# Occupancy Network based 3D Image Reconstruction using Single-Depth View

**Amit Baran Roy** **Aparajita Singh** **Sourav Chakraborty** **Tanmai Gajula**

Department of Computer Science,
University of Colorado Boulder,
Email: {Amit.Roy, Aparajita.Singh, Sourav.Chakraborty, taga5342}@colorado.edu

**Abstract**

3D reconstruction is a complex computer vision problem which has been explored by many for decades. It has various applications which help in environment reconstruction and medicine. In this project, we aim to acquire the complete 3D geometry of an object from a single 2.5D depth view by using deep learning techniques such as generative adversarial networks and 3D convolution neural networks. As an extension, we aim to improve the resolution of the final 3D voxelized output obtained, by transforming the voxel representation into another representation called occupancy networks.

*Keywords:* 3D reconstruction, occupancy networks, generative adversarial networks

## 1 Introduction

3D reconstruction is the process of estimating 3D geometry of an object using single or multiple views of the object. The goal of image-based reconstruction is to do the same using single or multiple 2D images. While humans are good at solving this problem by leveraging prior knowledge, it is a difficult task for computer vision algorithms. Many approaches use multiple depth images from different viewing angles of an object to estimate the complete 3D structure [1]. Acquiring multiple images of the same object from different views and well calibrated cameras may not be practical in many scenarios [2], especially the applications that may require real-time performance. Also, it is not feasible to scan all surfaces which leads to incomplete 3D shapes with occluded regions and several holes [1]. The paper is organized as follows: Section 2 explores the applications of 3D reconstruction and what makes the problem interesting. Section 3 describes the different representations that are used to represent points in 3 dimensions. Section 4 gives a summary of related work. Section 5 explains our approach and the details of our work. Results are shown in Section 6 and our current work is outlined in Section 7.

## 2 Applications

Reconstructing 3D geometry is helpful in a plethora of applications. It is used in several medical applications for better diagnosis. Reconstruction of 3D volume and surface models of the tissues provides many advantages to doctors. [3] 3D reconstruction provides detailed surface estimates which can help to guide a surgeon in performing a medical procedure. 3D reconstruction is used to generate 3D structure of pavement for pavement management systems. [4] It is also used in generation of virtual environments. Virtual representations of real world areas are employed in applications such as urban planning, personnel training, virtual tourism and geographical information systems. [5] Image based 3D reconstruction is a building block to applications such as robotics, graphics, augmented reality and entertainment. [2] In the field of robotics, it helps with robot grasping, autonomous navigation and obstacle avoidance [1]. 3D human body reconstruction finds use in gaming, visual effects and free viewpoint videos. [2] Another application where reconstruction is useful is scene parsing [2].

## 3 Different 3D Representations

Most of the existing work on learning-based 3D reconstruction can be broadly categorized by the output representation they produce as either voxel-based, point-based or mesh-based[7]. In this section, we will first describe voxel, point and mesh representations and then give an overview of other representations. Finite element models are divisions of solid objects into polyhedral chunks. This can be used for detailed engineering simulation to model internal forces within objects such as heat, pressure and fluid flow [9]. Diving a finite element into cubes gives rise to the representation which is known as voxel. It is commonly used for simulating objects where the underlying data is captured on a regular grid [9]. Fig. 1 shows what a voxel representation looks like. Fig. 2 gives a real world example of the game Minecraft, where voxels are used to represent all objects and characters.

Another representation of 3D geometry is given by 3D point clouds. Point clouds are increasingly used for surface reconstruction [6]. Fig. 3 shows point cloud representation. A point cloud is a simple and uniform structure. It is easier to learn as it does not have to encode multiple primitives or combinatorial connectivity patterns [11]. It is easier to manipulate in terms of transformations and deformations, as connectivity is not required to be updated [11].

