Dense 3D Object Reconstruction from a Single Depth View

Bo Yang, Stefano Rosa, Andrew Markham, Niki Trigoni, Hongkai Wen*

Abstract—In this paper, we propose a novel approach, **3D-RecGAN++**, which reconstructs the complete 3D structure of a given object from a single arbitrary depth view using generative adversarial networks. Unlike existing work which typically requires multiple views of the same object or class labels to recover the full 3D geometry, the proposed 3D-RecGAN++ only takes the voxel grid representation of a depth view of the object as input, and is able to generate the complete 3D occupancy grid with **a high resolution of $\mathbf{256^3}$** by recovering the occluded/missing regions. The key idea is to combine the generative capabilities of 3D encoder-decoder and the conditional adversarial networks framework, to infer accurate and fine-grained 3D structures of objects in high-dimensional voxel space. Extensive experiments on large synthetic datasets and real-world Kinect datasets show that the proposed 3D-RecGAN++ significantly outperforms the state of the art in single view 3D object reconstruction, and is able to reconstruct unseen types of objects.

Index Terms—3D Reconstruction, Shape Completion, Shape inpainting, Single Depth View, Adversarial Learning, Conditional GAN.

1 Introduction

To reconstruct the complete and precise 3D geometry of an object is essential for many graphics and robotics applications, from AR/VR [1] and semantic understanding, to object deformation [2], robot grasping [3] and obstacle avoidance. Classic approaches use the off-the-shelf low-cost depth sensing devices such as Kinect and RealSense cameras to recover the 3D shape of an object from captured depth images. Those approaches typically require multiple depth images from different viewing angles of an object to estimate the complete 3D structure [4] [5] [6]. However, in practice it is not always feasible to scan all surfaces of an object before reconstruction, which leads to incomplete 3D shapes with occluded regions and large holes. In addition, acquiring and processing multiple depth views require more computing power, which is not ideal in many applications that require real-time performance.

We aim to tackle the problem of estimating the complete 3D structure of an object using a single depth view. This is a very challenging task, since the partial observation of the object (*i.e.*, a depth image from one viewing angle) can be theoretically associated with an infinite number of possible 3D models. Traditional reconstruction approaches typically use interpolation techniques such as plane fitting, Laplacian hole filling [7] [8], or Poisson surface estimation [9] [10] to infer the underlying 3D structure. However, they can only recover very limited occluded or missing regions, *e.g.*, small holes or gaps due to quantization artifacts, sensor noise and insufficient geometry information.

Interestingly, humans are surprisingly good at solving such ambiguity by implicitly leveraging prior knowledge. For example, given a view of a chair with two rear legs occluded by front legs, humans are easily able to guess the most likely shape behind the visible parts. Recent advances in deep neural networks and data driven approaches show promising results in dealing with such a task.

In this paper, we aim to acquire the complete and high-resolution 3D shape of an object given a single depth view. By leveraging the high performance of 3D convolutional neural nets and large open datasets of 3D models, our approach learns a smooth function that maps a 2.5D view to a complete and dense 3D shape. In particular, we train an end-to-end model which estimates full volumetric occupancy from a single 2.5D depth view of an object.

While state-of-the-art deep learning approaches [11] [12] [3] for 3D shape reconstruction from a single depth view achieve encouraging results, they are limited to very small resolutions, typically at the scale of 32^3 voxel grids. As a result, the learnt 3D structure tends to be coarse and inaccurate. In order to generate higher resolution 3D objects with efficient computation, Octree representation has been recently introduced in [13] [14] [15]. However, increasing the density of output 3D shapes would also inevitably pose a great challenge to learn the geometric details for high resolution 3D structures, which has yet to be explored.

Recently, deep generative models achieve impressive success in modeling complex high-dimensional data distributions, among which Generative Adversarial Networks (GANs) [16] and Variational Autoencoders (VAEs) [17] emerge as two powerful frameworks for generative learning, including image and text generation [18] [19], and latent space learning [20] [21]. In the past few years, a number of works [22] [23] [24] [25] applied such generative models to learn latent space to represent 3D object shapes, in order to solve tasks such as new image generation, object classification, recognition and shape retrieval.

In this paper, we propose 3D-RecGAN++, a simple yet effective model that combines a skip-connected 3D encoder-

- Bo Yang, Stefano Rosa, Andrew Markham and Niki Trigoni are with the Department of Computer Science, University of Oxford, UK.
E-mail:{bo.yang,stefano.rosa,andrew.markham,niki.trigoni}@cs.ox.ac.uk
- Corresponding to Hongkai Wen, who is with the Department of Computer Science, University of Warwick, UK.
E-mail: hongkai.wen@dcs.warwick.ac.uk

decoder with adversarial learning to generate a complete and fine-grained 3D structure conditioned on a single 2.5D view. Particularly, our model firstly encodes the 2.5D view to a compressed latent representation which implicitly represents general 3D geometric structures, then decodes it back to the most likely full 3D shape. Skip-connections are applied between the encoder and decoder to preserve high frequency information. The rough 3D shape is then fed into a conditional discriminator which is adversarially trained to distinguish whether the coarse 3D structure is plausible or not. The encoder-decoder is able to approximate the corresponding shape, while the adversarial training tends to add fine details to the estimated shape. To ensure the final generated 3D shape corresponds to the input single partial 2.5D view, adversarial training of our model is based on a conditional GAN [26] instead of random guessing. The above network excels the competing approaches [3] [12] [27], which either use a single fully connected layer [3], a low capacity decoder without adversarial learning [12], or the multi-stage and ineffective LSTMs [27] to estimate the full 3D shapes.

Our contributions are as follows:

(1) We propose a simple yet effective model to reconstruct the complete and accurate 3D structure using a single arbitrary depth view. Particularly, our model takes a simple occupancy grid map as input without requiring object class labels or any annotations, while predicting a compelling shape within a high resolution of 256^3 voxel grid. By drawing on both 3D encoder-decoder and adversarial learning, our approach is end-to-end trainable with high level of generality.

(2) We exploit conditional adversarial training to refine the 3D shape estimated by the encoder-decoder. Our contribution here is that we use the mean value of a latent vector feature, instead of a single scalar, as the output of the discriminator to stabilize GAN training.

(3) We conduct extensive experiments for single category and multi-category object reconstruction, outperforming the state of the art. Importantly, our approach is also able to generalize to previously unseen object categories. At last, our model also performances robustly on real-world data, after being trained purely on synthetic datasets.

(4) To the best of our knowledge, there are no good open datasets which have the ground truth for occluded/missing parts and holes for each 2.5D view in real-world scenarios. We therefore collect and release our real-world testing dataset to the community.

A preliminary version of this work has been published in ICCV 2017 workshops [28]. Our code and data are available at: *https://github.com/Yang7879/3D-RecGAN-extended*

2 Related Work

We review different pipelines for 3D reconstruction or shape completion. Both conventional geometry based techniques and the state of the art deep learning approaches are covered.

(1) **3D Model/Shape Completion.** Monszpart *et al.* use plane fitting to complete small missing regions in [29], while shape symmetry is applied in [30] [31] [32] [33] [34] to fill in holes. Although these methods show good results, relying on predefined geometric regularities fundamentally limits the structure space to hand-crafted shapes. Besides, these approaches are likely to fail when missing or occluded regions are relatively big. Another similar fitting pipeline is to leverage database priors. Given a partial shape input, an identical or most likely 3D model is retrieved and aligned with the partial scan in [35] [36] [37] [38] [39] [40]. However, these approaches explicitly assume the database contains identical or very similar shapes, thus being unable to generalize to novel objects or categories.

(2) **Multiple RGB/Depth Images Reconstruction.** Traditionally, 3D dense reconstruction in SfM and visual SLAM requires a collection of RGB images [41]. Geometric shape is recovered by dense feature extraction and matching [42], or by directly minimizing reprojection errors [43] from color images. Shape priors are also concurrently leveraged with the traditional multi-view reconstruction for dense object shape estimation in [44] [45] [46]. Recently, deep neural nets are designed to learn the 3D shape from multiple RGB images in [47] [48] [49] [50] [51] [52]. However, resolution of the recovered occupancy shape is usually up to a small scale of 32^3. With the advancement of depth sensors, depth images are also used to recover the object shape. Classic approaches usually fuse multiple depth images through iterative closest point (ICP) algorithms [4] [53] [54], while recent work [14] learns the 3D shape using deep neural nets from multiple depth views.

(3) **Single RGB Image Reconstruction.** Predicting a complete 3D object model from a single view is a long-standing and extremely challenging task. When reconstructing a specific object category, model templates can be used. For example, morphable 3D models are exploited for face recovery [55] [56]. This concept was extended to reconstruct simple objects in [57]. For general and complex object reconstruction from a single RGB image, recent works [58] [59] [60] aim to infer 3D shapes using multiple RGB images for weak supervision. Shape prior knowledge is utilized in [61] [62] [63] for shape estimation. To recover high resolution 3D shapes, Octree representation is introduced in [13] [14] [15] to save computation, while an inverse discrete cosine transform (IDCT) technique is proposed in [64]. Lin *et al.* [65] designed a pseudo-renderer to predict dense 3D shapes, while 2.5D sketches and dense 3D shapes are sequentially estimated from a single RGB image in [66].

(4) **Single Depth View Reconstruction.** The task of reconstruction from a single depth view is to complete the occluded 3D structures behind the visible parts. 3D ShapeNets [11] is among the early work using deep neural nets to estimate 3D shapes from a single depth view. Firman *et al.* [67] trained a random decision forest to infer unknown voxels. Originally designed for shape denoising, VConv-DAE [1] can also be used for shape completion. To facilitate robotic grasping, Varley *et al.* proposed a neural network to infer the full 3D shape from a single depth view in [3]. However, all these approaches are only able to generate low resolution voxel grids which are less than 40^3 and unlikely to capture fine geometric details. Recent works [12] [68] [27] [69] can infer higher resolution 3D shapes. However, the pipeline in [12] relies on a shape database to synthesize a higher resolution shape after learning a small 32^3 voxel grid from a depth view, while SSCNet [68] requires voxel-level

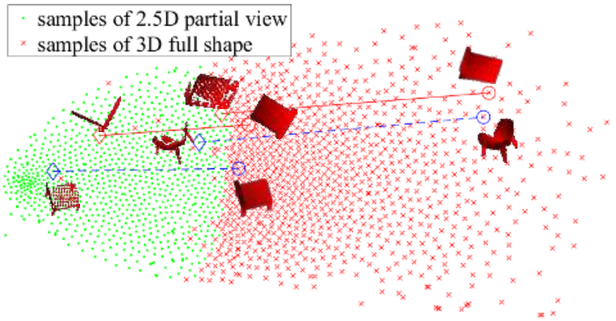


Fig. 1: t-SNE embeddings of 2.5D partial views and 3D complete shapes of multiple object categories.

annotations for supervised scene completion and semantic label prediction. Both [27] and [69] were originally designed for shape inpainting instead of directly reconstructing the complete 3D structure from a partial depth view. The recent 3D-PRNN [70] predicts simple shape primitives using RNNs, but the estimated shapes do not have finer geometric details.

