Not All Points Benefit Structural Learning

Hongye Hou
Xi'an Jiaotong University
houhongye2001@stu.xjtu.edu.cn

Jiaxin Ren
rjx123@stu.xjtu.edu.cn

Fangyu Du
zjdfy2019@stu.xjtu.edu.cn

Yuxuan Jiang
3123358014@stu.xjtu.edu.cn

Xiaobin Zhai
Spike@stu.xjtu.edu.cn

Yang Yang *
yyang@mail.xjtu.edu.cn

Abstract

Generating objects from point cloud data is becoming increasingly important in the field of computer vision. In this paper, the Contour2Roof model is proposed for the problem of constructing 3D wireframe models of buildings. First, we propose an Efficient Structural feature Extractor (ESE) for point clouds. The extractor downsamples the point cloud in the process to bias the contours and learns the more focused structural information in the point cloud, realizing the generation of more accurate wireframe models while using fewer points. We also design a global attention-based Feature Amplifier to enrich the acquired features with similar structures in the whole point cloud. Our approach achieves remarkable results in experimental validation and provides new ideas and methods for the field of structural learning of point cloud data.

1. Introduction

With the development of 3D scanning devices, point cloud data is becoming increasingly abundant and driving the field of computer vision towards 3D scene understanding. More and more tasks tend to understand the structure of 3D objects, such as defective point cloud completion, 3D semantic segmentation, and instance segmentation. One of the emerging tasks is constructing a point cloud wireframe model of a building, which is based on an urban rooftop dataset and generates the corner points of an object and the connecting lines on the contours to form a 3D wireframe model of the object[9].

Due to the development of deep learning, most of the recent approaches to solve this task are based on an end-to-end process, but these approaches have some drawbacks: 1) Not all points are helpful for contour generation and we only need to consider the points on the contour when constructing the wireframe model. 2) Prior approaches such as Point2Roof lacked the exploration of the global correlation[7], and the roof structures have a strong Inter-similarity is a strong guide for wireframe model generation.

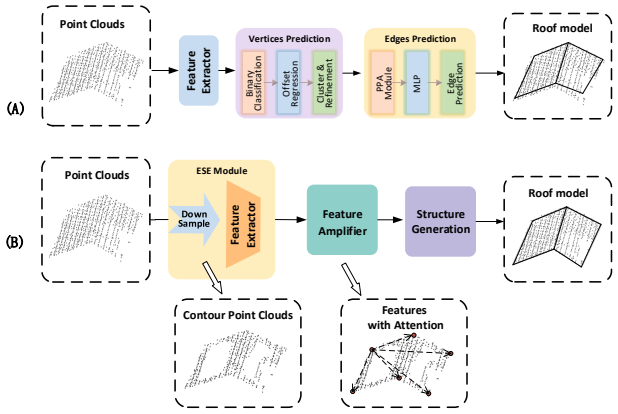

Figure 1. Illustration of the main ideas. Figure (A) shows the flow of Point2Roof and Figure (B) illustrates the innovative work of our proposed Contour2Roof. We have incorporated modules such as an Efficient Structural feature Extractor(ESE) and a Feature Amplifier, achieving better generation results.

We propose in this paper an end-to-end deep network, named Counter2Roof, which consists of an Efficient point cloud Structural feature Extractor (ESE) and a Feature Amplifier (FA) to solve the wireframe generation problem. Our philosophy can be summarized as "Not All Points Benefit Structural Learning ". The differences between our proposed model and Point2Roof are shown in Figure 1. ESE focuses on learning the more structurally useful points in the point cloud, which tend to be corner and contour points. FA makes each point learn the features of similarly structured points in the whole point cloud through a global attention mechanism. Doing so allows us to achieve better results with fewer points. Specifically, our ESE approach was inspired by Point2Roof in the completed process, but

**Corresponding author

we used a more efficient edge-conv feature extractor focusing on learning points on the contours, and this downsampling process, based on calculating and ordering the offsets between a point in the point cloud and the center of its k-neighbors, biased-sampled the 2,560 points down to 1,560 points while the results became better. Additionally, we designed a global attention-based Feature Amplifier in to distinguish points at different locations in the subsequent process. After validation, our plug-and-play feature extractor significantly improves the results and efficiency of our model and can be extended to other tasks of learning point cloud structure, such as partial point cloud completion. Overall our contributions are as follows:

- We propose a new Efficient point cloud Structural feature Extractor named ESE that focuses on learning features for points located on the contours, we focus on fewer points and learn richer features.
- We design a Feature Amplifier as a global attention-based recurrent learner that learns distantly related features to help generate more reliable key points and build connecting lines based on them.

