






# Leveraging Pyramidal Feature Hierarchy for 3D Reconstruction

Fairuz Safwan Mahad<sup>(✉)</sup>, Masakazu Iwamura, and Koichi Kise

Graduate School of Engineering, Osaka Prefecture University,  
1-1 Gakuen-cho, Naka, Sakai, Osaka 599-8531, Japan  
fsafwan88@gmail.com, {masa,kise}@cs.osakafu-u.ac.jp

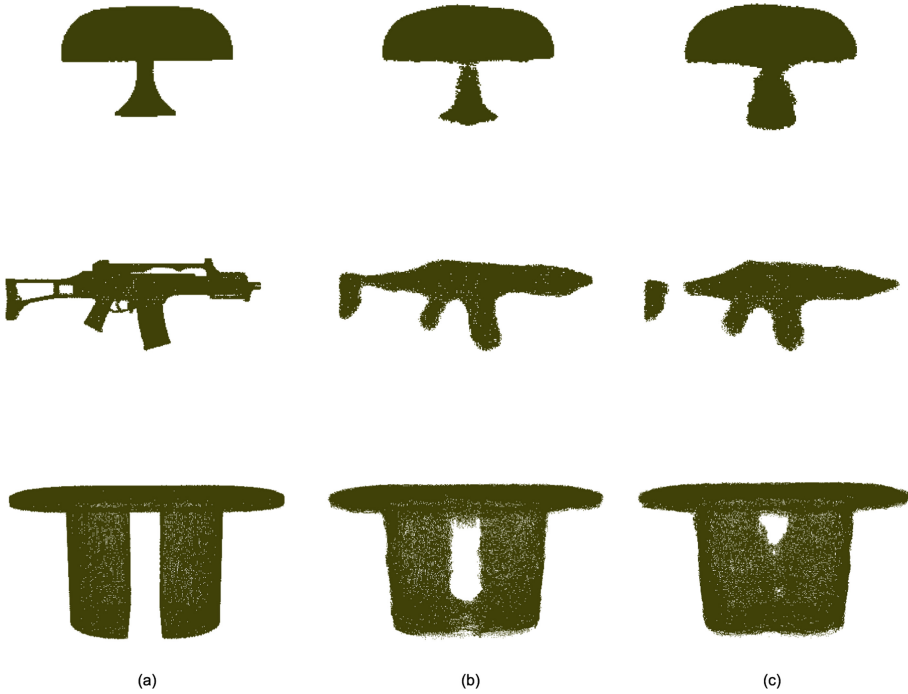
**Abstract.** Most state-of-the-art 3D reconstruction methods with CNNs have focused on completion and generalization of the reconstructed 3D models. Although the reconstructed 3D models may look complete, mostly lose out in detail causing a wider gap between the reconstructed 3D model and the groundtruth. We propose a method that adapts a pyramidal hierarchical-based network. Our strategy is to focus on improving the reconstruction of the detailed parts which comprise of thin and fine parts of the 3D models. Improving the detailed parts of the 3D model helps improve the accuracy and the overall shape of the 3D model resulting in a reconstructed 3D model which looks closer to the groundtruth. The advantage of using a pyramidal hierarchical-based network is that it builds a feature pyramid which considers high-level semantics at different scales. This builds a network that is tailored to focus on the detailed parts of the 3D model while considering the overall shape of the 3D model.

**Keywords:** Computer vision · 3D reconstruction · Deep learning

## 1 Introduction

3D reconstruction is an ill-posed problem which researchers have addressed for decades. The goal of 3D reconstruction is to reconstruct a 3D model from a given input. Ideally, the reconstructed 3D model has to look exactly the same as the input. However, this is far from easy to achieve as there are various challenges to overcome in 3D reconstruction. Conventional methods such as [7, 28, 32] reconstruct 3D models based on feature point correspondences. However, these methods are ineffective if there is a large baseline between viewpoints which makes finding feature point correspondences even more challenging or in the worst case, it fails. A typical solution is to acquire more viewpoints but this is not always convenient in most cases. Other disadvantages include occlusion and error in point correspondences.

In recent years, researches such as [4, 6, 12–16, 18, 26, 29, 30, 34–36] have started utilizing convolutional neural networks (CNNs) to reconstruct 3D models and have achieved considerable success. Unlike the conventional methods,



**Fig. 1.** Reconstructed 3D models of our proposed method and [33] from 20 depth map images with different viewpoints. (a) groundtruth. (b) Our results. (c) Soltani et al. [33].

CNNs-based methods do not require any feature point correspondences which conveniently overcomes one of the main problems in the conventional methods. However, these methods focus on the completion and generalization of the 3D models. The reconstructed 3D model often suffers from the loss in detail and resolution. Methods such as [20,33] have focused on achieving high resolution 3D models. Despite the high resolution 3D reconstructed models, they suffer from the loss in details as shown in Fig. 1(c). The mentioned detailed parts refer to the fine parts of the 3D models such as the stand of a lamp, the tip of a rifle, the stand of a chair or table. We saw a niche in further enhancing the quality of the reconstructed 3D model by focusing on the reconstruction of the detailed parts.

In this paper, we propose a simple yet effective way to further enhance the quality of the reconstructed models. Our network architecture is based on a pyramidal hierarchical-based network concept shown in Fig. 2 from [21]. [21] builds a multi-scale feature map where each feature map is made up of high-level semantic features with different spatial resolution. By leveraging the semantically strong features extracted at different levels, the reconstruction quality of the detailed parts of the 3D model can be further improved. [21] is actually designed for object detection but with a slight improvement we demonstrate that it is able

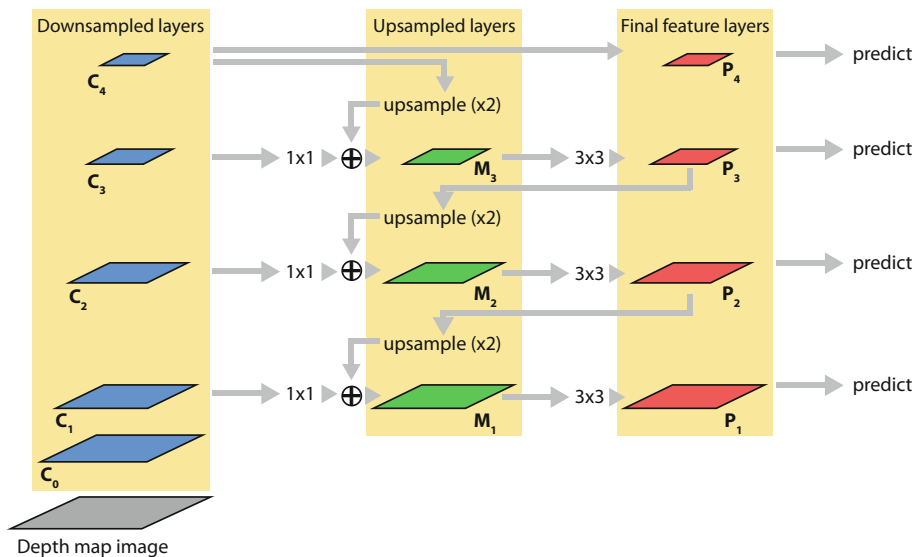


Fig. 2. Network architecture of Lin et al. [21].

to extract useful features at different scales with the purpose of enhancing the quality of the reconstructed 3D model as shown in Fig. 1. Our proposed method is an improved implementation of [33].

