Dynamic Plane Convolutional Occupancy Network

Stefan Lionar, Daniil Emtsev, Dusan Svilarkovic

Abstract

Learning-based 3D reconstruction using implicit representations [11][13] has shown promising progress, being able to produce expressive representations for 3D reconstruction tasks. More recently, Peng et al. proposed Convolutional Occupancy Networks [13], a flexible translation equivariant architecture which can capture the local structure of point clouds and achieve the state-of-the-art performance. His work, however, uses Manhattan-world assumption [6] where every encoded feature is projected onto the canonical planes. Aiming to increase the rotation invariance of Convolutional Occupancy Networks, we propose to extend it by introducing a plane predictor composed of fully connected networks to learn dynamic planes that directly capture the structures of input point cloud and project the encoded features to those planes. The pipeline of our model, called Dynamic Plane Convolutional Occupancy Networks (DPCO), is illustrated in Fig. 1. We experimentally demonstrate that better rotation invariance can be obtained by increasing the number of planes and applying data augmentation with rotated inputs. However, comparing the performance using three canonical and dynamic planes, we later show that it is hard to justify the use of learning dynamic planes to improve rotation invariance.

1. Introduction

Exploiting 3D information such as point clouds has been increasingly popular for various computer vision tasks, such as autonomous navigation, indoor navigation, self-driving vehicles, and robotics [19][18]. 3D reconstruction from point clouds promise better precision for those applications, as well as in many computer graphics applications. In recent years, learning-based 3D reconstruction using implicit neural representations in continuous domain have gained more attention due to its ability to produce smooth and expressive reconstruction with significant reduction of memory footprint [11] [13][12][3].

More focus has been directed towards methods that not is only able to accurately capture global structure but also local structure. Peng, et al.[13] identified the limitations of the older implicit 3D reconstruction method on capturing local structure and significantly alleviated it by removing the pooling of per-point representation. Instead, the representation of each point is projected onto planes or volume grids and processed using U-Net, a translation equivariant CNN architecture [17][5]. However, a strong prior on the orientation bias of the input exploiting Manhattan-world assumption [6] is incorporated in designing their architecture, and therefore would not work with rotated or tilted inputs.

In this work, we propose an extension of Convolutional Occupancy Networks [13] called *Dynamic Plane Convolutional Occupany Networks (DPCO)* with a main goal to improve its rotation invariance. Taking inspiration from [14], which demonstrates the ability of point cloud encoder network to learn planes that best describe its input distribution, we add plane predictor networks to predict arbitrary planes alongside the main Convolutional Occupancy Networks. We project the encoded per-point feature onto these learned planes, instead of canonical planes. Additionally, we explore another representation by projecting onto sphere surface. The details of experiment with sphere encoder is detailed in the *Supplementary Materials*. Our architectures are illustrated in Fig. 1.

In summary, our contributions are as follows:

- We propose Dynamic Plane Convolutional Network and demonstrate consistent performance improvement by increasing the learned planes from 1, 3, 5 to 7. We show that a better performance than Convolutional Occupancy Networks alone can be achieved by learning more than 3 planes.

- We show that adding more learned planes in our network increases the robustness towards rotated input. It improves further by introducing data augmentation with rotated input during training. However, it is hard to justify the use of dynamic planes to aid the rotation invariance. In our experiment with rotation augmentation, the model with 3 canonical planes outperforms 3 dynamic planes.

