The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Learning Efficient Point Cloud Generation for Dense 3D Object Reconstruction

Chen-Hsuan Lin, Chen Kong, Simon Lucey
The Robotics Institute
Carnegie Mellon University
chlin@cmu.edu, {chenk,slucey}@cs.cmu.edu

Abstract

Conventional methods of 3D object generative modeling learn volumetric predictions using deep networks with 3D convolutional operations, which are direct analogies to classical 2D ones. However, these methods are computationally wasteful in attempt to predict 3D shapes, where information is rich only on the surfaces. In this paper, we propose a novel 3D generative modeling framework to efficiently generate object shapes in the form of dense point clouds. We use 2D convolutional operations to predict the 3D structure from multiple viewpoints and jointly apply geometric reasoning with 2D projection optimization. We introduce the pseudo-renderer, a differentiable module to approximate the true rendering operation, to synthesize novel depth maps for optimization. Experimental results for single-image 3D object reconstruction tasks show that we outperforms state-of-the-art methods in terms of shape similarity and prediction density.

Introduction

Generative models using convolutional neural networks (CNNs) have achieved state of the art in image/object generation problems. Notable works of the class include variational autoencoders (Kingma and Welling 2013) and generative adversarial networks (Goodfellow et al. 2014), both of which have drawn large success in various applications (Isola et al. 2017; Radford, Metz, and Chintala 2015; Zhu et al. 2016; Wang and Gupta 2016; Yan et al. 2016a). With the recent introduction of large publicly available 3D model repositories (Wu et al. 2015; Chang et al. 2015), the study of generative modeling on 3D data using similar frameworks has also become of increasing interest.

In computer vision and graphics, 3D object models can take on various forms of representations. Of such, triangular meshes and point clouds are popular for their vectorized (and thus scalable) data representations as well as their compact encoding of shape information, optionally embedded with texture. However, this efficient representation comes with an inherent drawback as the dimensionality per 3D shape sample can vary, making the application of learning methods problematic. Furthermore, such data representations do not elegantly fit within conventional CNNs as Euclidean convolutional operations cannot be directly applied. Hitherto, most existing works on 3D model generation resort to volumetric representations, allowing 3D Euclidean convolution to operate on regular discretized voxel grids. 3D CNNs (as opposed to the classical 2D form) have been applied successfully to 3D volumetric representations for both discriminative (Wu et al. 2015; Maturana and Scherer 2015; Hegde and Zadeh 2016) and generative (Girdhar et al. 2016; Choy et al. 2016; Yan et al. 2016b; Wu et al. 2016) problems.

Despite their recent success, 3D CNNs suffer from an inherent drawback when modeling shapes with volumetric representations. Unlike 2D images, where every pixel contains meaningful spatial and texture information, volumetric representations are information-sparse. More specifically, a 3D object is expressed as a voxel-wise occupancy grid, where voxels "outside" the object (set to off) and "inside" the object (set to on) are unimportant and fundamentally not of particular interest. In other words, the richest information of shape representations lies on the surface of the 3D object, which makes up only a slight fraction of all voxels in an occupancy grid. Consequently, 3D CNNs are extremely wasteful in both computation and memory in trying to predict much unuseful data with high-complexity 3D convolutions, severely limiting the granularity of the 3D volumetric shapes that can be modeled even on high-end GPU-nodes commonly used in deep learning research.

In this paper, we propose an efficient framework to represent and generate 3D object shapes with dense point clouds. We achieve this by learning to predict the 3D structures from multiple viewpoints, which is jointly optimized through 3D geometric reasoning. In contrast to prior art that adopts 3D CNNs to operate on volumetric data, we leverage *2D convolutional operations* to predict point clouds that shape the *surface* of the 3D objects. Our experimental results show that we generate much denser and more accurate shapes than state-of-the-art 3D prediction methods.

Our contributions are summarized as follows:

- We advocate that deep networks with 2D convolutional operations are capable of generating dense point clouds that shapes the surface of 3D objects in an undiscretized 3D space.
- We introduce a pseudo-rendering pipeline to serve as a differentiable approximation of true rendering. We further utilize the pseudo-rendered depth images for