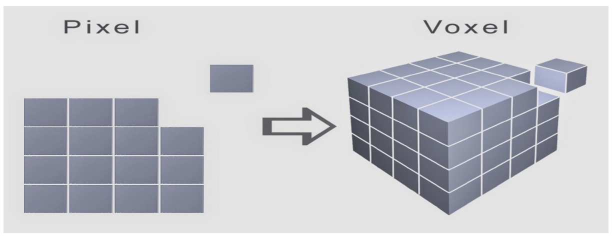


Figure 1: Voxel Representation.[10]

Figure 2: Minecraft models the world in voxels.[8][9]

A mesh is a collection of vertices that define a three dimensional object [12]. Each vertex is defined by 3 coordinates x,y and z which represent the location of the vertex in 3D space [14]. An edge is the connection between two vertices. A face is described by three or more edges in a closed loop. Fig. 4 shows a mesh representation in which triangles (three consecutive edges in a loop) are used as faces. The more faces in a model, the more details it holds. [14] Mesh representations have been successfully used in computer graphics for efficient modeling and rendering of 3D objects [13].

The three representations discussed till now - voxel, point and mesh - have their own set of problems. Using voxel grids, it is difficult to get high resolution reconstructions. This is due to memory requirements and slow training. [7] Recent works have proposed to reconstruct 3D objects in multi resolution fashion [15]. These approaches are still limited to comparatively small voxel grids and are often complicated to implement [7]. Point representations require post processing steps, which are often complex, to generate the final 3D object. Most of the approaches which use mesh representations are prone to generating self-intersecting meshes [7]. With the rise of deep neural networks, learning based approaches for 3D reconstruction are gaining popularity. Learning based 3D reconstruction approaches are able to encode rich prior information about the space of 3D shapes [7]. Recent work uses techniques that are being used to infer high resolution volumetric grids while considering the computational and memory requirements. [2]

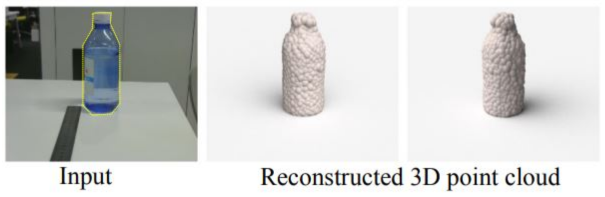


Figure 3: Point Cloud Representation [11]

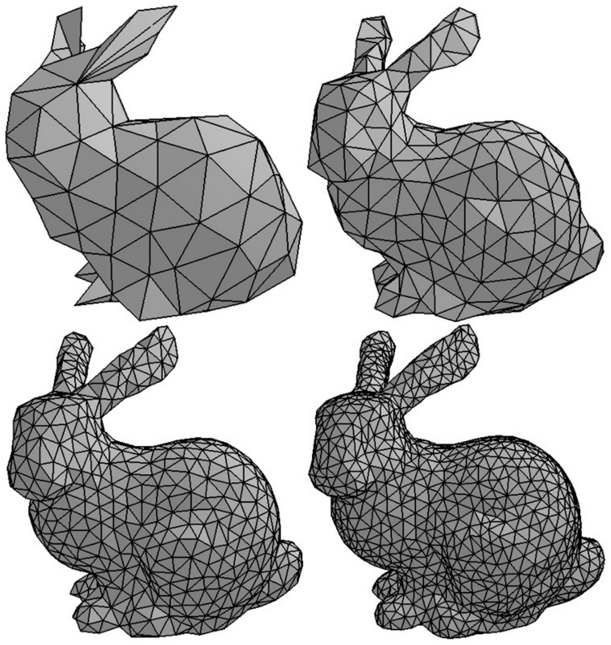


Figure 4: Mesh Representation [14]

These techniques are based on the following - a) space partitioning, b) shape partitioning, c) subspace parameterization and d) coarse-to-fine refinement strategies [2].