Our contributions are listed as follows:

- Implements a pyramidal hierarchical-based network in encoder
- Improves reconstruction of detailed parts which comprise of thin and fine parts of the 3D model
- Improves the overall shape and accuracy of the 3D reconstructed models.

## 2 Related Work

### 2.1 Single-View and Multi-view 3D Reconstruction Networks

3D reconstruction methods can be categorized in either single-view or multi-view. In a single-view 3D reconstruction method [6, 13, 15, 16, 18, 20, 29, 30, 34–36], a single image is required as input in order to reconstruct a 3D model. A single image is easily obtainable and can be convenient in certain cases. However, there are several downsides to it. A single-view 3D reconstruction is an ill-posed problem. Since only a single viewpoint is fed as input to the network, the network will estimate the unseen viewpoints of the object. In most cases, this produces a wider dissimilarity gap between the ground truth and the reconstructed 3D model. Furthermore, apart from relying heavily on how well the

network is trained, the reconstructed 3D model is also reliant on the angle of the single viewpoint that is fed as input to the network. This shows that a 3D model reconstructed from a single-view is more prone to suffer from loss in detail and low accuracy.

Multi-view 3D reconstruction methods however, involves two or more viewpoints. Unlike a single image, more viewpoints provide more information of the object which in turn produces a reconstructed 3D model with better accuracy. [12, 14] have proposed 3D reconstruction methods which reconstruct 3D models in a rasterized form which is in voxel. Voxels are basically an extended form of 2D pixels which are used to represent pixels in 3D. Voxels are easily integrated in CNN. However, it is not efficient due to its high memory consumption. On the contrary, [33] have proposed a method that reconstruct 3D models in a geometric form, point clouds. Despite its high resolution reconstructed 3D model, it focused on completion and generalization which in most cases failed in capturing the detailed parts of the objects.

## 2.2 Pyramidal-Based Networks

Exploiting features extracted from different layers or in other words, multi-scale layers have been practiced since the early days of neural networks. This technique is especially common in addressing issues such as detection and segmentation. Methods such as [8, 9, 23, 24, 27, 31] focused on segmentation while [1, 2, 11, 19, 22] focused on detection. There are also other method such as [25] that deals with pose estimation. Methods such as [8, 11, 25, 27, 31] have progressed even further by implementing skip connections that connects features from previous layers to current layers which achieved promising results. Although all of these methods adopt pyramidal-based architectures, [21] in particular stood out from the rest as predictions are made independently at all levels.

## 3 Approach

Our proposed method is an improved implementation of Soltani et al. [33]. [33] is a 3D shape synthesizing network that estimates a set of depth map and silhouettes. The network is trained from a collection of depth map images rendered from the ShapeNet dataset [3] using 20 pre-defined camera positions. The method takes in either 20 depth map or 20 silhouette images as input. It outputs the same number of depth map and silhouette images which amounts to a total of 40 output images. These output images are used to obtain a final 3D reconstructed model. In order to do so, each output depth map image is used to generate a group of point clouds. Accumulating all the groups of point clouds forms an initial 3D model. The output silhouette images are used to further

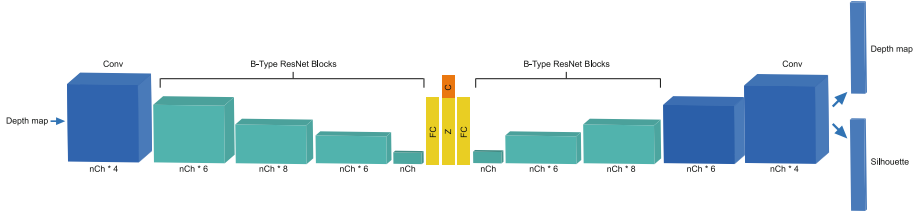


Fig. 3. Network architecture of Soltani et al. [33].

refine the initial 3D model by removing noisy points which ends up creating a final reconstructed 3D model with high resolution. Furthermore, [33] can be trained in three different ways. It can work with 20 views, a single view or 20 views with 15 to 18 of them being randomly zero out. However, the trained network cannot work with all of the three settings simultaneously. It can only be trained on any one of the three settings at a time. In other words, one would need to train three different networks in order to work with all three settings. In this paper, we will only train our network on the first setting which is working with 20 views. An overview of the network architecture of [33] is shown in Fig. 3. It implements a deep generative network, Variational Autoencoder (VAE) [17]. Both the encoder and decoder are made up of ResNet blocks [10]. Although it is able to reconstruct a high resolution 3D model, the network is designed for completion and generalization. It learns single-scaled features. Following the nature of the network, most of the detailed parts are being left out and not reconstructed.

Our goal is to further enhance the quality of the reconstructed 3D model. In order to do so, our strategy is to focus more on capturing features at tight spots such as small and thin parts of the object. This improves the quality of the reconstructed 3D model hence improving its accuracy. For this purpose, we implemented the multi-scale layered network with skip connections from [21] as shown in Fig. 2, in the encoder part of the VAE network structure used in [33]. According to [21], the multi-scale upsampled layers are made up of semantically stronger features than the downsampled layers which is the main advantage of using pyramidal-based networks. Figure 2 shows that [21] utilizes every output level  $\{P_1, P_2, P_3, P_4\}$  to make predictions independently. Our proposed network architecture shown in Fig. 4 adapts similar concept but with two distinct differences. Firstly, [21] considers every output level  $\{P_1, P_2, P_3, P_4\}$  independently while our proposed network concatenates the final feature maps  $\{P_2, P_4\}$  producing a final merged feature map followed by a fully-connected layer. Secondly, our proposed network is implemented in the encoder of a VAE structure.

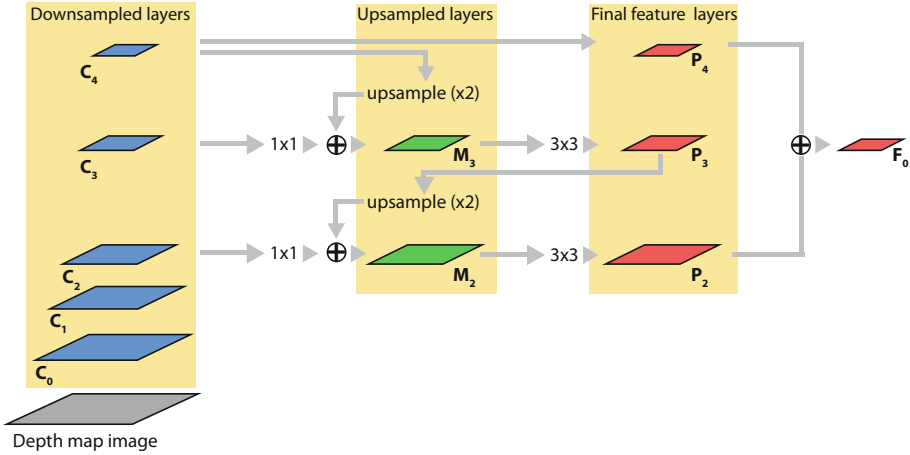


Fig. 4. Our proposed network architecture.