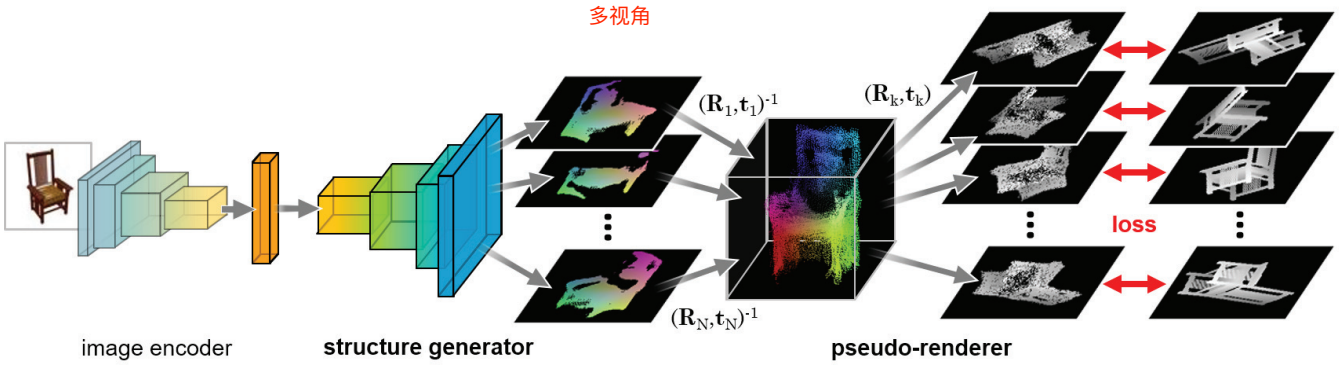

Figure 1: *Pipeline.* From an encoded latent representation, we propose to use a structure generator, which is based on *2D convolutional* operations, to predict the 3D structure at N viewpoints. The point clouds are fused by transforming the 3D structure at each viewpoint to the canonical coordinates. The pseudo-renderer synthesizes depth images from novel viewpoints, which are further used for joint 2D projection optimization. This contains no learnable parameters and reasons based purely on 3D geometry.

2D projection optimization for learning to generate dense 3D shapes.

- We demonstrate the efficacy of our method on single-image 3D reconstruction problems, which significantly outperforms state-of-the-art methods.

Related Work

3D shape generation. As 2D CNNs have demonstrated huge success on a myriad of image generation problems, most works on 3D shape generation follow the analogue using 3D CNNs to generate volumetric shapes. Prior works include using 3D autoencoders (Girdhar et al. 2016) and recurrent networks (Choy et al. 2016) to learn a latent representation for volumetric data generation. Similar applications include the use of an additional encoded pose embedding to learn shape deformations (Yumer and Mitra 2016) and using adversarial training to learn more realistic shape generation (Wu et al. 2016; Gadelha, Maji, and Wang 2016). Learning volumetric predictions from 2D projected observations has also been explored (Yan et al. 2016b; Rezende et al. 2016; Gadelha, Maji, and Wang 2016), which use 3D differentiable sampling on voxel grids for spatial transformations (Jaderberg et al. 2015). Constraining the ray consistency of 2D observations have also been suggested very recently (Tulsiani et al. 2017).

Most of the above approaches utilize 3D convolutional operations, which is computationally expensive and allows only coarse 3D voxel resolution. The lack of granularity from such volumetric generation has been an open problem following these works. Riegler, Ulusoys, and Geiger (2017) proposed to tackle the problem by using adaptive hierarchical octary trees on voxel grids to encourage encoding more informative parts of 3D shapes. Concurrent works follow to use similar concepts (Häne, Tulsiani, and Malik 2017; Tatarchenko, Dosovitskiy, and Brox 2017) to predict 3D volumetric data with higher granularity.

Recently, Fan, Su, and Guibas (2017) also sought to generate unordered point clouds by using variants of multi-layer perceptrons to predict multiple 3D coordinates. However, the required learnable parameters linearly proportional to the number of 3D point predictions and does not scale well; in addition, using 3D distance metrics as optimization criteria is intractable for large number of points. In contrast, we leverage convolutional operations with a joint 2D project criterion to capture the correlation between generated point clouds and optimize in a more computationally tractable fashion.

3D view synthesis. Research has also been done in learning to synthesize novel 3D views of 2D objects in images. Most approaches using CNNs follow the convention of an encoder-decoder framework. This has been explored by mixing 3D pose information into the latent embedding vector for the synthesis decoder (Tatarchenko, Dosovitskiy, and Brox 2016; Zhou et al. 2016; Park et al. 2017). A portion of these works also discussed the problem of disentangling the 3D pose representation from object identity information (Kulkarni et al. 2015; Yang et al. 2015; Reed et al. 2015), allowing further control on the identity representation space.

The drawback of these approaches is their inefficiency in representing 3D geometry — as we show in the experiments, one should explicitly factorize the underlying 3D geometry instead of implicitly encoding it into mixed representations. Resolving the geometry has been proven more efficient than tolerating in several works (*e.g.* Spatial Transformer Networks (Jaderberg et al. 2015; Lin and Lucey 2017)).

Approach

Our goal is to generate 3D predictions that compactly shape the surface geometry with dense point clouds. The overall pipeline is illustrated in Fig. 1. We start with an encoder that maps the input data to a latent representation space. The encoder may take on various forms of data depending on the application; in our experiments, we focus on encoding RGB

images for single-image 3D reconstruction tasks. From the latent representation, we propose to generate the dense point clouds using a structure generator based on 2D convolutions with a joint 2D projection criterion, described in detail as follows.

Structure Generator

The structure generator predicts the 3D structure of the object at N different viewpoints (along with their binary masks), *i.e.* the 3D coordinates $\hat{\mathbf{x}}_i = [\hat{x}_i \ \hat{y}_i \ \hat{z}_i]^\top$ at each pixel location. Pixel values in natural images can be synthesized through convolutional generative models mainly due to their exhibition of strong local spatial dependencies; similar phenomenons can be observed for point clouds when treating them as (x, y, z) multi-channel images on a 2D grid. Based on this insight, the structure generator is mainly based on 2D convolutional operations to predict the (x, y, z) images representing the 3D surface geometry. This approach circumvents the need of time-consuming and memory-expensive 3D convolutional operations for volumetric predictions. The evidence of such validity is verified in our experimental results.

