A Survey in 3D Point Cloud Reconstruction Using Deep Learning

Yipeng Liu, Yide Ma, Zhuocheng Guo, Mingwei Sui

1. Introduction to point cloud reconstruction

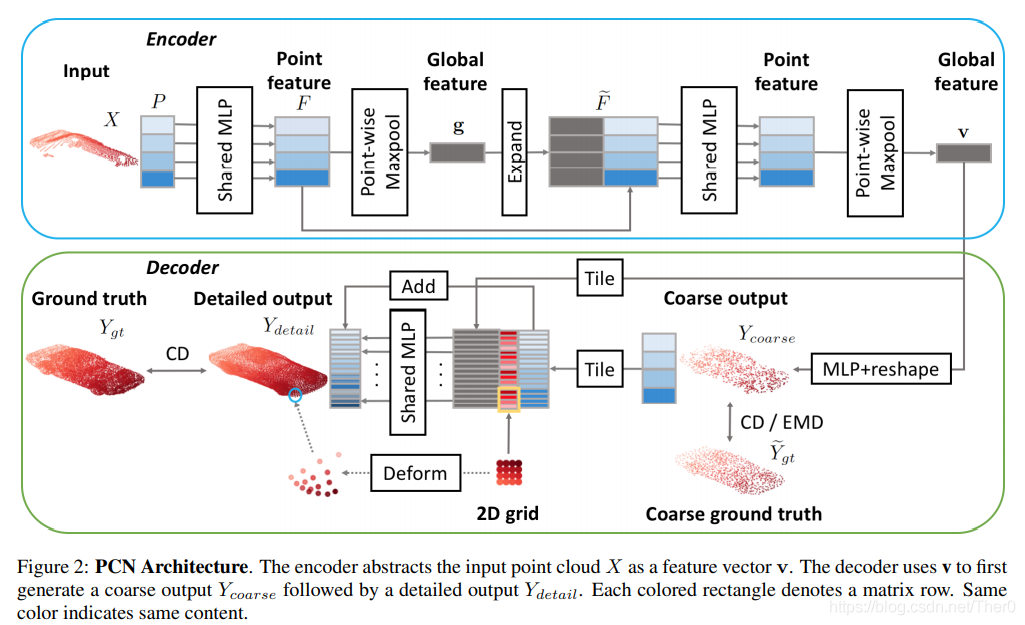
The problem of recognition and segmentation of three dimensional point cloud data is the key problem in the computer vision field. Point cloud data is usually obtained by three dimensional sensors, for instance, binocular cameras, three dimensional scanners and RGB-cameras. Point cloud data is very clear to observe because it has coordinates three dimensional. In the real world, the point cloud data we obtained is always partial, hence, to get a complete shape is crucial to computer vision. We chose to use point completion neural networks for point cloud reconstruction. In our project, we were given 4 groups of plants. We use v-lab to generate 1000 randomized samples for each group, then use a randomization algorithm to make 8 partials for each plant point cloud in 1000 samples of a group. This is the datasets we used to train the point completion networks we searched online.

2. Point Completion Networks

In our project, we used point completion networks for reconstruction. The purpose of this type of neural networks will use both partial and complete point cloud data to train the neural networks. Once we have trained the neural networks prourly, we could input and partial point cloud data to obtain the predicted reconstruction point cloud data. In our project, we researched and re-implemented the typical point completion, PCN, PF-net, Folding-net and recent point cloud nets, PMP-net, SnowFlake and PoinTr-net. For the recent neural nets we found, all of them were finished in 2021. In our project, before we implement these networks using our dataset, we think the recent ones will give better results as it should be. Once we have all the results, images, loss comparison, etc, we will discuss the performance in our report. In this survey, we compared the structure of different point completion networks and we will know which point completion networks fit best for plant-related point cloud data.

2.1 Point Completion Network

PCN is regarded as the first point completion network when using deep learning in 3-dimensional construction. It is an end to end network with a combination of encoder and decoder. The core work lies in the design of the decoder. For the data, PCN directly processes the original point cloud, hence structural consumption (such as symmetry) or annotation (such as semantic class) would not be introduced to the network by handling 3 dimensions in this way. It prevents loss of information caused by voxelization. For the encoder of PCN, it was used by two pointnet layers.



