International Journal of Digital Earth

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/tjde20

Detecting vertices of building roofs from ALS point cloud data

Gefei Kong, Yi Zhao & Hongchao Fan

To cite this article: Gefei Kong, Yi Zhao & Hongchao Fan (2023) Detecting vertices of building roofs from ALS point cloud data, International Journal of Digital Earth, 16:2, 4811-4830, DOI: 10.1080/17538947.2023.2283486

To link to this article: https://doi.org/10.1080/17538947.2023.2283486

INTERNATIONAL JOURNAL OF DIGITAL EARTH
2023, VOL. 16, NO. 2, 4811–4830
https://doi.org/10.1080/17538947.2023.2283486

Detecting vertices of building roofs from ALS point cloud data

Gefei Kong[a], Yi Zhao[a,b] and Hongchao Fan[a]

[a]Department of Civil and Environmental Engineering, Norwegian University of Science and Technology, Trondheim, Norway; [b]Research Area of Ecology and Biodiversity, School of Biological Sciences, The University of Hong Kong, Pokfulam, Hong Kong, People's Republic of China

ABSTRACT
Roof vertex information is vital for 3D roof structures. Reconstructing 3D roof structures from point cloud data using traditional methods remains a challenge because their extracted roof vertices are affected by uncertainty and additional errors from roof plane segmentation and supplementary sub-steps for extracting primitives. In this study, instead of segmenting roof planes and then extracting primitives based on them, a flexible rule-based method is proposed to directly detect the vertices of building roofs from point cloud data without the requirement of training data. The point cloud data is first voxelized with a dominant direction-based rotation. Based on the different features of the interior roof points and vertices, rules for voxel filtering and structure line determination are defined to extract the roof vertices. The experimental results on a custom dataset in Trondheim, Norway demonstrate that the proposed method can effectively and accurately extract roof vertices from point cloud data. The comparative experimental results with an unfine-tuned deep learning-based method on custom and benchmark datasets with different point densities further show that the proposed method has good generalization and can adapt to changes of datasets.

ARTICLE HISTORY
Received 20 September 2023
Accepted 8 November 2023

KEYWORDS
Roof vertex detection; 3D roof structure; voxelization; rule-based; point cloud data

1. Introduction

The 3D roof structure is an essential component in 3D building models from Level of Detail (LoD) 2 to LoD4 in the CityGML (Gröger et al. 2012; Kutzner, Chaturvedi, and Kolbe 2020). The roof structure consists of two elements: roof vertices and the edges between these vertices, and its reconstruction generally includes two steps: geometric primitive extraction (e.g. the extraction of roof plane polygons or boundaries) and primitive relationship inference. As point cloud data provides accurate 3D information with few occlusions and distortions, many methods for reconstructing 3D roof structures from point cloud data (Tarsha Kurdi, Awrangjeb, and Liew 2019; Tarsha Kurdi, Awrangjeb, and Munir 2021) have been proposed in the last decades.

Methods for 3D roof structure reconstruction from point cloud data can be classified into two types: model-driven and data-driven (Awrangjeb, Zhang, and Fraser 2013; Cao et al. 2017; Tarsha Kurdi et al. 2007). Top-down model-driven methods infer roof structures based on predefined roof primitives. Although robust, they are easily limited by a predefined primitive library. Bottom-up data-driven methods can overcome the problem of flexibility but require more sub-steps to achieve 3D roof reconstruction. This remains a challenge for data-driven methods: the additional steps and

CONTACT Hongchao Fan hongchao.fan@ntnu.no

rules for segmentation, plane fitting and refinement, and boundary tracing and regularization increase the uncertainty and lead to additional errors in the detected roof vertices.

To retain the advantages of flexibility offered by data-driven methods and address the challenges associated with the uncertainty of roof topology and vertices caused by roof plane segmentation and additional sub-steps of primitive extraction, Li et al. (2022) applied a new strategy in their research and proposed a deep learning-based method, Point2Roof. In their method, roof vertices were extracted, and edges were then predicted directly from point cloud data rather than from additional-step-extracted roof primitives. Their experimental results demonstrated its success of roof vertex detection and edge prediction. However, the deep learning-based method requires a large dataset for training, and its generalization is tightly linked to the diversity and richness of the dataset, whereas the roof structure dataset is limited. This particularly affects the performance of roof vertex detection. Hence, to directly extract roof vertices from point cloud data and avoid the need for a large labeled dataset for the deep learning-based method, a method for roof vertex detection without the requirement of training data is essential. In this study, a flexible voxel-based and noise-resistant method for detecting the vertices of building roofs from point cloud data is proposed to better support 3D roof structure reconstruction. The highlights of the present study are as follows.

(1) The proposed method directly detects roof vertices from roof point cloud data. This simplifies the process of existing data-driven methods, resulting in fewer errors in the extracted roof vertices that can ultimately improve the accuracy of the reconstructed 3D roofs.
(2) The proposed method achieves roof vertex detection without the requirement for training data. The filters for detecting the roof vertices and structure lines in the proposed method are rule-based, making the proposed method free of labeling data and training. This results in lower computational and data annotation costs.

The remainder of this paper is organized as follows: Section 2 summarizes related works on 3D roof reconstruction and roof vertex detection, Section 3 describes the detailed workflow of the proposed method, Section 4 introduces the experimental setup, Section 5 reports the experimental results and further discusses the critical issues of the proposed method, and Section 6 presents conclusions from this study and discusses future work.

2. Related works

Conventional methods of 3D roof reconstruction follow the strategy of geometric primitive extraction and primitive relationship inference. In the methods based on point cloud data, model-driven methods (Huang, Brenner, and Sester 2013; Jarząbek-Rychard and Borkowski 2016; Kwak and Habib 2014; Li and Shan 2022; Li, Xu, and Shan 2019) achieve the geometric primitive extraction by predefined primitives. Maas and Vosselman (1999) defined a standard gable roof house type as the basic primitive and then reconstructed building models using invariant moment analysis. Huang, Brenner, and Sester (2013) defined a library of roof primitives, which was divided into three groups with 11 types. The primitives were combined and merged with the proposed rules, and the roof model was reconstructed using generative modeling. Kwak and Habib (2014) decomposed a building into sets of rectangular parts and defined rectangular primitives to model the building parts with different roof types. The primitive relationship inference of a building model and its adjustment were based on the decomposed results, additional constraints, and sequential boundary fitting. Jarząbek-Rychard and Borkowski (2016) utilized the Roof Topology Graph (RTG) with a library of elementary structures to identify the topological structure of a segmented roof and achieved the reconstruction. Li and Shan (2022) first extracted planar patches by segmentation, and then, selected building primitives using four predefined primitives: shed, gable, pyramid, and hip. The reconstruction was finally achieved by holistic parameter estimation and 3D Bool operations.

Although model-driven methods perform well in terms of robustness and completeness during 3D roof reconstruction, they are limited by inflexible prior knowledge when reconstructing undefined shapes, and the high complexity of searching for appropriate primitives in large-scale scenes.

Data-driven methods based on point cloud data (Kim and Shan 2011; Nan and Wonka 2017; Sampath and Shan 2010; Wang et al. 2020; Xu et al. 2020) generally consider the geometric primitive extraction in two steps: roof plane segmentation and basic roof primitive extraction. Roof plane segmentation methods are commonly based on RANdom SAmple Consensus (RANSAC), Hough-Transform (HT), region growing, and clustering. These related works are reviewed in Tarsha Kurdi and Awrangjeb (2020). Some new methods (Chen et al. 2021; Dey et al. 2023; Wang and Ji 2021; Zhang and Fan 2022) have also been proposed in recent years to further improve segmentation performance. The segmented roof planes are usually further represented as plane equations, polygons, or outlines to organize them into abstract primitives that can be used in the subsequent topological analysis. Sampath and Shan (2010) used the convex-hull algorithm and the constraint of two mutually orthogonal directions to track and regularize roof plane boundaries as abstract primitives and finally reconstructed roofs and buildings based on these boundaries. Chen, Wang, and Peethambaran (2017) employed algorithms based on the Voronoi diagram, α-shape, and minimum spanning tree to achieve roof boundary tracing and further support reconstruction. Nan and Wonka (2017) refined the segmented building planes using an algorithm that considered the angle difference and shared points between two planes to prepare the plane selection and reconstruction. A similar strategy was applied in their improved method City3D (Huang et al. 2022) to achieve roof reconstruction. Wang et al. (2020) utilized the least-squares method to fit planes after plane segmentation for further building reconstruction.

These data-driven methods can achieve 3D roof reconstruction without requiring predefined information. However, more steps and rules in data-driven methods cause new issues: (1) The data-driven method requires the step of roof plane segmentation to obtain the roof plane information. This does not affect the visualization performance of the reconstructed roof structures, even in the case of over- and under-segmentation. However, the uncertainty associated with the segmentation step has the potential to influence the accuracy of the roof topology and vertices, and ultimately affect the calculability and practicality of the reconstruction result. Given that accurate roof structure information is a fundamental requirement for many applications, such as photovoltaic panel planning in solar energy analysis, addressing this issue is critical. (2) The step of roof primitive extraction after segmentation usually necessitates supplementary sub-steps and constraints, such as plane fitting and refinement, or boundary tracing and regularization of planar primitives. Additional constraint-based sub-steps increase the complexity of the method and lead to additional errors, thereby increasing the uncertainty and bias in the extraction results of the roof vertices.

Recently, some researchers have explored the possibility of applying a more concise and intuitive strategy to reconstruct 3D roof structures from point clouds (Li et al. 2022), in which the task of 3D roof reconstruction is simply divided into vertex detection and edge prediction from point clouds. A two-stage Deep Neural Network (DNN) named Point2Roof was designed to extract the roof vertices and predict the edges for reconstruction. This method achieved state-of-the-art results for both synthetic and real datasets. However, the need for training, fine-tuning, and a large training dataset for a deep learning-based method affects the practicality and generalization of their method. This particularly affects roof vertex detection because the essential point features are extracted in this module, and the following edge prediction is dependent on the results of vertex detection. For instance, when the model, trained on the synthetic dataset, was transferred to the real dataset, additional labeled training data was required.