2. Related Work

3D reconstruction of buildings is an important means of obtaining structural information about buildings. In recent years, constructing 3D building roof models from point cloud data has attracted more and more attention. Traditional methods usually use manual or semi-manual approaches to achieve alignment, stitching, feature extraction, and modeling of point cloud data. With the enrichment of publicly available point cloud datasets and the continuous development of deep learning, the application of deep learning techniques to building reconstruction from point clouds is increasing.

2.1. Traditional methods

Traditional methods can be roughly divided into two clusters: solutions based on the concepts of Top-down and Bottom-up.

The Top-down methods are realized by model-driven. A priori knowledge of known buildings is utilized to guide the processing and analysis of point cloud data. Kada et al.[6] proposed a 3D reconstruction technology based on the unit decomposition method. After that point cloud data is decomposed into several small units or areas, each unit is analyzed to identify the building structure and features, and the 3D model of the entire building is reconstructed by combining the information of these units. Poullis et al.[10] proposed an easily scalable, parameter-driven shape primitive to detect and reconstruct building structures. Henn et al.[4] used supervised machine-learning methods and model-driven approaches to derive the optimal roof model and improve the accuracy of model selection. Zheng et al.[21] divided buildings into four building primitives based on the physics and morphology of the buildings by using a decision tree classifier, and then completed roof reconstruction using the length, width, and direction of each building block. These methods require certain assumptions about the shape and structure of the building, and then the 3D reconstruction is achieved by fitting, optimizing, and combining these pre-defined models. The reliance on pre-defined models limits flexibility in dealing with buildings with complex shapes and structures and makes it difficult to reconstruct buildings that do not match the pre-defined models.

The Bottom-up methods are realized by data-driven. Unlike model-driven, data-driven does not rely on pre-defined prototypes or shape models but rather derives the structure and shape of the building based on the characteristics of the point cloud data itself. Chen et al.[3] proposed a method for automatically segmenting and reconstructing urban 3D buildings. This method uses a segmentation algorithm to divide point cloud data into different building subsets and reconstructs buildings by analyzing geometric and topological structure features. Li et al.[8] proposed modeling buildings from aerial lidar point clouds using triangular irregular meshes and label maps based on topological relationships and geometric features. Song et al.[14] used contour clustering technology to accurately extract the contours of buildings and then achieve the 3D reconstruction of curved buildings. Sohn et al.[13] used a binary space partition tree to spatially segment and organize point cloud data, extracted the facades and roof surfaces of the building by analyzing the structure of the binary space partition tree, and integrated these surfaces into a complete polyhedral building model. Awrangjeb et al.[2] constructed a matrix to represent the topological relationship between roof surfaces, using the minimum ring theory proposed by Verma et al.[15] to analyze the 3D plane intersections between adjacent planes to determine the missing planes, and finally obtained a complete building model by inserting approximate wall planes and building ground. These methods directly take the point cloud data as input and utilize an algorithm to extract some of the features of the building, transforming the 3D reconstruction problem into a topological consistency problem with multiple points, lines, surfaces, and their combinations. These methods do not impose specific constraints or requirements on the shape of the building but are difficult to determine for complex topological relationships or geometric structures. They are also computationally expensive and sensitive to density variations, absences, and spatial distribution inhomogeneities in the point cloud.

2.2. Deep learning-based methods

Recently, the great success of deep learning in image processing has catalyzed research on learning features from point cloud data, laying the groundwork for 3D reconstruc-

tion tasks on point cloud data.

Point cloud data feature extraction. PointNet [11] learns its corresponding spatial encoding for each point in the input point cloud and then aggregates individual point features to form global features. The extracted global features are able to perform the classification task well, but the local feature extraction capability is poor, which makes it difficult to analyze complex scenes. PointNet++ [12] proposes a multilevel feature extraction structure based on PointNet, which effectively extracts local features, but ignores the relationship between points. DGCNN [17] incorporates relationships between points to capture local geometric relationships in point cloud processing introduces a dynamically updated graph model, and recalculates distances between points in each layer of the feature space to better allow information to propagate between similar structures and accelerates the learning of local semantic information.

