapeNet using the respective choice of rendering (Visual Hull or Absorbtion-Only) from the 3D objects.

4.4. Evaluation Metrics

We chose to evaluate the quality of the generated 3D models by rendering them to 2D images and computing Frchet Inception Distances (FIDs) ([Heusel et al., 2017](#)). This focuses the evaluation on the observed visual quality, preventing if from considering what the model generates in the unobserved insides of the objects. All FID scores reported in the main paper use an Inception network ([Szegedy et al., 2016](#)) retrained to classify Images generated with our renderer because we found this to better align with our sense of the visual quality given the domain gap between ImageNet and our setting. We have trained the Inception network to classify rendered images of the 21 largest object classes in ShapeNet, achieving 95.3% accuracy. In the supplemental material we show FID scores using the traditional Inception network trained on ImageNet ([Deng et al., 2009](#)), and qualitatively they are very similar.

5. Results

5.1. Quantitative Evaluation

We can see clearly from Table 1 that our approach (IG-GAN) significantly out-performs the baselines on all dasaset. The largest of these gains is obtained on the data sets containing many concavities, like couches and bathtubs. Furthermore,

the advantage of the proposed method becomes more significant when the dataset is smaller and the viewpoint is more restrictive. Since our method can more easily take advantage of the lighting and shading cue provided by the images, we believe it can extract more meaningful information per training sample, hence producing better results in these settings. On the Unlimited dataset the baseline methods seem to be able to mitigate some of their disadvantage by simply seeing enough views of each training model, but still our approach generates considerably better FID scores even in this setting. We note that the very poor results from the 2D-DCGAN stem from computing FIDs using our retrained Inception network, which seems to easily pick up on any unrealistic artifacts generated by the GAN. The supplemental materials show the FID scores using the standard Inception net, and while IG-GAN still outperforms the 2D-DCGAN considerably, the resulting scores are closer.

5.2. Qualitative Evaluation

From Figure 3 we can see that IG-GAN produces high-quality samples on all three object categories. Furthermore, we can see from Figure 6 that the generated 3D shapes are superior to the baselines. This is particularly evident in the context of concave objects like bathtubs or couches. Here, generative models based on visual hull or absorption rendering fail to take advantage of the shading cues needed to detect the hollow space inside the object, leading to e.g. seemingly filled bathtubs. The shortcoming of the baseline models has already been noticed by [Gadelha et al. \(2017\)](#). Our approach, on the other hand, successfully detects the interior structure of concave objects using the differences in light exposures between surfaces, enabling it to accurately capture concavities and hollow spaces.

On the chair dataset, the advantages of our proposed method are evident on flat surfaces. Any uneven surfaces generated by mistake are promptly detected by our discriminator which can easily differences in light exposure, forcing the generator to produce clean and flat surfaces. The baseline methods however are unable to render such impurities in a way that is evident to the discriminator, leading to generated

¹https://lunz-s.github.io/iggan/iggan_supplemental.pdf

²A fair comparison to DCGAN is impossible, as the generator is trained on LVP data, but FID evaluations use a 360 ° view.