3. Methodology

The goal of the proposed method is to detect critical vertices of a building roof from point cloud data. As shown in Figure 1, the workflow of the proposed method consists of four major sections:

(1) voxelization of point cloud data with dominant-direction rotation, (2) detection of candidate voxels of potential roof structure lines, (3) determination of structure lines based on candidate voxels, and (4) segmentation of structure lines to obtain the final roof vertices.

The workflow input is the segmented point cloud data for each single roof (Figure 1(a)). In the workflow, the voxels of the input roof point cloud data are first generated using a voxelization method with a dominant-direction rotation (Figure 1(b)). Subsequently, candidate voxels on the potential roof structure lines are detected by implementing rules for dividing the interior and edge voxels (Figure 1(c)). Next, the candidate voxels are extended to straight lines, which are clustered and merged to determine the roof structure lines (Figure 1(d)). Finally, the structure lines are segmented by considering their included voxels and the intersection relationships between them. The endpoints of the structure line segments are regarded as the final roof vertices in the voxel coordinate system (Figure 1(e), left). After re-rotation, the final roof vertices in the geographic coordinate system are obtained and output as the final result (Figure 1(e), right).

3.1. Voxelization with dominant-direction rotation

Point cloud data is a type of data with the irregular format (Qi et al. 2017). Voxelization is a common approach to regularize the unordered point cloud data and establish clear neighbor relations in a set of point clouds (Xu et al. 2017). However, two challenges exist in the voxelization of raw point cloud data:

(1) Many empty voxels are generated because of the misalignment of the point cloud data with the voxel grid, as shown in Figure 2(a).

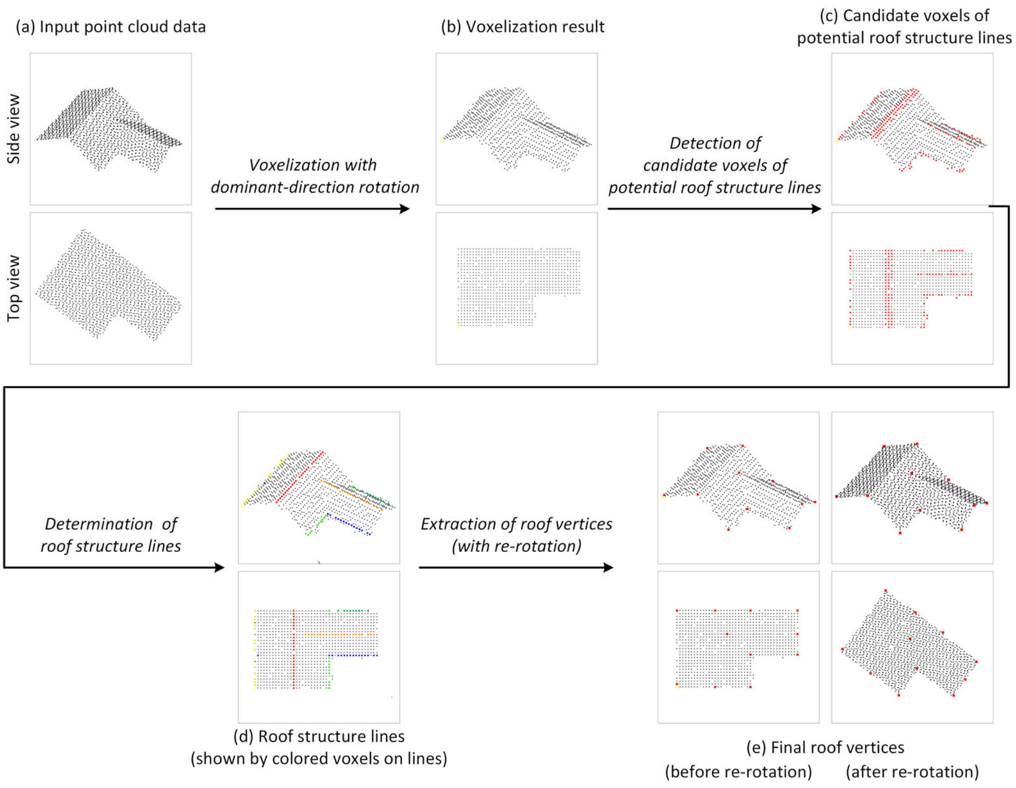

(a) Input point cloud data
(b) Voxelization result
(c) Candidate voxels of potential roof structure lines
Side view
Top view
Voxelization with dominant-direction rotation
Detection of candidate voxels of potential roof structure lines
Determination of roof structure lines
Extraction of roof vertices (with re-rotation)
(d) Roof structure lines (shown by colored voxels on lines)
(e) Final roof vertices
(before re-rotation)
(after re-rotation)

Figure 1. Overall workflow of the proposed method. The voxelization result shown in (b)~(e) is represented by voxel center coordinates.

(2) Although the first issue can be addressed by removing empty voxels from the voxel set, the neighborhood relationship of a voxel remains irregular, which affects the accuracy of the extracted roof edges. This is because, in the proposed method, only the nearest neighbors of a voxel are considered to extract the roof structure lines (which is a subset of all roof edges). However, the angles between a voxel and its nearest eight neighbors in 2D are always 45° or multiples thereof; therefore, edges with different angles are split into multiple segments, as shown by sl_1 and sl_2 in Figure 2(a).sub.

To address these issues, a voxelization method with rotation is used in this study. Instead of directly voxelizing the raw point cloud data, the point cloud data is first rotated according to its dominant direction and then voxelized using 3D cubic grids with the provided voxel size. The voxel size is set as 0.5 m to achieve the regularization of point cloud data, where the setting of voxel size follows the related research that used a point cloud data with a point density similar to that of the dataset used in the present study (Zięba-Kulawik et al. 2021). The dominant direction is defined as the rotation angle of the Minimum Area Rectangle (MAR) of the α-shape polygon of the point cloud data. The rotation angle is calculated using the minAreaRet function in OpenCV, as introduced by Zhao et al. (2022). Additional rotation based on the dominant direction is designed to assist in the generation of voxels with more regular neighbor relationships. This is illustrated in Figure 2(b).sub; through the application of rotation, the same roof edge as shown in Figure 2(a).sub can be represented more accurately as a single segment. The dominant-direction rotation effectively reduces the number of edges at other angles, considering that a building is generally formed of rectilinear shapes (Verma, Kumar, and Hsu 2006).

After voxelization and obtaining the voxels of the rotated point cloud data, empty voxels are removed from the voxel set. The final cleaned voxel set V will be used in the next sections, with $V = \{v|v = (x_v, y_v, z_v) \to (x_v^g, y_v^g, z_v^g)\}$, where v denotes a voxel, (x_v, y_v, z_v)denotes the voxel's index coordinates in the local voxel index coordinate system, and (x_v^g, y_v^g, z_v^g) denotes the voxel's center coordinates in the rotated geographic coordinate system. The voxels at the same horizontal slice (i.e. the voxels having the same z-indexes) are defined as one 'layer'.

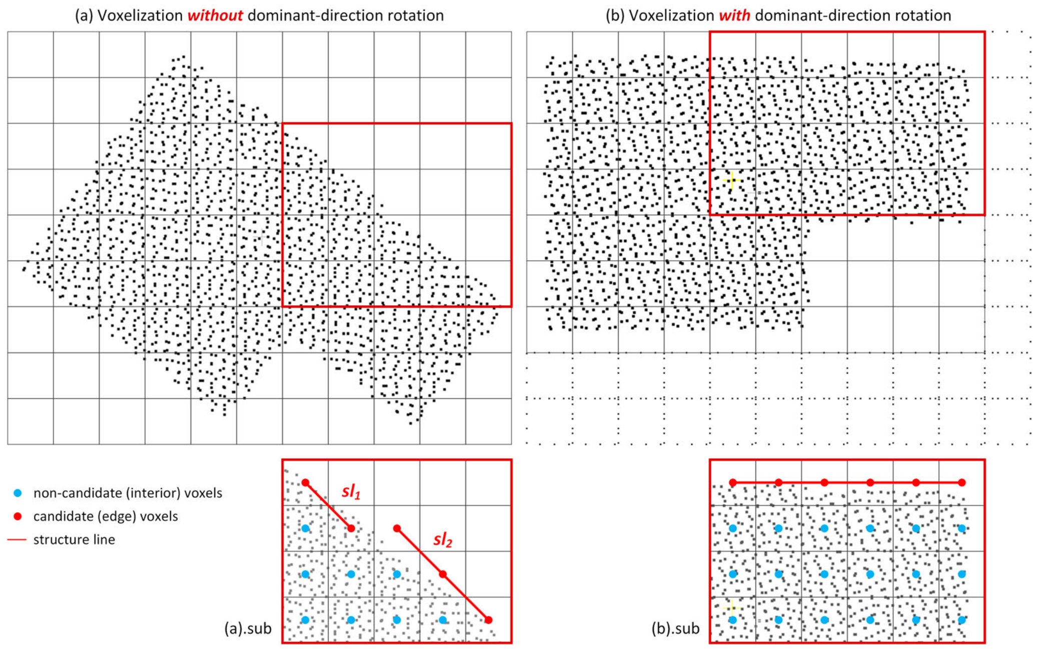

(a) Voxelization without dominant-direction rotation
(b) Voxelization with dominant-direction rotation
non-candidate (interior) voxels
candidate (edge) voxels
structure line
sl_1
sl_2
(a).sub
(b).sub

Figure 2. Example of voxelization without and with dominant-direction rotation.

3.2. Detection of candidate voxels of potential roof structure lines

In this study, a 'roof structure line' is defined as a roof edge where all voxels it comprises lie on the same layer, as shown by the red lines in Figure 3.

