Convolutional Occupancy Networks

Petropoulakis Panagiotis

Department of Informatics - Technische Universität München

Abstract

In the present report, the work of Convolutional Occupancy Networks from S. Peng et al. [14] will be discussed. The paper was published in 2020, at the ECCV conference, and is about a novel idea on how to utilize the Neural Implicit Representation [10] approach to reconstruct the shape of 3D objects using a single source of input, such as images, or point clouds. It belongs in the field of Learning-based 3D Reconstruction where Neural Networks learn the reconstruction of the objects instead of using traditional 3D reconstruction pipelines that require multiple images as input to solve this problem.

1 Introduction

Reconstructing the shape of objects given sparse inputs, like point clouds from RGB-D scans, or low-resolution voxels, is of great importance to many fields. For instance, in Robotics, creating maps of the real and dynamic world is one of the key components for developing autonomous systems that are able to navigate in human environments.

In Learning-based 3D reconstruction, many variants have been proposed. The main ingredient that classifies the methods into different categories is how the output of the Neural Network is defined. Prior works formulated the output by discretizing the 3D space. Voxel representations divide the space into a grid. It is a well-studied area [3, 15], as traditional Convolutional Networks are extended with 3D convolutions. Nevertheless, one big disadvantage of this approach is the cubic memory requirements for saving such voxel structures, which is certainly inefficient when having sparse data. Another Learning-based 3D reconstruction approach is to define the output of the network as a fixed number of points and regress their 3D coordinates. These types of methods produce point clouds, but again, this representation has some flaws. By defining the shape of the object into a point cloud [5, 9], the topological relationships are lost, and as discussed, the number of the predicted points is defined beforehand. A third direction to take is to regress the input into vertices and faces of a mesh [6, 7]. Many works require having a template mesh in advance, and then, they deform its shape with the corresponding prediction of the network. Another important note while using this pipeline is that oftentimes, the resulting reconstruction has many self-intersecting faces and it is characterized as a non-watertight reconstruction because holes, or missing parts, are present in the output.

Occupancy networks [10] (along with concurrent works) was the first publication that tried to give an answer to the mentioned issues. The authors instead of discretising the 3D space into a predefined structure, they defined the shape of the

objects in an implicit manner. A non-linear classifier needed to be learned and the underlying goal was to classify whether a point belongs to the surface or not. In the figure 1 below, an overview of the method is depicted.

To be more explicit, an input condition x, such as a point cloud or a mesh, is preprocessed with an encoder, and a global feature vector $\psi(x)$ is then extracted. Following, a 3D point p (or query point) is fed with the global feature vector to a ResNet type of network, and the occupancy probability of the 3D point is finally predicted.

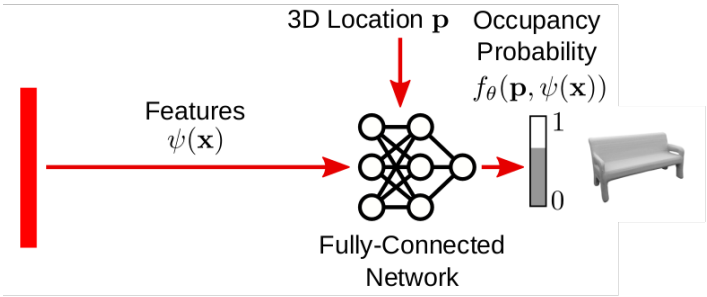

Figure 1: Occupancy Network [10]

To train the Occupancy Fully-Connected Network, a binary-cross entropy loss must be optimized between the predicted and the ground truth occupancy probabilities of sampled uniformed points from point clouds, or meshes, from the training set, as shown in the mathematical formula below. The reader can also be referred to the original paper and study how the authors reconstruct the final 3D shape of the object using the occupancy probabilities of the 3D points with the proposed multi-resolution isosurface extraction algorithm.

$$\mathcal{L}(\hat{o}_p, o_p) = -[o_p \cdot log(\hat{o}_p) + (1 - o_p) \cdot log(1 - \hat{o}_p)] \tag{1}$$

While Occupancy Networks [10] showed promising results, they still lack in some aspects to be considered a superior solution to the methods that discretize the 3D space. This approach mainly works well with simple objects and geometries due to the fact that for each different 3D point p, the same global feature vector $\psi(x)$ is being used. As a result, the reconstructions do not retain the local details of the objects and are overly smooth. In addition, the nature of the Occupancy Network module, fully-connected, does not induce any translation equivariance inductive bias into the model. When the input condition is translated, the output is not consistent. An aspect really important to have in real-life applications.

