Global-to-Local Generative Model for 3D Shapes

HAO WANG*, Shenzhen University
NADAV SCHOR*, Tel Aviv University
RUIZHEN HU, Shenzhen University
HAIBIN HUANG, Megvii / Face++ Research
DANIEL COHEN-OR, Shenzhen University and Tel Aviv University
HUI HUANG†, Shenzhen University

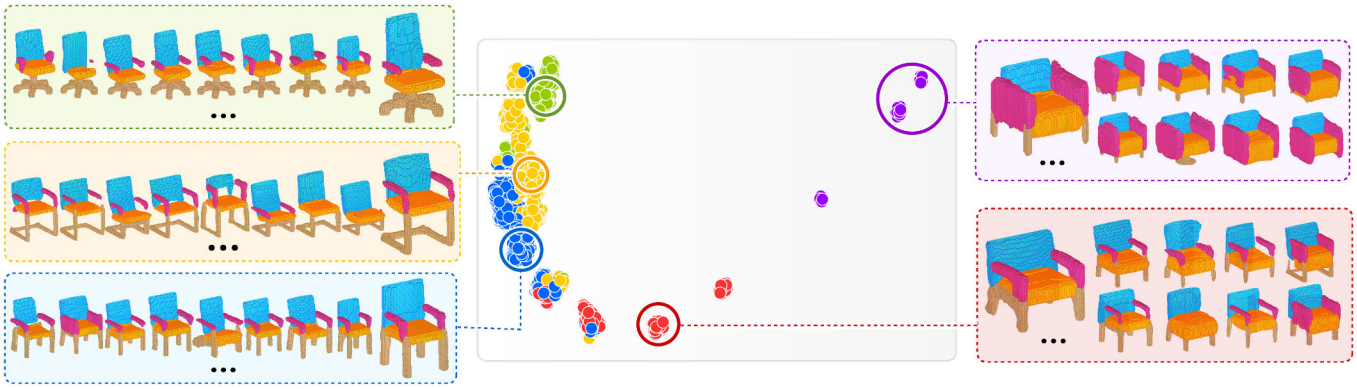


Fig. 1. Given a collection of 3D semantically segmented chairs, we train a network to generate new chairs from the same distribution. The 1024 generated chairs are encoded using an auto-encoder and embedded into 2D using MDS with the Euclidean distance in the latent space. The five colors of the displayed embedded points are associated with clusters of the training data. For each cluster, representative chairs are shown in groups with a similar color of the background. We can see rich variations in shape geometry.

We introduce a generative model for 3D man-made shapes. The presented method takes a global-to-local (G2L) approach. An adversarial network (GAN) is built first to construct the overall structure of the shape, segmented and labeled into parts. A novel conditional auto-encoder (AE) is then augmented to act as a part-level refiner. The GAN, associated with additional local discriminators and quality losses, synthesizes a voxel-based model, and assigns the voxels with part labels that are represented in separate channels. The AE is trained to amend the initial synthesis of the parts, yielding more plausible part geometries. We also introduce new means to measure and evaluate the performance of an adversarial generative model. We demonstrate that our global-to-local generative model produces significantly better results than a plain three-dimensional GAN, in terms of both their shape variety and the distribution with respect to the training data.

*Joint first authors
†Corresponding author: Hui Huang (hhzhiyan@gmail.com)

Authors' addresses: Hao Wang, Shenzhen University; Nadav Schor, Tel Aviv University; Ruizhen Hu, Shenzhen University; Haibin Huang, Megvii / Face++ Research; Daniel Cohen-Or, Shenzhen University and Tel Aviv University; Hui Huang, College of Computer Science & Software Engineering, Shenzhen University.

0730-0301/2018/11-ART214 $15.00
https://doi.org/10.1145/3272127.3275025

CCS Concepts: • **Computing methodologies → Shape modeling**;

Additional Key Words and Phrases: Shape modeling, generative adversarial networks, semantic segmentation, global-to-local, part refiner

ACM Reference Format:
Hao Wang, Nadav Schor, Ruizhen Hu, Haibin Huang, Daniel Cohen-Or, and Hui Huang. 2018. Global-to-Local Generative Model for 3D Shapes. *ACM Trans. Graph.* 37, 6, Article 214 (November 2018), 10 pages. https://doi.org/10.1145/3272127.3275025

1 INTRODUCTION

Three-dimensional content creation has been a central research area in computer graphics for decades. The main challenge is to minimize manual intervention, while still allowing the creation of a variety of plausible 3D objects. It is particularly challenging to create a novel shape that appears to be sampled from the distribution of a given class. It is hard to encapsulate the essence of a class of objects and express it in a compact model to guide the creation of a sample in the class. Common 3D generative techniques are based on existing parts taken from objects in the class, and synthesizing novel shapes by combining them, following probabilistic models or geometric constraints [Chaudhuri et al. 2011; Kalogerakis et al. 2012].

The emergence of generative neural networks and, in particular, adversarial networks [Goodfellow et al. 2014] offers new means to generate elements from a particular learned class. The key idea is that instead of explicitly modeling the class, a discriminator is trained to tell whether a generated model belongs to the target class

or not. The trained discriminator implicitly supervises the generator to model the class distribution.

These generative adversarial networks (GANs) have recently gained popularity in various domains and applications, showing promising results. Generally speaking, GANs achieve reasonable success in generating samples from a class distribution [Salimans et al. 2016a]. Nonetheless, they typically lack the ability to synthesize the details well [Berthelot et al. 2017; Gulrajani et al. 2017]. The problem is even more pronounced for GANs that aim at generating complex 3D man-made models [Liu et al. 2017; Wu et al. 2016]. Previous efforts in developing a 3D GAN for man-made objects have partial success in modeling their 3D structures. However, they behave rather poorly in modeling their finer geometric details, exhibiting significant noise, outliers and disconnected or missing parts. While the use of Hashing or Octree [Shao et al. 2018; Tatarchenko et al. 2017] can improve the output resolution, we opted to use a simple and plain voxel representation, with a high enough resolution to demonstrate the conceptual advancement of our approach.

In this work, we present a global-to-local generative model to synthesize 3D man-made shapes; see Fig. 1 as an example. It is based on an adversarial network to construct a global structure of the shape, with local part labels. The global discriminator is trained to distinguish between the whole real and generated 3D shapes, while the local discriminators focus on the individual local parts. A novel conditional auto-encoder is then introduced to enhance the part synthesis. Specifically, The GAN synthesizes a voxel-based model, and assigns part labels to each voxel. The auto-encoder acts as a part-level refiner to amend the initial synthesis of each part. To further enhance the shape generation, we introduce two more new losses in addition to the adversarial loss. One encourages the part compactness by measuring the *purity* of part regions, and the other helps to improve the *smoothness* of the synthesized surface. Nevertheless, the quality of the generated shapes is still rather low. The resolution of the 3D shape generation is in general too low to model fine details, like for example the blades of turbines of an airplane or the hands and fingers of a human body. Our G2L approach, however, introduces a conceptual advancement in the generative modeling of man-made shapes.

