Delaunay Canopy: Building Wireframe Reconstruction from Airborne LiDAR Point Clouds via Delaunay Graph (Supplementary Material)

Donghyun Kim[1,2], Chanyoung Kim[1,2], Youngjoong Kwon[2⋆], and Seong Jae Hwang[1⋆]

[1] Yonsei University, Seoul, Republic of Korea
seongjae@yonsei.ac.kr
[2] Emory University, Atlanta, GA, USA
{donghyun.kim, chanyoung.kim, youngjoong.kwon}@emory.edu

A Further Implementation Specifics

A.1 Training Details for Corner and Wire Selection

To train our corner and wire selection modules, we formulate the objective function based on set prediction. Let $\mathbf{C}^*$ and $\mathcal{W}^*$ denote the set of ground truth 3D corners and wires, respectively. Correspondingly, corner selection module and wire selection module predict the set of corners $\tilde{\mathbf{C}}$ and the set of wires $\tilde{\mathcal{W}}$, respectively. The total loss $\mathcal{L}_{\text{total}}$ is composed of the corner loss $\mathcal{L}_{\text{corner}}$ and the wire loss $\mathcal{L}_{\text{wire}}$.

Corner Loss. Since the corner selection module outputs a set of predictions without intrinsic ordering, we first find the optimal permutation $\hat{\sigma}$ by performing a bipartite matching between the predicted corners $\tilde{\mathbf{C}}$ and the ground truth corners $\mathbf{C}^*$. Thus, we minimize the matching cost $\mathcal{L}_{\text{match}}$ using the Cross-Entropy (CE) and the Manhattan distance ($D_{\text{Manhattan}}$) as

$$\mathcal{L}_{\text{match}}(c_i^*, \tilde{c}_{\sigma(i)}) = \alpha \text{CE}(c_i^*, \tilde{c}_{\sigma(i)}) + \beta D_{\text{Manhattan}}(c_i^*, \tilde{c}_{\sigma(i)}), \tag{1}$$

where c_i^* is the i-th ground-truth corner, and $\tilde{c}_{\sigma(i)}$ the i-th predicted corner under the influence of the bipartite-matching permutation σ. The weights α and β are hyperparameters balancing two terms. Based on the optimal permutation $\hat{\sigma}$ that minimizes the matching cost, the *Corner Loss* ($\mathcal{L}_{\text{corner}}$) is computed as

$$\mathcal{L}_{\text{corner}} = \frac{1}{|\mathbf{C}^*|} \sum_{i=1}^{|\mathbf{C}^*|} \alpha \text{CE}(c_i^*, \tilde{c}_{\hat{\sigma}(i)}) + \beta D_{\text{Manhattan}}(c_i^*, \tilde{c}_{\hat{\sigma}(i)}). \tag{2}$$

⋆ Corresponding authors.

Wire Loss. The wire selection module serves to adjudicate the structural validity of every wire candidate presented. For the predicted set of wires $\tilde{\mathcal{W}}$, we compute the binary cross-entropy loss against the ground truth wires $\mathcal{W}^*$. The *Wire Loss* ($\mathcal{L}_{\text{wire}}$) is defined as:

$$\mathcal{L}_{wire} = \text{CE}(\mathcal{W}^*, \tilde{\mathcal{W}}). \tag{3}$$

This loss term penalizes the model for incorrect wire selections, ensuring that the topological structure of the building is accurately reconstructed.

Total Loss. Finally, the total objective function is a weighted sum of the aforementioned losses:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{corner}} + \gamma \mathcal{L}_{\text{wire}}, \tag{4}$$

where γ is balancing coefficients for the corner and wire losses.

A.2 Details of Delaunay Graph Scoring

Delaunay Triangulation.

Algorithm 1 Delaunay Triangulation via Incremental Insertion

Input : Set of points $V = \{v_1, \ldots, v_N\}$
Output: Delaunay Triangulation $\mathcal{T}$ (Set of Simplices)
Initialize $\mathcal{T}$: Create an initial large bounding simplex S_{init} that contains all points in V
for $i = 1$ **to** N **do**
  Let v_i be the point to insert **if** v_i *is already a vertex of* $\mathcal{T}$ **then**
    continue
  Find Conflict Region: Identify the set of simplices $\mathcal{C} \subset \mathcal{T}$ whose circumcircles contain v_i
  if $\mathcal{C} = \emptyset$ **then**
    continue
  Compute Boundary: Determine the boundary $\partial\mathcal{C}$ of the conflict region $\mathcal{C}$
  Remove Conflict Simplices: $\mathcal{T} \leftarrow \mathcal{T} \setminus \mathcal{C}$
  Create New Simplices: For each edge $\mathbf{e} \in \partial\mathcal{C}$ on the boundary, create a new simplex S_{new} using $\mathbf{e}$ and the new point v_i
  $\mathcal{T} \leftarrow \mathcal{T} \cup \{S_{\text{new}} \mid \mathbf{e} \in \partial\mathcal{C}\}$
  Optional Optimization (Flip Test): For each new simplex S_{new} and its neighbors, perform a local check to ensure the Delaunay condition is maintained (e.g., edge flipping)
Remove Bounding Simplex: Delete all simplices connected to the initial bounding simplex S_{init} vertices from $\mathcal{T}$
return $\mathcal{T}$