The structure lines of a roof are a subset of roof edges that can contain all the roof vertices. This means that once all the structure lines of the roof are extracted, the roof vertices can be located by the points on these lines. These structure lines have distinctive features in the z-axis direction, whereas the features of other roof edges that contain voxels from multiple layers are variable and should be defined jointly in the x- and y- and even z- axes. For example, as shown in Figure 3, when calculating the normal of each roof edge based on its neighboring area without voxels (indicated by the blue twill area in Figure 3), the difference between the normals of the roof structure lines $\{nl_{sl}\}$ can be expressed in the z-axis direction, and their normalized normals have the same absolute value. In contrast, the normals of the other roof edges $\{nl_o\}$ have various directions, which results in the unfixed values of these normalized normals.

Therefore, considering that roof structure lines are easier to detect and can be used to locate all roof vertices, in this study, roof structure lines are extracted to identify roof vertices instead of extracting all roof edges.

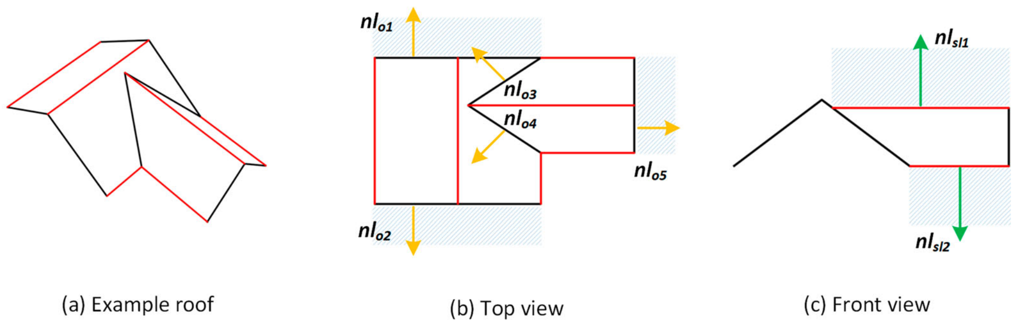
In the process of extracting the structure lines of a roof, the first step is the detection of candidate voxels situated on these roof structure lines. For this, in this section, **Rules 1** and **2** are used to divide voxels into two categories: interior voxels and candidate voxels on the roof structure lines, where the 'interior voxel' denotes the voxels that are not on the roof structure lines. **Rule 1** filters out the interior voxels on the sloped roof planes, and **Rule 2** filters out the interior voxels on the flat roof planes.

Rule 1: When considering the 3D neighbors of a voxel v ($v \in V$) from different layers, if v is positioned on a roof structure line, it should not have two neighbors from the different layers, which could combine to form a 180° structure with itself (i.e. v and its two neighbors are collinear). The definition of 'layer' is described at the end of Section 3.1. The mathematical representation of this rule is given by Equation (1).

$$\begin{aligned} &\forall \overrightarrow{vu_i} \cdot \overrightarrow{vu_j} \neq -1, \text{ when } z_v(u_i) \neq z_v(u_j) \neq z_v(v), \\ &\text{where } u_i, u_j \in U(v, r), i, j \in [0, s(U(v, r))), i \neq j \end{aligned} \tag{1}$$

$$U(v, r) = \{u | \mathrm{dist}_{Cb}(v, u) \leq r, u \neq v, u \in V\} \tag{2}$$

where $z_v(\cdot)$ denotes the z-value of the specified object (e.g. $z_v(v)$ represents the z-value of voxel v), $U(v, r) \subseteq V$ denotes the set comprising the neighbors of voxel v, r denotes the radius of the



▨	area without voxels
—	roof structure lines
—	other roof edges
→	normal of other edges in this direction
→	normal of structure lines in this direction

Figure 3. Example of roof structure lines and other roof edges.

searching neighbors (unit: voxel), and $s(\cdot)$ denotes the function that returns the number of objects in the specified set. Thus, $s(U(v, r))$ denotes the size of the neighbor set of voxel v when the searching radius is set to r. The mathematical representation of $U(v, r)$ is given by Equation (2), where -dist$_{Cb}(v, u)$ denotes the Chebyshev distance between voxels v and u.

Rule 2: When considering voxel v ($v \in V$) whose all neighbors from the same layer, if v is situated on a roof structure line, the number of 180° structures formed between itself and any two neighbors on the same layer should be less than four. The mathematical representation is given by Equation (3). The value '4' is constant because, for a voxel and its nearest neighbors on the same layer, the maximum number of 180° structures that include the voxel itself is four.

$$\text{count}\left(\left\{\begin{array}{c}(\overrightarrow{vu_i}, \overrightarrow{vu_j})|\overrightarrow{vu_i}\cdot\overrightarrow{vu_j} = -1,\\ z_v(u_i) = z_v(u_j) = z_v(v),\\ u_i, u_j \in U(v, r), i, j \in [0, s(U(v, r))), i \neq j\end{array}\right\}\right) < 4 \tag{3}$$

where count$(\cdot)$ denotes the function that returns the number of elements satisfying the given condition. In Equation (3), the 'element' is the neighbor combination.

Theoretically, the interior voxels can be removed by applying **Rules 1** and **2** for each voxel. However, in practical scenarios with real-world point cloud data, affected by noise and the angle of the slope roof plane, the slope plane of the roof appears 'stepped (staircase-like)' after voxelization, as shown in Figure 4. As a result, some unexpected interior voxels might be saved after filtering with the rules, even though they do not belong to the roof structure lines, such as voxels v_a and v_b shown in Figure 4, where $v_a, v_b \in V$. Hence, **Rules 1** and **2** should be iteratively applied to remove these voxels.

The full algorithm for detecting candidate voxels on roof structure lines is as follows. First, the set of candidate voxels on roof structure lines is initialized to include all voxels. Subsequently, **Rules 1** and **2** are iteratively applied to each voxel in the candidate voxel set until the maximum specified number of iterations *maxIter* is reached. In each iteration *iter*, the radius for searching for nearby neighbors (r) is expanded and set to the value equal to the current iteration number *iter*. Voxels that do not satisfy **Rules 1** and **2** are regarded as interior voxels and are removed from the candidate voxel set. The value of *maxIter* is half that of the number of layers. This setting ensures that all possible neighbor combinations from different layers for each voxel are covered. The mathematical representation of its

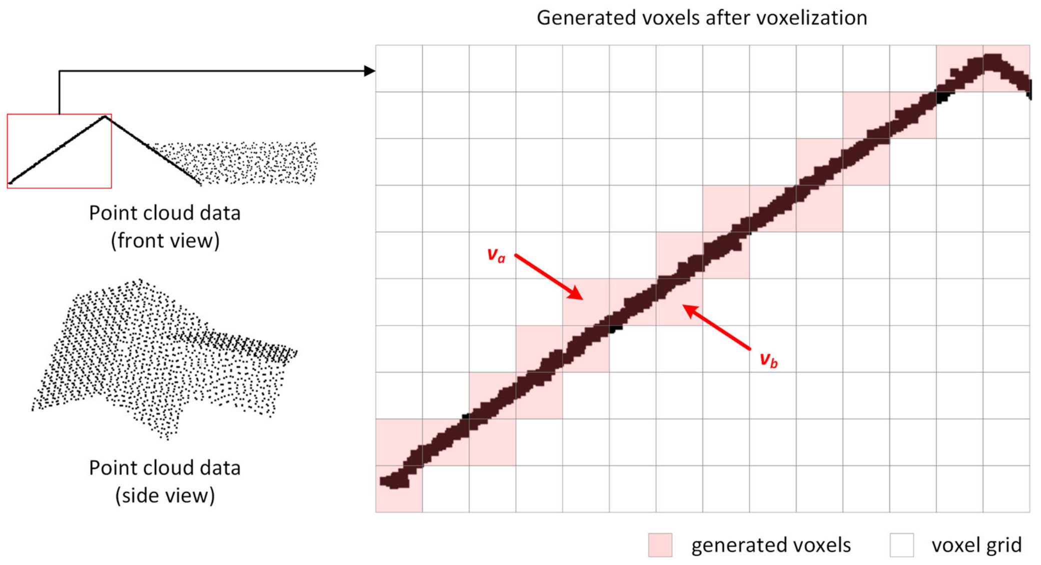

Generated voxels after voxelization
Point cloud data (front view)
Point cloud data (side view)
v_a
v_b
generated voxels
voxel grid

Figure 4. Example of laddering voxelization for a slope roof plane.

calculation is given by Equation (4).

$$maxIter = \left\lceil \frac{\max(\{z_v(v)\}) + 1}{2} \right\rceil \tag{4}$$

where $\{z_v(v)\}$ is the z-values of all voxels.

After applying this algorithm, the candidate voxel set $V_C = \{v_C\}$ is obtained. To further clean V_C, the isolated voxels in V_C are removed using a post-processing step. Here, 'isolated' means a voxel without any neighboring voxels at the same layer among its double nearest neighbors in V_C, where the 'double nearest' means that the neighbor search radius is set as 1×2 voxels. The cleaned candidate voxel set, V_C, is applied in Section 3.3.

3.3. Determination of roof structure lines

In this section, the structure lines are determined using the candidate voxel set V_C. This section consists of two sub-sections: (1) extraction of the set of candidate structure lines $L_C = \{l_C\}$ using V_C, and (2) clustering and merging of the candidate structure line set L_C to determine the structure lines $L = \{l\}$. L will be used in the next section for the roof vertex extraction. The workflow of this section is illustrated in Figure 5.

3.3.1. Extraction of candidate structure lines

The first sub-section for extracting the candidate structure lines L_C is achieved by applying **Rule 3**.