(5) **Deep Generative Frameworks.** Deep generative frameworks, such as VAEs [17] and GANs [16], have achieved impressive success in image super-resolution [71], image generation [19], text to image synthesis [72], *etc.* VAE and GAN are further combined in [73] and achieve compelling results in learning visual features. Recently, generative networks are applied in [74] [75] [76] [25] to generate low resolution 3D structures. However, incorporating generative adversarial learning to estimate high resolution 3D shapes is not straightforward, as it is difficult to generate samples for high dimensional and complex data distributions [77] and this may lead to the instability of adversarial generation.

3 3D-RecGAN++

3.1 Overview

Our method aims to estimate a complete and dense 3D structure of an object, which only takes an arbitrary single 2.5D depth view as input. The output 3D shape is automatically aligned with the corresponding 2.5D partial view. To achieve this task, each object model is represented by a high resolution 3D voxel grid. We use the simple occupancy grid for shape encoding, where 1 represents an occupied cell and 0 an empty cell. Specifically, the input 2.5D partial view, denoted as $\boldsymbol{x}$, is a 64^3 occupancy grid, while the output 3D shape, denoted as $\boldsymbol{y}$, is a high resolution 256^3 probabilistic voxel grid. The input partial shape is directly calculated from a single depth image given camera parameters. We use the ground truth dense 3D shape with aligned orientation as same as the input partial 2.5D depth view to supervise our network.

To generate ground truth training and evaluation pairs, we virtually scan 3D objects from ShapeNet [78]. Figure 1 is the t-SNE visualization [79] of partial 2.5D views and the corresponding full 3D shapes for multiple general chair and bed models. Each green dot represents the t-SNE embedding of a 2.5D view, whilst a red dot is the embedding of the corresponding 3D shape. It can be seen that multiple categories inherently have similar 2.5D to 3D mapping relationships. Essentially, our neural network is to learn a smooth function, denoted as f, which maps green dots to red dots as close as possible in high dimensional space as shown in Equation 1. The function f is parametrized by neural layers in general.

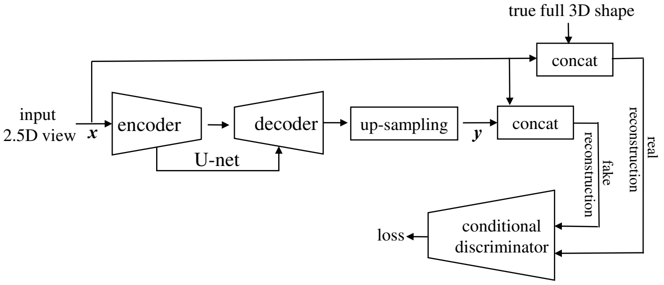


Fig. 2: Overview of the network architecture for training.

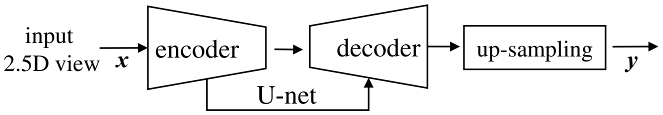


Fig. 3: Overview of the network architecture for testing.

$$\boldsymbol{y} = f(\boldsymbol{x}) \quad \left(\boldsymbol{x} \in Z^{64^3}, where\ Z = \{0,1\}\right) \tag{1}$$

After generating training pairs, we feed them into our network. The first part of our network loosely follows the idea of a 3D encoder-decoder with the U-net connections [80]. The skip-connected encoder-decoder serves as an initial coarse generator which is followed by an up-sampling module to further generate a higher resolution 3D shape within a 256^3 voxel grid. This whole generator learns a correlation between partial and complete 3D structures. With the supervision of complete 3D labels, the generator is able to learn a function f and infer a reasonable 3D shape given a brand new partial 2.5D view. During testing, however, the results tend to be grainy and without fine details.

To address this issue, in the training phase, the reconstructed 3D shape from the generator is further fed into a conditional discriminator to verify its plausibility. In particular, a partial 2.5D input view is paired with its corresponding complete 3D shape, which is called the 'real reconstruction', while the partial 2.5D view is paired with its corresponding output 3D shape from generator, which is called the 'fake reconstruction'. The discriminator aims to discriminate all 'fake reconstruction' from 'real reconstruction'. In the original GAN framework [16], the task of the discriminator is to simply classify real and fake inputs, but its Jensen-Shannon divergence-based loss function is difficult to converge. The recent WGAN [77] leverages Wasserstein distance with weight clipping as a loss function to stabilize the training procedure, whilst the extended work WGAN-GP [81] further improves the training process using a gradient penalty with respect to its input. In our 3D-RecGAN++, we apply WGAN-GP as the loss function on top of the mean feature of our conditional discriminator, which guarantees fast and stable convergence. The overall network architecture for training is shown in Figure 2, while the testing phase only needs the well trained generator as shown in Figure 3.

Overall, the main challenge of 3D reconstruction from an arbitrary single view is to generate new information including filling the missing and occluded regions from unseen views, while keeping the estimated 3D shape corresponding to the specific input 2.5D view. In the training phase, our 3D-RecGAN++ firstly leverages a skip-connected encoder-decoder together with an up-sampling module to generate a reasonable 'fake reconstruction' within a high resolution occupancy grid, then applies adversarial learning to refine the 'fake reconstruction' to make it as similar to 'real reconstruction' by jointly updating parameters of the generator. In the testing phase, given a novel 2.5D view as input, the jointly trained generator is able to recover a full 3D shape with satisfactory accuracy, while the discriminator is no longer used.

3.2 Architecture

Figure 4 shows the detailed architecture of our proposed 3D-RecGAN++. It consists of two main networks: the generator as in Figure 4a and the discriminator as in Figure 4b.

The generator consists of a skip-connected encoder-decoder and an up-sampling module. Unlike the vanilla GAN generator which generates data from arbitrary latent distributions, our 3D-RecGAN++ generator synthesizes data from 2.5D views. Particularly, the encoder has five 3D convolutional layers, each of which has a bank of $4 \times 4 \times 4$ filters with strides of $1 \times 1 \times 1$, followed by a leaky ReLU activation function and a max pooling layer with $2 \times 2 \times 2$ filters and strides of $2 \times 2 \times 2$. The number of output channels of max pooling layer starts with 64, doubling at each subsequent layer and ends up with 512. The encoder is lastly followed by two fully-connected layers to embed semantic information into a latent space. The decoder is composed of five symmetric up-convolutional layers which are followed by ReLU activations. Skip-connections between encoder and decoder guarantee propagation of local structures of the input 2.5D view. The skip-connected encoder-decoder is followed by the up-sampling module which simply consists of two layers of up-convolutional layers as detailed in Figure 4a. This simple up-sampling module directly upgrades the output 3D shape to a higher resolution of 256^3 without requiring complex network design and operations. It should be noted that without the two fully connected layers and skip-connections, the vanilla encoder-decoder would be unable to learn reasonable complete 3D structures as the latent space is limited and the local structure is not preserved. The loss function and optimization methods are described in Section 3.4.

The discriminator aims to distinguish whether the estimated 3D shapes are plausible or not. Based on the conditional GAN, the discriminator takes both real reconstruction pairs and fake reconstruction pairs as input. In particular, it consists of six 3D convolutional layers, the first of which concatenates the generated 3D shape (*i.e.*, a 256^3 voxel grid) and the input 2.5D partial view (*i.e.*, a 64^3 voxel grid), reshaped as a $256 \times 256 \times 4$ tensor. The reshaping process is done straightforwardly using Tensorflow 'tf.reshape()'. Basically, this is to inject the condition information with a matched tensor dimension, and then leave the network itself to learn useful features from this condition input. Each convolutional layer has a bank of $4 \times 4 \times 4$ filters with strides of $2 \times 2 \times 2$, followed by a ReLU activation function except for the last layer which is followed by a sigmoid activation function. The number of output channels of the convolutional layers starts with 8, doubling at each subsequent layer and ends up with 256. The output of the last neural layer is reshaped as a latent vector which is the latent feature of discriminator, denoted as $\boldsymbol{m}$.

3.3 Mean Feature for Discriminator

At the early training stage of GAN, as the high dimensional real and fake distributions may not overlap, the discriminator can separate them perfectly using a single scalar output, which is analyzed in [82]. In our experiments, due to the extremely high dimensionality (*i.e.*, $256^3 + 64^3$ dimensions) of the input data pair, the WGAN-GP always crashes in the early 3 epochs if we use a standard fully-connected layer followed by a single scalar as the final output for the discriminator.

To stabilize training, we propose to use the mean feature m (*i.e.*, mean of a vector feature $\boldsymbol{m}$) for discrimination. As the mean vector feature tends to capture more information from the input overall, it is more difficult for the discriminator to easily distinguish between fake or real inputs. This enables useful information to back-propagate to the generator. The final output of the discriminator $D()$ is defined as:

$$m = \mathbf{E}(\boldsymbol{m}) \tag{2}$$

Mean feature matching is also studied and applied in [83] [84] to stabilize GAN. However, Bao *et al.* [83] minimize the L_2 loss of the mean feature, as well as the original Jensen-Shannon divergence-based loss [16], requiring hyper-parameter tuning to balance the two losses. By comparison, in our 3D-RecGAN++ setting, the mean feature of discriminator is directly followed by the existing WGAN-GP loss, which is simple yet effective to stabilize the adversarial training.

Overall, our discriminator learns to distinguish the distributions of mean feature of fake and real reconstructions, while the generator is trained to make the two mean feature distributions as similar as possible.

