This is the Accepted Manuscript version of an article published by Elsevier in the journal Automation in Construction in 2023, which is available at: https://doi.org/10.1016/j.autcon.2023.104917
Cite as: Dayu Yua, Peng Yue, Fan Ye, Deodato Tapete, Zheheng Liang. Bidirectionally Greedy Framework for Unsupervised 3D Building Extraction from Airborne-based 3D Meshes. *Automation in Construction*, 152: 104917, 2023.

Bidirectionally Greedy Framework for Unsupervised 3D Building Extraction from Airborne-based 3D Meshes

Dayu Yu[a], Peng Yue[a,b,c,d*], Fan Ye[e], Deodato Tapete[f], Zheheng Liang[g]

[a]*School of Remote Sensing and Information Engineering, Wuhan University, Wuhan, Hubei, China*

[b]*Collaborative Innovation Center of Geospatial Technology, Wuhan, Hubei, China*

[c]*Hubei LuoJia Laboratory, 129 Luoyu Road, Wuhan, Hubei, 430079, China*

[d]*Hubei Province Engineering Center for Intelligent Geoprocessing (HPECIG), Wuhan University, Wuhan, Hubei, China*

[e]*School of Computer Science, China University of Geosciences, Wuhan, Hubei, China*

[f] *Italian Space Agency (ASI), Via del Politecnico snc, 00133, Rome, Italy*

[g]*South Digitial Technology Company, Guangzhou, China*

Abstract

Automatic building information extraction is an active research field in photogrammetry and remote sensing. However, most methods are proposed for supervised segmentation of point clouds or images, which can only capture limited building texture or geometric information, resulting in the obtained buildings being often fragmented. Therefore, we propose a bidirectionally greedy framework to extract spatial-continuous, geometry-complete, fine-textured 3D building models from large-scale 3D meshes captured by airborne in an unsupervised manner. The framework consists of two key steps in opposite directions, namely greedy culling and greedy recovery. Greedy culling will maximize the removal of non-building primitives based on geometric and textural features. Greedy recovery is designed to maximize the detection of building primitives that are mistakenly removed by the greedy culling, by utilizing topological accessibility. The framework is assessed quantitatively and visually on five high-resolution datasets with different scenes. The results indicate the framework's effectiveness in accurately extracting fine-grained building models with complete geometry that can be visualized and analyzed for various 3D applications.

keywords: geomesh; realistic 3D building model; building segmentation; photogrammetry

* *Correspondence author*

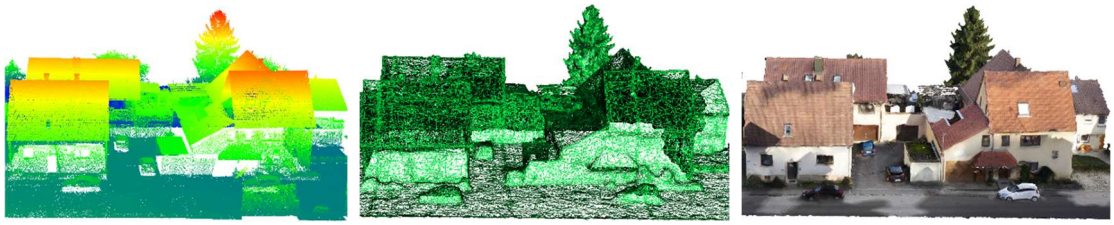
Email assress: dayuyu@whu.edu.cn (Dayu Yu), pyue@whu.edu.cn (Peng Yue), yefan@cug.edu.cn (Fan Ye), and deodato.tapete@asi.it (Deodato Tapete)

Preprint submitted to Automation in Construction

1. Introduction

In recent years, airborne platform-based photogrammetry and LiDAR technologies have substantially improved, up to the point that they can collect large-scale 3D geographical information simultaneously in a cost-effective, fast, and accurate manner. Prominent amongst these is oblique photogrammetry mounted on an unmanned aerial vehicle (UAV), which represents an attractive solution to simultaneously capture Cartesian coordinates and texture information of building roofs and façades even in the presence of slight obscuring. While in the past images and point clouds (Fig 1a) were the typical graphical outputs to display collected geographical information, 3D meshes (Fig. 1b-c) have increasingly become a common carrier for 3D geographic information and are widely used in many fields such as virtual geographical environment, 3D geographic information system (GIS), and digital twins [1,2,3,4].

To distinguish from ordinary 3D meshes in computer graphics (CG), we call "geomesh" a realistic 3D mesh of a large geographic scene captured from an airborne platform. Compared to 2D images, such as digital orthomosaic models (DOM) and digital elevation models (DEM), geomeshes contain detailed 3D geometric and texture features. Compared with discrete point clouds, geomeshes are characterized by advantages, such as spatial continuity, explicit adjacencies, and simultaneous geometric and texture information. In addition, geomeshes occupy less disk and memory, mainly because some geometrically irrelevant points are filtered out during the process of reconstruction and mesh generation from the point cloud. Therefore, geomeshes are preferred for 3D geographic applications and spatial analysis [5].



(a) (b) (c)

Fig. 1. Example of (a) point cloud, (b) 3D mesh, and (c) realistic 3D mesh with textures (i.e., geomesh) of the same scene encompassing buildings.

However, for visualization purposes, current realistic 3D modeling software (e.g., Photomesh and ContextCapture) rely on an automation mechanism by which the entire geographical scene is expressed with a continuous geomesh without category differentiation of feature objects. This process prevents the management, query, and analysis of 3D geographic information can be objectified. Therefore, the value of geomeshes is hampered for use in 3D geographic information engineering for city-scale topics such as digital twins and microclimate modeling [6,7,8]. Usually, buildings are the most valuable object category in a geomesh. However, their objectification process requires a high degree of human interaction, which is very time-consuming and accounts for more than half of the total time required for the 3D urban modeling process.

Automatic building extraction from remote sensing data increasingly benefits urban planning, disaster simulation, and updating of GIS databases. It has always been an active research field in photogrammetry and remote sensing [9]. However, it is still a challenging task due to the high complexity of urban scenes which include factors such as trees, water regions, and man-made non-buildings features, that in turn cause an increase of commission and/or omission errors in building extraction [10]. Efforts have been made to extract footprints and geometric information of buildings from images and point clouds [11,12,13,14,15,16,17], but there have been no relevant

investigations focused on geomeshes. Point clouds are indeed considered intermediate products [18], given that the building information extracted from a point cloud still needs to be reconstructed to generate building models. On the contrary, building models extracted from geomeshes are considered as end-user products and do not need to undergo such a process. These methods, either unsupervised [15,16,19] or supervised depth learning [17,20,21], are inevitably affected by fragmentation in the extracted buildings. This issue is particularly prevalent at the building boundaries, as these methods prioritize accuracy over the geometric integrity of the resulting buildings. Unfortunately, the loss of a geometric primitive during 3D building model extraction from a geomesh can substantially affect both the visualization and analysis capabilities of 3D applications based on them.

A novel bidirectionally greedy framework is therefore proposed to extract spatial-continuous, geometry-complete, fine-textured 3D building models from geomeshes in an unsupervised manner. The framework consists of two key steps in opposite directions, namely greedy culling and greedy recovery. Greedy culling will maximize the removal of non-building primitives based on geometric and textural features. Greedy recovery is designed to maximize the detection of the building primitives that are mistakenly removed by the greedy culling, by utilizing topological accessibility. After these two seemingly opposite steps, 3D building models are extracted with high recall and accuracy. The contributions of this paper can be listed as follows:

1) The first attempt to segment buildings from airborne-based 3D meshes in an unsupervised manner.
2) We proposed a bidirectionally greedy framework that can extract spatial-continuous, geometry-complete, fine-textured 3D building models from realistic 3D meshes with high recall and accuracy.
3) As far as we know, few studies utilize topology structures to extract building information from various sources. So, this paper shows a new approach that utilizes topological accessibility to ensure the geometric integrity of buildings.
4) We provide multi labeled geomeshes for the realistic 3D building extraction task, and the proposed methods are evaluated on the geomeshes

The rest of the paper is organized as follows: the related works are reviewed in Section 2. The proposed building extraction framework is introduced in detail in Section 3. The results of implementations are reported in Section 4, analyzed and discussed in Section 5. Conclusions are provided in Section 6.

2. Related Works

2.1. Building extraction

To date, geomesh is a recently proliferated 3D geographic data. However, not enough research has been made to extract building objects from geomesh. On the contrary, many approaches have been developed to extract buildings from other data sources. These research works can be generally divided into three method-related groups, namely 2D images-based, point clouds-based, and 2D-3D information fusion.

The 2D images-based method typically relies on visual or spectral features to segment buildings from images, the latter being collected from satellites [10,22,23,24,25,26] or airborne platforms [11,12,27,28,29]. Although promising 2D building footprints and rough heights have been obtained, occlusions and shadows leading to significant errors cannot be avoided, especially in densely built-up areas [9].

Various methods using geometric features or shape descriptors have been proposed to extract building points from point clouds, the latter being collected by terrestrial laser scanning (TLS), mobile laser scanning (MLS),

airborne laser scanning (ALS), and aerial oblique photogrammetry. The point clouds collected by LiDAR-only-based acquisition such as ALS, TLS, and MLS have significant fragmentation of individual buildings without multi-view stitching, partly due to data collection imperfections including occlusions, shadows, and general noise [10,30]. Oblique photogrammetry, on the other side, can capture complete building information even in densely developed areas because of its simultaneous acquisition from multiple views. These methods on point clouds can be mainly classified as detection-based, morphological, and classifier-based methods.

In detection-based methods, façade and roof features are detected from a point cloud by integration of pre-existing knowledge regarding building shapes, such as normal distribution, curvature distribution, density variation, and distance variation. For this objective, region growing (RG) [13,31,32,33], random sample consensus (RANSAC) [34,35], Hough transform (HT) [36,37], and clustering approaches [38,39,40], which are four major contenders, are often utilized for building detection. Intrinsically, these algorithms transform a building detection problem into a plane-fitting problem for façades and roofs. In principle, RG finds the planes by accumulating the neighboring pints into a region satisfying growing conditions. RANSAC is an iterative algorithm to correctly estimate planar parameters from a point cloud that may contain outliers. HT maps a point cloud into a discretized parameter space and then extracts planes by selecting those parameters with a significant number of votes. However, RANSAC and RG often extract unexpected planes in vegetation areas, while the HT is time-consuming [9,41]. On the other hand, clustering methods extract building planes by minimizing an energy function considering planar features.

Several morphological methods have been proposed for building detection based on dilate and erode operators. Chen et al. [42] proposed a modified morphology algorithm to mark points as building points, which used a large window and a small window to perform the opening. Cheng et al. [43] introduced the reversed iterative mathematic morphological algorithm that allows the thresholds to be determined in a self-adaptive way for the derivation of building regions from LiDAR data. Mongus et al. [41] used characteristic values mapped from differential morphological profiles for building extraction. In general, the performance of morphological methods in building extraction is significantly worse than that for ground filtering of point clouds [44,45], and the choice of a structuring element plays an important role when considering their accuracies.

Owing to the structural heterogeneity of buildings, the accuracy of detection-based and morphology methods is hard to meet industrial needs. Therefore, some studies have used classifiers to extract buildings, improving the accuracy of extraction. Zhang [46] applied geometry, intensity, and radiometry characteristics into a support vector machine to classify the objects of urban regions. A reliable result in complex urban scenes was obtained by integrating a random forest (RF) classifier into a conditional random field (CRF) framework [47]. An AdaBoost classifier was trained to separate the TLS point clouds based on multiscale and hierarchical point clusters [40]. More recently, the more popular classifiers for point clouds are the so-called deep learning methods such as convolutional neural network (CNN), graph convolutional network (GCN), and transformer. For example, an efficient and lightweight CNN, namely RandLA-Net, was proposed to directly infer per-point semantics for large-scale point clouds [21]. A very deep 56-layer GCN was successfully trained for point cloud segmentation [20]. Recently, a self-attention network, namely Point Transformer, was designed for the semantic segmentation of point clouds [17].

Although building points contain 3D geometric information that can be further reconstructed into 3D models of

buildings, extraction from a point cloud is still a challenging task. Hence, several studies [9,48,49,50,51] have attempted to extract buildings based on point clouds with 2D information as auxiliary data (i.e., 2D-3D information fused methods). While small improvements in accuracy can be obtained, these methods require both the point cloud and auxiliary data such as DOM, DTM, and NIR image which are difficult to meet in practice, making the application scenario very limited. Moreover, the process of registering the auxiliary data to a point cloud inevitably generates registration errors which increase the risk of error propagation in the building extraction [9].

Regardless of the type of data used, the aforementioned studies can be divided into unsupervised and supervised methods depending on whether ground truth is required. In general, supervised methods perform better than unsupervised ones, but many ground truths and hours are needed to precisely model the feature distribution to final target classes. Both unsupervised and supervised methods face the problem that the geometric structure and boundaries of the extracted buildings are incomplete, requiring larger manual editing and trimming efforts. For mesh with complex data structure, ground truth does not exist in the actual production and application process, and its annotation is difficult and inefficient.

For this reason, this study focuses on the use of unsupervised methods to extract buildings. Building extraction from 3D information in an unsupervised way tends to encompass two major components: segmentation and building detection [14,15,19,52,53,54,55,56].

2.2. 3D segmentation

In the last decades, extensive studies have been done to improve the efficiency and robustness of 3D segmentation, which can be roughly categorized into geometric fitting-based, RG-based, and clustering-based methods.