Rule 3: A candidate structure line l_C ($l_C \in L_C$) of a candidate voxel v_C ($v_C \in V_C$) is defined as follows. In considering all straight lines that pass through both the voxel v_C and any one of its nearest neighbors $U(v_C, 1)$ in the set of all voxels (V) on the same layer, the candidate structure line of v_C among these straight lines should be the line that includes the greatest number of voxels from V. The z-value of the candidate structure line $z_v(l_C)$ is equal to $z_v(v_C)$, where $z_v(v_C)$ is the z-value of the voxel v_C. The mathematical representation of the candidate structure line l_C is provided by Equation (5).

$$\begin{aligned} &l_C = l(v_{C,mc}, u_{C,mc}) : A_{l_C}x + B_{l_C}y + C_{l_C} = 0, \\ &\text{where } \operatorname{cntd}(l(v_{C,mc}, u_{C,mc})) = \max\left(\left\{\begin{array}{l} \operatorname{cntd}(l(v_C, u_{C,i}))| \\ u_{C,i} \in U(v_C, 1), i \in [0, s(U(v_C, 1))) \end{array}\right\}\right) \end{aligned} \tag{5}$$

where (A, B, C) denotes the line parameters, u_C denotes a voxel from $U(v_C, 1) \subseteq V$ ($U(v_C, 1) \subseteq V$: the nearest neighbors of voxel v_C in the set of all voxels V), and the other variables and functions in

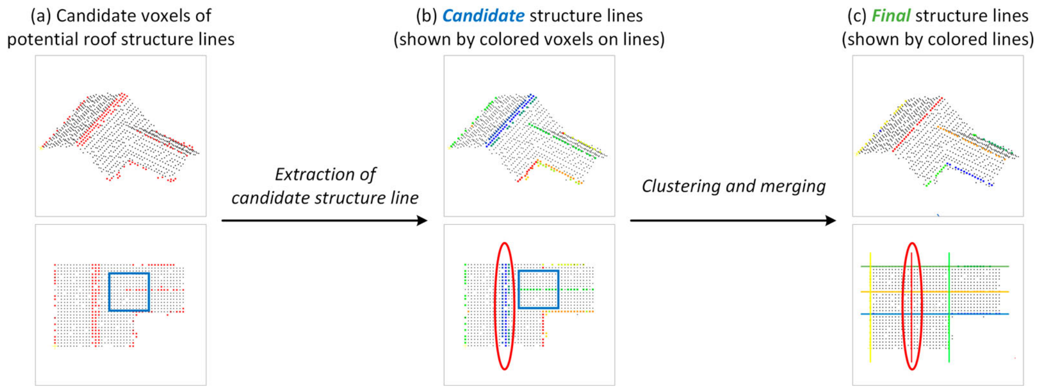

(a) Candidate voxels of potential roof structure lines
(b) Candidate structure lines (shown by colored voxels on lines)
(c) Final structure lines (shown by colored lines)
Extraction of candidate structure line
Clustering and merging

Figure 5. Workflow of determining structure lines.

Equation (5) are explained as follows.

$$l(v_C, u_C){:}A_{(v_C,u_C)}x + B_{(v_C,u_C)}y + C_{(v_C,u_C)} = 0,\ u_C \in U(v_C, 1) \tag{6}$$

$$\mathrm{dist}_l(v, l(v_C, u_C)) = \frac{|A_{(v_C,u_C)} \cdot x_v(v) + B_{(v_C,u_C)} \cdot y_v(v) + C_{(v_C,u_C)}|}{\sqrt{A^2_{(v_C,u_C)} + B^2_{(v_C,u_C)}}},\ v \in V \tag{7}$$

$$\mathrm{cntd}(l(v_C, u_C)) = \mathrm{count}\left(\left\{\begin{array}{c} v|\mathrm{dist}_l(v, l(v_C, u_C)) \le dis_{onl}, \\ z_v(v) = z_v(u_C) = z_v(v_C), \\ v \in V \end{array}\right\}\right) \tag{8}$$

where Equation (6) provides the mathematical representation of $l(v_C, u_C)$, which denotes a straight line passing through v_C and u_C. As defined in Equation (7), $\mathrm{dist}_l(v, l(v_C, u_C))$ is used to calculate the distance between voxel v in V and the line $l(v_C, u_C)$, where $x_v(v)$ and $y_v(v)$ denote the x and y values of voxel v. As defined in Equation (8), $\mathrm{cntd}(l(v_C, u_C))$ is used to count the number of voxels that are on line $l(v_C, u_C)$ and are from the same layer of v_C. A voxel positioned on a line is mathematically represented as a voxel whose distance from the line is less than $1/\sqrt{2}$ voxel, where $1/\sqrt{2}$ is the maximum distance for excluding voxels that deviate from the line. To simplify the representation of the subsequent equations, Equation (7) uses dis_{onl} to represent a special distance that determines whether a voxel is on a line. Moreover, in this study, dis_{onl} is set as 0.4, which is the round of half of $1/\sqrt{2}$, to provide a stricter limitation.

After applying **Rule 3** to each voxel v_C in candidate voxel set V_C, the candidate structure line set L_C can be obtained. Simultaneously, for each candidate structure line l_C, its V_{l_C} (the set of voxels on this candidate line) can also be obtained. The mathematical representation of V_{l_C} is given by Equation (9).

$$V_{l_C} = \{v|\mathrm{dist}_l(v, l_C) \le dis_{onl}, z_v(v) = z_v(l_C), v \in V\} \tag{9}$$

where $\mathrm{dist}_l(v, l_C)$ denotes the distance between a voxel v and a candidate structure line l_C.

An example of the extracted candidate structure lines is presented in Figure 5(b). This sub-section is not only used to extract candidate structure lines, but also contributes to complementing the roof structure, as shown by the blue boxes in Figure 5(a) and (b).

3.3.2. Clustering and merging of candidate structure lines

The second sub-section 'clustering and merging' is implemented to merge the candidate structure lines which represent the same structure, as shown by the red ellipse in Figure 5(b). This sub-section describes the application of **Rules 4 and 5** for clustering, and an algorithm with **Rules 6 and 7** for merging.

During the clustering process, all candidate structure lines $L_C = \{l_C\}$ are partitioned into several clusters $CLS = \{cls\}$, and each cluster is required to satisfy the two conditions in **Rules 4 and 5**. One cluster corresponds to a single roof structure line. A line l_C assigned to a cluster cls is rebranded as l_P. This renaming is undertaken to declare the differences between the clusters to which they belong, as shown in Equation (10).

Rule 4: In a cluster, all included lines should come from the same layer and have the same slope. This implies that they share the same z-values and line parameters (A, B). The mathematical representation of this rule is given by Condition (i) in Equation (10).

Rule 5: In a cluster, all included lines should be located close to one another, meaning that each line in this cluster should have at least one nearest neighboring line. As shown by Condition (ii) in Equation (10), its mathematical representation is that the minimum intercept distance between a line and its nearest neighbor line is less than or equal to 1 (unit: voxel), where 1 represents the distance between the nearest neighbor line.

$$\begin{aligned}
&cls = \{l_P \in L_C{:}A_{l_P}x + B_{l_P}y + C_{l_P} = 0\},\ cls \subseteq CLS \text{ and } cls \subseteq L_C,\\
&cls \text{ should satisfy following conditions:}\\
&\quad \text{(i) } z_v(l_{P,i}) = z_v(l_{P,j}),\ l_{P,i} \parallel l_{P,j};\\
&\quad \text{(ii) } \min(C_{l_P} - C_{\{l_P\}^C}) \le 1;\\
&\quad \forall l_P, l_{P,i}, l_{P,j} \in cls,\ i, j \in [0, s(cls)),\ i \ne j
\end{aligned} \tag{10}$$

where l_P is a candidate structure line in a cluster cls, $(A_{l_P}, B_{l_P}, C_{l_P})$ represents the line parameters of line l_P, and $\{l_P\}^C$ is the complement of l_P in universe cls.

After obtaining CLS (i.e. the cluster sets of candidate structure lines), the lines in each cluster are sorted based on the x and y values of their contained voxels to obtain the ordered cluster lines. Clusters with ordered lines are denoted as $CLS_O = \{cls_O\}$, and the lines in a cluster are denoted as $cls_O = \{l_O\}$.

As previously mentioned, each cluster corresponds to one roof structure line; however, some clusters may contain multiple lines. These lines in each cluster should be merged to obtain the simplest set of roof structure lines, $L = \{l\}$. To achieve the merging process, two merging rules, **Rules 6** and **7**, are defined as follows.

Rule 6: For a cluster $cls_O = \{l_O\}$ with an odd number of included lines, the central-most line (i.e. the line whose line parameter C is the closest to the mean C value of this cluster) is regarded as the merging result and saved as one roof structure line $l \in L$.

Rule 7: For a cluster $cls_O = \{l_O\}$ with an even number of included lines, the roof structure line $l \in L$ extracted from this cluster is determined as follows. It is the line with better continuity between the two central-most lines. If these two lines have the same continuity, the longer line is regarded as the roof structure line. The length of a line is equal to the distance between the two voxels with the minimum and maximum x-y coordinates on this line. In this rule, the continuity of a line l_O, $fc(l_O)$, is defined by Equation (11). This means that, by counting the number of intersection points between (1) line l_O and (2) a set of lines $cls^+_{l_O}$ that can interrupt l_O, the line with fewer intersection points is considered to have better continuity. The set $cls^+_{l_O}$ of line l_O is obtained using Equation (12).

$$fc(l_O) = -s(IP(l_O, cls^+_{l_O})),\ l_O \in cls_O,\ cls^+_{l_O} \subset cls_O \tag{11}$$

$$cls^+_{l_O} = \begin{cases} \{l^+_O | z_v(l^+_O) > z_v(l_O),\ l^+_O \in cls_O\} & (z_v(l_O) = \min Z_v(cls_O)) \\ \{l^+_O | z_v(l^+_O) = \max Z_v(cls_O),\ l^+_O \in cls_O\} & (\min Z_v(cls_O) < z_v(l_O) < \max Z_v(cls_O)) \\ \emptyset & (z_v(l_O) = \max Z_v(cls_O)) \end{cases} \tag{12}$$

where $Z_v\ (cls_O) = \{z_v(l_O),\ l_O \in cls_O\}$

where $IP(l_O, cls^+_{l_O})$ in Equation (11) denotes the set of intersection points between a line l_O and its $cls^+_{l_O}$, and the coordinates of each intersection point should be between the minimum and maximum x-y coordinates of V_{l_O} (the set of voxels on the line l_O). $s(IP(l_O, cls^+_{l_O}))$ returns the number of intersection points. In Equation (12), $Z_v(cls_O)$ denotes the z-value set of the voxels in the set cls_O.