End-to-end 3D reconstruction. PC2WF [9] is the first end-to-end trainable deep network architecture for converting 3D point clouds into wireframe models. The method formulates the point cloud 3D reconstruction problem as a two-step classification problem for points and edges. First, the points are encoded as feature vectors, and the candidate vertices are identified based on the features and then pruned to obtain the final corner vertices. Then the corners are connected to the exhaustive set of marquee edges and pruned again to obtain the final wireframe model. Point2Roof [7] follows this process, using the PointNet++ network to analyze and process the point cloud data to obtain a more comprehensive representation of the features and refine the vertex determination process. The paired point attention method is also used to fuse vertex feature pairs to extract salient edge features and effectively predict edges in real 3D structures. These methods process and analyze all points in the input point cloud data, but not all points carry valid information, which wastes computational resources and increases inference time. At the same time, these methods lack the exploration of global correlations, which is highly instructive for reconstructing the roof structure.

3. Method

Regarding the research problem of this task, we can understand it as inputting 3D point cloud data and outputting the final vertex-edge-constructed roof model. Therefore, we choose to use an end-to-end model called Contour2Roof, which can achieve this goal conveniently and effectively. The pipeline of Contour2Roof is shown in Figure 2. Based on the human thought process for solving this problem, Contour2Roof is divided into two major processes: the first step is to get the features of the 3D point cloud, and the second step is to predict the vertices and generate the correct edges based on the identified vertices. We'll specifically introduce the structure and functions of Contour2Roof.

3.1. Contour Biased Down-sampling

Since the wireframe generation task is essentially a process of learning and building the structure of a point cloud contour, points that are far from the contour do not contribute much to learning that structure, but instead add computational overhead and difficulty in learning important information. We, therefore, draw on Ahmed's method [1] of preserving points on the contour during the downsampling process. The procedure is as follows, first, we obtain the neighbors of each point by k-nearest neighbor search, represented as $\mathrm{V}_i = \{n_1, n_2, \ldots . n_k\}$. Then we calculate the center of its k-nearest neighbor.

$$C_i = \frac{1}{k} \sum_{j=1}^{k} n_j \tag{1}$$

And calculate the deviation D of each point from the center of its k-nearest neighbor, to enhance the robustness and prevent the uneven distribution of points from leading to different scales of deviation, this deviation is also divided by the distance from the point to the farthest neighbor.

$$D_i\left(\mathrm{V}_i\right) = \max_{n \in \mathrm{V}_i} \|p_i - n_i\| \tag{2}$$

$$I_i = \frac{\| p_i - C_i \|}{D_i} \tag{3}$$

Thus we obtain an ordering of the value I, which geometrically means that in the same point cloud, the larger the value, the closer the point is to a region of significant structural change, i.e., a contour or corner region. So we take the first M points of the ranking to filter out the points that are far from the contour.

3.2. ESE Module

To focus on the contour information, we design an Efficient Structural feature Extractor using a combination of biased contour downsampling and Edgeconv [18] proposed in the previous section for feature extraction.

First, for each point p_i , we identify its neighbor as p_j, forming its local neighborhood $\mathbf{N}(p_i)$. The edge vector e_{ij} for each pair of neighboring points is calculated as $e_{ij} = p_i - p_j$, providing the foundation for constructing the local structures. Next, EdgeConv is used to compute edge features by combining the feature vectors of the central point p_i and its neighboring point p_j. This is achieved through a non-linear transformation:

$$h_{ij} = \Phi(p_i, p_j - p_i) \tag{4}$$

The edge features are then aggregated using a max pooling operation as MAX:

$$x_i' = MAX(h_{ij} \mid p_j \in \mathbf{N}(p_i)) \tag{5}$$

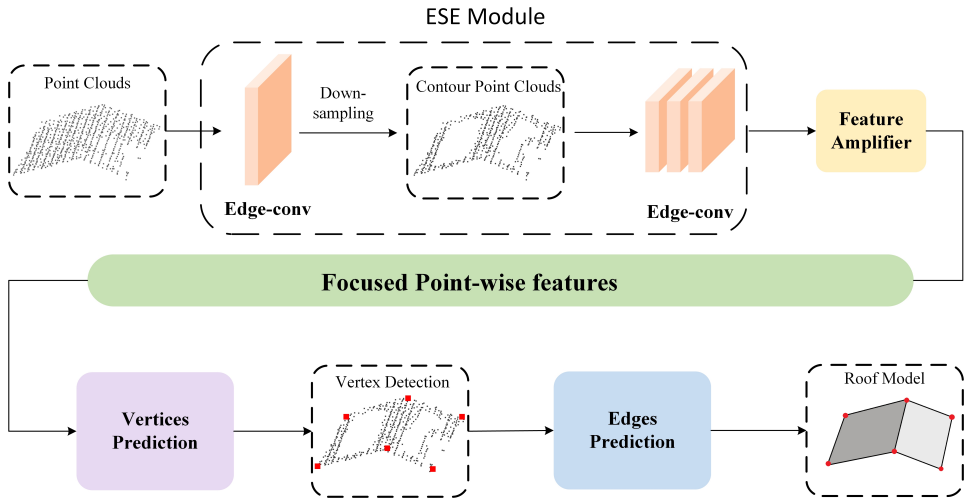

Figure 2. Pipeline of Contour2Roof. We obtain the initial features of the input 3D point cloud through the ESE module. Then, these features are amplified by the Feature Amplifier to get focused Point-wise features. Using these features, we further predict the vertices and, based on these vertices, predict the edges, ultimately outputting our generated roof model.

This generates new point features that encapsulate local geometric information. By stacking multiple EdgeConv layers, where each subsequent layer builds upon the features extracted by the previous layer, we achieve a hierarchical learning structure. In this way, we can capture the geometric and semantic information of a larger receptive field, gradually expanding from local details to a global receptive field.

3.3. Feature Amplifier And Vertex Prediction

After the ESE module, we believe that the feature information mainly comes from the local neighborhood, which lacks the guidance of global information, especially for the points with unclear distinction, and is likely to be misjudged in the subsequent process. Therefore, we decided to use Transformers to amplify the sensory field of features. Using the attention mechanism of Transformers, we can find the relationship between all the points. This enhances the role of location and its interrelationships in feature extraction, thus making feature extraction more effective. The specific structure is shown in the following figure:

We input the features $\mathbf{X}$ extracted by the ESE along with the positional vectors $\mathbf{P}$ into our Feature Amplifier. The $\mathbf{X}$ is passed through three linear layers to obtain the corresponding Queries, Keys, and Values. The $\mathbf{P}$ is passed through an MLP layer to get the Position Embedding of the same dimension as $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$. Referring to the method proposed by Zhao[20], we choose to use $\mathbf{Q} - \mathbf{K} + \mathbf{P}$ to obtain the intermediate variable $\mathbf{M_1}$. After passing through an MLP and normalization, we get the intermediate variable $\mathbf{M_2}$. The intermediate variable $\mathbf{N}$ is obtained by $\mathbf{V} + \mathbf{P}$. Finally, the Hadamard product of $\mathbf{M_2}$ and $\mathbf{N}$ is taken, and

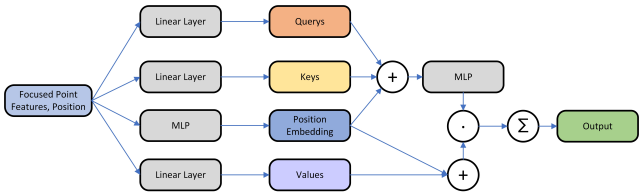

Figure 3. Module of Feature Amplifier. We take the original features and position information as input and output the amplified features.

the results are summed element-wise to get the final output $\mathbf{Y}$ and $\mathbf{P}$. We chose to repeat this process four times to fully extract the features. The structure is shown in Figure 3.

The formulas are as follows:

$$\mathbf{y_i} = \sum_{x_j \in \mathbf{X}} std(\phi(\alpha(\mathbf{x}_i) - \beta(\mathbf{x}_j) + \delta)) \odot (\gamma(\mathbf{x}_j) + \delta) \quad (6)$$

Where std stands for the function of standardization, α, β, γ represent linear layers, δ stands for position embedding, and ϕ represents the function of an MLP.

Vertex prediction is the key part of this task, serving as a link between the preceding and the following. This step predicts the roof vertices from the 3D point cloud data and provides the raw materials for subsequent edge prediction. For vertex prediction, Contour2Roof is divided into two main parts: obtaining dense point groups from the original point cloud, and predicting vertices from these dense point groups.