### 3.1 Network Architecture

Figure 4 shows the network architecture of our proposed method which we adapted from [21]. It features bottom-up and top-down pathways with skip connections. All blocks in both the bottom-up and top-down pathways are made up of ResNet blocks [10]. The network starts with a bottom-up pathway by scaling down the input image to feature maps of sizes  $\{110^2, 53^2, 25^2, 11^2, 4^2\}$  which we denote them as  $\{C_0, C_1, C_2, C_3, C_4\}$  respectively. The bottom-up pathway ends with  $C_4$  as its last layer.

The top-down pathway begins by first upsampling the last layer which is  $C_4$  back to a feature map of size  $11^2$ . The layer  $C_3$  (which is of the same spatial size as the upsampled layer from  $C_4$ ) from the bottom-up pathway is associated with the upsampled layer with a  $1 \times 1$  convolution. Similar to [21], the features in the upsampled layer and the skip connection layer are then concatenated which produce a feature map denoted as  $M_3$ . The layer  $M_3$  goes through a  $3 \times 3$  convolution which acts as an anti-aliasing measure to reduce the aliasing effect caused by sampling the layers, producing a final feature map for this level denoted as  $\{P_3\}$ . This iterates until we have the final features maps denoted as  $\{P_2, P_3, P_4\}$ . It is to note that  $\{P_4\}$  is the same layer as  $\{C_4\}$ . Finally, we concatenate the final feature maps  $\{P_2, P_4\}$  to produce a final merged feature map denoted as  $\{F_0\}$  followed by a fully connected layer. In order to concatenate the final feature maps  $\{P_2\}$  and  $\{P_4\}$ , we had to further downsample the layer  $\{P_2\}$  in order to match the size of layer  $\{P_4\}$ . We did not include  $\{P_3\}$  in the final merged feature map in order to reduce the number of parameters. The same reason applies to not further upsampling the layer  $\{P_2\}$  to spatial size of  $\{110^2, 53^2\}$ . Our proposed network architecture is implemented only in the encoder of the VAE structure of [33]. We used the same decoder structure as [33].

The network in [33] can be trained either in an unsupervised manner or conditionally. For our purpose, we will train our network using the unsupervised method. Therefore, we use the same loss function as Equation (1) in [33].

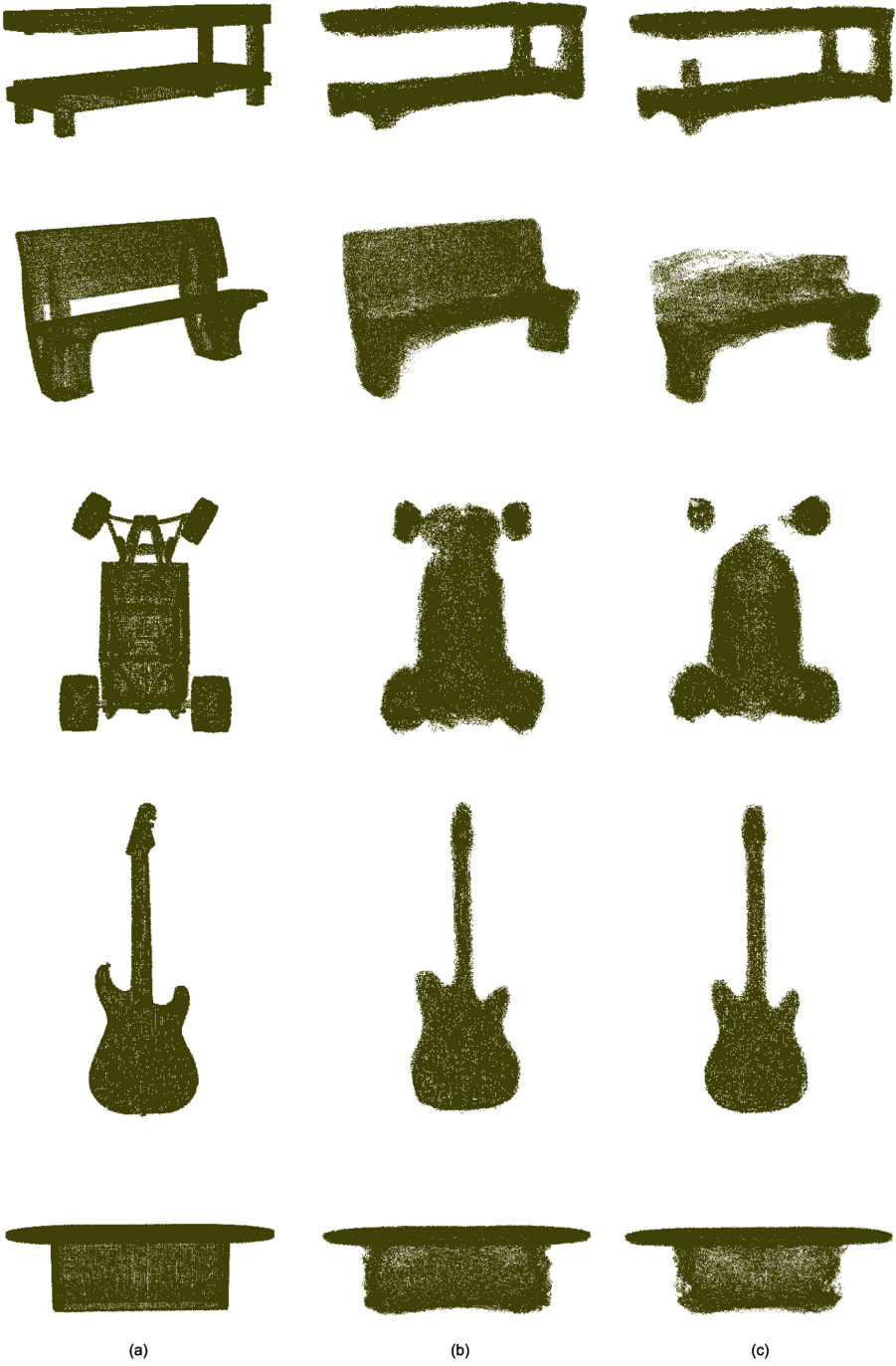
## 4 Experiments

We evaluated our proposed method against Soltani et al. [33]. Similar to [33], we trained our network using the ShapeNet dataset [3]. The ShapeNet dataset consists of 57 object categories such as aeroplane, car, table, chair, vase. It involved a total of 56,652 3D models. Each 3D model was first rendered into 20 depth map images with distinct views. All 3D models are rendered with the same set of pre-defined camera angles. Each rendered depth map image was of size  $224 \times 224$ . In order to evaluate our method against Soltani et al. [33], we used the same dataset distribution as [33] which was 92.5% and 7.5% for training and validation respectively. We also used their pre-trained model which was used to produce the results in [33]. We present our results qualitatively and quantitatively.