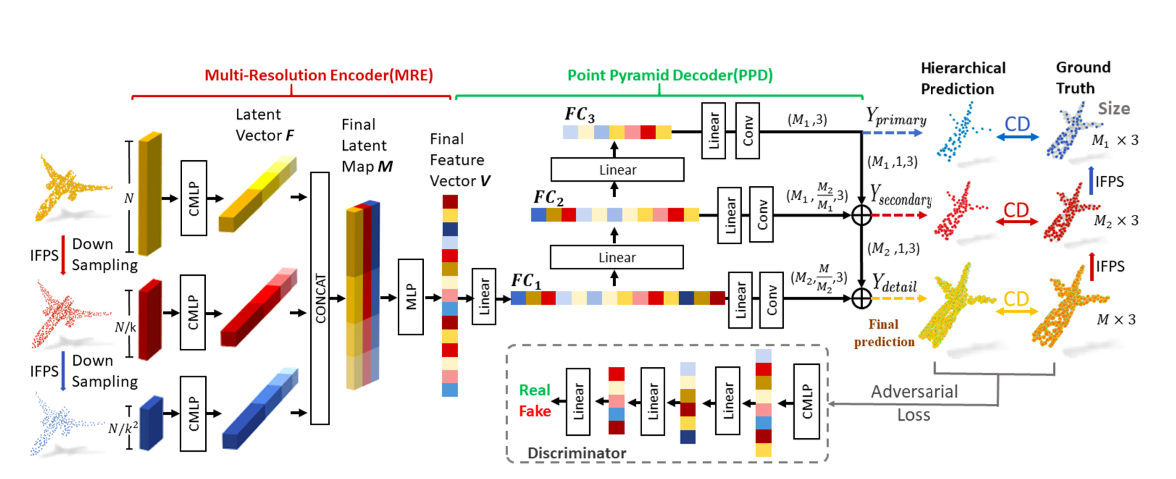
For the encoder, it will transfer the input point cloud data as a vector which represents the features of the data. (We also want to mention in GR-net, it uses the same way in this part) The encode is a stack of two pointnet layers. The first Pn layer will use m input points to show a m by 3 matrix P. For each line is x, y, z. Then it uses a MLP with Relu full connection layer to transfer each P matrix to the vector for the features, f. All of the fs make a matrix F. Once it has F, it use point-wise maxpooling for F to obtain a k dimensional global feature g. The second layer PN uses F and g as input, concatenating to get an augmented point feature matrix. Then pass it to another ML and point-wise maxpooling to get the final feature vector V. Specifically, we want to mention that a popular recent point completion net, GR-net use the same structure,(Cubic Feature Sampling).

For the decoder, it uses the feature vector V obtained from the encoder to output point cloud. In PCN, it combines a fully-connected decoder and folding-based decoder. The former has better performance at global geometric features, the latter could approximate smooth shape partial mesh well. For the decoder, it has two steps, the first one will output Y coarse, to convert V pass a fully-connected , output a s by 3 matrix using 3s. The second, use each point in Y coarse, by folding operation, to output a patch, then convert it to global coordinate. It will combine all s patch to get a Y details with all n point, which n=s\*t.

For the loss function of PCN, it uses two distances, chamfer distance and earth mover’s distance. For nets after PCN, these two ways of calculating distance are the most two widely used.

2.2 Point Fractal Network

PF-Net comes from the unsupervised Point Fractal Network; it uses the thought of Fractal geometry. Comparing many other networks, the similarity is that it uses incomplete point clouds as input, but the difference is it only outputs the cloud points that are missing. PF-Net consists of three parts: Multi-resolution Encoder (MRE), Point Pyramid Decoder (PPD) and Discriminator. MRE improves the efficiency of point cloud information extraction; PPD uses multi-stage completion loss, supervising the generation of key point clouds, thereby reducing geometric structural defects; Discriminator improves the phenomenon that the features of objects of the same category affect each other.



