BuildingGPT: Auto-Regressive Building Wireframe Reconstruction Model with Reinforcement Learning

Supplementary Material

In the supplementary material, we first provide an illustration of the different tokenization schemes in Section 1. Then, we provide more details of the MunichWF dataset including the dataset construction pipeline, statistics, and visualizations in Section 2. Later, more details of the training process are presented in Section 3. Finally, we provide additional experimental results to demonstrate the effectiveness of our model design, including extended samples in the preference pair dataset, additional qualitative evaluations against other baselines, more visualizations for the effectiveness of post-training, and more results of the cross-data generation in Section 4.

1. Illustrations of the Sequence Orderings

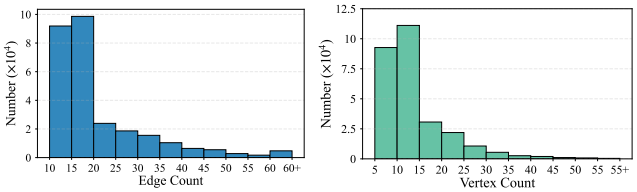
For the vanilla tokenization, vertices are first arranged in ascending z-y-x order. Then edges are sorted first by their lowest vertex index, followed by the next lowest one [10, 11]. For the hierarchical tokenization, the ordering is both structurally and semantically aware, following a footprint-to-roof progression. An example of different building wireframe sequence orderings is presented in Figure 3.

2. Details of the MunichWF Dataset

Dataset Construction Pipeline. We construct our MunichWF dataset based on the Munich City dataset [1] which consists of the LOD2 building meshes and the paired point clouds. All the meshes and point clouds are first normalized to be centered at the origin and scaled to the range of [-0.5, 0.5]. We then apply planar decimation to the LOD2 mesh models with an angle tolerance parameter of 0.175 to transform the triangle meshes into planar meshes. Later, we merge nearby vertices within a threshold of 0.01. After that, we extract the building wireframes from the meshes. After filtering the samples with broken structures, the final dataset is obtained. All the operations are performed in Blender [2].
Statistics and visualizations of MunichWF dataset. The MunichWF dataset exhibits diverse structures. The count distributions of edges and vertices are presented in Figure 1 and the visualizations of some examples along with their corresponding point clouds are shown in Figure 2.

3. Training Details

For the preference pair dataset, the top-m samples with the largest score differences are selected, where m is set to 40000 in our experiments. During pre-training, we use the AdamW optimizer [9] with a cosine-annealed learning rate decreasing from 1×10^{-5} to 1×10^{-6}. For the post-training, we use a learning rate of 1×10^{-6} and train the model for 10 epochs. For the baselines, we retrain them on the MunichWF dataset with their default settings for a fair comparison.



(a) Edge count distribution (b) Vertex count distribution

Figure 1. **Edge and vertex count distributions in the MunichWF dataset.** This figure presents the edge (a) and vertex (b) count distributions in the MunichWF dataset.

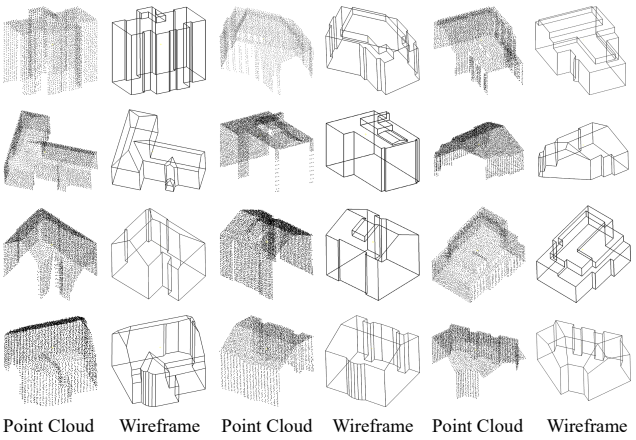


Figure 2. **Visualizations of MunichWF dataset.** Examples of point clouds and their corresponding wireframes are presented.

