Inverse Graphics GAN

Sebastian Lunz
University of Cambridge *
sl767@cam.ac.uk

Yingzhen Li
Microsoft Research
yingzhen.li@microsoft.com

Andrew Fitzgibbon
Microsoft Research
awf@microsoft.com

Nate Kushman
DeepMind *
nkushman@google.com

Abstract

In this paper we introduce the first scalable training technique to utilize an off-the-shelf non-differentiable renderer to train a 3D generative model from only unstructured 2D data. 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 appropriately smooth over the non-differentiable rasterization step in the discrete renderer. We show that our model can consistently learn to generate better shapes than existing models when trained with exclusively unstructured 2D images. The approach works with any rendering algorithm and hence opens possibilities to take advantage of the photo-realistic industrial renderers built by the graphics industry.

1 Introduction

Generative adversarial networks (GANs) have produced impressive results on 2D image data [12, 1]. However, many visual applications, such as gaming, require 3D models as inputs instead of just images, and directly extending existing GAN models to 3D, requires access to 3D training data [31, 26]. This data is expensive to generate and so exists in abundance only for only very common classes. We would hence like to be able to generate 3D models while training with only 2D image data which is much more widely available.

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 arbitary 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 that are hand built from scratch to smooth over this non-differentiable step in various ways. This prevents the use of standard photo realistic industrial renderers created by the gaming industry (e.g. UnReal Engine, Unity) because they cannot easily be made differentiable. To enable the use of such renderers, we must deal with two aspects of the rendering process that 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

*Work done at Microsoft Research

Workshop on Differentiable Vision, Graphics, and Physics in Machine Learning at NeurIPS 2020.

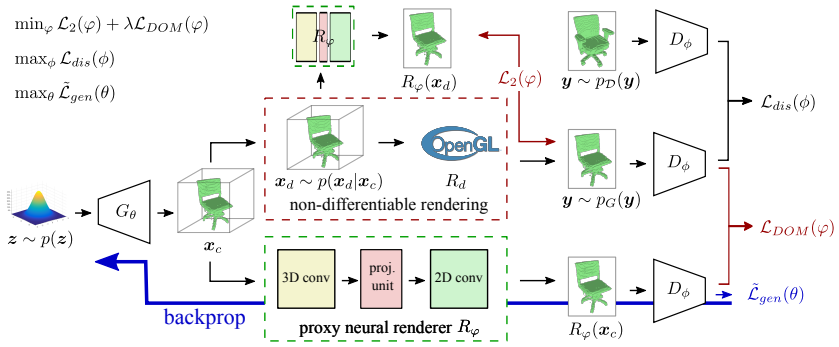


Figure 1: The architecture and training setup for IG-GAN.

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, generating a mesh from this grid and feeding the mesh to the discrete renderer.

In this paper we introduce the first *scalable* training technique to utilize an off-the-shelf non-differentiable renderer to train a 3D generative model from only unstructured 2D data. Key to our method is the introduction of a proxy neural renderer [20] which directly renders the continuous voxel grid generated by the 3D generative model. Our method 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 resulting 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 smooth interpolation between the binary choices from the perspective of the discriminator.

A detailed discussion on the relationship between our work and past work such as [19, 28, 4, 15, 14, 25, 5, 8, 11, 3, 6, 7, 17, 22, 21, 16] can be found in Section A of the supplemental material.

2 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 a training image dataset. The generative model $G_\theta(\cdot)$ takes in a random input vector $\boldsymbol{z} \sim p(\boldsymbol{z})$ and generate a continuous voxel representation of the 3D object $\boldsymbol{x}_c = G_\theta(\boldsymbol{z})$. Then the voxels $\boldsymbol{x}_c$ are fed to a *non-differentiable* renderering process, where the voxels first are thresholded to discrete values $\boldsymbol{x}_d \sim p(\boldsymbol{x}_d|\boldsymbol{x}_c)$, then the discrete-value voxels $\boldsymbol{x}_d$ are renderred using the off-the-shelf renderer (e.g. OpenGL) $\boldsymbol{y} = R_d(\boldsymbol{x}_d)$. In summary, this generating process samples a 2D image $\boldsymbol{y} \sim p_G(\boldsymbol{y})$ as follows:

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

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

$$\max_{\phi} \mathcal{L}_{\text{dis}}(\phi) := \mathbb{E}_{p_{\mathcal{D}}(\boldsymbol{y})} \left[\log D_\phi(\boldsymbol{y})\right] + \mathbb{E}_{p_G(\boldsymbol{y})} \left[\log(1 - D_\phi(\boldsymbol{y}))\right]. \tag{2}$$

The generator $G_\theta(\cdot)$ is trained by e.g. $\max_\theta \mathcal{L}_{\text{gen}}(\theta) := \mathbb{E}_{p_G(\boldsymbol{y})} \left[\log D_\phi(\boldsymbol{y})\right]$. Unfortunately, the generation process (1) involves sampling discrete variable $\boldsymbol{x}_d$ thus making the generator's loss non-differentiable w.r.t. θ. An initial solution would be to use the REINFORCE gradient estimator [30]:

$$\nabla_\theta \mathcal{L}_{\text{gen}}(\theta) = \mathbb{E}_{p_G(\boldsymbol{x}_c)} \left[\mathbb{E}_{p(\boldsymbol{x}_d|\boldsymbol{x}_c)} \left[\log D_\phi(R_d(\boldsymbol{x}_d))\right] S_G(\boldsymbol{x}_c)\right], \quad S_G(\boldsymbol{x}_c) = \nabla_\theta \log p_G(\boldsymbol{x}_c). \tag{3}$$

Intuitively, the gradient ascent update using (3) would encourage the generator to generate $\boldsymbol{x}_c$ with high reward, thus fooling the discriminator. But REINFORCE is not applicable because the score function $S_G(\boldsymbol{x}_c)$ is intractable and, furthermore, we cannot use a continuous relaxation [10, 18] because the off-the-shelf renderer does not support back-propagation. See the supplemental material, Section B, for further discussion of these issues.

To address the non-differentiability issues, we introduce a proxy neural renderer $\tilde{\boldsymbol{y}} = R_\varphi(\boldsymbol{x}_c)$ for rendering continuous voxel representations as a pathway for back-propagation in generator training. To encourage realistic renderings that are close 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:

$$\mathcal{L}_2(\varphi) = \mathbb{E}_{p_G(\boldsymbol{x}_c)p(\boldsymbol{x}_d|\boldsymbol{x}_c)} \left[||R_\varphi(\boldsymbol{x}_d) - R_d(\boldsymbol{x}_d)||_2^2\right]. \tag{4}$$

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 (3) with the neural renderer $R_\varphi(\cdot)$:

$$\nabla_\theta \mathcal{L}_{\text{gen}}(\theta) \approx \mathbb{E}_{p_G(\boldsymbol{x}_c)} \left[\mathbb{E}_{p(\boldsymbol{x}_d|\boldsymbol{x}_c)} \left[\log D_\phi(R_\varphi(\boldsymbol{x}_d))\right] S_G(\boldsymbol{x}_c)\right]. \tag{5}$$

However, we must still address the intractability of $S_G(\boldsymbol{x}_c)$. Notice that the neural renderer can take in both discrete and continuous voxel grids, therefore the instance-level gradient $\nabla_{\boldsymbol{x}} \log D_\phi(R_\varphi(\boldsymbol{x}))$ is well-defined and computable for both $\boldsymbol{x} = \boldsymbol{x}_d$ and $\boldsymbol{x} = \boldsymbol{x}_c$. This motivates the "reward approximation" approach which approximates $\mathbb{E}_{p(\boldsymbol{x}_d|\boldsymbol{x}_c)} \left[\log D_\phi(R_\varphi(\boldsymbol{x}_d))\right]$ in (5) with $\log D_\phi(R_\varphi(\boldsymbol{x}_c))$, sidestepping the intractability of $S_G(\boldsymbol{x}_c)$ via the reparameterisation trick [27, 13, 24]:

$$\begin{aligned}
\nabla_\theta \mathcal{L}_{\text{gen}}(\theta) &\approx \mathbb{E}_{p_G(\boldsymbol{x}_c)} \left[\log D_\phi(R_\varphi(\boldsymbol{x}_c)) \nabla_\theta \log p_G(\boldsymbol{x}_c)\right] \\
&= \mathbb{E}_{p(\boldsymbol{z})} \left[\nabla_\theta G_\theta(\boldsymbol{z}) \nabla_{\boldsymbol{x}_c} \log D_\phi(R_\varphi(\boldsymbol{x}_c))|_{\boldsymbol{x}_c = G_\theta(\boldsymbol{z})}\right] \\
&= \nabla_\theta \mathbb{E}_{p_G(\boldsymbol{x}_c)} \left[\log D_\phi(R_\varphi(\boldsymbol{x}_c))\right] := \nabla_\theta \tilde{\mathcal{L}}_{\text{gen}}(\theta).
\end{aligned} \tag{6}$$