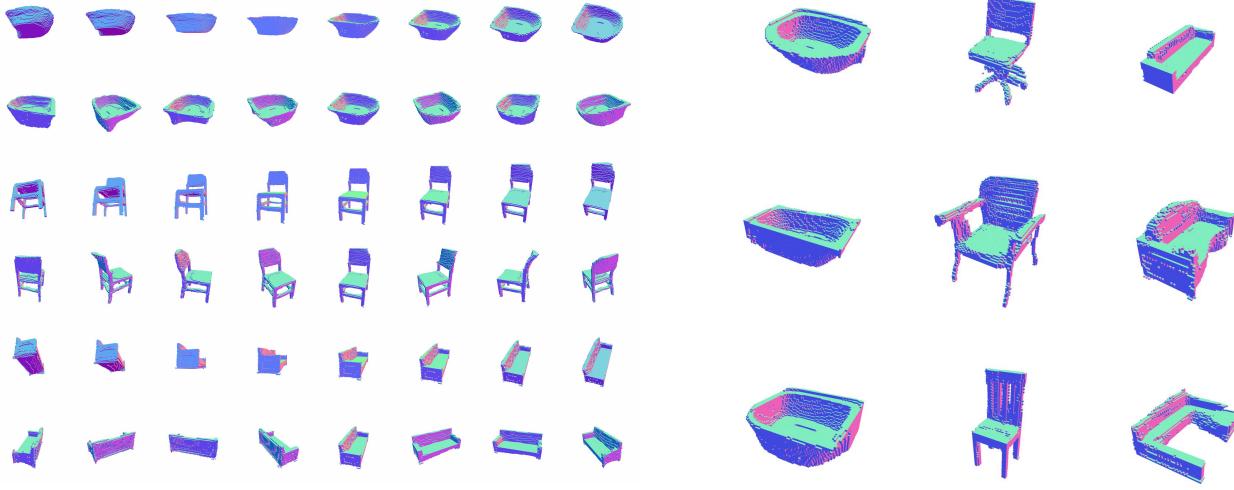


Figure 3. Normal Maps of objects generated by IG-GAN on the ‘Unlimited’ datasets. The left panel shows a single sample rendered in different view points, and the right panel shows multiple samples rendered from a canonical viewpoint.



Figure 4. Results on the chairs LVP dataset. Unlike our method, the baseline can not extract sufficient information from the data to create chair samples with flat surfaces.

samples with grainy and uneven surfaces. This effect is most obvious on the chairs with limited views (LVP), as shown in Figure 4. A large selection of randomly generated samples from all methods can be found in the supplemental material.

Training on Natural Images Finally, we demonstrate that the proposed method is able to produce realistic samples when trained on a dataset of natural images. Figure 5 shows samples from a model trained on the Chanterelle mushrooms dataset from Henzler et al. (2019).

5.3. Ablations

Discriminator output matching We study the effect of the proposed discriminator output matching (DOM) loss in various scenarios. In Table 2, we report the FID scores on the models trained without the DOM loss, from this compari-



Figure 5. Chanterelle mushroom dataset samples and generated shapes from our model trained on this dataset.

son we see that the DOM loss plays a crucial role in learning to generate high-quality 3D objects. In Figure 7 we can see that that the non-binary volumes sampled from the generator can be rendered to a variety of different images by OpenGL, depending on the random choice of threshold. Without the DOM loss, the trained neural renderer simply averages over these potential outcomes, considerably smoothing the result in the process and losing information about fine structures in the volume. This leads to weak gradients being passed to the generator, considerably deteriorating sample quality.

Another setting from Table 2 shows the discriminator trained using generated samples rendered by the neural renderer instead of OpenGL. This inherently prevents the mode collapse observed in the above setting. However, it leads to the generator being forced by the discriminator to produce binary voxel maps early on in training. This seems to lead

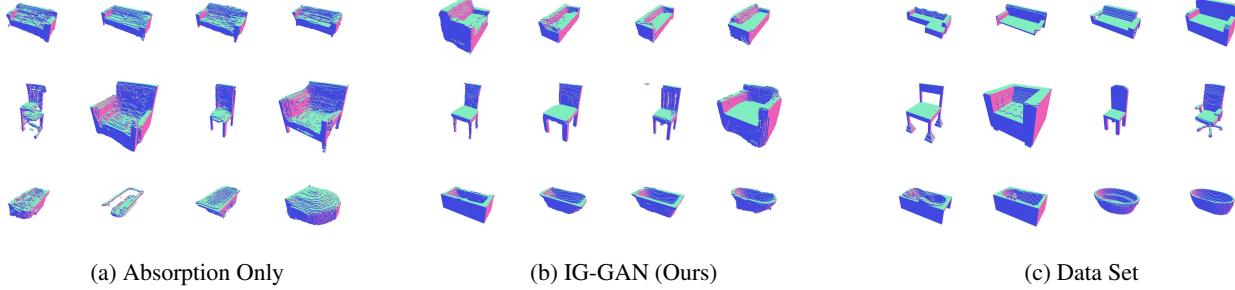


Figure 6. Samples generated by (a) the AO baseline and (b) our model on the ‘One per model’ setting (Dataset samples in panel (c)). The samples from the VH baseline is visually similar to those from the AO baseline. Our method is able to recognize concavities correctly, leading to realistic samples of bathtubs and couches.