The evaluation of our generative model, like any other GAN, is far from trivial. The performance of a GAN should be measured by the quality of each of the synthesized objects and by their overall variety and agreement to the training data distribution. These measures were overlooked in previous works. We introduce new means to measure and evaluate the performance of an adversarial generative model, and demonstrate that our system produces significantly better results than a plain three-dimensional GAN.

2 RELATED WORK

2.1 Assembly-based Shape Synthesis

There are numerous works that create new 3D models by assembling from existing components. The pioneer work [Funkhouser et al. 2004] composes shapes by retrieving relevant shapes from a repository, then cuts and extracts components from these shapes and glues them together to form a new shape. The following works [Chaudhuri and Koltun 2010; Fish et al. 2014; Huang et al. 2015; Kalogerakis et al. 2012; Kim et al. 2013; Talton et al. 2012; Xu et al. 2012] try to improve the modeling process with more sophisticated techniques that consider the part relations or shape structures, e.g., employing Bayesian networks or modular templates. We refer to a STAR report [Mitra et al. 2013] for an overview of works on this aspect. The technique we propose here is also data-driven, but it trains a GAN to synthesize 3D shapes, rather than assembling existing parts.

2.2 Generative Neural Networks

With the development of deep learning techniques, there has been a surge of research interest in deep generative models in recent years. Deep generative models based on variational auto-encoders (VAE) [Kingma and Welling 2014] or generative adversarial networks (GAN) [Goodfellow et al. 2014] have made remarkable progress in image generation problems [Isola et al. 2017; Radford et al. 2015; Wang and Gupta 2016; Yan et al. 2016; Zhu et al. 2016]. With the recent introduction of large publicly available 3D model repositories [Chang et al. 2015; Wu et al. 2015], there have been attempts to generate 3D shapes using similar methods.

Most existing methods explore the problem of 3D reconstruction from a given 2D image or sketch. Girdhar et al. [2016] combine a 3D auto-encoder with an image encoder, to map 3D shapes and 2D images together into a common latent space to build the connection, so that 3D shapes can be encoded given new 2D images. Wu et al. [2017] propose a disentangled, two-step end-to-end trainable model that sequentially estimates 2.5D sketches and a 3D object shape from a given image. Instead of working on a volumetric representation of 3D shapes, Fan et al. [2017] propose using a point cloud representation for the reconstructed 3D shapes from a single image to encode more geometric details, without adding much computational overhead. Lin et al. [2018] further increase the efficiency by introducing a differentiable pseudo-renderer for 3D generation.

2.3 Learning-based Shape Synthesis

More related to our method are adversarial networks that learn to generate 3D shapes from noise input directly [Li et al. 2017; Nash and Williams 2017; Wu et al. 2016]. Wu et al. [2016] are the first to extend GAN for 3D shape generation, which shows that the generator can capture the object structure implicitly and synthesize plausible 3D objects. Nonetheless, without explicitly constraining the part structures, semantic validation of the generated shapes cannot be guaranteed and important structure properties cannot be well-preserved during the generation.

Common GAN networks consist of one generator and one discriminator. Recently, Some recent attempts have been made to combine multiple discriminators or generators together. Hoang et al. [2017] train multiple generators to explore different modes of the data distribution. The array of generators shares their parameters and a softmax classifier is used to classify which generator the data comes from. Similarly, MIX+GAN proposed in [Arora et al. 2017] uses a mixture of generators to enforce the generated distribution to be of higher diversity. In contrast, the combination of multiple discriminators and a single generator aims at constructing a stronger discriminator to guide the generator. D2GAN proposed in [Nguyen et al. 2017] employs two discriminators to optimize both Kullback-Leibler

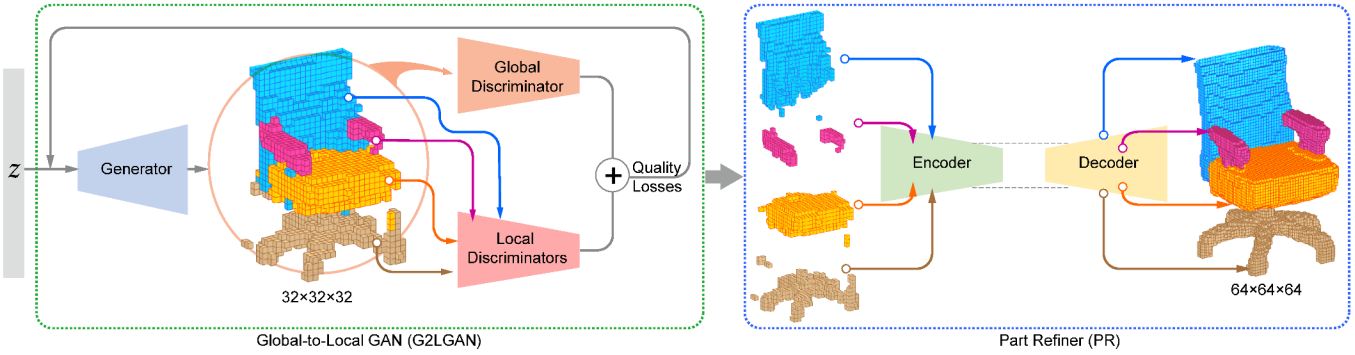


Fig. 2. An overview of our generative model. It consists of two parts: a global-to-local GAN (left) that synthesizes a 32^3 model and a part refiner (right) that enhances the synthesized parts of the model, by refining and completing missing regions, and increasing the resolution to 64^3.

(KL) and reverse KL divergences. GMAN proposed in [Durugkar et al. 2016] explores an array of discriminators to boost the learning of a generator. Some methods [Iizuka et al. 2017; Liu et al. 2018; Yu et al. 2018] that deal with image completion, use a global discriminator together with multiple local discriminators. The global discriminator takes the entire image as input to assess whether it is coherent as a whole, while the local discriminators focus on missing regions to guarantee the completion consistency around those areas. We employ a global and a few local discriminators in a similar manner on 3D shapes, while our local discriminators focus on semantically segmented parts for enhancing their fine geometry.

Li et al. [2017] present a top-down approach, focusing more on the part structure for 3D shape generation. They learn symmetry hierarchies of shapes with an auto-encoder and then generate variations of these hierarchies, using an adversarial discriminator. The nodes of the hierarchies are independently instantiated with parts. However, these parts are not necessarily connected and their aggregation does not form a coherent connected shape. In our work, the shapes are generated coherently as a whole, and special care is given to the inter-parts relation and their connectivity.

The most relevant work to ours, is the shape variational auto-encoder presented by Nash and Williams [2017]. They developed an auto-encoder to learn a low-dimensional latent space, and then novel shapes can be generated by sampling vectors in the learned latent space. Like our GAN, the generated shapes are segmented into the relevant semantic parts. Unlike our technique, they require a one-to-one dense correspondence among the training shapes, since they represent the shapes as an order vector. Their auto-encoder learns the overall (global) 3D shapes with no attention to the local details. Our approach pays particular attention to both the generated surface and the relation in-between local shape parts.