To better facilitate this reward approximation, we train 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

$$\mathcal{L}_{\text{DOM}}(\varphi) = \mathbb{E}_{p_G(\boldsymbol{x}_c)p(\boldsymbol{x}_d|\boldsymbol{x}_c)} \left[(F(R_d(\boldsymbol{x}_d)) - F(R_\varphi(\boldsymbol{x}_c)))^2\right]. \tag{7}$$

The optimal neural renderer achieves $\log D_\phi(R_{\varphi^*}(\boldsymbol{x}_c)) = \mathbb{E}_{p(\boldsymbol{x}_d|\boldsymbol{x}_c)} \left[\log D_\phi(R_d(\boldsymbol{x}_d))\right]$ with enough network capacity. It 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 combined loss function

$$\min_{\varphi} \mathcal{L}_{\text{render}}(\varphi) := \mathcal{L}_2(\varphi) + \lambda \mathcal{L}_{\text{DOM}}(\varphi). \tag{8}$$

We name the proposed method *inverse graphics GAN* (IG-GAN), as the neural renderer in back-propagation time "inverts" the off-the-shelf renderer providing useful gradients for the 3D generative model training. The model, visualised in Figure 1, is trained end-to-end. To speed up the generative model training, the neural renderer can be pretrained on a generic data set, like tables or cubes.

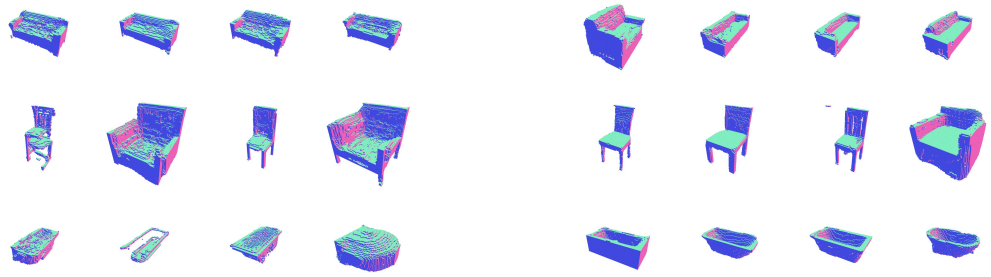
3 Results [2]

Experimental Setup We evaluate our model on synthetic datasets generated from 3D models of the *Chairs, Couches* and *Bathtubs* categories of ShapeNet [2] objects. For each category, we generate

[2] Implementation details, further experiments and ablations studies can be found in the Supplemental Material. Full supplemental material available on `https://figshare.com/s/56084c4d7df8f57f15d9`

# of Images	**One Per Model** (≈ 3000)			**Unlimited**		
Dataset	**Tubs**	**Couches**	**Chairs**	**Tubs**	**Couches**	**Chairs**
2D-DCGAN [23]	461.8	354.3	362.3	226.7	210.9	133.2
Visual Hull [5]	184.6	106.2	37.1	90.1	35.1	15.7
Absorbtion Only [8]	275.8	78.0	32.8	104.5	25.5	23.8
IG-GAN (Ours)	**67.5**	**35.8**	**20.7**	**44.0**	**17.8**	**13.6**

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



(a) Absorption Only

(b) IG-GAN (Ours)

Figure 2: Samples generated by (a) the AO baseline and (b) our model on the 'One per model' setting. 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.

one small data set by sampling a single fixed view point per 3D object ('One per Model') and a second, larger one, by rendering a different viewpoint for the same 3D object at each training epoch ('Unlimited'). We evaluate the quality of the generated 3D models by rendering them to 2D images and computing Fréchet Inception Distances (FIDs) [9], using an Inception network [29] trained to classify ShapeNet Images generated with our renderer. We compare to the visual hull model from Gadelha et al. [5] that uses a smoothed version of object silhouetts for differentiable rendering and against the absorption model from Henzler et al. [8] which assumes voxels absorb light based on their fraction of occupancy.