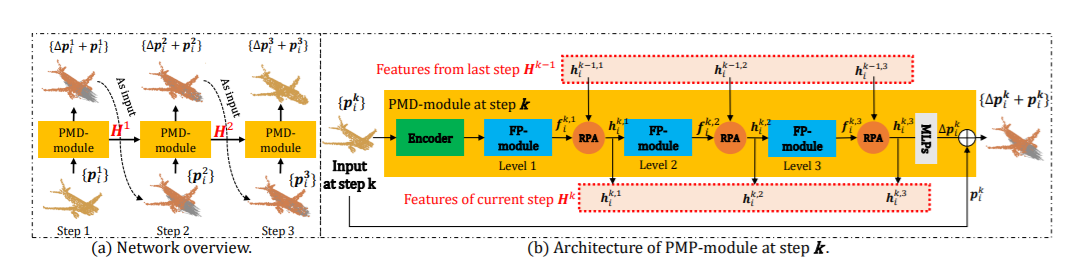
Firstly, Iterative farthest point sampling (IFPS) does better than random points sampling, the sampling result is able to represent better distributions of the entire point cloud. Secondly, Combined Multi-Layer Perception combines the last four layers, it takes care of both high-level and low-level features, and presents better results than PointNet-MLP that only outputs the last layer. Thirdly, PPD gets features FC1, FC2, FC3 by using the final vector V. FC3 gets the primary center points. FC2 gets the coordinates of secondary center points. From FC1 and secondary center points, getting the detail points. ALso, the detail points look for features from sampling of Ground Truth. Fourthly, PFNet uses Chamfer Distance (CD) for getting completion loss, which is more efficient by comparison with Earth Mover’s Distance (EMD).

As the final testing result, PFNet whether with discriminator or not, it gets the error value lower than many other nets, such as LGAN-AE, PCN, 3D-Capule. PFNet does pretty well in both details and the entire image, also keeping the geometry features of the input point cloud.

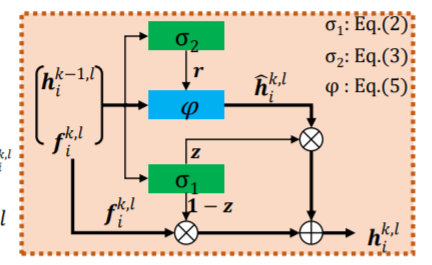
2.3 Point Cloud Completion By Learning Multi-step Point Moving Paths

On the basis of PCN, people developed different ways of handling point cloud data instead of just using MLP. In this net, the general idea is to move each point in the incomplete point cloud. By predicting the moving path for each point, PMP-net could complete the missing part.

The whole process in PMP-net is an iterative process, when P is the input point cloud, the task is to predict the movement to get a completed point cloud P’.



On the left of the picture, the network overview. It shows the three iterative processes. The first iteration, it will input the original point cloud, output the first amount of movement, use the original and result of the first iteration to do the same process for the second iterative and third iterative. For the architecture, it will pass a feature encoder first, in PMP-net, it uses pointnet++ to obtain global features. Once it has the feature, it uses a module to hand out global features to each point. The h, is the feature by point. The last step is to output the expectation of movement by each point to get by using h in an MLP. The FP-module is a structure in Pointnet++. It could get a global feature by feature of each point. Once it has the result, it will pass to the RPA module