The contributors of Convolutional Occupancy Networks [14] examined the missing aspects of the previous work, Occupancy Networks [10], in the 3D Reconstruction task. They revisited, primarily, the feature extraction step by generating local feature vectors that depend on the input condition x as well as on the 3D point p to be queried instead of always using the same global feature vector that depends only on the input condition, and added translation equivariance inductive bias into the model with an extra module, as it will be discussed in the next chapter.

2 Method description

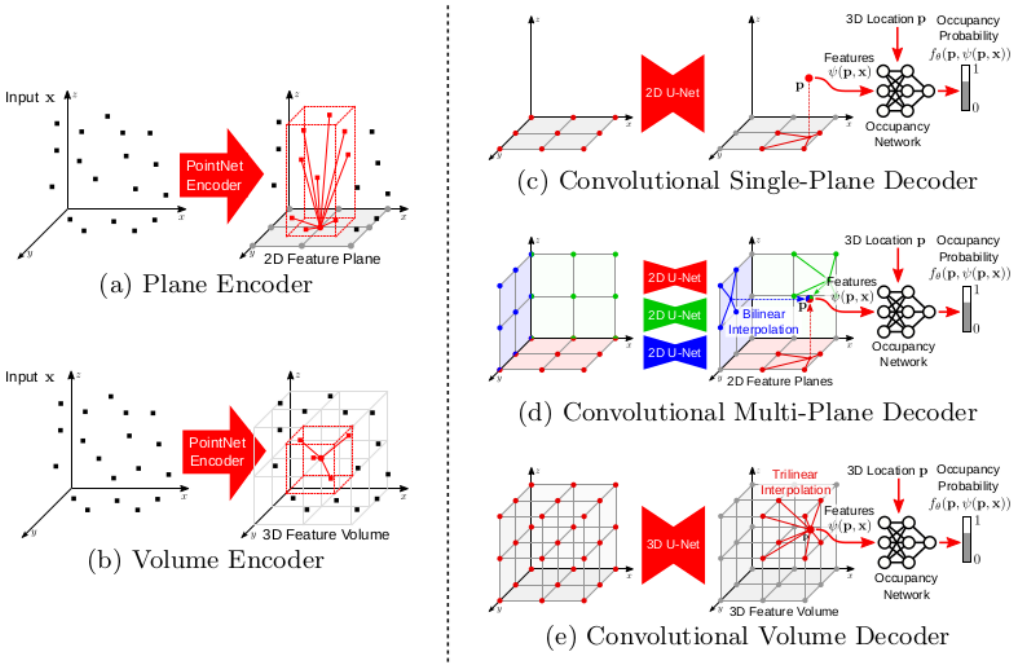

Figure 2: Convolutional Occupancy Networks [14]

Convolutional Occupancy Networks [14] is a multi-stage approach and entails several variants for different use cases. Starting with the 2D variant, a and c subfigures in figure 2, the model takes as an input a condition x, here a sparse point cloud, and preprocess it with a PointNet encoder. This encoder maps each input 3D point to a feature point. The PointNet encoder is not identical to the work [13] where a transformation matrix needed to be learned to map the input point clouds to a global frame. Instead, in Convolutional Occupancy Networks [14], after the 3D points have been preprocessed into features, these features are then projected onto a canonical plane. The projection is done per column, meaning that the features falling into the same pixel are being aggregated with the max operation.

To provide an interpretation, using the PointNet encoder and aggregation steps, local information is being added to the input 3D points from their neighbors so that the model is able to learn the local details of the objects. As for the projection of the features onto the canonical plane, the model becomes invariant to the input frame of the condition x, and the features, henceforth, reside on a global uniform (feature) frame.