## 4 Related Work

For 3D reconstruction, there are many different approaches in literature - Shape completion, multiple RGB/depth images reconstruction, single RGB image reconstruction, single depth view reconstruction and deep generative frameworks. [1] Image based 3D reconstruction includes methods such as a) manual interactive operation method, b) stereo vision method, c) motion image method and d) photometric stereo method.[16] Reconstruction of 3D surfaces from RGB images is mainly implemented by stereo correspondences from multi view geometry. [17] However, stereo based approaches have several restrictions. Hence, in recent years, there is a rise in the use of learning based approaches which allow using a single image as well.

In Single Depth View Reconstruction, the task was to reconstruct the occluded 3D images behind the visible parts. Datasets were used from Shapenet database[18] which were among the early works of deep neural nets to estimate 3D shapes from single depth view. The neural network proposed by Varley et. al. was responsible for facilitating robotic grasping of the input occluded images to infer 3D image outputs[19]. Even VConvDAE[20] was used for shape completion whose initial objective was shape denoising. But these approaches were generally used for generating low resolution voxel grids. Recent works[21] [22] [23] have been done to produce high resolution 3D output. There was a recent work on 3D-PRNN[24] that predicts simple basic shapes using RNN models but they were not able to capture detailed geometric attributes.

## 5 Our Approach

Our aim was to create a pipeline to perform 3D reconstruction using a single depth view and then improve the resolution of the output obtained from the first part. To implement the former, we replicated and then modified the 3D-RecGAN++ architecture as described in [1] . To implement the latter, we replicated and then modified occupancy networks representation as described in [7]. In this section, we first explain the 3D-RecGAN++ architecture, our modifications, dataset preprocessing and training. We then describe the Flask Web application we designed to visualize our results. After that, we describe occupancy networks and its architecture.

### 5.1 3DRecGan++

Our first objective in the project was to train the baseline model and achieve results close to the reported figures in the paper. We experimented with the baseline code and modified the architecture and hyper parameters. We added an extra 3D convolution layer and an up-convolution layer at the encoder and decoder side respectively. This made the generator part to consist of 6 3D convolution and 7 up convolution layers resulting in $256^3$ resolution voxel output. The layers use Tensorflow 1.5 Xavier initializer (similar to Tensorflow 2.0 GlorotUniform initializer) and ReLU activations. We also added an extra 3D convolution layer at the discriminator side to keep up with the change at the generator side.

#### 5.1.1 3D-RecGAN++ Architecture [1]

This section describes the architecture in the baseline implementation [1] Our modifications to the architecture are described in the previous section. 3D-RecGAN has two main networks: generator network as shown in Fig. 5 and discriminator network as shown in Fig. 6. The generator consists of a skip-connected encoder decoder and an upsampling module. The encoder consists of 5 3D convolution layers, followed by leaky ReLU activation function and a max pooling layer. 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 convolution layers which are followed by ReLU activations. The discriminator aims to distinguish whether the estimated 3D shapes are plausible or not. It consists of six 3D convolution layers, followed by ReLU activation function except for the last layer which is followed by a sigmoid activation function.

#### 5.1.2 Dataset and Preprocessing

We have used the preprocessed dataset from ShapeNet database which consists of the voxelized versions of 2.5D images as samples and completed 3D versions as corresponding ground truth. This dataset is available as NPZ files which is fed to the model. The preprocessing steps include taking the raw images from ShapeNet database from four different categories - table, couch, chair and bench. The raw images were first converted to the RGB-D format which is essentially a 2.5D image having a single depth view and then voxelized. The resulting image is saved in numpy zip format as NPZ file. The original model was trained on a total 26,500 NPZ files, but we trained the baseline model on around 2200 NPZ files due to time and resource constraint.

#### 5.1.3 3D-RecGAN++ Training

For training, Adam Optimizer is used for both the generator and discriminator networks with training batch size as 4. We experimented with some of the hyper parameters of the model. We used generator learning rate as $1e^{-5}$ and discriminator learning rate as $5e^{-6}$. We trained the model on Google Cloud Platform using 16 core CPU's, NVIDIA Tesla K80 GPU with tensorflow version as 1.5. With total 15 epochs and around 2200 NPZ files, the training took around 36 hours. We couldn't try lot of experimentation with the hyper parameters as training was taking too long and limited GCP free credits.