Delaunay graph scoring is a method that is predicated on receiving an airborne LiDAR point cloud of a scanned structure. This process begins with the requisite

application of *Delaunay Triangulation* to this input point cloud. The Delaunay Triangulation is a fundamental tessellation that maximizes the minimum angle of all triangles in the partitioning, thereby ensuring mesh quality. The core principle adheres to the empty circumsphere criterion: for any simplex in the triangulation, the circumsphere defined by its vertices must contain no other points from the point cloud. In this work, the Delaunay Triangulation is computed using the `scipy.spatial.Delaunay` implementation, which relies on the established `Qhull` library. The detailed procedure can be found in Algorithm 1. Following this algorithm, the resulting set of simplices $\mathcal{T}$ is then approached from a graph-theoretic perspective, allowing us to extract the geometric information for each constituent element from the resultant Delaunay graph $G = (V, E, F)$.

Wire-Wise Path Score. To guide the discrimination of wire candidates based on their likelihood of validity within the wire selection module, we compute the *wire-wise path score*, a metric reflective of the candidate's structural validity. In a more detailed fashion, for a given wire candidate w_{ij}, we first determine the shortest path, $P^G_{i\to j}$, which interconnects its two endpoints, v_i and v_j, within the Delaunay graph $G = (V, E, F)$. The final metric is then derived by aggregating the dihedral angles of all edges that lie along this designated path. Crucially, this shortest path is elucidated using Dijkstra's algorithm [1] on the Delaunay graph $G = (V, E, F)$, where the path cost is parameterized by the actual Euclidean distance between the constituent points.

B Additional Evaluation Results

To further substantiate the efficacy of *Delaunay Canopy* for building wireframe reconstruction, we provide supplementary evaluation experiments and accompanying analysis. The comparative results on the official Building3D leaderboard are presented in Sec. B.1, and the quantitative results derived from K-fold cross-validation are fully detailed within Sec. B.2. To address potential dataset biases, an evaluation on the Tokyo LoD2 dataset [11] is detailed in Sec. B.3. To evaluate the robustness under domain shift, a cross-dataset generalization study is presented in Sec. B.4. Furthermore, an expanded collection of wireframe outputs and their corresponding qualitative analyses are presented in Sec. B.5.

B.1 Performance on the Official Building3D Leaderboard

To ensure a rigorous and standardized comparison, we evaluate our framework according to the official Building3D [11] leaderboard protocol. As shown in Tab. 1, we compare *Delaunay Canopy* with various state-of-the-art methods and established baselines based on the leaderboard. Our framework not only surpasses the methods utilized as feature extractors (denoted with †) but also demonstrates superior structural fidelity compared to recent leading approaches such as PBWR [2] and BWFormer [6]. This confirms that the geometric prior embedded within the Delaunay Canopy provides a generalized, state-of-the-art capability in wireframe reconstruction under official benchmark standards.

Table 1: Performance evaluation based on the official Building3D [11] Tallinn city dataset leaderboard. Baseline metrics are sourced from the public leaderboard. The notation `MODEL`† designates models acting exclusively as the feature extraction backbone within the standard Building3D [11] architecture. Best, second best, and third best are highlighted.

Method	Distance		Corner			Wire		
	WED ↓	ACO ↓	CP ↑	CR ↑	CF1 ↑	EP ↑	ER ↑	EF1 ↑
PointMAE† [8]	-	0.330	75.0	47.0	58.0	52.0	12.0	20.0
PointM2AE† [12]	-	0.320	79.0	58.0	67.0	50.0	7.0	12.0
Point2Roof [3]	-	0.390	65.0	30.0	41.0	66.0	8.0	14.0
Linear self-supervised [11]	-	0.350	70.0	60.0	65.0	67.0	16.0	25.0
Supervised [11]	-	0.290	90.0	53.0	66.0	88.0	23.0	36.0
PC2WF [5]	-	0.520	18.0	67.0	28.0	2.0	15.0	1.0
PBWR [2]	0.271	0.222	98.5	68.8	81.0	94.3	65.4	77.2
BWFormer [6]	0.238	0.204	94.9	82.7	88.4	85.5	74.1	79.4
Delaunay Canopy	0.225	0.206	95.7	83.9	89.4	87.6	75.4	81.1

B.2 *K*-Fold Cross-Validation

Table 2: Robustness validation via 5-Fold Cross-Validation. The notation `MODEL`* signifies a model acting as the feature extractor within the Point2Roof [3] pipeline. Best and second best are highlighted.