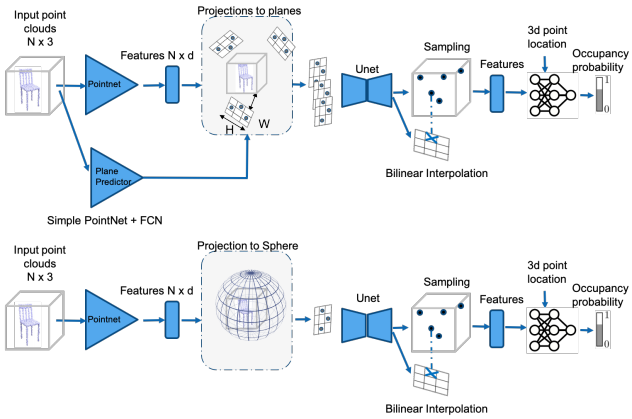

Figure 1: Pipeline of Dynamic Plane Convolutional Occupancy Networks (Top). We also explore a similar pipeline with spherical projection (Bottom). The experiment with spherical projection is detailed in the Supplementary Materials.

2. Related Work

Existing works on learning-based 3D reconstruction can be broadly categorized by the output representation they use as either voxels, points, meshes or implicit representations.

Voxel Representations: Due to their straightforward implementation, voxels are the most widespread methods in early works of learning-based 3D reconstruction. However, even with adaptive data structures, representation of an object is still limited by the resolution of the underlying 3D grid as the complexity grows cubically.

Point Representations: 3D point clouds are widely used either in the robotics or computer graphics. There have been several works on analyzing raw point clouds without changing its nature [15][16]. PointNet [15] is a pioneer of this group which directly takes point sets for 3D classification and segmentation tasks. It uses per-point Multi-Layer Perceptron (MLP) to extract features from individual points and aggregates these features into a global feature vector by a max pooling layer. PointNet++ [16] divides the input cloud into multiple overlapping neighbouring areas and applies PointNet as a feature extractor for each neighbourhood. Treating each point independently limits such models to exploit local geometrical features.

Mesh Representations: Meshes have been used for segmentation and discriminative 3D classification by applying convolutions on the graph spanned by the mesh's vertices and edges [8], [1].

More recently, meshes have also been considered as the output representation for 3D reconstruction. Several works [7][9] used neural network to directly predict vertices and faces of a mesh.

Implicit Representations: Learning continuous implicit representations using neural networks have gained traction recently due to its advantages in representing shape continuously and handling complex topology [11][13][3]. The recent method [13] achieved state-of-the-art results by projecting per-point representations onto canonical planes (two walls and ground). To date, however, there is still no work concerning rotation invariance of implicit representations. In this work, we propose to aggregate features by projecting them onto learned planes based on the topology of input point clouds.

3. Method

Our pipeline is mainly similar to Convolutional Occupancy Networks [13], which includes deep neural network of encoder and decoder with point clouds as its input. We explore a geometrical projection of the encoded features to achieve robustness given input with any orientation by learning dynamic planes that best describe the topology of the input and project the features onto them. Figure 1 presents an illustration of our pipeline.

3.1. Encoder

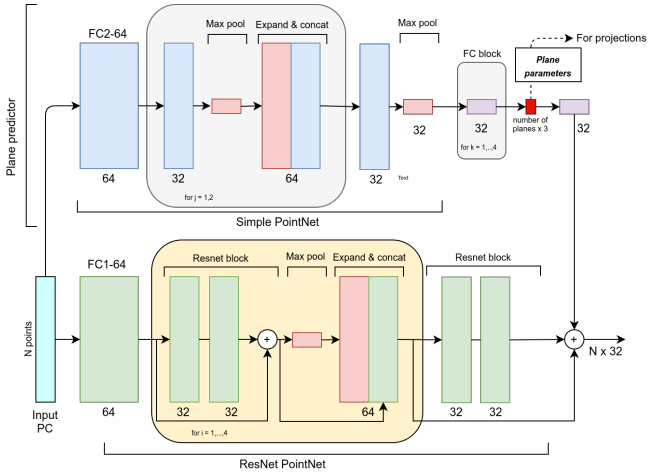

Figure 2: The encoder of Dynamic Plane Convolutional Occupancy Networks. On top of ResNet PointNet [15] used by [13], we add plane predictor, which consists of simple PointNet [15] continued by fully connected networks.