The full algorithm of the merging process can be described as follows. First, based on the parity of the number of included lines in each cluster, all clusters are categorized into two types: odd and even. Next, **Rule 6** is applied to clusters with an odd number of included lines, and **Rule 7** is applied to clusters with an even number of included lines. Following the application of these two rules to CLS_O (the clusters with ordered lines), the result of merging can be obtained for each cluster.

After the merging process, a set of roof structure lines $L = \{l\}$ is determined. The red ellipses in Figure 5(b) and (c) illustrate examples of the roof structure lines before and after clustering and merging. Figure 5(c) also provides a full example of the roof structure lines ultimately obtained in this sub-section. This set of roof structure lines is used in the next section to extract the roof vertices.

Furthermore, additional information for use in the next section, $\{V_l, l \in L\}$, is obtained. This represents the set of voxels on these structure lines. The detailed mathematical representation of V_l is given by Equation (13).

$$V_l = \{v|\text{dist}_l(v, l) \leq dis_{onl}, z_v(v) = z_v(l), v \in V\}, l \in L \quad (13)$$

where $\text{dist}_l(v, l)$ denotes the distance between a voxel v and a roof structure line l.

3.4. Extraction of roof vertices by segmenting structure lines

The roof structure lines determined in Section 3.3 include the information of all the roof vertices. These roof vertices are represented as the endpoints of the roof line segments. Therefore, in this section, the roof structure lines are first converted into roof line segments. Next, the voxels at the roof vertices are extracted using the endpoints of these segments. Finally, the roof vertices are obtained by extracting the center coordinates of these voxels and re-rotating them. The workflow of this section is illustrated in Figure 6.

3.4.1. Extraction of roof line segments

In this sub-section, two sets of data are used: (1) L (i.e. the set of roof structure lines) and (2) $\{V_l, l \in L\}$ (i.e. the set of voxels on the roof structure lines). These data are utilized in the following algorithm to extract roof line segments. An example of the algorithm used to extract roof line segments is shown in Figure 7.

In the algorithm, first, the final output set of line segments is set as Seg_L, and Seg_L is initialized as an empty set. Then, for each structure line l ($l \in L$), the following steps (1)–(4) are implemented.

(1) Calculate the intersection points between a line l ($l \in L$) and the set of lines which can interrupt it. The intersection points of l are denoted as $IP(l, L_l^+)$, where $L_l^+ = \{l^+\}$ denotes the set of lines which can interrupt l. The L_l^+ of l is obtained using the same method as that shown in Equation (12), where cls_O in Equation (12) is replaced by L, l_O is replaced by l, and l_O^+ is replaced by l^+. For example, as shown in Figure 7(a), the orange line l_1 is selected as the structure line prepared to be interrupted. Then, its $L_{l_1}^+$ is obtained by applying Equation (12), which includes a single line l_2, as shown by the red line. Finally, $IP(l_1, l_2)$, the intersection point between l_1 and $L_{l_1}^+$ ($L_{l_1}^+ = \{l_2\}$), can be calculated, which is represented by the blue box in Figure 7(b).

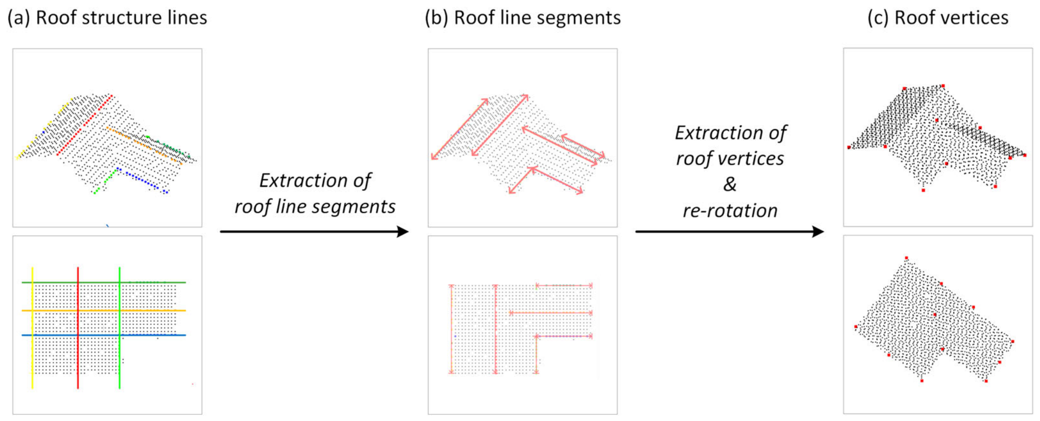

(a) Roof structure lines
(b) Roof line segments
(c) Roof vertices
Extraction of roof line segments
Extraction of roof vertices & re-rotation

Figure 6. Workflow of roof vertex extraction.

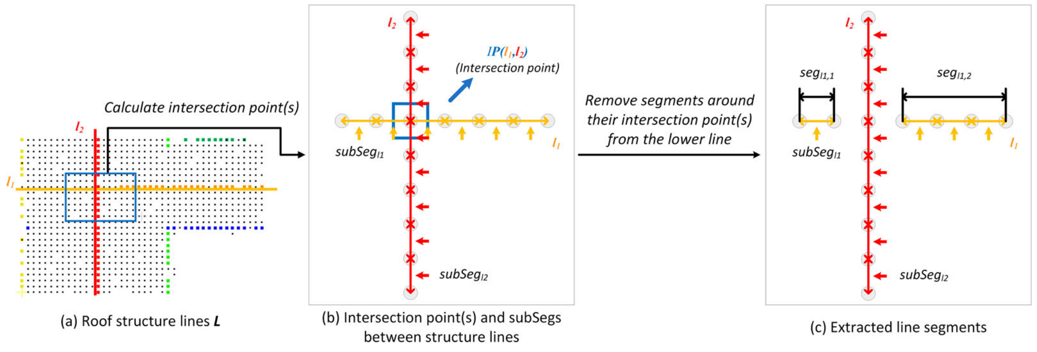

Calculate intersection point(s)
l_2
l_1
(a) Roof structure lines ***L***
$IP(l_1,l_2)$
(Intersection point)
$subSeg_{l1}$
$subSeg_{l2}$
(b) Intersection point(s) and subSegs between structure lines
Remove segments around their intersection point(s) from the lower line
$seg_{l1,1}$
$seg_{l1,2}$
(c) Extracted line segments

Figure 7. Example of roof line segment extraction.

(2) For l, first, sort its corresponding V_l (the set of voxels on l) by the x and y values of these voxels. Then, connect each two neighboring voxels to obtain all line sub-segments $\{subSeg_l\}$ for l. An example of the line sub-segments is shown by the orange and red line segments with arrows in Figure 7(b).

(3) Remove the sub-segment(s) in l that is(are) closest to the intersection point(s). If an intersection point is located in a sub-segment, this sub-segment, including the intersection, is removed. If the intersection point is at the endpoint of the two sub-segments, both sub-segments will be removed. An example of the latter case is shown in Figure 7(b). $IP(l_1, l_2)$ is at the endpoints of both l_1 and l_2. Hence, the two sub-segments closest to $IP(l_1, l_2)$ are removed, as shown in Figure 7(c).

(4) After step (3), remove the recurring voxels on the saved sub-segments of l and re-organize the saved voxels into new segments $\{seg_l\}$. An example of the new segments, $\{seg_{l_1,1}, seg_{l_1,2}\}$ of l_1, is shown by black segments in Figure 7(c). Each segment seg_l of l includes two voxels $\{v_S, v_E | v_S, v_E \in V_l\}$, where v_S and v_E are the endpoints of this segment. The re-organized line segments $\{seg_l\}$ of l will be added into the final output set Seg_L.

After implementing these steps for each structure line, the final output set $Seg_L = \{\{seg_l\} | l \in L\}$ can be obtained. An example of these segments of a roof is shown in Figure 6(b). Seg_L includes the line segment information of all roof structure lines L. This information is then used in sub-section 3.4.2 to extract the roof vertices.

3.4.2. Extraction of roof vertices with re-rotation

As mentioned in sub-section 3.4.1, each line segment is formed by two end voxels ($seg_l = \{v_S, v_E | v_S, v_E \in V_l\}$). Hence, these voxels can be obtained directly from the set of line segments Seg_L and considered as voxels at the roof vertices. The center coordinates of these voxels are extracted as the coordinates of roof vertices, where these coordinates are in the local voxel coordinate system. Subsequently, after re-rotating these coordinates using the transformation parameters from Section 3.1, the 3D roof vertices in the global geographic coordinate system can be obtained, as shown in Figure 6(c).

4. Experimental setup

4.1. Experimental datasets

Two datasets are utilized for the experiments. The first custom dataset uses the Airborne Laser Scanning (ALS) point cloud data in Trondheim, Norway, which was collected in 2018 and provided by the mapping authority of Trondheim Municipality. The point density of this point cloud data is 12–20 points/m^2. For the quantitative evaluation, 50 roofs are manually extracted from the point cloud data. Different roof types, including both primary (e.g. gabled and hipped) and combined (e.g. L-shaped, T-shaped, and cross-shaped) (Zhang and Fan 2022), are considered to ensure a comprehensive evaluation. Their roof vertices are manually labeled as the ground truth data.

Furthermore, to better evaluate the proposed method and facilitate comparisons with existing methods, a public dataset called RoofN3D (Wichmann et al. 2019; Wichmann, Agoub, and Kada 2018), is also selected. The ALS point cloud data of RoofN3D was obtained from the publicly available New York City dataset with a point density of approximately 4.72 points/m^2. Li et al. (2022) provided 500 manually labeled 3D structures of roofs from RoofN3D of which 450 were used for training and the remaining 50 for testing. In the present study, these 50 roofs from the testing dataset are utilized to quantitatively evaluate the proposed method and compare the evaluation results with those of Li et al. (2022).

4.2. Evaluation metrics

Similar to the study by Li et al. (2022), two types of metrics are used to quantitatively evaluate the performance of the proposed method: (1) vertex distance errors and (2) precision and recall for vertices. The former is used to evaluate the geometric quality of the detected roof vertices, and the latter is used to evaluate the overall quality of the detection results.

