SUM: A Benchmark Dataset of **S**emantic **U**rban **M**eshes

Weixiao GAO[a,*], Liangliang Nan[a], Bas Boom[b], Hugo Ledoux[a]

[a] *3D Geoinformation Research Group, Faculty of Architecture and the Built Environment, Delft University of Technology, 2628 BL Delft, The Netherlands*
[b] *CycloMedia Technology, Zaltbommel, The Netherlands*

Abstract

Recent developments in data acquisition technology allow us to collect 3D texture meshes quickly. Those can help us understand and analyse the urban environment, and as a consequence are useful for several applications like spatial analysis and urban planning. Semantic segmentation of texture meshes through deep learning methods can enhance this understanding, but it requires a lot of labelled data. The contributions of this work are three-fold: (1) a new benchmark dataset of semantic urban meshes, (2) a novel semi-automatic annotation framework, and (3) an annotation tool for 3D meshes. In particular, our dataset covers about 4 km^2 in Helsinki (Finland), with six classes, and we estimate that we save about 600 hours of labelling work using our annotation framework, which includes initial segmentation and interactive refinement. We also compare the performance of several state-of-the-art 3D semantic segmentation methods on the new benchmark dataset. Other researchers can use our results to train their networks: the dataset is publicly available, and the annotation tool is released as open-source.

Keywords: Texture meshes; Urban scene understanding; Mesh annotation; Semantic segmentation; Over-segmentation; Benchmark dataset

1. Introduction

Understanding the urban environment from 3D data (e.g. point clouds and 3D meshes) is a long-standing goal in photogrammetry and computer vision [1, 2]. The fast recent developments in data acquisition technologies and processing pipelines have allowed us to collect a great number of datasets on our 3D urban environments. Prominent examples are Google Earth [3], texture meshes covering entire cities (e.g. Helsinki [4]), or point clouds covering entire countries (e.g., the Netherlands AHN [5]). These datasets have attracted

[*] Corresponding author
Email addresses: w.gao-1@tudelft.nl (Weixiao GAO), liangliang.nan@tudelft.nl (Liangliang Nan), bboom@cyclomedia.com (Bas Boom), h.ledoux@tudelft.nl (Hugo Ledoux)

interest because of their potential in several applications, for instance, urban planning [6, 7], positioning and navigation [8, 9, 10], spatial analysis [11], environmental analysis [12], and urban fluid simulation [13].

To effectively understand the urban phenomena behind the data, a large amount of ground truth is typically required, especially when applying supervised learning-based techniques, such as a deep Convolutional Neural Network (CNN). The recent development of machine learning (especially deep learning) techniques has demonstrated promising performance in semantic segmentation of 3D point clouds [14, 15, 16]. Compared to point clouds, a surface representation (in the form of a 3D mesh, often with textures, see Figure 1 and 2 for an example) of the urban scene has multiple advantages: easy to acquire, compact storage, accurate, and with well-defined topological structures.