Assuming the 3D rigid transformation matrices of the N viewpoints $(\mathbf{R}_1, \mathbf{t}_1)...(\mathbf{R}_N, \mathbf{t}_N)$ are given a priori, each 3D point $\hat{\mathbf{x}}_i$ at viewpoint n can be transformed to the canonical 3D coordinates as $\hat{\mathbf{p}}_i$ via

$$\hat{\mathbf{p}}_i = \mathbf{R}_n^{-1} \left(\mathbf{K}^{-1} \hat{\mathbf{x}}_i - \mathbf{t}_n \right) \quad \forall i \ , \tag{1}$$

where $\mathbf{K}$ is a camera calibration matrix to rescale and reposition the point cloud between 3D coordinate systems. This defines the relationship between the predicted 3D points $\{\hat{\mathbf{x}}_i\}$ and the fused collection of point clouds $\{\hat{\mathbf{p}}_i\}$ in the canonical 3D coordinates, which is our final product.

Joint 2D Projection Optimization

To learn point cloud generation using the provided 3D CAD models as supervision, the standard approach would be to optimize over a 3D-based metric that defines the distance between the point cloud and the ground-truth CAD model (*e.g.* Chamfer distance (Fan, Su, and Guibas 2017)). However, such metric would involve computing the projection of every generated point onto the triangular mesh, which can be computationally expensive for very dense predictions, making this approach infeasible. Computing surface projections can sometimes be accelerated using approximate nearest-neighbor methods such as KD-trees; however, this usually requires the vertices of the CAD model to be distributed to some uniformity. We find that this method fails for a large portion of the CAD models, especially with unevenly distributed vertices.

We overcome this issue by alternatively optimizing over the *joint 2D projection error* of novel viewpoints. Instead of using only projected binary masks as supervision (Yan et al. 2016b; Gadelha, Maji, and Wang 2016; Rezende et al. 2016), we conjecture that a well-generated 3D shape should also have the ability to render reasonable depth images from any viewpoint. To realize this concept, we introduce the pseudo-renderer, a differentiable module to approximate true rendering, to synthesize novel depth images from dense point clouds. We find this type of 2D-based optimization to run about 100× faster than 3D-based optimization such as Chamfer distance.

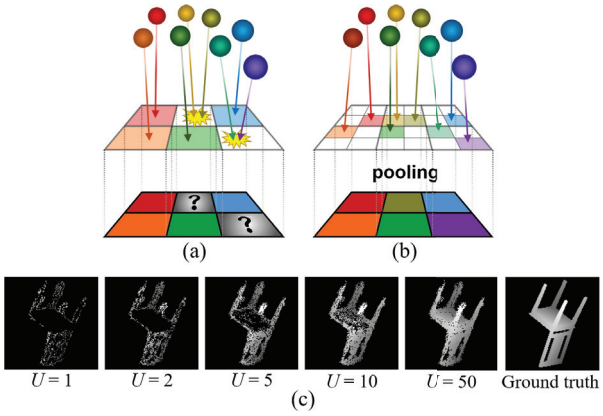

Figure 2: *Concept of pseudo-rendering.* Multiple transformed 3D points may correspond to projection on the same pixels in the image space. (a) Collision could easily occur if $(\hat{x}'_i, \hat{y}'_i)$ were directly discretized. (b) Upsampling the target image increases the precision of the projection locations and thus alleviates the collision effect. A max-pooling operation on the inverse depth values follows as to maintain the effective depth value in each pixel at the original resolution. (c) Examples of pseudo-rendered depth images with various upsampling factors U (only valid depth values without collision are shown). Pseudo-rendering achieves closer performance to true rendering with a higher value of U.

Pseudo-rendering. The forward process of rendering is an old and well-developed concept in computer graphics. True rendering is typically achieved by using a Z-buffer at every pixel location to maintain the effective visible value (RGB or depth) from the camera. Although this is parallelizable and can be run efficiently on GPUs, it is generally not differentiable and cannot be directly utilized and incorporated into a deep learning framework. Here, we present a solution to a differentiable approximation of such operation.

Given the 3D rigid transformation matrix of a novel viewpoint $(\mathbf{R}_k, \mathbf{t}_k)$, each canonical 3D point $\hat{\mathbf{p}}_i$ can be further transformed to $\hat{\mathbf{x}}'_i$ back in the image coordinates via

$$\hat{\mathbf{x}}'_i = \mathbf{K} \left(\mathbf{R}_k \hat{\mathbf{p}}_i + \mathbf{t}_k \right) \quad \forall i \ . \tag{2}$$

This is the inverse operation of Eq. (1) with different transformation matrices and can be combined with Eq. (1) together, composing a single effective transformation. By such, we obtain the $(\hat{x}'_i, \hat{y}'_i)$ location as well as the new depth value $\hat{z}'_i$ at viewpoint k.