### 5.2 Flask Web Application

Using the implementation above, we completed a working proof-of-concept of the proposed design. For the application we chose the Flask framework (as shown in Fig. 7) , which uses python for creating backend server applications. We created a simple frontend page which basically asks for the input file and the application calls the corresponding methods to do the prediction for the input image. During the prediction, we feed in the 2.5D voxelized image as input which was available from the collection of real Shapenet dataset, as well as the 3D ground truth image of the same. The 3D predicted output image is obtained which we can see at the results section below. You can also check demo video here. Our next ongoing task was to add the trained model of the Occupancy Networks as the next layer in the framework. We can then feed the $256^3$ voxelized output from the 3DRecGan++ baseline model as input to this second model and obtain a higher $512^3$ resolution output.

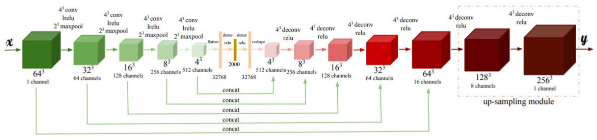


Figure 5: Generator for 3D estimation.[7]

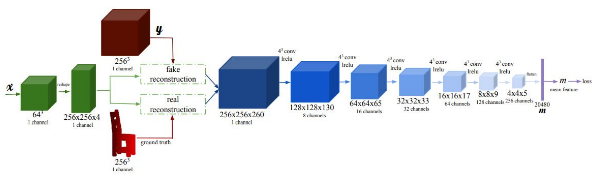
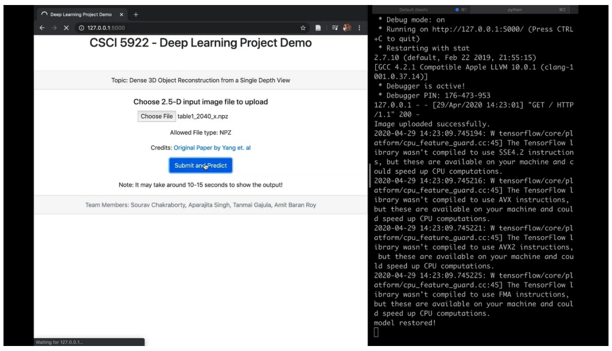


Figure 6: Discriminator for 3D shape refinement. [7]



CSCI 5922 - Deep Learning Project Demo

Topic: Dense 3D Object Reconstruction from a Single Depth View

Choose 2.5-D input image file to upload

table1_2040_x.npz

Allowed File type: NPZ

Credits: Original Paper by Yang et. al

Note: It may take around 10-15 seconds to show the output!

Team Members: Sourav Chakraborty, Aparajita Singh, Tanmai Gajula, Amit Baran Roy

```
 * Debug mode: on
 * Running on http://127.0.0.1:5000/ (Press CTRL
+C to quit)
 * Restarting with stat
2.7.10 (default, Feb 22 2019, 21:55:15)
[GCC 4.2.1 Compatible Apple LLVM 10.0.1 (clang-1
001.0.37.14)]
 * Debugger is active!
 * Debugger PIN: 176-473-953
127.0.0.1 - - [29/Apr/2020 14:23:01] "GET / HTTP
/1.1" 200 -
Image uploaded successfully.
2020-04-29 14:23:09.745194: W tensorflow/core/pl
atform/cpu_feature_guard.cc:45] The TensorFlow l
ibrary wasn't compiled to use SSE4.2 instruction
s, but these are available on your machine and c
ould speed up CPU computations.
2020-04-29 14:23:09.745216: W tensorflow/core/pl
atform/cpu_feature_guard.cc:45] The TensorFlow l
ibrary wasn't compiled to use AVX instructions,
but these are available on your machine and coul
d speed up CPU computations.
2020-04-29 14:23:09.745221: W tensorflow/core/pl
atform/cpu_feature_guard.cc:45] The TensorFlow l
ibrary wasn't compiled to use AVX2 instructions,
 but these are available on your machine and cou
ld speed up CPU computations.
2020-04-29 14:23:09.745225: W tensorflow/core/pl
atform/cpu_feature_guard.cc:45] The TensorFlow l
ibrary wasn't compiled to use FMA instructions,
but these are available on your machine and coul
d speed up CPU computations.
model restored!
```