Table 2. Ablation results without discriminator output matching (DOM) when training on *chairs/couches* “one per model” datasets. We either fix the pre-trained neural renderer (“Fixed”), or continuing to train it during GAN training (“Retrained”). The generator samples fed to the discriminator are rendered using either OpenGL or the neural renderer. For reference, our model is equivalent to the Retrained OpenGL setup with the addition of the DOM loss and achieves FID scores **20.7/35.8**.

| | OpenGL | RenderNet |
|-----------|--------------------|--------------------|
| Retrained | 86.4/180.1 | 74.7/144.8 |
| Fixed | 113.7/323.5 | 103.9/124.6 |

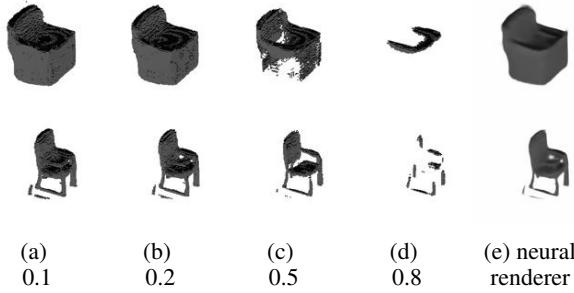


Figure 7. Samples from a generator trained without DOM, rendered using the neural renderer on the continuous sample and OpenGL on various thresholds.

to the generator getting stuck in a local optima, hence deteriorating sample quality.

Pre-training We investigate the effect of various pre-trainings of the neural renderer. All other experiments were conducted with the neural renderer pre-trained on the *Tables* data from ShapeNet (see Table 1). As a comparison, we run the proposed algorithm on the chair data using a neural renderer pre-trained on either the Chair data itself or a simple data set consisting of randomly sampled cubes. As shown in Table 3, the quality of the results produced by our method is robust to changes in the pre-training of the neural

Table 3. Comparisons of neural renderer pre-trainings on different 3D shapes. FIDs are reported for the ‘One per model’ chairs.

| | Chairs | Tables | Random |
|-------|--------|--------|--------|
| Ours | 22.6 | 20.7 | 20.4 |
| Fixed | 37.8 | 105.8 | 141.9 |

renderer. In contrast, if we use a fixed pre-trained renderer it produces reasonable results if pre-trained directly on the domain of interest, but deteriorates significantly if trained only on a related domain. Note that we assume no access to 3D data in the domain of interest so in practice we cannot pre-train in this way.

6. Conclusion

We have presented the first scalable algorithm using an arbitrary off-the-shelf renderer to learn to generate 3D shapes from unstructured 2D data. We have introduced a novel loss term, Discriminator Output Matching, that allows stably training our model on a variety of datasets, thus achieving significantly better FID scores than previous work. In particular, Using light exposure and shadow information in our rendering engine, we are able to generate high-quality convex shapes, like bathtubs and couches, that prior work failed to capture.

The presented framework is general and in-concept can be made to work with any off-the-shelf renderer. In practice, our work can be extended by using more sophisticated photo-realistic rendering engines, to be able to learn even more detailed information about the 3D world from images. By incorporating color, material and lighting prediction into our model we hope to be able to extend it to work with more general real world datasets.