4. More Experiment Results

In this section, additional experimental results are presented, including extended samples from the preference-pair dataset (*cf.* Fig. 4), further qualitative evaluations (*cf.* Fig. 5), more visualizations demonstrating the effectiveness of post-training (*cf.* Fig. 6), and more results of the cross-data generation (*cf.* Fig. 7). The comprehensive ablation studies further demonstrate the effectiveness of our model design and confirm its state-of-the-art performance.

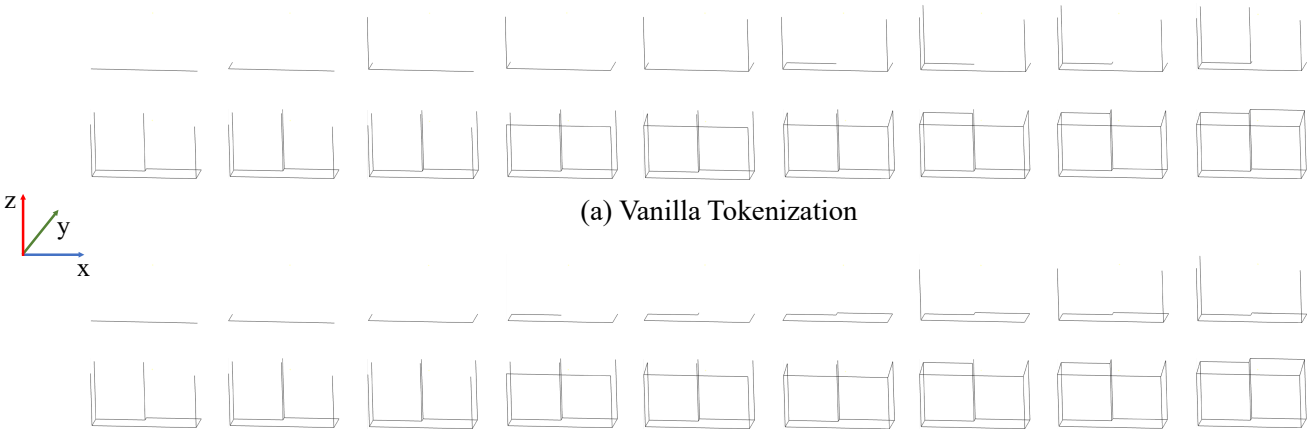


Figure 3. **An example of different tokenization schemes including vanilla tokenization (a) and hierarchical tokenization (b).**