Method	Distance		Corner			Edge		
	WED ↓	ACO ↓	CP ↑	CR ↑	CF1 ↑	EP ↑	ER ↑	EF1 ↑
Building3D (PointNet) [9]	0.271	0.245	87.8	77.9	82.5	80.8	71.5	75.9
PC2WF [5]	0.548	0.512	21.9	50.1	30.4	3.1	16.5	5.3
Point2Roof [3]	0.265	0.239	89.0	78.2	83.3	81.5	71.8	76.3
PointTransformer* [13]	0.260	0.242	89.4	78.8	83.8	81.7	72.0	76.5
PointMLP* [7]	0.258	0.236	90.3	79.3	84.4	82.1	72.5	77.0
PointNeXt* [10]	0.259	0.232	90.7	79.7	84.8	82.5	73.0	77.5
PointMeta* [4]	0.254	0.228	91.0	80.2	85.3	82.9	73.5	77.9
BWFormer [6]	0.248	0.217	91.8	80.5	85.8	84.5	74.3	79.1
Delaunay Canopy	0.250	0.199	93.8	79.9	86.2	86.8	74.5	80.2

While our initial evaluation utilizes a conventional random train-test split, this approach may risks introducing statistical bias and may fail to fully capture the performance generalization capacity of the model. To ensure the robustness and statistical reliability of our results, we adopt 5-fold cross-validation on the combined training and test sets. This procedure systematically partitions the dataset into five equal subsets, where each subset serves as the validation set exactly once, mitigating the sensitivity of the final performance metric to any single, arbitrary data split. Tab. 2 attests to the fact that the Delaunay Canopy exhibits a distinct and favorable performance advantage over other competing methodologies.

B.3 Evaluation on Tokyo LoD2

Table 3: Quantitative comparisons on the Tokyo LoD2 dataset. Models are trained and evaluated from scratch. The notation `MODEL`* signifies a model acting as the feature extractor within the Point2Roof [3] pipeline.

Method	Distance		Corner			Edge		
	WED ↓	ACO ↓	CP ↑	CR ↑	CF1 ↑	EP ↑	ER ↑	EF1 ↑
PointMeta* [4]	0.267	0.244	88.5	77.2	82.5	80.1	72.3	76.0
BWFormer [6]	0.262	0.229	91.2	79.6	85.0	83.8	72.8	77.9
Ours	**0.256**	**0.208**	**93.5**	**81.2**	**86.9**	**86.4**	**73.1**	**79.2**

To provide a more robust validation and address potential concerns regarding the label quality of the Building3D dataset, we evaluate our framework on the recently released Tokyo LoD2 dataset [11], the only other publicly available benchmark. We train and evaluate all models from scratch on 29,853 samples (27,000 for training and 2,853 for testing). As shown in Tab. 3, our framework consistently outperforms baselines on this more reliable and complex dataset, demonstrating that the performance gains of Delaunay Canopy are robust and not merely artifacts of dataset-specific biases.

B.4 Cross-Dataset Generalization

Table 4: Cross-dataset generalization performance on 1,000 Tokyo LoD2 test samples using models trained on the Building3D Tallinn dataset. The notation `MODEL`* signifies a model acting as the feature extractor within the Point2Roof [3] pipeline.

Method	Distance		Corner			Edge		
	WED ↓	ACO ↓	CP ↑	CR ↑	CF1 ↑	EP ↑	ER ↑	EF1 ↑
PointMeta* [4]	0.298	0.276	84.1	72.3	77.8	76.5	66.4	71.1
BWFormer [6]	0.282	0.254	86.4	75.8	80.8	79.1	**72.2**	75.5
Ours	**0.274**	**0.241**	**88.2**	**77.5**	**82.5**	**81.2**	71.8	**76.2**

Moving beyond evaluations restricted to the Building3D dataset, we further evaluate the cross-dataset generalization capacity of our framework. Specifically, models trained solely on the Building3D Tallinn dataset were directly tested on 1,000 randomly sampled Tokyo LoD2 [11] samples without any fine-tuning. As shown in Tab. 4, our framework consistently outperforms both PointMeta* (serving as a Point2Roof backbone) and the recent state-of-the-art BWFormer. This confirms that the geometric prior embedded within Delaunay Canopy preserves strong structural guidance, demonstrating robust generalization despite the significant cross-dataset distribution shift.