### 4.1 Qualitative Evaluation

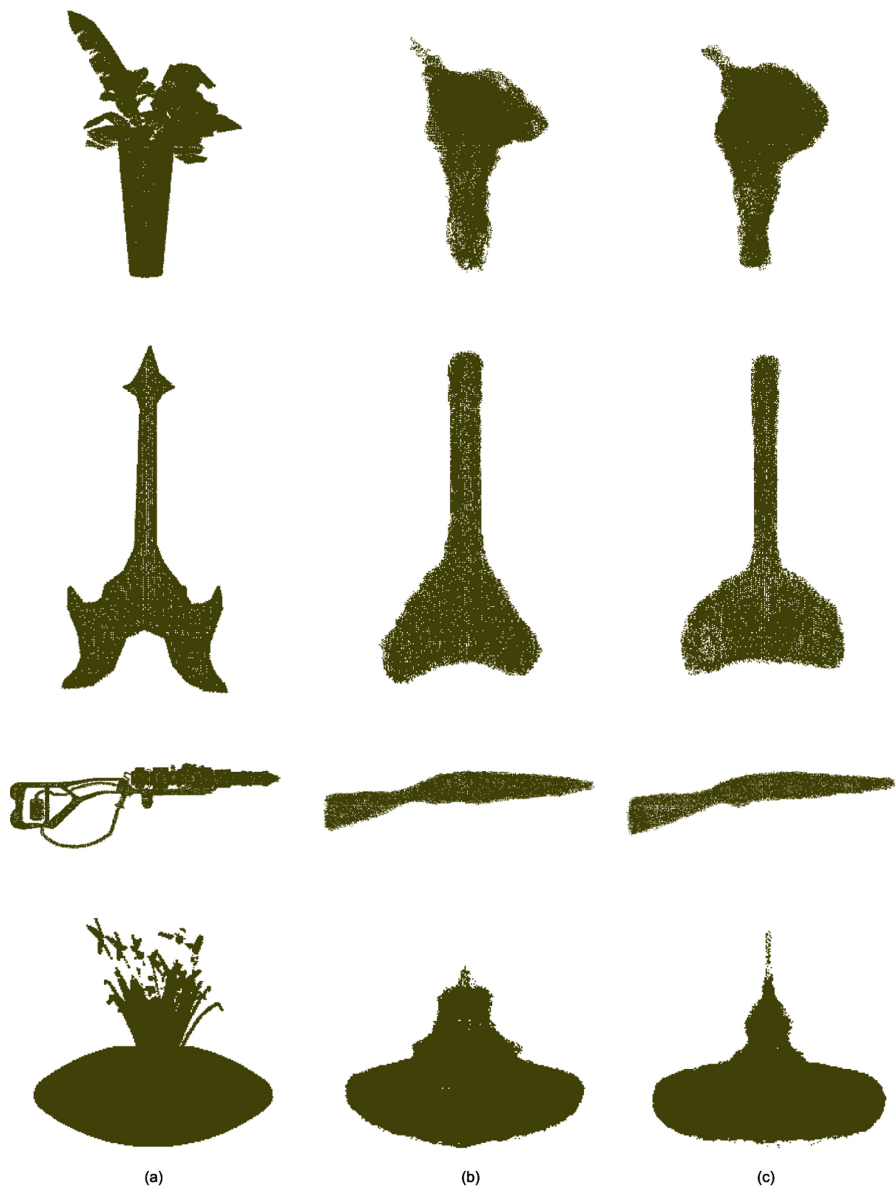
In this section, we present our qualitative results. We reconstructed the 3D models from the test dataset and compared our results against the groundtruth and Soltani et al. [33]. The reconstructed 3D models are all represented in point cloud form and rendered using [5]. Figure 5 shows the results for several categories such as bench, car, guitar and table. The results in Fig. 5 show that the proposed method is able to reconstruct a denser model as compared to [33]. Furthermore, it is able to capture the overall shape of the 3D model better as compared to [33]. These results in a 3D reconstructed model looking much closer to the groundtruth. Apart from reconstructing parts that could not be reconstructed by [33] such as in row 2 and 3 of Fig. 5, it can also be observed that our proposed method is able to eliminate the incorrectly reconstructed parts such as in the first row in Fig. 5. This proves that concatenating the multiple-level features from the deepest layer (top-most layer of the pyramid) and the lowest layer of the feature pyramid is effective in refining parts of the reconstructed 3D model. The contributing factor lies in the multiple-level feature maps,  $\{P_2\}$  and  $\{P_4\}$ , which have features from different levels. Concatenating features extracted on different levels allows the network to learn both coarse and fine details of the image.

However, our proposed method failed under certain circumstances. As shown in Fig. 6, our proposed method could not cope with reconstructing 3D models with complicated shapes. It especially did not work well for objects with uncommon shapes such as in row 2 and 3. Our goal is to improve the reconstruction of the fine parts of the 3D models. Therefore, although our results in Fig. 6 are better shaped than Soltani et al. [33], they are still considered a failure as the fine parts are not reconstructed.



**Fig. 5.** Qualitative results depicting 3D reconstructed models for success cases. (a) groundtruth. (b) Our results. (c) Soltani et al. [33].





**Fig. 6.** Qualitative results depicting 3D reconstructed models for failure cases. (a) groundtruth. (b) Our results. (c) Soltani et al. [33].

## 4.2 Quantitative Evaluation

In this section, we compare our results against Soltani et al. [33] quantitatively. We used the same evaluation metric as Soltani et al. [33] which is Intersection-over-Union (IoU) in order to provide a fair comparison between the results of ours and Soltani et al. [33]. The IoU is computed by converting the projected

point clouds from the estimated depth map into 3D voxels. As mentioned, the ShapeNet dataset [3] involves 57 categories. All categories do not have an equal distribution of the number of models. A category can have as many as over 500 models such as the table and car category while others can have as few as 2 models such as the birdhouse category. Due to the uneven distribution of models, we provided a breakdown of the IoU for each category for both Soltani et al. [33] and our results. We compare our results for epochs 80, 90 and 100. A random seed is used which causes fluctuation in the values every time the depth map is estimated from the model. Therefore, in order to provide a more insightful data, we ran the depth map estimation 10 times for each epoch (epoch 80, 90 and 100) and calculated the mean from the 10 IoU values. Similarly, we calculated the mean from the 10 IoU values category-wise. In addition, we calculated the standard deviation for each category using the 10 IoU values obtained from the 10 separate depth map estimation. The average IoU obtained in [33] was 84.0 at epoch 80. In order to provide a fair comparison between ours and Soltani et al. [33], similar to our experiment settings, we estimated the depth map using the test dataset for 10 times using the pre-trained model provided by Soltani et al. It is to be noted that the provided pre-trained model was the exact same model used to produce the results in [33]. However, after estimating the depth map for 10 times, we achieved an average IoU of 80.9, 81.1 and 81.0 for epoch 80, 90 and 100 respectively. The computed standard deviation for Soltani et al. [33] shown in Table 1 proves that the difference is very minimal. Therefore, we used these IoU values as our benchmark for Soltani et al. [33]. It can be seen in Table 1 that our proposed method outperforms Soltani et al. [33] with an IoU value of 81.5, 81.6 and 81.8 for epoch 80, 90 and 100 respectively. The values in bold shown in Table 1 indicates that it is higher than its compared value for each given category for the respective epoch. Both values are marked as bold if they have the same IoU value. Breaking down the results in category-wise, our proposed method outperforms Soltani et al. [33] in 40 categories for epoch 80, 41 categories for epoch 90 and 47 categories for epoch 100 out of 57 categories.