In this method, as what we proposed upon, we choose to use ESE to obtain features and amplify them. After obtaining the features, we start the first part, which is obtaining

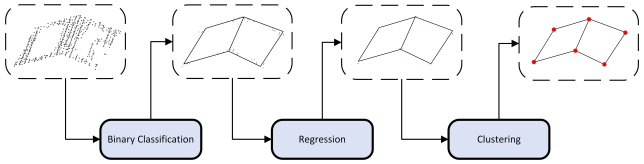

Figure 4. Procedure of vertex prediction. We first perform binary classification, then regress the offset values, and finally use clustering to obtain the predicted vertices.

dense point groups. Firstly, we notice that many points in the original point cloud do not help find vertices. Therefore, we decide to use binary classification to screen out points in the point cloud that are likely to be vertices, and we call these points candidate points. This step allows us to avoid many unnecessary calculations later, greatly improving the algorithm's efficiency and speed. However, the points we obtain are sparse, and the distance between points and the ground truth may be large. Therefore, we need to calculate the distance between these points and the ground truth and add offset values to correct them. This way, we can obtain a series of dense point clusters, which serve as the basis for subsequent clustering.

After completing these operations, we move on to the second step, which is to predict vertices using dense point groups. For a new 3D point cloud input, even if we obtain dense point clusters, it does not mean that the number of point clusters is the number of roof vertices. Therefore, we choose to use DBSCAN to cluster the dense point clusters and call the cluster centers initial vertices. After obtaining the initial vertices from clustering, we choose to use the Set Abstraction(SA) module of PointNet++[12] to extract features from the initial vertices. For each real vertex, we find the neighboring points within a certain distance and use the PointNet[11] layer to calculate its feature $\mathbf{F}_i$. After completing this step, we adjust the coordinates of the initial vertices based on the feature differences between the initial vertices and real vertices, thus obtaining the final predicted vertices. The structure of vertex prediction is shown in Figure 4.

3.4. Edge Prediction

After predicting the vertices, we naturally move on to predicting the edges of the roof. First, we choose to connect all the edges between the vertices and construct a graph $\mathbf{G}$, where $\mathbf{G} = \{\mathbf{V}, \mathbf{E}\}$, with $\mathbf{V} = \{\mathbf{v}_i\}_{i=1}^{N_v}$ representing all the vertices and $\mathbf{E} = \{\mathbf{e}_{ij}, i \neq j\}$ representing all the edges. N_v is the total number of vertices. Our task is to determine whether each edge should be retained in the final roof result based on the features of the vertices on both sides of the edge. In other words, we need to obtain the edge feature $\mathbf{F}_{ij}$ based on the features $\mathbf{F}_i$ and $\mathbf{F}_j$ of two vertices. In Contour2Roof, we choose to use Paired Point Attention

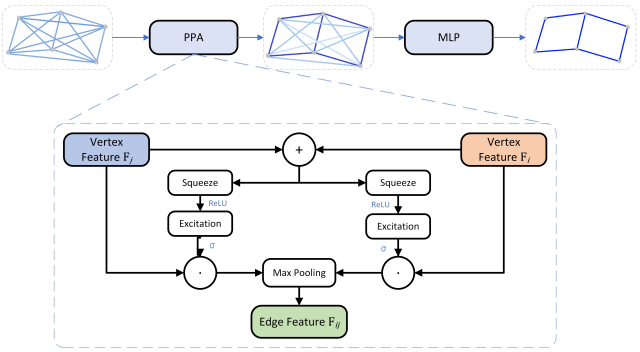

Figure 5. Procedure of edge prediction. First, we connect each pair of vertices to obtain all possible edges. Then, we use the PPA module to extract features for each edge. These features are fed into an MLP network, which learns the confidence score for the existence of each edge and produces the final result. The PPA module is presented in the lower part of the figure.