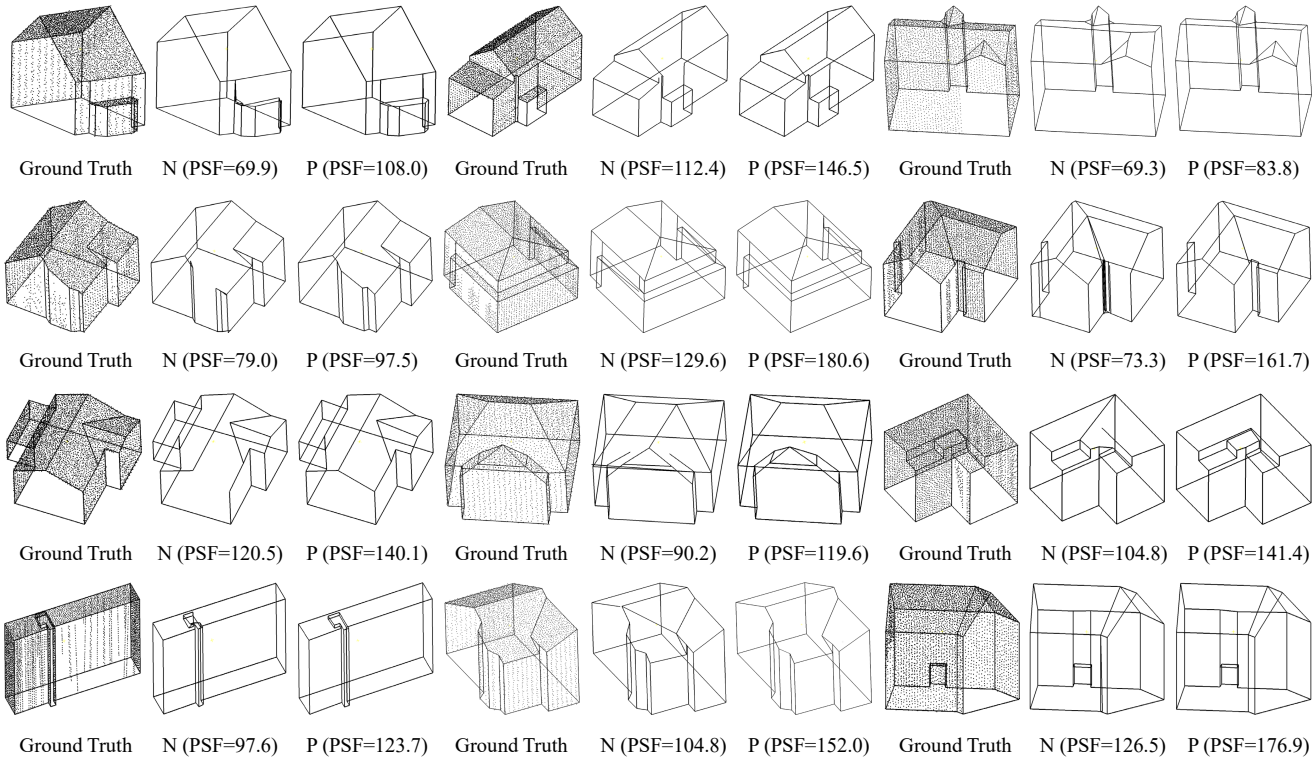


Figure 4. **More examples of the preference pair dataset.** N and P denote negative and positive samples.