The architecture of our *Dynamic Plane Convolutional Occupancy Networks* encoder is illustrated in Figure 2. On top of ResNet PointNet [15] used by [13], we add plane predictor networks, which consists of simple PointNet [15] and fully connected networks. To allow backpropagation into the plane predictor networks, we sum its output to the last layer of ResNet PointNet.

We project these encoded features onto planes with 64×64 grids and apply max pooling for the features falling on the same grid. Afterwards, we process them using UNet [17] with shared weights between every plane and aggregate the features from each plane by summing them up. The final planar features have a dimension of $64 \times 64 \times 32$, where 32 is the predetermined hidden dimension. The normals of the planes (3 parameters for each plane) are learned through the plane predictor networks. We do not consider learning the intercept of a plane and always assume it as 0 (passing through the origin).

Planar projection: Denoting the three unit basis vectors of canonical axes, $\vec{i}$, $\vec{j}$, $\vec{k}$, where $\vec{k}$ is the basis vector of ground and normal vector of a plane, $\vec{n} = [A, B, C]^T$, where A, B, C are the learned plane parameters, to project features onto the plane and keep them inside the 64×64 grids, we apply the following:

1. **Basis change:** Normalize $\vec{n}$ into a unit vector, $\vec{n}_{normalized}$ and obtain a rotation matrix R that aligns $\vec{k}$ to $\vec{n}_{normalized}$. Using the same rotation matrix R, rotate $\vec{i}$ and $\vec{j}$ to obtain $\vec{i}_{plane}$ and $\vec{j}_{plane}$.

   The vectors $\vec{i}_{plane}$, $\vec{j}_{plane}$ and $\vec{n}_{normalized}$ are orthogonal and serve as the basis of plane coordinate, where $\vec{n}_{normalized}$ is the new basis of ground.

2. Convert the coordinates from canonical axes into plane coordinate and project the features orthographically to the plane ("new ground").

3. **Normalization:** Given a 3D space with a maximum valid range of a and a possible outermost point in the positive octant of this space, $p_{outermost} = [a, a, a]^T$, we want to obtain a normalization constant, c, such that the orthographic projection planar coordinates of $p_{outermost}$ to the "new ground", $P_{outermost} \in \mathbb{R}^2$ after divided by c is always inside the valid range a, but still as close as possible to a. We obtain c by taking the maximum between the norms of $p_{outermost}$ projected to $\vec{i}_{plane}$ and $\vec{j}_{plane}$ converted into positive octant.

4. The planar coordinates of the projected features are divided by c and discretized into 64×64 grids.

3.2. Decoder

The goal of the decoder is to obtain the occupancy prediction of a query point $\mathbf{p} \in \mathbb{R}^3$ given the aggregated feature maps. Similar to how we project features in encoder, we project $\mathbf{p}$ onto either the learned planes or sphere's surface. The feature vector of a query point is obtained from billinear interpolation of the features encoded at the four neighbouring vertices.

Occupancy prediction: Given an input $\mathbf{x}$, we predict the occupancy of $\mathbf{p}$ based on the feature vector at point $\mathbf{p}$, denoted as $\psi(\mathbf{p}, \mathbf{x})$:

$$f_\theta(\mathbf{p}, \psi(\mathbf{p}, x)) \rightarrow [0, 1] \tag{1}$$

We use the same network as [12], which consists of multiple ResNet blocks, adding ψ to the input features of every block.

3.3. Training and Inference

During training, we apply binary cross-entropy loss between the occupancy prediction, $f_\theta(\mathbf{p}, \psi(\mathbf{p}, x))$ and the true occupancy value of $\mathbf{p}$ using Adam optimizer [10] with a learning rate of 10^{-4}. During inference, we apply *Multiresolution IsoSurface Extraction* (MISE) [11] to construct meshes from occupancy predictions given an input $\mathbf{x}$.