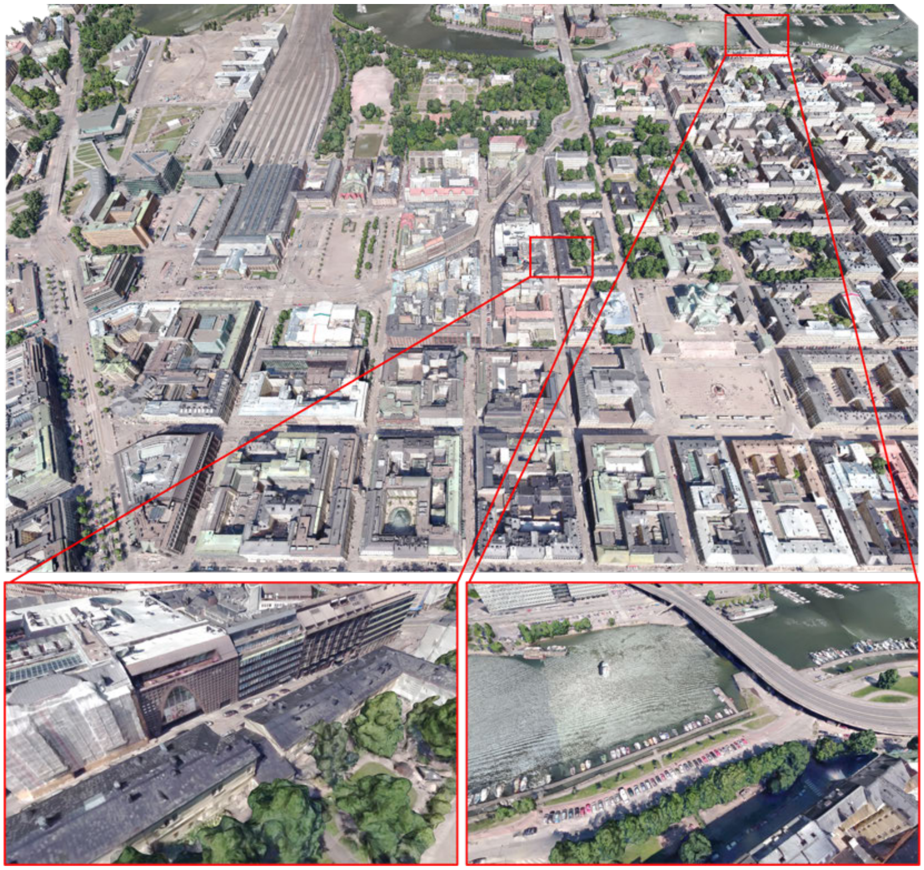


Figure 1: Part of the semantic urban mesh benchmark dataset shown as a texture mesh.

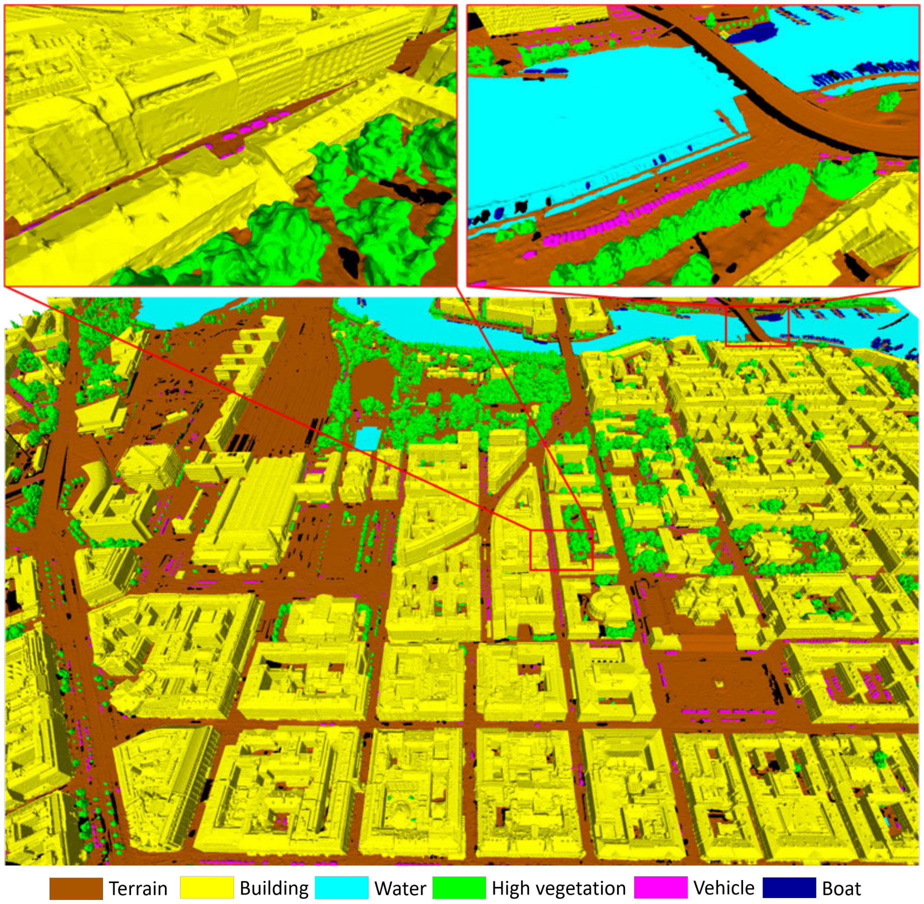


Figure 2: Part of the semantic urban mesh benchmark dataset, showing the semantic classes (unclassified regions are in black).