To add translation equivariance inductive bias to their method and global information to the feature points, the authors added a 2D U-Net [12] decoder to process even further the feature space of the canonical plane. The 2D U-Net has an equal or larger receptive field than the size of the canonical plane to capture the whole space using an input and output feature dimension of 128 and 5 convolutional layers.

Once the new feature space with the global and local information of the shape (incorporated into the features) has been built, a 3D point p can be queried for its occupancy probability. The 3D point p (c subfigure in figure 2) is first projected to

the feature space of the decoder and its corresponding feature value, in this space, is calculated using bi-linear interpolation. This modification, above all, played a major role in the results, as we would see in comparison to the previous work [10] where the feature vector $\psi(x)$, was only dependent on the input condition. In this work, though, the Occupancy Network module takes as an input the feature vector $\psi(x, p)$, and the 3D point p (of this vector), and generates the occupancy probability. As explained in section 1, during training, a binary-cross entropy loss is finally being optimized using uniform sampled points from the training set.

Of course, instead of using only one canonical plane (a subfigure), all of the three canonical planes (d subfigure) can be in use to produce even stronger features. This can be done by projecting the features from the PointNet encoder to all the different canonical planes and by stacking three 2D U-Nets with shared weights in the decoding step. Lastly, to project the 3D point p and generate its corresponding feature vector $\psi(x, p)$, the point is projected first onto the three canonical feature spaces of the decoder, and then, the resulting features are summed up.

The last proposed variant, in this work, is the 3D model (b and e subfigures), as canonical planes form a 2D space and might not be always optimal to encapsulate the information of the object into 2D structures. After processing the input point cloud x into feature points with the Encoder, these features are projected to the closest center of the cell of a 3D volume that they belong to, and again, if many features end up in the same cell, they are aggregated with the max operation. The 2D U-Net is also now replaced by a 3D U-Net with 3D convolutions, and finally, to project the 3D point p into the feature space of the canonical volume of the decoder, trilinear interpolation is being employed for finding the feature vector $\psi(x, p)$. As we would see in the experiments section, this modification can be advantageous even at the expense of having higher memory consumption.

3 Experiments and results

The authors evaluated their model into four different datasets, two synthetic and two real-world:

1. ShapeNet [2] is a synthetic dataset that contains single objects and was used to measure the performance on the Object-Level Reconstruction and Super-Resolution tasks, where in the latter, the output resolution is being set higher than the input.

2. The authors also built a custom Synthetic Dataset by sampling objects from the ShapeNet [2] and adding random walls to create different room layouts to test how well their model could be scaled in bigger scenes.

3. ScanNet v2 [4] is a real dataset of RGB-D scans of single rooms and was selected to test the robustness of the model to noisy inputs by extracting point clouds from the provided meshes of the set.

4. Matterport3D [1] is the most challenging dataset, and it contains big buildings with numerous rooms.

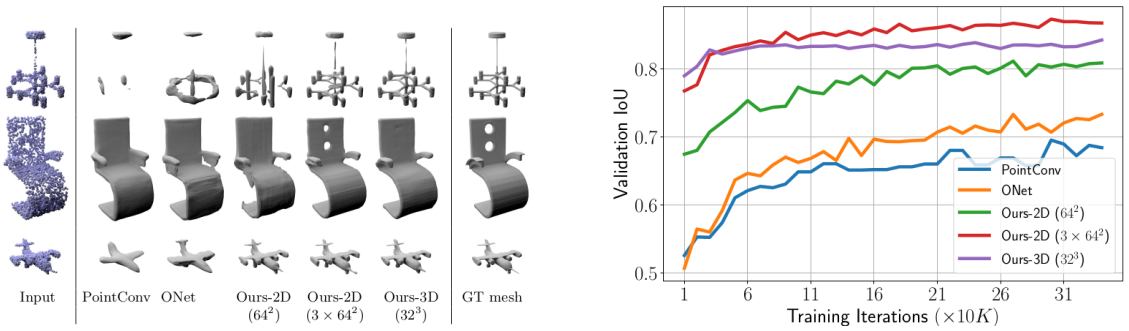

Figure 3: Object-Level 3D Reconstruction [14]