4. Experiments

We conducted object-level reconstruction from point clouds using ShapeNet [2] subset of Choy et al. [4] as our dataset. Following [13], we subsample 3000 points from the surface of ShapeNet objects, feed them to our models to get the encoded representations, and query the occupancy probability of 2048 uniformly sampled points during training.

First, we evaluate the reconstruction of our models without any data augmentation. Second, we introduce data augmentation by applying random rotation in the input. For both experiments, we assess the learned plane normals from our models and their rotation-invariance.

Metrics: We follow the metrics used by [11] and [13], namely Volumetric Intersection over Union (IoU), Chamfer-L_1 and Normal Consistency.

4.1. Without rotation augmentation

The sample reconstructions of our DPCO models are illustrated in Figure 3. Table 1 summarizes the metrics of our models. The validation score over training steps of our models can be seen in Figure 4. Our implementation achieves lower metrics and slower convergence than [13]. However, refining implementation to have a matching performance with [13] is not our priority, but rather to study the properties of our DPCO models and the impact of increasing the number of learned planes. We demonstrate progressive performance by increasing the number of learned planes, and thus expect that learning more than 3 planes can produce better metrics than [13] with a refined implementation.

Distribution of the learned plane normals: The learned plane normals are evaluated using the 8751 objects in test set. We expected our models to generate varying plane normals among different objects. However, this is

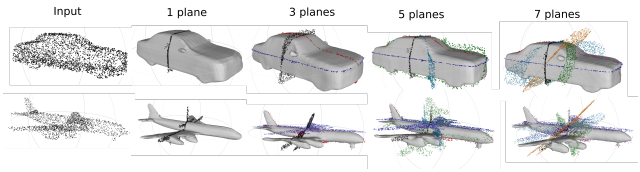

Figure 3: Sample reconstructions of our DPCO models with overlaid projected input point clouds.

Figure 4: Validation score over training steps after running training for 5 days in Leonhard cluster. The difference of accomplished steps is mainly due to intermittent CPU utilization drop.

	IoU	Chamfer-$L1$	Normal C.
ONet [11]	0.778	0.079	0.895
Conv ONet (Peng et al. [13])*			
1 plane	0.780	0.070	0.898
3 planes	0.861	0.048	0.937
Conv ONet (ours)			
1 plane	0.801	0.072	0.900
3 planes (base)	0.851	0.054	0.916
Dynamic Plane			
1 plane	0.777	0.079	0.894
3 planes	0.833	0.061	0.917
5 planes	0.853	0.054	0.927
7 planes	0.859	0.052	0.928

**trained and evaluated on object "chair". The unpublished extension on all objects has better metrics (0.8333 and 0.884 IoU for 1 and 3 planes).*

Table 1: Metrics of our models without any rotation. IoU and Normal Consistency: Higher better. Chamfer-$L1$: Lower better.

not the case for 1 and 3 planes models where all objects have identical planes. In 1 plane DPCO, our model learned the wall of canonical planes. For 3 planes DPCO, we ran experiments 3 times resulting in 3 different combinations of normals. It indicates that the final learned plane normals can be non-identical across runs due to non-determinism and different local minima. The results presented here has 2 canonical planes (wall and ground) and another wall plane with a normal direction of $2.32x + -y = 0$. Our previous run that produced 3 canonical planes—but with a bug causing inability to generate viable mesh—has a higher validation IoU of 0.8450. In our experiments, it is consistent that the further the predicted normals from 3 canonical planes, the lower the validation score.

The distribution of plane normals for 5 and 7 planes DPCO is illustrated in Figure 5. It can be seen that 3 canonical planes always exist in both models. In 5 planes DPCO, the fourth normal has moderate variability and the fifth has slight variability across objects. We find that objects with different global structure favor different regions for the normal with moderate variability, corroborating our hypothesis that dynamic planes based on the topology of the input clouds can be learned. A more detailed observation also indicates that different object classes can be distinguished from their moderate plane distribution. In 7 planes DPCO, on top of the normals of 3 canonical planes, there is an addition of 1 identical diagonal normal, two normals with slight variability across objects and 1 redundant ground plane. We posit canonical planes always are always present due to the fact that all of the ShapeNet objects are aligned that way and our models managed to learn the orientations with the highest variances.