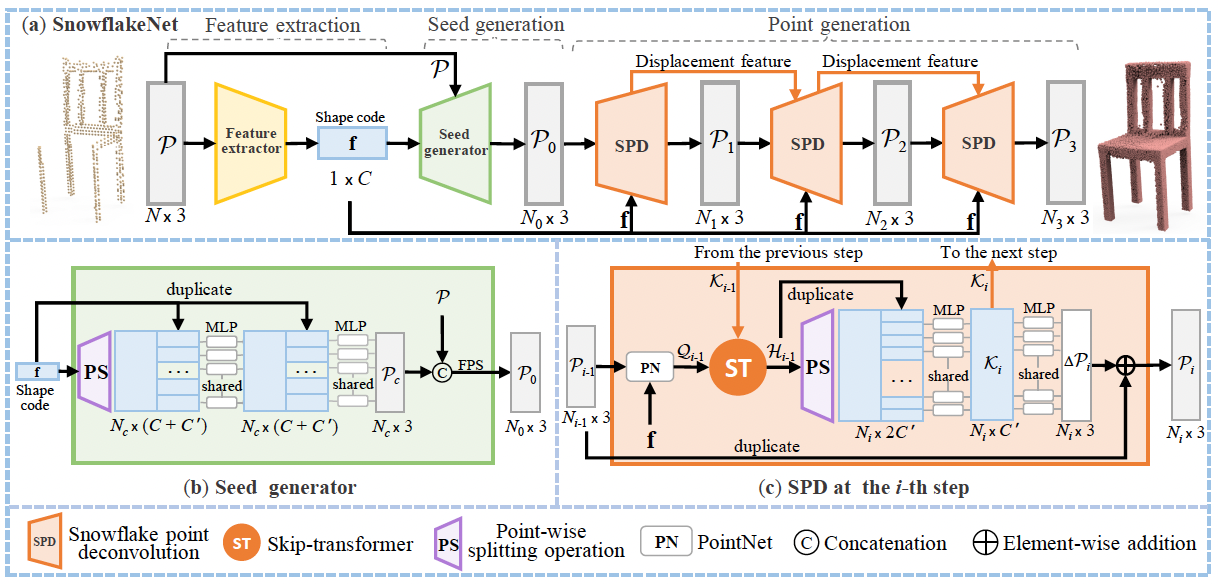


For this module, it uses the sum of information of the previous path to reason the position of each point. last, save the feature and to calculate the global feature. Since it is an iterative process, it needs RNN to save the information by time. For loss, it uses emd and chamfer loss to calculate.

2.4 SnowFlakeNet

SnowFlakeNet is a deep learning based point cloud completion method. It generates the completed point clouds as the snowflake-like growth of points in 3D space. The advantage of SnowFlakeNet is that it can predict precisely local details such as smooth regions, sharp edges and corners. Most of the point cloud completion networks have the ability to predict the whole framework, but ignore the local patches. Moreover, this network has another advantage. It enforces constraints on the process of children point cloud generation. It has a hierarchical structure like TopNet. At the same time, it keeps the generated process explicit and explainable like PF-Net.

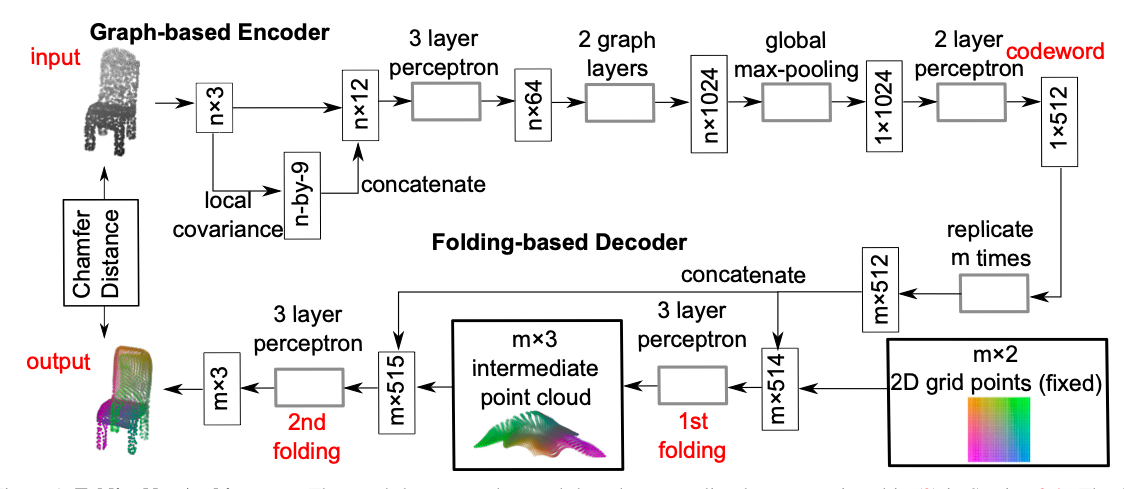
The core idea of this network is the relationship between the parent points and children points. This network will generate the parent potins at the beginning, and learn the basic 3D-geometry features. Then the next step is to generate the children points and embed the features of the parent points to the children potins. Finally, the network will put the generated children potins into the right position. After three iterations, the dense prediction will be generated.