In the geometric fitting-based methods, the most widely employed strategies are RANSAC and HT which have been proven to fit shapes successfully in 2D as well as 3D [34]. Many variants of RANSAC or HT have been proposed for segmentation [31,34,35,57,58]. For example, an efficient RANSAC was developed to fit multiple geometric shapes, even in the presence of up to 50% outliers [34]. Chen [35] proposed an improved RANSAC with localized sampling which significantly improves the segmentation efficiency. An efficient multi-resolution method was presented to segment a point cloud into planner components by combining RANSAC and HT [58]. Although these methods segment point clouds into a set of planes with a relatively high level of success, a large computational cost and high memory consumption are caused by the iteration process of using robust estimators or the voting process [59].

RG is another common approach. For RG, the consistency of normal vector, curvature of points or surfaces, and texture are commonly used growing criteria [13,31,60,61,62,63]. To improve the robustness, more accurate normal vectors were estimated by fast minimum covariance determinants based on robust principal component analysis (PCA) [63]. Xiao [62] presented a subwindow-based RG and a hybrid RG for plane segmentation in structured and unstructured environments, respectively. For efficiency improvement, an adaptive Octree-based RG was introduced by incrementally grouping adjacent voxels with similar saliency features [61]. The RG methods using curvature labeling were proposed for the decomposition of 3D triangular meshes by grouping the topologically adjacent mesh elements with the same surface type [64,65]. RG is a robust and fast segmentation method, but it is not always highly accurate due to the sensitivity to the selection of initial seeds [13]. Normally,

this issue can be mitigated by identifying the smallest curvature or surfaces with minimal residual of the plane fitting as seeds [66].

In clustering methods, points having proximity or similarity meeting the acceptable threshold will be identified as one cluster. Euclidean distance, density, normal, and texture are the most adopted clustering criteria [39,40,59,67]. The computational cost of clustering methods is usually higher than those of the above other two types of methods due to the complexity of calculating the similarity as well as the optimization of energy functions [59]. To better segment, complex clustering criteria such as local convexity, local concavity, and fast point feature histograms (FPFH) [68] are intensively applied [69,70,71]. However, this further increases the computational cost. Recently, an over-segmentation algorithm based on supervoxels, namely voxel cloud connectivity segmentation (VCCS), was proposed [71]. For better object boundaries, boundary points are first excluded to generate supervoxels that maintain the shape of target boundaries [72]. Lin et al. [73] proposed an approach that can produce supervoxels with adaptive resolutions and does not rely on initial seeds. The over-segmentation method can produce a set of well-bounded and uniform supervoxels with high efficiency, thus it is widely used as a preprocessing step in the segmentation or classification of point clouds [16,59,67,72,74,75,76].

2.3. Feature detection

After coarsely segmenting 3D information into a set of patches such as planes, clusters, and supervoxels, feature detection is typically needed to identify building patches from vegetation and other objects [13,14,19,52,53]. For this purpose, several patch characteristics, such as position, orientation, roughness, mean elevation, angular, and density, were applied based on region growing segmentation [54,55]. Mongus et al. [41] regarded buildings as large above-ground objects with planar surfaces and identified them by regional analysis considering geometric, surface, and regional attributes. Furthermore, the penetrating feature was employed because a laser beam hardly penetrates the building [19]. Gilani et al. [13] accounted that the non-building region is usually small in size and has a high concentration of sharp feature points. The computationally-efficient slicing methods were introduced for detecting overall faced and window boundary points based on a local density analysis technique [30,56]. In summary, these studies perform a regional analysis for each patch based on certain manually defined characteristics to identify buildings.

Another unsupervised solution is to detect building objects by global energy optimization based on probability theory. For example, Yang et al. [15] presented a marked point process for extracting buildings utilizing the Gibbs energy model and a reversible jump Markov chain Monte Carlo algorithm. Zhu et al. [16] proposed a high-order Markov random field framework for point classification based on multi-level semantic relationships including point-homogeneity, supervoxel-adjacency, and class-knowledge constraints. Compared with regional analysis methods, the methods based on energy function have higher accuracy. However, heavy computational costs are required during the energy function optimization.

3. Methodology

3.1. Overview

The proposed bidirectional greedy framework for building extraction is based on three basic assumptions: 1) the main (not the whole) building geometric structure consists of planes; 2) the color of leafy trees is close to green; 3) the above-ground objects are topologically separated after ground primitives filtering. The framework is

composed of two greedy processes in opposite directions, i.e., greedily culling of non-building objects based on ensuring the integrity of the main building geometric structure without caring that part of building primitives is removed mistakenly, and afterward, greedily recovering the mistakenly removed building primitives based on the main geometric structure primitives using topology without caring that part of the non-building primitives are recovered mistakenly. A detailed block diagram of the proposed framework is depicted in Fig. 2 with involved algorithms as well as illustrations of intermediate results.

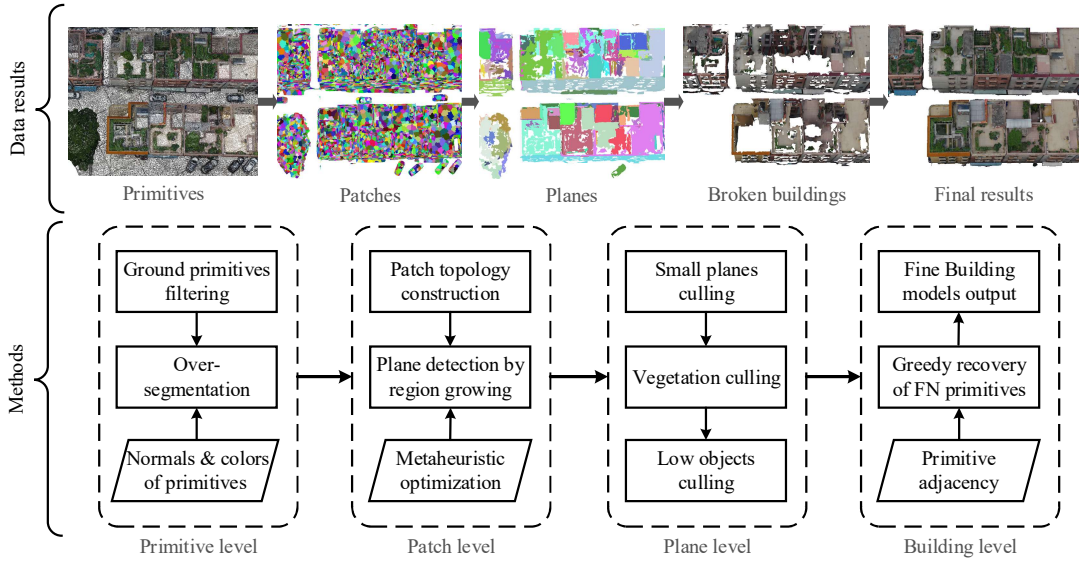


Fig. 2. Workflow of the proposed framework for building extraction. Notation: FN: false negative.

The framework consists of four different levels: primitive, patch, plane, and building levels. At the primitive level, the ground primitives are first filtered by a ground filter (e.g., CSF4Geomesh) [77] for the topological separation of building primitives from other primitives of above-ground objects. After that, the above-ground primitives are further over-segmented to produce patches with uniform geometric and color properties. These patches will replace the primitives as the basic entities for subsequent operations, achieving two advantages: First, the computational efficiency is improved because the number of patches is significantly lower than the primitives. Second, plane detection (c.f. Section 3.3.2) based on patches is more likely to generate results with regular shapes.

Although a building is not required to be composed entirely of planes according to Assumption 1, the planar feature is necessary to identify buildings. Thus, the parameter-free planar detection based on patches is designed to find planar structures that shape buildings at the patch level. The detection method maximizes the utilization of topology among patches and improves its automation and performance by coupled metaheuristics.

After the above parameter-free plane detection, all planar features are extracted, not only from buildings, but also from tree canopies, city furniture, cars, and so on. Extracting building planes directly from these planes inevitably leads to vestigial building in the results. Therefore, at the plane level, instead of extracting buildings directly, we greedily cull non-buildings even if some of the building planes are also excluded.

The plane level seeks to maximize the culling of non-building objects while preserving the main planes of buildings. As a consequence, a portion of the building primitives is inevitably culled. At the building level, we

present a greedy recovery method for culled building primitives based on the explicit topological adjacency of primitives. Finally, 3D building models were extracted with high accuracy and integrity.

3.2. Primitive over-segmentation

An over-segmentation method is designed to segment geomeshes into meaningful patches with homogeneous geometric and textural properties. The details of the over-segmentation are as follows.

A geomesh without ground primitives G is defined as a tuple $\{V, F\}$ of vertices $V = \{v_i | v_i \in \mathbb{R}^3, 1 \le i \le m\}$, and primitives $T = \{t_i = (v_j, v_k, v_l) | v_j, v_k, v_l \in V, j \ne k, k \ne l, l \ne j, 1 \le i \le n\}$, which are usually triangles, but can also include other types of planar polygons. An over-segmentation Σ of G is the set of patches, i.e., $\Sigma = \{M_0, ..., M_{k-1}\}$, induced by a partition of T into k disjoint sub-sets for minimizing the homogeneity criterion function $J(\cdot)$ of patches under a set of constraints (i.e., $\bigcup_{i=0}^{k-1} M_i = G$).

The criteria J for deciding which primitives belong to the same patch is the most important factor affecting the result, and it is computed according to Eq. 1.

$$J(r_{ij}) = \sum_{i=1}^{n} \sum_{j=1}^{n} r_{ij} D(t_i, t_j), \quad (1)$$

where, $D(\cdot)$ is used to measure the homogeneity between any two primitives as per Eq. 2,

$$D(t_i, t_j) = \mu_e D_e(t_i, t_j) + \mu_s D_s(t_i, t_j) + \mu_c D_c(t_i, t_j), \quad (2)$$

where, D_e is the orientation item that controls the geomatic similarity (i.e., cosine similarity) as per Eq. 3; D_s is the L_2 distance norm that controls the spatial difference as per Eq. 4; D_c is the L_2 CIE76 norm that controls the color difference as per Eq. 5; μ_e, μ_s, and μ_c are three weights. For the experiments in this paper, the three weights are respectively set to 1, 0.4, and 0.1.

$$D_e(t_i, t_j) = 1 - |n_i \cdot n_j|, \quad (3)$$

$$D_s(t_i, t_j) = \frac{1}{3} \sum_{q=1}^{3} \left\| t_i^{(q)}, t_j^{(q)} \right\|_2, \quad (4)$$

$$D_c(t_i, t_j) = \left\| lab_i^*, lab_j^* \right\|_2, \quad (5)$$

where n_i is the normal of t_i; $t_i^{(q)}$ denotes the first q vertex of $t_i = (v_j, v_k, v_l)$; lab_i^* is the color of t_i in the CIELab color space. As shown in Fig. 3, the lab_i^* is a mean color of all textural pixels within the range of the UV texture coordinates of t_i, which is calculated by the rasterization algorithm based on scanlines and UV mapping.

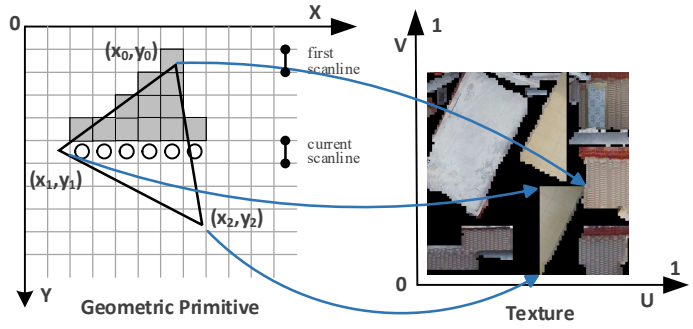


Fig. 3. Illustration of the mean color calculation of geometric primitives.

For minimizing the criteria $J(r_{ij})$ under the constraints, we adopt the efficient energy descent method [73] based on energy function as per Eq. 6.

$$E(r_{ij}) = J(r_{ij}) + |I(\sum_{j=1}^{n} r_{ij}) - \frac{\max(V^x) - \min(V^x)}{d_1} \times \frac{\max(V^y) - m \quad (V^y)}{d_1}|, \tag{6}$$

where, $V^x = \{v_i^{(x)} | v_i \in V\}$, and $V^y = \{v_i^{(y)} | v_i \in V\}$.

3.3. Parameter-free plane detection

In this section, we propose a novel parameter-free plane detection method with a coupling scheme of RG and metaheuristic. As shown in Fig. 4, instead of primitives, we use patches as the basic units for plane detection based on an efficient RG that incrementally groups adjacent patches having similar features into a plane set. The benefits of using patches have been mentioned in Section 3.2. The utilization of patchwise topology based on the adjacency graph constructed in Section 3.3.1 allows a rapid search for adjoining patches, further improving efficiency and performance compared to K-nearest neighbor algorithm (KNN) which is computationally intensive and often results in topologically disconnected being assigned to the same plane. The requested parameters are automatically configured toward the best detection result by a coupling mechanism named self-adaptive Jaya [78] for plan detection and metaheuristics proposed in Section 3.3.3, thus improving the automation and performance.

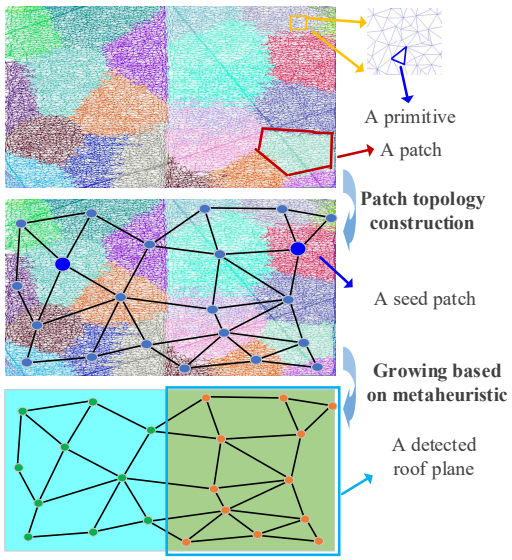


Fig. 4. Illustration of the parameter-free plane detection method.