To produce a pixelated depth image, one would also need to discretize all $(\hat{x}'_i, \hat{y}'_i)$ coordinates, resulting in possibly multiple transformed points projecting and "colliding" onto the same pixel (Fig. 2). We resolve this issue with the pseudo-renderer $f_{\text{PR}}(\cdot)$, which increases the projection resolution to alleviate such collision effect. Specifically, $\hat{\mathbf{x}}'_i$ is

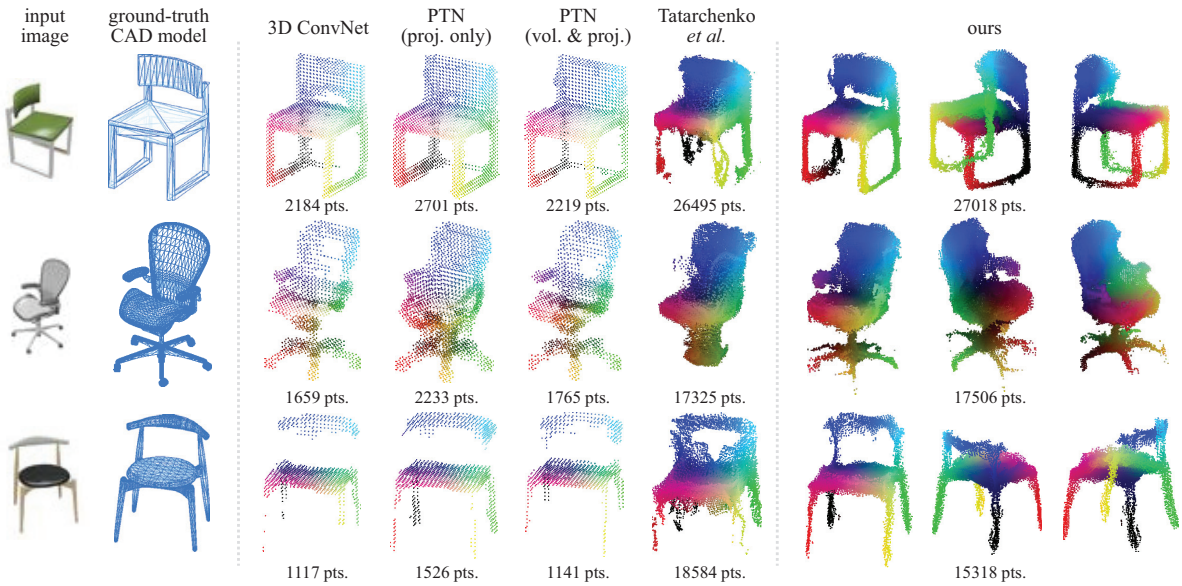

Figure 3: *Qualitative results* from the single-category experiment. Our method generates denser predictions compared to the volumetric baselines and more accurate shapes than Tatarchenko, Dosovitskiy, and Brox (2016), which learns 3D synthesis implicitly. The RGB values of the point cloud represents the 3D coordinate values. Best viewed in color.

projected onto a target image upsampled by a factor of U, reducing the quantization error of $(\hat{x}'_i, \hat{y}'_i)$ as well as the probability of collision occurrence. A max-pooling operation on the inverse depth values with kernel size U follows to downsample back to the original resolution while maintaining the minimum depth value at each pixel location. We use such approximation of the rendering operation to maintain differentiability and parallelizability within the backpropagation framework.

Optimization. We use the pseudo-rendered depth images $\hat{\mathbf{Z}} = f_{\text{PR}}(\{\hat{\mathbf{x}}'_i\})$ and the resulting masks $\hat{\mathbf{M}}$ at novel viewpoints for optimization. The loss function consists of the mask loss $\mathcal{L}_{\text{mask}}$ and the depth loss $\mathcal{L}_{\text{depth}}$, defined as

$$\mathcal{L}_{\text{mask}} = \sum_{k=1}^{K} -\mathbf{M}_k \log \hat{\mathbf{M}}_k - (1-\mathbf{M}_k) \log\left(1-\hat{\mathbf{M}}_k\right)$$
$$\mathcal{L}_{\text{depth}} = \sum_{k=1}^{K} \left\| \hat{\mathbf{Z}}_k - \mathbf{Z}_k \right\|_1 , \tag{3}$$

where we simultaneously optimize over K novel viewpoints at a time. $\mathbf{M}_k$ and $\mathbf{Z}_k$ are the ground-truth mask and depth images at the kth novel viewpoint. We use element-wise L_1 loss for the depth (posing it as a pixel-wise binary classification problem) and cross-entropy loss for the mask. In order to maintain parallelizability, we choose to sum up the colliding depth values in the same pixel and mask them out during optimization. The overall loss function is defined as $\mathcal{L} = \mathcal{L}_{\text{mask}} + \lambda \cdot \mathcal{L}_{\text{depth}}$, where λ is the weighting factor.

Optimizing the structure generator over novel projections enforces joint 3D geometric reasoning between the predicted point clouds from the N viewpoints. It also allows the optimization error to evenly distribute across novel viewpoints instead of focusing on the fixed N viewpoints.

Experiments

We evaluate our proposed method by analyzing its performance in the application of single-image 3D reconstruction and comparing against state-of-the-art methods.

