Supplementary material for Dynamic Plane Convolutional Occupancy Network

Stefan Lionar, Daniil Emtsev, Dusan Svilarkovic

Abstract

*In this **supplementary document** we first give a detailed overview of our architectures of additional experiments with spherical encodings and rotation matrixes we tried for our 3d object reconstruction in section 1. Next we introduce our improvement of the plane predictor network. We then provide additional experimental results, both qualitatively and quantitatively in section 4.*

1. Spheres encodings

For our experiments with sphere, we remove the plane predictor networks and solely use ResNet PointNet. Our new pipeline with sphere encodings is illustrated in the 1

Spherical Projection: As depicted in Figure 2 and 3, the sphere is divided into latitudes and meridians, (α, β) based on its angles. A point is projected by casting a ray passing through it from the origin to the surface of the sphere. The sphere surface is discretized into an equirectangular grid.

Point is assigned a label according to the region this point belongs to. For example, if the sphere is "voxelized" by 3 meridians then point laying between 0 and 120 degrees is assigned a label 0. So, with that, in encoder part every point has coordinates [i,j] where i $\in$ [0, m], j $\in$ [0, l], l,m are the number of latitudes and meridians correspondingly.

Sphere Height Map Surface of the sphere is discretised in the same way as in "Sphere Projection". We have also included radius distance since encoding using latitude and meridian is ambiguous. Multi-radius height descriptor represents a 3D shape with multiple instance of radius distance map. So, every bin determined by exact latitude and meridian also stores a list of radius distances between points and sphere center. We include in the list just 0, 50 and 100 percentage percentiles (min, mean and max) of all radius distances. Then we concatenate d features from pointnet and the list of 3 radius-heights. Finally, we store d + 3 radius heights in every bin. In the 9 we have shoen an example with 2 planes

Sphere Voxel Encoder In addition to the latitude/meridian discretisation we divide points into n groups depending on the distance to the center. Every point is assigned meridian/latitude coordinates and the length of its radius vector (from the center of sphere). Depending on radios we cluster points into several "depth-layers". For every layer we query just points belonging to this layer and distribute over bins (latitudes/meridians). As we did with several planes-channels we have here several channels corresponding to the every depth-layer. On the picture 5 there is an example with n = 3 layers.

Multiple Spheres

Since if the center of sphere is in the origin there can be non-propotional captioning of points. So, for those parts of reconstruction object closer to the center every spherical bin captures smaller chunks than for more distant parts. To resolve that problem we have come up with 2 solutions:

1. Move center of sphere from the origin to some vector. To do that we can:
   (a) Learn positions as planes orientations
   (b) Predefine constant spheres positions
2. Change kernel size of convolution

in the "latitude-meridian bin" if points are closer to the center Instead of having one sphere positioned in the center (origin [0,0,0]) we learn the position of centers of several spheres. We use PlanenetEncoder architecture and output [Batch, numspheres, 3] Here instead of "number of planes" we have "number of spheres". We learn vectors connecting global origin with centers of spheres. On the picture 6 there is an example with 4 spheres.

Aggregation of Planes and Sphere encodings

For every bin of the plane we have added meridian/latitude information (we have tried trigonometric sin/cos and number of latitude/meridian). So, for every point projected to the special bin on the learned plane we add to the net feature vector the radius-distance from the origin till points, latitude/meridian coordinates, height-map as the distance from points till the plane

2. Learning Rotation Matrix

Inspired by works of Camera Position Estimation and rotation of cameras ([3], [2]) we have decided to learn 3*3 rotation matrix. If we want to predict for example the orientation of 7 planes then firstly we have 7 normals equally oriented. Then we learn 7 Rotation matrixes to rotate these 7 first normals and get the orientation of 7 planes.