Figure 7: Flask Application

### 5.3 Occupancy Networks

Occupancy networks is an approach based on space partitioning[2]. Occupancy networks is an approach to representation for learning based 3D reconstruction methods. Occupancy networks implicitly represent the 3D surface as the continuous decision function of a deep neural network classifier.[7] Each location is assigned an occupancy probability between 0 and 1.

#### 5.3.1 Occupancy Network Architecture[7]

In the occupancy network architecture, there are five ResNet blocks and conditional batch normalization is used. For single view 3D reconstruction, ResNet18 architecture is used. For voxelized input, 3D convolution layers are used. We made a few changes to the base code to enable it to take $256^3$ resolution input and give $512^3$ resolution or higher as output. These changes have been documented in the code.

## 6 Evaluation and Results

We trained the 3DRecGan++ baseline model and evaluated the model using the Flask Web application. We got voxelized 3D outputs as shown in Fig. 8, Fig. 9 and Fig. 10.

We got results from occupancy network implementation as shown in Fig.11. We used MATLAB to render .OFF files generated by the code.

## 7 Conclusion

We have implemented 3D-RecGAN++ with our modifications to the architecture. We initially trained the baseline model with less data and number of epochs. This is because taking only 15% of the available training data resulted in training time of over 36hours and GCP credits of around 200$. That didn't give a good output. So we added

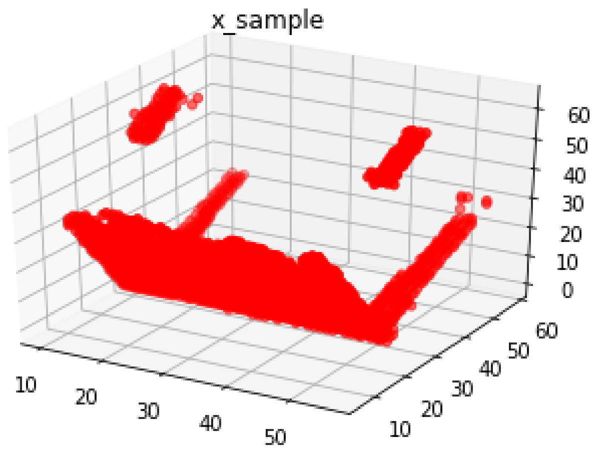


Figure 8: The 2.5D voxelized input image which is fed as input to the model

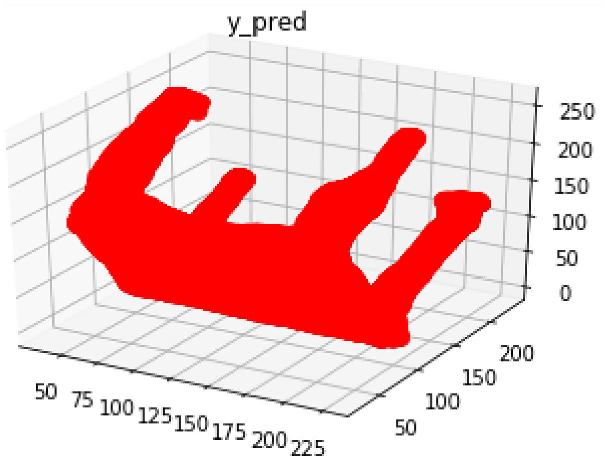


Figure 9: The 3D voxelized predicted image generated by the model

additional convolution layers at the generator and discriminator side as well as tuned the hyperparameters as discussed in above sections to generate a comparable output as generated by the original paper. Next we have also implemented the original Occupancy Networks code taking input voxel resolution as $32^3$. Our aim is to integrate the two models together in our Flask framework so that we can achieve higher resolution of the output obtained from the modified 3D-RecGAN++ architecture. Currently, we are training the Occupancy Networks code by changing the architecture to take $256^3$ resolution voxel as input and give $512^3$ or higher resolution occupancy grid as output. The code can be found in the drive folder or in the Github(not all data files are here).