Data preparation. We train and evaluate all networks using the ShapeNet database (Chang et al. 2015), which contains a large collection of categorized 3D CAD models. For each CAD model, we pre-render 100 depth/mask image pairs of size 128×128 at random novel viewpoints as the ground truth of the loss function. We consider the entire space of possible 3D rotations (including in-plane rotation) for the viewpoints and assume identity translation for simplicity. The input images are objects pre-rendered from a fixed elevation and 24 different azimuth angles.

Settings. The structure generator predicts a $4N$-channel image, which consists of the x, y, z coordinates and the binary mask from each of the N fixed viewpoint. We chose $N = 8$ with those viewpoints looking from the 8 corners of a centered cube. Orthographic projection is assumed in the transformation in (1) and (2). We take a two-stage training procedure: the structure generator is first pretrained to predict the x, y regular grids and depth images (z) from the N viewpoints (also pre-rendered with size 128×128), and then the network is fine-tuned with joint 2D projection optimization. Please refer to the supplementary materials for more architectural and training details.

Method	3D error metric	
	pred. $\rightarrow$ GT	GT $\rightarrow$ pred.
3D CNN (vol. loss only)	1.827	2.660
PTN (proj. loss only)	2.181	2.170
PTN (vol. & proj. losses)	1.840	2.585
Tatarchenko et al.	2.381	3.019
Proposed method	**1.768**	**1.763**

Table 1: *Average 3D test error* of the single-category experiment. Our method outperforms all baselines in both metrics, indicating the superiority in fine-grained shape similarity and point cloud coverage on the surface. (All numbers are scaled by 100)

Quantitative metrics. We present quantitative results using the average point-wise 3D Euclidean distance between two 3D models: for each point $\hat{\mathbf{p}}_i$ in the source model, the distance to the target model $\mathcal{S}$ is defined as $\mathcal{E}_i = \min_{\mathbf{p}_j \in \mathcal{S}} \|\hat{\mathbf{p}}_i - \mathbf{p}_j\|_2$. This metric is defined bidirectionally as the distance from the predicted point cloud to the ground-truth CAD model and vice versa. It is necessary to report both metrics for they represent different aspects of quality — the former measures 3D shape similarity and the latter measures surface coverage (Kong, Lin, and Lucey 2017). We represent the ground-truth CAD models as collections of uniformly densified 3D points on the surfaces (100K densified points in our settings).

Single Object Category

We start by evaluating the efficacy of our dense point cloud representation on 3D reconstruction for a single object category. We use the chair category from ShapeNet, which consists of 6,778 CAD models. We compare against (a) Tatarchenko, Dosovitskiy, and Brox (2016), which learns implicit 3D representations through a mixed embedding, and (b) Perspective Transformer Networks (PTN) (Yan et al. 2016b), which learns to predict volumetric data by minimizing the projection error. We include two variants of PTN as well as a baseline 3D CNN from Yan et al. (2016b). We use the same 80%-20% training/test split provided by Yan et al. (2016b).

For the method of Tatarchenko, Dosovitskiy, and Brox (2016), we evaluate by predicting depth images from our same N viewpoints and transform the resulting point clouds to the canonical coordinates. This shares the same network architecture to ours, but with 3D pose information additionally encoded using 3 linear layers (with 64 filters) and concatenated with the latent vector. We use the novel depth/mask pairs as direct supervision for the decoder output. For PTN (Yan et al. 2016b), we extract the surface voxels (by subtracting the prediction by its eroded version) and rescale them such that the tightest 3D bounding boxes of the prediction and the ground-truth CAD models have the same volume. We use the pretrained models readily provided by the authors.

The quantitative results on the test split are reported in Table 1. We achieve a lower average 3D distance than all baselines in both metrics, even though our approach is optimized with joint 2D projections instead of these 3D error metrics. This demonstrates that we are capable of predicting more accurate shapes with higher density and finer granularity. This highlights the efficiency of our approach using 2D CNNs to generate 3D shapes compared to 3D CNN methods such as PTN (Yan et al. 2016b) as they attempt to predict all voxel occupancies inside a 3D grid space. Compared to Tatarchenko, Dosovitskiy, and Brox (2016), an important takeaway is that 3D geometry should explicitly factorized when possible instead of being implicitly learned by the network parameters. It is much more efficient to focus on predicting the geometry from a sufficient number of viewpoints and combining them with known geometric transformations.

We visualize the generated 3D shapes in Fig. 3. Compared to the baselines, we predict more accurate object structures with a much higher point cloud density (around $10\times$ higher than 32^3 volumetric methods). This further highlights the desirability of our approach — we are able to efficiently use 2D convolutional operations and utilize high-resolution supervision given similar memory budgets.

General Object Categories

We also evaluate our network on the single-image 3D reconstruction task trained with multiple object categories. We compare against (a) 3D-R2N2 (Choy et al. 2016), which learns volumeric predictions through recurrent networks, and (b) Fan, Su, and Guibas (2017), which predicts an unordered set of 1024 3D points. We use 13 categories of ShapeNet for evaluation (listed in Table 2), where the 80%-20% training/test split is provided by Choy et al. (2016). We evaluate 3D-R2N2 by its surface voxels using the same procedure as described in the single-category experiment. For all baselines, we use the pretrained models readily provided by the authors.