3. Point Plane net Kernel

We have also implemented Plane kernel from [1]. We had an idea initially to learn planes which capture main -”principal” components of the object. Then we decided to proceed with projections of points to multiple ”dynamic” learned planes

4. Triplet Loss

To penalise the position (orientation) of learning planes normals we have tried ”cosinus similarity loss” pairwise between normals:

$$L_{similarity} = \frac{\sum_{\substack{i,j \\ i \neq j}}^{n} |\cos(normal_i - normal_j)|}{n} \quad (1)$$

Then to ensure that we do not have all identical planes (cos = 1) between lots of pairs we use margin:

$$L_{triplet} = min(L_{similarity} - 1, 0) \quad (2)$$

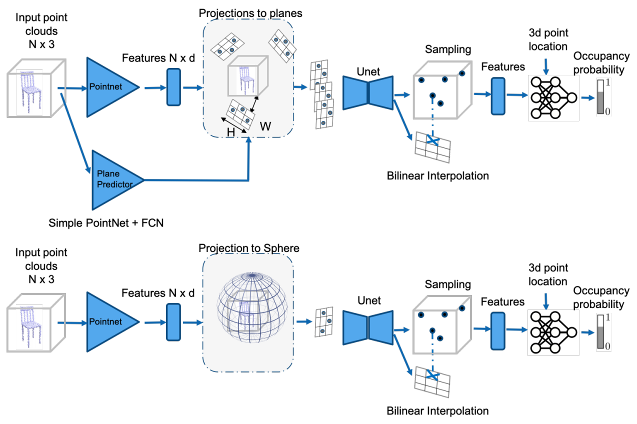

Figure 1: Pipelines with plane projections (top) and sphere projection (bottom)

5. Possible improvement of the plane predictor network

A possible improvement of the plane predictor network is illustrated in Figure 7. To allow backpropagation, the last layer of plane predictor networks is added to the last layer of ResNet PointNet. The difference between the version in the report is that it has L × subnetworks where L is the number of planes. The output of this subnetwork is added to ResNet

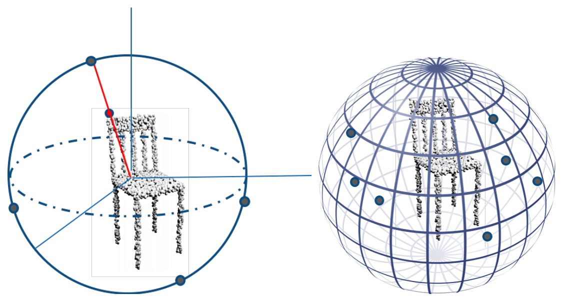

Figure 2: A point, p is projected by casting a ray from the origin to the surface of the sphere passing through p. The coordinates on the surface of the sphere are represented by meridian and latitude angles: (α, β) The image on the right represents ”sphere discretization”

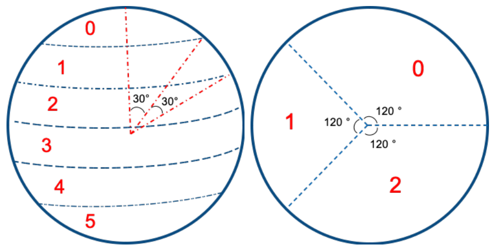

Figure 3: Example of sphere discretization with 5 Latitudes and 3 meridians.

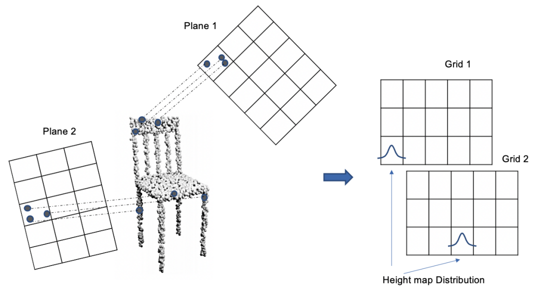

Figure 4: Height map. Every bin stores the distribution of distances from points quered in this bin till the plane

PointNet individually depending on the plane that is being processed, e.g. subnetwork 1 is added when processing the first plane, subnetwork 2 when processing the second plane, etc.

