International Journal of Computer Vision (2020) 128:2651–2664
https://doi.org/10.1007/s11263-020-01335-w

Inferring 3D Shapes from Image Collections Using Adversarial Networks

Matheus Gadelha[1] · **Aartika Rai**[1] · **Subhransu Maji**[1] · **Rui Wang**[1]

Received: 16 May 2019 / Accepted: 25 April 2020 / Published online: 24 June 2020

Abstract
We investigate the problem of learning a probabilistic distribution over three-dimensional shapes given two-dimensional views of multiple objects taken from unknown viewpoints. Our approach called *projective generative adversarial network* (PrGAN) trains a deep generative model of 3D shapes whose projections (or renderings) matches the distribution of the provided 2D views. The addition of a *differentiable projection module* allows us to infer the underlying 3D shape distribution without access to any explicit 3D or viewpoint annotation during the learning phase. We show that our approach produces 3D shapes of comparable quality to GANs trained directly on 3D data. Experiments also show that the disentangled representation of 2D shapes into geometry and viewpoint leads to a good generative model of 2D shapes. The key advantage of our model is that it estimates 3D shape, viewpoint, and generates novel views from an input image in a completely unsupervised manner. We further investigate how the generative models can be improved if additional information such as depth, viewpoint or part segmentations is available at training time. To this end, we present new differentiable projection operators that can be used to learn better 3D generative models. Our experiments show that PrGAN can successfully leverage extra visual cues to create more diverse and accurate shapes.

Keywords 3D generative models · Unsupervised learning · Differentiable rendering · Adversarial networks

1 Introduction

The ability to infer 3D shapes of objects from their 2D views is one of the central challenges in computer vision. For example, when presented with a catalogue of airplane silhouettes as shown in the top of Fig. 1, one can mentally infer their 3D shapes by simultaneously reasoning about the shape and viewpoint variability. In this work, we investigate the problem of learning a generative model of 3D shapes from a collection of images of an unknown set of objects within a category taken from an unknown set of views. The images can be thought of as generalized projections of 3D shapes into a 2D space in the form of silhouettes, depth maps, or even part segmentations. The problem is challenging as one is not provided with the information about which object instance was used to generate each image, the viewpoint from which each image was taken, the parameterization of the underlying shape distribution, or even the number of underlying instances. Hence, traditional techniques based on structure from motion (Hartley and Zisserman 2003; Blanz and Vetter 1999) or visual hulls (Laurentini 1994), cannot be directly applied.

We use the framework of generative adversarial networks (GANs) (Goodfellow et al. 2014) and augment the 3D shape generator with a *projection module*, as illustrated in Fig. 2. The generator produces 3D shapes, the projection module renders the shape from viewpoint sampled from a viewpoint distribution, and the adversarial network discriminates real images from generated ones. The projection module is a *differentiable renderer* that allows us to map 3D shapes to 2D

Communicated by Jun-Yan Zhu, Hongsheng Li, Eli Shechtman, Ming-Yu Liu, Jan Kautz, Antonio Torralba.

✉ Matheus Gadelha
mgadelha@cs.umass.edu

Aartika Rai
aartikarai@cs.umass.edu

Subhransu Maji
smaji@cs.umass.edu

Rui Wang
ruiwang@cs.umass.edu

[1] College of Information and Computer Sciences, University of Massachusetts Amherst, 140 Governors Dr, Amherst, MA 01003, USA

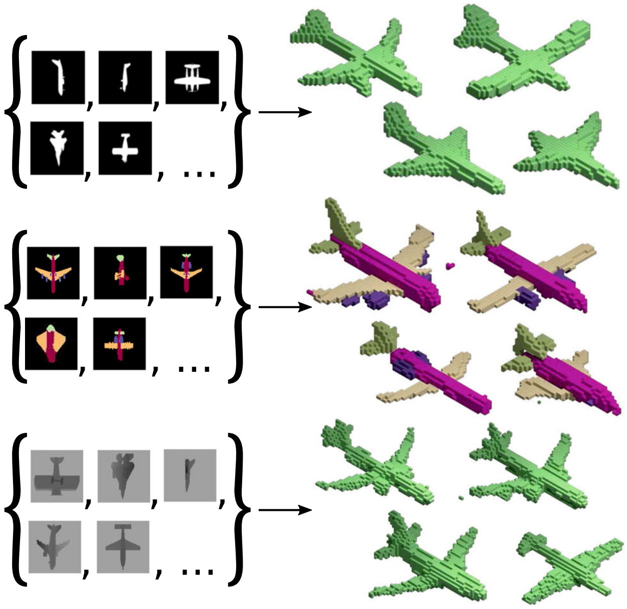


Fig. 1 Our algorithm infers a generative model of the underlying 3D shapes given a collection of unlabeled images rendered as silhouettes, semantic segmentations or depth maps. To the left, images representing the input dataset. To the right, shapes generated by the generative model trained with those images

images, as well as back-propagate the gradients of 2D images to 3D shapes. Once trained, the model can be used to infer 3D shape distributions from a collection of images (Fig. 1 shows some samples drawn from the generator), and to infer depth or viewpoint from a single image, without using any 3D or viewpoint information during learning. We call our approach *projective generative adversarial network* (PrGAN).

While there are several ways of rendering a 3D shape, we begin with a silhouette representation. The motivation is that silhouettes can be easily extracted when objects are photographed against clear backgrounds, such as in catalogue images, but nevertheless they contain rich shape information. Real-world images can also be used by removing background and converting them to binary images. Our generative 3D model represents shapes using a voxel representation that indicates the occupancy of a volume in a fixed-resolution 3D grid. Our projection module is a feed-forward operator that renders the volume as an image. The feed-forward operator is differentiable, providing the ability to adjust the 3D volume based on projections. Finally, we assume that the distribution over viewpoints is known (assumed to be uniform in our experiments, but it could be any distribution).

We then extend our analysis first presented in our earlier work (Gadelha et al. 2017) by incorporating recent advances in training GANs and designing projection modules to incorporate richer supervision. The latter includes the availability of viewpoint information for each image, depth maps instead of silhouettes, or semantic segmentations such as part labels during learning. Such supervision is easier to collect than acquiring full 3D scans of objects. For example, one can use a generic object viewpoint estimator (Su et al. 2015) as weak supervision for our problem. Similarly, semantic parts can be labeled on images directly and already exist for many object categories such as airplanes, birds, faces, and people. We show that such information can be used to improve 3D reconstruction by using an appropriate projection module.

To summarize our main contributions are as follows: (i) we propose PrGAN, a framework to learn probabilistic distributions over 3D shapes from a collection of 2D views of objects. We demonstrate its effectiveness on learning shape categories such as chairs, airplanes, and cars sampled from online shape repositories (Chang et al. 2015; Wu et al. 2015). The results are reasonable even when views from multiple

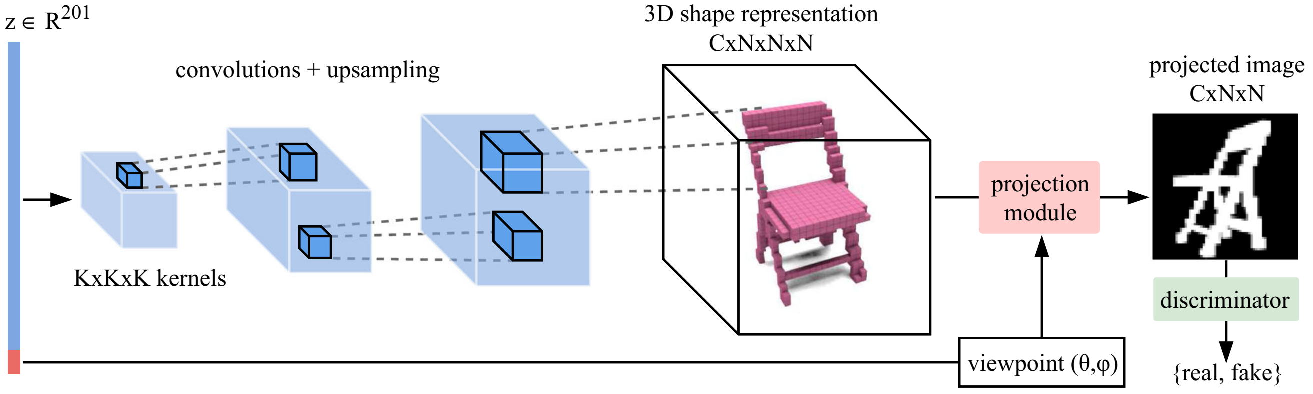


Fig. 2 The PrGAN architecture for generating 2D silhouettes of shapes factorized into a 3D shape generator and viewpoint generator and projection module. A 3D voxel representation (C × N^3) and viewpoint are independently generated from the input z (201-d vector). The projection module renders the voxel shape from a given viewpoint (θ, ϕ) to create an image. The discriminator consists of 2D convolutional and pooling layers and aims to classify if the generated image is "real" or "fake". The number of channels C in the generated shape is equal to one for an occupancy-based representation and is equal to the number of parts for a part-based representation

categories are combined; (ii) PrGAN generates 3D shapes of comparable quality to GANs trained directly on 3D data (Wu et al. 2016); (iii) The learned 3D representation can be used for unsupervised estimation of 3D shape and viewpoint given a novel 2D shape, and for interpolation between two different shapes, (iv) Incorporating additional cues as weak supervision improves the 3D shapes reconstructions in our framework.