3 OVERVIEW

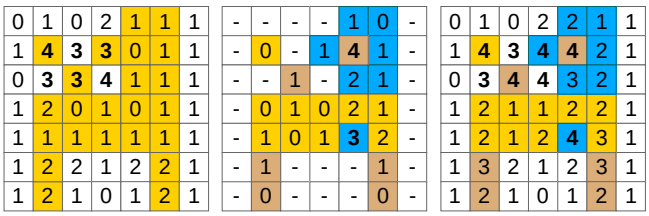
Our generative model consists of two units: a Global-to-Local GAN (G2LGAN) and a Part Refiner (PR)(see Fig. 2). Using a generative adversarial network training, with global and local discriminators, the global generator generates 3D semantically segmented models. The PR is based on an auto-encoder architecture. Its goal is to refine the individual semantic part output from G2LGAN.

The training of GAN presents several challenges. First, the generator tends to take the easiest way to fool the discriminator. It suffices to generate a rather small number of shapes that are good enough to be perceived as real by the discriminator. This notorious behavior of GANs is known as "mode collapse". Second, the delicate min-max game between the generator and the discriminator is not easy to tune to reach a stable convergence. We, therefore, design a global-to-local GAN derived from the WGAN-GP model [Gulrajani et al. 2017] that has the theoretical support of reaching a convergence and was shown to avoid mode-collapse on multiple tasks. In our model, the global discriminator is trained to distinguish between real and synthesized 3D models, leading to the valid generation of global shapes. The local discriminators are focusing on the individual semantic parts, each aiming to improve its designated part. Nonetheless, noting that a high resolution 3D model, represented by voxels, has a very sparse presence in its surrounding volume. This sparseness of the representation makes the learning task overly difficult, resulting in various artifacts and disconnected components. We, therefore, augment G2LGAN with a PR, which amends the local geometry and adds small scale details to the parts generated by the global generator.

Semantically Segmented Shape Generation. We trained G2LGAN using a WGAN-GP loss [Gulrajani et al. 2017], on the segmented ShapeNet dataset [Yi et al. 2016]. The generator takes a random noise vector as input, and passes it through a series of convolution layers to produce a 3D semantically segmented shape. Compared to the training data, the generated results have two noticeable artifacts: i) at the shape level, they are usually rough and often contain floating disconnected voxels; and ii) at the part level, voxels tagged as one part often penetrate and intermix with those of an adjacent part. To improve the generation results, we introduce three novel modifications: i) the use of *multiple local discriminators*, each of which focuses on a single semantic part; ii) the addition of a *smoothness loss* to improve the smoothness of the synthesized shape surfaces and alleviate the generation of disconnected voxels; and iii) the introduction of a *purity loss* to prevent the intermingling of different parts and to enhance the connectivity. During training, the generator's output is fed into a global discriminator and a set of

local discriminators. The global discriminator evaluates the quality of the overall shape, while each local discriminator focuses on its dedicated semantic part. All the global and local discriminators are trained simultaneously.

Part-wise Refinement. The G2LGAN generates semantically segmented models. We then separate the different semantic parts, and feed them part-by-part to the part refiner. The PR is trained to encode a given, possibly rough or incomplete part, into a latent vector and decode it back into a complete part. To train a universal PR for all different types of parts, we concatenate the part label to the latent vector that encodes the part. The PR refines two aspects of the generated parts; i) the resolution of the input part is enlarged from 32^3 to 64^3 to generate finer details in the model; and ii) rough or incomplete parts become smoother and are filled as needed.



(a) smoothness (b) purity (c) combination

Fig. 3. Voxels contribution to the different losses: (a) smoothness, (b) purity, and (c) their combination. The value inside each voxel represents its contribution to the specific loss. As can be seen from (a) and (b), the voxels that have the highest values (shown in bold) are the most problematic ones, floating or intermingling. The summary of the voxels contribution is shown in (c), where a voxel is marked in bold, if it was marked in bold in (a) or (b).

4 METHOD

4.1 Generator and Discriminators

We adopt the WGAN-GP [Gulrajani et al. 2017] as the base architecture of the generator. The generator takes a random noise vector $z \in \mathbb{R}^{200}$ as input. It consists of four transposed convolution layers. We use filter size of 4^3 with strides $[1, 1, 1]$ for the first layer and $[2, 2, 2]$ for the other layers. We use batch normalization and a ReLU activation function after each of the first three transposed convolution layers. We use softmax as the activation function of the last layer. The generator outputs a volume of $32^3 \times C$, where C denotes the number of semantic parts in the segmented model. Each voxel is represented by a probability vector, with the length of C. Each channel corresponds to a part label. When we visualize the generated models, we assign each voxel with the part label that has the highest probability.

The two main well-known artifacts caused by a vanilla generator, are rough models and voxels intermingling between the different model parts. To improve the performance, we introduce global-to-local discriminators and two new quality loss terms to help guide the generator to produce decent results.

The global discriminator has four convolution layers, with a filter size of 4^3. The first three layers use strides of $[2, 2, 2]$ and a leaky ReLU, while the last layer uses strides of $[1, 1, 1]$ and no activation layer. We use a batch size of 64, and Adam optimizer with settings recommended in [Gulrajani et al. 2017], i.e., learning rate $= 1e^{-4}$, $\beta_1 = 0.5$, $\beta_2 = 0.9$, $\lambda = 10$ and five discriminator iterations.

The local discriminators follow the same pattern of the global discriminator, except that their inputs are semantically segmented parts instead of a whole shape. The softmax output of the generator $G(z)$ is converted into multiple-channel one-hot vector. The multiple-channel vector is then split into single channel vectors, serving as binary attention masks $\{M^i\}$ for different parts. By the element-wise multiplication between the attention mask and softmax output, i.e., $G(z) \circ M^i$, local discriminators only focus on their corresponding semantic parts.

By assigning different weights for the global and local discriminators, their combinations are trained jointly to guide the generator synthesizing shapes with enhanced global structure and more fine scale details. Our adversarial loss function is thus defined as:

$$\mathcal{L}_a = \mathcal{L}_{\text{global}} + \sum_{i=1}^{c} w_i \mathcal{L}^i_{\text{local}}, \tag{1}$$

where $\mathcal{L}_{\text{global}}$ and $\mathcal{L}^i_{\text{local}}$ are set as the same in [Gulrajani et al. 2017]. In all our experiments, we set $w_i = 1$ for all the categories.

4.2 Quality Losses

The main goal of adding the quality losses, i.e., smoothness and purity losses, is to penalize the generator based on the effect of each individual voxel in the generated model. We would like to compute the contribution of each voxel on the artifacts of the vanilla GAN. As can be seen from Fig. 3, the unwanted voxels receive the highest values, and thus have a larger effect on the losses.

Note that since the generated shapes are eventually visualized by assigning each voxel with the label of maximal probability and using the softmax output as a means to compute the penalty of each voxel may lead to closer label distributions between neighboring voxels, while still maintaining their original unwanted label, we decided to penalize the intractable voxels, using the equivalent one-hot representation, instead of the direct softmax results. However, since converting a softmax output into a one-hot vector is not a differentiable operation, we use a sigmoid function, centered at 0.5, with a steep slope to approximate the one-hot representation: $\text{sig}(x) = 1/(1 + e^{-100(x-0.5)})$.