We tried this architecture and the previous version, adding to Peng, et al. Convolutional Occupancy Networks code instead of our implementation. We trained 3-plane DPCO until 430,000 iterations. The architecture in Figure 7 achieved the lowest validation loss of 17.67, while the version presented in report achieved 18.37, but both models achieved similar highest validation IoU score (0.8716 vs 0.8717).

In all of the 3-DPCO experiments using Peng’s code,

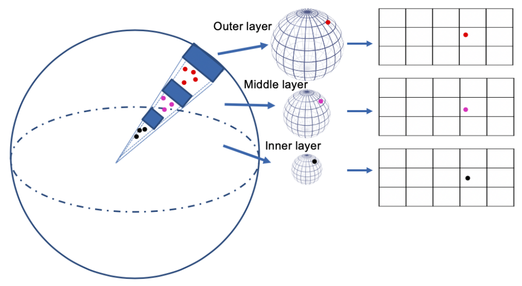

Figure 5: Voxels. There are 3 depth-layers on the picture For every depth layer points laying in this layer (black, purple and red) are quered using a spherical latitudes/meridians. Then 3 spheres is transformed into equidistant mesh

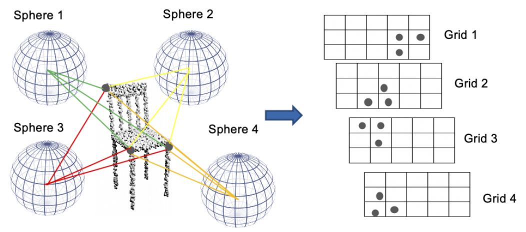

Figure 6: Multi spheres. We learn the position of spheres centers. Every sphere is transformed then into equidistant mesh

three canonical planes are always obtained. The version on the report was run once, and the version presented here was run multiple times (about 5) with different tuning.

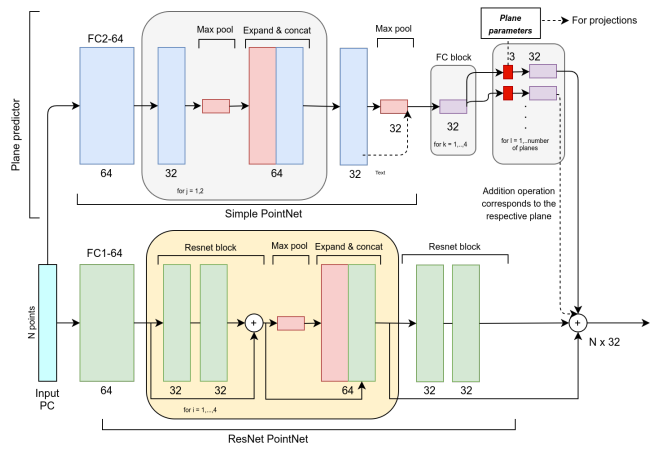

Figure 7: A possible improvement of the plane predictor network

6. Results

Results of Spheres Encodings The best result is with 7 multiple learned spheres. Sphere with the radius distance and voxels haven't shown improvement over just sphere. Sphere divided by depth voxels has shown the worse result. Aggregation of sphere-plane-height map hasn't shown superior result as well.

Results of Height Map We have tried with 2 planes and with 7 planes. Height-map with 2 planes has given us better result as it was expected given that height map with planes more than 2 contains redundant information. However, the results with 2 dynamic planes are better than with height map.

Results of the Triplet Loss Results have shown that triplet loss allows to learn more diverse planes. It reduces redundance and allows to learn planes different from canonical ones. With 0 margin we got a slight improvement in 2 objects. The best result has shown the triplet loss with margin cos(360/7) We interpret it as the most suitable margin to learn the most diverse distribution of 7 planes in 360 degrees space.