The basic architecture of SnowFlakeNet has three parts: feature extraction module, seed generation module and point generation module. The first part is feature extraction. It takes the point cloud input and uses the feature extractor to extract a shape code. This shape code captures the global structure and local details of the input. Second part is seed generation. This seed generator aims to produce a coarse complete point cloud. In detail, the purpose of point-wise splitting operation is to generate each point feature. During this step, it captures the shape information about the input part and missing part by using the shape code. Then the generated per-point features integrate with shape code by using Multi-layer Perceptrons. The output is a coarse point cloud. The last step is to merge the coarse point cloud with the input point cloud through farthest point sampling. The final output is a down-sampling point cloud. It will be the seed point cloud for the next module. Third part is point generation. This part aims to generate the final prediction of a dense point cloud. It contains three iterations and each iteration will produce a bunch of children points from previous parent points. Also, this part is the essence of the network. It proposes Snowflake Point Deconvolution (SPD) and Skip-transformer structure. In each iteration, the input is parent point. Then the parent points through the SPD to get the next level children point cloud. In each SPD block, the first mission is to extract the shape code of parent points by using PointNet. Then the Skip-transformer will learn the spatial context from the displacement feature and shape code. The displacement feature from the last layer and the shape code from the parent points. After the Skip-transformer, the output is a shape code. Then the Point-wise splitting operation generates the children points features. This process is the same as the second part. Then the children points features will combine with shape codes through multilayer perceptrons. The result is a displacement feature for the next step. This feature will generate point displacement through other Multi-layer Perceptrons. Finally, the point displacement concatenates with the parent point cloud to generate the high resolution points. For the training loss, the Chamfer Distance(CD) is the primary loss function. Besides, the sum of four density CD losses are defined as completion loss. Total loss is the sum of completion loss and preservation loss. Preservation loss is the evaluation about how the shape structure is related to the input point cloud.

2.5 Folding Net

Folding Net includes two main parts which are graph-based encoder and Folding-based decoder. The input is the original point cloud, then the 2D grid points are collapsed during the decoding process. Next, first, second and third folding operations respectively, and the output is the reconstructed point cloud surface. The first fold is to fold the 2D grid into a 3D space, and the second fold is to fold the interior of the 3D space. After two folds, the shape of the folded surface is very delicate. Do more folds, output more accurate results.



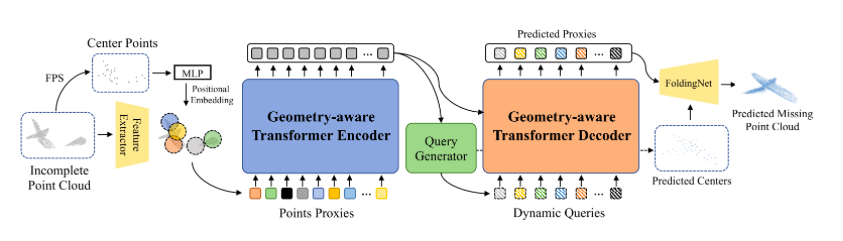
Firstly input is cloud with n points, calculate the local covariance matrix of 3\*3 for each point, and vectorize it to size of 1\*9, and then connect the n\*9 matrix and n\*3 matrix obtained into n\*12. Secondly, the matrix is input into the three-layer perceptron. Through a 3-layer perceptron, the output of the perceptron is input to two consecutive image layers, where each layer of image performs maxpooling dimensionality reduction for the neighbors of each node, and then 2-layer perceptron, get the codeword. Thirly, the obtained codeword vector is superimposed on m layers, and a two-dimensional grid point is added to perform the folding operation. Fourthly, the decoder contains two folding operations. This design reduces the number of parameters. Finally, a distance function is defined to evaluate the difference.

When the original surface is too complicated, there will be distortion, it needs more input point samples, more complicated encoding and decoding networks. When the dataset is large(e.g. 10k training models, 2.5k testing models), the folding net has higher accuracy. When the dataset is small(e.g. 4k training models, 1k testing models), the accuracy is not as good as a large dataset.

Overall, the folding decoder has high accuracy and low reconstruction loss. Compared with the fully-connected decoder, our proposed decoder operates more naturally and uses fewer parameters.