Smoothness Loss. We introduce the smoothness loss to smooth the generated models and reduce the number of floating voxels. In this loss, we ignore the part labels and treat the model as one (unsegmented) global shape. We consider the relation and connectivity between the object and its background, and define the *Smoothness Contribution* (S_c) of each voxel, as the sum of L_1 distances between its approximate one-hot label vector and those of its adjacent voxels; see Fig. 3(a). Meanwhile, we set a threshold η_s (= 2.89 by default) to avoid penalizing border voxels, i.e., $\forall x, S_c(x) = 0$, if $S_c(x) < \eta_s$. The smoothness loss follows as:

$$\mathcal{L}_s = \sum_x (S_c(x))^2, \tag{2}$$

where we square $S_c(x)$ to achieve an exponential difference between floating voxels and voxels that stick out of the model's surface.

Purity Loss. We introduce the purity loss to prevent intermingling between different parts. In this loss, we ignore the background voxels and consider only the object voxels. We define the *Purity Contribution* (P_c) of each voxel, as the sum of L_1 distances between its approximate one-hot label vector and those of its adjacent *parts* voxels; see Fig. 3(b). Again, we define a threshold η_p (= 1.69 by default) to avoid penalizing border voxels, i.e., $\forall x, P_c(x) = 0$, if $P_c(x) < \eta_p$. The purity loss follows as:

$$\mathcal{L}_p = \sum_x (P_c(x))^2, \tag{3}$$

where, similar to (2), we square $P_c(x)$ to achieve an exponential difference between the different levels of voxels intermingling.

Our Generator Loss is thus defined as a weighted sum of the aforementioned three losses (1), (2), and (3):

$$\mathcal{L}_{gen} = \mathcal{L}_a + w_s\mathcal{L}_s + w_p\mathcal{L}_p, \tag{4}$$

where we set $w_s = 0.05$ and $w_p = 0.0667$ by default.

4.3 Part Refiner

We use an auto-encoder architecture with a reconstruction loss as a Part Refiner (PR). The PR receives a part with its label to produce a refined model of that part. The input part is represented as a 32^3 volume, the label is a one-hot vector and the output part is a volume of size 64^3. We use three convolution layers, with filter size 4^3 and strides of $[2, 2, 2]$, followed by a FC layer, to encode the input part into a 128-dimension latent space. We use three 128-dimensional FC layers to encode the part-label vector. The two 128-vectors are then concatenated and fed through two 256-dimensional FC layers to combine the encoding of both inputs. The decoder consists of five transposed convolution layers with a filter size of 4^3 and strides of $[2, 2, 2]$. We use batch normalization and ReLU, after each layer in both the encoder and decoder.

To compute the reconstruction loss, we need to have both the input, i.e., part and label, and the expected output. We can generate trivial pairs from the training set, where the input is from the 32^3 resolution samples and the output is taken from the same model, but at 64^3 resolution. To grant the PR ability to refine imperfect parts, we further construct a second training set by pairing imperfect parts and their perfect counterparts. We randomly generate 1,024 models using the trained G2LGAN and separate them into different parts as inputs to the PR. As the expected outputs, for each generated part, we retrieve its 3-nearest-neighbors (to prevent the PR from memorizing a one-to-one mapping) from the 32^3 training set, and then use their corresponding 64^3 representations. At training time, when a generated part is fed into the PR, an average voxel-wise cross-entropy is computed as the reconstruction loss by comparing the output to one of the input's nearest-neighbors, picked randomly.

When searching for those pairing nearest neighbors, we would also like to enable our PR to map a given part into an empty output, so that it can remove noise or small unorganized part outliers. To achieve this, we separate the generated parts into three categories: empty parts, parts that consist of only few (less than 10 by default) voxels, and the remaining parts. We directly remove the empty parts from the training set, since we do not want to burden our model with learning empty-to-empty mapping. The parts that consist of a few voxels would be very likely a noise or outlier, for which we aim to clean up by mapping them to an empty part. Thus, as a metric to find the nearest-neighbors, we use L_1 distance for these suspicious parts, allowing a mapping to an empty volume. For the remaining parts, we use the Intersection-over-Union (IoU) metric instead. The random noise may accidentally intersect with different parts in the training set and be paired to them instead of an empty part since the IoU with an empty part is always zero. We use the Adam optimizer for training, with a batch size of 64 and a learning rate of $5e^{-5}$.

5 RESULTS AND EVALUATION

To evaluate our method, we set up two baselines. The first is [Wu et al. 2015] with one modification, referred to as *3DGAN*, which outputs C channels for every voxel instead of one. We train the 3DGAN as presented in the paper. The second one, referred to as *Baseline*, is similar to G2LGAN without the local discriminators and the quality losses. The Baseline's discriminator is identical to the global discriminator of G2LGAN (see Sec. 4.1). We use a batch size of 64, and an Adam optimizer with settings recommended in [Gulrajani et al. 2017], i.e., learning rate = $1e^{-4}$, $\beta_1 = 0.5$, $\beta_2 = 0.9$, $\lambda = 10$ and 5 discriminator iterations. We present randomly picked generated results from both two baselines in the supplementary material.

5.1 Training Sets and Generation Results

We build our training set from a collection of 3D meshes on four categories: Chair (900), Airplane(2690), Lamp(1546), and Table(5256). These models are taken from ShapeNet [Chang et al. 2015], which have been consistently aligned, scaled, and semantically segmented. The semantic labels are from [Huang et al. 2015], and each object category has a fixed number of semantic parts: four parts (back, seat, leg, armrest) for Chair; four parts (body, wing, tail, engine) for Airplane; four parts (base, shade, canopy and tubing) for Lamp; and three parts (top, leg and drawer) for Table. Individual shapes may or may not contain all of these parts. By combining these 3D meshes and their part label information together, we voxelize them to form our volumetric training sets with corresponding semantic indexes.

After training, we generate 1024(= 64×16) new models for each object category. Recall that the goal of the generator in a GAN setting is to learn the mapping from a latent space to the distribution of the training data. Thus, to show some representative results, we define a *confidence score* for each generated shape. More specifically, for each category, we first train a shape-level auto-encoder to extract a feature representation for each of the training shapes, and perform K-Means on the whole training set to get 10 clusters. We train a classifier to map each of the training shapes into one of the clusters. For each generated shape, we then define the highest Softmax probability obtained from the classifier, as its confidence score.