We list the quantitative results in Table 2, where the metrics are reported per-category. Our method achieves an overall lower error in both metrics. We outperform the volumetric baselines (3D-R2N2) by a large margin and has better prediction performance than Fan, Su, and Guibas (2017) in most cases. We also visualize the predictions in Fig. 4; again we see that our method predicts more accurate shapes with higher point density. We find that our method can be more problematic when objects contain very thin structures (*e.g.* lamps); adding hybrid linear layers (Fan, Su, and Guibas 2017) may help improve performance.

Generative Representation Analysis

We analyze the learned generative representations by observing the 3D predictions from manipulation in the latent space. Previous works have demonstrated that deep generative networks can generate meaningful pixel/voxel predictions by performing linear operations in the latent space (Radford, Metz, and Chintala 2015; Dosovitskiy, Tobias Springenberg, and Brox 2015; Wu et al. 2016); here, we explore the possibility of such manipulation for dense point clouds in an undiscretized space.

We show in Fig. 5 the resulting dense shapes generated from the embedding vector interpolated in the latent space.

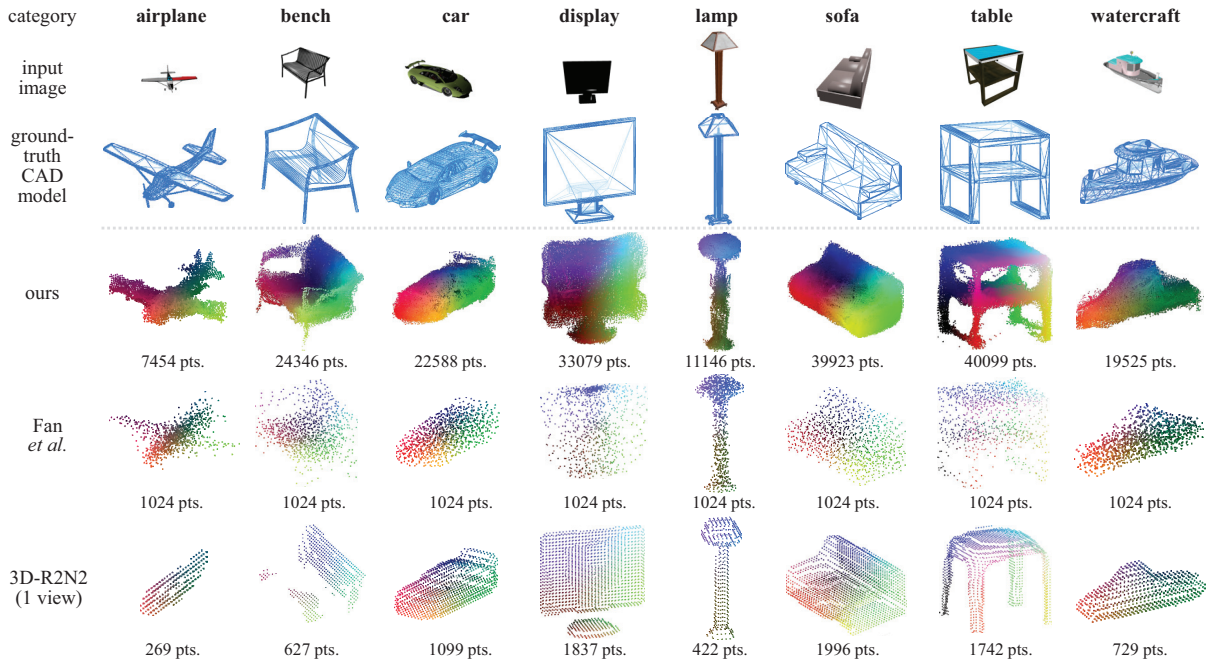

Figure 4: *Qualitative results* from the multi-category experiment. Our method generates denser and more accurate predictions compared to the baselines.

Category	3D-R2N2			Fan et al.	Proposed
	1 view	3 views	5 views	(1 view)	(1 view)
airplane	3.207 / 2.879	2.521 / 2.468	2.399 / 2.391	1.301 / **1.488**	**1.294** / 1.541
bench	3.350 / 3.697	2.465 / 2.746	2.323 / 2.603	1.814 / 1.983	**1.757** / **1.487**
cabinet	1.636 / 2.817	1.445 / 2.626	**1.420** / 2.619	2.463 / 2.444	1.814 / **1.072**
car	1.808 / 3.238	1.685 / 3.151	1.664 / 3.146	1.800 / 2.053	**1.446** / **1.061**
chair	2.759 / 4.207	1.960 / 3.238	**1.854** / 3.080	1.887 / 2.355	1.886 / **2.041**
display	3.235 / 4.283	2.262 / 3.151	2.088 / 2.953	**1.919** / 2.334	2.142 / **1.440**
lamp	8.400 / 9.722	6.001 / 7.755	5.698 / 7.331	**2.347** / **2.212**	2.635 / 4.459
loudspeaker	2.652 / 4.335	2.577 / 4.302	2.487 / 4.203	3.215 / 2.788	**2.371** / **1.706**
rifle	4.798 / 2.996	4.307 / 2.546	4.193 / 2.447	1.316 / **1.358**	**1.289** / 1.510
sofa	2.725 / 3.628	2.371 / 3.252	2.306 / 3.196	2.592 / 2.784	**1.917** / **1.423**
table	3.118 / 4.208	2.268 / 3.277	2.128 / 3.134	1.874 / 2.229	**1.689** / **1.620**
telephone	2.202 / 3.314	1.969 / 2.834	1.874 / 2.734	**1.516** / 1.989	1.939 / **1.198**
watercraft	3.592 / 4.007	3.299 / 3.698	3.210 / 3.614	**1.715** / 1.877	1.813 / **1.550**
mean	3.345 / 4.102	2.702 / 3.465	2.588 / 3.342	1.982 / 2.146	**1.846** / **1.701**