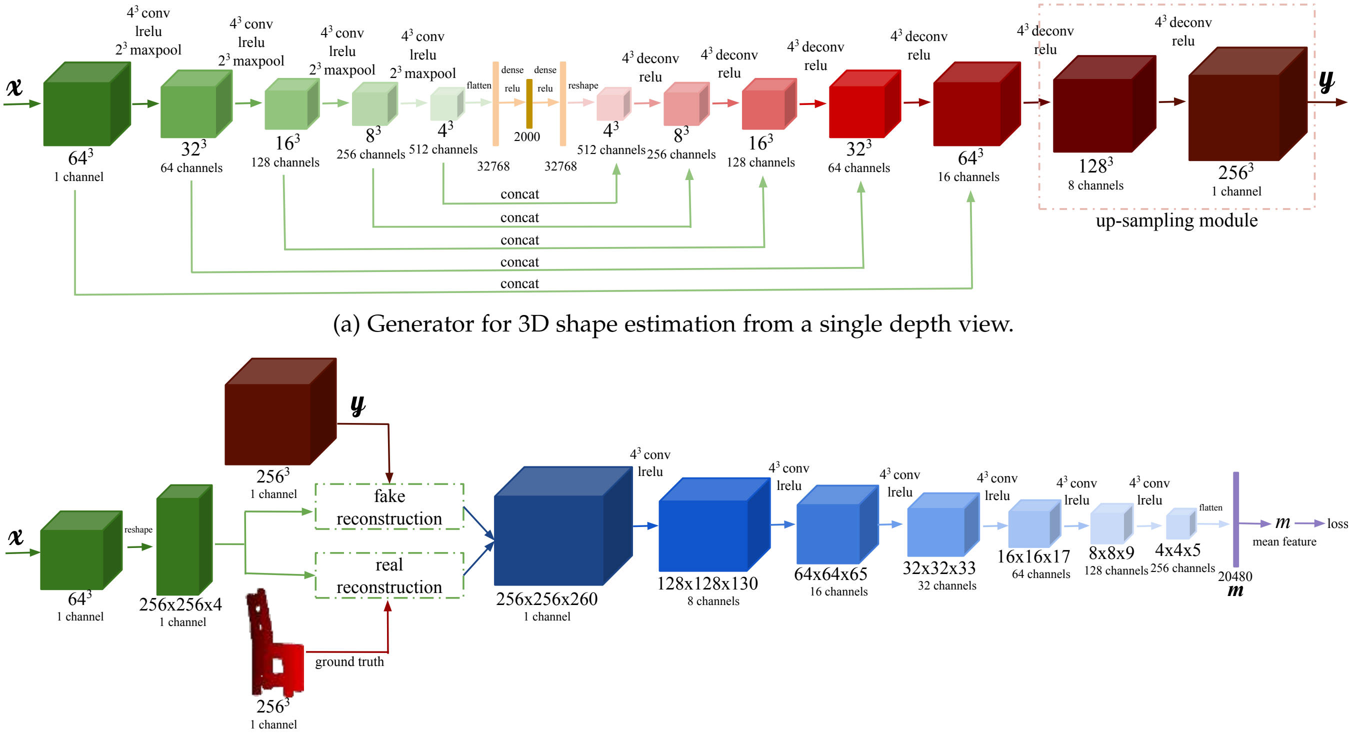
3.4 Objectives

The objective function of 3D-RecGAN++ includes two main parts: an object reconstruction loss ℓ_{en} for the generator; the objective function ℓ_{gan} for the conditional GAN.

(1) ℓ_{en} For the generator, inspired by [85], we use modified binary cross-entropy loss function instead of the standard version. The standard binary cross-entropy weights both false positive and false negative results equally. However, most of the voxel grid tends to be empty, so the network easily gets a false positive estimation. In this regard, we impose a higher penalty on false positive results than on false negatives. Particularly, a weight hyper-parameter α is assigned to false positives, with (1-α) for false negative results, as shown in Equation 3.

$$\ell_{en} = \frac{1}{N}\sum_{i=1}^{N}\Big[-\alpha\bar{y_i}\log(y_i) - (1-\alpha)(1-\bar{y_i})\log(1-y_i)\Big] \tag{3}$$

$where$ $\bar{y_i}$ is the target value {0,1} of a specific i^{th} voxel in the ground truth voxel grid $\bar{\boldsymbol{y}}$, and y_i is the corresponding estimated value (0,1) in the same voxel from the generator output $\boldsymbol{y}$. We calculate the mean loss over the total N voxels in the whole voxel grid.



(a) Generator for 3D shape estimation from a single depth view.

(b) Discriminator for 3D shape refinement.

Fig. 4: Detailed architecture of 3D-RecGAN++, showing the two main building blocks. Note that, although these are shown as two separate modules, they are trained end-to-end.

(2) ℓ_{gan} For the discriminator, we leverage the state of the art WGAN-GP loss functions. Unlike the original GAN loss function which presents an overall loss for both real and fake inputs, we separately represent the loss function ℓ_{gan}^g in Equation 4 for generating fake reconstruction pairs and ℓ_{gan}^d in Equation 5 for discriminating fake and real reconstruction pairs. Detailed definitions and derivation of the loss functions can be found in [77] [81], but we modify them for our conditional GAN settings.

$$\ell_{gan}^g = -\mathbf{E}\left[D(\boldsymbol{y}|\boldsymbol{x})\right] \tag{4}$$

$$\begin{aligned}\ell_{gan}^d = \mathbf{E}\left[D(\boldsymbol{y}|\boldsymbol{x})\right] - \mathbf{E}\left[D(\bar{\boldsymbol{y}}|\boldsymbol{x})\right] \\ +\lambda\mathbf{E}\left[\left(\left\|\nabla_{\hat{\boldsymbol{y}}}D(\hat{\boldsymbol{y}}|\boldsymbol{x})\right\|_2 - 1\right)^2\right]\end{aligned} \tag{5}$$

$where\ \hat{\boldsymbol{y}} = \epsilon\bar{\boldsymbol{y}} + (1-\epsilon)\boldsymbol{y}, \epsilon \sim U[0,1]$, $\boldsymbol{x}$ is the input partial depth view, $\boldsymbol{y}$ is the corresponding output of the generator, $\bar{\boldsymbol{y}}$ is the corresponding ground truth. λ controls the trade-off between optimizing the gradient penalty and the original objective in WGAN.

For the generator in our 3D-RecGAN++ network, there are two loss functions, ℓ_{en} and ℓ_{gan}^g, to optimize. As we discussed in Section 3.1, minimizing ℓ_{en} tends to learn the overall 3D shapes, whilst minimizing ℓ_{gan}^g estimates more plausible 3D structures conditioned on input 2.5D views. To minimize ℓ_{gan}^d is to improve the performance of discriminator to distinguish fake and real reconstruction pairs. To jointly optimize the generator, we assign weights β to ℓ_{en} and $(1-\beta)$ to ℓ_{gan}^g. Overall, the loss functions for generator and discriminator are as follows:

$$\ell_g = \beta\ell_{en} + (1-\beta)\ell_{gan}^g \tag{6}$$

$$\ell_d = \ell_{gan}^d \tag{7}$$

3.5 Training

We adopt an end-to-end training procedure for the whole network. To simultaneously optimize both generator and discriminator, we alternate between one gradient decent step on the discriminator and then one step on the generator. For the WGAN-GP, λ is set as 10 for gradient penalty as in [81]. α ends up as 0.85 for our modified cross entropy loss function, while β is 0.2 for the joint loss function ℓ_g.

The Adam solver [86] is used for both discriminator and generator with a batch size of 4. The other three Adam parameters are set to default values. Learning rate is set to $5e^{-5}$ for the discriminator and $1e^{-4}$ for the generator in all epochs. As we do not use dropout or batch normalization, the testing phase is exactly the same as the training stage. The whole network is trained on a single Titan X GPU from scratch.

3.6 Data Synthesis

For the task of 3D dense reconstruction from a single depth view, obtaining a large amount of training data is an obstacle. Existing real RGB-D datasets for surface reconstruction suffer from occlusions and missing data and there is no ground truth of complete and high resolution 256^3 3D shapes for each view. The recent work [12] synthesizes data for 3D object completion, but the object resolution is only up to 128^3.

To tackle this issue, we use the ShapeNet [78] database to generate a large amount of training and testing data with synthetically rendered depth images and the corresponding complete 3D shape ground truth. Interior parts of individual objects are set to be filled, *i.e.*, '1', while the exterior to be empty, *i.e.*, '0'. A subset of object categories and CAD models are selected for our experiments. As some CAD models in ShapeNet may not be watertight, in our ray tracing based voxelization algorithm, if a specific point is inside of more than 5 faces along X, Y and Z axes, that point is deemed to be interior of the object and set as '1', otherwise '0'.

For each category, to generate **training data**, around 220 CAD models are randomly selected. For each CAD model, we create a virtual depth camera to scan it from 125 different viewing angles, 5 uniformly sampled views for each of roll, pitch and yaw space ranging from $0 \sim 2\pi$ individually. Note that, the viewing angles for all 3D models are the same for simplicity. For each virtual scan, both a depth image and the corresponding complete 3D voxelized structure are generated with regard to the same camera angle. That depth image is simultaneously transformed to a point cloud using virtual camera parameters [87] followed by voxelization which generates a partial 2.5D voxel grid. Then a pair of partial 2.5D view and the complete 3D shape is synthesized. Overall, around 26K training pairs are generated for each 3D object category.

For each category, to synthesize **testing data**, around 40 CAD models are randomly selected. For each CAD model, two groups of testing data are generated. **Group 1**, each model is virtually scanned from 125 viewing angles which are the same as used in training dataset. Around 4.5k testing pairs are generated in total. This group of testing dataset is denoted as **same viewing** (SV) angles testing dataset. **Group 2**, each model is virtually scanned from 216 different viewing angles, 6 uniformly sampled views from each of roll, pitch and yaw space ranging from $0 \sim 2\pi$ individually. Note that, these viewing angles for all testing 3D models are completely different from training pairs. Around 8k testing pairs are generated in total. This group of testing dataset is denoted as **cross viewing** (CV) angles testing dataset. Similarly, we also generate around 1.5k SV and 2.5k CV **validation data** split from another 12 CAD models, which are used for hyperparameter searching.

As our network is initially designed to predict an aligned full 3D model given a depth image from an arbitrary viewing angle, these two SV and CV testing datasets are generated separately to evaluate the viewing angle robustness and generality of our model.

Besides the large quantity of synthesized data, we also collect a **real-world dataset** in order to test the proposed network in a realistic scenario. We use a Microsoft Kinect camera to manually scan a total of 20 object instances belonging to 4 classes {bench, chair, couch, table}, with 5 instances per class from different environments, including offices, homes, and outdoor university parks. For each object we acquire RGB-D images of the object from multiple angles by moving the camera around the object. Then, we use the dense visual SLAM algorithm ElasticFusion [54] in order to reconstruct the full 3D shape of each object, as well as the camera pose in each scan.

We sample 50 random views from the camera trajectory, and for each one we obtain the depth image and the relative camera pose. In each depth image the 3D object is segmented from the background, using a combination of floor removal and manual segmentation. We finally generate ground truth information by aligning the full 3D objects with the partial 2.5D views.

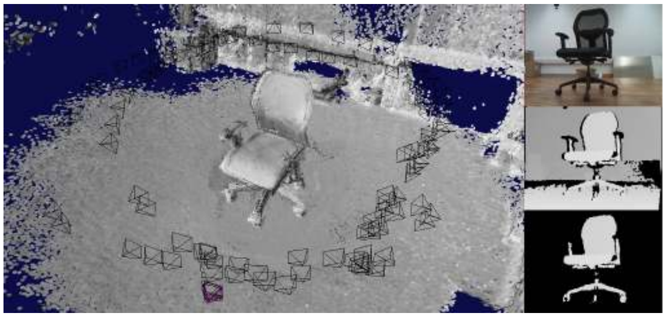


Fig. 5: An example of ElasticFusion for generating real world data. Left: reconstructed object; sampled camera poses are shown in black. Right: Input RGB, depth image and segmented depth image.

It should be noted that, due to noise and quantization artifacts of low-cost RGB-D sensors, and the inaccuracy of the SLAM algorithm, the full 3D ground truth is not 100% accurate, but can still be used as a reasonable approximation. The real-world dataset highlights the challenges related to shape reconstruction from realistic data: noisy depth estimates, missing depth information, depth quantization. In addition, some of the objects are acquired outdoors (*e.g., bench*), which is challenging for the near-infrared depth sensor of the Micorsoft Kinect. However, we argue that a real-world benchmark for shape reconstruction is necessary for a thorough validation of future approaches. Figure 5 shows an example of the reconstructed object and camera poses in ElasticFusion.