In the Object-Level 3D Reconstruction from noisy (Gaussian noise added to the synthetic dataset) point clouds task, Convolutional Occupancy Networks [14] outperformed the previous works as shown in the left picture in figure 3. Occupancy Networks [10] struggled to reconstruct complex shapes, such as lamps, and the reconstructions were overly smooth. The authors also compared their model with a PointConv network. This variant does not contain canonical planes or volumes neither U-Net decoders. The encoder is replaced by a PointNet++ [11] network that learns a transformation matrix to project the input condition x to a global frame, and for finding the corresponding features of the 3D points p, the interpolation step is performed by weighted Gaussian regression. Clearly, projecting into canonical planes and volumes and adding global information into the features using U-Nets produces considerably better results. In addition, the best variant for this task was the three canonical planes while using less GPU memory from the second best variant, the volumetric representation, and lastly, the single canonical plane variant came third. In the right picture in figure 3, the progression of the Volumetric IoU along with the training iterations is also shown. Convolutional Occupancy Networks [14] not only achieved the best performance overall but in less time too. The Occupancy Network is considerably smaller than in the previou work.

In the same dataset, Convolutional Occupancy Networks [14] were able to reconstruct finer outputs from low-resolution input voxels on the Super-Resolution task. The three canonical planes performed similarly to the volumetric representation while consuming 37% less memory. However, using a single canonical plane, the task could not be solved due to the potential unresolved ambiguity nature of projecting coarse and regular structured voxel inputs to a canonical plane, as hypothesised by the authors.

In bigger synthetic scenes originating from the custom-built dataset, most of the previous methods can not scale well. The reconstructions lose the local details, and the prominent method Screened Poisson Surface Reconstruction (SPSR) [8] produces noisy results while also requiring the normals of the points to be calculated to generate the reconstructions. An additional step that is not needed in Convolutional Occupancy Networks [14] and Implicit Representation Learning.

In fact, in Implicit Representation Learning approaches, the normals can also be computed (if needed later) easily by taking the gradient of the output (occupancy probability) with respect to the input (Neural Networks in essence approximate a

function and the gradient of this function can be found as well). On the other hand, classical approaches need to rely on the neighbors of the points and calculate the normals using Principal Component Analysis (PCA).

The authors also tested the generalization ability of their model in real scans while training their model only on synthetic data. In ScanNet v2 [4], again the winning method was Convolutional Occupancy Networks [14], while the other learning-based methods almost failed. Even though the SPSR [8] method produced quite well reconstructions, the trimming parameter must be fine-tuned appropriately beforehand. In this task also, the volumetric representation performed the best and the authors explain that this representation can handle better the domain shift from synthetic to real-world scans.

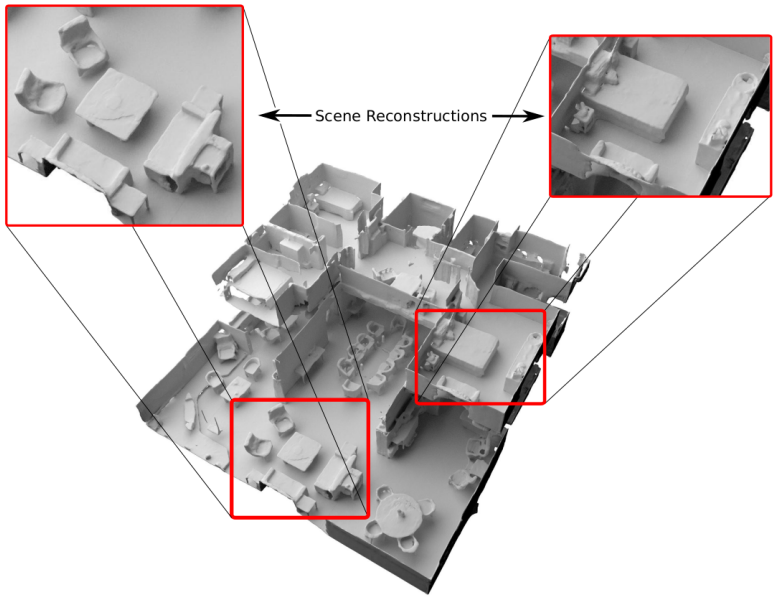

Figure 4: Reconstruction on Matterport3D [14, 1]