Quantitative Evaluation We can see in Table 1 that our approach (IG-GAN) significantly outperforms the baselines on all datasets. The largest gain 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 size is restricted. Since our method can more easily take advantage of the lighting and shading cues provided by the images, we believe it can extract more meaningful information per training sample, hence producing better results in these settings. In the Unlimited dataset setting 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.

Qualitative Evaluation We can see in Figure 2 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[3]. Our approach, on the other hand, successfully detects the interior structure of concave objects using the differences in light exposure 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 detect 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 samples with grainy and uneven surfaces. A large selection of randomly generated samples from all methods can be found in the supplemental material.

[3]This shortcoming of the baseline models has already been noticed by Gadelha et al. [5].

References

[1] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019.

[2] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[3] Wenzheng Chen, Huan Ling, Jun Gao, Edward Smith, Jaakko Lehtinen, Alec Jacobson, and Sanja Fidler. Learning to predict 3d objects with an interpolation-based differentiable renderer. In *Advances in Neural Information Processing Systems*, pages 9605–9616, 2019.

[4] SM Ali Eslami, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S Morcos, Marta Garnelo, Avraham Ruderman, Andrei A Rusu, Ivo Danihelka, Karol Gregor, et al. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018.

[5] Matheus Gadelha, Subhransu Maji, and Rui Wang. 3d shape induction from 2d views of multiple objects. In *2017 International Conference on 3D Vision (3DV)*, pages 402–411. IEEE, 2017.

[6] Kyle Genova, Forrester Cole, Aaron Maschinot, Aaron Sarna, Daniel Vlasic, and William T Freeman. Unsupervised training for 3d morphable model regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8377–8386, 2018.

[7] Paul Henderson and Vittorio Ferrari. Learning single-image 3d reconstruction by generative modelling of shape, pose and shading. *International Journal of Computer Vision*, pages 1–20, 2019.

[8] Philipp Henzler, Niloy J. Mitra, and Tobias Ritschel. Escaping plato's cave: 3d shape from adversarial rendering. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[9] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pages 6626–6637, 2017.

[10] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017.

[11] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 371–386, 2018.

[12] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

[13] Diederick P Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.

[14] Yiyi Liao, Simon Donne, and Andreas Geiger. Deep marching cubes: Learning explicit surface representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2916–2925, 2018.

[15] Yiyi Liao, Katja Schwarz, Lars Mescheder, and Andreas Geiger. Towards unsupervised learning of generative models for 3d controllable image synthesis. *arXiv preprint arXiv:1912.05237*, 2019.

[16] Shichen Liu, Shunsuke Saito, Weikai Chen, and Hao Li. Learning to infer implicit surfaces without 3d supervision. In *Advances in Neural Information Processing Systems*, pages 8293–8304, 2019.

[17] Matthew M Loper and Michael J Black. Opendr: An approximate differentiable renderer. In *European Conference on Computer Vision*, pages 154–169. Springer, 2014.

[18] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.

[19] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. *arXiv preprint arXiv:1904.01326*, 2019.

[20] Thu H Nguyen-Phuoc, Chuan Li, Stephen Balaban, and Yongliang Yang. Rendernet: A deep convolutional network for differentiable rendering from 3d shapes. In *Advances in Neural Information Processing Systems 31*, pages 7891–7901, 2018.

[21] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. *arXiv preprint arXiv:1912.07372*, 2019.

[22] Andrea Palazzi, Luca Bergamini, Simone Calderara, and Rita Cucchiara. End-to-end 6-dof object pose estimation through differentiable rasterization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018.

[23] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *International Conference on Learning Representations*, 2016.

[24] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 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, pages II–1278–II–1286, 2014.

[25] Danilo Jimenez Rezende, SM Ali Eslami, Shakir Mohamed, Peter Battaglia, Max Jaderberg, and Nicolas Heess. Unsupervised learning of 3d structure from images. In *Advances in neural information processing systems*, pages 4996–5004, 2016.

[26] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3577–3586, 2017.

[27] Tim Salimans, David A Knowles, et al. Fixed-form variational posterior approximation through stochastic linear regression. *Bayesian Analysis*, 8(4):837–882, 2013.

[28] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Advances in Neural Information Processing Systems*, pages 1119–1130, 2019.

[29] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[30] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[31] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in neural information processing systems*, pages 82–90, 2016.