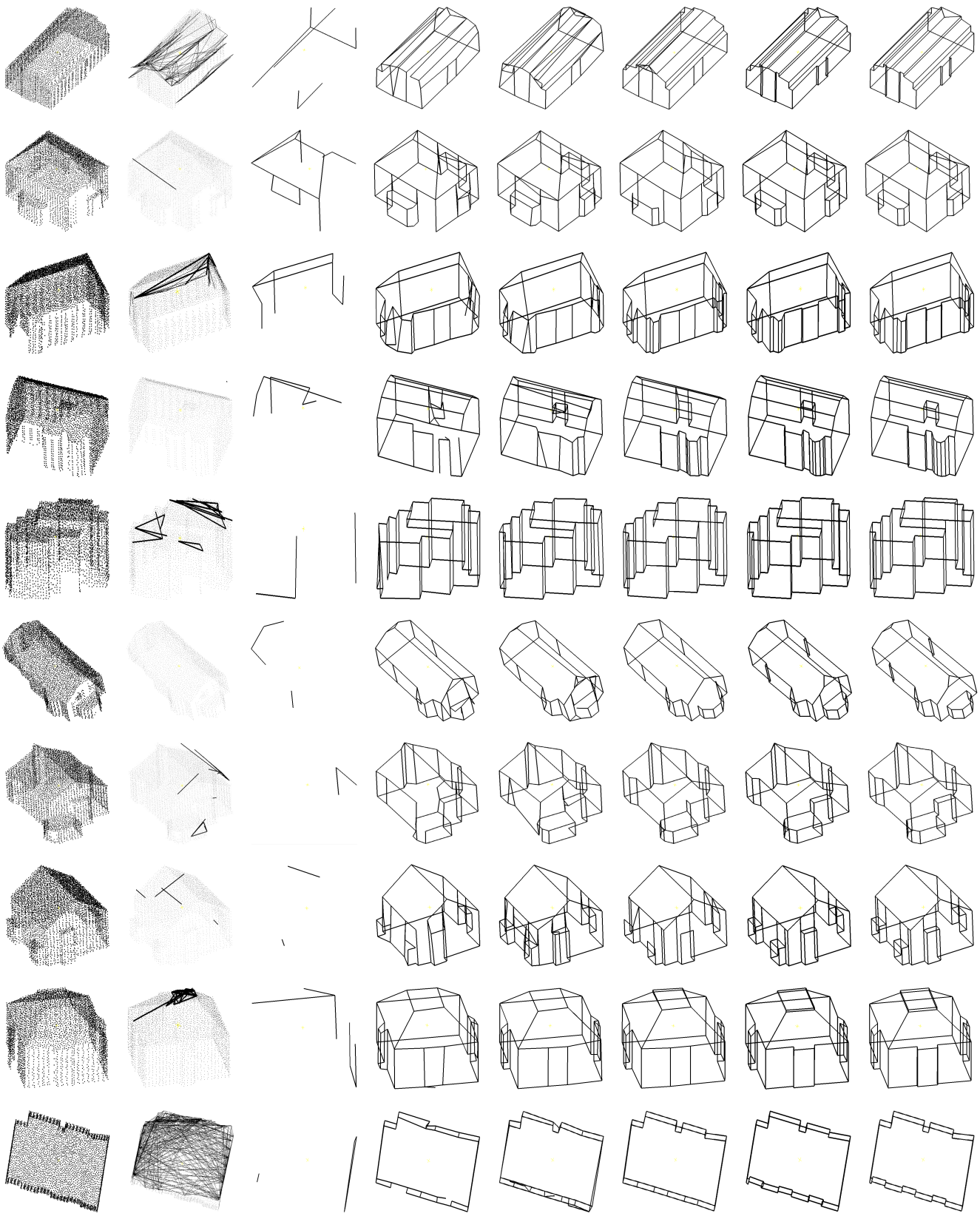


Figure 5. **Qualitative comparison between BuildingGPT and baseline methods.** The baseline methods include PC2WF [6], Point2Roof [5], PBWR [4], EdgeDiff [7], and BWFormer [8].