B.5 Additional Qualitative Results

We provide supplementary visualizations of the outcomes wherein airborne LiDAR building point clouds are transformed into a wireframe via *Delaunay Canopy*. Figs. 3 to 5 attest to the framework's capacity to competently and robustly reconstruct the wireframe across a spectrum of challenging scenarios, including those marked by acute sparsity, pervasive noise, and point clouds exhibiting subtle curvature. Moreover, a clear advantage is demonstrated by Delaunay Canopy in its capacity to accurately reconstruct the internal corner features, an undertaking that the strongest baseline, BWFormer [6], is unable to accomplish with a commensurate level of precision.

C Additional Analyses

C.1 Computational Speed of Delaunay Graph Scoring

To proactively mitigate potential concerns regarding the computational overhead of our core approach, *Delaunay graph scoring*, the empirical processing time on the Building3D dataset [11] is rigorously measured and presented. When evaluated on the Building3D Tallinn city train dataset, the mean processing time per point cloud is a mere **0.367 seconds**. This throughput translates to the ability to execute the Delaunay graph scoring on approximately **2.7 point clouds per second**. This performance firmly establishes the approach as a highly efficient and computationally meritorious strategy, especially when factoring in that the 3D building point clouds in the dataset typically encompass several thousand, often exceeding ten thousand, constituent points.

C.2 Robustness in Challenging Scenarios.

Table 5: Robustness validation assessed against a perturbed dataset characterized by augmented sparsity and noise. The notation `MODEL`* signifies a model acting as the feature extractor within the Point2Roof [3] pipeline.

Method	Distance		Corner			Edge		
	WED ↓	ACO ↓	CP ↑	CR ↑	CF1 ↑	EP ↑	ER ↑	EF1 ↑
Building3D (PointNet) [9]	0.407	0.354	78.1	68.9	73.2	72.6	64.2	68.2
PC2WF [5]	0.658	0.612	14.2	38.7	20.8	1.6	10.4	2.7
Point2Roof [3]	0.384	0.341	79.5	70.3	74.6	73.9	65.4	69.4
PointTransformer* [13]	0.367	0.325	80.8	71.5	75.9	75.2	66.5	70.6
PointMLP* [7]	0.359	0.318	81.3	72.0	76.4	75.6	67.1	71.1
PointNeXt* [10]	0.355	0.311	82.1	72.8	77.2	76.3	68.6	72.3
PointMeta* [4]	0.348	0.302	83.5	73.6	78.3	77.1	69.3	73.1
BWFormer [6]	0.339	0.291	84.4	74.5	79.1	78.7	70.1	74.2
Delaunay Canopy	0.301	0.263	88.2	78.1	82.8	80.6	71.3	75.7

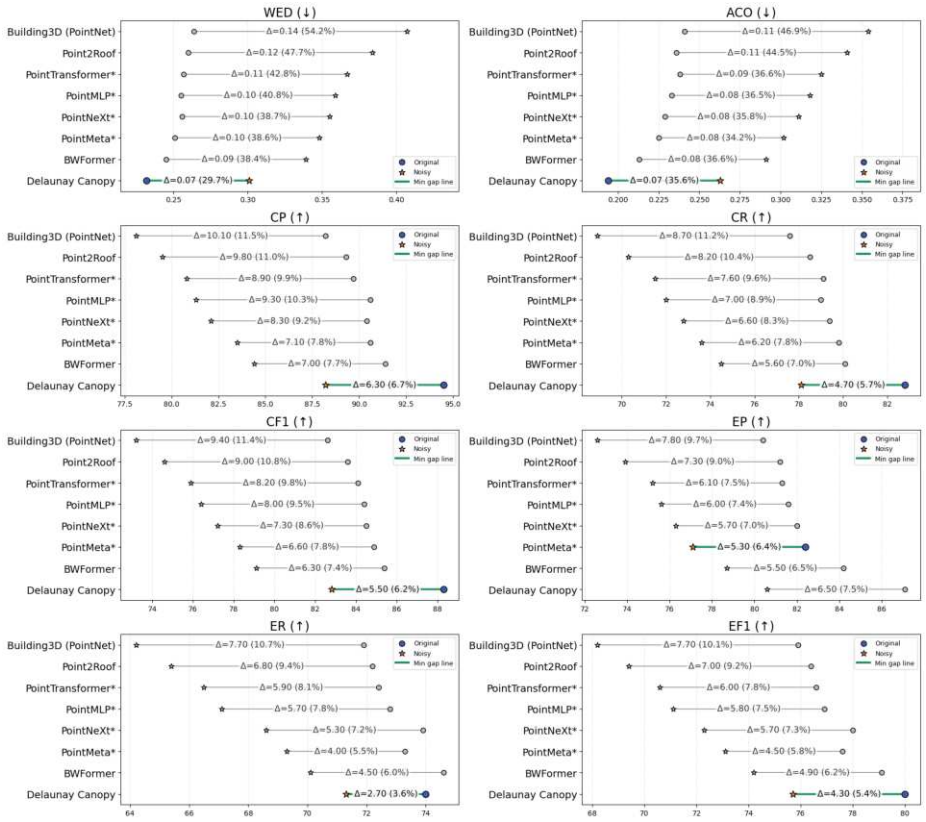

Fig. 1: Visualization of performance differences between challenging (perturbed point cloud) and original inputs. The number next to Δ indicates the absolute performance difference, and the percentage (%) shows the relative decrease compared to the original performance. The notation `MODEL`* signifies a model acting as the feature extractor within the Point2Roof [3] pipeline.