References

- Achlioptas, P., Diamanti, O., Mitliagkas, I., and Guibas, L. Learning representations and generative models for 3d point clouds. *International Conference on Machine Learning*, 2018.
- Atzmon, M., Haim, N., Yariv, L., Israelov, O., Maron, H., and Lipman, Y. Controlling neural level sets. In *Advances in Neural Information Processing Systems*, pp. 2032–2041, 2019.
- Bleyer, M., Rhemann, C., and Rother, C. Patchmatch stereo - stereo matching with slanted support windows. In *BMVC*, January 2011.
- Broadhurst, A., Drummond, T. W., and Cipolla, R. A probabilistic framework for space carving. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 1, pp. 388–393. IEEE, 2001.
- Brock, A., Lim, T., Ritchie, J. M., and Weston, N. Generative and discriminative voxel modeling with convolutional neural networks. *arXiv preprint arXiv:1608.04236*, 2016.
- Brock, A., Donahue, J., and Simonyan, K. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019.
- Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.
- Chen, Z. and Zhang, H. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5939–5948, 2019.
- Choy, C. B., Xu, D., Gwak, J., Chen, K., and Savarese, S. 3d-rn2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pp. 628–644. Springer, 2016.
- De Bonet, J. S. and Viola, P. Voxels: Probabilistic voxelized volume reconstruction. In *Proceedings of International Conference on Computer Vision (ICCV)*, pp. 418–425, 1999.
- Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.
- Eslami, S. A., Rezende, D. J., Besse, F., Viola, F., Morcos, A. S., Garnelo, M., Ruderman, A., Rusu, A. A., Danihelka, I., Gregor, K., et al. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018.
- Fan, H., Su, H., and Guibas, L. J. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 605–613, 2017.
- Gadelha, M., Maji, S., and Wang, R. 3d shape induction from 2d views of multiple objects. In *2017 International Conference on 3D Vision (3DV)*, pp. 402–411. IEEE, 2017.
- Galliani, S., Lasinger, K., and Schindler, K. Massively parallel multiview stereopsis by surface normal diffusion. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 873–881, 2015.
- Genova, K., Cole, F., Maschinot, A., Sarna, A., Vlasic, D., and Freeman, W. T. Unsupervised training for 3d morphable model regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8377–8386, 2018.
- Genova, K., Cole, F., Vlasic, D., Sarna, A., Freeman, W. T., and Funkhouser, T. Learning shape templates with structured implicit functions. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 7154–7164, 2019.
- Groueix, T., Fisher, M., Kim, V. G., Russell, B. C., and Aubry, M. A papier-mâché approach to learning 3d surface generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 216–224, 2018.
- Henderson, P. and Ferrari, V. Learning single-image 3d reconstruction by generative modelling of shape, pose and shading. *International Journal of Computer Vision*, pp. 1–20, 2019.
- Henzler, P., Mitra, N. J., and Ritschel, T. Escaping plato’s cave: 3d shape from adversarial rendering. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.
- Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pp. 6626–6637, 2017.
- Huang, Z., Li, T., Chen, W., Zhao, Y., Xing, J., LeGendre, C., Luo, L., Ma, C., and Li, H. Deep volumetric video from very sparse multi-view performance capture. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 336–354, 2018.
- Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017.

- Jiang, L., Shi, S., Qi, X., and Jia, J. Gal: Geometric adversarial loss for single-view 3d-object reconstruction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 802–816, 2018.
- Kanazawa, A., Tulsiani, S., Efros, A. A., and Malik, J. Learning category-specific mesh reconstruction from image collections. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 371–386, 2018.
- Karras, T., Laine, S., and Aila, T. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4401–4410, 2019.
- Kingma, D. P. and Welling, M. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.
- Kutulakos, K. N. and Seitz, S. M. A theory of shape by space carving. *International journal of computer vision*, 38(3):199–218, 2000.
- Li, C.-L., Zaheer, M., Zhang, Y., Poczos, B., and Salakhutdinov, R. Point cloud gan. *arXiv preprint arXiv:1810.05795*, 2018.
- Liao, Y., Donne, S., and Geiger, A. Deep marching cubes: Learning explicit surface representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2916–2925, 2018.
- Liao, Y., Schwarz, K., Mescheder, L., and Geiger, A. Towards unsupervised learning of generative models for 3d controllable image synthesis. *arXiv preprint arXiv:1912.05237*, 2019.
- Liu, S., Saito, S., Chen, W., and Li, H. Learning to infer implicit surfaces without 3d supervision. In *Advances in Neural Information Processing Systems*, pp. 8293–8304, 2019.
- Loper, M. M. and Black, M. J. Opendr: An approximate differentiable renderer. In *European Conference on Computer Vision*, pp. 154–169. Springer, 2014.
- Maddison, C. J., Mnih, A., and Teh, Y. W. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.
- Matl, M. Pyrender. URL <https://github.com/mmatl/pyrender>.
- Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., and Geiger, A. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4460–4470, 2019.
- Michalkiewicz, M., Pontes, J. K., Jack, D., Baktashmotagh, M., and Eriksson, A. Implicit surface representations as layers in neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4743–4752, 2019.
- Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.
- Nguyen-Phuoc, T., Li, C., Theis, L., Richardt, C., and Yang, Y.-L. Hologan: Unsupervised learning of 3d representations from natural images. *arXiv preprint arXiv:1904.01326*, 2019.
- Nguyen-Phuoc, T. H., Li, C., Balaban, S., and Yang, Y. Rendernet: A deep convolutional network for differentiable rendering from 3d shapes. In *Advances in Neural Information Processing Systems 31*, pp. 7891–7901, 2018.
- Niemeyer, M., Mescheder, L., Oechsle, M., and Geiger, A. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. *arXiv preprint arXiv:1912.07372*, 2019.
- Palazzi, A., Bergamini, L., Calderara, S., and Cucchiara, R. End-to-end 6-dof object pose estimation through differentiable rasterization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 0–0, 2018.
- Pan, J., Han, X., Chen, W., Tang, J., and Jia, K. Deep mesh reconstruction from single rgb images via topology modification networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 9964–9973, 2019.
- Park, J. J., Florence, P., Straub, J., Newcombe, R., and Lovegrove, S. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 165–174, 2019.
- Prock, A. and Dyer, C. Towards real-time voxel coloring. In *Proceedings of the DARPA Image Understanding Workshop*, volume 1, pp. 2. Citeseer, 1998.
- Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *International Conference on Learning Representations*, 2016.
- Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32, ICML’14*, pp. II–1278–II–1286, 2014.