Results of the Rotation Matrix Learning the rotation matrix of planes normals hasn't improved our results

	IoU	Chamfer-$L1$	Normal C.
Sphere Encodings			
sphere	0.750	0.009	0.803
sphere height	0.731	0.009	0.786
sphere 3 voxels	0.675	0.012	0.769
sphere const 4	0.723	0.010	0.760
sphere 3 learned	0.754	0.008	0.818
sphere 7 learned	**0.760**	**0.007**	**0.823**
aggregation	0.712	0.016	0.799

Table 1: Metrics of our models with sphere encodings. IoU and Normal Consistency: Higher better. Chamfer-$L1$: Lower better.

	IoU	Chamfer-$L1$	Normal C.
Height Map			
height 2 planes	**0.832**	**0.006**	**0.909**
height 7 planes	0.641	0.013	0.855

Table 2: Metrics of our models with height map and learned planes. IoU and Normal Consistency: Higher better. Chamfer-$L1$: Lower better.

	IoU	Chamfer-$L1$	Normal C.
Rotation Matrix			
7 planes	**0.859**	**0.052**	**0.928**
Rot matrix 7 planes	0.810	0.012	0.855

Table 3: Metrics of our models with rotation matrix and learned planes. IoU and Normal Consistency: Higher better. Chamfer-$L1$: Lower better.

	IoU	Chamfer-$L1$	Normal C.
Triplet Loss			
margin 0	**0.849**	**0.005**	**0.917**
margin $\frac{1}{7}$	0.854	0.005	0.918
margin $\frac{1}{3}$	0.851	0.005	0.916
margin $\frac{2}{3}$	**0.861**	0.005	**0.922**
margin 1	**0.861**	0.005	0.921

Table 4: Metrics of our models with triplet loss for 7 learned planes. IoU and Normal Consistency: Higher better. Chamfer-$L1$: Lower better.

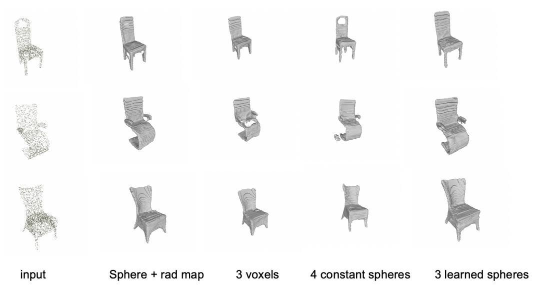

Figure 8: Object-level 3D reconstruction from Point Clouds. We compare different spherical encodings on the ShapeNet ”chair” category

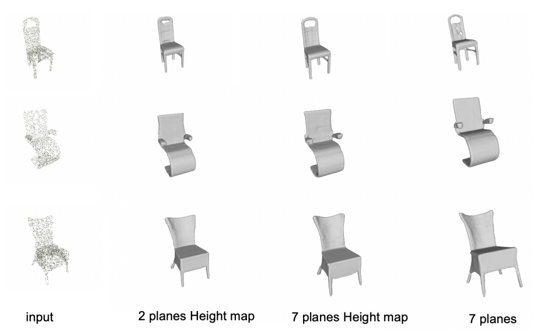

Figure 9: Object-level 3D reconstruction from Point Clouds. We compare different planes with height maps with 2 and 7 planes and just 7 planes on the ShapeNet ”chair” category

7. Discussions

Learning sphere centers positions with the application of meridian/latitude spherical encodings is a novel idea. Learning of spheres positions has shown better results with comparison to the one/multiple constant spheres. We have shown that increased number of sphere improves accuracy.

Triplet loss allows to learn the diverse distribution of planes.

References

[1] 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. 2

[2] Y. Wu and Y. Liu. Position estimation of camera based on unsupervised learning. *CoRR*, abs/1802.05384, 2019. 1

[3] W. Xian, Z. Li, M. Fisher, J. Eisenmann, E. Shechtman, and N. Snavely. Uprightnet: Geometry-aware camera orientation estimation from single images. *CoRR*, abs/1802.05384, 2019. 1