3.3.1. Patchtwise adjacency graph construction

KNN is commonly used to find neighbors of primitives in feature detection, however, the computation of KNN leads to high computational complexity. Moreover, since topology is not considered, KNN often results in topologically disconnected entities that are considered to belong to different planes being assigned to the same plane. For instance, if KNN is used, patch A in Fig. 5 will be assigned to a plane with patches B and C which are disjointed from it.

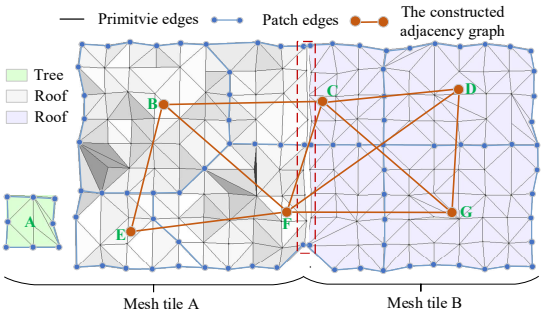


Fig. 5. Illustration for the topology of primitives and planes in the case of two adjacent building roofs and a neighboring tree: A-F denote different patches; the central red dashed lines denote disjoint topology between mesh tile A and mesh tile B.

For meshes, KNN is not necessary because the topology between primitives can provide the adjacency graph. However, to improve visualization efficiency, geomeshes are often sliced so that the adjacency is disconnected between different tiles as shown by the red dashed line in Fig. 5. For this reason, we propose a method for constructing patchwise adjacency graphs. The construction problem of the adjacency graph for patches is defined as follows: for each patch $M_i \in \Sigma$, find the index of all patches adjacent to it. Here we use the indexes of primitives in T to denote the primitive set T_i of M_i. First, the set of all primitives adjacent to a patch M_i is computed as per Eq. 7,

$$A_i = \{\cup_{a \in T_i} A(t_a) \,|t_a \in T\}, \tag{7}$$

where function $A(\cdot)$ is used to get all the primitives adjacent to t_a based on a criterion J_1 (as per Eq. 8) that two adjacent primitives have at least one common vertex, as per Eq. 9:

$$J_1(t_a, t_b) = \begin{cases} 1, & \text{if } \exists\ 1 \le i, j \le 3 \wedge a \in T_i \wedge 1 \le b \le n, t_a^{(i)} = t_b^{(j)} \\ 0, & \text{else} \end{cases}, \tag{8}$$

$$A(t_a) = \{t_b \in T | J_1(t_a, t_b) = 1\}, \tag{9}$$

where $t_a^{(i)}$ is the i-th vertex of $t_a = (v_j, v_k, v_l)$. Then, A_i is reassigned as the adjacent patch set of M_i following Eq. 10,

$$A_i = \{l_t | t \in A_i\}, \tag{10}$$

where l_t denotes the index of the patch to which primitive t belongs; A_i is a collection without repetitions. Orange lines in Fig. 3 show how the adjacent graph is constructed for patches A-G according to Eq. 10. Unlike KNN, patch A will not be treated as a neighbor of B and C in Fig. 5.

3.3.2. Patch-based region growing

Here a plane detection ψ of Σ is the set of patches $\psi = \{R_0, ..., R_{m-1}\}$ induced by a partition of Σ into m plane sub-sets and a not-plane sub-set R_m following a growing criterion $J_2(\cdot)$, where $\cup_{i=0}^{m} R_i = \Sigma$, and $R_i \cap R_j = \emptyset, i \neq j$. As shown in Fig. 4, each plane $R_i \in \psi$ is detected as follows: 1) pick a seed patch M_i from Σ; 2) find neighbors of M_i by the constructed adjacent graph constructed; 3) include those neighbors which satisfy the growing criterion $J_2(\cdot) = 1$ as per Eq. 11; 4) repeat the step 2) and 3) for all included neighbors; 5) if no further neighbor satisfies $J_2(\cdot) = 1$, repeat 1)-4) until Σ is empty. In step 5), if the number of patches in the plane does not exceed 3, discard it.

$$J_2(M_a, M_b) = \begin{cases} 1, & \text{if } \arccos\left(\frac{\bar{n}_r \cdot \bar{n}_b}{|\bar{n}_r||\bar{n}_b|}\right) < \theta \wedge \frac{(\bar{v}_b - \bar{v}_a)\cdot \bar{n}_r}{|\bar{n}_r|} < d_2 \\ 0, & \text{else} \end{cases}, \tag{11}$$

where M_a is the seed patch; M_b denote a neighbor of M_a; $\bar{n}_r$ are the mean normal associated with all primitives in current plane R_a; $\bar{n}_b$ are the mean normal associated with all primitives in M_b; $\bar{v}_a$ and $\bar{v}_b$ are the mass centers of M_a and M_b, respectively ; θ is an angle threshold indicating the maximum accepted angle

between the normals associated with M_a and M_b; d_2 is a distance threshold indicating the maximum accepted distance between $\overline{v}_a$ and $\overline{v}_b$. The detected planes are shown in Fig. 4.

3.3.3. Metaheuristic optimization

The optimization problem of plane detection can be formulated as per Eq. 12. In general, the planes that shape a building are approximated as rectangles, so we expect that the detected plane shape is close to rectangular with integrity (i.e., with regular boundaries and no missing primitives inside) and flatness. Thus, we proposed an evaluation function $J_3(\cdot) \in [0,1]$ as per Eq. 13 for the planes generated by θ, d_2.

$$(\theta', d_2') = \max_{\theta, d_2} \{J_3(\theta, d_2) | \theta \in [10^\circ, 45^\circ], d_2 \in [2,20]\}, \quad (12)$$

$$J_3(\theta, d_2) = 0.5 \sum_{i=0}^{m-1} \omega_i \, (C_f(R_i) + C_i(R_i)), \quad (13)$$

where θ' and d_2' are optimal values searched in empirical intervals; ω_i is a weight coefficient defined by the ratio of the area of R_i to the total area of ψ induced by θ and d_2; $C_f(\cdot)$ and $C_i(\cdot)$ measure the flatness and integrity of a plane, respectively.

$C_f(R_i)$ measures the flatness of R_i by the principal PCA applies to the set of triangles in R_i and is computed as per Eq. 14. The main idea of PCA is to compute the covariance matrix of triangle set in R_i, perform its eigendecomposition and then determine the top 3 eigenvalues. The greatest eigenvalue λ_1 lines on the first eigenvector in which the data has a strong component, the second eigenvalue λ_2 lies on the second eigenvector, and so on.

$$C_f(R_i) = 1 - \frac{\lambda_{R_i}^3}{\lambda_{R_i}^2}, \quad (14)$$

where $\lambda_{R_i}^2$ is the 2nd eigenvalue of R_i, and $\lambda_{R_i}^3$ is the 3rd eigenvalue.

We derive the closed form of covariance matrix C_{R_i} for plane R_i consisting of l trianglesas per Eq. 15.

$$\begin{aligned} C_{R_i} &= \int_{R_i} (x_{R_i} - c_{R_i})(x_{R_i} - c_{R_i})^T dx_{R_i} \\ &= \int_{R_i} x_{R_i} x_{R_i}^T - c_{R_i} x_{R_i}^T - x_{R_i} c_{R_i}{}^T + c_{R_i} c_{R_i}{}^T dx_P \\ &= O_{R_i} - c_{R_i} \int_{R_i} x_{R_i}^T dx_{R_i} - \int_{R_i} x_{R_i} dx_{R_i} c_{R_i}{}^T + c_{R_i} c_{R_i}{}^T \int_{R_i} dx_{R_i} \\ &= O_{R_i} - m_{R_i} c_{R_i} c_{R_i}^T \end{aligned}, \quad (15)$$

where O_{R_i} is the second order moment of R_i with respect to origin and is computed as per Eq. 16; m_{R_i} is the area of plane R_i; c_{R_i} is the center of mass of R_i.

$$O_{R_i} = \sum_{j=1}^{l} \frac{m_{t_j}}{m_s} \left(A_{t_j} O_s A_{t_j}^T \right), \quad (16)$$

where $s = \{(1,0,0), (0,1,0), (0,0,1)\}$ is a standard triangle; m_{t_j} is the area of triangle t_j; A_{t_j} is an affine transformation matrix that maps the vertex coordinates on s onto t_j and is given as per Eq. 18

$$A_{t_j} = \begin{bmatrix} v_1^{(x)} & v_2^{(x)} & v_3^{(x)} \\ v_1^{(y)} & v_2^{(y)} & v_3^{(y)} \\ v_1^{(z)} & v_2^{(z)} & v_3^{(z)} \end{bmatrix}, \quad (17)$$

where v_1, v_2, and v_3 are the three vertices of t_j. O_s is the second order moment of s with respect to origin and is given as per Eq. 18.

$$O_s = \begin{bmatrix} 1/12 & 1/24 & 1/24 \\ 1/24 & 1/12 & 1/24 \\ 1/24 & 1/24 & 1/12 \end{bmatrix}. \quad (18)$$

We measure the integrity $C_i(R_i)$ of R_i by an area ratio of R_i to the largest face of its oriented bounding box (OBB) as per Eq. 19. As shown in Fig. 6a if the plane R_i is intact, $C_i(R_i)$ is equal to or slightly greater than 1. In contrast, if R_i is mutilated (i.e., misses some triangles), $C_i(R_i)$ is less than 1, and the more mutilated the plane is, the smaller the ratio is.

$$C_i(R_i) = \min\left\{\frac{\sum_{j=1}^{l} m_{t_j}}{\max\{a_i \times b_i, b_i \times c_i, c_i \times a_i\}}, 1\right\}, \tag{19}$$

where, a_i, b_i, and c_i are the lengths of the three mutually perpendicular sides of OBB of R_i.

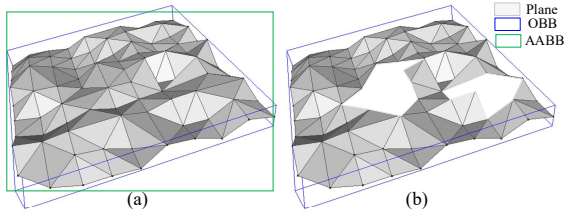


Fig. 6. Illustration for (a) intact and (b) mutilated planes and their respective oriented bounding boxes (OBBs).

As shown in Fig. 16a, differently from an axis-aligned bounding box (AABB), OBB provides a tighter cuboid with an orientation adapted to the shape of the object. However, the minimal OBB is difficult to compute. Here, we use an efficient method to compute an approximate minimal OBB for a plane. First, we compute the eigenvectors of plane R_i by eigendecomposition of the covariance matrix using Eq. 15. Then, these eigenvectors are used to transform the vertexes to the origin so that the eigenvectors corresponding to the principal axes. Finally, we compute the center, min, and max of the diagonal.

The solution of Eq. 13 based on the proposed criterion function $J_3(\cdot)$ can be optimized by a metaheuristic algorithm, which relies on an approximation strategy to find "best available" result within an acceptable computational cost. Here, the metaheuristic we choose is self-adaptive Jaya [78]. It should be noted that self-adaptive Jaya is not the only choice, but can also be replaced by other metaheuristic algorithms such as ant colony methods and genetic algorithms.

3.4. Greedy culling of non-building planes

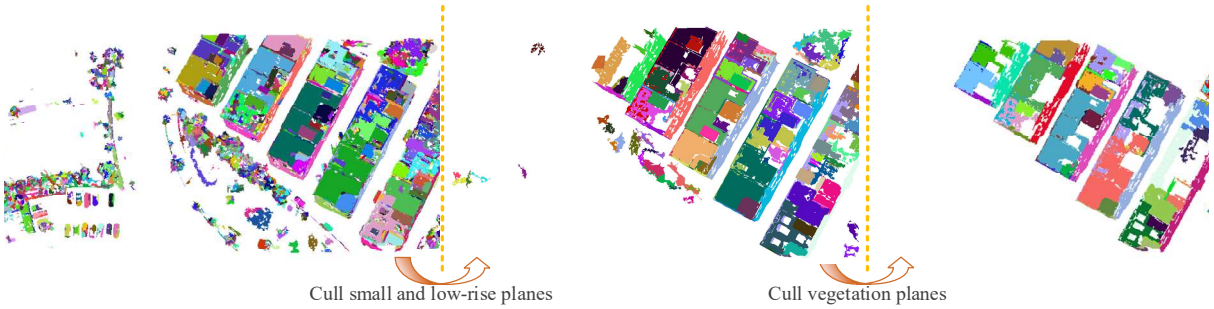


Fig. 7. Illustration for greedy culling of non-building planes.