We compare the robustness of our framework, *Delaunay Canopy*, against the baselines [3–7, 9, 10, 13] introduced in the main paper by creating a more challenging test set than the existing Building3D [11] Tallinn city and entry-level datasets. For this evaluation, we utilize a custom test set split from the Tallinn city dataset, as mentioned in the main paper, but drastically increase its sparsity and noise compared to the original. To induce severe localized sparsity across all point clouds, we employ a controlled removal strategy: in each point cloud, a single point was randomly selected, and this point, along with its nearest neighbors constituting 5% of the total points in that cloud, was subsequently removed. Furthermore, to ensure a globally noisy state in the input point clouds, we add Gaussian noise to every point cloud within the test set.

As evinced by Fig. 1 and Table 5, the Delaunay Canopy exhibits a pronounced advantage over other baselines concerning robustness. This superior performance is attributable to its methodology: by gleaning the intrinsic information of the point cloud's underlying surface through the Delaunay graph, it facilitates geometric extrapolation even into zones lacking explicit data points. This very mechanism, which allows the framework to discern the overarching structural trend from local curvature despite the presence of stochastic perturbations, is what ultimately enables a resilient and robust wireframe reconstruction.

Table 6: Robustness validation under varying point density (FPS subsampling to 90% and 70%). The notation `MODEL`* signifies a model acting as the feature extractor within the Point2Roof [3] pipeline.

Method	FPS	Distance		Corner			Edge		
		WED ↓	ACO ↓	CP ↑	CR ↑	CF1 ↑	EP ↑	ER ↑	EF1 ↑
PointMeta* [4]	90%	0.302	0.278	84.1	73.2	78.3	76.1	66.5	71.0
	70%	0.352	0.324	79.4	68.8	73.7	72.2	63.1	67.3
BWFormer [6]	90%	0.292	0.263	85.6	74.2	79.5	77.5	68.2	72.6
	70%	0.339	0.304	81.3	70.5	75.5	74.6	64.8	69.4
Ours	90%	**0.281**	**0.245**	**88.4**	**76.5**	**82.0**	**80.2**	**68.4**	**73.8**
	70%	**0.316**	**0.281**	**84.6**	**73.2**	**78.5**	**76.5**	**65.7**	**70.7**

Furthermore, to evaluate the sensitivity to varying point density, we conduct an additional robustness study where the input point clouds are subsampled to 90% and 70% of their original density using Farthest Point Sampling (FPS). As shown in Tab. 6, across all density levels and perturbations, our framework consistently outperforms baselines, even though the geometric prior itself is also distorted by the corrupted input.

D Discussion

D.1 Limitations

While *Delaunay Canopy* exhibits robust performance across diverse scenarios, it is not without its limitations, presenting opportunities for refinement. The overall pipeline is comprised of the corner selection stage and the wire selection stage, each employing a dedicated mechanism to "*select*" valid corners and wires from their respective candidates. Consequently, a critical challenge emerges when the input LiDAR point cloud fails to capture the requisite valid corner and wire features, *i.e.*, when crucial critical points are omitted during scanning and the corresponding geometric region is severely sparse. Under such conditions, the pipeline's capacity to effectively detect both corners and wires is significantly diminished. A direct inspection of Fig. 2(a) intuitively demonstrates that in instances where regions where a valid corner is structurally mandated remain unscanned and are consequently absent from the point cloud, the corner selection