2.6 PoinTr Net

When completing the point cloud, PointTr first processes the point cloud into a fixed number of point proxies, which can be used as the input of the Transformer. Then encode the existing point cloud through Encoder, and generate the point cloud center and the corresponding dynamic Queries of the first stage through the Query Generator. Finally, these Queries are translated into point agents through Decoder. Point proxy through a FoldingNet to get the offset relative to a specific center point. By moving the corresponding center, we can get the local point cloud corresponding to a point proxy.



- Transformer's encoding and decoding structure:

Using the encoder and decoder structure to complete the conversion of the point cloud into a set-to-set translation problem. The transformers’ self-attention mechanism models all pairwise interactions between elements in the encoder, while the decoder interprets the missing elements based on the features of the input point cloud and learnable pairwise interactions between queries

- Point Proxy

To use the point cloud as the input of the Transformer, first to process the point cloud into a sequence. The simplest idea is to take each point as an element of the sequence as input, but this will bring a very large burden of computing resources. Therefore, we propose that the point cloud can be processed into a series of point proxies to represent a local area feature on the point cloud. First, we sample the farthest point (FPS) of the point cloud to obtain a fixed N center points; then, we use a lightweight DGCNN to extract the features of the local area, so that we can get the features of the N local areas, Which corresponds to the characteristics of the area as the center point. Finally, we use an MLP network to extract the positional embedding of each local feature, and add it to get a point proxy, that is, as the input of the Encoder.

- Geometric-aware Transformer block

In order to facilitate the Transformer to make better use of the induced deviation of the three-dimensional geometric structure of the point cloud, PoinTr designed a geometric perception block, which clearly simulates the geometric relationship.

- Query generator

Author using dynamic queries in the decoder instead of fixed queries. They are generated by the query generation module, which summarizes the features generated by the encoder and represents the initial sketch of the missing points;

- Multi-scale point cloud generation

The goal of the encoder-decoder network is to predict the missing part of the incomplete point cloud. However, we can only obtain predictions of missing agents from the converter decoder. Therefore, Author proposes a multi-scale point cloud generation framework to recover the lost point cloud with full resolution. In order to reduce redundant calculations, authors reuse the coordinates generated by the query generator as the local center of the missing point cloud.

PoinTr Net only predicts the missing parts of the point cloud and connects them with the input point cloud to obtain a complete object. Both the predicted agent and the restored point cloud are supervised during the training process.

- Optimization

The loss function completed by the point cloud should provide a quantitative measure of the output effect. However, because the point cloud is disordered, many loss functions that directly measure the distance between two points are not appropriate. Author uses Chamfer Distance and EMD as the loss function in this network.

2.7 Summary

In summary, the overall point cloud completion networks could be divided into two parts. Based on decoding methods, PCN, PF-Net and PMP belong to the coarse-to-fine network. They all generate the different layers of point clouds from coarse to dense result. PoinTr Net has been optimized on the Folding Net, they belong to folding based decoding. PF Net and PMP Net has been optimized on the PCN.

Reference:

[1] W. Yuan, T. Khot, D. Held, C. Mertz and M. Hebert, "PCN: Point Completion Network," *2018 International Conference on 3D Vision (3DV)*, 2018, pp. 728-737, doi: 10.1109/3DV.2018.00088.

[2] Wen, X., Xiang, P., Han, Z., Cao, Y., Wan, P., Zheng, W. and Liu, Y., 2021. PMP-Net: Point Cloud Completion by Learning Multi-step Point Moving Paths. [online] arXiv.org. Available at: <https://arxiv.org/abs/2012.03408> [Accessed 12 December 2021].

[3] Zitian Huang, Yikuan Yu, Jiawen Xu, Feng Ni, and Xinyi Le. PF-Net: Point fractal network for 3D point cloud completion. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 7662–7670, 2020.

[4] Xiang, P., Wen, X., Liu, Y. S., Cao, Y. P., Wan, P., Zheng, W., & Han, Z. (2021). Snowflakenet: Point cloud completion by snowflake point deconvolution with skip-transformer. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 5499-5509).

[5] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. FoldingNet: Point cloud auto-encoder via deep grid deformation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 206–215, 2018.

[6] Yu, X., Rao, Y., Wang, Z., Liu, Z., Lu, J., & Zhou, J. (2021). Pointr: Diverse point cloud completion with geometry-aware transformers. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 12498-12507).