Table 2: *Average 3D test error* of the multi-category experiment, where the numbers are shown as [prediction→GT / GT→prediction]. The mean is computed across categories. For the single-view case, we outperform all baselines in 8 and 10 out of 13 categories for the two 3D error metrics. (All numbers are scaled by 100)

The morphing transition is smooth with plausible interpolated shapes, which suggests that our structure generator can generate meaningful 3D predictions from convex combinations of encoded latent vectors. The structure generator is also capable of generating reasonable novel shapes from arithmetic results in the latent space — from Fig. 6 we observe semantic feature replacement of table height/shape as well as chair arms/backs. These results suggest that the high-level semantic information encoded in the latent vectors are manipulable and interpretable of the resulting dense point clouds through the structure generator.

Ablative Analysis

We provide ablation studies on two of the key components of our proposed method: (1) the joint 2D optimization step and (2) the variability of the x, y coordinates of the structure generator output. We focus this analysis on the single-category experiment.

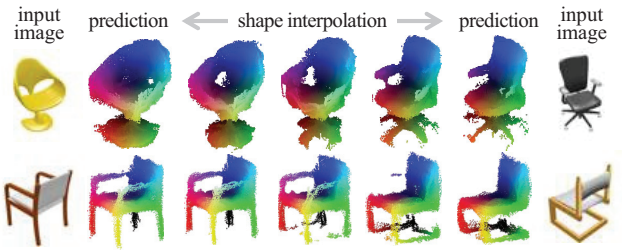

Figure 5: Shapes generated from interpolated latent embeddings of two input images (leftmost and rightmost). The interpolated shapes maintain reasonable chair structures.

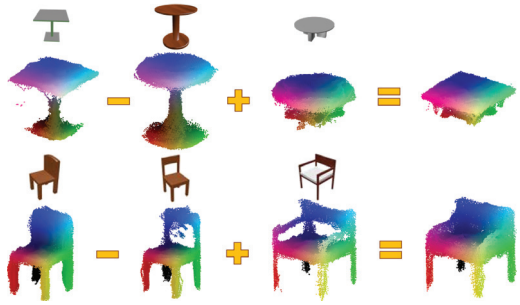

Figure 6: Dense shapes generated from arithmetic operations in the latent space (left: tables, right: chairs), where the input images are shown in the top row.

Joint 2D opt.	3D error metric	Predicted points
Before	1.933 / 1.307	31,972
After	1.768 / 1.763	25,401

Table 3 & Figure 7: Comparison on the effects of the joint 2D optimization step (left figure: before, right figure: after). Optimizing only on the fixed viewpoints results in a denser point cloud but also with higher noise. Optimizing on the novel viewpoints greatly reduces the noise while trading off partial surface coverage density. (The errors are shown as [prediction→GT / GT→prediction] and scaled by 100)

Joint 2D optimization. We validate the necessity of the second training stage of optimizing the network with supervision from novel viewpoints. We compare the performance of our network before and after the joint 2D projection optimization step in Table 3. We see that while optimizing only on the fixed viewpoints results in more generated points closer to the ground-truth surface, it also creates a considerable amount of noisy points in loss of shape accuracy. Fig. 7 visualizes the effect of joint optimization to eliminate most of the noisy points, demonstrating the necessity of such additional step.

Variability of x, y coordinates. Instead of having the structure generator to predict the x, y, z coordinates and the binary masks, one could alternatively design it to predict only the masked depth image (*i.e.* the z coordinates and the mask) and fix the x, y coordinates to the image regular grids. We show the difference in performance in Table 4. We see that enabling the x, y coordinates to vary not only leads to similar accuracy in shape prediction, but also allows higher surface coverage. There is also little increase in the number of learnable parameters from doubling the output channels in the final convolution layer.

Output of structure generator	3D error metric pred. → GT	GT → pred.
Depth image (z) only	1.764	2.086
xyz-channel images	1.768	1.763

Table 4: Comparison on the effects of the variability of the x, y coordinates of the multi-view output. Allowing the x, y coordinates to vary improves surface coverage. (All numbers are scaled by 100)

Conclusion

In this paper, we introduced a framework for generating 3D shapes in the form of dense point clouds. Compared to conventional volumetric prediction methods using 3D CNNs, it is more efficient to utilize 2D convolutional operations to predict surface information of 3D shapes. We showed that by introducing a pseudo-renderer, we are able to synthesize approximate depth images from novel viewpoints to optimize the 2D projection error within a backpropagation framework. Experimental results for single-image 3D reconstruction tasks showed that we generate more accurate and much denser 3D shapes than state-of-the-art 3D reconstruction methods.