The most interesting results of this work were captured on the Matterport3D [1] dataset (figure 4). The method is quite flexible and can be adapted into reconstructing whole buildings. This can be done by training the model on synthetic crops, and during inference, in a sliding window manner, small overlapping patches (i.e. areas) of the scene, equal to the size of the receptive field of the trained network, are processed. Then, the reconstructions are aggregated into the whole scene. This could not be done, of course, without the U-Net decoders and their convolutional nature that can handle translated inputs. In short, Convolutional Occupancy Networks [14] could surprisingly scale to novel room layouts and objects. As a final note on this task, the authors do not mention explicitly how they aggregate the reconstruction of the patches to the whole scene and how they merge the overlapping parts.

4 Discussion / Conclusion

Convolutional Occupancy Networks [14] showed state-of-the-art results in many tasks, such as Object-Level Reconstruction and Super-Resolution, and retained good

performance in real-world noisy scans. The reconstructions preserved the local details of the objects while previous works failed to address this aspect when it comes to reconstructing complex shapes and geometries. The reconstructions also are not characterized by overly smooth surfaces, and the method does not require any additional information of the input, like the points normals as in SPSR [8], to learn the reconstructions. The training time is even shorter than the previous Learning-based approaches.

To sum up the main ideas and components of the method: 1) PointNet encoders preprocess and incorporate local information into the input condition x, 2) Canonical planes and canonical volumes project the features onto a uniform global frame and the approach becomes independent of the different coordinates frames of the inputs, and lastly, 3) U-Net Decoders add global information and translation equivariance inductive bias into the model.

In this work, however, there is still enough room for improvement. In detail, the proposed method is only translation equivariance with respect to the defined voxel size, it is not rotational equivariance due to the nature of the convolutional operations, and certainly, there is still a present reality gap when transferring the trained model on synthetic dataset into the real-world.

Closing, I would like to add some personal comments. In the method variant of using all of the three different canonical planes, it is not very clear to me why the authors selected U-Net decoders with shared weights instead of training them separately. An additional case study could reveal whether there is any performance difference in this aspect. Another point in this variant is that the 3D point p is projected onto the 3 different canonical planes, and then, the features are aggregated using the sum operation. It seems more natural, in this case, to use a shallow Neural Network to aggregate the features, as some canonical planes might be more relevant in some tasks. Hence, attaching a more appropriate weight could lead to better results. Finally, this work is a multi-stage pipeline and it would be interesting in future works to explore how this can be trained in an end-to-end fashion (if possible at all due to the projection step into the canonical planes/volumes).

References

[1] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017.

[2] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository. 2015.

[3] Christopher B. Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling,

editors, *Computer Vision – ECCV 2016*. Springer International Publishing, 2016.

[4] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas A. Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[5] Haoqiang Fan, Hao Su, and Leonidas Guibas. A point set generation network for 3d object reconstruction from a single image. 2017.

[6] Georgia Gkioxari, Justin Johnson, and Jitendra Malik. Mesh r-cnn. 2019.

[7] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan Russell, and Mathieu Aubry. AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[8] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Screened poisson surface reconstruction. 2006.

[9] Chen-Hsuan Lin, Chen Kong, and Simon Lucey. Learning efficient point cloud generation for dense 3d object reconstruction. *ArXiv*, abs/1706.07036, 2018.

[10] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[11] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017.

[12] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. volume abs/1505.04597, 2015.

[13] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. 2016.

[14] Lars Mescheder Marc Pollefeys Andreas Geiger Songyou Peng, Michael Niemeyer. Convolutional occupancy networks. In *European Conference on Computer Vision (ECCV)*, 2020.

[15] Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T. Freeman, and Joshua B. Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. NIPS'16, Red Hook, NY, USA, 2016.