(PPA)[7] to accomplish this task. The structure is shown in Figure 5. First, we add the features of the two vertices to get $\mathbf{F}_{add}$. Then we input $\mathbf{F}_{add}$ into two independent MLP networks. The MLP network consists of Squeeze and Excitation parts, and respectively, we use ReLU and sigmoid functions as activation functions to generate the weight maps $\mathbf{w}_i$ and $\mathbf{w}_j$ corresponding to $\mathbf{F}_i$ and $\mathbf{F}_j$. Then we multiply $\mathbf{F}_i$ and $\mathbf{w}_i$, $\mathbf{F}_j$ and $\mathbf{w}_j$. At last, we perform max pooling on the results to obtain the final edge feature $\mathbf{F}_{ij}$. According to $\mathbf{F}_{ij}$, we can make the binary classification to decide which edges to be reserved. The specific formula is as follows:

$$\mathbf{w}_i = \sigma(\mathbf{W}_e^i \delta(\mathbf{W}_s^i(\mathbf{F}_i + \mathbf{F}_j))) \tag{7}$$

$$\mathbf{w}_j = \sigma(\mathbf{W}_e^j \delta(\mathbf{W}_s^j(\mathbf{F}_i + \mathbf{F}_j))) \tag{8}$$

$$\mathbf{F}_{ij} = \max(\mathbf{w}_i \odot \mathbf{F}_i, \mathbf{w}_j \odot \mathbf{F}_j) \tag{9}$$

where $\mathbf{W}_s, \mathbf{W}_e$ are MLPs of Squeeze part and Excitation part respectively. δ denotes ReLU function and σ denotes sigmoid function. $\odot$ refers to Hadamard product.

3.5. Loss Function

In this section, we introduce the loss function of Contour2Roof. The loss function of our model consists of stepwise losses, which are described in detail below. The total loss L consists of two partial losses L_{vtx} and L_{edge}. As mentioned in the structure section, L_{vtx} consists of L_{cls}, L_{reg}, and L_{ref}, with specific formulas as follows:

$$L = L_{vtx} + L_{edge} \tag{10}$$

$$L_{vtx} = L_{cls} + L_{reg} + \alpha L_{clus} \tag{11}$$

where L_{cls} denotes loss of classification, L_{reg} denotes loss of the regression procedure in which we adjust the position of candidate points, L_{clus} denotes loss of clustering, and α

is a hyperparameter to control weight, we set $\alpha = 1$ in this paper.

$$L_{cls} = \sum_{\mathbf{p}} L_{focal}(s_{\mathbf{p}}, l_{cls}(\mathbf{p})) \tag{12}$$

$$L_{reg} = L_{ref} = \frac{1}{N_{pos}} \sum_{\mathbf{p}} L_{smooth-L1}(\Delta\hat{\mathbf{p}}, \Delta_{reg}(\mathbf{p})) \cdot s_{\mathbf{p}} \tag{13}$$

where $s_{\mathbf{p}}, l_{cls}(\mathbf{p})$ denote the predicted classification value and the label of each point $\mathbf{p}$. $\Delta\hat{\mathbf{p}}, \Delta_{reg}(\mathbf{p})$ denote the predicted offset valve and the ground truth offset value of each point $\mathbf{p}$. N_{pos} is the number of positive points.

$$L_{edge} = \sum_{\mathbf{e}} L_{bce}(s_{\mathbf{e}}, l_{edge}(\mathbf{e})) \tag{14}$$

where $s_{\mathbf{e}}, l_{edge}(\mathbf{e})$ denote the predicted confidence value and the label of each edge $\mathbf{e}$.

4. Experiment

4.1. Related Datasets

Building3D is a processed data set formed from raw data provided by the Land Commission of the Republic of Estonia. The Building3D dataset includes 160,000 building point clouds, roof point clouds, grid models, and wireframing models with corresponding grid and wireframing models. The largest city, Tallinn, includes about 37,000 labeled architectural objects. The Building3D dataset covers 16 cities with an area of approximately 998 square kilometers. The order of magnitude of the data concentration point and the object is represented by the symbols M (millions) and K (thousands), respectively. The average memory consumption for building point clouds was between 83.89KB and 672.97KB, while the corresponding grid and wireframe models consumed less than 6.5KB and 2 KB. The average size ratio between building point cloud, grid, and wireframe models is about 400:4:1.

RoofN3D designed for three-dimensional architectural reconstruction. There will be poor mesh model quality and a large number of shape and vertex mismatches between the point cloud and the mesh model. The end-to-end 3D building roof modeling RoofN3D dataset based on airborne Lidar point clouds builds a small real-world dataset on top of the simulated dataset by selecting 500 roof point clouds and manually creating a grid model. However, both real-world and simulated datasets consist of very few classes of raw shapes, which do not exhibit considerable variability.