Vertex distance errors are calculated as follows (Equation (14)).

$$\begin{aligned} vd_x &= |x_{pred} - x_{gt}| \\ vd_y &= |y_{pred} - y_{gt}| \\ vd_z &= |z_{pred} - z_{gt}| \end{aligned} \tag{14}$$

where vd_x, vd_y, and vd_z are the distance errors of the roof vertex in the x-, y-, and z- directions, respectively. $\{x_{pred}, y_{pred}, z_{pred}\}$ denotes the coordinates of a roof vertex detected using the proposed method, and $\{x_{gt}, y_{gt}, z_{gt}\}$ denotes the corresponding ground truth coordinates for the same vertex. A correctly detected roof vertex is defined as a detected roof vertex whose distance error between itself and its nearest vertex in the ground truth data is less than a specified threshold.

The calculations of precision (P) and recall (R) are given by Equations (15) and (16), respectively.

$$P = \frac{TP}{TP + FP} \tag{15}$$

$$R = \frac{TP}{TP + FN} \tag{16}$$

where TP is the number of correctly detected roof vertices, FP is the number of incorrectly detected roof vertices, and FN is the number of undetected roof vertices which actually exist in the ground truth data.

4.3. Experimental design

To evaluate the proposed method, two experiments are conducted: (1) a basic experiment on the custom dataset, which evaluate the proposed method qualitatively and quantitatively; and (2) a comparative experiment with another method on both the custom dataset and a public benchmark dataset (RoofN3D) with different point densities, to comprehensively evaluate the generalization and practicality of the proposed method.

In the comparative experiment, Point2Roof (Li et al. 2022) is compared with the proposed method because it shares the same strategy as the proposed method, both of which separate the task of roof structure reconstruction into roof vertex detection and edge prediction. The same parameters are used when testing the proposed method on both datasets; the voxel size is set as 0.5 m. In comparing the proposed method with Point2Roof, different experiments are considered on two different datasets: (1) when comparing Point2Roof on the custom dataset, only Point2Roof (not fine-tuned) is compared. The trained Point2Roof model provided by the authors is used to predict the roof vertices for the test data of the custom dataset (link for the trained model: https://github.com/Li-Li-Whu/Point2Roof). This is because no training data is available for fine-tuning the custom dataset. In addition, because the proposed method does not require training, a comparison between the proposed method and the model (not fine-tuned) is fairer. (2) When testing the proposed method and comparing it with Point2Roof on RoofN3D, in addition to testing Point2Roof (not fine-tuned) using their provided trained model, the experimental results of Point2Roof (fine-tuned) reported in their paper are also considered and discussed for a comprehensive comparison.

5. Results and discussion

This section presents and discusses the experimental results. The results of the basic experiment are reported in Section 5.1 and the results of the comparative experiment are presented in Section 5.2. Two critical issues with the proposed method are discussed in Sections 5.3 and 5.4.

5.1. Evaluation results of the basic experiment

Figure 8 shows the qualitative evaluation results of the proposed method on the custom dataset, as well as the detailed results of each sub-section. In Figure 8, the black points in Figure 8(a)–(c) represent the roof voxels, whereas those in Figure 8(d)–(e) represent the roof points from the raw point cloud data. The colored points in Figure 8(b)–(e) represent the extracted/detected results of the corresponding sub-sections, and the point sets with different colors in Figure 8(c) correspond to different structure lines. The qualitative results indicate that the proposed method could accurately detect roof vertices directly from point cloud data. Moreover, these results indicate that the proposed method can adapt to different roof types. For primary roof types such as gabled (R1) and hipped (R2) roofs, all roof vertices are successfully detected using the proposed method. In the case of more complex combined roof types (R3–R5), the proposed method also successfully detects all roof vertices, except for one vertex in R5 (as shown in the purple box in Figure 8(e)).

The quantitative evaluation results of the proposed method on the custom dataset are as follows. The vertex distance errors vd_x, vd_y, and vd_z are 0.219, 0.249, and 0.151 m, respectively, with $P = 65.96\%$ and $R = 83.40\%$. This result is calculated based on 50 roofs with ground truth data, as introduced in Section 4.1. In addition, the mean running time of the proposed method is calculated to be 50.73 s/roof on the custom dataset, with a mean of 2765.32 points per roof. According to the obtained vertex distance errors, the geometric errors (quality) of roof vertices detected by the proposed method are approximately 0.2 m in each direction, which satisfies the accuracy requirement of LoD3 specified in CityGML 2.0 (Gröger et al. 2012). This demonstrates the practicality of the proposed method for various applications. Furthermore, the high recall in the evaluation results of the proposed method implies that it can successfully detect most roof vertices, which is preferred in practical applications. The relatively good precision and running time further confirm the practicality of the proposed method.

5.2. Comparative experiment

Table 1 presents the quantitative evaluation results of the comparative experiment. As shown in the first two rows of Table 1, on the custom dataset, when comparing the proposed method with Point2Roof (not fine-tuned), the vertex distance errors vd_x, vd_y, and vd_z of the proposed method reduce by 0.066, 0.075, and 0.005 m, respectively, and the precision and recall increase by 19.35% and 57.99%, respectively. These results indicate that the performance of the proposed method is

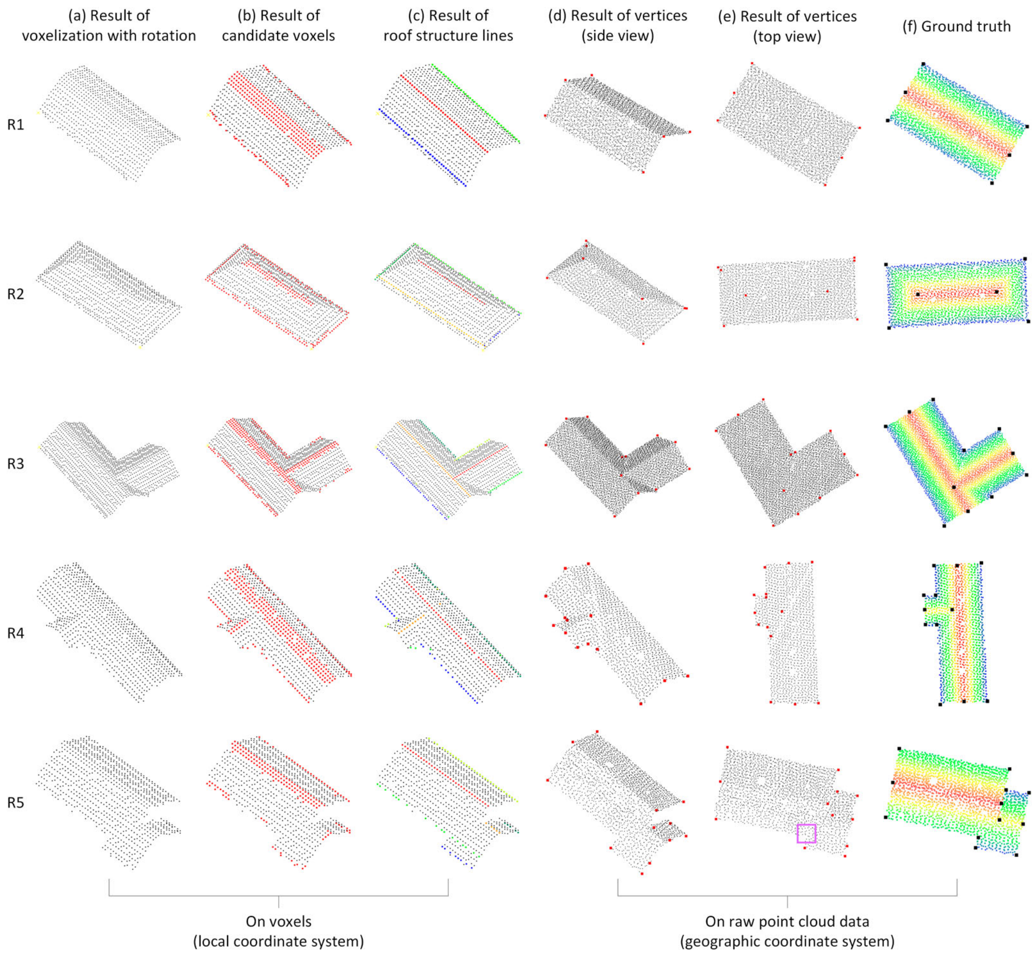

(a) Result of voxelization with rotation
(b) Result of candidate voxels
(c) Result of roof structure lines
(d) Result of vertices (side view)
(e) Result of vertices (top view)
(f) Ground truth
R1
R2
R3
R4
R5
On voxels (local coordinate system)
On raw point cloud data (geographic coordinate system)

Figure 8. Qualitative evaluation results on the custom dataset with detailed results for each section. (a) result of Section 3.1-voxelization with rotation; (b) result of Section 3.2-detection of candidate voxels of potential roof structure lines; (c) result of Section 3.3-determination of structure lines; (d) and (e) are the different-view results of Section 3.4-extraction of roof vertices; and (f) is the ground truth data of roof vertices.

significantly better than that of Point2Roof without fine-tuning. The reduced vertex distance errors signify that the roof vertices detected by the proposed method have better geometric quality, and the higher precision and recall indicate that the proposed method can detect the roof vertices more accurately and completely.

As shown in the last three rows of Table 1, on RoofN3D, when comparing the proposed method with the fine-tuned Point2Roof reported in the paper of Li et al. (2022), the vertex distance errors vd_x, vd_y, and vd_z of the proposed method are larger by 0.044, 0.033, and 0.136 m, respectively, and the precision and recall are lower by 59.23% and 10.65%, respectively. Although the evaluation result of the proposed method is worse than that of Point2Roof, it is important to note that the result for Point2Roof is obtained using a fine-tuned model. Li et al. (2022) extra manually label 450 roofs to achieve this result, which limits the generalization of their method. Compared to Point2Roof which uses the provided trained model (not fine-tuned), the performance of the proposed method is better than that of Point2Roof. The vertex distance errors vd_x and vd_y of the proposed method are reduced by 0.121 and 0.121 m, respectively, and the recall increases by 51.55%. Only vd_z in the proposed method is larger by 0.092 m, with a slight reduction of the precision by 3.23%.