## References

[1] B. Yang, S. Rosa, A. Markham, N. Trigoni and H. Wen *Dense 3D Object Reconstruction from a Single Depth View* , IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 41, no. 12, pp. 2820-2834, 1 Dec. 2019, doi: 10.1109/TPAMI.2018.2868195.

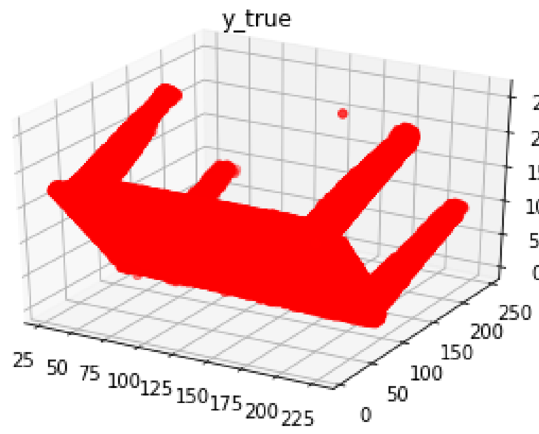


Figure 10: The true 3D voxelized image which is used for ground truth reference

[2] X. Han, H. Laga and M. Bennamoun *Image-based 3D Object Reconstruction: State-of-the-Art and Trends in the Deep Learning Era*, IEEE Transactions on Pattern Analysis and Machine Intelligence, doi: 10.1109/TPAMI.2019.2954885.

[3] H. Wang *Three-Dimensional Medical CT Image Reconstruction*, 2009 International Conference on Measuring Technology and Mechatronics Automation, Zhangjiajie, Hunan, 2009, pp. 548-551, doi: 10.1109/ICMTMA.2009.10

[4] Mahmoudzadeh, A.; Yeganeh, S.F.; Arezoumand, S.; Golroo, A. *3D Pavement Surface Reconstruction Using An RGB-D Sensor*, Proceedings 2020, 42, 47.

[5] Poullis, Charalambos and You, Suya. *3D Reconstruction of Urban Areas*, 33-40. 10.1109 3DIMPVT.2011.14.

[6] Matthew Berger, Andrea Tagliasacchi, Lee Seversky, Pierre Alliez, Gael Guennebaud, et al *A Survey of Surface Reconstruction from Point Clouds*,Computer Graphics Forum, Wiley, 2016, pp.27. ff10.1111/cgf.12802ff. ffhal-01348404v2

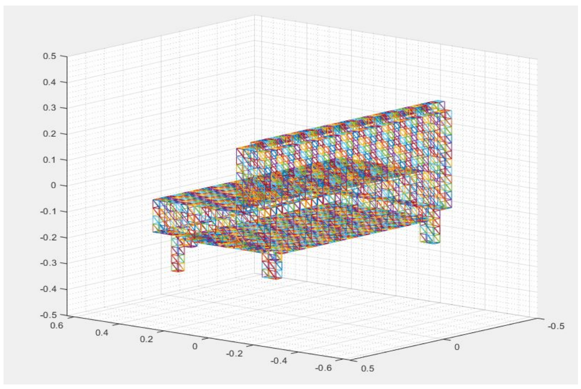
[7] Mescheder, Lars and Oechsle, Michael and Niemeyer, Michael and Nowozin, Sebastian and Geiger, Andreas *Occupancy Networks: Learning 3D Reconstruction in Function Space*, 2019, Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)