2 Related Work

Estimating 3D shape from image collections. The difficulty of estimating 3D shape can vary widely based on how the images are generated and the assumptions one can make about the underlying shapes. Visual-hull techniques (Laurentini 1994) can be used to infer the shape of an object by computing the intersection of the projected silhouettes taken from known viewpoints. When the viewpoint is fixed and the lighting is known, photometric stereo (Woodham 1980) can provide accurate geometry estimates for rigid and diffuse surfaces. Structure from motion (SfM) (Hartley and Zisserman 2003) can be used to estimate the shape of *rigid objects* from their views taken from unknown viewpoints by jointly reasoning about point correspondences and camera projections. Non-rigid SfM can be used to recover shapes from image collections by assuming that the 3D shapes can be represented using a compact parametric model. An early example is that of Blanz and Vetter (1999) for estimating 3D shapes of faces from image collections where each shape is represented as a linear combination of bases (Eigen shapes). However, 3D shapes need to be aligned in a consistent manner to estimate the bases which can be challenging. Recently, non-rigid SfM has been applied to categories such as cars and airplanes by manually annotating a fixed set of keypoints across instances to provide correspondences (Kar et al. 2015). Our work augments non-rigid SfM using a learned 3D shape generator, which allows us to generalize the technique to categories with diverse structures *without* requiring correspondence annotations. Our work is also related to recent work of Kulkarni et al. (2015) for estimating a disentangled representation of images into shape, viewpoint, and lighting variables (dubbed "inverse graphics networks"). However, the shape representation is not explicit, and the approach requires the ability to generate training images while varying one factor at a time.

Inferring 3D shape from a single image. Optimization-based approaches put priors on geometry, material, and light to estimate all of them by minimizing the reconstruction error when rendered (Land and McCann 1971; Barrow and Tenenbaum 1978; Barron and Malik 2015). Our approach on the other hand exploits implicit priors induced by deep networks (Gadelha et al. 2019; Cheng et al. 2019) for generative modeling. Recognition-based methods have been used to estimate geometry of outdoor scenes (Hoiem et al. 2005; Saxena et al. 2005), indoor environments (Eigen and Fergus 2015; Schwing and Urtasun 2012), and objects (Andriluka et al. 2010; Savarese and Fei-Fei 2007). More recently, convolutional networks have been trained to generate views of 3D objects given their attributes and camera parameters (Dosovitskiy et al. 2015), to generate 3D shape given a 2D view of the object (Tatarchenko et al. 2016), and to generate novel views of an object (Zhou et al. 2016). Most of these approaches are trained in a fully-supervised manner and require 3D data or multiple views of the same object during training.

Generative models for images and shapes. Our work builds on the success of GANs for generating images across a wide range of domains (Goodfellow et al. 2014). Recently, Wu et al. (2016) learned a generative model of 3D shapes using GANs equipped with 3D convolutions. However, the model was trained with aligned 3D shape data. Our work aims to solve a more difficult question of learning a 3D-GAN from 2D images. Several recent works are in this direction. Rezende et al. (2016) show results for 3D shape completion for simple shapes when views are provided, but require the viewpoints to be known and the generative models are trained on 3D data. Yan et al. (2016) learn a mapping from an image to 3D using multiple projections of the 3D shape from known viewpoints and object identification, i.e., which images correspond to the same object. Their approach employs a 3D volumetric decoder and optimizes a loss that measures the overlap of the projected volume on the multiple silhouettes from known viewpoints, similar to a visual-hull reconstruction. Tulsiani et al. (2017) learn a model to map images to 3D shape provided with color images or silhouettes of objects taken from known viewpoints using a "ray consistency" approach similar to our projection module. Kanazawa et al. (2018) employs additional supervision in the form of keypoint annotations to generate textured 3D meshes. On the other hand, our method does not assume known viewpoints, object associations of the silhouettes making the problem considerably harder. If object associations are given and viewpoints are unknown, a possible solution is to use multi-view consistency across similar objects, as demonstrated in Tulsiani et al. (2018). More similar to our setup, Henderson and Ferrari (2018) propose a method to learn a generative model of 3D shapes from a set of images without viewpoint supervision. However, their approach uses a more constrained shape representation—sets of blocks or deformations in a subdivided cube—and other visual cues such as lighting configuration and normals.

Differentiable renderers. Our generative models rely on a differentiable projection module to incorporate image-based supervision. Since our images are rendered as silhouettes, the process can be approximated using differentiable functions composed of spatial transformations and projections as described in Sect. 3. However, more sophisticated differentiable renders, such as Kato et al. (2018), Liu et al. (2018), Li et al. (2018), that take into account shading and material properties could provide richer supervision or enable learning from real images. These renderers rely on mesh-based or surface-based representations which are challenging to generate due to their unstructured nature. Recent work on generative models of 3D shapes with point clouds (Lin et al. 2018; Gadelha et al. 2018, 2017; Fan et al. 2017; Groueix et al. 2018; Achlioptas et al. 2017) or multiview (Lun et al. 2017; Tatarchenko et al. 2016) representations provide a possible alternative to our voxel based approach that we aim to investigate in the future.

3 Method

Our method builds upon GANs proposed in Goodfellow et al. (2014). The goal of a GAN is to train a generative model in an adversarial setup. The model consists of two parts: a *generator* and a *discriminator*. The generator G aims to transform samples drawn from a simple distribution $\mathcal{P}$ that appear to have been sampled from the original dataset. The discriminator D aims to distinguish samples generated by the generator from real samples (drawn from a data distribution $\mathcal{D}$). Both the generator and the discriminator are trained jointly by optimizing:

$$\min_G \max_D \mathbb{E}_{x\sim\mathcal{D}}[\log(D(x))] + \mathbb{E}_{z\sim\mathcal{P}}[\log(1 - D(G(z)))]. \tag{1}$$

Our main task is to train a generative model for 3D shapes without relying on 3D data itself, instead relying on 2D images from those shapes, without any view or shape annotation[1]. In other words, the data distribution consists of 2D images taken from different views and are of different objects. To address this mismatch we factorize the 2D image generator into a 3D shape generator ($\mathcal{G}_{3D}$), viewpoint generator (θ, ϕ), and a projection module $\mathcal{P}_{\theta,\phi}$ as seen in Fig. 2. The challenge is to identify a representation for a diverse set of shapes and a differentiable projection module to create final 2D images and enable end-to-end training. We describe the architecture employed for each of these next.

3D shape generator (G_{3D}). The input to the entire generator is $z \in \mathbb{R}^{201}$ with each dimension drawn independently from a uniform distribution $\mathrm{U}(-1, 1)$. Our 3D shape generator $\mathcal{G}_{3D}$ transforms the first 200 dimensions of z to a $N \times N \times N$ voxel representation of the shape. Each voxel contains a value $v \in [0, 1]$ that represents its occupancy. The architecture of the 3D shape generator is inspired by the DCGAN (Radford et al. 2015) and 3D-GAN (Wu et al. 2016) architectures. It consists of several layers of 3D convolutions, upsampling, and non-linearities, as shown in Fig. 2. The first layer transforms the 200 dimensional vector to a $256 \times 4 \times 4 \times 4$ vector using a fully-connected layer. Subsequent layers have batch normalization and ReLU layers between them and use 3D kernels of size $5 \times 5 \times 5$. At every layer, the spatial dimensionality is increased by a factor of 2 and the number of channels is decreased by the same factor, except for the last layer whose output only has one channel (voxel occupancy). The last layer is succeeded by a sigmoid activation instead of a ReLU in order to keep the occupancy values in [0, 1].