3.4.1. Culling of small and low-rise planes

As shown in Fig. 7, there are some small and low planes in ψ, which are usually not from buildings but small or low-rise objects such as part of vegetation, cars, and city furniture. To obtain the set ψ' containing only

building planes, we first cull small planes as per Eq. 20 by specifying an area threshold d^2 that is smaller than the area of most planes belonging to the roof or façade of buildings.

$$\psi' = \{m_{R_i} \geq d^2 | R_i \in \psi\} \quad (20)$$

Then, we further cull low-rise planes by a height threshold d_3 and update ψ'. d_3 is a height above ground level (HAGL) measured with respect to the underlying ground surface. Planes with weighted HAGLs below this threshold are culled as per Eq. 21, where h_{t_j} denotes the HAGL of triangle t_j. h_{t_j} is calculated by subtracting the mean height of nearby ground triangles searched by KNN from the mean height of t_j. In most situations, the heights of low-rise plane object such as cars, urban furniture, pedestrians, and shrubs do not take values greater than 2 m, so d_3 is set to 2.

$$\psi' = \{\frac{\sum_{t_j \in R_i} m_{t_j} h_{t_j}}{m_{R_i}} \geq d_3 | R_i \in \psi'\}. \quad (21)$$

3.4.2. Culling of vegetation planes

Although the small and low-rise vegetation has been culled as per the workflow step described in Section 3.4.1, it is likely that there are still large and high-rise tree canopies. Based on Assumption 3 (cf. Section 3.1), these canopy planes are culled by a color index (i.e., excess green minus excess red). The colors of the triangle have been calculated in Section 3.2. ψ' is further updated for culling tree canopies as per Eq. 22, where rgb_{R_i} is the color of R_i, and d_4 is a threshold beyond which the plane is considered as vegetation.

$$\psi' = \{3rgb_{R_i}^{(g)} - 2.4rgb_{R_i}^{(r)} - rgb_{R_i}^{(b)} < d_4 | R_i \in \psi'\}. \quad (22)$$

The value of d_4 is automatically set based on the maximum interclass difference method based on an analysis of the histogram resulting from the color index calculation. The optimal value is given by exhaustively searching for d_4 that maximizes the inter-class variance between vegetation and other objects.

3.5. Greedy recovery of building primitives

According to Assumption 3 (cf. Section 3.1), the above-ground objects are topologically separated after ground primitives are filtered out. Therefore, the neighbors topologically adjacent to the main structure of the building do not contain non-building primitives. With this feature, we present the greedy recovery method of building primitives. The method greedily searches for the neighbors adjacent to ψ' in culled objects by topological accessibility and marks them as building primitives.

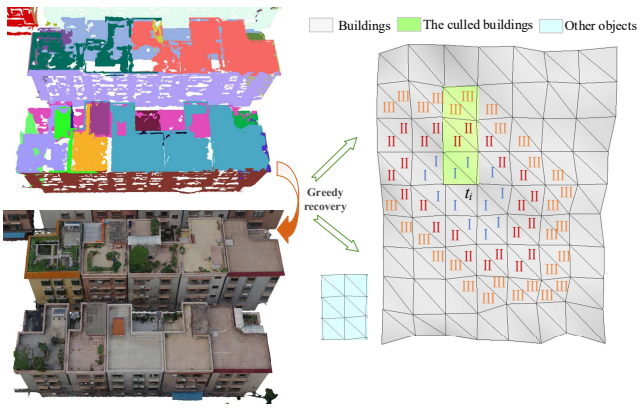


Fig. 8. Illustration for greedy recovery of building primitives based on topology. Notation: t_i is a primitive; I, II, and III are the first-, second-, and third- neighbors of t_i, respectively.

Topological accessibility between primitives a and b is defined as a vertex of b to be reached from a vertex of a after traveling through edges of zero or more intermediate primitives. Let T_b is the set of primitives forming ψ', T is the set of above-ground primitives, and T_r is the final result considering structural integrity. For each $t_i \in T_b$, the algorithm searches for all primitives in T that it can topologically access, and mark these primitives as buildings to be added to the final building set T_r. Specifically, all primitives to be reached from t_i are obtained by ceaselessly finding first- and multi-order neighbors of it according to Eq. 8. As shown in Fig. 8, the primitives of types I, II, and III are the first-, second-, and third-order neighbors of t_i, respectively. These neighbors including the broken (i.e., grey triangles in Fig. 8) and the wrongly culled (i.e., green triangles in Fig. 5) building parts will be added to T_r, while the other objects (i.e., cyan triangles in Fig. 8) such as trees will not be added because they are topologically isolated from the broken building parts.

Algorithm 1: Greedy recovery method

Input: Above-ground primitives set T; Broken building set T_b; Adjacent funtion adjacentTs();
Output: Recovered primitive set T_r

```
1  T_r ← ∅;
2  Let S be a stack;
3  Let V be a index set of visited vertices;
4  foreach i in T_b do
5  |  S.push(i);
6  end
7  repeat
8  |  t ← S.top();
9  |  S.pop();
10 |  if T_r.find(t) ≠ true then
11 |  |  T_r.insert(t);
12 |  |  for j = 1 to 3 do
13 |  |  |  v ← t^j;
14 |  |  |  if V.find(v) ≠ true then
15 |  |  |  |  V.insert(v);
16 |  |  |  |  T_v ← adjacentTs(T, v);
17 |  |  |  |  foreach w in T_v do
18 |  |  |  |  |  S.push(w);
19 |  |  |  |  end
20 |  |  |  end
21 |  |  end
22 |  end
23 until S is empty;
24 Return T_r;
```

Fig. 9. Pseudocode for greedy recovery.

Based on the depth-first search strategy, the method is implemented as per Algorithm 1 displayed in Fig. 9. In the input list, the function *adjacentTs*(T, v) is used to determine the primitives formed by vertex v from the above-ground primitives set T. In addition, to prevent the non-building primitives in T from being widely mis-recovered due to the presence of a few non-building primitives in T_b that are not culled, we split T uniformly into slices with side length d_5. Then, the algorithm finds all topologically accessible primitives of t_i in the slice where t_i is located, thus effectively mitigating this problem. In most cases, modeling software for visualization purposes run the slicing step automatically, so no extra computational costs are requested.

4. Experiments

4.1. Evaluation metrics

To quantitatively evaluate the performance of the proposed framework, we adopt four standard metrics: *accuracy*, *precision*, *recall*, and F1-score (*F1*). These metrics can be calculated using the following formulas:

$$accuracy = \frac{TP+TN}{FP+FN+TP+TN}, \quad (24)$$

$$precision = \frac{TP}{TP+FP}, \quad (25)$$

$$recall = \frac{TP}{TP+FN}, \quad (26)$$

$$F1 = 2 \times \frac{precision \times recall}{precision + recall}, \quad (27)$$

where TP, FP, FN, and TN are true positive, false positive, false negative and true negative classifications, respectively.

4.2. Datasets

Five experiments with different datasets are reported in this section. The first and second experiments were performed using the high-resolution benchmark dataset (H3D) captured with LiDAR over Hessigheim in Germany [79], while the remaining ones were done using a high-resolution dataset (named Gm4B) collected by oblique photogrammetry. The geomeshes used in all five tests have no ground primitives.

The geomeshes in the H3D dataset were produced by point clouds and images collected by a LiDAR and two cameras mounted on a UAV. Because the point cloud was collected using LiDAR, the building façades include fewer valid points and less geometric accuracy, especially in dense building areas. The first experiment used a geomesh which integrates the meshes identified with ID 51392. Similarly, the meshes identified with ID 51398 were used to perform the second experiment.

Gm4B was collected by 5-view oblique photogrammetry from a suburb that contains high-density residential areas, schools, trees, etc. Compared with H3D, Gm4B has higher geometric accuracy, finer texture, and higher topological quality. Gm4B was first filtered off-ground primitives by CSF4Geomesh. Then, it was labeled manually. The details of the five geomeshes are listed in Table 1.

Table 1. The details of the geomeshes used in the experiments.

Geomesh ID	*Vertex Number*	*Primitives Number*	*Box Size (m)*	*Tile Number*	*Tile Size (m)*
H3D_51392	841257	1644517	52*468*25	16	53
H3D_51398	806644	1564992	57*468*24	27	53
Gm4B_1	281732	5215771	94*288*24	781	8
Gm4B_2	3077862	5694838	65*406*23	923	8
Gm4B_3	2694588	4935134	65*389*62	1115	8

4.3. Experimental setup

The proposed bidirectionally greedy framework was coded in C++17, and the experiments were run in a Linux system on an Intel Core i7-10700 CPU with 32 GB memory. Seven user-specified parameters are required for

our framework: three over-segmentation weights (i.e., μ_e, μ_s, μ_c, and d_1), area threshold d^2, height threshold d_3, and slice length d_5. The values of the parameters are shown in Table 2.

Table 2. Parameter setting of the experiments.

Geomesh ID	*Primitive level*				*Plane level*		*Building level*
	μ_e	μ_s	μ_c	d_1 (m)	d^2 (m^2)	d_3 (m)	d_5 (m)
H3D_51392	1	0.4	0.1	1	10	2	53
H3D_51398	1	0.4	0.1	1	10	2	53
Gm4B_1	1	0.4	0.1	1	10	2	8
Gm4B_2	1	0.4	0.1	1	10	2	8
Gm4B_3	1	0.4	0.1	1	10	2	8

4.4. Results

The evaluation metrics of the five tests on a per-primitives level listed in Table 3 highlight very promising building extraction results obtained with the proposed framework. Almost all the buildings were extracted correctly with a particularly high *recall*, even close to 99% on Gm4B_2 and Gm4B_3. For the overall classification performance, the framework achieved more than 86% *accuracy*.

The visualized results are shown in Fig. 10-15, from which it can be seen that, unlike other methods that extract 2D footprints or 3D geometry points of buildings from images or point clouds, our method extracts, spatial-continuous, structural-complete, fine-textured 3D building models with various shapes from geomeshes. Typical extraction results are shown in Figure 10, including bungalows, multi-story buildings, high-rise buildings, attached buildings, and agricultural buildings.

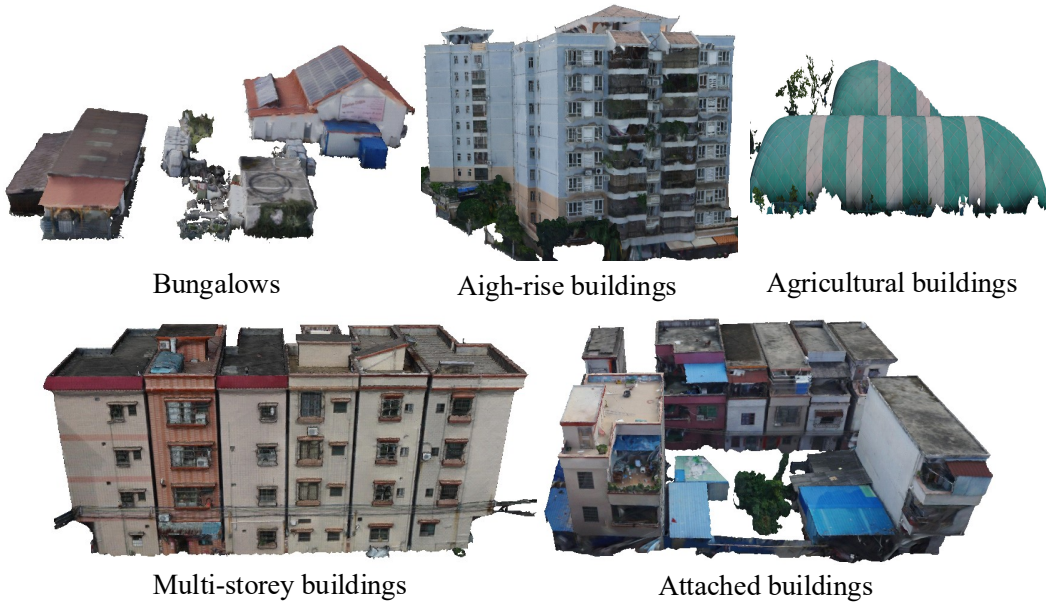


Fig. 10. Typical buildings extracted by the proposed method.

Table 3. Results of quantitative performance assessment of the proposed framework.

Geomesh ID	*TP*	*FP*	*TN*	*FN*	*recall*	*precision*	*accuracy*	*F1*
H3D_51392	713961	176,918	709,456	44082	0.942	0.801	0.866	0.866
H3D_51398	255378	186996	1094663	27955	0.901	0.577	0.862	0.704
Gm4B_1	1991715	305140	2,816,965	101951	0.951	0.867	0.923	0.907
Gm4B_2	3309678	149346	2198409	37405	0.989	0.957	0.967	0.973

Gm4B_3	2934452	185,257	1,779,828	35597	0.988	0.941	0.955	0.964

As revealed in Table 3 and Fig. 11-15, the framework performs better on Gm4B than on the H3D. Three factors can explain this result. First, the number of building primitives in H3D is extremely few and significantly lower than those of non-building, thus resulting in some of the metrics being too low even if a small part of buildings is missed, especially on the H3D_51398. Second, the geometric and topological quality of H3D is worse than that of Gm4B, especially the side elevation of the building, because of the difference in collection and production means. Third, H3D is a rural scene in which the buildings are of low height and topologically connected to a number of trees.