In Table 2, we compare our best performing model against Soltani et al. [33] which is at epoch 100 and 90 respectively. Our proposed method gained an average IoU of 0.7 over Soltani et al. [33] with 43 out of 57 categories falling in our favour while achieving the same IoU values for three categories which are bus, display and rifle. We observed that all eleven categories that we fell behind all have less than 50 models each with only one category having 85 models. These quantitative results show that our proposed method is able to improve the reconstruction quality for majority of the categories. Our proposed method is able improve the quality of the reconstruction while focusing on both shape completion and also on the fine parts of the models. The improvement in IoU in categories such as chair, lamp and table as shown in Table 2 further supports this claim. However, not all categories with fine parts excelled with our proposed method. Our proposed method caused a decline in IoU values in categories such as tower, motorcycle and faucet. This is due to the complex shapes that these categories have. On top of the complexity of the shapes, these categories have very few models for the network to train with which weakens the effectiveness of the proposed method.

**Table 1.** IoU comparison between [33] and ours. “#M” indicates #Model.

Category	#M	Epoch 80		Epoch 90		Epoch 100	
		[33]	Ours	[33]	Ours	[33]	Ours
Aeroplane	304	72.3 ±0.001	<b>72.8</b> ±0.002	72.4 ±0.001	<b>72.7</b> ±0.001	72.2 ±0.001	<b>72.6</b> ±0.001
Bag	8	78.8 ±0.007	<b>80.2</b> ±0.007	79.0 ±0.006	<b>80.6</b> ±0.006	79.0 ±0.006	<b>79.7</b> ±0.004
Basket	11	84.0 ±0.007	<b>85.3</b> ±0.007	83.8 ±0.005	<b>84.5</b> ±0.006	83.8 ±0.006	<b>84.7</b> ±0.005
Bathtub	62	87.7 ±0.003	<b>88.8</b> ±0.003	87.7 ±0.004	<b>87.8</b> ±0.004	88.1 ±0.004	<b>88.3</b> ±0.003
Bed	10	75.6 ±0.009	<b>78.5</b> ±0.007	76.0 ±0.006	<b>77.2</b> ±0.005	75.9 ±0.005	<b>77.6</b> ±0.004
Bench	132	78.1 ±0.002	<b>78.6</b> ±0.002	78.2 ±0.003	<b>78.3</b> ±0.002	78.1 ±0.002	<b>78.6</b> ±0.003
Bicycle	6	<b>43.6</b> ±0.013	43.5 ±0.013	<b>43.2</b> ±0.01	38.6 ±0.012	<b>43.2</b> ±0.011	40.3 ±0.01
Birdhouse	2	81.4 ±0.033	<b>86.6</b> ±0.03	83.7 ±0.033	<b>85.9</b> ±0.018	82.9 ±0.021	<b>87.3</b> ±0.027
Bookshelf	39	73.3 ±0.003	<b>75.8</b> ±0.006	73.7 ±0.008	<b>74.6</b> ±0.006	73.8 ±0.007	<b>74.4</b> ±0.003
Bottle	35	92.5 ±0.003	<b>92.6</b> ±0.003	<b>92.7</b> ±0.002	91.5 ±0.003	92.7 ±0.002	<b>93.0</b> ±0.002
Bowl	13	93.6 ±0.004	<b>93.7</b> ±0.006	<b>93.5</b> ±0.004	92.5 ±0.005	<b>93.8</b> ±0.005	93.4 ±0.006
Bus	82	<b>91.2</b> ±0.002	90.9 ±0.002	<b>91.1</b> ±0.001	<b>91.1</b> ±0.002	91.3 ±0.002	<b>91.8</b> ±0.002
Cabinet	115	90.7 ±0.001	<b>91.6</b> ±0.002	90.8 ±0.002	<b>91.2</b> ±0.002	90.9 ±0.002	<b>91.4</b> ±0.002
Camera	8	67.5 ±0.004	<b>68.2</b> ±0.008	68.1 ±0.006	<b>69.9</b> ±0.008	67.9 ±0.005	<b>69.8</b> ±0.009
Can	8	94.7 ±0.007	<b>95.3</b> ±0.005	<b>94.7</b> ±0.004	94.1 ±0.005	<b>94.8</b> ±0.007	94.6 ±0.006
Cap	3	<b>79.2</b> ±0.01	77.1 ±0.02	<b>81.0</b> ±0.018	79.2 ±0.027	78.6 ±0.011	<b>80.5</b> ±0.01
Car	554	82.6 ±0.001	<b>82.7</b> ±0.001	<b>82.5</b> ±0.001	82.4 ±0.001	82.7 ±0.001	<b>82.8</b> ±0.001
Cellphone	45	91.6 ±0.005	<b>92.4</b> ±0.003	<b>91.7</b> ±0.003	91.6 ±0.004	92.4 ±0.004	<b>92.6</b> ±0.003
Chair	491	77.5 ±0.001	<b>78.2</b> ±0.001	<b>77.7</b> ±0.001	77.3 ±0.001	77.5 ±0.001	<b>78.6</b> ±0.001
Clock	40	<b>80.4</b> ±0.004	79.8 ±0.004	80.1 ±0.004	<b>81.2</b> ±0.005	80.5 ±0.004	<b>82.3</b> ±0.006
Dishwasher	8	93.4 ±0.005	<b>94.8</b> ±0.002	<b>94.0</b> ±0.004	93.1 ±0.006	<b>94.4</b> ±0.005	<b>93.5</b> ±0.005
Display	81	<b>86.3</b> ±0.002	86.0 ±0.004	86.5 ±0.003	<b>86.6</b> ±0.002	86.4 ±0.002	<b>87.6</b> ±0.004
Faucet	47	<b>66.0</b> ±0.007	64.6 ±0.007	<b>66.8</b> ±0.009	65.6 ±0.007	<b>66.8</b> ±0.008	<b>66.8</b> ±0.008
File cabinet	18	92.4 ±0.003	<b>92.6</b> ±0.004	<b>92.8</b> ±0.004	92.2 ±0.006	<b>92.6</b> ±0.005	92.0 ±0.005
Flowerpot	42	65.5 ±0.002	<b>66.4</b> ±0.003	65.5 ±0.003	<b>65.8</b> ±0.003	65.5 ±0.002	<b>66.1</b> ±0.004
Guitar	65	78.3 ±0.006	<b>79.7</b> ±0.005	78.4 ±0.004	<b>79.3</b> ±0.004	78.3 ±0.004	<b>81.9</b> ±0.003
Headphone	5	55.4 ±0.009	<b>55.6</b> ±0.015	55.6 ±0.01	<b>60.3</b> ±0.005	55.1 ±0.007	<b>62.6</b> ±0.012
Helmet	16	<b>75.5</b> ±0.003	<b>75.5</b> ±0.002	<b>75.4</b> ±0.002	75.1 ±0.003	75.3 ±0.004	<b>75.9</b> ±0.004
Keyboard	5	<b>87.9</b> ±0.013	87.3 ±0.007	87.8 ±0.009	<b>91.4</b> ±0.009	88.1 ±0.01	<b>90.9</b> ±0.008
Knife	42	78.3 ±0.007	<b>78.8</b> ±0.011	<b>79.4</b> ±0.006	78.4 ±0.015	78.7 ±0.006	<b>79.2</b> ±0.008
Lamp	181	67.2 ±0.003	<b>68.2</b> ±0.004	<b>67.8</b> ±0.003	67.0 ±0.005	67.3 ±0.005	<b>68.4</b> ±0.005
Laptop	34	96.6 ±0.001	<b>96.9</b> ±0.003	96.8 ±0.002	<b>97.0</b> ±0.003	96.7 ±0.002	<b>97.3</b> ±0.001
Letterbox	7	<b>70.1</b> ±0.011	68.2 ±0.01	70.1 ±0.007	<b>71.6</b> ±0.006	69.8 ±0.013	<b>72.5</b> ±0.009
Microphone	6	62.1 ±0.007	<b>65.2</b> ±0.053	<b>62.8</b> ±0.008	62.1 ±0.033	62.5 ±0.008	<b>68.8</b> ±0.009
Microwave	11	<b>93.8</b> ±0.003	93.2 ±0.006	93.6 ±0.003	<b>93.7</b> ±0.003	93.4 ±0.004	<b>94.2</b> ±0.004
Motorcycle	28	<b>75.4</b> ±0.004	75.0 ±0.004	<b>75.4</b> ±0.003	74.8 ±0.003	75.0 ±0.006	<b>75.8</b> ±0.003
Mug	17	84.4 ±0.006	<b>84.6</b> ±0.004	<b>84.2</b> ±0.004	83.0 ±0.003	<b>84.4</b> ±0.004	84.0 ±0.002
Piano	13	79.4 ±0.003	<b>80.3</b> ±0.005	79.4 ±0.004	<b>80.1</b> ±0.005	79.2 ±0.005	<b>80.6</b> ±0.007
Pillow	6	86.9 ±0.012	<b>87.8</b> ±0.014	<b>86.9</b> ±0.009	<b>84.0</b> ±0.009	<b>86.7</b> ±0.01	86.5 ±0.013
Pistol	19	84.4 ±0.007	<b>85.3</b> ±0.008	84.7 ±0.006	<b>85.8</b> ±0.004	84.5 ±0.006	<b>86.8</b> ±0.007
Printer	18	79.1 ±0.007	<b>81.3</b> ±0.005	79.2 ±0.007	<b>80.8</b> ±0.008	79.3 ±0.006	<b>80.7</b> ±0.004
Remote control	4	90.9 ±0.009	<b>92.3</b> ±0.005	89.4 ±0.268	<b>91.2</b> ±0.012	<b>91.0</b> ±0.006	89.6 ±0.005
Rifle	171	<b>77.5</b> ±0.003	<b>77.5</b> ±0.003	<b>77.6</b> ±0.003	77.0 ±0.003	77.2 ±0.003	<b>78.2</b> ±0.003
Rocket	7	72.4 ±0.01	<b>73.1</b> ±0.011	<b>73.2</b> ±0.012	71.0 ±0.012	71.8 ±0.015	<b>72.3</b> ±0.011