Rotation Invariance: In order to assess the rotation invariance of Dynamic Plane Convolutional Networks and compare them to Peng's Convolutional Networks as a baseline, we use the test set of the ShapeNet dataset and apply the following :

1. Rotate input point clouds in x,y and z axis from randomly sampled values from the discrete degree intervals. That means that for the Θ degree interval, object is given randomly chosen degree values for x,y and z rotations from the $[0, \Theta]$ range. In example case for $\Theta = 45$, an object can be rotated 13 degrees in x-axis, 43 degrees in y-axis and 25 degrees in z-axis.

2. From the rotated input point clouds, we generate the output object in which we expect to preserve rotations from the rotated input.

3. Compare them to the rotated ground truth objects in the same rotation values as input object (e.g. 13° in x-axis, 43° in y-axis and 25° in z-axis).

To fully test rotational invariance, we create separate experiments in the ranges of 15, 30, 45 and 60 degrees which have shown to be substantial to show different trends in reconstruction drop in different models when rotation degree range is increased.

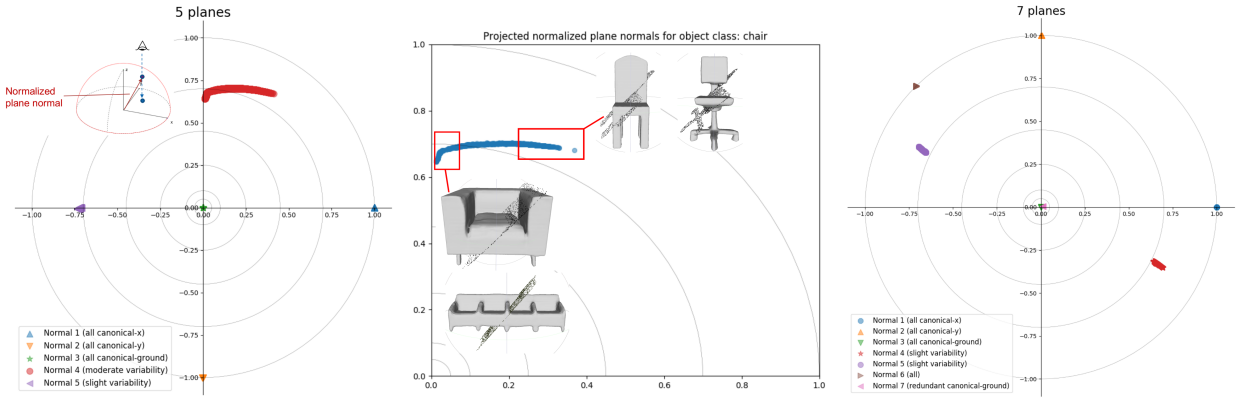

Figure 5: **Figure convention:** In the three figures above, plane normals are converted into upwards direction, normalized into unit length and projected as if viewed from the top. **Left:** Plane normal distribution of 5-plane DPCO. **Middle:** Distribution of plane normal with moderate variability for object chair in 5-plane DPCO. The projected point clouds are overlaid on the meshes for illustration. **Right:** Plane normal distribution of 7-plane DPCO.

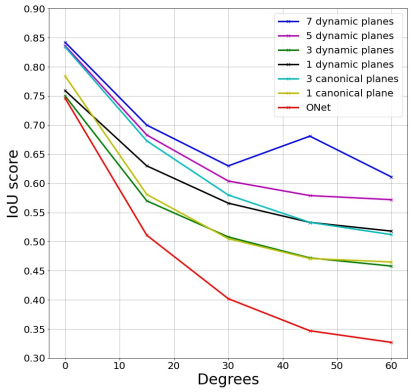

Figure 6: Rotation invariance testing with no rotation augmentation.