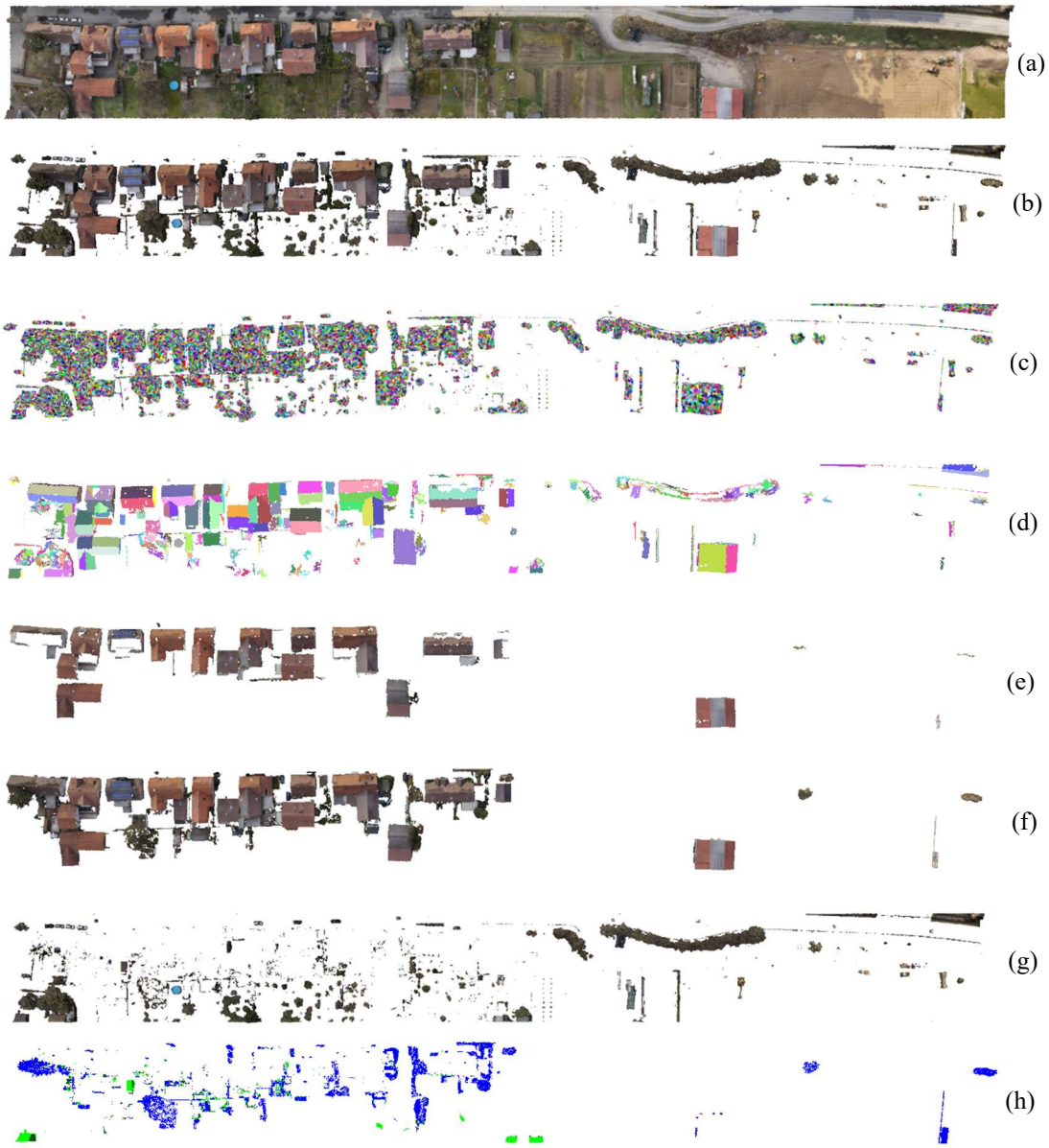


Fig. 11. Detailed visualization of extraction results for H3D_51392: (a) geomesh; (b) geomesh without ground primitives; (c) patches generated by over-segmentation; (d) planes generated by parameter-free plane detection; (e) geomesh after greedy culling; (f) extracted buildings; (g) extracted non-buildings; (h) FP primitives (blue) and FN primitives (green).

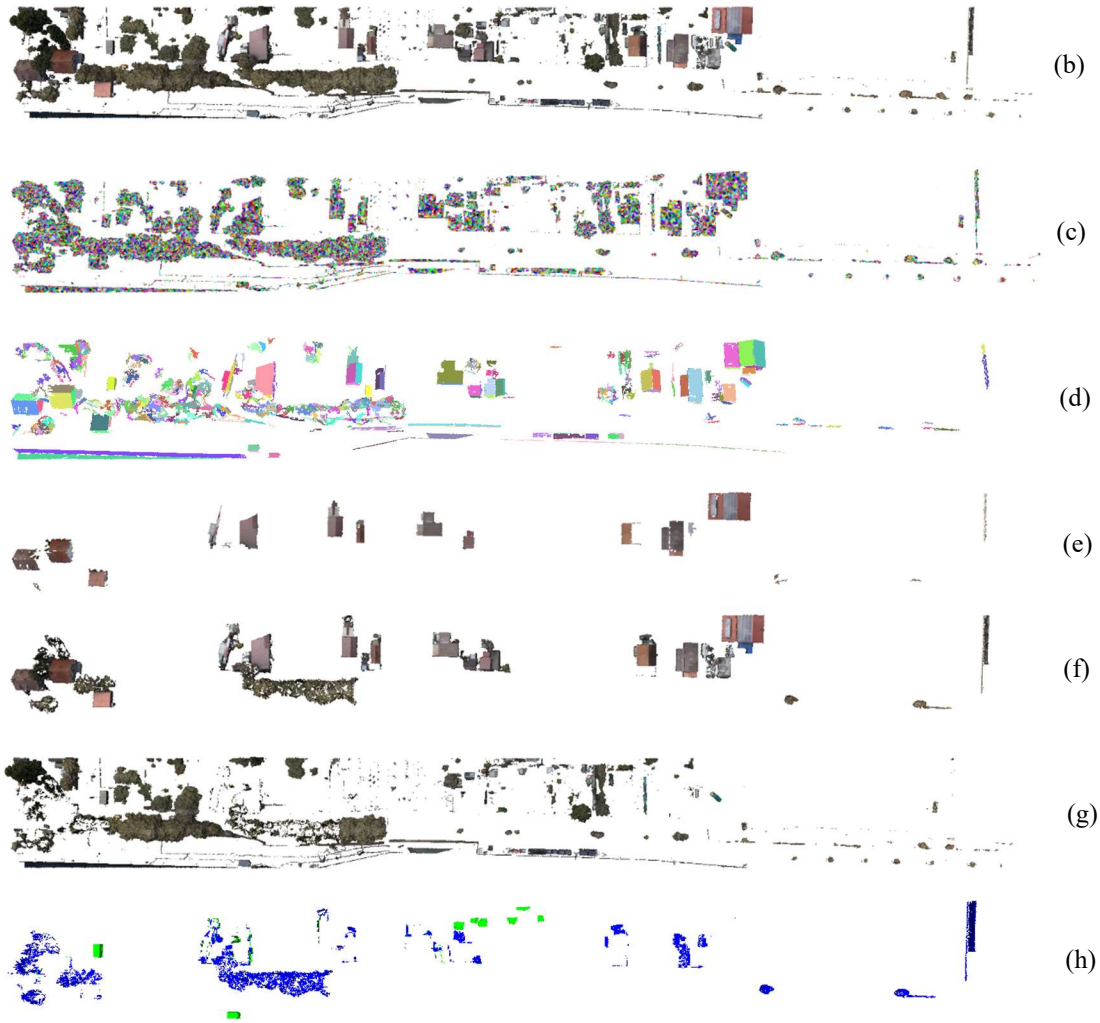
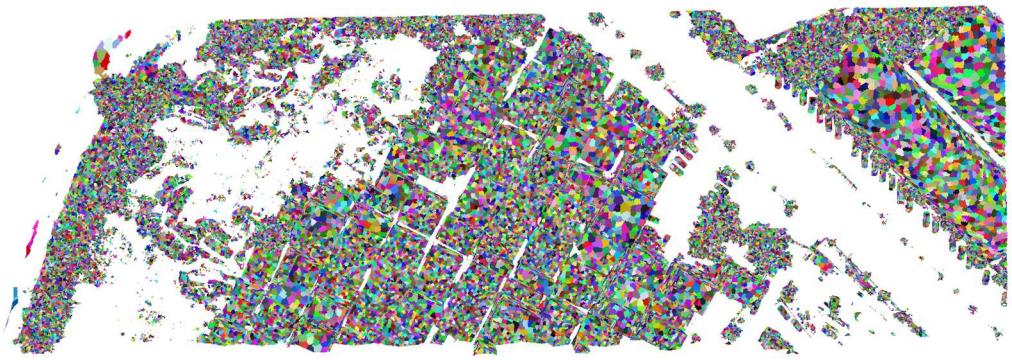


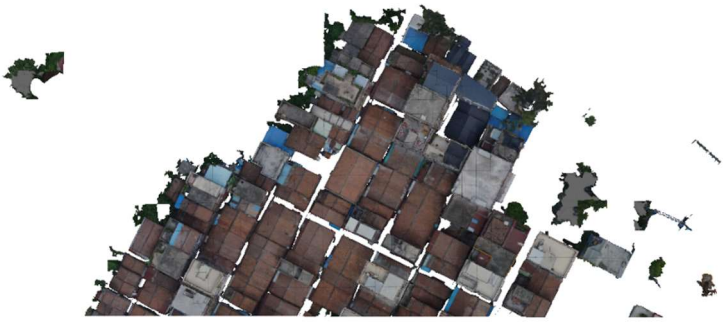
Fig. 12. Detailed visualization of extraction results for H3D_51398: the figure captions for (a)-(f) are the same as those in Figure 11.



(c)

(d)

(e)

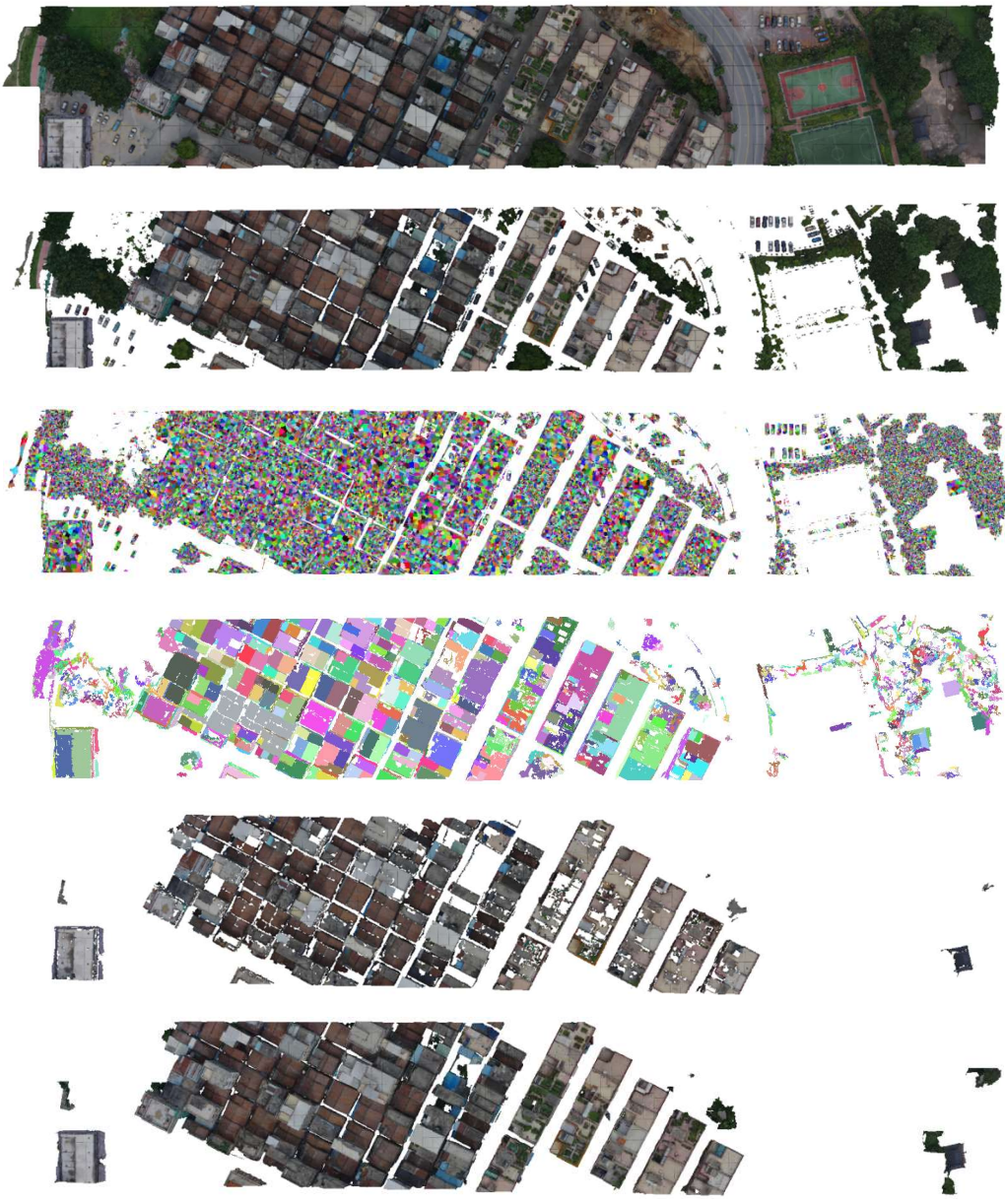


(f)

(g)

(h)

Fig. 13. Detailed visualization of extraction results for Gm4B_1: the figure captions for (a)-(f) are the same as those in Figure 11.