4 Evaluation

In this section, we evaluate our 3D-RecGAN++ with comparison to the state of the art approaches and an ablation study to fully investigate the proposed network.

4.1 Metrics

To evaluate the performance of 3D reconstruction, we consider two metrics. The first metric is the mean Intersection-over-Union (IoU) between predicted 3D voxel grids and their ground truth. The IoU for an individual voxel grid is formally defined as follows:

$$IoU = \frac{\sum_{i=1}^{N}\left[I(y_i > p) * I(\bar{y_i})\right]}{\sum_{i=1}^{N}\left[I\left(I(y_i > p) + I(\bar{y_i})\right)\right]}$$

$where$ $I(\cdot)$ is an indicator function, y_i is the predicted value for the i^{th} voxel, $\bar{y_i}$ is the corresponding ground truth, p is the threshold for voxelization, N is the total number of voxels in a whole voxel grid. In all our experiments, p is searched using the validation data split per category for each approach. Particularly, p is searched in the range $[0.1, 0.9]$ with a step size 0.05 using the validation datasets. The higher the IoU value, the better the reconstruction of a 3D model.

The second metric is the mean value of standard Cross-Entropy loss (CE) between a reconstructed shape and the ground truth 3D model. It is formally defined as:

$$CE = -\frac{1}{N}\sum_{i=1}^{N}\left[\bar{y_i}\log(y_i) + (1-\bar{y_i})\log(1-y_i)\right]$$

where y_i, $\bar{y}_i$ and N are the same as defined in above IoU. The lower CE value is, the closer the prediction to be either '1' or '0', the more robust and confident the 3D predictions are.

We also considered the Chamfer Distance (CD) or Earth Mover's Distance (EMD) as an additional metric. However, it is computationally heavy to calculate the distance between two high resolution voxel grids due to the large number of points. In our experiments, it takes nearly 2 minutes to calculate either CD or EMD between two 256^3 shapes on a single Titan X GPU. Although the 256^3 dense shapes can be downsampled to sparse point clouds on object surfaces to quickly compute CD or EMD, the geometric details are inevitably lost due to the extreme downsampling process. Therefore, we did not use CD or EMD for evaluation in our experiments.

4.2 Competing Approaches

We compare against three state of the art deep learning based approaches for single depth view reconstruction. We also compare against the generator alone in our network, *i.e.*, without the GAN, named 3D-RecAE for short.

(1) **3D-EPN.** In [12], Dai *et al.* proposed a neural network, called "3D-EPN", to reconstruct the 3D shape up to a 32^3 voxel grid, after which a high resolution shape is retrieved from an existing 3D shape database, called "Shape Synthesis". In our experiment, we only compared with their neural network (*i.e.*, 3D-EPN) performance because we do not have an existing shape database for similar shape retrieval during testing. Besides, occupancy grid representation is used for the network training and testing.

(2) **Varley *et al.*** In [3], a network was designed to complete the 3D shape from a single 2.5D depth view for robot grasping. The output of their network is a 40^3 voxel grid.

Note that, the low resolution voxel grids generated by 3D-EPN and Varley *et al.* are all upsampled to 256^3 voxel grids using trilinear interpolation before calculating the IoU and CE metrics. The linear upsampling is a widely used post-processing technique for fair comparison in cases where the output resolution is not identical [13]. However, as both 3D-EPN and Varley *et al.* are trained using lower resolution voxel grids for supervision, while the below Han *et al.* and our 3D-RecGAN++ are trained using 256^3 shapes for supervision, it is not strictly fair comparison in this regard. Considering both 3D-EPN and Varley *et al.* are among the early works and also solid competing approaches regarding the single depth view reconstruction task, we therefore include them as baselines.

(3) **Han *et al.*** In [27], a global structure inference network and a local geometry refinement network are proposed to complete a high resolution shape from a noisy shape. The network is not originally designed for single depth view reconstruction, but its output shape is up to a 256^3 voxel grid and is comparable to our network. For fair comparison, the same occupancy grid representation is used for their network. It should be noted that [27] involves convoluted designs, thus the training procedure is slower and less efficient due to many LSTMs integrated.

(4) **3D-RecAE.** As for our 3D-RecGAN++, we remove the discriminator and only keep the generator to infer the complete 3D shape from a single depth view. This comparison illustrates the benefits of adversarial learning.

4.3 Single-category Results

(1) **Results.** All networks are separately trained and tested on four different categories, {bench, chair, couch, table}, with the same network configuration. Table 1 shows the IoU and CE loss of all methods on the testing dataset with same viewing angles on 256^3 voxel grids, while Table 2 shows the IoU and CE loss comparison on testing dataset with cross viewing angles. Figure 6 shows the qualitative results of single category reconstruction on testing datasets with same and cross viewing angles. The meshgrid function in Matlab is used to plot all 3D shapes for better visualization.

(2) **Analysis.** Both 3D-RecGAN++ and 3D-RecAE significantly outperform the competing approaches in terms of IoU and CE loss on both the SV and CV testing datasets for dense 3D shape reconstruction (256^3 voxel grids). Although our approach is trained on depth input with a limited set of viewing angles, it still performs well to predict aligned 3D shapes from novel viewing angles. The 3D shapes generated by 3D-RecGAN++ and 3D-RecAE are much more visually compelling than others.

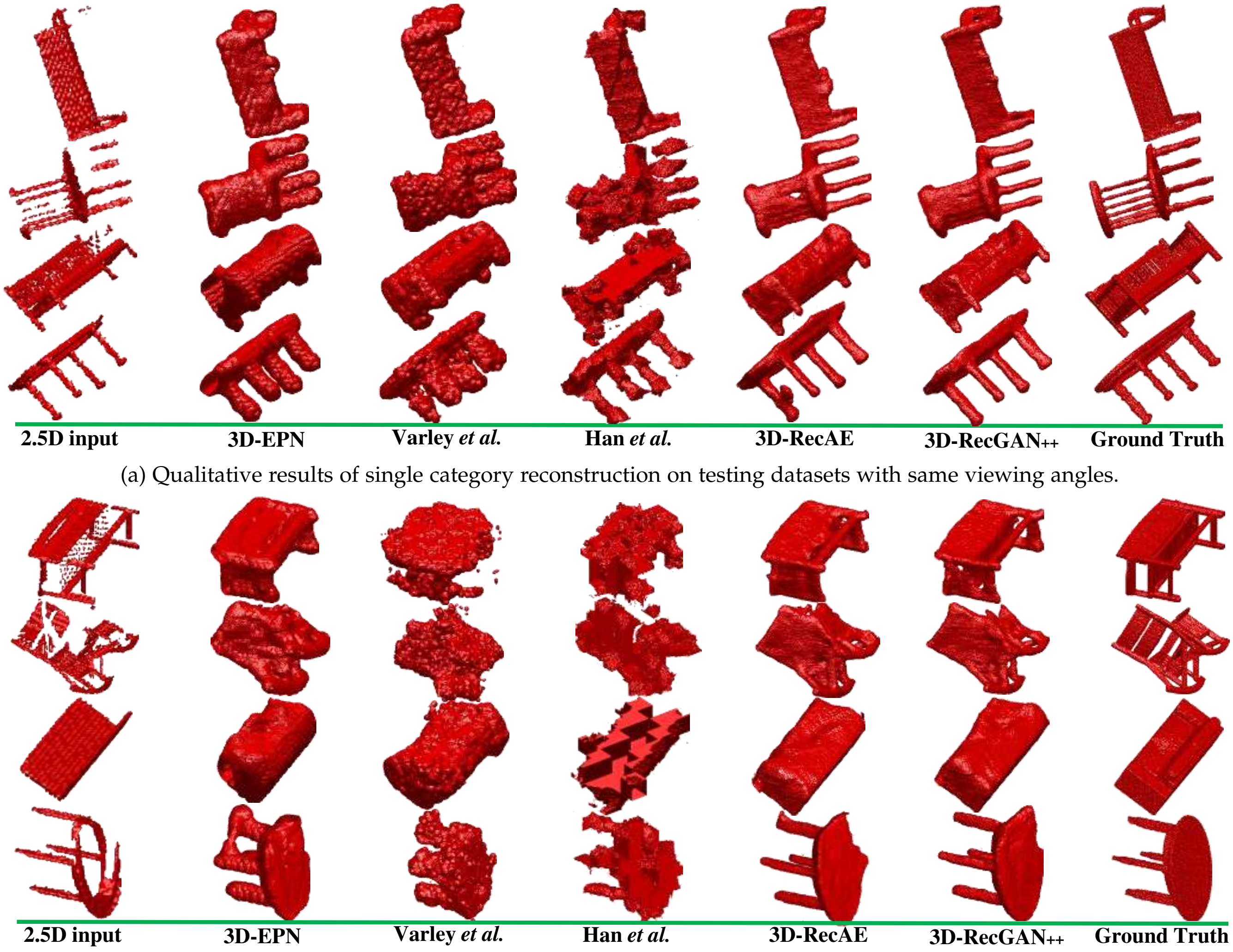
Compared with 3D-RecAE, 3D-RecGAN++ achieves better IoU scores and smaller CE loss. Basically, adversarial learning of the discriminator serves as a regularizer for fine-grained 3D shape estimation, which enables the output of 3D-RecGAN++ to be more robust and confident. We also notice that the increase of 3D-RecGAN++ in IoU and CE scores is not dramatic compared with 3D-RecAE. This is primarily because the main object shape can be reasonably predicted by 3D-RecAE, while the finer geometric details estimated by 3D-RecGAN++ are usually smaller parts of the whole object shape. Therefore, 3D-RecGAN++ only obtains a reasonable better IoU and CE scores than 3D-RecAE. The 4^{th} row of Figure 6a shows a good example in terms of finer geometric details prediction of 3D-RecGAN++. In fact, in all the remaining experiments, 3D-RecGAN++ is constantly, but not significantly, better than 3D-RecAE.

TABLE 1: Per-category IoU and CE loss on testing dataset with same viewing angles (256^3 voxel grids).

	IoU				CE Loss			
	bench	chair	couch	table	bench	chair	couch	table
3D-EPN [12]	0.423	0.488	0.631	0.508	0.087	0.105	0.144	0.101
Varley *et al.* [3]	0.227	0.317	0.544	0.233	0.111	0.157	0.195	0.191
Han *et al.* [27]	0.441	0.426	0.446	0.499	0.045	0.081	0.165	0.058
3D-RecAE (ours)	0.577	0.641	0.749	0.675	0.036	0.063	0.067	0.043
3D-RecGAN++ (ours)	**0.580**	**0.647**	**0.753**	**0.679**	**0.034**	**0.060**	**0.066**	**0.040**

TABLE 2: Per-category IoU and CE loss on testing dataset with cross viewing angles (256^3 voxel grids).

	IoU				CE Loss			
	bench	chair	couch	table	bench	chair	couch	table
3D-EPN [12]	0.408	0.446	0.572	0.482	0.086	0.112	0.163	0.103
Varley *et al.* [3]	0.185	0.278	0.475	0.187	0.108	0.171	0.210	0.186
Han *et al.* [27]	0.439	0.426	0.455	0.482	0.047	0.090	0.163	0.060
3D-RecAE (ours)	0.524	0.588	0.639	0.610	0.045	0.079	0.117	0.058
3D-RecGAN++ (ours)	**0.531**	**0.594**	**0.646**	**0.618**	**0.041**	**0.074**	**0.111**	**0.053**



(a) Qualitative results of single category reconstruction on testing datasets with same viewing angles.