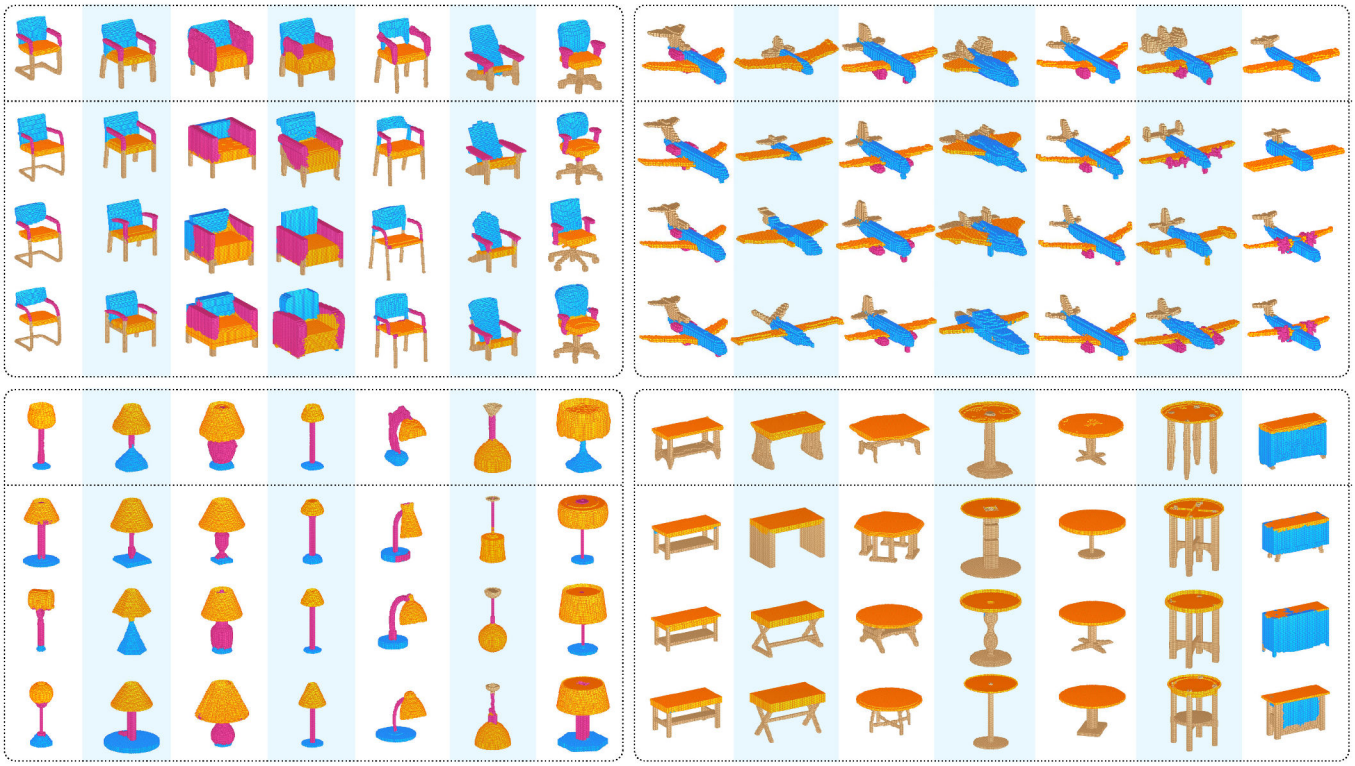


Fig. 4. A gallery of our generated Chairs, Airplanes, Lamps, and Tables shown above, with their 3-nearest-neighbors retrieved from the training set.

Fig. 5. PR Improvement. For each category, we present four examples of the improvement achieved by the PR. Shapes generated by G2LGAN are shown on the top row, and their PR enhanced versions are provided underneath for a clear comparison.

In Fig. 4, for each category, we present a set of representative generated shapes located in different clusters, which have the highest confidence scores in the top row and retrieve the 3-nearest-neighbors from the training set for each shape to show below for a visual quality comparison. The shape matching distance is computed as the average of the Chamfer distances [Fan et al. 2017] over all paired semantic parts. As we can see, our generated shapes look similar to their closest training neighbors, i.e., with high validity, yet also contain meaningful variations in geometry. More generated results are shown in the supplementary material.

It is worth emphasizing that our PR cannot only increase the output resolution to 64^3, but also add important missing fine details while cleaning up noisy components; see Fig. 5, where we can clearly observe the significant reduction of noise and refined details. More examples are included in the supplementary material.

5.2 Quantitative Evaluation on Shape Synthesis

We present three different evaluation metrics to quantitatively evaluate the variety and quality of our synthesized shapes. The first metric (*3D inception score*) evaluates the variety and quality of the

Table 1. Inception and Symmetry comparisons of generated chairs. Both baselines receive lower scores in all categories, in comparison to our models, G2LGAN and G2LGAN+PR. Local Discriminators (LD) better improve the Baseline in the symmetry score for Leg and Arm while the Quality Losses (QL) better improve the other categories.

Models	Inception	Symmetry				
		Shape	Back	Seat	Leg	Arm
3DGAN	5.84	0.70	0.71	0.76	0.35	0.16
Baseline (BL)	5.55	0.78	0.76	0.80	0.55	0.51
BL & LD	5.62	0.80	0.78	0.80	0.64	0.63
BL & QL	5.99	0.82	0.82	0.85	0.60	0.53
G2LGAN	6.00	0.84	0.83	0.85	0.63	**0.66**
G2LGAN+PR	**6.17**	**0.91**	**0.93**	**0.94**	**0.71**	0.64
GT	8.16	0.96	0.93	0.94	0.84	0.84

generated shape set, the second metric (*symmetry score*) evaluates the symmetry of the synthesized parts, and the third metric (*distribution distance*) uses a few statistical measures to evaluate the distance between the synthesized shapes and the training set.

3D Inception Score. The inception score was first introduced by [Salimans et al. 2016b] and is commonly used for evaluating generated images. The score is based on the inception classification network [Szegedy et al. 2016] which was pre-trained on the ImageNet dataset. It measures both the variety of the generated content and its quality, based on the confidence of the classifier and the variance of the generated classes.

Since the generated 3D models are from a single category and the inception score is defined based on a classifier, we cannot directly apply the computation of the inception score to our data. However, as described in Section 5.1, we can cluster the training data from the same category into 10 clusters, and then consider the clusters as different classes to train the corresponding 3D inception classification network for computing the inception score. For the PR output, which is in higher resolution 64^3, we down-sample it to 32^3 first and then compute the inception score. As we can see from Table 1, the PR improves the inception score of the G2LGAN model and receives a higher score than the baseline models. Although the 3DGAN baseline obtains a better inception score than the Baseline and Baseline with local discriminators, its results for the symmetry are the lowest and its visual results, e.g., shown in the supplementary material, are extremely poor in comparison to our models.

Symmetry Score. All the categories that were tested are man-made objects, thus, they are more likely to have a symmetric structure, as we can see from the symmetry score of the training data (GT) in the bottom row of Table 1. It is, therefore, expected that generated model should also demonstrate the appropriate symmetry.

We only evaluate bilateral symmetry here and define the symmetry score as the percentage of voxels which got matched after a reflectance transformation given a symmetry plane. Since all the shapes in each category are aligned, we set a fixed symmetry plane for all of the training and generated shapes for the same category. We then compute the corresponding symmetry score for each semantic part and the global shape. We can see in Table 1 that the symmetry scores of our method are consistently higher than those of the baseline models.