This means that 3D meshes have the potential to serve as input for scene understanding. As a consequence, there is an urgent demand for large-scale urban mesh datasets that can be used as ground truth for both training and evaluating the 3D semantic segmentation workflows.

In this paper, we aim to establish a benchmark dataset of large-scale urban meshes reconstructed from aerial oblique images. To achieve this goal, we propose a semi-automatic mesh annotation framework that includes two components: (1) an automatic process to generate intermediate labels from the raw 3D mesh; (2) manual semantic refinement of those labels. For the intermediate label generation step, we have developed a semantic mesh segmentation method that classifies each triangle into a pre-defined object class. This semantic initialization allows us to achieve an overall accuracy of 93.0% in the classification of the triangle faces in our dataset, saving significant efforts for manually labelling. Then, in the semantic refinement step, a mesh annotation tool (which we have developed) is used to refine the semantic labels of the pre-labelled data (at the triangle and segment levels).

We have used our proposed framework to generate a semantic-rich urban mesh dataset consisting of 19 million triangles and covering about 4 km^2 with six object classes commonly found in an urban environment: terrain, high-vegetation, building, water, vehicle, and boat (Figure 2 shows an example from our dataset). With our semi-automatic annotation framework, generating the ground truth took only about 400 hours; we estimate that manually labelling the triangles would have taken more than 1000 hours. The contributions of our work are:

- a semantic-rich urban mesh dataset of six classes of common urban objects with texture information;
- a semi-automatic mesh annotation framework consisting of two parts: a pipeline for semantic mesh segmentation and an annotation tool for semantic refinement;
- a comprehensive evaluation and comparison of the state-of-the-art semantic segmentation methods on the new dataset.

The benchmark dataset is freely available, and the semantic mesh segmentation methods and the annotation software for 3D meshes are released as open-source[1].

2. Related Work

Urban datasets can be captured with different sensors and be reconstructed with different methods, and the resulting datasets will have different properties. Most benchmark urban datasets focus on point clouds, whereas our semantic urban benchmark dataset is based on textured triangular meshes.

The input of the semantic labelling process can be raw or pre-labelled urban datasets such as the automatically generated results from over-segmentation or semantic segmentation (see Section 3.3). Regardless of the input data, it still needs to be manually checked and annotated with a labelling tool, which involves selecting a correct semantic label from a predefined list for each triangle (or point, depending on the dataset) by users. In addition, some interactive approaches can make the labelling process semi-manual. However, unlike our proposed approach, the labelling work of most of the 3D benchmark data does not take full advantage of over-segmentation and semantic segmentation on 3D data, and interactive annotation in the 3D space.

We present in this section an overview of the publicly available semantic 3D urban benchmark datasets categorised by sensors and reconstruction types (see Table 1). More specifically, we elaborate on the quality, scale, and labelling strategy of the existing urban datasets regarding semantic segmentation.