Viewpoint generator (θ, ϕ). The viewpoint generator takes the last dimension of $z \in \mathrm{U}(-1, 1)$ and transforms it to a viewpoint vector (θ, ϕ). The training images are assumed to have been generated from 3D models that are upright oriented along the y-axis and are centered at the origin. Most models in online repositories and the real world satisfy this assumption (*e.g.*, chairs are on horizontal planes). We generate images by sampling views uniformly at random from one of eight pre-selected directions evenly spaced around the y-axis (*i.e.*, $\theta = 0$ and $\phi = 0°, 45°, 90°, \ldots, 315°$), as seen in Fig. 3. Thus the viewpoint generator picks one of these directions uniformly at random.

Projection module (Pr). The projection module Pr renders the 3D shape from the given viewpoint to produce an image. For example, a silhouette can be rendered in the following steps. The first step is to rotate the voxel grid to the corresponding viewpoint. Let $V : \mathbb{Z}^3 \rightarrow [0, 1] \in \mathbb{R}$ be the voxel grid, a function that given an integer 3D coordinate $c = (i, j, k)$ returns the occupancy of the voxel centered at c. The rotated version of the voxel grid $V(c)$ is defined as $V_{\theta,\phi} = V(\lfloor R(c, \theta, \phi)\rfloor)$, where $R(c, \theta, \phi)$ is the coordinate obtained by rotating c around the origin according to the spherical angles (θ, ϕ).. Notice that R is straightforwardly implemented as a matrix multiplication and can be extended to model other types of transformations, *e.g.* perspective transformations. Refer to the Appendix A in Yan et al. (2016) for more details.

The second step is to perform the projection to create an image from the rotated voxel grid. This is done by applying the projection operator $Pr((i, j), V) = 1 - e^{-\sum_k V(i,j,k)}$. Intuitively, the operator sums up the voxel occupancy values along each line of sight (assuming orthographic projection), and applies exponential falloff to create a smooth and differentiable function. When there is no voxel along the line of sight, the value is 0; as the number of voxels increases, the value approaches 1. Combined with the rotated version of the voxel grid, we define our final pro-

[1] We later relax this assumption to incorporate extra supervision.

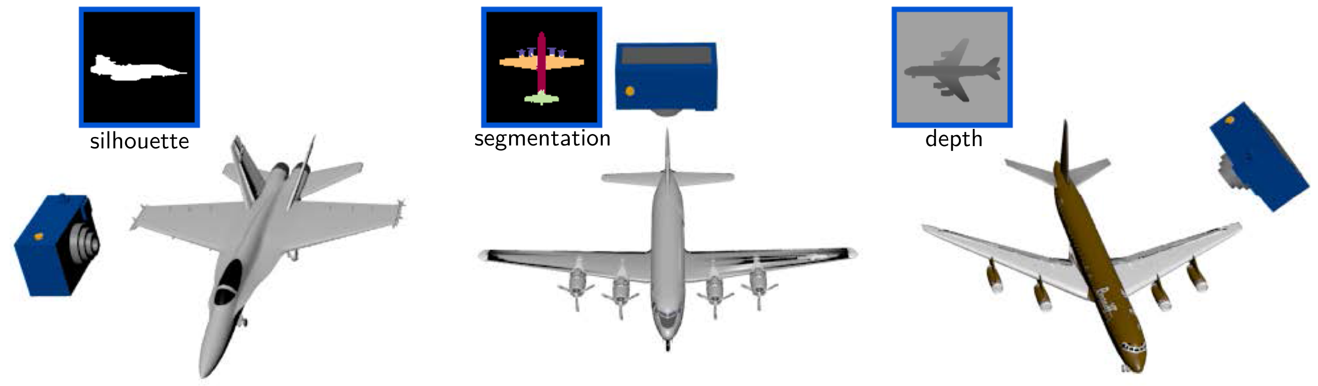


Fig. 3 The input to our model consists of multiple renderings of different objects taken from different viewpoints. Those image are *not* annotated with identification or viewpoint information. Our model is able to handle images from objects rendered as silhouettes (left), semantic segmentation maps (middle) or depth maps (right)

jection module as: $Pr_{\theta,\phi}((i,j),V) = 1 - e^{-\sum_k V_{\theta,\phi}(i,j,k)}$. As seen in Fig. 3 the projection module can well approximate the rendering of a 3D shape as a binary silhouette image, and is differentiable. Section 5 presents projection modules that render the shape as a depth image or one labeled with part segmentations using similar projection operations, as seen in Fig. 3. Thus the 2D image generator G_{2D} can be written compositionally as $G_{2D} = Pr_{(\theta,\phi)} \circ G_{3D}$.

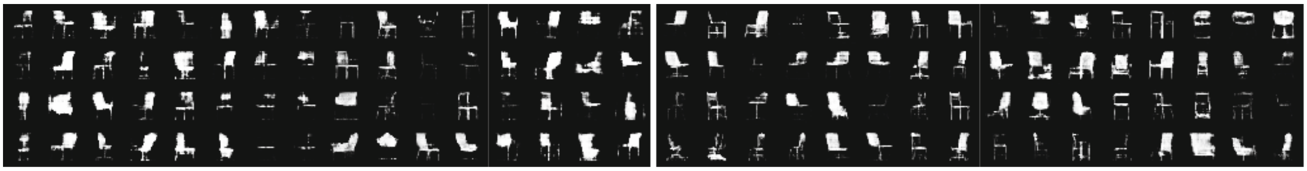
Discriminator (D_{2D}). The discriminator consists of a sequence of 2D convolutional layers with batch normalization layer and LeakyReLU activation (Maas et al. 2013) between them. Inspired by recent work (Radford et al. 2015; Wu et al. 2016), we employ multiple convolutional layers with stride 2 while increasing the number of channels by 2, except for the first layer, whose input has 1 channel (image) and output has 256. Similar to the generator, the last layer of the discriminator is followed by a sigmoid activation instead of a LeakyReLU.

Training details. The entire architecture is trained by optimizing the objective in Equation 1. Usually, updates to minimize each one of the losses is applied once at each iteration. However, in our model, the generator and the discriminator have a considerably different number of parameters, as the generator is trying to create 3D shapes, while the discriminator is trying to classify 2D images. To mitigate this issue, we employ an adaptive training strategy. At each iteration of the training, if the discriminator accuracy is higher than 75%, we skip its training. We also set different learning rates for the discriminator and the generator: 10^{-5} and 0.0025, respectively. Similarly to the DCGAN architecture (Radford et al. 2015), we use ADAM with $\beta_1 = 0.5$ for the optimization.

4 Experiments

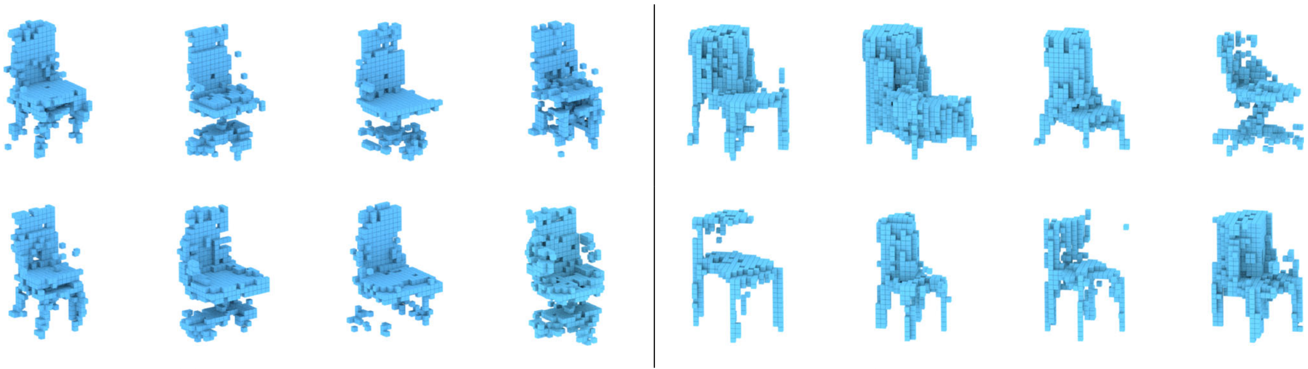
This section describes a set of experiments to evaluate our basic method and several extensions. First, we compare our model with a traditional GAN for the task of image generation and a GAN for 3D shapes. We present quantitative and qualitative results. Second, we demonstrate that our method is able to induce 3D shapes from unlabeled images even when the collection contains only a single view per object. Third, we present 3D shapes induced by our model from a variety of categories such as airplanes, cars, chairs, motorbikes, and vases. Using the same architecture, we show how our model is able to induce coherent 3D shapes when the training data contains images mixed from multiple categories. Finally, we show applications of our method in predicting 3D shape from a novel 2D shape, and performing shape interpolation.