(a)

(b)

(c)

(d)

(e)

(f)

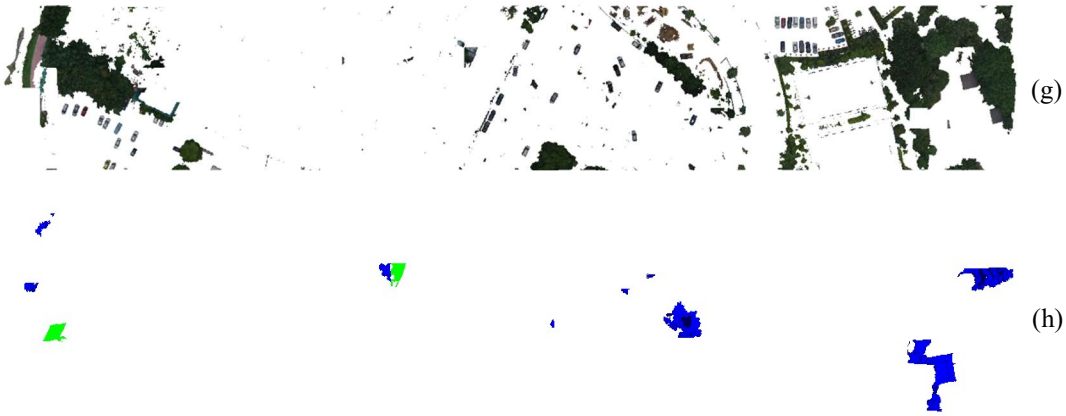
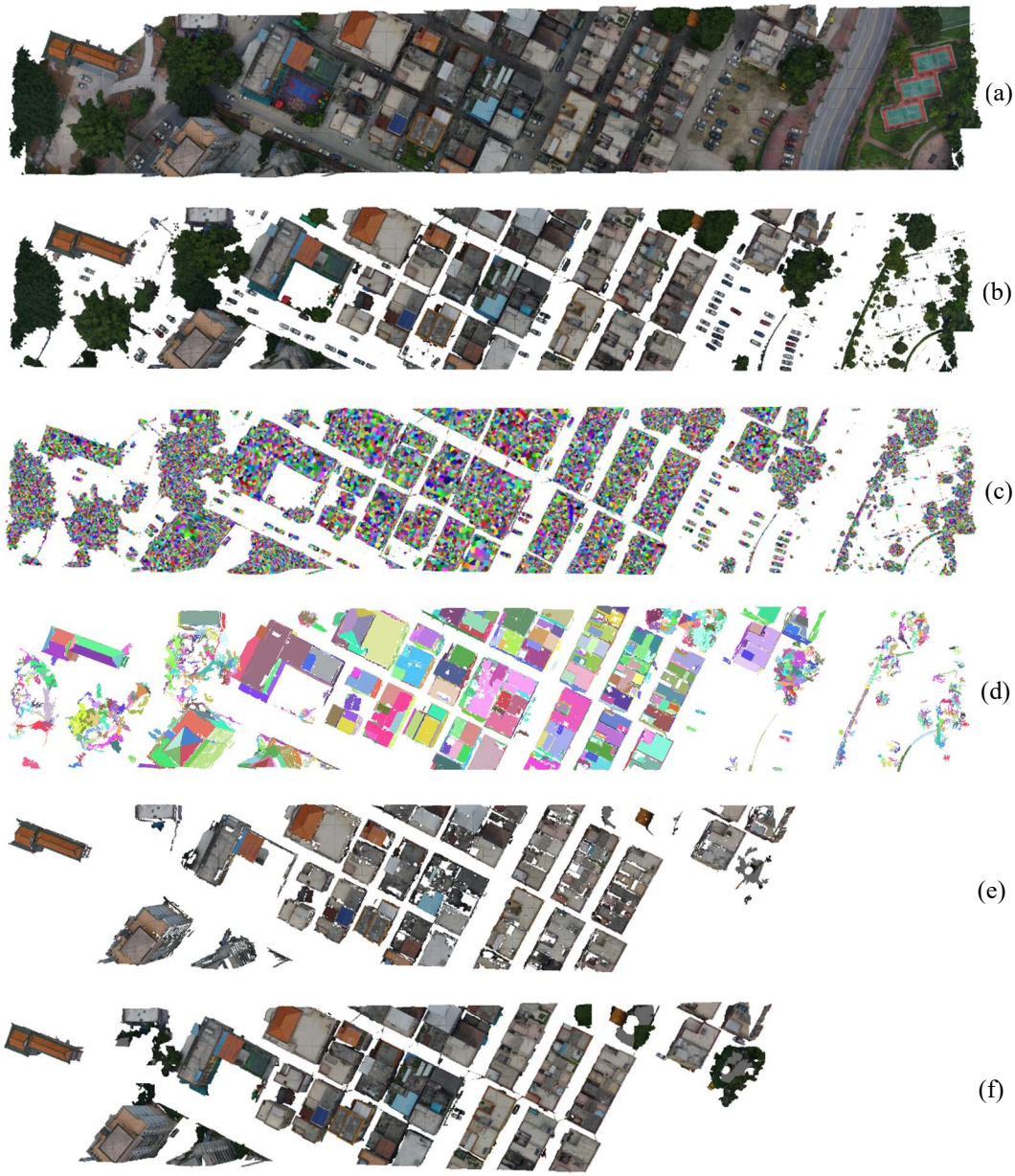
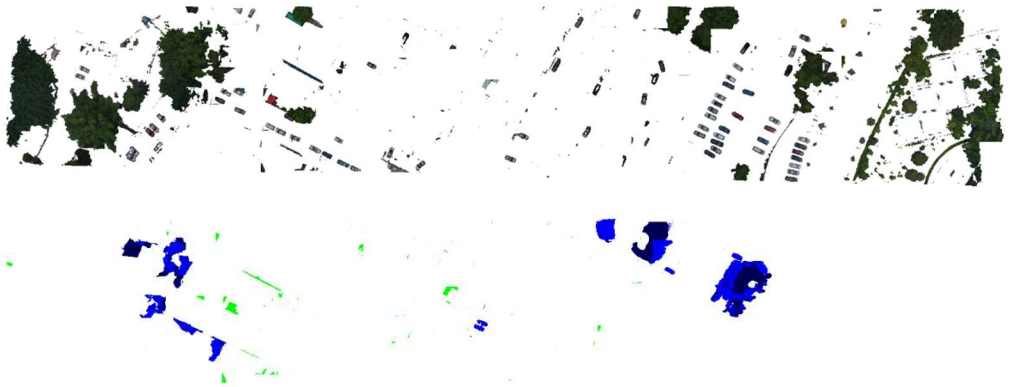


Fig. 14. Detailed visualization of extraction results for Gm4B_2: the figure captions for (a)-(f) are the same as those in Figure 11.





(g)

(h)

Fig. 15. Detailed visualization of extraction results for Gm4B_3: the figure captions for (a)-(f) are the same as those in Figure 11.

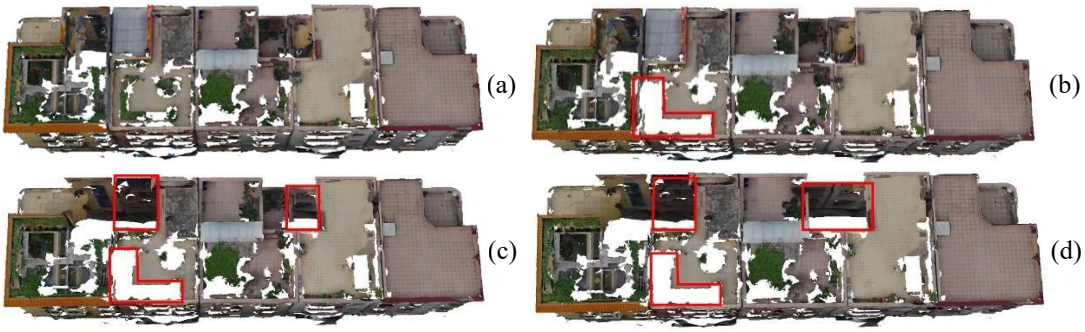
5. Discussion

5.1. Parameter setting

Among the user-specified parameters, the values of patch-level and building-level parameters (i.e., d^2, d_3, d_5) have a relatively large impact on the results, while the default values of over-segmentation parameters (i.e., μ_e, μ_s, μ_c, and d_1) are adapted to suit almost all scenarios. The default values were determined by prior knowledge and a number of tests as follows: $\mu_e = 1, \mu_s = 0.4, \mu_c = 0.1, d_1 = 1\ m, d^2 = 10\ m^2, d_3 = 2\ m$, and $d_5 = 10\ m$.

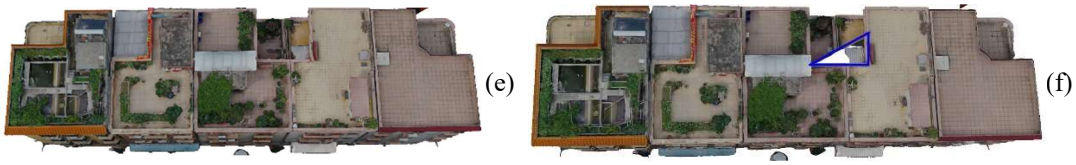
In particular, d_1 controls the mean size of patches. Smaller d_1 would make patches more homogeneous, but it dramatically increases the computing time of plane detection. d^2 and d_3 in the greedy process have a strong relationship with extraction results because they represent the distinguishing threshold between the planes of small or low objects and those of buildings. If d^2 and d_3 are too small, they may lead to incomplete culling of small or low non-building objects. On the contrary, they may cause building primitives being over-culled mistakenly. The suggested ranges for d^2 and d_3 are: $5\ m^2 \leq d^2 \leq 30\ m^2$; $0.5\ m \leq d^2 \leq 2.5\ m$.

As shown in Fig. 16a-d, the larger d^2, the more building primitives are incorrectly culled. As illustrated in Fig. 13e-f, almost all mistakenly culled building planes can be greedily recovered, except for those whose area is much larger than d_5. Therefore, d_5 is usually recommended to be slightly larger than d^2. However, for efficient visualization and quick production, the geomesh is already sliced into tiles in the modeling software, and it is difficult to reconstruct the topology between tiles and re-slice them, so d_5 is usually set to the size of the tiles.



(a) (b)

(c) (d)

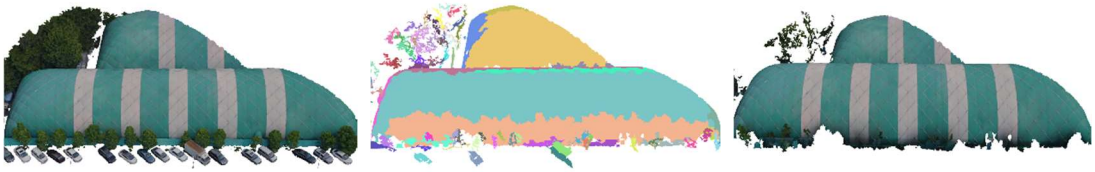


(e) (f)

Fig. 16. Effect of parameter d^2 on extraction results: (a), (b), (c), and (d) show a building after culling planes with an area less than $5\ m^2$, $10\ m^2$, $20\ m^2$, and $40\ m^2$, respectively; (e) the same building after greedy recovery with $d_5 = 10\ m$ and $5\ m^2 \leq d^2 \leq 30\ m^2$; (f) and after greedy recovery with $d_5 = 10\ m$ and $d^2 = 40\ m^2$. Red polygons indicate the planes that are noticeably culled compared to (a); blue polygons indicate the primitives that are missed in the final building.

5.2. Building geometric structure

Since the planar characteristics of buildings are important to consider in the proposed framework, it is usually more beneficial for the detection that buildings are composed of a set of planes in a geomesh. However, buildings in cities have various shapes, and although rare, some may be entirely non-planar, such as agricultural greenhouses. Although a building cannot be completely non-planar according to Assumption 1 (cf. Section 3.1), the proposed method has been tested on a sample of such non-planar buildings. For a completely curved agricultural greenhouse as shown in Fig. 17a, the proposed plane detection method can still find relatively flat areas (Fig. 17b), and the geometric structure (Fig. 17c) remains intact after the culling of small and low planes.



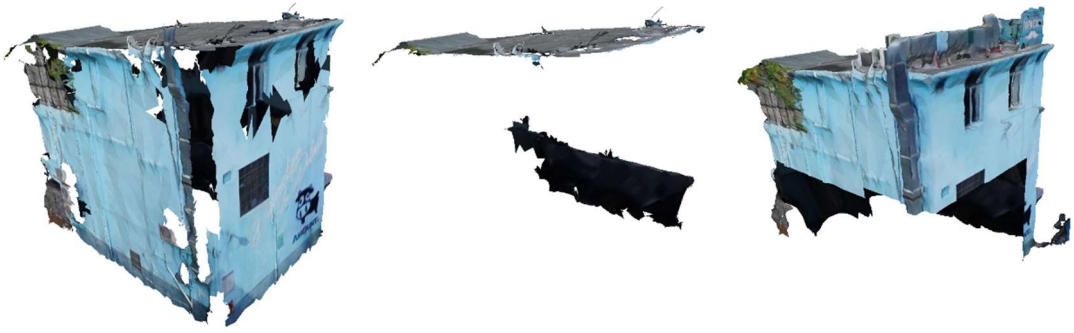
(a) (b) (c)

Fig. 17. Example of the non-planar building at different steps during the extraction process: (a) the building in the original dataset; (b) the planes detected from the building; (c) the building after the culling of small and low-rise planes.

5.3. Building & canopy colors

Since the color index is used to cull vegetation planes during the greedy culling process of non-building objects in the proposed framework, a small fraction of building planes that are very close to the vegetation color will be culled by mistake. For example, after vegetation planes are culled, the façades whose color is close to vegetation are culled (Fig. 18). Similarly, the green greenhouse in Fig. 17c is also culled by mistake as shown in Fig. 13 after the culling of vegetation. Increasing the d_5 or replacing the vegetation culling method can address this rare phenomenon.

Based on the datasets analyzed, it appears that color index is also difficult to adapt to non-green lush canopies such as maple canopies in the fall and grey canopies (fill color for empty textures as shown in Fig. 19). This may result in a small portion of vegetation not being culled cleanly, as illustrated in Fig. 19.