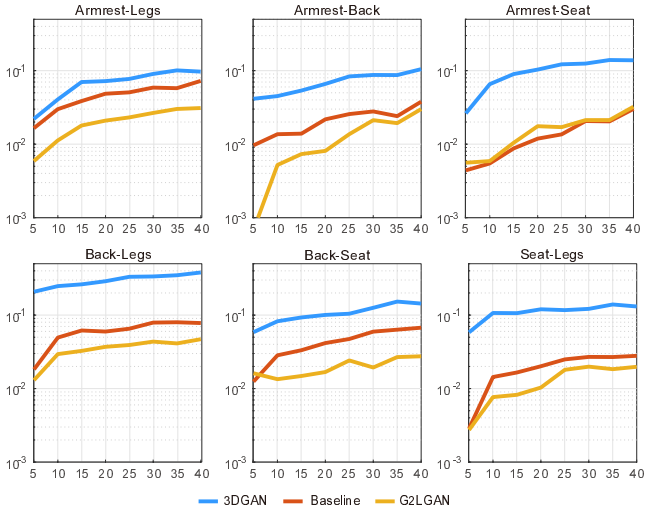


Fig. 6. Comparison on the JSD of pair-wise features of the generated shapes from the baselines and G2LGAN relative to the training data. The horizontal axis represents the number of clusters and the vertical axis represents the JSD value. Smaller values are better.

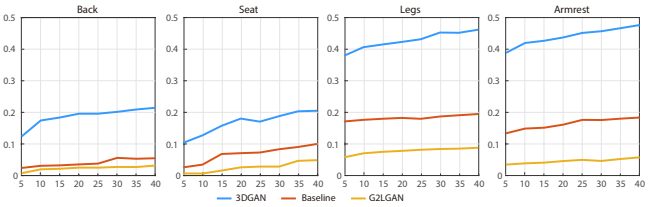


Fig. 7. Comparison on the JSD of part-wise features of the generated shapes from the baselines and G2LGAN relative to the training data. The horizontal axis represents the number of clusters and the vertical axis represents the JSD value. Smaller values are better.

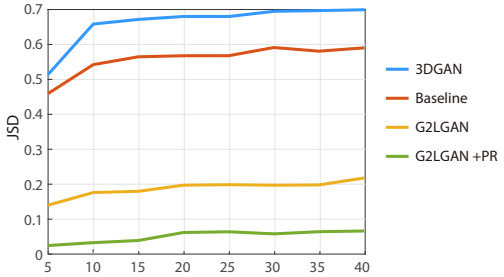


Fig. 8. Comparison on the JSD of component-wise features of the generated shapes from the baselines, G2LGAN and G2LGAN+PR relative to the training data. The horizontal axis represents the number of clusters and the vertical axis represent the JSD value. Smaller values are better.

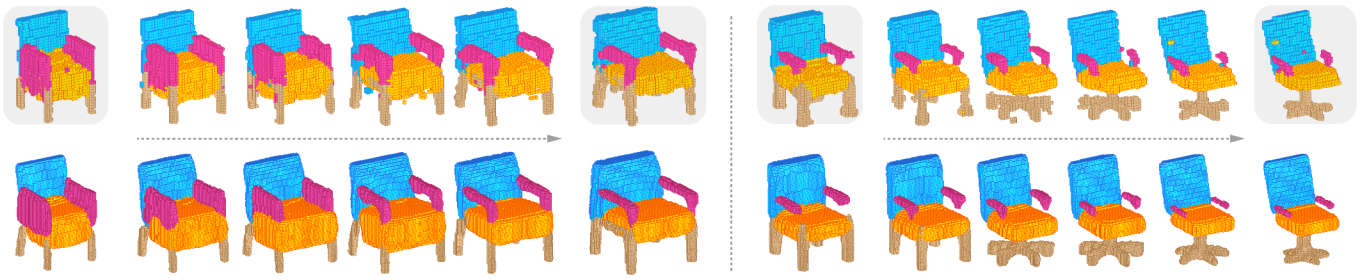


Fig. 9. Interpolating between two pairs of chairs at the left and right, respectively. Shapes generated by G2LGAN are shown on the top row, and their PR enhanced versions are provided underneath. The PR results stay sharp and clear throughout the different stages, where usual latent space interpolation results, such as G2LGAN, have some artifacts and noise. This results in a clear-cut transition between the different stages in the PR interpolation, which can be seen from the arm-rests of the left example and the legs in the right example.

Distribution Distance. The synthesized shapes are represented in a high-dimensional space, and thus, it is not trivial to measure the distance between their distribution and the training data distribution. To cope with this difficulty, we consider and compare three different feature distributions, which account for different shape properties including: i) binary part features (pair-wise); ii) unary part features (part-wise); and iii) global component features (component-wise).

In the following we elaborate on these three feature distributions:

i To capture the pair-wise feature between different parts, we define a pair-wise feature vector for each pair of two different parts i and j. More specifically, for each voxel from part i, we check its 27 neighboring voxels and see if any of them belongs to part j, which results in a 27-dimensional binary vector. For all the voxels from part i, we accumulate those corresponding vectors and get a 27-dimensional vector with each entry indicating the number of neighboring voxels belonging to part j that come from the corresponding direction.
Once we have these pair-wise feature vectors of the training data, we cluster them to k clusters, using K-Means and Euclidean distance. For the 1024 generated shapes, we compute their pair-wise feature vectors accordingly and distribute them to the pre-computed clusters. The distribution of shapes over those different clusters can be presented as a histogram. We then use the Jensen-Shannon Divergence (JSD) to measure the similarity between these histograms of the training data and the generated data.
Note that the smaller the k, the easier it is for the generated set to match the target distribution. Therefore, to better estimate the variety, we vary the cluster number k over a range of values, and sum the k-variety measures. As we can see from Fig. 6, our G2LGAN model performs better than the 3DGAN baseline in all cases, and also performs better than the Baseline model in most of the cases except for Armrest-Seat pair, where the performances are similar.

ii To capture the unary feature for each individual part i, we simply check the part size, i.e., the amount of voxels that construct this part. The part-size distribution is easy to compute for both training and generated set. We again use JSD to compute the distance between the distributions. As can be seen from Fig. 7, our G2LGAN model performs consistently better than the baselines, which indicates that the parts we generated are more likely to have the right size.

iii To capture a more global shape-level feature, we count the number of connected components for each part in each shape. If the generated samples are good enough, their connected components on each semantic part should be as pure and clean as the real data. For a shape coming from a category with c part labels, it will have a c-dimensional vector, where each entry indicates the number of connected components of the corresponding part. We again compute a distribution for the training shapes and another for the generated shapes, using those c-dimensional vectors, and then use JSD to compute the distance. Since the input vectors z are randomly sampled from a normal distribution, artifacts, such as noise and incomplete parts, are inevitable in the generated shapes. However, as shown in Fig. 8, G2LGAN achieves better performance than the baselines and the PR is able to improve even further the quality of the shapes generated by G2LGAN.