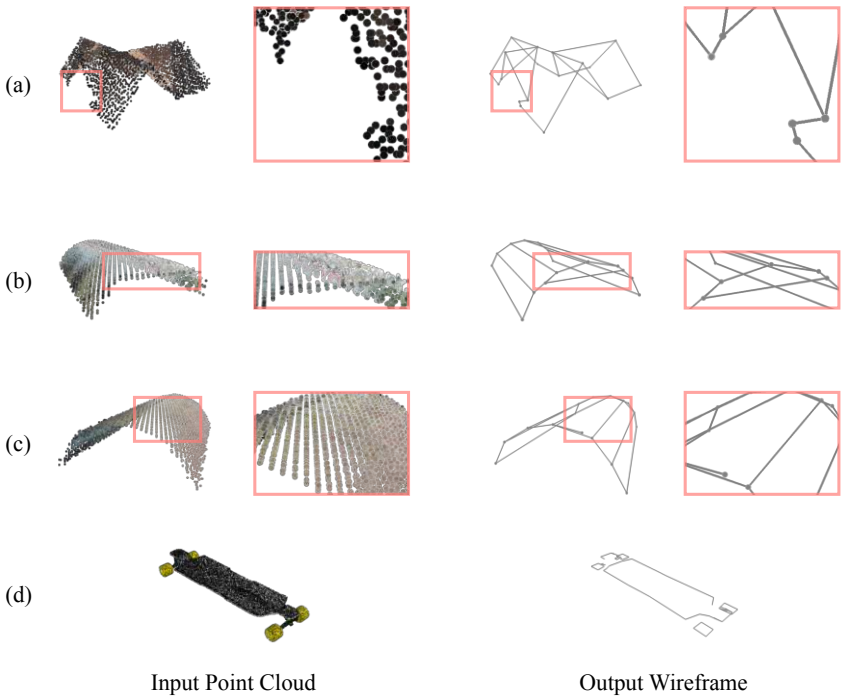

Fig. 2: Visualization of failure cases. Precise wireframe reconstruction is demonstrably compromised under several key scenarios. (a) When critical regions containing valid corner and wire features are unscanned and consequently absent from the point cloud. (b & c) When the intrinsic surface topology of the point cloud exhibits excessive smoothness. (d) When processing objects characterized by a closed topology, such as non-building volumetric structures.

mechanism fails to correctly identify the requisite feature during the wireframe reconstruction pipeline. This failure, in turn, results in the resultant wireframe exhibiting suboptimal quality.

A second limitation arises when the intrinsic surface of the point cloud is devoid of sharp angular features, rendering the identification of true wireframe corners and wires inherently ambiguous. In such circumstances, the Delaunay Canopy framework is similarly hindered in its capacity for precise wireframe reconstruction. Fig. 2(b) and Fig. 2(c) visually confirms that in regions characterized by rounded surfaces, the framework experiences erroneous corner selection and wire selection.

The final and most principal methodological constraint is the fundamental assumption that Delaunay graph scoring requires input point clouds to be limited to topologically open structures (*e.g.*, non-closed surfaces like building rooftops). This is because the method strictly requires that only a single z-coordinate exists for any given ground coordinate (x, y). Consequently, our methodology cannot

be directly or effectively transposed to point clouds captured from topologically closed (volumetric) objects (Fig. 2(d)).

D.2 Future Works

Delaunay Canopy constitutes a framework engineered for the reconstruction of wireframes from 3D building point clouds. Its efficacy stems from utilizing the core approach, *Delaunay graph scoring*, which first yields a robust approximation of the intrinsic surface of the input cloud, then subsequently extracts local curvature information. This information is utilized as a geometric prior to provide critical guidance to the reconstruction pipeline. Crucially, this approach is not solely confined to the architectural domain; it is highly extensible to diverse subjects encountered in autonomous driving and interior scanning applications. Consequently, our primary avenue for future investigation is the generalization of this approach across these disparate fields, culminating in the development of a truly universal wireframe reconstruction framework.