Input data. We generate training images synthetically using 3D shapes available in the ModelNet (Wu et al. 2015) and ShapeNet (Chang et al. 2015) databases. Each category contains a few hundred to thousand shapes. We render each shape from 8 evenly spaced viewing angles with orthographic projection to produce binary images. Hence our assumption is that the viewpoints of the training images (which are unknown to the network) are uniformly distributed. If we have prior knowledge about the viewpoint distribution (e.g. there may be more frontal views than side views), we can adjust the projection module to incorporate this knowledge. To reduce aliasing, we render each image at 64 × 64 resolution and downsample to 32 × 32. We have found that this generally improves the results. Using synthetic data allows us to easily perform controlled experiments to analyze our method. It is also possible to use real images downloaded from a search engine as discussed in Sect. 5.



(a) Results from 2D-GAN. **(a)** Results from PRGAN.

Fig. 4 Comparison between 2D-GAN (Goodfellow et al. 2014) and our PRGAN model for image generation on the chairs dataset. Refer to Fig. 9 third row, left column for samples of the input data



(a) Results from 3D-GAN. **(a)** Results from PRGAN.

Fig. 5 Comparison between 3D-GAN (Wu et al. 2016) and our PRGAN for 3D shape generation. The 3D-GAN is trained on 3D voxel representation of the chair models, and the PRGAN is trained on images of the chair models (refer to Fig. 9 third row)

4.1 Results

We quantitatively evaluate our model by comparing its ability to generate 2D and 3D shapes. To do so, we use 2D image GAN similar to DCGAN (Radford et al. 2015) and a 3D-GAN similar to the one presented in Wu et al. (2016). At the time of this writing the implementation of Wu et al. (2016) is not public yet, therefore we implemented our own version. We will refer to them as 2D-GAN and 3D-GAN, respectively. The 2D-GAN has the same discriminator architecture as the PRGAN, but the generator contains a sequence of 2D transposed convolutions instead of 3D ones, and the projection module is removed. The 3D-GAN has a discriminator with 3D convolutions instead of 3D ones. The 3D-GAN generator is the same as the PRGAN, but without the projection module.

The models used in this experiment are chairs from ModelNet dataset (Wu et al. 2015). From those models, we create two sets of training data: voxel grids and images. The voxel grids are generated by densely sampling the surface and inside of each mesh, and binning the sample points into 32 × 32 × 32 grid. A value 1 is assigned to any voxel that contains at least one sample point, and 0 otherwise. Notice that the voxel grids are only used to train the 3D-GAN, while the images are used to train the 2D-GAN and our PRGAN.

Our quantitative evaluation is done by taking the Maximum Mean Discrepancy (MMD) (Gretton et al. 2006) between the data created by the generative models and the training data. We use a kernel bandwidth of 10^{-3} for images and 10^{-2} for voxel grids. The training data consists of 989 voxel grids and 7912 images. To compute the MMD, we draw 128 random data points from each one of the generative models. The distance metric between the data points is the hamming distance divided by the dimensionality of the data. Because the data represents continuous occupancy values, we binarize them by using a threshold of 0.001 for images or voxels created by PRGAN, and 0.1 for voxels created by the 3D-GAN.

Results show that for 2D-GAN, the MMD between the generated images and the training data is **90.13**. For PRGAN, the MMD is **88.31**, which is slightly better quantitatively than 2D-GAN. Figure 4 shows a qualitative comparison. The results are visually very similar. For 3D-GAN, the MMD between the generated voxel grids and the training voxel grids is **347.55**. For PRGAN, the MMD is **442.98**, which is worse compared to 3D-GAN. This is not surprising as 3D-GAN is trained on 3D data, while PRGAN is trained on the image views only. Figure 5 presents a qualitative comparison. In general PRGAN has trouble generating interior structures because the training images are binary, carry no shading information, and are taken from a limited set of viewing angles. Nonetheless, it learns to generate exterior structures reasonably well.

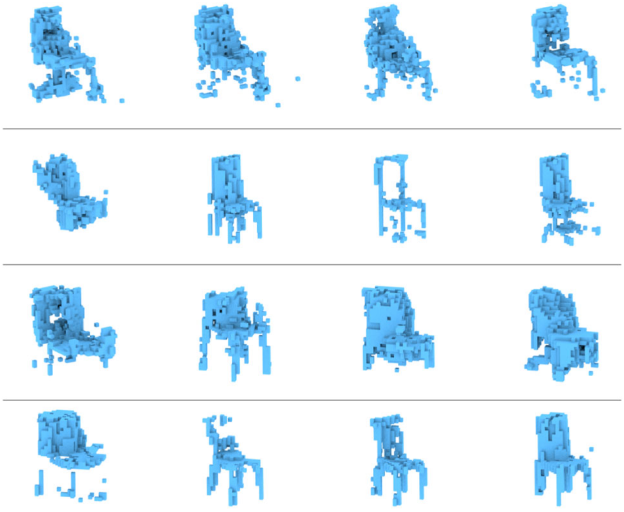


Fig. 6 Shapes generated from PrGAN by varying the number of views per object in the training data. From the top row to the bottom row, the number of views per object in the training set are 1, 2, 4, and 8 respectively

4.1.1 Varying the Number of Views per Model

In the default setting, our training data is generated by sampling 8 views per object. Note that we do not provide the association between views and instances to the generator. Here we study the ability of our method in the more challenging case where the training data contains fewer number of views per object. To do so, we generate a new training set that contains only 1 randomly chosen view per object and use it to train PrGAN. We then repeat the experiments for 2 randomly chosen views per object, and also 4. The results are shown in Fig. 6. Notice that the 3D shapes generated by PrGAN become slightly better as the number of views increase. An interesting question is what's the root cause for such improvements—it may be due to the fact that more training data is available as the number of views per object increases; or it could be that the presence of multiple views of the same object lead to better reconstruction. Thus we further investigate this question by performing an additional experiment, where the training data consists of 8 views per instance, but only using half of the instances available in the dataset. In other words, this setup has the same number of images as the experiment with 4 views of all instances, which makes them comparable in terms of the total amount of training data. We observed no qualitative or quantitative difference in the objects generated in these two scenarios. Quantitative results using the model and metrics described in Sect. 5 are shown in Table 1. Therefore, we believe the improved quality is most likely a consequence of extra data available during training. Nevertheless, it is important to highlight that differently from other approaches that require object correspondence (Yan et al. 2016; Tulsiani et al. 2018) our method is able to induce reasonable shapes, even in the case of a single view per object.

4.1.2 Shape Interpolation

Once the generator is trained, any encoding z supposedly generates a plausible 3D shape, hence z represents a 3D shape manifold. Similar to previous work, we can interpolate between 3D shapes by linearly interpolating their z codes. Figure 7 shows the interpolation results for two airplane models and two chair models.

4.1.3 Unsupervised Shape and Viewpoint Prediction

Our method is also able to handle unsupervised prediction of shapes in 2D images. Once trained, the 3D shape generator is capable of creating shapes from a set of encodings $z \in \mathbb{R}^{201}$. One application is to predict the encoding of the underlying 3D object given a single view image of the object. We do so by using the PrGAN's generator to produce a large number of encoding-image pairs, then use the data to train a neural network (called encoding network). In other words, we cre-

Table 1 Quantitative comparison between models trained with different projection operators

Model	Supervision	$\mathcal{D} \rightarrow G(z)$	$G(z) \rightarrow \mathcal{D}$	Avg.
PrGAN*	Silhouette	0.431	0.391	0.411
PrGAN†	Silhouette	0.429	0.391	0.410
PrGAN	Silhouette	0.442	0.400	0.421
PrGAN	Silhouette + View	0.439	0.431	0.435
PrGAN	Depth	0.497	0.448	0.472
PrGAN	Part Segmentation	0.496	0.507	0.502
3D-GAN	Volumetric	0.538	0.530	0.534

The Chamfer similarity under the volumetric intersection over union (IoU) is shown for PrGAN trained with varying amounts of supervision and a 3D-GAN trained with volumetric supervision. The metric (higher the better) indicates that PrGAN with richer supervision are better and approaches the quality of 3D-GAN. PrGAN* is trained using only 4 out of 8 views per object. PrGAN† is trained using all 8 views but for half of the objects

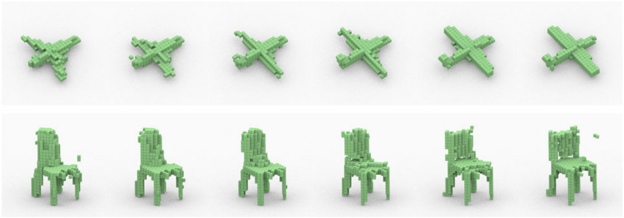


Fig. 7 Shape interpolation by linearly interpolating the encodings of the starting shape and ending shape

ate a training set that consists of images synthesized by the PrGAN and the encodings that generated them. The encoding network is fully connected, with 2 hidden layers, each with 512 neurons. The input of the network is an image and the output is an encoding. The last dimension of z describes the view, and the first 200 dimensions describe the code of the shape, which allows us to further reconstruct the 3D shape as a 32^3 voxel grid. With the encoding network, we can present to it a single view image, and it outputs the shape code along with the viewing angle. Experimental results are shown in in Fig. 8. This whole process constitutes a completely unsupervised approach to creating a model that infers a 3D shape from a single image.