*(continued)*

**Table 1.** (*continued*)

Category	#M	Epoch 80		Epoch 90		Epoch 100	
		[33]	Ours	[33]	Ours	[33]	Ours
Ship	147	<b>79.3</b> $\pm 0.003$	79.2 $\pm 0.001$	<b>79.5</b> $\pm 0.002$	78.9 $\pm 0.003$	79.3 $\pm 0.001$	<b>80.2</b> $\pm 0.002$
Skateboard	18	78.9 $\pm 0.017$	<b>80.5</b> $\pm 0.013$	<b>80.7</b> $\pm 0.009$	79.8 $\pm 0.012$	80.0 $\pm 0.013$	<b>82.0</b> $\pm 0.009$
Sofa	242	87.2 $\pm 0.001$	<b>87.3</b> $\pm 0.001$	<b>87.1</b> $\pm 0.001$	86.9 $\pm 0.001$	87.1 $\pm 0.001$	<b>87.2</b> $\pm 0.001$
Speaker	121	84.1 $\pm 0.003$	<b>84.3</b> $\pm 0.002$	<b>84.1</b> $\pm 0.002$	<b>84.1</b> $\pm 0.002$	84.0 $\pm 0.002$	<b>84.4</b> $\pm 0.002$
Stove	8	88.2 $\pm 0.009$	<b>88.8</b> $\pm 0.005$	88.5 $\pm 0.006$	<b>88.9</b> $\pm 0.006$	88.5 $\pm 0.006$	<b>90.4</b> $\pm 0.008$
Table	652	<b>84.6</b> $\pm 0.001$	<b>84.6</b> $\pm 0.001$	<b>84.8</b> $\pm 0.001$	84.4 $\pm 0.001$	84.5 $\pm 0.001$	<b>85.3</b> $\pm 0.001$
Telephone	92	92.8 $\pm 0.006$	<b>93.1</b> $\pm 0.003$	92.5 $\pm 0.006$	<b>93.0</b> $\pm 0.002$	93.0 $\pm 0.005$	<b>93.5</b> $\pm 0.004$
Tower	12	<b>76.3</b> $\pm 0.005$	74.8 $\pm 0.006$	<b>76.4</b> $\pm 0.006$	75.6 $\pm 0.005$	<b>76.5</b> $\pm 0.007$	<b>76.5</b> $\pm 0.005$
Train	25	84.2 $\pm 0.005$	<b>85.0</b> $\pm 0.005$	84.0 $\pm 0.005$	<b>84.6</b> $\pm 0.003$	84.3 $\pm 0.007$	<b>85.9</b> $\pm 0.005$
Trashcan	28	<b>85.4</b> $\pm 0.005$	<b>85.4</b> $\pm 0.002$	85.2 $\pm 0.004$	<b>85.4</b> $\pm 0.004$	85.5 $\pm 0.003$	<b>85.6</b> $\pm 0.005$
Vase	38	82.0 $\pm 0.004$	<b>83.2</b> $\pm 0.004$	<b>82.4</b> $\pm 0.004$	82.0 $\pm 0.002$	82.1 $\pm 0.004$	<b>83.6</b> $\pm 0.003$
Vessel	85	<b>81.0</b> $\pm 0.002$	80.5 $\pm 0.003$	<b>81.1</b> $\pm 0.003$	80.2 $\pm 0.002$	81.2 $\pm 0.002$	<b>81.5</b> $\pm 0.003$
Washing machine	17	93.0 $\pm 0.002$	<b>93.2</b> $\pm 0.003$	92.4 $\pm 0.277$	<b>93.0</b> $\pm 0.003$	93.0 $\pm 0.003$	<b>93.2</b> $\pm 0.003$
<b>Average</b>		<b>80.9</b>	<b>81.5</b>	<b>81.1</b>	<b>81.6</b>	<b>81.0</b>	<b>81.8</b>