(a) (b) (c)

Fig. 18. Example of building characterized by a color close to vegetation, at different extraction steps: (a) after culling of small and low-rise planes; (b) after culling of vegetation planes; (c) after greedy recovery.

(a) (b) (c)

Fig. 19. Example of green primitives of a tree at different extraction steps: (a) the tree in the original dataset; (b) culling of vegetation planes; (c) after greedy recovery.

5.4. Objects adjacent to buildings

As shown in Fig 11-15, the extracted FPs are mainly non-building primitives that are adjacent to buildings. These primitives are not topologically isolated from the buildings after ground primitives filtering, thus do not fit with Assumption 3 (cf. Section 3.1) so they end up being added during the greedy recovery step of the proposed framework. More details are visualized in Fig. 20. These incorrectly extracted non-building objects can be easily removed with some post-processing methods or manually. In addition, a smaller d_5 is beneficial in decreasing the number of these incorrectly extracted primitives.

Fig. 20. Three extracted buildings and some incorrectly extracted objects adjacent to them (marked by red polygons).

5.5. Efficiency

Ignoring the time of file input and output, the proposed framework and the implementation time for each step are listed in Table 4. Without any parallelism, the computation can be completed within 30 seconds for a mesh of about 1.5 million primitives and within 60 seconds for a mesh of about 5 million primitives. In the framework, over-segmentation takes nearly half of the total elapsed time. It is to be noted that the values of time required to undertake plane detection that is reported in Table 4 refer to the case of specified parameters rather than meta-heuristic optimization because the implementation time of Jaya optimization is random and difficult to count.

For the 5 geomeshes used in the experiments, the implementation times of the observed parameter-free plane detection range between 20 and 60 minutes. The θ found by metaheuristic optimization usually lies between $[28^{\circ},\ 32^{\circ}]$, and d_2 usually lie between [3,8]. After testing, θ and d_2 can take any value within the intervals to obtain an ideal segmentation result. Therefore, the two above-mentioned intervals can be directly used for the recommended values of θ and d_2 to avoid excessive execution time.

Table 4. Consuming time required by each part during the proposed framework.

Geomesh ID	*over-segmentation*	*Plane detection*	*Small planes culling*	*Low planes culling*	*Vegetation culling*	*Greedy recovery*	*Total time*
H3D_51392	9.323 *s*	5.232 *s*	0.300 *s*	2.385 *s*	8.158 *s*	0.646 *s*	25.594 *s*
H3D_51398	8.927 *s*	3.926 *s*	0.021 *s*	1.386 *s*	6.124 *s*	0.23 *s*	20.614 *s*
Gm4B_1	28.153 *s*	14.006 *s*	0.107 *s*	3.475 *s*	5.619 *s*	2.063 *s*	53.423 *s*
Gm4B_2	30.421 *s*	14.747 *s*	0.131 *s*	6.154 *s*	2.947 *s*	3.181 *s*	57.581 *s*
Gm4B_3	26.132 *s*	12.981 *s*	0.112 *s*	5.179 *s*	6.274 *s*	2.830 *s*	53.508 *s*

6. Conclusions

Realistic 3D mesh objectification plays an essential role in generating semantic 3D models to support entity-level query and analysis in various 3D GIS applications such as digital twin cities and city information modeling (CIM). Efforts have been made to extract footprints and geometric information of buildings from images or point clouds, which can only capture limited building texture or geometric information, resulting in the obtained buildings being often fragmented.

To address these challenges, this study presents a novel bidirectionally greedy framework for extracting geometrically complete 3D building models from geomeshes in an unsupervised manner. Differently from the objects extracted from point clouds that need to be reconstructed to generate 3D models, the framework can directly separate the realistic 3D building models from geomeshes. The proposed framework encompasses primitive over-segmentation, parameter-free plane detection, greedy culling of non-building planes, and greedy recovery of building primitives. This framework requires only a few parameters to set up by the operator that is easy to understand. The framework performance was quantitatively and visually assessed on five geomeshes with labels. Almost all the buildings were extracted correctly, with *recall* up to about 99% on Gm4B and overall classification accuracy higher than 86%. In addition, unlike existing methods that extract buildings that are often fragmented, the proposed framework extracts 3D building models with complete geometry, avoiding compromising their visualization and analysis capabilities.

Furthermore, the extracted models can be used to boost the instantiation of buildings, high LOD level CityGML construction, and the objectification of vegetation, roads, city furniture, etc., with a view to improving the query, analysis, and rendering capabilities of 3D, GIS and CIM applications. It can also assist in the production of labels for deep neural network training.

Although the proposed framework is proven to be effective in the tasks of building extraction, there are still some problems to be solved: Firstly, since the color index is used to cull vegetation, a small fraction of building planes that are very close to the vegetation color will be culled by mistake. Similarly, the color index is difficult to adapt to a non-green lush tree canopy. Secondly, non-building primitives that are topologically adjacent to broken buildings obtained by greedy culling are prone to be mistakenly extracted. Future work will therefore focus on reducing the sensitivity of the method to color and instantiating the building further.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

The authors wish to express their thanks to the anonymous reviewers for their valuable comments in the earlier version of this paper. The work was supported by the Hubei Provincial Technology Innovation Project (major project, No. 2020AAA004) and Hubei Provincial Natural Science Foundation of China (No. 2020CFA001). The authors would like to thank SOUTH DIGITAL TECHNOLOGY CO., LTD for providing oblique photogrammetry data. The authors would like to thank the ISPRS for providing benchmark datasets.

Data and Codes Availability Statement

The data used in this study are available as follows:
ISPRS H3D: http://ifpwww.ifp.uni-stuttgart.de/benchmark/hessigheim/Default.aspx
Gm4B: https://github.com/Dyuzz/BuildingEx4Mesh
Source codes: https://github.com/Dyuzz/BuildingEx4Mesh/dataset

References

1 Jaillot, V., Servigne, S. and Gesquière, G. Delivering time-evolving 3D city models for web visualization. *International Journal of Geographical Information Science*, 2020, 34 (10), pp. 2030–2052. DOI: 10.1080/13658816.2020.1749637

2 Shan, P. and Sun, W. Research on 3D urban landscape design and evaluation based on geographic information system. *Environmental Earth Sciences 2021 80:17*, 2021, 80 (17), pp. 1–15. DOI: 10.1007/S12665-021-09886-Y

3 Machete, R., Falcão, A.P., Gomes, M.G. and Moret Rodrigues, A. The use of 3D GIS to analyse the influence of urban context on buildings' solar energy potential. *Energy and Buildings*, 2018, 177, pp. 290–302. DOI: 10.1016/J.ENBUILD.2018.07.064

4 Yu, D., Tang, L., Ye, F. and Chen, C. A virtual geographic environment for dynamic simulation and analysis of tailings dam failure. *International Journal of Digital Earth*, 2021, 14 (9), pp. 1194–1212. DOI: 10.1080/17538947.2021.1945151

5 Liang, J., Shen, S., Gong, J., Liu, J. and Zhang, J. Embedding user-generated content into oblique airborne photogrammetry-based 3D city model. *International Journal of Geographical Information Science*, 2016, 31 (1), pp. 1–16. DOI: 10.1080/13658816.2016.1180389

6 Wang, L., Tian, W.X., Zhao, X.Y. and Huang, C.Q. Numerical simulation of the effects of canopy properties on airflow and pollutant dispersion in street canyons: *Indoor and Built Environment*, 2021, 31 (2), pp. 466–478. DOI: 10.1177/1420326X211005174

7 Tang, L., Yin, D., Chen, C., Yu, D. and Han, W. Optimal design of plant canopy based on light interception: A case study with loquat. *Frontiers in Plant Science*, 2019, 10, p. 364. DOI: 10.3389/FPLS.2019.00364/BIBTEX

8 Yu, D., Tang, L. and Chen, C. Three-dimensional numerical simulation of mud flow from a tailing dam failure across complex terrain. *Natural Hazards and Earth System Sciences*, 2020, 20 (3), pp. 727–741. DOI: 10.5194/NHESS-20-727-2020

9 Du, S., Zhang, Y., Zou, Z., Xu, S., He, X. and Chen, S. Automatic building extraction from LiDAR data fusion of point and grid-based features. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2017, 130, pp. 294–307. DOI: 10.1016/j.isprsjprs.2017.06.005

10 Ghanea, M., Moallem, P. and Momeni, M. Building extraction from high-resolution satellite images in urban areas: recent methods and strategies against significant challenges. *International Journal of Remote Sensing*, 2016, 37 (21), pp. 5234–5248. DOI: 10.1080/01431161.2016.1230287

11 Na, Y., Kim, J.H., Lee, K., Park, J., Hwang, J.Y. and Choi, J.P. Domain adaptive transfer attack-based segmentation networks for building extraction from aerial images. *IEEE Transactions on Geoscience and Remote Sensing*, 2021, 59 (6), pp. 5171–5182. DOI: 10.1109/TGRS.2020.3010055

12 Fan, X., Nie, G., Gao, N., Deng, Y., An, J. and Li, H. Building extraction from UAV remote sensing data based on photogrammetry method. *International Geoscience and Remote Sensing Symposium (IGARSS)*, 2017, 2017-July, pp. 3317–3320. DOI: 10.1109/IGARSS.2017.8127707

13 Gilani, S.A.N., Awrangjeb, M. and Lu, G. Segmentation of airborne point cloud data for automatic building roof Extraction. *GIScience & Remote Sensing*, 2018, 55 (1), pp. 63–89. DOI: 10.1080/15481603.2017.1361509

14 Jochem, A., Höfle, B., Wichmann, V., Rutzinger, M. and Zipf, A. Area-wide roof plane segmentation in airborne LiDAR point clouds. *Computers, Environment and Urban Systems*, 2012, 36 (1), pp. 54–64. DOI: 10.1016/J.COMPENVURBSYS.2011.05.001

15 Yang, B., Xu, W. and Yao, W. Extracting buildings from airborne laser scanning point clouds using a marked point process. *GIScience and Remote Sensing*, 2014, 51 (5), pp. 555–574. DOI: 10.1080/15481603.2014.950117

16 Zhu, Q., Li, Y., Hu, H. and Wu, B. Robust point cloud classification based on multi-level semantic relationships for urban scenes. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2017, 129, pp. 86–102. DOI: 10.1016/J.ISPRSJPRS.2017.04.022

17 Zhao, H., Jiang, L., Jia, J., Torr, P. and Koltun, V. Point transformer. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 16239–16248. DOI: 10.1109/ICCV48922.2021.01595

18 Laupheimer, D., Shams Eddin, M.H. and Haala, N. On the association of LiDAR point clouds and textured meshes for multi-modal semantic segmentation. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2020, 5 (2), pp. 509–516. DOI: 10.5194/ISPRS-ANNALS-V-2-2020-509-2020

19 Huang, R., Yang, B., Liang, F., Dai, W., Li, J., Tian, M. and Xu, W. A top-down strategy for buildings extraction from complex urban scenes using airborne LiDAR point clouds. *Infrared Physics & Technology*,

2018, 92, pp. 203–218. DOI: 10.1016/J.INFRARED.2018.05.021

20 Li, G., Muller, M., Thabet, A. and Ghanem, B. DeepGCNs: Can GCNs go as deep as CNNs? *Proceedings of the IEEE International Conference on Computer Vision*, 2019, 2019-October, pp. 9266–9275. DOI: 10.48550/arxiv.1904.03751

21 Hu, Q., Yang, B., Xie, L., Rosa, S., Guo, Y., Wang, Z., Trigoni, N. and Markham, A. RandLA-Net: Efficient semantic segmentation of large-scale point clouds. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11105–11114. DOI: 10.48550/arxiv.1911.11236

22 Liasis, G. and Stavrou, S. Building extraction in satellite images using active contours and colour features. *International Journal of Remote Sensing*, 2016, 37 (5), pp. 1127–1153. DOI: 10.1080/01431161.2016.1148283

23 Chaudhuri, D., Kushwaha, N.K., Samal, A. and Agarwal, R.C. Automatic building detection from high-resolution satellite images based on morphology and internal gray variance. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2016, 9 (5), pp. 1767–1779. DOI: 10.1109/JSTARS.2015.2425655

24 Ji, S., Wei, S. and Lu, M. Fully convolutional networks for multisource building extraction from an open aerial and satellite imagery data Set. *IEEE Transactions on Geoscience and Remote Sensing*, 2019, 57 (1), pp. 574–586. DOI: 10.1109/TGRS.2018.2858817

25 Huang, X. and Zhang, L. Morphological building/shadow index for building extraction from high-resolution imagery over urban areas. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2012, 5 (1), pp. 161–172. DOI: 10.1109/JSTARS.2011.2168195

26 Sharma, D. and Singhai, J. An unsupervised framework to extract the diverse building from the satellite images using Grab-cut method. *Earth Science Informatics 2021 14:2*, 2021, 14 (2), pp. 777–795. DOI: 10.1007/S12145-021-00569-7

27 Cote, M. and Saeedi, P. Automatic rooftop extraction in nadir aerial imagery of suburban regions using corners and variational level set evolution. *IEEE Transactions on Geoscience and Remote Sensing*, 2013, 51 (1), pp. 313–328. DOI: 10.1109/TGRS.2012.2200689

28 Manno-Kovacs, A. and Sziranyi, T. Orientation-selective building detection in aerial images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2015, 108, pp. 94–112. DOI: 10.1016/J.ISPRSJPRS.2015.06.007

29 Xiao, J., Gerke, M. and Vosselman, G. Building extraction from oblique airborne imagery based on robust façade detection. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2012, 68 (1), pp. 56–68. DOI: 10.1016/J.ISPRSJPRS.2011.12.006