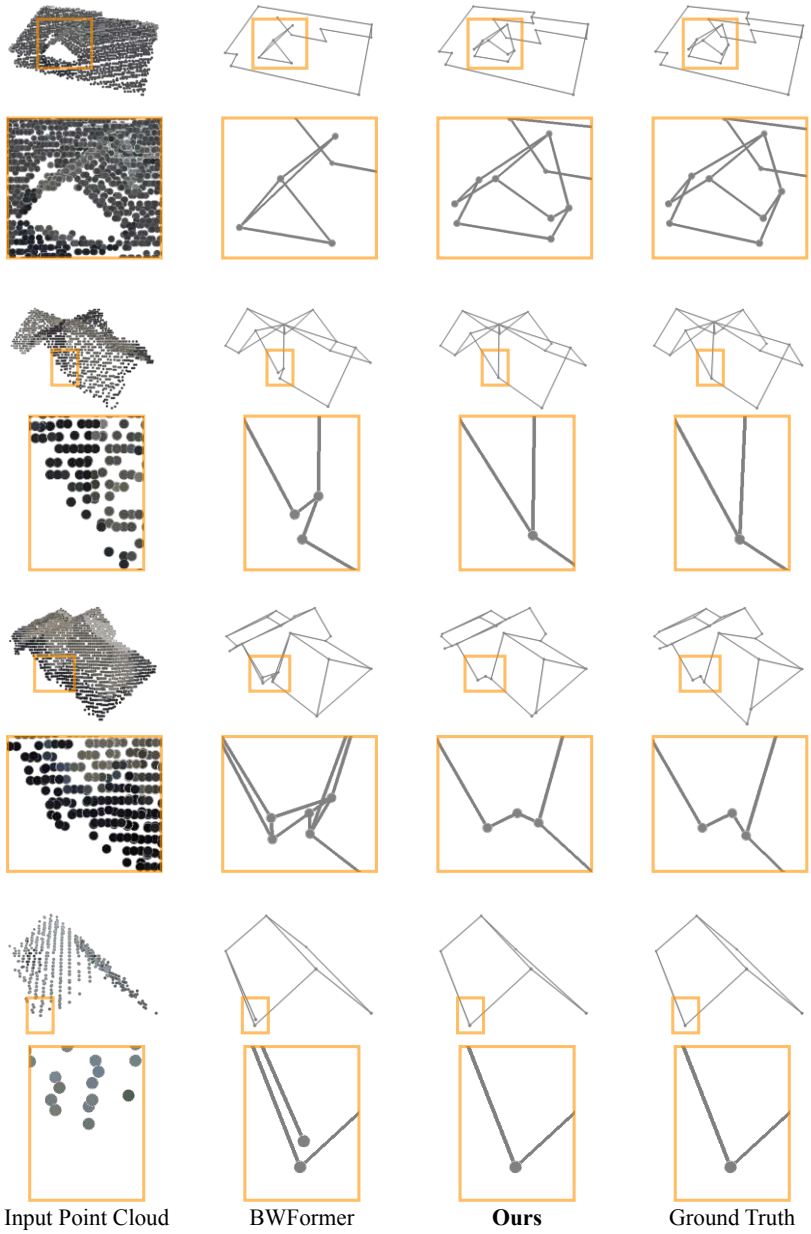

Fig. 3: Qualitative comparison with the strongest baseline BWFormer [6]. The orange bounding boxes specifically denote the regions where our method achieves a demonstrably superior reconstruction result.

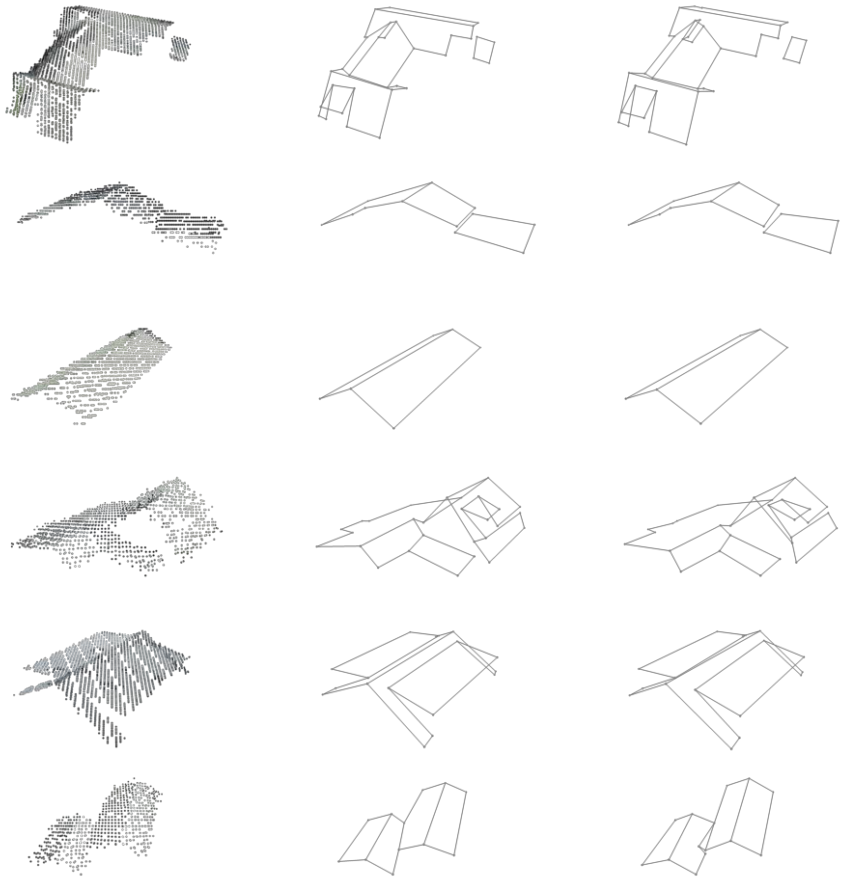

Fig. 4: Wireframe reconstruction results of Delaunay Canopy on Building3D dataset.