4.1.4 Visualizations Across Categories

Our method is able to generate 3D shapes for a wide range of categories. Figure 9 show a gallery of results, including airplanes, car, chairs, vases, motorbikes. For each category we show 64 randomly sampled training images, 64 generated images from PrGAN, and renderings of 128 generated 3D shapes (produced by randomly sampling the 200-d input vector of the generator). One remarkable property is that the generator produces 3D shapes in a consistent horizontal and vertical axes, even though the training data is only consistently oriented along the vertical axis. Our hypothesis for this is that the generator finds it more efficient to generate shapes in a consistent manner by sharing parts across models. Figure 10 shows selected examples from Fig. 9 that demonstrates the quality and diversity of the generated shapes.

The last row in Fig. 9 shows an example of a “mixed” category, where the training images combine the three categories of airplane, car, and motorbike. The same PrGAN network is used to learn the shape distributions. Results show that PrGAN learns to represent all three categories well, without any additional supervision.

4.2 Failure Cases

Compared to 3D-GANs, the proposed PrGAN models cannot discover structures that are hidden due to occlusions from all views. For example, it fails to discover that some chairs have concave interiors and the generator simply fills these since it does not change the silhouette from any view as we can see at Fig. 11. However, this is a natural drawback of view-based approaches since some 3D ambiguities cannot be resolved (*e.g.*, Necker cubes) without relying on other cues. Despite this, one advantage over 3D-GAN is that our model does not require consistently aligned 3D shapes since it reasons over viewpoints.

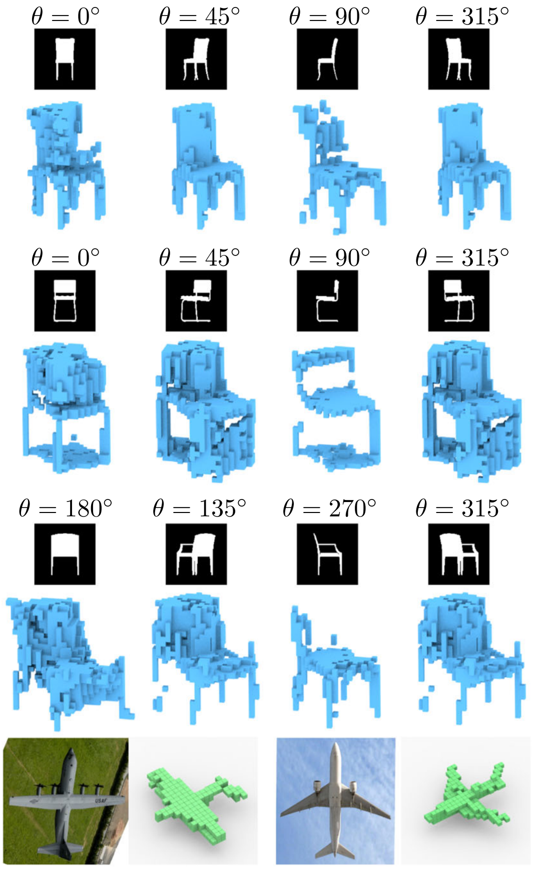


Fig. 8 At top 3 rows, the four images are different views of the same chair, with predicted viewpoint on the top. Shapes are different but plausible given the single view. In the bottom row, shape inferred (right) by a single view image (left) using the encoding network. Input images were segmented, binarized and resized to match the network input

5 Improving PrGAN with Richer Supervision

This section shows how the generative models can be improved to support higher resolution 3D shapes and by incorporating richer forms of view-based supervision.

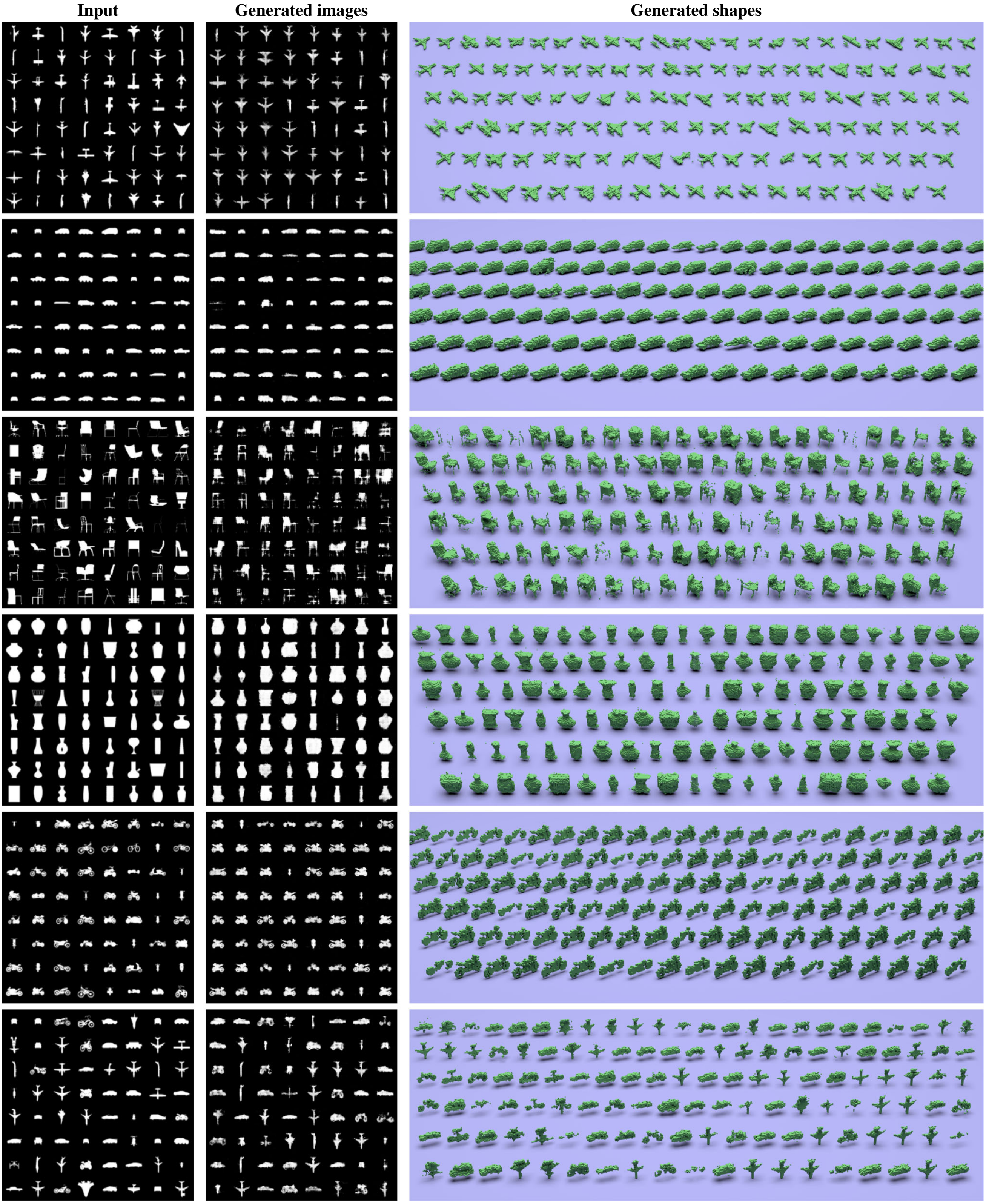


Fig. 9 Results for 3D shape induction using PrGANs. From top to bottom we show results for airplane, car, chair, vase, motorbike, and a 'mixed' category obtained by combining training images from airplane, car, and motorbike. In each row, we show on the left 64 randomly sampled images from the input data to the algorithm, on the right 128 sampled 3D shapes from PrGAN, and in the middle 64 sampled images after the projection module is applied to the generated 3D shapes. The model is able to induce a rich 3D shape distribution for each category. The mixed-category produces reasonable 3D shapes across all three combined categories. Zoom in to see details