In Figure 6, we show the IoU score for different models when rotation degree range is increased. Legend "n dynamic plane" represents our Dynamic Plane Convolutional ONets, while "n canonical planes" represents Peng's Convolutional ONets [13]. As a baseline, we also did rotation invariance for "ONet" which represents Mescheder et al. Occupancy Networks [11]. Note that there is a slight drop for 0 degrees, which comes from the default scaling when trying out

It can be seen that ONet shows no substantial rotation invariance properties, while our dynamic planes and Peng's canonical planes, despite still not indicating the state-of-the-art behavior (in $[0.5, 0.6]$ range of IoU score), they seem to show that rotation preservation is more stable than the baseline ONet. Even with Peng's and our work overlapping in scores, our 7 dynamic and 5 dynamic planes clearly show a slightly better behavior than Peng's 3 plane model when degree ranges are increased. This can be attributed to the property of our plane predictor which gives unique planes per each object, instead of global canonical planes as in Peng's work.

Further, we also examined the plane distribution for our DPCO models when input point clouds are rotated. We observed that there was negligible response to the predicted planes with rotated input.

4.2. With rotation augmentation

To further investigate whether the plane predictor for dynamic planes actually improves the score, we explored the hypothesis that this outcome might be due to the object distribution of the ShapeNet, which was set up such that every object is in default position, with no additional rotations. Therefore, we introduce rotation augmentation during the training, which rotates the objects in the same manner as in the rotation invariance testing, with a degree range of 45°. We use it for training on 3 canonical planes, 3 dynamical planes and 7 dynamical planes.

In Table 2, we present the results of training with rotational augmentation and evaluated against the default positioned objects on ShapeNet dataset, with no rotation nor scaling on the ground truth object nor input object. First row shows our 3 canonical planes result, while the other two are our results given from the work we did. This result seems to give a perception that overall our dynamic planes are not better than 3 canonical planes, but these results are not including rotated ShapeNet objects.

Rotation invariance: testing done on ShapeNet objects, which is the same rotation testing as in previous figure, we present Figure 7, where we see that although 3 canonical

planes model works better than 3 dynamic planes model when the degree range is increased, we can see that 7 dynamic planes shows slightly better IoU score results than 3 canonical planes when rotation degree increases, which means that for rotated objects, more dynamic planes show stronger rotation invariance property than canonical planes.

The reason why 3 dynamic planes model tends to have lower score than 3 canonical planes model with increased degree range (when rotation augmentation is included) can be argued that it comes from not capturing the same information as 3 canonical planes, which on other hand is not the case when compared to 7 dynamic planes where score is better than 3 canonical planes. The distribution of plane normals shown in Figure 5 where we see that 5 and 7 dynamic plane models tend to almost always implicitly predict 3 canonical planes out of their 5 or 7 planes, shows that additional planes besides 3 canonical ones can help in having better pointcloud reconstructions when objects position is not included.

	IoU	Chamfer-$L1$	Normal C.
Conv ONet (ours) 3 planes planes	0.779	0.080	0.888
Dynamic Plane 3 planes	0.760	0.064	0.873
Dynamic Plane 7 planes	0.780	0.078	0.891

Table 2: Metrics of our models with rotation augmentation during training evaluated on the original ShapeNet test set.

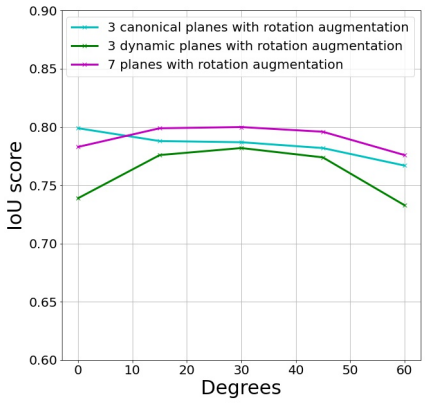

Figure 7: Rotation invariance testing with rotation augmentation.