(b) Qualitative results of single category reconstruction on testing datasets with cross viewing angles.

Fig. 6: Qualitative results of single category reconstruction on testing datasets with same and cross viewing angles.

4.4 Multi-category Results

(1) **Results.** All networks are also trained and tested on multiple categories without being given any class labels. The networks are trained on four categories: {bench, chair, couch, table}; and then tested separately on individual categories. Table 3 shows the IoU and CE loss comparison of all methods on testing dataset with same viewing angles for dense shape reconstruction, while Table 4 shows the IoU and CE loss comparison on testing dataset with cross viewing angles. Figure 7 shows the qualitative results of all approaches on testing datasets of multiple categories with same and cross viewing angles.

(2) **Analysis.** Both 3D-RecGAN++ and 3D-RecAE significantly outperforms the state of the art by a large margin in all categories which are trained together on a single model. Besides, the performance of our network trained on multiple categories, does not notably degrade compared with training the network on individual categories as shown in previous Table 1 and 2. This confirms that our network has enough capacity and capability to learn diverse features from multiple categories.

TABLE 3: Multi-category IoU and CE loss on testing dataset with same viewing angles (256^3 voxel grids).

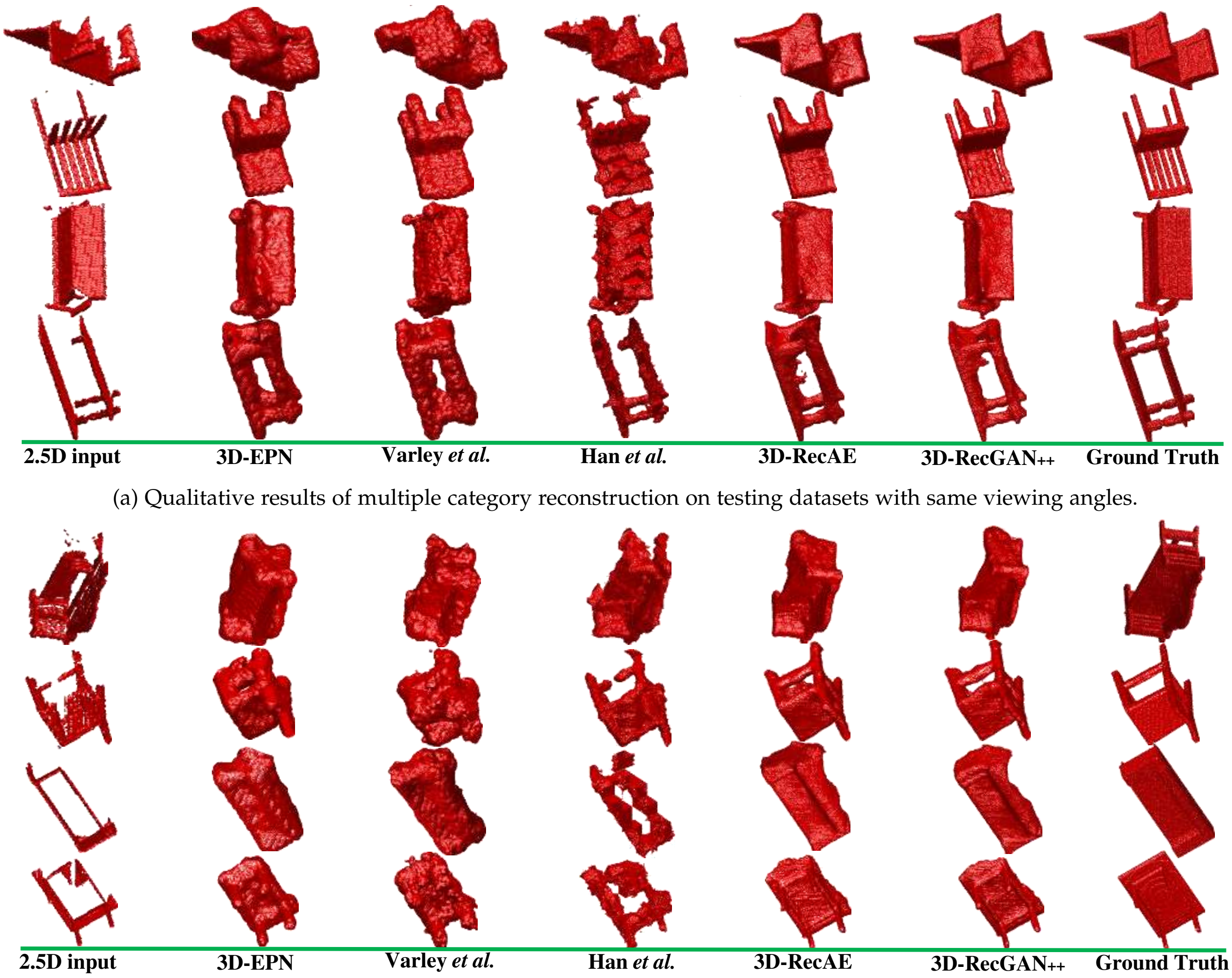
	IoU				CE Loss			
	bench	chair	couch	table	bench	chair	couch	table
3D-EPN [12]	0.428	0.484	0.634	0.506	0.087	0.107	0.138	0.102
Varley *et al.* [3]	0.234	0.317	0.543	0.236	0.103	0.132	0.197	0.170
Han *et al.* [27]	0.425	0.454	0.440	0.470	0.045	0.087	0.172	0.065
3D-RecAE (ours)	0.576	0.632	0.740	0.661	0.037	0.060	0.069	0.044
3D-RecGAN++ (ours)	**0.581**	**0.640**	**0.745**	**0.667**	**0.030**	**0.051**	**0.063**	**0.039**

TABLE 4: Multi-category IoU and CE loss on testing dataset with cross viewing angles (256^3 voxel grids).

	IoU				CE Loss			
	bench	chair	couch	table	bench	chair	couch	table
3D-EPN [12]	0.415	0.452	0.531	0.477	0.091	0.115	0.147	0.111
Varley *et al.* [3]	0.201	0.283	0.480	0.199	0.105	0.143	0.207	0.174
Han *et al.* [27]	0.429	0.444	0.447	0.474	0.045	0.089	0.172	0.063
3D-RecAE (ours)	0.530	0.587	0.640	0.610	0.043	0.068	0.096	0.055
3D-RecGAN++ (ours)	**0.540**	**0.594**	**0.643**	**0.621**	**0.038**	**0.061**	**0.091**	**0.048**

4.5 Cross-category Results

(1) **Results.** To further investigate the generality of networks, we train all networks on {bench, chair, couch, table}, and then test them on another 6 totally different categories: {car, faucet, firearm, guitar, monitor, plane}. For each of the 6 categories, we generate the same amount of testing datasets with same and cross viewing angles, which is similar to the previous {bench, chair, couch, table}. Table 5 and 6 shows the IoU and CE loss comparison of all approaches on the testing dataset with same viewing angles,



(a) Qualitative results of multiple category reconstruction on testing datasets with same viewing angles.

(b) Qualitative results of multiple category reconstruction on testing datasets with cross viewing angles.

Fig. 7: Qualitative results of multiple category reconstruction on testing datasets with same and cross viewing angles.

TABLE 5: Cross-category IoU on testing dataset with same viewing angles (256^3 voxel grids).

	car	faucet	firearm	guitar	monitor	plane
3D-EPN [12]	0.450	0.442	0.339	0.351	0.444	0.314
Varley *et al.* [3]	0.484	0.260	0.280	0.255	0.341	0.295
Han *et al.* [27]	0.360	0.402	0.333	0.353	0.450	0.306
3D-RecAE (ours)	**0.557**	0.530	0.422	0.440	0.556	0.390
3D-RecGAN++ (ours)	0.555	**0.536**	**0.426**	**0.442**	**0.562**	**0.394**

TABLE 6: Cross-category CE loss on testing dataset with same viewing angles (256^3 voxel grids).

	car	faucet	firearm	guitar	monitor	plane
3D-EPN [12]	0.170	0.088	0.036	0.036	0.123	0.066
Varley *et al.* [3]	0.173	0.122	0.029	0.030	0.130	0.042
Han *et al.* [27]	0.167	0.077	0.018	0.015	0.088	0.031
3D-RecAE (ours)	0.110	0.057	0.018	0.016	0.072	0.036
3D-RecGAN++ (ours)	**0.102**	**0.053**	**0.016**	**0.014**	**0.067**	**0.031**

TABLE 7: Cross-category IoU on testing dataset with cross viewing angles (256^3 voxel grids).

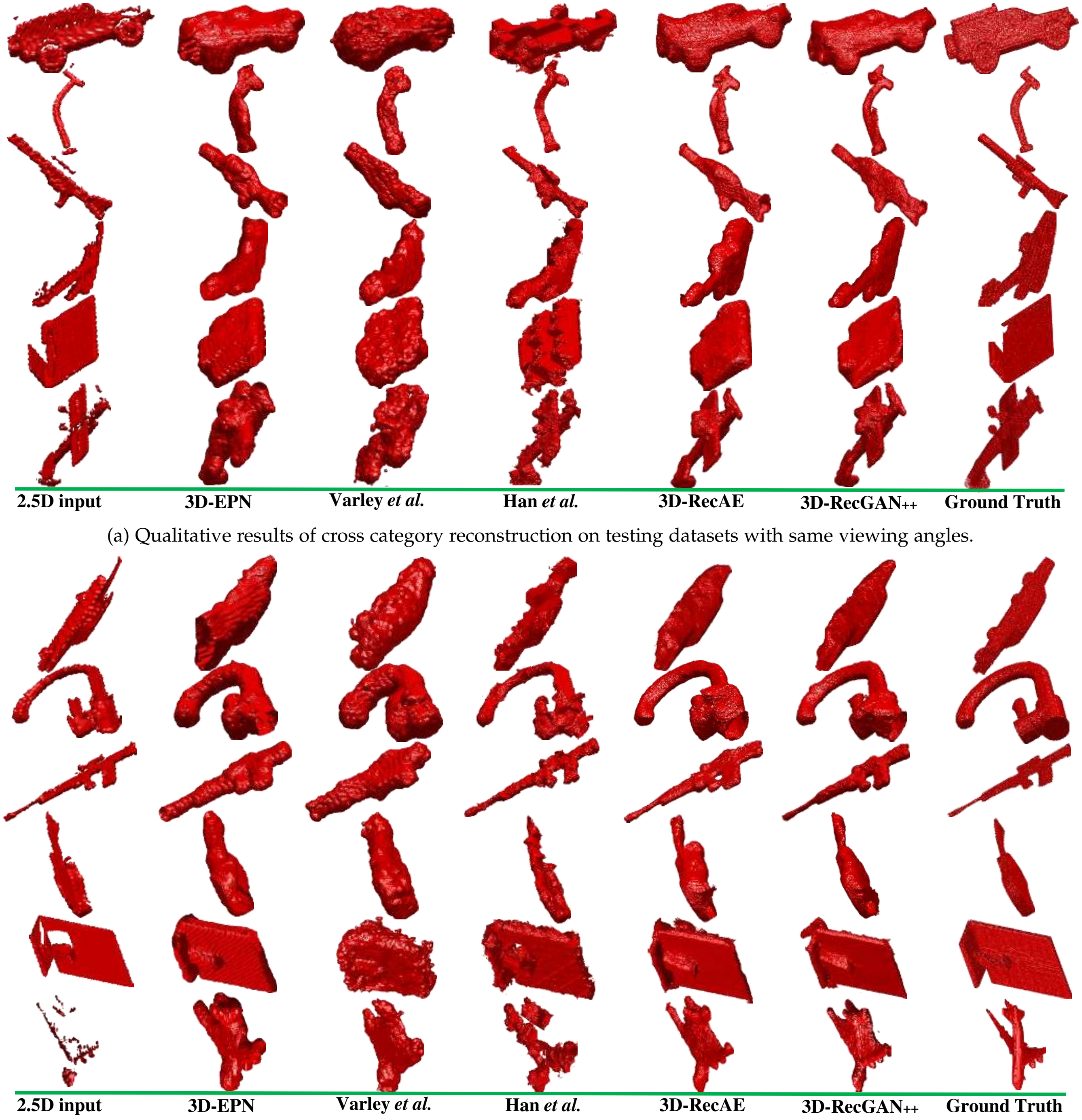
	car	faucet	firearm	guitar	monitor	plane
3D-EPN [12]	0.446	0.439	0.324	0.359	0.448	0.309
Varley *et al.* [3]	0.489	0.260	0.274	0.255	0.334	0.283
Han *et al.* [27]	0.349	0.402	0.321	0.363	0.455	0.299
3D-RecAE (ours)	0.550	0.521	0.411	0.441	0.550	0.382
3D-RecGAN++ (ours)	**0.553**	**0.529**	**0.416**	**0.449**	**0.555**	**0.390**