**Table 2.** IoU comparison between the best performance models for [33] and ours.

Category	#Model	[33] of epoch 90	Ours of epoch 100	Our gain
Aeroplane	304	72.4 $\pm 0.001$	<b>73.2</b> $\pm 0.001$	0.8
Bag	8	79.0 $\pm 0.006$	<b>80.2</b> $\pm 0.004$	1.2
Basket	11	83.8 $\pm 0.005$	<b>85.3</b> $\pm 0.005$	1.5
Bathtub	62	87.7 $\pm 0.004$	<b>89.1</b> $\pm 0.003$	1.4
Bed	10	76.0 $\pm 0.006$	<b>79.2</b> $\pm 0.004$	3.2
Bench	132	78.2 $\pm 0.003$	<b>79.2</b> $\pm 0.003$	1.0
Bicycle	6	43.2 $\pm 0.01$	<b>43.8</b> $\pm 0.01$	0.6
Birdhouse	2	83.7 $\pm 0.033$	<b>88.6</b> $\pm 0.027$	4.9
Bookshelf	39	73.7 $\pm 0.008$	<b>76.2</b> $\pm 0.003$	2.5
Bottle	35	92.7 $\pm 0.002$	<b>92.8</b> $\pm 0.002$	0.1
Bowl	13	93.5 $\pm 0.004$	<b>94.2</b> $\pm 0.006$	0.7
Bus	82	<b>91.1</b> $\pm 0.001$	<b>91.1</b> $\pm 0.002$	0.0
Cabinet	115	90.8 $\pm 0.002$	<b>91.6</b> $\pm 0.002$	0.8
Camera	8	68.1 $\pm 0.006$	<b>69.0</b> $\pm 0.009$	0.9
Can	8	94.7 $\pm 0.004$	<b>95.3</b> $\pm 0.006$	0.6
Cap	3	<b>81.0</b> $\pm 0.018$	77.9 $\pm 0.01$	-3.1
Car	554	82.5 $\pm 0.001$	<b>82.8</b> $\pm 0.001$	0.3
Cellphone	45	91.7 $\pm 0.003$	<b>92.5</b> $\pm 0.003$	0.8
Chair	491	77.7 $\pm 0.001$	<b>78.7</b> $\pm 0.001$	1.0
Clock	40	<b>80.1</b> $\pm 0.004$	79.9 $\pm 0.006$	-0.2

*(continued)*

**Table 2.** (continued)

Category	#Model	[33] of epoch 90	Ours of epoch 100	Our gain
Dishwasher	8	94.0 $\pm$ 0.004	<b>94.3</b> $\pm$ 0.005	0.3
Display	81	<b>86.5</b> $\pm$ 0.003	<b>86.5</b> $\pm$ 0.004	0.0
Faucet	47	<b>66.8</b> $\pm$ 0.009	65.2 $\pm$ 0.008	-1.6
Filecabinet	18	<b>92.8</b> $\pm$ 0.004	92.5 $\pm$ 0.005	-0.3
Flowerpot	42	65.5 $\pm$ 0.003	<b>66.6</b> $\pm$ 0.004	1.1
Guitar	65	78.4 $\pm$ 0.004	<b>80.2</b> $\pm$ 0.003	1.8
Headphone	5	55.6 $\pm$ 0.01	<b>56.8</b> $\pm$ 0.012	1.2
Helmet	16	75.4 $\pm$ 0.002	<b>75.8</b> $\pm$ 0.004	0.4
Keyboard	5	87.8 $\pm$ 0.009	<b>88.2</b> $\pm$ 0.008	0.4
Knife	42	<b>79.4</b> $\pm$ 0.006	79.0 $\pm$ 0.008	-0.4
Lamp	181	67.8 $\pm$ 0.003	<b>68.5</b> $\pm$ 0.005	0.7
Laptop	34	96.8 $\pm$ 0.002	<b>97.3</b> $\pm$ 0.001	0.5
Letterbox	7	<b>70.1</b> $\pm$ 0.007	68.1 $\pm$ 0.009	-2.0
Microphone	6	62.8 $\pm$ 0.008	<b>63.0</b> $\pm$ 0.009	0.2
Microwave	11	<b>93.6</b> $\pm$ 0.003	93.5 $\pm$ 0.004	-0.1
Motorcycle	28	<b>75.4</b> $\pm$ 0.003	75.3 $\pm$ 0.003	-0.1
Mug	17	84.2 $\pm$ 0.004	<b>84.6</b> $\pm$ 0.002	0.2
Piano	13	79.4 $\pm$ 0.004	<b>80.8</b> $\pm$ 0.007	1.4
Pillow	6	86.9 $\pm$ 0.009	<b>88.8</b> $\pm$ 0.013	1.9
Pistol	19	84.7 $\pm$ 0.006	<b>85.6</b> $\pm$ 0.007	0.9
Printer	18	79.2 $\pm$ 0.007	<b>82.0</b> $\pm$ 0.004	2.8
Remote control	4	89.4 $\pm$ 0.268	<b>92.0</b> $\pm$ 0.005	2.6
Rifle	171	<b>77.6</b> $\pm$ 0.003	<b>77.6</b> $\pm$ 0.003	0.0
Rocket	7	<b>73.2</b> $\pm$ 0.012	72.8 $\pm$ 0.011	-0.4
Ship	147	79.5 $\pm$ 0.002	<b>79.8</b> $\pm$ 0.002	0.3
Skateboard	18	80.7 $\pm$ 0.009	<b>80.9</b> $\pm$ 0.009	0.2
Sofa	242	87.1 $\pm$ 0.001	<b>87.6</b> $\pm$ 0.001	0.5
Speaker	121	84.1 $\pm$ 0.002	<b>84.3</b> $\pm$ 0.002	0.2
Stove	8	88.5 $\pm$ 0.006	<b>88.9</b> $\pm$ 0.008	0.4
Table	652	84.8 $\pm$ 0.001	<b>85.2</b> $\pm$ 0.001	0.4
Telephone	92	92.5 $\pm$ 0.006	<b>93.1</b> $\pm$ 0.004	0.6
Tower	12	<b>76.4</b> $\pm$ 0.006	75.9 $\pm$ 0.005	-0.5
Train	25	84.0 $\pm$ 0.005	<b>85.1</b> $\pm$ 0.005	1.1
Trashcan	28	85.2 $\pm$ 0.004	<b>85.5</b> $\pm$ 0.005	0.3
Vase	38	82.4 $\pm$ 0.004	<b>83.7</b> $\pm$ 0.003	1.3
Vessel	85	<b>81.1</b> $\pm$ 0.003	80.8 $\pm$ 0.003	-0.3
Washing machine	17	92.4 $\pm$ 0.277	<b>93.3</b> $\pm$ 0.003	0.9
<b>Average</b>		<b>81.1</b>	<b>81.8</b>	0.7

### 4.3 Conclusion

In order to improve the accuracy and overall shape of the 3D model, we focus on reconstructing the detailed parts of the 3D models that most state-of-the-art 3D reconstruction methods with CNNs have overlooked. We presented a simple yet effective way to improve the quality of the reconstructed 3D model by leveraging the advantage of a pyramidal hierarchical-based network. We show that building a feature pyramid of high-level semantics with different scales and concatenating the layers is able to reconstruct a denser 3D model while improving the reconstruction of the detailed parts. We demonstrated that our proposed method outperformed the state-of-the-art method in most categories and also in terms of overall IoU. However, our proposed method is not fully able to cope with the reconstruction of 3D models with uncommon and complex shapes.