5. Work Distribution

Stefan Lionar (1), Daniil Emtsev (2), Dusan Svilarkovic(3)

	(1)	(2)	(3)
DPCO implementation	+	+	+
Experiments	+	+	+
Rotation Invariance	+	+	+
Planes Distribution	+	-	-
Exploration of Sphere Encoder/Decoder and height map	-	+	-
Triplet Loss	+	+	-
Report	+	+	+

Table 3: Work Distribution among team-members.

6. Conclusions and Future Work

We extended Convolutional Occupancy Networks by adding plane predictor networks and demonstrate the possibility of learning planes which follow the structure of input point clouds. We show that increasing the number of learned planes can improve rotation invariance, and it can be enhanced further by applying rotation augmentation during training. From the performance comparison between the models with 3 canonical and dynamic planes, however, it is still hard to justify the use of dynamic planes to improve rotation invariance.

Several questions are raised, such as why the dynamic planes do not respond to the rotated input when the model is trained without rotation augmentation, and why there is an observed performance decrease of 1 and 3 dynamic planes models compared to the models with canonical planes. Future experiments in this direction are worthwhile to be explored. Several possibilities include reducing the complexity of the plane predictor networks and reducing the magnitude of interference from the plane predictor network to the last layer of ResNet PointNet networks.

It was also observed that the predicted planes can be sub-optimal, e.g. not being able to always ”recover” the canonical planes in 3-plane DPCO and the existence of redundant planes in 7-plane DPCO. To alleviate this occurrence, it is worthwhile to explore the use of similarity loss to govern the plane normals or refining the plane predictor networks to have a better backpropagation flow. The idea of the similarity loss and the improved plane predictor networks are laid out in the Supplementary Materials.

References

[1] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. Geometric deep learning: going beyond euclidean data. *CoRR*, abs/1611.08097, 2016. 2

[2] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 3

[3] J. Chibane, T. Alldieck, and G. Pons-Moll. Implicit functions in feature space for 3d shape reconstruction and completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6970–6981, 2020. 1, 2

[4] 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. In *European conference on computer vision*, pages 628–644. Springer, 2016. 3

[5] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger. 3d u-net: Learning dense volumetric segmentation from sparse annotation. *CoRR*, abs/1606.06650, 2016. 1

[6] J. M. Coughlan and A. L. Yuille. Manhattan world: Compass direction from a single image by bayesian inference. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 941–947. IEEE, 1999. 1

[7] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry. Atlasnet: A papier-mâché approach to learning 3d surface generation. *CoRR*, abs/1802.05384, 2018. 2

[8] K. Guo. 3d mesh labeling via deep convolutional neural networks. *CoRR*, abs/1712.06584, 2015. 2

[9] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik. End-to-end recovery of human shape and pose. *CoRR*, abs/1712.06584, 2017. 2

[10] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3

[11] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. Occupancy networks: Learning 3d reconstruction in function space, 2018. 1, 2, 3, 4, 5

[12] M. O. A. G. Michael Niemeyer, Lars Mescheder. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. 2020. 1, 3

[13] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger. Convolutional occupancy networks, 2020. 1, 2, 3, 4, 5

[14] M. Peyghambarzadeh, F. Azizmalayeri, H. Khotanlou, and A. Salarpour. Point-planenet: Plane kernel based convolutional neural network for point clouds analysis. *Digital Signal Processing*, 98:102633, 03 2020. 1

[15] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *CoRR*, abs/1612.00593, 2016. 2

[16] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *CoRR*, abs/1706.02413, 2017. 2

[17] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. 2015. 1, 3

[18] C. Urmson. Autonomous driving in urban environments: Boss and the urban challenge. *CoRR*, abs/1609.05143, 2008. 1

[19] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. *CoRR*, abs/1609.05143, 2016. 1