TABLE 8: Cross-category CE loss on testing dataset with cross viewing angles (256^3 voxel grids).

	car	faucet	firearm	guitar	monitor	plane
3D-EPN [12]	0.160	0.087	0.033	0.036	0.127	0.065
Varley *et al.* [3]	0.171	0.123	0.028	0.030	0.136	0.043
Han *et al.* [27]	0.171	0.076	0.018	0.016	0.088	0.031
3D-RecAE (ours)	0.101	0.059	0.017	0.017	0.079	0.036
3D-RecGAN++ (ours)	**0.100**	**0.055**	**0.014**	**0.015**	**0.074**	**0.031**

while Table 7 and 8 shows the IoU and CE loss comparison on the testing dataset with cross viewing angles. Figure 8 shows the qualitative results of all methods on 6 unseen categories with same and cross viewing angles.

We further evaluate the generality of our 3D-RecGAN++ on a specific category. Particularly, we conduct four groups of experiments. In the first group, we train our 3D-RecGAN++ on bench, then separately test it on the remaining 3 categories: {chair, couch, table}. In the second group, the network is trained on chair and separately tested on {bench, couch, table}. Similarly, another two groups of experiments are conducted. Basically, this experiment is to investigate how well our approach learns features from one category and then generalizes to a different category, and vice versa. Table 9 shows the cross-category IoU and CE loss of our 3D-RecGAN++ trained on individual category



(a) Qualitative results of cross category reconstruction on testing datasets with same viewing angles.

(b) Qualitative results of cross category reconstruction on testing datasets with cross viewing angles.

Fig. 8: Qualitative results of cross category reconstruction on testing datasets with same and cross viewing angles.

and then tested on the testing dataset with same viewing angles over 256^3 voxel grids.

(2) **Analysis.** The proposed 3D-RecGAN++ achieves much higher IoU and smaller CE loss across the unseen categories than competing approaches. Our network not only learns rich features from different object categories, but also is able to generalize well to completely new types of categories. Our intuition is that the network may learn geometric features such as lines, planes, curves which are common across various object categories. As our network involves skip-connections between intermediate neural layers, it is not straightforward to visualize and analyze the learnt latent features.

It can be also observed that our model trained on *bench* tends to be more general than others. Intuitively, the bench category tends to have general features such as four legs, seats, and/or a back, which are also common among other categories $\{$chair, couch, table$\}$. However, not all chairs or couches consist of such general features that are shared across different categories.

Overall, we may safely conclude that the more similar features two categories share, including both the low-level lines/planes/curves and the high-level shape components, the better generalization of our model achieves cross those

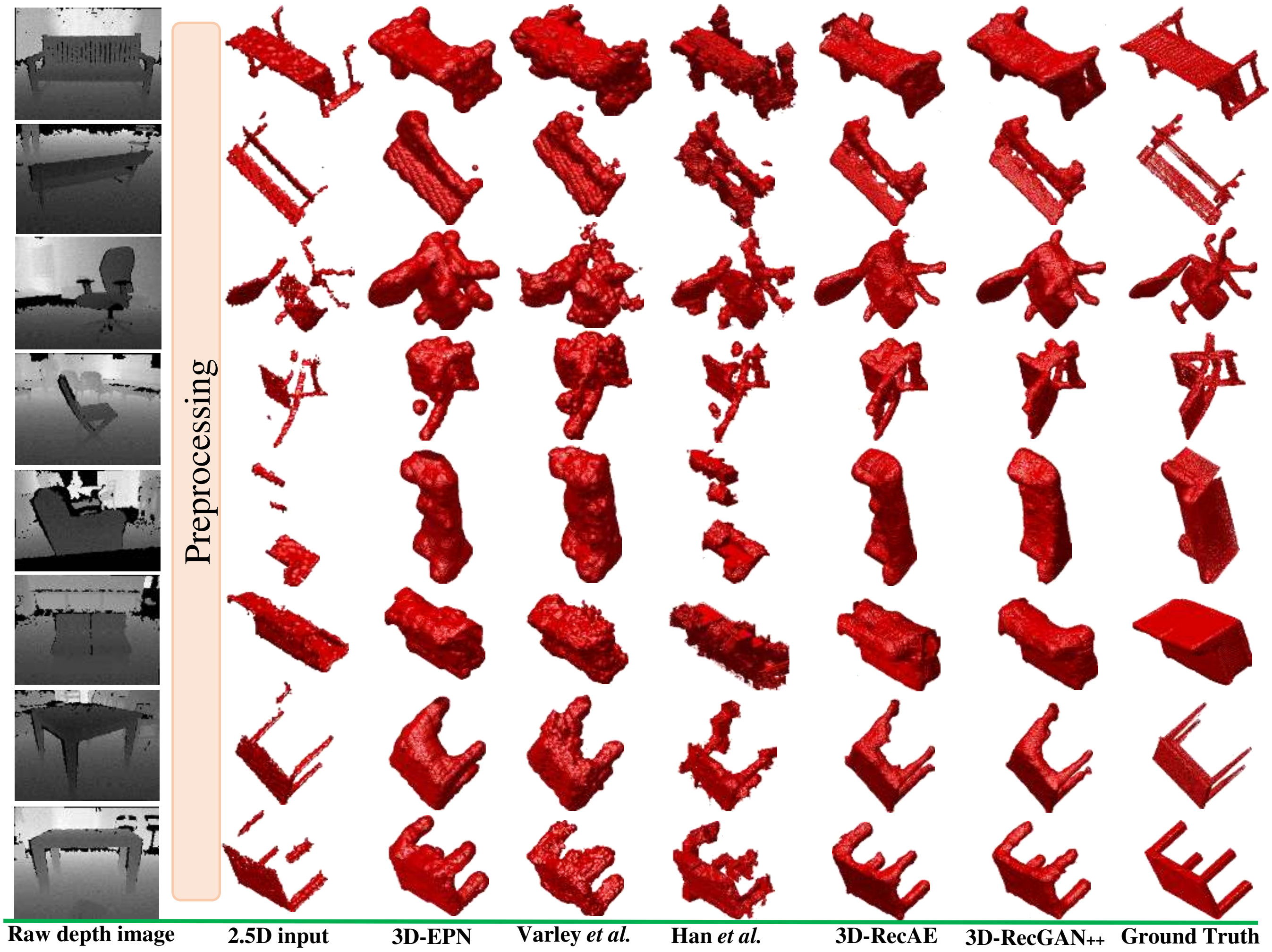


Fig. 9: Qualitative results of real-world objects reconstruction from different approaches. The object instance is segmented from the raw depth image in preprocessing step.

categories.

TABLE 9: Cross-category IoU and CE loss of 3D-RecGAN++ trained on individual category and then tested on the testing dataset with same viewing angles (256^3 voxel grids).

	IoU				CE Loss			
	bench	chair	couch	table	bench	chair	couch	table
Group 1 (trained on bench)	**0.580**	0.510	0.507	0.599	**0.034**	0.110	0.164	0.062
Group 2 (trained on chair)	0.508	**0.647**	0.469	0.564	0.048	**0.060**	0.184	0.069
Group 3 (trained on couch)	0.429	0.504	**0.753**	0.437	0.070	0.105	**0.066**	0.126
Group 4 (trained on table)	0.510	0.509	0.402	**0.679**	0.049	0.111	0.260	**0.040**

4.6 Real-world Experiment Results

(1) **Results.** Lastly, in order to evaluate the domain adaptation capability of the networks, we train all networks on synthesized data of categories {bench, chair, couch, table}, and then test them on real-world data collected by a Microsoft Kinect camera. Table 10 compares the IoU and CE loss of all approaches on the real-world dataset. Figure 9 shows some qualitative results for all methods.

(2) **Analysis.** There are two reasons why the IoU is significantly lower compared with testing on the synthetic dataset. First, the ground truth objects obtained from ElasticFusion are not as solid as the synthesized datasets. However, all networks predict dense and solid voxel grids, so the interior parts may not match though the overall object shapes are satisfactorily recovered as shown in Figure 9. Secondly, the input 2.5D depth view from real-world dataset is noisy and incomplete, due to the limitation of the RGB-D sensor (*e.g.*, reflective surfaces, outdoor lighting). In some cases, the input 2.5D view does not capture the whole object and only contains a part of the object, which also leads to inferior reconstruction results (*e.g.*, the 5^{th} row in Figure 9) and a lower IoU scores overall. However, our proposed network is still able to reconstruct reasonable 3D dense shapes given the noisy and incomplete 2.5D input depth views, while the competing algorithms (*e.g.*, Varley *et al.*) are not robust to real-world noise and unable to generate compelling results.

TABLE 10: Multi-category IoU and CE loss on real-world dataset (256^3 voxel grids).