- Rezende, D. J., Eslami, S. A., Mohamed, S., Battaglia, P., Jaderberg, M., and Heess, N. Unsupervised learning of 3d structure from images. In *Advances in neural information processing systems*, pp. 4996–5004, 2016.
- Riegler, G., Osman Ulusoy, A., and Geiger, A. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3577–3586, 2017.
- Saito, S., Huang, Z., Natsume, R., Morishima, S., Kanazawa, A., and Li, H. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2304–2314, 2019.
- Salimans, T., Knowles, D. A., et al. Fixed-form variational posterior approximation through stochastic linear regression. *Bayesian Analysis*, 8(4):837–882, 2013.
- Schönberger, J. L., Zheng, E., Frahm, J.-M., and Pollefeys, M. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision*, pp. 501–518. Springer, 2016.
- Seitz, S. M. and Dyer, C. R. Photorealistic scene reconstruction by voxel coloring. *International Journal of Computer Vision*, 35(2):151–173, 1999.
- Seitz, S. M., Curless, B., Diebel, J., Scharstein, D., and Szeliski, R. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, volume 1, pp. 519–528. IEEE, 2006.
- Sitzmann, V., Zollhöfer, M., and Wetzstein, G. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Advances in Neural Information Processing Systems*, pp. 1119–1130, 2019.
- Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.
- Tucker, G., Mnih, A., Maddison, C. J., Lawson, J., and Sohl-Dickstein, J. Rebar: Low-variance, unbiased gradient estimates for discrete latent variable models. In *Advances in Neural Information Processing Systems*, pp. 2627–2636, 2017.
- Wang, N., Zhang, Y., Li, Z., Fu, Y., Liu, W., and Jiang, Y.-G. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 52–67, 2018.
- Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.
- Wu, J., Zhang, C., Xue, T., Freeman, B., and Tenenbaum, J. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in neural information processing systems*, pp. 82–90, 2016.
- Xie, H., Yao, H., Sun, X., Zhou, S., and Zhang, S. Pix2vox: Context-aware 3d reconstruction from single and multi-view images. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2690–2698, 2019.
- Xu, Q., Wang, W., Ceylan, D., Mech, R., and Neumann, U. Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. In *Advances in Neural Information Processing Systems*, pp. 490–500, 2019.
- Yang, G., Huang, X., Hao, Z., Liu, M.-Y., Belongie, S., and Hariharan, B. Pointflow: 3d point cloud generation with continuous normalizing flows. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4541–4550, 2019.