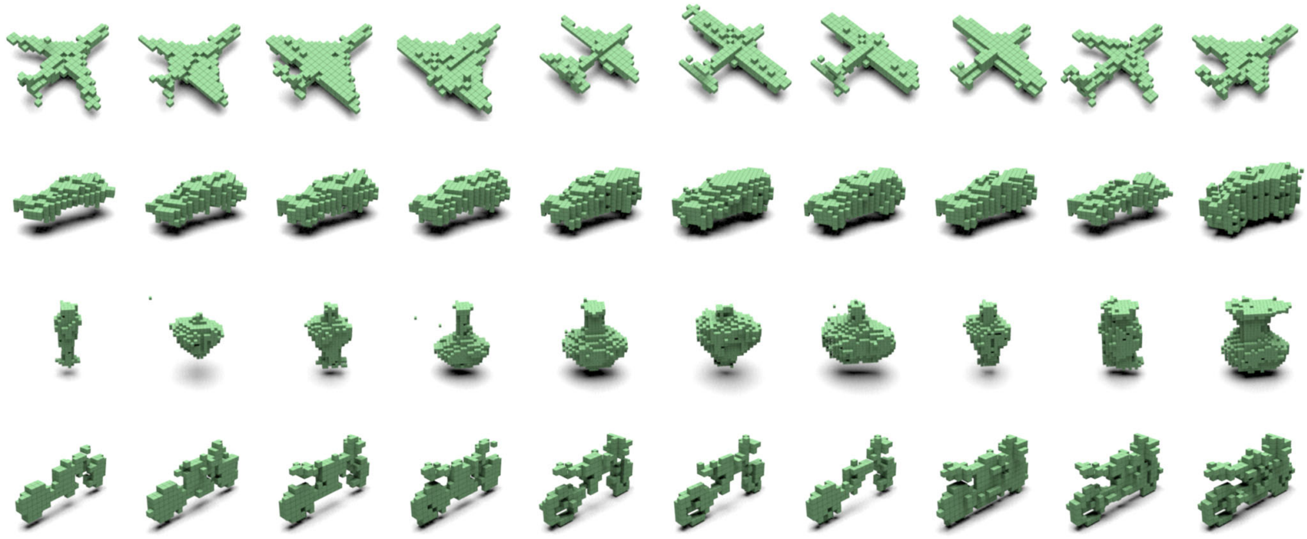


Fig. 10 A variety of 3D shapes generated by PRGAN trained on 2D views of (from the top row to the bottom row) airplanes, cars, vases, and bikes. These examples are chosen from the gallery in Fig. 9 and demonstrate the quality and diversity of the generated shapes

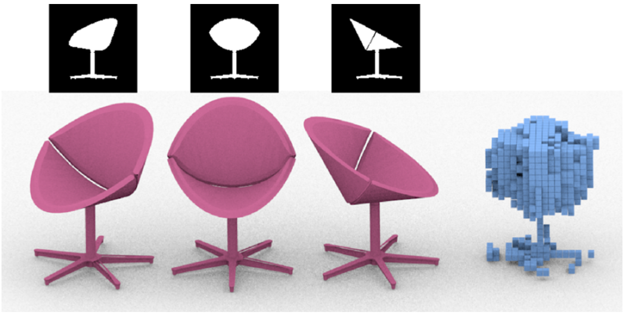


Fig. 11 Our method is unable to capture the concave interior structures in this chair shape. The pink shapes show the original shape used to generate the projected training data, shown by the three binary images on the top (in high resolution). The blue voxel representation is the inferred shape by our model. Notice the lack of internal structure

5.1 Higher-Resolution Models

We extend the vanilla PRGAN model to handle higher resolution volumes. There are two key modifications. First, we replace the transposed convolutions in the generator by trilinear upsampling followed by a 3D convolutional layer. In our experiments, we noticed that this modification led to smoother shapes with less artifacts. This fact was also verified for image generators (Odena et al. 2016). Second, we add a feature matching component to the generator objective. This component acts by minimizing the difference between features computed by the discriminator from real and fake images. More precisely, the feature matching loss can be defined as:

$$\mathcal{L}_{FM}(G, D) = \left\| \mathbb{E}_{x\sim\mathcal{D}}[D_k(x)] - \mathbb{E}_{z\sim\mathcal{N}(0,I)}[D_k(G(z))] \right\|_2^2 \tag{2}$$

where $D_k(x)$ are the features from the kth layer of the discriminator when given an input x. In our experiments we define k to be the last convolutional layer of the discriminator. We empirically verified that this component promotes diversity in the generated samples and makes the training more stable.

5.2 Using Multiple Cues for Shape Reasoning

So far our approach only relies on binary silhouettes for estimating the shape, which contributes to the lack of geometric details. One strategy is replace the projection module with a differentiable function, *e.g.*, a convolutional network, to approximate a sophisticated rendering pipeline, like the one presented in Nguyen-Phuoc et al. (2018), Nalbach et al. (2016). Such a *neural renderer* could be a plug-in replacement for the *projection module* in the PRGAN framework. This would provide the ability to use collections of realistically-shaded images for inferring probabilistic models of 3D shapes and other properties.

We explore an alternate direction using differentiable projection operators that do not rely on training procedures. This choice fits well in the PRGAN formulation as it does not rely on 3D supervision for training any part of the model. In this section, we present differentiable operators to render depth images and semantic segmentation maps. We demonstrate that the extra supervision enables generating more accurate 3D shapes and allows relaxing the prior assumption on viewpoint distribution.

Learning from depth images. Our framework can be adapted to learn from depth images instead of binary images.