	IoU				CE Loss			
	bench	chair	couch	table	bench	chair	couch	table
3D-EPN [12]	0.162	0.190	0.508	0.140	0.090	0.158	0.413	0.187
Varley *et al.* [3]	0.118	0.152	0.433	0.075	0.073	0.155	0.436	0.191
Han *et al.* [27]	0.166	0.164	0.235	0.146	0.083	0.167	0.352	0.194
3D-RecAE (ours)	0.173	0.203	0.538	0.151	0.065	0.156	0.318	0.180
3D-RecGAN++ (ours)	**0.177**	**0.208**	**0.540**	**0.156**	**0.061**	**0.153**	**0.314**	**0.177**

4.7 Impact of Adversarial Learning

(1) **Results.** In all above experiments, the proposed 3D-RecGAN++ tends to outperform the ablated network 3D-RecAE which does not include the adversarial learning of GAN part. In all visualization of experiment results, the 3D shapes from 3D-RecGAN++ are also more compelling than 3D-RecAE. To further quantitatively investigate how the adversarial learning improves the final 3D results comparing with 3D-RecAE, we calculate the mean precision and recall from the previous multi-category experiment results in Section 4.4. Table 11 compares the mean precision and recall of 3D-RecGAN++ and 3D-RecAE on individual categories using the network trained on multiple categories.

(2) **Analysis.** It can be seen that the results of 3D-RecGAN++ tend to constantly have higher precision scores than 3D-RecAE, which means 3D-RecGAN++ has less false positive estimations. Therefore, the estimated 3D shapes from 3D-RecAE are likely to be 'fatter' and 'bigger', while 3D-RecGAN++ tends to predict 'thinner' shapes with much more shape details being exposed. Both 3D-RecGAN++ and 3D-RecAE can achieve high recall scores (*i.e.*, above 0.8), which means both 3D-RecGAN++ and 3D-RecAE are capable of estimating the major object shapes without too many false negatives. In other words, the ground truth 3D shape tends to be a subset of the estimated shape result.

TABLE 11: Multi-category mean precision and recall on testing dataset with same viewing angles (256^3 voxel grids).

	mean precision				mean recall			
	bench	chair	couch	table	bench	chair	couch	table
3D-RecAE	0.668	0.740	0.800	0.750	**0.808**	0.818	0.907	0.845
3D-RecGAN++	**0.680**	**0.747**	**0.804**	**0.754**	0.804	**0.820**	**0.910**	**0.853**

Overall, with regard to experiments on per-category, multi-category, and cross-category experiments, our 3D-RecGAN++ outperforms others by a large margin, although all other approaches can reconstruct reasonable shapes. In terms of the generality, Varley *et al.* [3] and Han *et al.* [27] are inferior because Varley *et al.* [3] use a single fully connected layers, instead of 3D ConvNets, for shape generation which is unlikely to be general for various shapes, and Han *et al.* [27] apply LSTMs for shape blocks generation which is inefficient and unable to learn general 3D structures. However, our 3D-RecGAN++ is superior thanks to the generality of the 3D encoder-decoder and the adversarial discriminator. Besides, the 3D-RecAE tends to over estimate the 3D shape, while the adversarial learning of 3D-RecGAN++ is likely to remove the over-estimated parts, so as to leave the estimated shape to be clearer with more shape details.

4.8 Computation Analysis

Table 12 compares the computation efficiency of all approaches regarding the total number of model parameters and the average time consumption to recover a single object.

The model proposed by Han *et al.* [27] has the least number of parameters because most of the parameters are shared to predict different blocks of an object. Our 3D-RecGAN++ has reasonable 167.1 millions parameters, which is on a similar scale to VGG-19 (*i.e.*, 144 millions) [88].

To evaluate the average time consumption for a single object reconstruction, we implement all networks in Tensorflow 1.2 and Python 2.7 with CUDA 8.0 and cuDNN 7.1 as the back-end driver and library. All models are tested on a single Titan X GPU in the same hardware and software environments. 3D-EPN [12] takes the shortest time to predict a 32^3 object on GPU, while our 3D-RecGAN++ only needs around 40 milliseconds to recover a dense 256^3 object. Comparatively, Han *et al.* takes the longest GPU time to generate a dense object because of the time-consuming sequential processing of LSTMs. The low resolution objects predicted by 3D-EPN and Varley *et al.* are further upsampled to 256^3 using existing SciPy library on a CPU server (Intel E5-2620 v4, 32 cores). It takes around 7 seconds to finish the upsampling for a single object.

TABLE 12: Comparison of model parameters and average time consumption to reconstruction a single object.

	parameters (millions)	**GPU time** (milliseconds)	**predicted 3D shapes** (resolution)
3D-EPN [12]	52.4	**15.8**	32^3
Varley *et al.* [3]	430.3	16.1	40^3
Han *et al.* [27]	**7.5**	276.4	$\mathbf{256^3}$
3D-RecGAN++	167.1	38.9	$\mathbf{256^3}$

5 Discussion

Although our 3D-RecGAN++ achieves the state of the art performance in 3D object reconstruction from a single depth view, it has limitations. Firstly, our network takes the volumetric representation of a single depth view as input, instead of taking a raw depth image. Therefore, a preprocessing of raw depth images is required for our network. However, in many application scenarios such as robot grasping, such preprocessing would be trivial and straightforward given the depth camera parameters. Secondly, the input depth view of our network only contains a clean object information without cluttered background. One possible solution is to leverage an existing segmentation algorithm such as Mask-RCNN [89] to clearly segment the target object instance from the raw depth view.

6 Conclusion

In this work, we proposed a framework 3D-RecGAN++ that reconstructs the full 3D structure of an object from an arbitrary depth view. By leveraging the generalization capabilities of 3D encoder-decoder and generative adversarial networks, our 3D-RecGAN++ predicts dense and accurate 3D structures with fine details, outperforming the state of the art in single-view shape completion for individual object category. We further tested our network's ability to reconstruct multiple categories without providing any object class labels during training or testing, and it showed that our network is still able to predict precise 3D shapes. Besides, we investigated the network's reconstruction performance on unseen categories, our proposed approach can also predict satisfactory 3D structures. Finally, our model is robust to real-world noisy data and can infer accurate 3D shapes although the model is purely trained on synthesized data. This confirms that our network has the capability of learning general 3D latent features of the objects, rather than simply fitting a function for the training datasets, and the adversarial learning of 3D-RecGAN++ learns to add geometric details for estimated 3D shapes. In summary, our network only requires a single depth view to recover a dense and complete 3D shape with fine details.

References

[1] A. Sharma, O. Grau, and M. Fritz, "VConv-DAE : Deep Volumetric Shape Learning Without Object Labels," *ECCV*, 2016.

[2] Z. Wang, S. Rosa, L. Xie, B. Yang, S. Wang, N. Trigoni, and A. Markham, "Defo-Net: Learning Body Deformation Using Generative Adversarial Networks," *ICRA*, 2018.

[3] J. Varley, C. Dechant, A. Richardson, J. Ruales, and P. Allen, "Shape Completion Enabled Robotic Grasping," *IROS*, 2017.

[4] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion: Real-time dense surface mapping and tracking," *ISMAR*, 2011.

[5] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger, "Real-time 3D reconstruction at scale using voxel hashing," *ACM Transactions on Graphics*, vol. 32, no. 6, pp. 1–11, 2013.

[6] F. Steinbrucker, C. Kerl, J. Sturm, and D. Cremers, "Large-Scale Multi-Resolution Surface Reconstruction from RGB-D Sequences," *ICCV*, 2013.

[7] A. Nealen, T. Igarashi, O. Sorkine, and M. Alexa, "Laplacian Mesh Optimization," *SIGGRAPH*, 2006.

[8] W. Zhao, S. Gao, and H. Lin, "A robust hole-filling algorithm for triangular mesh," *The Visual Computer*, vol. 23, no. 12, pp. 987–997, 2007.

[9] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson Surface Reconstruction," *Symposium on Geometry Processing*, 2006.

[10] M. Kazhdan and H. Hoppe, "Screened poisson surface reconstruction," *ACM Transactions on Graphics*, vol. 32, no. 3, pp. 1–13, 2013.

[11] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3D ShapeNets: A Deep Representation for Volumetric Shapes," *CVPR*, 2015.

[12] A. Dai, C. R. Qi, and M. Nießner, "Shape Completion using 3D-Encoder-Predictor CNNs and Shape Synthesis," *CVPR*, 2017.

[13] M. Tatarchenko, A. Dosovitskiy, and T. Brox, "Octree Generating Networks: Efficient Convolutional Architectures for High-resolution 3D Outputs," *ICCV*, 2017.

[14] G. Riegler, A. O. Ulusoy, H. Bischof, and A. Geiger, "OctNetFusion: Learning Depth Fusion from Data," *3DV*, 2017.

[15] H. Christian, S. Tulsiani, and J. Malik, "Hierarchical Surface Prediction for 3D Object Reconstruction," *3DV*, 2017.

[16] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Nets," *NIPS*, 2014.

[17] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," *ICLR*, 2014.

[18] Z. Hu, Z. Yang, X. Liang, R. Salakhutdinov, and E. P. Xing, "Controllable Text Generation," *ICML*, 2017.

[19] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive Growing of GANs for Improved Quality, Stability, and Variation," *ICLR*, 2018.

[20] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets," *NIPS*, 2016.

[21] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. B. Tenenbaum, "Deep Convolutional Inverse Graphics Network," *NIPS*, 2015.

[22] E. Grant, P. Kohli, and M. V. Gerven, "Deep Disentangled Representations for Volumetric Reconstruction," *ECCV Workshops*, 2016.

[23] R. Girdhar, D. F. Fouhey, M. Rodriguez, and A. Gupta, "Learning a Predictable and Generative Vector Representation for Objects," *ECCV*, 2016.

[24] H. Huang, E. Kalogerakis, and B. Marlin, "Analysis and synthesis of 3D shape families via deep-learned generative models of surfaces," *Computer Graphics Forum*, vol. 34, no. 5, pp. 25–38, 2015.

[25] J. Wu, C. Zhang, T. Xue, W. T. Freeman, and J. B. Tenenbaum, "Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling," *NIPS*, 2016.

[26] M. Mirza and S. Osindero, "Conditional Generative Adversarial Nets," *arXiv*, 2014.

[27] X. Han, Z. Li, H. Huang, E. Kalogerakis, and Y. Yu, "High-Resolution Shape Completion Using Deep Neural Networks for Global Structure and Local Geometry Inference," *ICCV*, 2017.

[28] B. Yang, H. Wen, S. Wang, R. Clark, A. Markham, and N. Trigoni, "3D Object Reconstruction from a Single Depth View with Adversarial Learning," *ICCV Workshops*, 2017.

[29] A. Monszpart, N. Mellado, G. J. Brostow, and N. J. Mitra, "RAPter: Rebuilding Man-made Scenes with Regular Arrangements of Planes," *ACM Transactions on Graphics*, vol. 34, no. 4, pp. 1–12, 2015.

[30] N. J. Mitra, L. J. Guibas, and M. Pauly, "Partial and Approximate Symmetry Detection for 3D Geometry," *SIGGRAPH*, 2006.

[31] M. Pauly, N. J. Mitra, J. Wallner, H. Pottmann, and L. J. Guibas, "Discovering structural regularity in 3D geometry," *ACM Transactions on Graphics*, vol. 27, no. 3, p. 1, 2008.

[32] I. Sipiran, R. Gregor, and T. Schreck, "Approximate Symmetry Detection in Partial 3D Meshes," *Computer Graphics Forum*, vol. 33, no. 7, pp. 131–140, 2014.