### References

1. Bell, S., Lawrence Zitnick, C., Bala, K., Girshick, R.: Inside-outside net: detecting objects in context with skip pooling and recurrent neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2874–2883 (2016)
2. Cai, Z., Fan, Q., Feris, R.S., Vasconcelos, N.: A unified multi-scale deep convolutional neural network for fast object detection. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 354–370. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-46493-0\\_22](https://doi.org/10.1007/978-3-319-46493-0_22)
3. Chang, A.X., et al.: ShapeNet: an information-rich 3D model repository. arXiv preprint [arXiv:1512.03012](https://arxiv.org/abs/1512.03012) (2015)
4. Choy, C.B., Xu, D., Gwak, J.Y., Chen, K., Savarese, S.: 3D-R2N2: a unified approach for single and multi-view 3D object reconstruction. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9912, pp. 628–644. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-46484-8\\_38](https://doi.org/10.1007/978-3-319-46484-8_38)
5. Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., Ranzuglia, G.: MeshLab: an open-source mesh processing tool. In: Scarano, V., Chiara, R.D., Erra, U. (eds.) Eurographics Italian Chapter Conference. The Eurographics Association (2008). <https://doi.org/10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136>
6. Fan, H., Su, H., Guibas, L.J.: 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, pp. 605–613 (2017)
7. Furukawa, Y., Ponce, J.: Accurate, dense, and robust multiview stereopsis. IEEE Trans. Pattern Anal. Mach. Intell. **32**(8), 1362–1376 (2009)
8. Ghiasi, G., Fowlkes, C.C.: Laplacian pyramid reconstruction and refinement for semantic segmentation. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9907, pp. 519–534. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-46487-9\\_32](https://doi.org/10.1007/978-3-319-46487-9_32)
9. Hariharan, B., Arbeláez, P., Girshick, R., Malik, J.: Hypercolumns for object segmentation and fine-grained localization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 447–456 (2015)

10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
11. Honari, S., Yosinski, J., Vincent, P., Pal, C.: Recombinator networks: learning coarse-to-fine feature aggregation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5743–5752 (2016)
12. Ji, M., Gall, J., Zheng, H., Liu, Y., Fang, L.: SurfaceNet: an end-to-end 3D neural network for multiview stereopsis. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2307–2315 (2017)
13. Kanazawa, A., Tulsiani, S., Efros, A.A., Malik, J.: Learning category-specific mesh reconstruction from image collections. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11219, pp. 386–402. Springer, Cham (2018). [https://doi.org/10.1007/978-3-030-01267-0\\_23](https://doi.org/10.1007/978-3-030-01267-0_23)
14. Kar, A., Häne, C., Malik, J.: Learning a multi-view stereo machine. In: Advances in Neural Information Processing Systems, pp. 365–376 (2017)
15. Kato, H., Harada, T.: Learning view priors for single-view 3D reconstruction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 9778–9787 (2019)
16. Kato, H., Ushiku, Y., Harada, T.: Neural 3D mesh renderer. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3907–3916 (2018)
17. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint [arXiv:1312.6114](https://arxiv.org/abs/1312.6114) (2013)
18. Kong, C., Lin, C.H., Lucey, S.: 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, pp. 4857–4865 (2017)
19. Kong, T., Yao, A., Chen, Y., Sun, F.: HyperNet: towards accurate region proposal generation and joint object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 845–853 (2016)
20. Lin, C.H., Kong, C., Lucey, S.: Learning efficient point cloud generation for dense 3D object reconstruction. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
21. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2117–2125 (2017)
22. Liu, W., et al.: SSD: single shot multibox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 21–37. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-46448-0\\_2](https://doi.org/10.1007/978-3-319-46448-0_2)
23. Liu, W., Rabinovich, A., Berg, A.C.: Parsenet: looking wider to see better. arXiv preprint [arXiv:1506.04579](https://arxiv.org/abs/1506.04579) (2015)
24. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3431–3440 (2015)
25. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9912, pp. 483–499. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-46484-8\\_29](https://doi.org/10.1007/978-3-319-46484-8_29)
26. Niu, C., Li, J., Xu, K.: Im2Struct: recovering 3D shape structure from a single RGB image. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4521–4529 (2018)

27. Pinheiro, P.O., Lin, T.-Y., Collobert, R., Dollár, P.: Learning to refine object segments. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 75–91. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-46448-0\\_5](https://doi.org/10.1007/978-3-319-46448-0_5)
28. Pons, J.P., Keriven, R., Faugeras, O.: Modelling dynamic scenes by registering multi-view image sequences. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), vol. 2, pp. 822–827. IEEE (2005)
29. Pontes, J.K., Kong, C., Eriksson, A., Fookes, C., Sridharan, S., Lucey, S.: Compact model representation for 3D reconstruction. arXiv preprint [arXiv:1707.07360](https://arxiv.org/abs/1707.07360) (2017)
30. Pontes, J.K., Kong, C., Sridharan, S., Lucey, S., Eriksson, A., Fookes, C.: Image2Mesh: a learning framework for single image 3D reconstruction. In: Jawahar, C.V., Li, H., Mori, G., Schindler, K. (eds.) ACCV 2018. LNCS, vol. 11361, pp. 365–381. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-20887-5\\_23](https://doi.org/10.1007/978-3-030-20887-5_23)
31. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-24574-4\\_28](https://doi.org/10.1007/978-3-319-24574-4_28)
32. Snavely, N., Seitz, S.M., Szeliski, R.: Photo tourism: exploring photo collections in 3D. In: ACM Transactions on Graphics (TOG), vol. 25, pp. 835–846. ACM (2006)
33. Soltani, A.A., Huang, H., Wu, J., Kulkarni, T.D., Tenenbaum, J.B.: Synthesizing 3D shapes via modeling multi-view depth maps and silhouettes with deep generative networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1511–1519 (2017)
34. Sun, Y., Liu, Z., Wang, Y., Sarma, S.E.: Im2Avatar: Colorful 3D reconstruction from a single image. arXiv preprint [arXiv:1804.06375](https://arxiv.org/abs/1804.06375) (2018)
35. Tulsiani, S., Zhou, T., Efros, A.A., Malik, J.: Multi-view supervision for single-view reconstruction via differentiable ray consistency. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2626–2634 (2017)
36. Wang, N., Zhang, Y., Li, Z., Fu, Y., Liu, W., Jiang, Y.-G.: Pixel2Mesh: generating 3D mesh models from single RGB images. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11215, pp. 55–71. Springer, Cham (2018). [https://doi.org/10.1007/978-3-030-01252-6\\_4](https://doi.org/10.1007/978-3-030-01252-6_4)