30 Zolanvari, S.M.I. and Laefer, D.F. Slicing method for curved façade and window extraction from point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2016, 119, pp. 334–346. DOI: 10.1016/J.ISPRSJPRS.2016.06.011

31 Chen, D., Zhang, L., Li, J. and Liu, R. Urban building roof segmentation from airborne lidar point clouds. *International Journal of Remote Sensing*, 2012, 33 (20), pp. 6497–6515. DOI: 10.1080/01431161.2012.690083

32 Zhou, Q.Y. and Neumann, U. Fast and extensible building modeling from airborne LiDAR data. *GIS: Proceedings of the ACM International Symposium on Advances in Geographic Information Systems*, 2008, pp. 43–50. DOI: 10.1145/1463434.1463444

33 Zhang, K., Yan, J. and Chen, S.C. Automatic construction of building footprints from airborne LIDAR data. *IEEE Transactions on Geoscience and Remote Sensing*, 2006, 44 (9), pp. 2523–2533. DOI:

10.1109/TGRS.2006.874137

34 Schnabel, R., Wahl, R. and Klein, R. Efficient RANSAC for point-cloud shape detection. *Computer Graphics Forum*, 2007, 26 (2), pp. 214–226. DOI: 10.1111/J.1467-8659.2007.01016.X

35 Chen, D., Zhang, L., Mathiopoulos, P.T. and Huang, X. A methodology for automated segmentation and reconstruction of urban 3-D buildings from ALS point clouds. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2014, 7 (10), pp. 4199–4217. DOI: 10.1109/JSTARS.2014.2349003

36 Boulch, A. and Marlet, R. Fast and robust normal estimation for point clouds with sharp features. *Computer Graphics Forum*, 2012, 31 (5), pp. 1765–1774. DOI: 10.1111/J.1467-8659.2012.03181.X

37 Huang, H. and Brenner, C. Rule-based roof plane detection and segmentation from laser point clouds. *2011 Joint Urban Remote Sensing Event, JURSE 2011 - Proceedings*, 2011, pp. 293–296. DOI: 10.1109/JURSE.2011.5764777

38 Kim, K.H. and Shan, J. Building roof modeling from airborne laser scanning data based on level set approach. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2011, 66 (4), pp. 484–497. DOI: 10.1016/J.ISPRSJPRS.2011.02.007

39 Biosca, J.M. and Lerma, J.L. Unsupervised robust planar segmentation of terrestrial laser scanner point clouds based on fuzzy clustering methods. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2008, 63 (1), pp. 84–98. DOI: 10.1016/J.ISPRSJPRS.2007.07.010

40 Wang, Z., Zhang, L., Fang, T., Mathiopoulos, P.T., Tong, X., Qu, H., Xiao, Z., Li, F. and Chen, D. A multiscale and hierarchical feature extraction method for terrestrial laser scanning point cloud classification. *IEEE Transactions on Geoscience and Remote Sensing*, 2015, 53 (5), pp. 2409–2425. DOI: 10.1109/TGRS.2014.2359951

41 Mongus, D., Lukač, N. and Žalik, B. Ground and building extraction from LiDAR data based on differential morphological profiles and locally fitted surfaces. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2014, 93, pp. 145–156. DOI: 10.1016/J.ISPRSJPRS.2013.12.002

42 Chen, Q., Gong, P., Baldocchi, D. and Xie, G. Filtering airborne laser scanning data with morphological methods. *Photogrammetric Engineering and Remote Sensing*, 2007, 73 (2), pp. 175–185. DOI: 10.14358/PERS.73.2.175

43 Cheng, L., Zhao, W., Han, P., Zhang, W., Shan, J., Liu, Y. and Li, M. Building region derivation from LiDAR data using a reversed iterative mathematic morphological algorithm. *Optics Communications*, 2013, 286 (1), pp. 244–250. DOI: 10.1016/J.OPTCOM.2012.08.028

44 Pingel, T.J., Clarke, K.C. and McBride, W.A. An improved simple morphological filter for the terrain classification of airborne LIDAR data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2013, 77, pp. 21–30. DOI: 10.1016/J.ISPRSJPRS.2012.12.002

45 Rodríguez-Caballero, E., Afana, A., Chamizo, S., Solé-Benet, A. and Canton, Y. A new adaptive method to filter terrestrial laser scanner point clouds using morphological filters and spectral information to conserve surface micro-topography. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2016, 117, pp. 141–148. DOI: 10.1016/J.ISPRSJPRS.2016.04.004

46 Zhang, J., Lin, X. and Ning, X. SVM-based classification of segmented airborne LiDAR point clouds in urban areas. *Remote Sensing 2013, Vol. 5, Pages 3749-3775*, 2013, 5 (8), pp. 3749–3775. DOI: 10.3390/RS5083749

47 Niemeyer, J., Rottensteiner, F. and Soergel, U. Contextual classification of lidar data and building object detection in urban areas. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2014, 87, pp. 152–165. DOI: 10.1016/J.ISPRSJPRS.2013.11.001

48 Chen, M., Feng, A., McAlinden, R. and Soibelman, L. Photogrammetric point cloud segmentation and object information extraction for creating virtual environments and simulations. *Journal of Management in Engineering*, 2019, 36 (2), p. 04019046. DOI: 10.1061/(ASCE)ME.1943-5479.0000737

49 Akbulut, Z., Özdemir, S., Acar, H. and Karsli, F. Automatic building extraction from image and LiDAR data with active contour segmentation. *Journal of the Indian Society of Remote Sensing*, 2018, 46 (12), pp. 2057–2068. DOI: 10.1007/s12524-018-0871-2

50 Zarea, A. and Mohammadzadeh, A. A novel building and tree detection method from LiDAR data and aerial images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2016, 9 (5), pp. 1864–1875. DOI: 10.1109/JSTARS.2015.2470547

51 Awrangjeb, M., ChunSun, Z. and Fraser, C.S. Building detection in complex scenes thorough effective separation of buildings from trees. *Photogrammetric Engineering & Remote Sensing*, 2012, 78 (7), pp. 729–745.

52 Sánchez-Lopera, J. and Lerma, J.L. Classification of lidar bare-earth points, buildings, vegetation, and small objects based on region growing and angular classifier. *International Journal of Remote Sensing*, 2014, 35 (19), pp. 6955–6972. DOI: 10.1080/01431161.2014.960619

53 Richter, R., Behrens, M. and Döllner, J. Object class segmentation of massive 3D point clouds of urban areas using point cloud topology. *International Journal of Remote Sensing*, 2013, 34 (23), pp. 8408–8424. DOI: 10.1080/01431161.2013.838710

54 Miliaresis, G. and Kokkas, N. Segmentation and object-based classification for the extraction of the building class from LIDAR DEMs. *Computers & Geosciences*, 2007, 33 (8), pp. 1076–1087. DOI: 10.1016/J.CAGEO.2006.11.012

55 Pu, S. and Vosselman, G. Knowledge based reconstruction of building models from terrestrial laser scanning data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2009, 64 (6), pp. 575–584. DOI: 10.1016/J.ISPRSJPRS.2009.04.001

56 Zolanvari, S.M.I., Laefer, D.F. and Natanzi, A.S. Three-dimensional building façade segmentation and opening area detection from point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2018, 143, pp. 134–149. DOI: 10.1016/J.ISPRSJPRS.2018.04.004

57 Boulaassal, H., Landes, T. and Grussenmeyer, P. Automatic extraction of planar clusters and their contours on building façades recorded by terrestrial laser scanner. *International Journal of Architectural Computing*, 2009, 7 (1), pp. 1–20. DOI: 10.1260/147807709788549411

58 Oehler, B., Stueckler, J., Welle, J., Schulz, D. and Behnke, S. Efficient multi-resolution plane segmentation of 3D point clouds. *Intelligent Robotics and Applications*, 2011, 7102 LNAI (PART 2), pp. 145–156. DOI: 10.1007/978-3-642-25489-5_15

59 Xu, Y., Yao, W., Tuttas, S., Hoegner, L. and Stilla, U. Unsupervised segmentation of point clouds from buildings using hierarchical clustering based on gestalt principles. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2018, 11 (11), pp. 4270–4286. DOI: 10.1109/JSTARS.2018.2817227

60 Besl, P.J. and Jain, R.C. Segmentation through variable-order surface fitting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1988, 10 (2), pp. 167–192. DOI: 10.1109/34.3881

61 Vo, A.V., Truong-Hong, L., Laefer, D.F. and Bertolotto, M. Octree-based region growing for point cloud segmentation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2015, 104, pp. 88–100. DOI: 10.1016/J.ISPRSJPRS.2015.01.011

62 Xiao, J., Zhang, J., Adler, B., Zhang, H. and Zhang, J. Three-dimensional point cloud plane segmentation in both structured and unstructured environments. *Robotics and Autonomous Systems*, 2013, 61 (12), pp.

1641–1652. DOI: 10.1016/J.ROBOT.2013.07.001

63 Nurunnabi, A., Belton, D. and West, G. Robust segmentation in laser scanning 3D point cloud data. *2012 International Conference on Digital Image Computing Techniques and Applications, DICTA 2012*, 2012. DOI: 10.1109/DICTA.2012.6411672

64 Jagannathan, A. and Miller, E.L. Three-dimensional surface mesh segmentation using curvedness-based region growing approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007, 29 (12), pp. 2195–2204. DOI: 10.1109/TPAMI.2007.1125

65 Wang, J. and Yu, Z. Geometric decomposition of 3D surface meshes using Morse theory and region growing. *International Journal of Advanced Manufacturing Technology*, 2011, 56 (9–12), pp. 1091–1103. DOI: 10.1007/S00170-011-3259-9

66 Liu, Y. and Xiong, Y. Automatic segmentation of unorganized noisy point clouds based on the Gaussian map. *Computer-Aided Design*, 2008, 40 (5), pp. 576–594. DOI: 10.1016/J.CAD.2008.02.004

67 Dong, Z., Yang, B., Hu, P. and Scherer, S. An efficient global energy optimization approach for robust 3D plane segmentation of point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2018, 137, pp. 112–133. DOI: 10.1016/J.ISPRSJPRS.2018.01.013

68 Rusu, R.B., Blodow, N. and Beetz, M. Fast point feature histograms (FPFH) for 3D registration. *Proceedings - IEEE International Conference on Robotics and Automation*, 2009, pp. 3212–3217. DOI: 10.1109/ROBOT.2009.5152473

69 Stein, S.C., Schoeler, M., Papon, J. and Worgotter, F. Object partitioning using local convexity. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2014, pp. 304–311. DOI: 10.1109/CVPR.2014.46

70 Schoeler, M., Papon, J. and Wörgötter, F. Constrained planar cuts - Object partitioning for point clouds. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, 07-12-June-2015, pp. 5207–5215. DOI: 10.1109/CVPR.2015.7299157

71 Papon, J., Abramov, A., Schoeler, M. and Worgotter, F. Voxel cloud connectivity segmentation - Supervoxels for point clouds. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2027–2034. DOI: 10.1109/CVPR.2013.264

72 Song, S., Lee, H. and Jo, S. Boundary-enhanced supervoxel segmentation for sparse outdoor LiDAR data. *Electronics Letters*, 2014, 50 (25), pp. 1917–1919. DOI: 10.1049/EL.2014.3249

73 Lin, Y., Wang, C., Zhai, D., Li, W. and Li, J. Toward better boundary preserved supervoxel segmentation for 3D point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2018, 143, pp. 39–47. DOI: 10.1016/J.ISPRSJPRS.2018.05.004

74 Zai, D., Li, J., Guo, Y., Cheng, M., Lin, Y., Luo, H. and Wang, C. 3-D road boundary extraction from mobile laser scanning data via supervoxels and graph cuts. *IEEE Transactions on Intelligent Transportation Systems*, 2018, 19 (3), pp. 802–813. DOI: 10.1109/TITS.2017.2701403

75 Ramiya, A.M., Nidamanuri, R.R. and Ramakrishnan, K. A supervoxel-based spectro-spatial approach for 3D urban point cloud labelling. *International Journal of Remote Sensing*, 2016, 37 (17), pp. 4172–4200. DOI: 10.1080/01431161.2016.1211348

76 HuangShi-Sheng, MaZe-Yu, MuTai-Jiang, FuHongbo and HuShi-Min. Supervoxel convolution for online 3D semantic segmentation. *ACM Transactions on Graphics (TOG)*, 2021, 40 (3), pp. 1–15. DOI: 10.1145/3453485

77 Yu, D., He, L., Ye, F., Jiang, L., Zhang, C., Fang, Z. and Liang, Z. Unsupervised ground filtering of airborne-based 3D meshes using a robust cloth simulation. *International Journal of Applied Earth Observation and Geoinformation*, 2022, 111, p. 102830. DOI: 10.1016/J.JAG.2022.102830

78 Rao, R. V. and More, K.C. Design optimization and analysis of selected thermal devices using self-adaptive Jaya algorithm. *Energy Conversion and Management*, 2017, 140, pp. 24–35. DOI: 10.1016/J.ENCONMAN.2017.02.068

79 Kölle, M., Laupheimer, D., Schmohl, S., Haala, N., Rottensteiner, F., Wegner, J.D. and Ledoux, H. The hessigheim 3D (H3D) benchmark on semantic segmentation of high-resolution 3D point clouds and textured meshes from UAV LiDAR and Multi-View-Stereo. *ISPRS Open Journal of Photogrammetry and Remote Sensing*, 2021, 1, p. 100001. DOI: 10.1016/J.OPHOTO.2021.100001