[33] P. Speciale, M. R. Oswald, A. Cohen, and M. Pollefeys, "A Symmetry Prior for Convex Variational 3D Reconstruction," *ECCV*, 2016.

[34] S. Thrun and B. Wegbreit, "Shape from symmetry," *ICCV*, 2005.

[35] Y. M. Kim, N. J. Mitra, D.-M. Yan, and L. Guibas, "Acquiring 3D Indoor Environments with Variability and Repetition," *ACM Transactions on Graphics*, vol. 31, no. 6, 2012.

[36] Y. Li, A. Dai, L. Guibas, and M. Nießner, "Database-Assisted Object Retrieval for Real-Time 3D Reconstruction," *Computer Graphics Forum*, vol. 34, no. 2, pp. 435–446, 2015.

[37] L. Nan, K. Xie, and A. Sharf, "A Search-Classify Approach for Cluttered Indoor Scene Understanding," *ACM Transactions on Graphics*, vol. 31, no. 6, pp. 1–10, 2012.

[38] T. Shao, W. Xu, K. Zhou, J. Wang, D. Li, and B. Guo, "An interactive approach to semantic modeling of indoor scenes with an RGBD camera," *ACM Transactions on Graphics*, vol. 31, no. 6, pp. 1–11, 2012.

[39] Y. Shi, P. Long, K. Xu, H. Huang, and Y. Xiong, "Data-driven contextual modeling for 3d scene understanding," *Computers & Graphics*, vol. 55, pp. 55–67, 2016.

[40] J. Rock, T. Gupta, J. Thorsen, J. Gwak, D. Shin, and D. Hoiem, "Completing 3D Object Shape from One Depth Image," *CVPR*, 2015.

[41] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[42] R. A. Newcombe, S. J. Lovegrove, and A. J. Davision, "DTAM: Dense Tracking and Mapping in Real-time," *ICCV*, 2011.

[43] S. Baker and I. Matthews, "Lucas-Kanade 20 Years On : A Unifying Framework : Part 1," *International Journal of Computer Vision*, vol. 56, no. 3, pp. 221–255, 2004.

[44] S. Y. Bao, M. Chandraker, Y. Lin, and S. Savarese, "Dense Object Reconstruction with Semantic Priors," *CVPR*, 2013.

[45] A. Dame, V. A. Prisacariu, C. Y. Ren, and I. Reid, "Dense Reconstruction Using 3D Object Shape Priors," *CVPR*, 2013.

[46] F. Engelmann, J. Stuckler, and B. Leibe, "Joint Object Pose Estimation and Shape Reconstruction in Urban Street Scenes Using 3D Shape Priors," *GCPR*, 2016.

[47] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, "3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction," *ECCV*, 2016.

[48] X. Di, R. Dahyot, and M. Prasad, "Deep Shape from a Low Number of Silhouettes," *ECCV*, 2016.

[49] Z. Lun, M. Gadelha, E. Kalogerakis, S. Maji, and R. Wang, "3D Shape Reconstruction from Sketches via Multi-view Convolutional Networks," *3DV*, 2017.

[50] D. J. Rezende, S. M. A. Eslami, S. Mohamed, P. Battaglia, M. Jaderberg, and N. Heess, "Unsupervised Learning of 3D Structure from Images," *NIPS*, 2016.

[51] A. Kar, C. Häne, and J. Malik, "Learning a Multi-View Stereo Machine," *NIPS*, 2017.

[52] M. Ji, J. Gall, H. Zheng, Y. Liu, and L. Fang, "SurfaceNet: An End-to-end 3D Neural Network for Multiview Stereopsis," *ICCV*, 2017.

[53] T. Whelan, J. McDonald, M. Kaess, M. Fallon, H. Johannsson, and J. J. Leonard, "Kintinuous: Spatially Extended Kinectfusion," *RSS Workshops*, 2012.

[54] T. Whelan, S. Leutenegger, R. F. Salas-moreno, B. Glocker, and A. J. Davison, "ElasticFusion : Dense SLAM Without A Pose Graph," *RSS*, 2015.

[55] V. Blanz and T.Vetter, "Face Recognition based on Fitting a 3D Morphable Model," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 9, pp. 1063–1074, 2003.

[56] P. Dou, S. K. Shah, and I. A. Kakadiaris, "End-to-end 3D face reconstruction with deep neural networks," *CVPR*, 2017.

[57] A. Kar, S. Tulsiani, J. Carreira, and J. Malik, "Category-specific object reconstruction from a single image," *CVPR*, 2015.

[58] J. Gwak, C. B. Choy, M. Chandraker, A. Garg, and S. Savarese, "Weakly supervised 3D Reconstruction with Adversarial Constraint," *3DV*, 2017.

[59] S. Tulsiani, T. Zhou, A. A. Efros, and J. Malik, "Multi-view Supervision for Single-view Reconstruction via Differentiable Ray Consistency," *CVPR*, 2017.

[60] X. Yan, J. Yang, E. Yumer, Y. Guo, and H. Lee, "Perspective Transformer Nets: Learning Single-View 3D Object Reconstruction without 3D Supervision," *NIPS*, 2016.

[61] C. Kong, C.-H. Lin, and S. Lucey, "Using Locally Corresponding CAD Models for Dense 3D Reconstructions from a Single Image," *CVPR*, 2017.

[62] A. Kurenkov, J. Ji, A. Garg, V. Mehta, J. Gwak, C. Choy, and S. Savarese, "DeformNet: Free-Form Deformation Network for 3D Shape Reconstruction from a Single Image," *NIPS*, 2017.

[63] J. K. Murthy, G. V. S. Krishna, F. Chhaya, and K. M. Krishna, "Reconstructing Vechicles from a Single Image: Shape Priors for Road Scene Understanding," *ICRA*, 2017.

[64] A. Johnston, R. Garg, G. Carneiro, I. Reid, and A. v. d. Hengel, "Scaling CNNs for High Resolution Volumetric Reconstruction from a Single Image," *ICCV Workshops*, 2017.

[65] C.-H. Lin, C. Kong, and S. Lucey, "Learning Efficient Point Cloud Generation for Dense 3D Object Reconstruction," *AAAI*, 2018.

[66] J. Wu, Y. Wang, T. Xue, X. Sun, W. T. Freeman, and J. B. Tenenbaum, "MarrNet: 3D Shape Reconstruction via 2.5D Sketches," *NIPS*, 2017.

[67] M. Firman, O. M. Aodha, S. Julier, and G. J. Brostow, "Structured Prediction of Unobserved Voxels From a Single Depth Image," *CVPR*, 2016.

[68] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser, "Semantic Scene Completion from a Single Depth Image," *CVPR*, 2017.

[69] W. Wang, Q. Huang, S. You, C. Yang, and U. Neumann, "Shape Inpainting using 3D Generative Adversarial Network and Recurrent Convolutional Networks," *ICCV*, 2017.

[70] C. Zou, E. Yumer, J. Yang, D. Ceylan, and D. Hoiem, "3D-PRNN: Generating Shape Primitives with Recurrent Neural Networks," *ICCV*, 2017.

[71] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network," *CVPR*, 2017.

[72] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative Adversarial Text to Image Synthesis," *ICML*, 2016.

[73] A. B. L. Larsen, S. K. Sonderby, H. Larochelle, and O. Winther, "Autoencoding beyond pixels using a learned similarity metric," *ICML*, 2016.

[74] M. Gadelha, S. Maji, and R. Wang, "3D Shape Induction from 2D Views of Multiple Objects," *3DV*, 2017.

[75] E. Smith and D. Meger, "Improved Adversarial Systems for 3D Object Generation and Reconstruction," *CoRL*, 2017.

[76] A. A. Soltani, H. Huang, J. Wu, T. D. Kulkarni, and J. B. Tenenbaum, "Synthesizing 3D Shapes via Modeling Multi-View Depth Maps and Silhouettes with Deep Generative Networks," *CVPR*, 2017.

[77] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," *ICML*, 2017.

[78] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An Information-Rich 3D Model Repository," *arXiv*, 2015.

[79] L. v. d. Maaten and G. Hinton, "Visualizing Data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[80] O. Ronneberger, P. Fischer, and T. Brox, "U-Net : Convolutional Networks for Biomedical Image Segmentation," *MICCAI*, 2015.

[81] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved Training of Wasserstein GANs," *NIPS*, 2017.

[82] M. Arjovsky and L. Bottou, "Towards Principled Methods for Training Generative Adversarial Networks," *ICLR*, 2017.

[83] J. Bao, D. Chen, F. Wen, H. Li, and G. Hua, "CVAE-GAN: Fine-Grained Image Generation through Asymmetric Training," *ICCV*, 2017.

[84] Y. Mroueh, T. Sercu, and V. Goel, "McGAN: Mean and Covariance Feature Matching GAN," *ICML*, 2017.

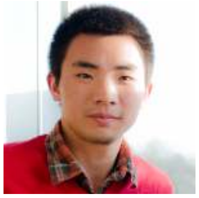
[85] A. Brock, T. Lim, J. M. Ritchie, and N. Weston, "Generative and Discriminative Voxel Modeling with Convolutional Neural Networks," *NIPS Workshops*, 2016.

[86] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *ICLR*, 2015.

[87] K. Khoshelham and S. O. Elberink, "Accuracy and Resolution of Kinect Depth Data for Indoor Mapping Applications," *Sensors*, vol. 12, pp. 1437–1454, 2012.

[88] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *ICLR*, 2015.

[89] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," *ICCV*, 2017.



Bo Yang is currently a DPhil candidate in the department of Computer Science at University of Oxford. His research interests lie in deep learning, computer vision and robotics.

Stefano Rosa received his Ph.D. in Mechatronics Engineering from Politecnico di Torino, Italy, in 2014. He is currently a research fellow in the Department of Computer Science at University of Oxford, UK, working on long-term navigation, Human-Robot Interaction and intuitive physics.

Andrew Markham is an Associate Professor at the Department of Computer Science, University of Oxford. He obtained his BSc (2004) and PhD (2008) degrees from the University of Cape Town, South Africa. He is the Director of the MSc in Software Engineering. He works on resource-constrained systems, positioning systems, in particular magneto-inductive positioning and machine intelligence.

Niki Trigoni is a Professor at the Oxford University Department of Computer Science and a fellow of Kellogg College. She obtained her DPhil at the University of Cambridge (2001), became a postdoctoral researcher at Cornell University (2002-2004), and a Lecturer at Birkbeck College (2004-2007). At Oxford, she is currently Director of the EPSRC Centre for Doctoral Training on Autonomous Intelligent Machines and Systems, a program that combines machine learning, robotics, sensor systems and verification/control. She also leads the Cyber Physical Systems Group http://www.cs.ox.ac.uk/activities/sensors/index.html, which is focusing on intelligent and autonomous sensor systems with applications in positioning, healthcare, environmental monitoring and smart cities. The groups research ranges from novel sensor modalities and low level signal processing to high level inference and learning.

Hongkai Wen is an Assistant Professor at the Department of Computer Science, University of Warwick. Before that he was a post-doctoral researcher in the University of Oxford, where he obtained his D.Phil in Computer Science in 2014. His research interests lie in Cyber-Physical systems, human-centric sensing, and pervasive data science.