4.2. Evaluation metrics

In point cloud processing and 3D model generation, one of the most important metrics is the Average Corner Offset (ACO). Corners typically represent significant geometric features of buildings and are fundamental for constructing accurate 3D models. Therefore, precise corner detection is crucial for the subsequent generation of 3D models, highlighting the importance of ACO. Additionally, 3D Mesh IoU (Intersection over Union) measures the degree of match between the generated 3D model and the real model, serving as a vital metric for evaluating the overall accuracy and quality of the model. However, 3D Mesh IoU only reflects the overall fit. To capture the error at each point, we also use RMSE (Root Mean Square Error) to ensure the accuracy of each point prediction.

Furthermore, CP (Corner Precision), EP (Edge Precision), CR (Corner Recall), ER (Edge Recall), and F1 Score are also essential metrics. CP and CR evaluate the performance of our algorithm in detecting building corners, while EP and ER ensure the algorithm accurately detects building edges. The F1 Score balances precision and recall to provide a single performance measure. The combined use of these metrics allows for a comprehensive assessment of the reconstruction algorithm's performance.

4.3. Training

During the training process, we used 8 NVIDIA V100 16G GPUs. We divided the Tallinn dataset into a training set and a test set according to the Building3d[16]. The dataset contains a total of 36,000 buildings, of which the test set consists of 3,472 data and the other buildings are the training set. To efficiently utilize computational resources, we used a batch size of 16 samples per GPU. This configuration allows for efficient parallelization of training on all 8 GPUs, thus facilitating fast convergence of the model during training.

4.4. Results on Building3D

In this section, we show the model's excellent performance and efficient inference speed on the test set, and we train and validate on Tallinn dataset of Building3d. We show our results against the baseline model Point2Roof in Table 1. Our model improves 0.02 of ACO and the accuracy of key point prediction by 29% over the baseline model.

Since we downsampled the process to use fewer points, this greatly simplified the computational overhead of our subsequent process. We executed the inference process and recorded the inference time in an equivalent environment configuration on a V100 GPU, you can see the results of our comparative experiments in Table 2, our inference is more than three times as fast as using all points, but our point accuracy and edge accuracy are improved by 29% and 19% respectively.

4.5. Ablation Studies

In this section, we describe our motivation for designing two innovative modules, ESE and Feature Amplifier, in

Table 1. Results on Building3d

Methods	ACO	CP	CR	C-F1	EP	ER	E-F1
PointNet[11]	0.36	0.71	0.50	0.59	0.81	0.26	0.39
PointNet++[12]	0.34	0.79	0.52	0.63	0.84	0.33	0.47
RandLA-Net[5]	0.35	0.70	0.60	0.65	0.67	0.16	0.25
DGCNN[17]	0.32	0.73	0.58	0.65	0.81	0.30	0.40
PAConv[19]	0.33	0.75	0.57	0.65	0.85	0.31	0.45
Point2Roof[7]	0.30	0.66	0.48	0.56	0.71	0.26	0.38
OURS	**0.28**	**0.95**	**0.66**	**0.78**	**0.90**	**0.48**	**0.62**

our model and designing experiments to prove the effectiveness of our proposed modules, where we are performing the same training process and testing that, the features of these points are more desired when constructing the wireframe model.

Our ESE module goes through an edgeconv module that processes all the points and then performs a biased downsampling operation on the points close to the contour, and the processed portion of the point cloud is subjected to three new edgeconv modules to obtain more contour-focused features. A in Table 3 is the result of the test using only the ESE module, and our ESE module significantly improves metrics such as the accuracy of the test results.

We designed this Feature Amplifier module to increase feature differentiation from global point cloud. On the one hand, the global attention can help the point cloud to learn relevant features at long distances, on the other hand, we use the same attention layer recursively to enhance the feature differentiation. B in Table 3 shows the test results using the Feature Amplifier module alone, and A+B shows our final model with the best performance.

Table 2. Time and Efficiency Analysis

Methods	times	CP	EP
Point2Roof[7]	2.1s	0.66	0.71
OURS	**0.6s**	**0.95**	**0.90**

Table 3. Ablation Results

Methods	CP	EP
A	0.88	0.83
B	0.78	0.79
A+B	**0.95**	**0.90**

5. Conclusion

This paper presents a novel approach to the problem of constructing 3D wireframe models of buildings, namely the Contour2Roof model. With the increasing importance of point cloud data in computer vision, the method proposed in this paper aims to achieve more accurate wireframe model generation by efficiently extracting structural features from point clouds. First, we propose an efficient point cloud structural feature extractor (ESE) to generate more accurate wireframe models using fewer points by downsampling the point cloud with biased contours and learning more structurally significant information from the point cloud in a targeted manner. Second, we design a global attention-based Feature Amplifier for enriching the acquired features to better distinguish points at different locations. The experimental results show that our proposed Contour2Roof model achieves remarkable results and provides new ideas and methods for the field of structural learning of point cloud data. The successful application of this method provides a powerful tool and reference for solving the problem of 3D model construction for complex scenes such as buildings, which has important theoretical and application value.