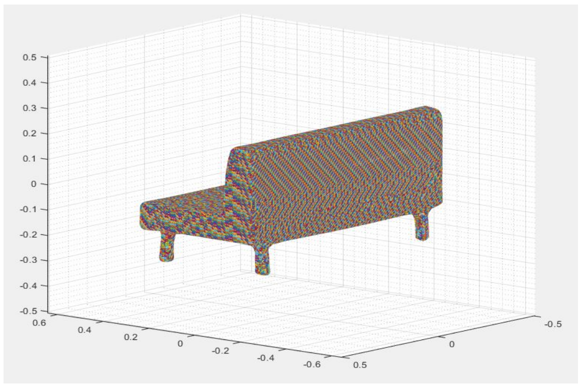
[8] *https://minecraft.gamepedia.com/Minecraft_Dungeons:Valorie*

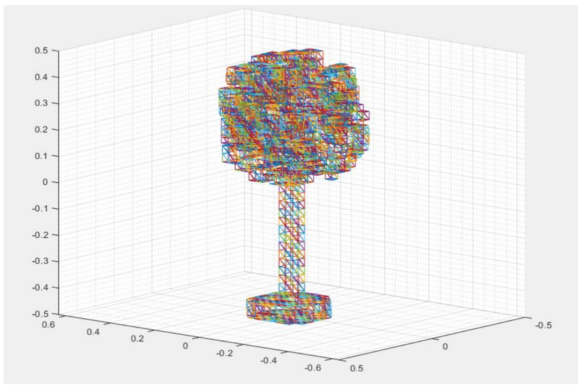
[9] Hughes, John F. *Computer Graphics: Principles and Practice*,2014. Print.

[10] *https://blog.usejournal.com/3d-learning-its-time-to-teach-in-voxels-instead-of-pixels-d644df659328*

[11] Haoqiang Fan and Hao Su and Leonidas Guibas *A Point Set Generation Network for 3D Object Reconstruction from a Single Image*, 2016, 1612.00603


(a) Rendered 2.5D Input


(b) Rendered 3D output


(c) Rendered 2.5D Input

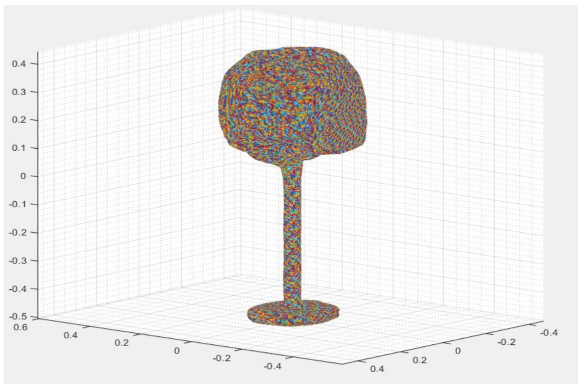

(d) Rendered 3D output

Figure 11: Results obtained from Occupancy Networks

[12] Blender 3D: Noob to Pro *https://upload.wikimedia.org/wikipedia/commons /b/b4/BlenderDocumentation4.pdf*

[13] *https://www.sciencedirect.com/topics/computer-science/mesh-representation*

[14] *https://cathyatseneca.gitbooks.io/3d-modelling-for-programmers/content/3ds_max _basics/3d_representation.html*

[15] Maxim Tatarchenko and Alexey Dosovitskiy and Thomas Brox *Octree Generating Networks: Efficient Convolutional Architectures for High-resolution 3D Outputs*, 2017, 1703.09438

[16] Xinfang Fu and Yueqiang Li *A Survey of Image-based 3D Reconstruction* 2017 International Conference on Mechanical, Electronic, Control and Automation Engineering (MECAE 2017)

[17] Junyi Pan and Xiaoguang Han and Weikai Chen and Jiapeng Tang and Kui Jia *Deep Mesh Reconstruction from Single RGB Images via Topology Modification Networks* 2019, 1909.00321

[18] 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

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

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

[21] A. Dai, C. R. Qi, and M. Nießner *Shape Completion using 3DEncoder-Predictor CNNs and Shape Synthesis* CVPR, 2017

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

[23] 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

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