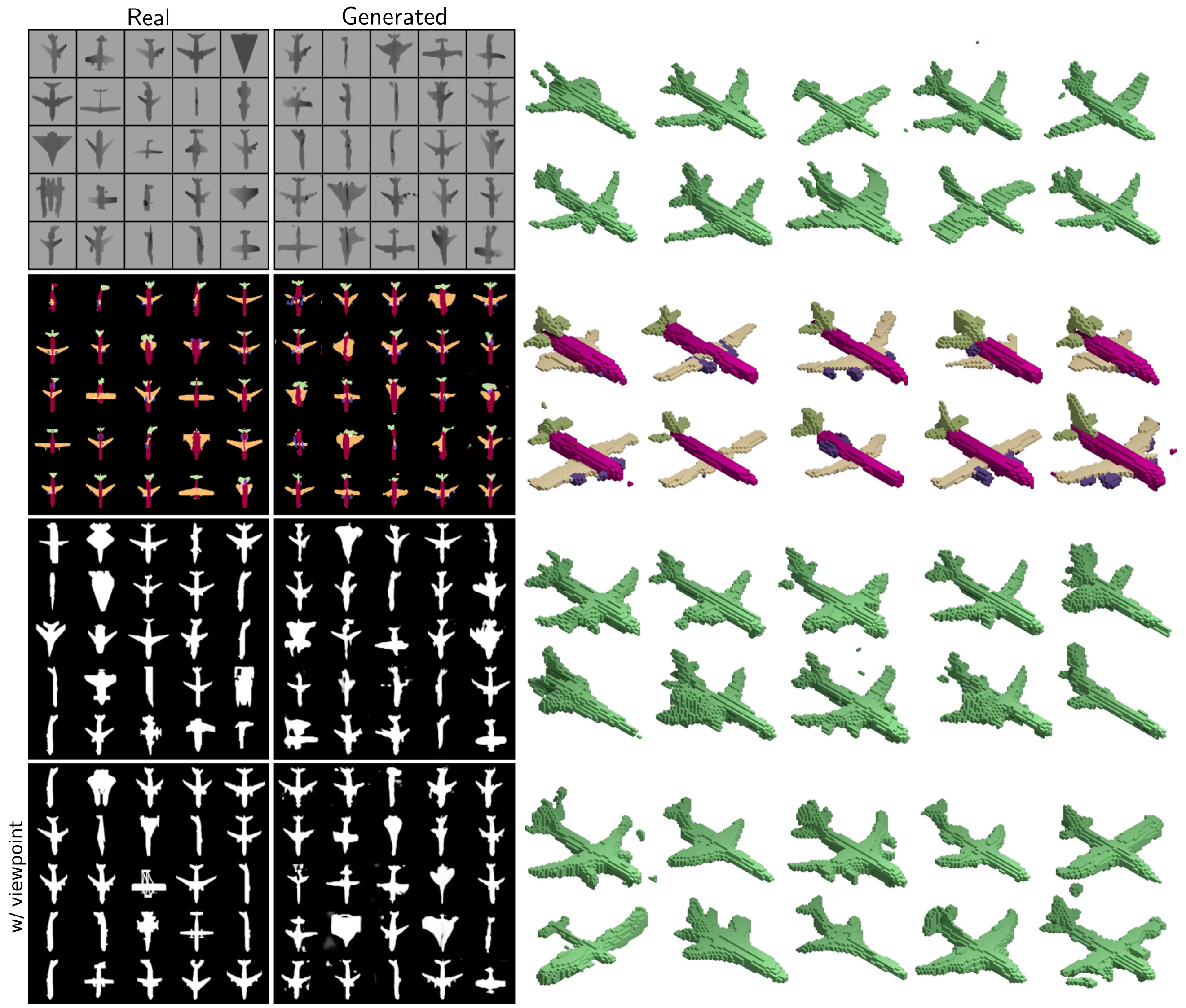


Fig. 12 Shapes generated using new part segmentations and depth maps. From top to bottom, results using depth images, images with part segmentation, silhouettes and silhouettes annotated with viewpoints. Models trained with images containing additional visual cues are able to generated more accurate shapes. Similarly, viewpoint annotation also helps. Notice that shapes generated from images with part annotation are able to generate part-annotated 3D shapes, highlighted by different colors

This is done by replacing the binary projection operator Pr to one that can be used to generate depth images. We follow an approach inspired by the binary projection. First, we define an accessibility function $A(V, \phi, c)$ that describes whether a given voxel c inside the grid V is visible, when seen from a view ϕ:

$$A(V, \phi, i, j, k) = \exp\left\{ -\tau \sum_{l=1}^{k-1} V_\phi(i, j, l) \right\}. \quad (3)$$

Intuitively, we are incrementally accumulating the occupancy (from the first voxel on the line of sight) as we traverse the voxel grid instead of summing all voxels on the entire the line of sight. If voxels on the path from the first to the current voxel are all empty, the value of A is 1 (indicating the current voxel is "accessible" to the view ϕ). If there is at least one non-empty voxel on the path, the value of A will be close to 0 (indicating this voxel is inaccessible). A similar approach was used in our earlier work (Gadelha et al. 2019).

Using A, we can define the depth value of a pixel in the projected image as the line integral of A along the line of sight: $Pr^D_\phi(i, j, V) = \sum_k A(V, \phi, i, j, k)$. This operation computes the number of accessible voxels from a particular direction ϕ, which corresponds to the distance of the surface seen in (i, j) to the camera. Finally, we apply a smooth map to the previous operation in order to have depth values in the range [0, 1]. Thus, the projection module is defined as:

$$Pr^D_\phi((i,j),V) = 1 - \exp\left\{-\sum_k A(V,\phi,i,j,k)\right\}. \tag{4}$$

Learning from part segmentations. We also explore learning 3D shapes from sets of images with dense semantic annotation. Similarly to the depth projection, we modify our projection operator to enable generation of images whose pixels correspond to the label of particular class (or none if there is no object). In this case, the output of the generator is multi-channel voxel grid $V : \mathbb{Z}^3 \times C \rightarrow [0,1] \in \mathbb{R}$, where C is the number of parts present in a particular object category.

Let G to be the aggregated occupancy grid defined as $G = \sum_{c=1}^{C} V(i,j,k,c)$. The semantic projection operator $Pr^S_\phi((i,j,c),V)$ is defined as:

$$Pr^S_\phi((i,j,c),V) = 1 - \exp\left\{\sum_k V_\phi(i,j,k,c) A(G_\phi,i,j,k)\right\}, \tag{5}$$

where A is the accessibility operator defined previously. Intuitively, $A(G,\phi)$ encodes if a particular voxel is visible from a viewpoint ϕ. When we multiply the visibility computed with the aggregated occupancy grid by the value of a specific channel c in V, we generate a volume that contains visibility information per part. Finally, we take the line integral along the line of sight to generate the final image. Examples of images and shapes generated by this operator can be seen in Fig. 12.

Learning with viewpoint annotation. We also experiment with the less challenging setup where our model has access to viewpoint information for every training image. Notice that this problem is different from Kato et al. (2018), Yan et al. (2016), since we still do not know which images correspond to the same object. Thus, multi-view losses are not a viable alternative. Our model is able to leverage viewpoint annotation by using conditional discriminators. The conditional discriminator has the same architecture as the vanilla discriminator but the input image is modified to contain its corresponding viewpoint annotation. This annotation is represented by an one-hot encoding concatenated to every pixel in the image. For example, if a binary image from a dataset with shapes rendered from 8 viewpoints will be represented as a 9-channel image. This procedure is done for images generated by our generator and images coming from the dataset.

5.3 Experiments

Setup. We generate training images using airplanes from the ShapeNet part segmentation dataset (Chang et al. 2015). Those shapes have their surface densely annotated as belonging to one of four parts: body, wing, tail or engine. We render those shapes using the same viewpoint configuration described in Sect. 4. However, in this scenario we use 64 × 64 images instead of 32 × 32. The models are rendered as binary silhouettes, depth maps and part segmentation masks. We train a high resolution PrGAN model for every set of rendered images using the corresponding projection operator. Each model is trained for 50 epochs and trained with the Adam optimizer. We use a learning rate of 2.5×10^{-3} for the generator and 2×10^{-5} for the discriminator.

Evaluation. The models trained with different visual clues are evaluated through the following metric:

$$\frac{1}{|\mathcal{D}|}\sum_{x\in\mathcal{D}} \min_{g\in\mathcal{G}} IoU(x,g) + \frac{1}{|\mathcal{G}|}\sum_{g\in\mathcal{G}} \min_{x\in\mathcal{D}} IoU(x,g) \tag{6}$$

where IoU corresponds to intersection over union, $\mathcal{G}$ is a set of generated shapes and $\mathcal{D}$ is a set of shapes from the training data. In our setup, both $\mathcal{G}$ and $\mathcal{D}$ contain 512 shapes. Shapes in $\mathcal{D}$ are randomly sampled from the same dataset that originated the images, whereas shapes in $\mathcal{G}$ are generated through $G(z)$. Noticeably, the shapes generated by PrGAN do not have the same orientation as the shapes in $\mathcal{D}$ but are consistently oriented among themselves. Thus, before computing Equation 6, we select one of 8 possible transformations that minimizes IoU—there are 8 rendering viewpoints in the training set. Additionally, the components in Equation 6 indicate two different aspects: the first term ($\mathcal{D} \rightarrow G(z)$) indicates how the variety in the dataset is covered whereas the second term ($G(z) \rightarrow \mathcal{D}$) indicates how accurate the generated shapes are. A comparison between models trained with different projection operators can be seen in Table 1. The model trained with part segmentation clues yields the best results. As expected, using only silhouettes leads to worse results in both metrics and adding viewpoint supervision improves upon this baseline. Interestingly, depth and part segmentation supervision clues lead to models that generate shapes with similar variety (similar $\mathcal{D} \rightarrow G(z)$). However, shapes generated from models using part segmentation clues are more similar to the ones in the dataset (higher $G(z) \rightarrow \mathcal{D}$).

6 Conclusion and Future Work

We proposed a framework for inferring 3D shape distributions from 2D shape collections by augmenting a convnet-based 3D shape generator with a projection module. This complements existing approaches for non-rigid SfM since these models can be trained without prior knowledge about the shape family, and can generalize to categories with variable structure. We showed that our models can infer 3D shapes for a wide range of categories, and can be used to infer shape and viewpoint from a silhouettes in a completely unsupervised manner. We believe that the idea of using a differentiable render to infer distributions over unobserved scene properties from images can be applied to other problems.

A limitation is that our approach cannot directly learn from real-world images, as they usually have background pixels, and contain complex shading. In the future, our method can be extended to accommodate real images using semantic segmentation to extract foreground object from the background. In addition, it is possible to incorporate photorealistic differentiable rendering modules capable of handling richer surface colors, materials, and camera parameters. One could also incorporate other forms of supervision, such as viewpoint or coarse shape estimates, to improve the 3D shape inference. For example, camera parameters can be estimated using a generic viewpoint estimator (Tulsiani et al. 2015; Su et al. 2015).