Ablation Study. We conduct an ablation study to emphasize the benefits of each added module on top of our Baseline. As can be seen in Table 1, the local discriminator improves the Baseline results in all the categories, except for Seat symmetry. Moreover, in the Arm-rests and Legs, which usually have more than one piece and small volume, the local discriminators achieve significant improvement over the Baseline. The quality losses improve the results of the Baseline in all categories. Their major advantage is on the Inception score and the Back and Seat parts, which usually consist of one large piece. The local discriminator and quality losses complete one another. This is mentioned above and can be seen in Table 1, where G2LGAN obtains better results than both of them on all the categories, except for Leg. G2LGAN+PR, our final model, achieves the best scores for all categories, except for Arm, where it is second to G2LGAN.

Latent Space Interpolation. We also test whether our model supports latent space interpolation, especially due to our two-step generation process. As shown in Fig. 9, it is possible to do the latent space interpolation with our model. The major difference in our latent space interpolation is that the PR results stay sharp and clear, where usual latent space interpolation results, such as G2LGAN,

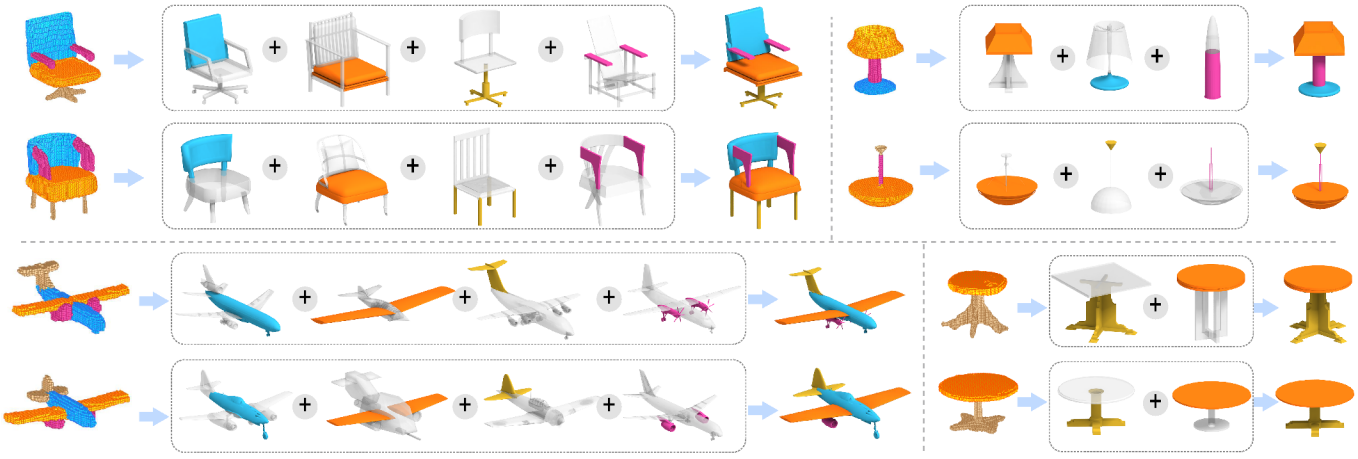


Fig. 10. Part-wise surface assembly. We present two examples for each category. On the left column is our generated shape, in the middle columns are the part-wise nearest neighbors from the training set. On the right column is our reconstructed mesh that fits the generated shape.

have some artifacts in the middle stages. This results in a clear cut transition between the different stages in the PR interpolation.

5.3 Surface Assembly

The generated shapes are represented by voxels and thus their resolution is highly limited. To finalize the shape production with finer and richer details, for each generated part, we find the most similar part in our training set by measuring their Chamfer distance on their voxel representation. This matched training part is also associated with a complete and continuous surface mesh part (see, e.g., the middle columns in Fig. 10) from ShapeNet. The retrieved mesh parts can then be warped to fit into the voxel-based input parts (the left of each block in Fig. 10) using the detail-preserving deformation method [Sumner et al. 2007], yielding assembled mesh models with desirable surface details (the right of each block in Fig. 10). More results can be found in the supplementary material.

6 CONCLUSION AND FUTURE WORK

We present a Global-to-Local approach for generating semantically segmented 3D man-made shapes. A GAN with global and multiple local discriminators (G2LGAN) is developed, where two new losses, i.e., smoothness and purity losses, are added to improve the shape generation with more coherent parts. Moreover, we introduce a part refiner to increase the model resolution, while enhancing the synthesis quality of the individual local parts. For applications, our generated volumetric models can be part-wisely reconstructed and assembled into a variety of mesh models with rich surface details, as shown in Fig. 10. We have also proposed quantitative measurements to evaluate the quality of generated shapes, which will allow a more objective means to evaluate future GAN models.

The main limitation of our work stems from the fact that the models are represented with voxels, which heavily bounds the resolution of the generated shapes. It is alleviated by the use of local refiners, but is still far from the convention standard of a high resolution delicate shape. The use of two-step generation process could introduce another limitation. The G2LGAN may output shapes which are not close to the PR training data distribution. It is mitigated by sampling 1,024 shapes from the trained G2LGAN, to form a PR training data, consisting of $1,024 \times \#(parts)$ shapes, which cover the parts variability well. This offline process is computed only once. It is easy to samples even more parts from G2LGAN, if needed.

In the near future, we would like to develop a similar Global-to-Local approach for point clouds analysis and synthesis. Point sets are, by nature, unordered nor regular, and although recently there have been a number of works that successfully analyze them, e.g., [Qi et al. 2017]), the development of a generative synthesis network remains a challenge. Another interesting direction would be generating the semantic parts individually, and then assembling them into a complete and coherent 3D model.

ACKNOWLEDGEMENTS

We thank the anonymous reviewers for their valuable comments. This work was supported in parts by NSFC (61522213, 61761146002, 61861130365, 61602311), 973 Program (2015CB352501), GD Science and Technology Program (2015A030312015), Shenzhen Innovation Program (KQJSCX20170727101233642, JCYJ20170302153208613) and ISF-NSFC Joint Research (2472/17).

REFERENCES

Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. 2017. Generalization and Equilibrium in Generative Adversarial Nets (GANs). In *Proc. Int. Conf. on Machine Learning*, Vol. 70. 224–232.

David Berthelot, Thomas Schumm, and Luke Metz. 2017. BEGAN: boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717* (2017).

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, and others. 2015. Shapenet: An information-rich 3D model repository. *arXiv preprint arXiv:1512.03012* (2015).

Siddhartha Chaudhuri, Evangelos Kalogerakis, Leonidas Guibas, and Vladlen Koltun. 2011. Probabilistic Reasoning for Assembly-based 3D Modeling. *ACM Trans. on Graphics* 30, 4 (2011), 35:1–35:10.

Siddhartha Chaudhuri and Vladlen Koltun. 2010. Data-driven Suggestions for Creativity Support in 3D Modeling. *ACM Trans. on Graphics* 29, 6 (2010), 183:1–183:10.