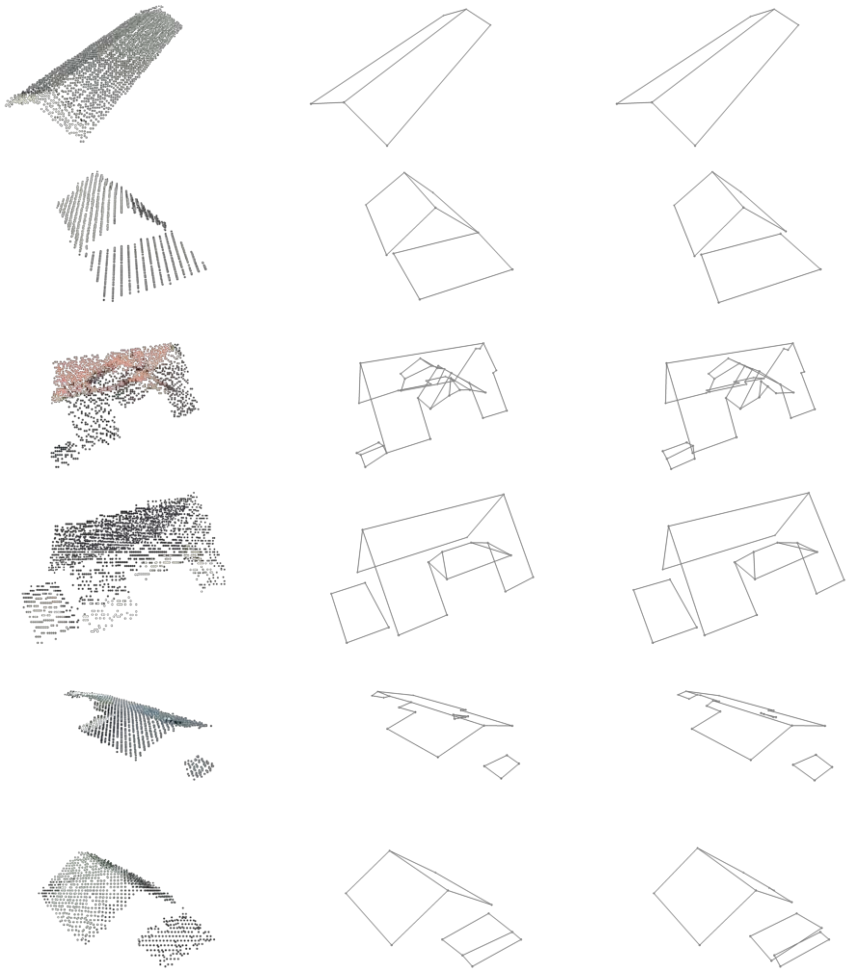

Fig. 5: Wireframe reconstruction results of Delaunay Canopy on Building3D dataset.

References

1. Dijkstra, E.W.: A note on two problems in connexion with graphs. In: Edsger Wybe Dijkstra: his life, work, and legacy, pp. 287–290 (2022)
2. Huang, S., Wang, R., Guo, B., Yang, H.: Pbwr: Parametric-building-wireframe reconstruction from aerial lidar point clouds. In: CVPR (2024)
3. Li, L., Song, N., Sun, F., Liu, X., Wang, R., Yao, J., Cao, S.: Point2roof: End-to-end 3d building roof modeling from airborne lidar point clouds. ISPRS Journal of Photogrammetry and Remote Sensing (2022)
4. Lin, H., Zheng, X., Li, L., Chao, F., Wang, S., Wang, Y., Tian, Y., Ji, R.: Meta architecture for point cloud analysis. In: CVPR (2023)
5. Liu, Y., D'Aronco, S., Schindler, K., Wegner, J.D.: Pc2wf: 3d wireframe reconstruction from raw point clouds. arXiv preprint arXiv:2103.02766 (2021)
6. Liu, Y., Zhu, L., Ye, H., Huang, S., Gao, X., Zheng, X., Shen, S.: Bwformer: Building wireframe reconstruction from airborne lidar point cloud with transformer. In: CVPR (2025)
7. Ma, X., Qin, C., You, H., Ran, H., Fu, Y.: Rethinking network design and local geometry in point cloud: A simple residual mlp framework. arXiv preprint arXiv:2202.07123 (2022)
8. Pang, Y., Wang, W., Tay, F.E., Liu, W., Tian, Y., Yuan, L.: Masked autoencoders for point cloud self-supervised learning. In: Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part II. pp. 604–621. Springer (2022)
9. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: CVPR (2017)
10. Qian, G., Li, Y., Peng, H., Mai, J., Hammoud, H., Elhoseiny, M., Ghanem, B.: Pointnext: Revisiting pointnet++ with improved training and scaling strategies (2022)
11. Wang, R., Huang, S., Yang, H.: Building3d: A urban-scale dataset and benchmarks for learning roof structures from point clouds. In: CVPR (2023)
12. Zhang, R., Guo, Z., Gao, P., Fang, R., Zhao, B., Wang, D., Qiao, Y., Li, H.: Point-m2ae: multi-scale masked autoencoders for hierarchical point cloud pre-training. Advances in neural information processing systems **35**, 27061–27074 (2022)
13. Zhao, H., Jiang, L., Jia, J., Torr, P.H., Koltun, V.: Point transformer. In: ICCV (2021)