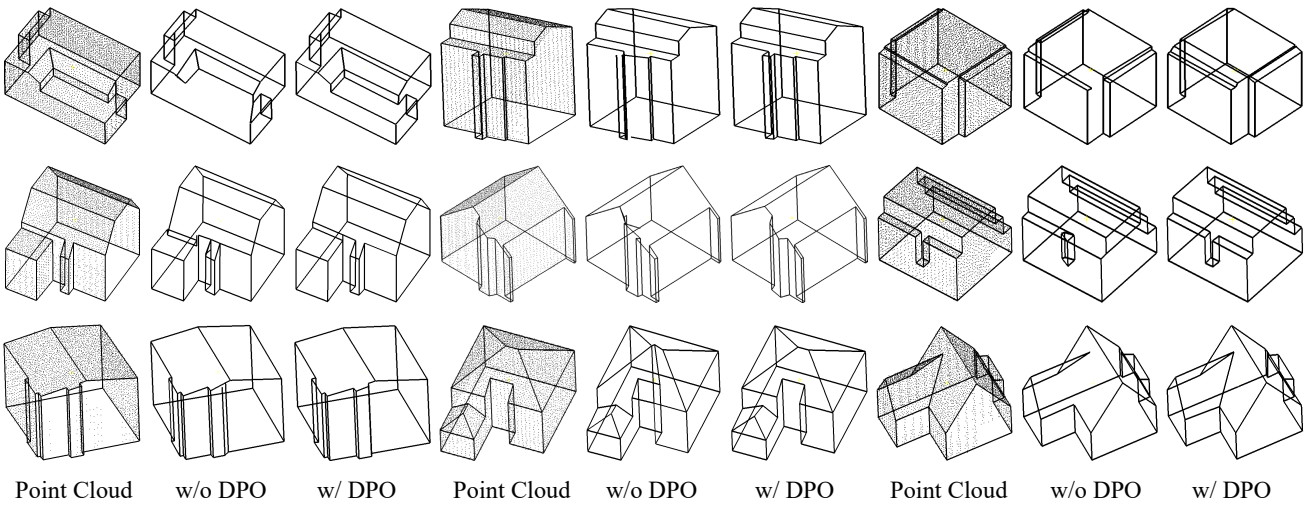


Figure 6. **Qualitative evaluation of post-training effectiveness.**

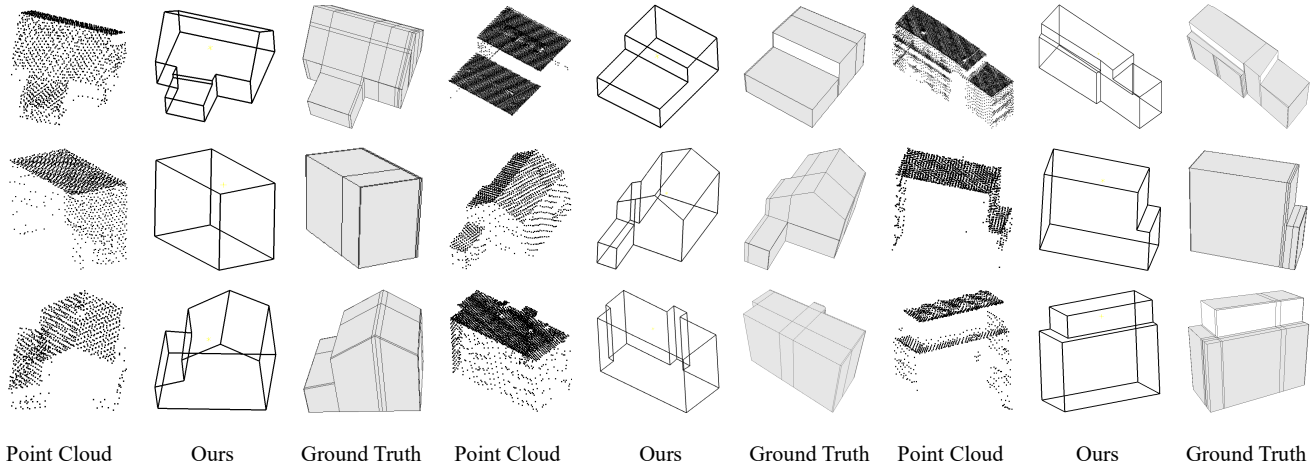


Figure 7. **Cross-data generalization on the unseen AHN3 dataset [3].**

References

[1] Zhaiyu Chen, Yilei Shi, Liangliang Nan, Zhitong Xiong, and Xiao Xiang Zhu. PolyGNN: Polyhedron-based graph neural network for 3D building reconstruction from point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 218:693–706, 2024. 1

[2] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 1

[3] Jin Huang, Jantien Stoter, Ravi Peters, and Liangliang Nan. City3D: Large-scale building reconstruction from airborne LiDAR point clouds. *Remote Sensing*, 14(9):2254, 2022. 4

[4] Shangfeng Huang, Ruisheng Wang, Bo Guo, and Hongxin Yang. PBWR: Parametric building wireframe reconstruction from aerial LiDAR point clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 27778–27787, 2024. 3

[5] 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*, 193: 17–28, 2022. 3

[6] Yujia Liu, Stefano D'Aronco, Konrad Schindler, and Jan Dirk Wegner. PC2WF: 3D wireframe reconstruction from raw point clouds. In *International Conference on Learning Representations (ICLR)*, 2021. 3

[7] Yujun Liu, Ruisheng Wang, Shangfeng Huang, and Guorong Cai. EdgeDiff: Edge-aware diffusion network for building reconstruction from point clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17008–17018, 2025. 3

[8] Yuzhou Liu, Lingjie Zhu, Hanqiao Ye, Shangfeng Huang, Xiang Gao, Xianwei Zheng, and Shuhan Shen. BWFormer: Building wireframe reconstruction from airborne LiDAR point cloud with Transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22215–22224, 2025. 3

[9] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 1

[10] Charlie Nash, Yaroslav Ganin, SM Ali Eslami, and Peter Battaglia. PolyGen: An autoregressive generative model of 3D meshes. In *International Conference on Machine Learning (ICML)*, pages 7220–7229, 2020. 1

[11] Yawar Siddiqui, Antonio Alliegro, Alexey Artemov, Tatiana Tommasi, Daniele Sirigatti, Vladislav Rosov, Angela Dai, and Matthias Nießner. MeshGPT: Generating triangle meshes with decoder-only Transformers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19615–19625, 2024. 1