Acknowledgements This research was supported in part by the NSF Grants 1617917, 1749833, 1661259 and 1908669. The experiments were performed using equipment obtained under a grant from the Collaborative R&D Fund managed by the Massachusetts Tech Collaborative.

References

Achlioptas, P., Diamanti, O., Mitliagkas, I., & Guibas, L. J. (2017). Learning representations and generative models for 3d point clouds. arXiv preprint arXiv:1707.02392.

Andriluka, M., Roth, S., & Schiele, B. (2010). Monocular 3D pose estimation and tracking by detection. In *Computer vision and pattern recognition (CVPR)*. IEEE.

Barron, J. T., & Malik, J. (2015). Shape, illumination, and reflectance from shading. *Transactions of Pattern Analysis and Machine Intelligence (PAMI)*, *37*, 1670–1687.

Barrow, H., & Tenenbaum, J. (1978). Recovering intrinsic scene characteristics. In A. Hanson & E. Riseman (Eds.), *Comput. vis. syst.* (pp. 3–26).

Blanz, V., & Vetter, T. (1999). A morphable model for the synthesis of 3d faces. In *Proceedings of the 26th annual conference on computer graphics and interactive techniques* (pp. 187–194). ACM Press/Addison-Wesley Publishing Co.

Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., & Su, H., et al. (2015). Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012.

Cheng, Z., Gadelha, M., Maji, S., & Sheldon, D. (2019). A Bayesian perspective on the deep image prior. In *The IEEE conference on computer vision and pattern recognition (CVPR)*.

Dosovitskiy, A., Tobias Springenberg, J., & Brox, T. (2015). Learning to generate chairs with convolutional neural networks. In *Conference on computer vision and pattern recognition (CVPR)*.

Eigen, D., & Fergus, R. (2015). Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *International conference on computer vision (ICCV)*.

Fan, H., Su, H., & Guibas, L. J. (2017). A point set generation network for 3D object reconstruction from a single image. In *Computer vision and pattern recognition (CVPR)*.

Gadelha, M., Maji, S., & Wang, R. (2017). 3D shape generation using spatially ordered point clouds. In *British machine vision conference (BMVC)*.

Gadelha, M., Maji, S., & Wang, R. (2017). 3D shape induction from 2D views of multiple objects. In *International conference on 3D vision (3DV)*.

Gadelha, M., Wang, R., & Maji, S. (2018). Multiresolution tree networks for 3D point cloud processing. In *European conference on computer vision (ECCV)*.

Gadelha, M., Wang, R., & Maji, S. (2019). Shape reconstruction using differentiable projections and deep priors. In *International conference on computer vision (ICCV)*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems (NIPS)*.

Gretton, A., Borgwardt, K. M., Rasch, M., Schölkopf, B., & Smola, A. J. (2006). A kernel method for the two-sample-problem. In *Advances in neural information processing systems (NIPS)*.

Groueix, T., Fisher, M., Kim, V. G., Russell, B., & Aubry, M. (2018). AtlasNet: A Papier-Mâché approach to learning 3D surface generation. In *Computer vision and pattern recognition (CVPR)*.

Hartley, R., & Zisserman, A. (2003). *Multiple view geometry in computer vision*. Cambridge: Cambridge University Press.

Henderson, P., & Ferrari, V. (2018). Learning to generate and reconstruct 3D meshes with only 2D supervision. In *British machine vision conference (BMVC)*.

Hoiem, D., Efros, A. A., & Hebert, M. (2005). Geometric context from a single image. In *International conference on computer vision (ICCV)*.

Kanazawa, A., Tulsiani, S., Efros, A. A., & Malik, J. (2018). Learning category-specific mesh reconstruction from image collections. In *European conference on computer vision (ECCV)*.

Kar, A., Tulsiani, S., Carreira, J., & Malik, J. (2015). Category-specific object reconstruction from a single image. In *Computer vision and pattern recognition (CVPR)*.

Kato, H., Ushiku, Y., & Harada, T. (2018). Neural 3d mesh renderer. In *Computer vision and pattern recognition (CVPR)*.

Kulkarni, T. D., Whitney, W. F., Kohli, P., & Tenenbaum, J. (2015). Deep convolutional inverse graphics network. In *Advances in neural information processing systems (NIPS)*.

Land, E. H., & McCann, J. J. (1971). Lightness and retinex theory. *JOSA*, *61*(1), 1–11.

Laurentini, A. (1994). The visual hull concept for silhouette-based image understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *16*(2), 150–162.

Li, T.-M., Aittala, M., Durand, F., & Lehtinen, J. (2018). Differentiable monte carlo ray tracing through edge sampling. *ACM Transactions on Graph (SIGGRAPH Asia)*, *37*, 1–11.

Lin, C.-H., Kong, C., & Lucey, S. (2018). Learning efficient point cloud generation for dense 3D object reconstruction. In *AAAI conference on artificial intelligence (AAAI)*.

Liu, H. T. D., Tao, M., & Jacobson, A. (2018). Paparazzi: Surface editing by way of multi-view image processing. *ACM Transactions on Graphcs*, *37*, 221.

Lun, Z., Gadelha, M., Kalogerakis, E., Maji, S., & Wang, R. (2017). 3D shape reconstruction from sketches via multi-view convolutional networks. In *International conference on 3D vision (3DV)* (pp. 67–77).

Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. In *International conference on machine learning (ICML)*.

Nalbach, O., Arabadzhiyska, E., Mehta, D., Seidel, H.-P., & Ritschel, T. (2016). Deep shading: Convolutional neural networks for screen-space shading. arXiv preprint arXiv:1603.06078.

Nguyen-Phuoc, T., Li, C., Balaban, S., & Yang, Y.-L. (2018). Rendernet: A deep convolutional network for differentiable rendering from 3d shapes. In *Advances in neural information processing systems 31*.

Odena, A., Dumoulin, V., & Olah, C. (2016). Deconvolution and checkerboard artifacts. *Distill*.

Radford, A., Metz, L., & Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434.

Rezende, D. J., Eslami, S. M., Mohamed, S., Battaglia, P., Jaderberg, M., & Heess, N. (2016). Unsupervised learning of 3D structure from images. In *Advances in neural information processing systems (NIPS)*.

Savarese, S., & Fei-Fei, L. (2007). 3D generic object categorization, localization and pose estimation. In *International conference on computer vision (ICCV)*.

Saxena, A., Chung, S. H., & Ng, A. (2005). Learning depth from single monocular images. In *Advances in neural information processing systems (NIPS)*.

Schwing, A. G., & Urtasun, R. (2012). Efficient exact inference for 3d indoor scene understanding. In *European conference on computer vision (ECCV)*.

Su, H., Qi, C. R., Li, Y., & Guibas, L. J. (2015). Render for CNN: Viewpoint estimation in images using cnns trained with rendered 3D model views. In *International conference on computer vision (ICCV)*.

Tatarchenko, M., Dosovitskiy, A., & Brox, T. (2016). Multi-view 3D models from single images with a convolutional network. In *European conference on computer vision (ECCV)*.

Tulsiani, S., Carreira, J., & Malik, J.. (2015). Pose induction for novel object categories. In *International conference on computer vision (ICCV)*.

Tulsiani, S., Efros, A. A., & Malik, J. (2018). Multi-view consistency as supervisory signal for learning shape and pose prediction. In *Computer vision and pattern regognition (CVPR)*.

Tulsiani, S., Zhou, T., Efros, A. A., & Malik, J. (2017). Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Computer vision and pattern regognition (CVPR)*.

Woodham, R. J. (1980). Photometric method for determining surface orientation from multiple images. *Optical Engineering*, *19*(1), 191139–191139.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., & Xiao, J. (2015). 3d shapenets: A deep representation for volumetric shapes. In *Conference on computer vision and pattern recognition (CVPR)*.

Wu, J., Zhang, C., Xue, T., Freeman, W. T., & Tenenbaum, J. B. (2016). Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In *Advances in neural information processing systems (NIPS)*.

Yan, X., Yang, J., Yumer, E., Guo, Y., & Lee, H. (2016). Perspective transformer nets: Learning single-view 3D object reconstruction without 3D supervision. In *Advances in neural information processing systems*.

Zhou, T., Tulsiani, S., Sun, W., Malik, J., & Efros, A. A. (2016). View synthesis by appearance flow. In *European conference on computer vision (ECCV)*.