Table 1. Quantitative evaluation results on different datasets for different methods (*: For these experiments, the threshold for identifying the correctly detected roof vertex (mentioned in Section 4.2) is set as 1 (m)).

Dataset	Method	vd_x (m)	vd_y (m)	vd_z (m)	P (%)	R (%)
The custom dataset	Point2Roof* (*not fine-tuned*)	0.285	0.323	0.156	46.62	25.41
	The proposed Method*	**0.219**	**0.249**	**0.151**	**65.96**	**83.40**
RoofN3D	Point2Roof (*fine-tuned*)	**0.183**	**0.188**	**0.045**	**99.29**	**96.56**
	Point2Roof* (*not fine-tuned*)	0.347	0.342	0.089	43.29	34.36
	The proposed Method*	0.226	0.221	0.181	40.06	85.91

Overall, the proposed method performs well when dealing with different point cloud data and shows better generalization than deep learning-based methods, especially when these methods are not fine-tuned.

5.3. Discussion for the effect of voxel size selection

In the proposed method, the parameter requiring manual consideration and setting is the voxel size. Generally, the selection of voxel size depends on the point density of the dataset and the required level of abstraction (Zięba-Kulawik et al. 2021). In this study, the voxel size is set as 0.5 m based on the point density of the common ALS dataset (12–20 points/m^2). To further explore the effect of voxel size, the tests are conducted for the proposed method on the custom dataset using various voxel sizes. In this experiment, considering that the minimum point spacing of the two datasets is approximately 0.3 m (calculated based on the point densities), different voxel sizes ranging from 0.3 to 1.0 m are used with the interval set as 0.1 m. The abovementioned metrics vd_x, vd_y, and vd_z and precision and recall are used to evaluate the experimental results. In addition, a metric $vd_{sum} = \sqrt{vd_x^2 + vd_y^2 + vd_z^2}$ is added to comprehensively represent the vertex distance error. The experimental results are shown in Figure 9. When setting voxel size as 0.5 m on the custom dataset, the proposed method achieves a balance between recall and precision, with achieving the highest recall (83.40%) and maintaining a relatively high precision (65.96%). This result demonstrates that a voxel size of 0.5 m is one of the best choices. Nevertheless, when changing the value of voxel size, the maximum differences of vd_{sum}, precision, and recall are 0.020 m, 14.68%, and 20.49%, respectively. This indicates that the critical role of voxel size selection in the voxel-based method.

The proposed method is further tested with different voxel sizes on RoofN3D which has a lower point density of approximately 4.72 points/m^2. As shown in Figure 10, for RoofN3D, the maximum differences in vd_{sum}, precision, and recall are 0.046 m, 21.34%, and 28.87%, respectively, which agree with the results on the custom dataset. Moreover, the most balanced result also occurs when setting the voxel size to approximately 0.5 m on this dataset, even though the point density changes. This indicates that a smaller voxel size (approximately 0.5 m) would provide better generalization and be appropriate for different datasets.

Furthermore, as shown in Figures 9 and 10, on both datasets, the three metrics exhibit similar trends: recall decreases and precision increases with increasing voxel size, whereas the change in vd_{sum} is less significant. This is because, when the voxel size is gradually increased in a point cloud set of a roof, the main roof structure in the point clouds can be saved and gradually highlighted, while the small and detailed structures (as well as noise) would be abstracted and lost. This also indicates that when planning a task aiming for higher recall, a smaller voxel size is preferred, whereas a larger voxel size should be considered if the task requires higher precision.

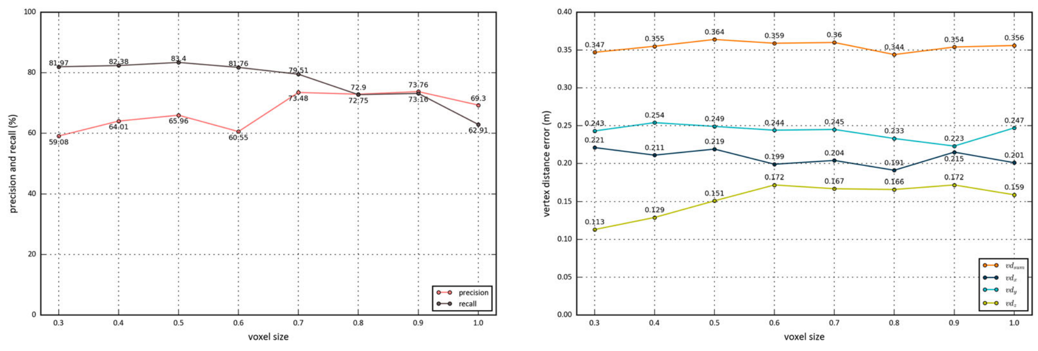


Figure 9. Experimental results of applying different voxel sizes on the custom dataset (left: curves of precision and recall; right: curves of vd_{sum}, vd_x, vd_y, and vd_z).

5.4. Discussion for the effect of point density

As mentioned in Section 4.1, the point density of the custom dataset is 12–20 points/m^2, and the point density of RoofN3D is approximately 4.72 points/m^2. These two datasets with different point densities are helpful in exploring the effect of the data point density on the performance of the proposed method.

The experimental results of the proposed method on the different datasets are presented in Table 1. Regarding precision, the comparative experimental result shows that the point density of point cloud data affects the precision performance of the proposed method. As shown in the second and fifth rows of Table 1, the proposed method performs better on the custom dataset (65.96%), with a relatively high point density. This is because point cloud data with a high point density is more integral than that with a lower point density. Data integrity determines the significance and integrity of the roof structure in the voxelization result of the point cloud data. A more significant and integral representation of the roof structure in the voxelization space ensures the accuracy of the extracted candidate voxels and the determined roof structure lines in the proposed method, which ultimately results in higher precision.

In terms of the vertex distance errors, as shown in the same rows of Table 1, the proposed method achieves $vd_x = 0.219$ m, $vd_y = 0.249$ m, and $vd_z = 0.151$ m on the custom dataset. On RoofN3D, the proposed method achieves $vd_x = 0.226$ m, $vd_y = 0.221$ m, and $vd_z = 0.181$ m. Comparing the results of the proposed method on different datasets, the differences in vd_x, vd_y, and vd_z are only 0.007, 0.028 and 0.030 m, respectively. In addition, the recalls on both datasets are similar (83.40% on the custom dataset and 85.91% on RoofN3D). These

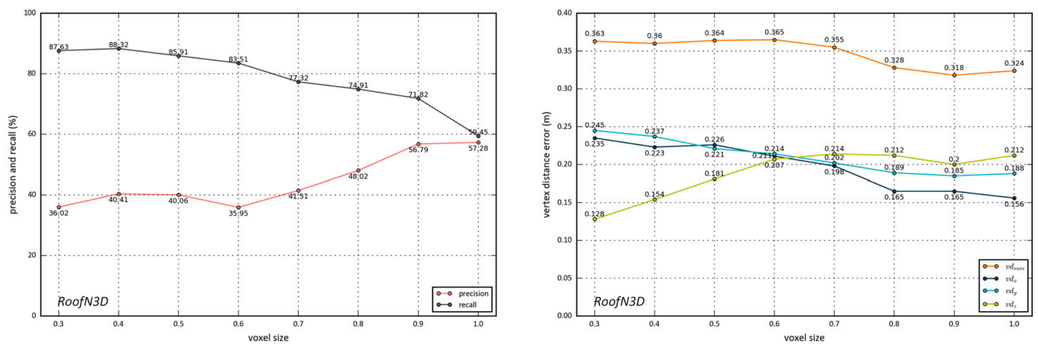


Figure 10. Experimental results of applying different voxel sizes on RoofN3D (left: curves of precision and recall; right: curves of vd_{sum}, vd_x, vd_y, and vd_z).

small differences indicate that the proposed method can perform stably in terms of the vertex distance error and recall, even for point cloud data with different point densities. Accordingly, considering the needs of practical applications that prefer high completeness of the detected roof vertices (recall), the proposed method can work well on different point cloud data with different point densities.

6. Conclusion

In this study, a new method is proposed for directly detecting roof vertices from point cloud data. In the proposed method, after voxelization with dominant direction-based rotation of the point cloud data, the rules for voxel filtering are defined and applied to detect candidate voxels on the roof structure lines. The roof structure lines are then determined by neighbor analysis, clustering, and merging of the candidate voxels. Ultimately, the roof vertices are obtained using these lines' segments. The workflow of the proposed method is general, meaningful, and flexible. No training process or data are required. The proposed method is evaluated on two segmented point cloud datasets with various roof types and different point densities. The qualitative and quantitative evaluation results demonstrate that the proposed method can stably and accurately detect roof vertices from point cloud data. Moreover, a comparative experiment with an unfine-tuned deep learning-based method demonstrates the generalization and flexibility of the proposed method. However, the proposed method has relatively low precision when processing point cloud data with low point density, although most of the roof vertices could be detected and a high recall could be guaranteed. In the future, this method can be further improved by adding postprocessing and considering the collinearity of the roof edges predicted from these detected vertices. Additionally, the proposed method may not yield the desired result when dealing with highly complex buildings with curved shapes, such as the Sydney Opera House, because these buildings lack clearly defined roof vertices. A potential solution to address this issue is to stop the proposed method early and consider candidate voxels that contain more vertices with more detail as the output. This work will also be combined with further research on roof edge prediction to provide an automatic solution for the reconstruction of 3D roof structures and 3D buildings in models over LoD2.

Acknowledgement

The authors would like to thank the mapping authority of Trondheim Municipality who provided the point cloud data in Trondheim, Norway.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported by NTNU Digital project (project No. 81771593).

Data availability statement

The authors confirm that the data and source codes used in this study can be available upon individual request.