References

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*.

Choy, C. B.; Xu, D.; Gwak, J.; Chen, K.; and Savarese, S. 2016. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European Conference on Computer Vision*, 628–644. Springer.

Dosovitskiy, A.; Tobias Springenberg, J.; and Brox, T. 2015. Learning to generate chairs with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1538–1546.

Fan, H.; Su, H.; and Guibas, L. 2017. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Gadelha, M.; Maji, S.; and Wang, R. 2016. 3d shape induction from 2d views of multiple objects. *arXiv preprint arXiv:1612.05872*.

Girdhar, R.; Fouhey, D. F.; Rodriguez, M.; and Gupta, A. 2016. Learning a predictable and generative vector repre-

sentation for objects. In *European Conference on Computer Vision*, 484–499. Springer.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, 2672–2680.

Häne, C.; Tulsiani, S.; and Malik, J. 2017. Hierarchical surface prediction for 3d object reconstruction. *arXiv preprint arXiv:1704.00710*.

Hegde, V., and Zadeh, R. 2016. Fusionnet: 3d object classification using multiple data representations. *arXiv preprint arXiv:1607.05695*.

Isola, P.; Zhu, J.-Y.; Zhou, T.; and Efros, A. A. 2017. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Jaderberg, M.; Simonyan, K.; Zisserman, A.; et al. 2015. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, 2017–2025.

Kingma, D. P., and Welling, M. 2013. Auto-encoding variational bayes.

Kong, C.; Lin, C.-H.; and Lucey, S. 2017. Using locally corresponding cad models for dense 3d reconstructions from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Kulkarni, T. D.; Whitney, W. F.; Kohli, P.; and Tenenbaum, J. 2015. Deep convolutional inverse graphics network. In *Advances in Neural Information Processing Systems*, 2539–2547.

Lin, C.-H., and Lucey, S. 2017. Inverse compositional spatial transformer networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Maturana, D., and Scherer, S. 2015. Voxnet: A 3d convolutional neural network for real-time object recognition. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, 922–928. IEEE.

Park, E.; Yang, J.; Yumer, E.; Ceylan, D.; and Berg, A. C. 2017. Transformation-grounded image generation network for novel 3d view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Radford, A.; Metz, L.; and Chintala, S. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*.

Reed, S. E.; Zhang, Y.; Zhang, Y.; and Lee, H. 2015. Deep visual analogy-making. In *Advances in Neural Information Processing Systems*, 1252–1260.

Rezende, D. J.; Eslami, S. A.; Mohamed, S.; Battaglia, P.; Jaderberg, M.; and Heess, N. 2016. Unsupervised learning of 3d structure from images. In *Advances In Neural Information Processing Systems*, 4997–5005.

Riegler, G.; Ulusoys, A. O.; and Geiger, A. 2017. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Tatarchenko, M.; Dosovitskiy, A.; and Brox, T. 2016. Multi-view 3d models from single images with a convolutional network. In *European Conference on Computer Vision*, 322–337. Springer.

Tatarchenko, M.; Dosovitskiy, A.; and Brox, T. 2017. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. *arXiv preprint arXiv:1703.09438*.

Tulsiani, S.; Zhou, T.; Efros, A. A.; and Malik, J. 2017. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Wang, X., and Gupta, A. 2016. Generative image modeling using style and structure adversarial networks. In *European Conference on Computer Vision*, 318–335. Springer.

Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; and Xiao, J. 2015. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1912–1920.

Wu, J.; Zhang, C.; Xue, T.; Freeman, B.; and Tenenbaum, J. 2016. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in Neural Information Processing Systems*, 82–90.

Yan, X.; Yang, J.; Sohn, K.; and Lee, H. 2016a. Attribute2image: Conditional image generation from visual attributes. In *European Conference on Computer Vision*, 776–791. Springer.

Yan, X.; Yang, J.; Yumer, E.; Guo, Y.; and Lee, H. 2016b. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. In *Advances in Neural Information Processing Systems*, 1696–1704.

Yang, J.; Reed, S. E.; Yang, M.-H.; and Lee, H. 2015. Weakly-supervised disentangling with recurrent transformations for 3d view synthesis. In *Advances in Neural Information Processing Systems*, 1099–1107.

Yumer, M. E., and Mitra, N. J. 2016. Learning semantic deformation flows with 3d convolutional networks. In *European Conference on Computer Vision*, 294–311. Springer.

Zhou, T.; Tulsiani, S.; Sun, W.; Malik, J.; and Efros, A. A. 2016. View synthesis by appearance flow. In *European Conference on Computer Vision*, 286–301. Springer.

Zhu, J.-Y.; Krähenbühl, P.; Shechtman, E.; and Efros, A. A. 2016. Generative visual manipulation on the natural image manifold. In *European Conference on Computer Vision*, 597–613. Springer.