References

[1] Syeda Mariam Ahmed, Yan Zhi Tan, Chee Meng Chew, Abdullah Al Mamun, and Fook Seng Wong. Edge and corner detection for unorganized 3d point clouds with application to robotic welding. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, page 7350–7355. IEEE Press, 2018. 3

[2] Mohammad Awrangjeb, Syed Ali Naqi Gilani, and Fasahat Ullah Siddiqui. An effective data-driven method for 3-d building roof reconstruction and robust change detection. *Remote Sensing*, 10(10):1512, 2018. 2

[3] Dong Chen, Liqiang Zhang, P Takis Mathiopoulos, and Xianfeng Huang. A methodology for automated segmentation and reconstruction of urban 3-d buildings from als point clouds. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 7(10):4199–4217, 2014. 2

[4] Andre Henn, Gerhard Gröger, Viktor Stroh, and Lutz Plümer. Model driven reconstruction of roofs from sparse lidar point clouds. *ISPRS Journal of photogrammetry and remote sensing*, 76:17–29, 2013. 2

[5] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11108–11117, 2020. 7

[6] Martin Kada and Laurence McKinley. 3d building reconstruction from lidar based on a cell decomposition approach. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 38(Part 3):W4, 2009. 2

[7] Li Li, Nan Song, Fei Sun, Xinyi Liu, Ruisheng Wang, Jian Yao, and Shaosheng Cao. Point2roof: End-to-end 3d building roof modeling from airborne lidar point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2022. 1, 3, 5, 7

[8] Minglei Li, Franz Rottensteiner, and Christian Heipke. Modelling of buildings from aerial lidar point clouds using tins and label maps. *ISPRS Journal of Photogrammetry and Remote Sensing*, 154:127–138, 2019. 2

[9] Yujia Liu, Stefano D'aronco, Konrad Schindler, and Jan Dirk Wegner. Pc2wf: 3d wireframe reconstruction from raw point clouds. *ArXiv*, abs/2103.02766, 2021. 1, 3

[10] Charalambos Poullis and Suya You. Photorealistic large-scale urban city model reconstruction. *IEEE transactions on visualization and computer graphics*, 15(4):654–669, 2008. 2

[11] Charles R. Qi, Hao Su, Mo Kaichun, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 77–85, 2017. 3, 5, 7

[12] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5099–5108, 2017. 3, 5, 7

[13] Gunho Sohn, Xianfeng Huang, and Vincent Tao. Using a binary space partitioning tree for reconstructing polyhedral building models from airborne lidar data. *Photogrammetric Engineering & Remote Sensing*, 74(11):1425–1438, 2008. 2

[14] Jingwei Song, Jianwei Wu, and Yongyao Jiang. Extraction and reconstruction of curved surface buildings by contour clustering using airborne lidar data. *Optik*, 126(5):513–521, 2015. 2

[15] Vivek Verma, Rakesh Kumar, and Stephen Hsu. 3d building detection and modeling from aerial lidar data. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, pages 2213–2220. IEEE, 2006. 2

[16] Ruisheng Wang, Shangfeng Huang, and Hongxin Yang. Building3d: An urban-scale dataset and benchmarks for learning roof structures from point clouds. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 20019–20029, 2023. 6

[17] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (tog)*, 38(5):1–12, 2019. 3, 7

[18] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Trans. Graph.*, 38(5), 2019. 3

[19] Mutian Xu, Runyu Ding, Hengshuang Zhao, and Xiaojuan Qi. Paconv: Position adaptive convolution with dynamic kernel assembling on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3173–3182, 2021. 7

[20] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16239–16248, 2021. 4

[21] Yuanfan Zheng and Qihao Weng. Model-driven reconstruction of 3-d buildings using lidar data. *IEEE geoscience and remote sensing letters*, 12(7):1541–1545, 2015. 2