References

Awrangjeb, M., C. Zhang, and C. S. Fraser. 2013. "Automatic Extraction of Building Roofs Using LIDAR Data and Multispectral Imagery." *ISPRS Journal of Photogrammetry and Remote Sensing* 83 (September): 1–18. https://doi.org/10.1016/j.isprsjprs.2013.05.006.

Cao, R., Y. Zhang, X. Liu, and Z. Zhao. 2017. “3D Building Roof Reconstruction from Airborne LiDAR Point Clouds: A Framework Based on a Spatial Database.” *International Journal of Geographical Information Science* 31 (7): 1359–1380. https://doi.org/10.1080/13658816.2017.1301456.

Chen, H., W. Chen, R. Wu, and Y. Huang. 2021. “Plane Segmentation for a Building Roof Combining Deep Learning and the RANSAC Method from a 3D Point Cloud.” *Journal of Electronic Imaging* 30 (5): 053022. https://doi.org/10.1117/1.JEI.30.5.053022.

Chen, D., R. Wang, and J. Peethambaran. 2017. “Topologically Aware Building Rooftop Reconstruction From Airborne Laser Scanning Point Clouds.” *IEEE Transactions on Geoscience and Remote Sensing* 55 (12): 7032–7052. https://doi.org/10.1109/TGRS.2017.2738439.

Dey, E. K., M. Awrangjeb, F. Tarsha Kurdi, and B. Stantic. 2023. “Machine Learning-Based Segmentation of Aerial LiDAR Point Cloud Data on Building Roof.” *European Journal of Remote Sensing* 56 (1): 2210745. https://doi.org/10.1080/22797254.2023.2210745.

Gröger, G., T. H. Kolbe, C. Nagel, and K.-H. Häfele. 2012. “OGC City Geography Markup Language (CityGML) Encoding Standard.”.

Huang, H., C. Brenner, and M. Sester. 2013. “A Generative Statistical Approach to Automatic 3D Building Roof Reconstruction from Laser Scanning Data.” *ISPRS Journal of Photogrammetry and Remote Sensing* 79 (May): 29–43. https://doi.org/10.1016/j.isprsjprs.2013.02.004.

Huang, J., J. Stoter, R. Peters, and L. Nan. 2022. “City3D: Large-Scale Building Reconstruction from Airborne LiDAR Point Clouds.” *Remote Sensing* 14 (9): 2254. https://doi.org/10.3390/rs14092254.

Jarząbek-Rychard, M., and A. Borkowski. 2016. “3D Building Reconstruction from ALS Data Using Unambiguous Decomposition into Elementary Structures.” *ISPRS Journal of Photogrammetry and Remote Sensing* 118 (August): 1–12. https://doi.org/10.1016/j.isprsjprs.2016.04.005.

Kim, K., and J. Shan. 2011. “Building Roof Modeling from Airborne Laser Scanning Data Based on Level Set Approach.” *ISPRS Journal of Photogrammetry and Remote Sensing* 66 (4): 484–497. https://doi.org/10.1016/j.isprsjprs.2011.02.007.

Kutzner, T., K. Chaturvedi, and T. H. Kolbe. 2020. “CityGML 3.0: New Functions Open Up New Applications.” *PFG – Journal of Photogrammetry, Remote Sensing and Geoinformation Science* 88 (1): 43–61. https://doi.org/10.1007/s41064-020-00095-z.

Kwak, E., and A. Habib. 2014. “Automatic Representation and Reconstruction of DBM from LiDAR Data Using Recursive Minimum Bounding Rectangle.” *ISPRS Journal of Photogrammetry and Remote Sensing* 93 (July): 171–191. https://doi.org/10.1016/j.isprsjprs.2013.10.003.

Li, Zhixin, and J. Shan. 2022. “RANSAC-Based Multi Primitive Building Reconstruction from 3D Point Clouds.” *ISPRS Journal of Photogrammetry and Remote Sensing* 185 (March): 247–260. https://doi.org/10.1016/j.isprsjprs.2021.12.012.

Li, L., N. Song, F. Sun, X. Liu, R. Wang, J. Yao, and S. Cao. 2022. “Point2Roof: End-to-End 3D Building Roof Modeling from Airborne LiDAR Point Clouds.” *ISPRS Journal of Photogrammetry and Remote Sensing* 193 (November): 17–28. https://doi.org/10.1016/j.isprsjprs.2022.08.027.

Li, Z., B. Xu, and J. Shan. 2019. “GEOMETRIC OBJECT BASED BUILDING RECONSTRUCTION FROM SATELLITE IMAGERY DERIVED POINT CLOUDS.” *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XLII-2/W13 (June): 73–78. https://doi.org/10.5194/isprs-archives-XLII-2-W13-73-2019.

Maas, H.-G., and G. Vosselman. 1999. “Two Algorithms for Extracting Building Models from Raw Laser Altimetry Data.” *ISPRS Journal of Photogrammetry and Remote Sensing* 54 (2–3): 153–163. https://doi.org/10.1016/S0924-2716(99)00004-0

Nan, L., and P. Wonka. 2017. “PolyFit: Polygonal Surface Reconstruction from Point Clouds.” In *2017 IEEE International Conference on Computer Vision (ICCV), 2372–2380*. Venice: IEEE. https://doi.org/10.1109/ICCV.2017.258.

Qi, C. R., H. Su, K. Mo, and L. J. Guibas. 2017. “PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation.” In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 652–660. Honolulu, HI, USA: IEEE.

Sampath, A., and J. Shan. 2010. “Segmentation and Reconstruction of Polyhedral Building Roofs From Aerial Lidar Point Clouds.” *IEEE Transactions on Geoscience and Remote Sensing* 48 (3): 1554–1567. https://doi.org/10.1109/TGRS.2009.2030180.

Tarsha Kurdi, F., and M. Awrangjeb. 2020. “Automatic Evaluation and Improvement of Roof Segments for Modelling Missing Details Using Lidar Data.” *International Journal of Remote Sensing* 41 (12): 4702–4725. https://doi.org/10.1080/01431161.2020.1723180.

Tarsha Kurdi, F., M. Awrangjeb, and A. W.-C. Liew. 2019. “Automated Building Footprint and 3D Building Model Generation from Lidar Point Cloud Data.” In *2019 Digital Image Computing: Techniques and Applications (DICTA)*, 1–8. Perth, WA, Australia: IEEE. https://doi.org/10.1109/DICTA47822.2019.8946008.

Tarsha Kurdi, F., M. Awrangjeb, and N. Munir. 2021. “Automatic Filtering and 2D Modeling of Airborne Laser Scanning Building Point Cloud.” *Transactions in GIS* 25 (1): 164–188. https://doi.org/10.1111/tgis.12685.

Tarsha Kurdi, F., T. Landes, P. Grussenmeyer, and M. Koehl. 2007. “Model-Driven and Data-Driven Approaches Using LIDAR Data: Analysis and Comparison.” In *ISPRS Workshop, Photogrammetric Image Analysis (PIA07)*, edited by U. Stilla, H. Mayer, F. Rottensteiner, C. Heipke, and S. Hinz, 87–92. Munich, Germany: Institute of Photogrammetry and Cartography.

Verma, V., R. Kumar, and S. Hsu. 2006. “3D Building Detection and Modeling from Aerial LIDAR Data.” In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06)*, 2:2213–2220. https://doi.org/10.1109/CVPR.2006.12.

Wang, S., G. Cai, M. Cheng, J. Marcato Junior, S. Huang, Z. Wang, S. Su, and J. Li. 2020. “Robust 3D Reconstruction of Building Surfaces from Point Clouds Based on Structural and Closed Constraints.” *ISPRS Journal of Photogrammetry and Remote Sensing* 170 (December): 29–44. https://doi.org/10.1016/j.isprsjprs.2020.09.004.

Wang, X., and S. Ji. 2021. “Roof Plane Segmentation From LiDAR Point Cloud Data Using Region Expansion Based L0 Gradient Minimization and Graph Cut.” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 14: 10101–10116. https://doi.org/10.1109/JSTARS.2021.3113083.

Wichmann, A., A. Agoub, and M. Kada. 2018. “ROOFN3D: DEEP LEARNING TRAINING DATA FOR 3D BUILDING RECONSTRUCTION.” *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XLII–2 (May): 1191–1198. https://doi.org/10.5194/isprs-archives-XLII-2-1191-2018.

Wichmann, Andreas, A. Agoub, V. Schmidt, and M. Kada. 2019. “RoofN3D: A Database for 3D Building Reconstruction with Deep Learning.” *Photogrammetric Engineering & Remote Sensing* 85 (6): 435–443. https://doi.org/10.14358/PERS.85.6.435.

Xu, Y., W. Yao, L. Hoegner, and U. Stilla. 2017. “Segmentation of Building Roofs from Airborne LiDAR Point Clouds Using Robust Voxel-Based Region Growing.” *Remote Sensing Letters* 8 (11): 1062–1071. https://doi.org/10.1080/2150704X.2017.1349961.

Xu, B., X. Zhang, Z. Li, M. Leotta, S.-F. Chang, and J. Shan. 2020. Deep Learning Guided Building Reconstruction from Satellite Imagery-Derived Point Clouds.” arXiv.

Zhang, C., and H. Fan. 2022. “An Improved Multi-Task Pointwise Network for Segmentation of Building Roofs in Airborne Laser Scanning Point Clouds.” *The Photogrammetric Record* 37 (179): 260–284. https://doi.org/10.1111/phor.12420.

Zhao, Y., J. Zhang, H. Li, X. Gu, Z. Li, and S. Zhang. 2022. “Automatic Cobb Angle Measurement Method Based on Vertebra Segmentation by Deep Learning.” *Medical & Biological Engineering & Computing* 60 (8): 2257–2269. https://doi.org/10.1007/s11517-022-02563-7.

Zięba-Kulawik, K., K. Skoczylas, P. Wezyk, J. Teller, A. Mustafa, and H. Omrani. 2021. “Monitoring of Urban Forests Using 3D Spatial Indices Based on LiDAR Point Clouds and Voxel Approach.” *Urban Forestry & Urban Greening* 65 (September): 127324. https://doi.org/10.1016/j.ufug.2021.127324.