Ishan Durugkar, Ian Gemp, and Sridhar Mahadevan. 2016. Generative multi-adversarial networks. *arXiv preprint arXiv:1611.01673* (2016).

Haoqiang Fan, Hao Su, and Leonidas J Guibas. 2017. A Point Set Generation Network for 3D Object Reconstruction From a Single Image. In *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*. 605–613.

Noa Fish, Melinos Averkiou, Oliver van Kaick, Olga Sorkine-Hornung, Daniel Cohen-Or, and Niloy J. Mitra. 2014. Meta-representation of Shape Families. *ACM Trans. on Graphics* 33, 4 (2014), 34:1–34:11.

Thomas Funkhouser, Michael Kazhdan, Philip Shilane, Patrick Min, William Kiefer, Ayellet Tal, Szymon Rusinkiewicz, and David Dobkin. 2004. Modeling by Example. *ACM Trans. on Graphics* 23, 3 (2004), 652–663.

Rohit Girdhar, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta. 2016. Learning a predictable and generative vector representation for objects. In *Proc. Euro. Conf. on Computer Vision*. 484–499.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems (NIPS)*. 2672–2680.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. 2017. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems (NIPS)*. 5769–5779.

Quan Hoang, Tu Dinh Nguyen, Trung Le, and Dinh Phung. 2017. Multi-Generator Gernerative Adversarial Nets. *arXiv preprint arXiv:1708.02556* (2017).

Haibin Huang, Evangelos Kalogerakis, and Benjamin Marlin. 2015. Analysis and Synthesis of 3D Shape Families via Deep-learned Generative Models of Surfaces. *Computer Graphics Forum* 34, 5 (2015), 25–38.

Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. 2017. Globally and Locally Consistent Image Completion. *ACM Trans. on Graphics* 36, 4 (2017), 107:1–107:14.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. 2017. Image-to-Image Translation with Conditional Adversarial Networks. *Proc. IEEE Conf. on Computer Vision & Pattern Recognition* (2017), 5967–5976.

Evangelos Kalogerakis, Siddhartha Chaudhuri, Daphne Koller, and Vladlen Koltun. 2012. A Probabilistic Model for Component-based Shape Synthesis. *ACM Trans. on Graphics* 31, 4 (2012), 55:1–55:11.

Vladimir G. Kim, Wilmot Li, Niloy J. Mitra, Siddhartha Chaudhuri, Stephen DiVerdi, and Thomas Funkhouser. 2013. Learning Part-based Templates from Large Collections of 3D Shapes. *ACM Trans. on Graphics* 32, 4 (2013), 70:1–70:12.

Diederik P Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *Proc. Int. Conf. on Learning Representations*.

Jun Li, Kai Xu, Siddhartha Chaudhuri, Ersin Yumer, Hao Zhang, and Leonidas Guibas. 2017. GRASS: Generative Recursive Autoencoders for Shape Structures. *ACM Trans. on Graphics* 36, 4 (2017), 52:1–52:14.

Chen-Hsuan Lin, Chen Kong, and Simon Lucey. 2018. Learning Efficient Point Cloud Generation for Dense 3D Object Reconstruction. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. 2018. Image Inpainting for Irregular Holes Using Partial Convolutions. *arXiv preprint arXiv:1804.07723* (2018).

Jerry Liu, Fisher Yu, and Thomas Funkhouser. 2017. Interactive 3D modeling with a generative adversarial network. In *Proc. Int. Conf. on 3D Vision*. 126–134.

Niloy Mitra, Michael Wand, Hao (Richard) Zhang, Daniel Cohen-Or, Vladimir Kim, and Qi-Xing Huang. 2013. Structure-aware Shape Processing. In *SIGGRAPH Asia 2013 Courses*. 1:1–1:20.

C. Nash and C. K. I. Williams. 2017. The Shape Variational Autoencoder: A Deep Generative Model of Part-segmented 3D Objects. *Computer Graphics Forum* 36, 5 (2017), 1–12.

Tu Nguyen, Trung Le, Hung Vu, and Dinh Phung. 2017. Dual discriminator generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*. 2667–2677.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 2017. Pointnet: Deep learning on point sets for 3D classification and segmentation. *Proc. IEEE Conf. on Computer Vision & Pattern Recognition* (2017), 652–660.

Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016a. Improved Techniques for Training (GANs). In *Advances in Neural Information Processing Systems (NIPS)*. 2234–2242.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016b. Improved Techniques for Training (GANs). In *Advances in Neural Information Processing Systems (NIPS)*. 2234–2242.

Tianjia Shao, Yin Yang, Yanlin Weng, Qiming Hou, and Kun Zhou. 2018. H-CNN: Spatial Hashing Based CNN for 3D Shape Analysis. *arXiv preprint arXiv:1803.11385* (2018).

Robert W. Sumner, Johannes Schmid, and Mark Pauly. 2007. Embedded Deformation for Shape Manipulation. *ACM Trans. on Graphics* 26, 3 (2007), 80:1–80:7.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*. 2818–2826.

Jerry Talton, Lingfeng Yang, Ranjitha Kumar, Maxine Lim, Noah Goodman, and Radomír Měch. 2012. Learning Design Patterns with Bayesian Grammar Induction. In *Proc. ACM Symp. on User Interface Software and Technology*. 63–74.

Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. 2017. Octree generating networks: Efficient convolutional architectures for high-resolution 3D outputs. In *Proc. Int. Conf. on Computer Vision*. 2088–2096.

Xiaolong Wang and Abhinav Gupta. 2016. Generative image modeling using style and structure adversarial networks. In *Proc. Euro. Conf. on Computer Vision*. 318–335.

Jiajun Wu, Yifan Wang, Tianfan Xue, Xingyuan Sun, Bill Freeman, and Josh Tenenbaum. 2017. Marrnet: 3D shape reconstruction via 2.5D sketches. In *Advances in Neural Information Processing Systems (NIPS)*. 540–550.

Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T. Freeman, and Joshua B. Tenenbaum. 2016. Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-adversarial Modeling. In *Advances in Neural Information Processing Systems (NIPS)*. 82–90.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 2015. 3d shapenets: A deep representation for volumetric shapes. In *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*. 1912–1920.

Kai Xu, Hao Zhang, Daniel Cohen-Or, and Baoquan Chen. 2012. Fit and Diverse: Set Evolution for Inspiring 3D Shape Galleries. *ACM Trans. on Graphics* 31, 4 (2012), 57:1–57:10.

Xinchen Yan, Jimei Yang, Kihyuk Sohn, and Honglak Lee. 2016. Attribute2image: Conditional image generation from visual attributes. In *Proc. Euro. Conf. on Computer Vision*. 776–791.

Li Yi, Vladimir G. Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. 2016. A Scalable Active Framework for Region Annotation in 3D Shape Collections. *ACM Trans. on Graphics* 35, 6 (2016), 210:1–210:12.

Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. 2018. Generative Image Inpainting With Contextual Attention. In *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*. 5505–5514.

Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. 2016. Generative visual manipulation on the natural image manifold. In *Proc. Euro. Conf. on Computer Vision*. 597–613.