[1] `https://3d.bk.tudelft.nl/projects/meshannotation/`

Name	Platforms	Year	Data Type	Area[a]/ Length	Classes	Points / Triangles	RGB	Automatic Pre-labelling	Annotation	Time Cost (hours)
Oakland 3D [17]	MLS	2009	Point Cloud	1.5 km	5	1.6 M	No	No	3D Manually	Not reported
Paris-rue-Madame [18]	MLS	2014	Point Cloud	0.16 km	17	20 M	No	2D semantic segmentation	3D Semi-manually	Not reported
iQmulus [19]	MLS	2015	Point Cloud	10 km	8	300 M	No	No	2D Semi-manually	Not reported
Semantic3D [2]	TLS	2017	Point Cloud	-	8	4000 M	Yes	No	2D & 3D Semi-manually	Not reported
Paris-Lille-3D [20]	MLS	2018	Point Cloud	1.94 km	9	143 M	No	No	3D Manually	Not reported
SemanticKITTI [21]	MLS	2019	Point Cloud	39.2 km	25	4549 M	No	No	3D Manually	1700
Toronto-3D [22]	MLS	2020	Point Cloud	1.0 km	8	78.3 M	Yes	No	3D Manually	Not reported
ISPRS [23]	ALS	2012	Point Cloud	0.1 km^2	9	1.2 M	No	No	3D Manually	Not reported
AHN3 [5]	ALS	2019	Point Cloud	41,543 km^2	4	415.43 B[b]	No	3D semantic segmentation	3D Manually	Not reported
DublinCity [24]	ALS	2019	Point Cloud	2.0 km^2	13	260 M	No	No	3D Manually	2500
DALES [25]	ALS	2020	Point Cloud	10.0 km^2	8	505.3 M	No	3D semantic segmentation	3D Manually	Not reported
LASDU [26]	ALS	2020	Point Cloud	1.02 km^2	5	3.12 M	No	No	3D Manually	Not reported
ETHZ RueMonge [27, 28]	Auto-mobile camera	2014	Mesh	0.7 km	9	1.8 M (lowres)[c]	Yes (per vertex)[d]	2D over-segmentation	2D Semi-manually	230 (701 frames)[e]
Campus3D [29]	UAV camera	2020	Point Cloud	1.58 km^2	14	937.1 M	Yes	No	2D & 3D Manually	Not reported
SensatUrban [30]	UAV camera	2020	Point Cloud	6 km^2	13	2847.1 M	Yes	No	3D Manually	600
Swiss3DCities [31]	UAV camera	2020	Point Cloud	2.7 km^2	5	226 M	Yes	No	3D Manually (on mesh)	144 (1 M Triangles)[f]
Hessigheim 3D [32, 33]	UAV Lidar & camera	2021	Point Cloud & Mesh	0.19 km^2	11	125.7 M / 36.76 M[g]	Yes (texture)[h]	No	3D Manually[i]	Not reported
SUM-Helsinki (Ours)	Airplane camera	2021	Mesh	4 km^2	6	19 M	Yes (texture)[h]	3D over-segmentation & 3D semantic segmentation	3D Semi-manually	400

[a] The area was measured in a 2D map.
[b] The number of total points (i.e., 415.43 billion) is estimated.
[c] The low-resolution meshes contain 1.8 million triangle faces, according to the publications.
[d] An RGB colour was assigned to each triangle vertex.
[e] The frames were from video sequences.
[f] About one million triangles (16 tiles) from simplified mesh were labelled, which took around 6 to 12 hours per tile.
[g] The number of LiDAR points is 125.7 million and the number of triangle faces is 36.76 million.
[h] The colour of each triangle face corresponds to a patch of the texture image.
[i] The LiDAR point clouds were manually annotated and the labels were transferred to the mesh.

Table 1: Comparison of existing 3D urban benchmark datasets.

2.1. Photogrammetric Products

2.1.1. Dense Point Clouds

The *Campus3D* [29] is to our knowledge the first aerial point cloud benchmark. The coarse labelling is conducted in 2D projected images with three views, and the grained labels are refined in 3D with user-defined rotation angles. The dataset covers only the campus of the National University of Singapore and is thus not representative of a typical urban scene.

SensatUrban [30] is another example of the photogrammetric point clouds covering various urban landscapes in two cities of the UK. The semantic points are manually annotated via the off-the-shelf software tool CloudCompare [34], and the overall annotation is reported to have taken around 600 hours. The dataset also contains several areas without points, especially for water surfaces and regions with dense objects. The leading causes are the Lambertion surface assumption during the image matching and the inadequate image overlapping rate during the flight.

Similarly, the *Swiss3DCities* [31] was recently released that covers three cities in Zurich but twice smaller than the SensatUrban. The annotation work was conducted on a simplified mesh in the software Blender [35], and then the semantics were transferred to the mesh vertices, which are regarded as point clouds, via the nearest neighbour search. The mesh simplification may result in the loss of small-scale objects such as building dormers and chimneys, and the automatic transfer of the labels could have introduced errors in the ground truth.

2.1.2. Triangle Meshes

To the best of our knowledge, the *ETHZ RueMonge 2014* [28] is the first urban-related benchmark dataset available as surface meshes. The label for each triangle is obtained from projecting selected images that are manually labelled from over-segmented image sequences [27]. In fact, due to the error of multi-view optimisation and the ambiguous object boundary within triangle faces, the datasets contain many misclassified labels, making them unsuitable for training and evaluating supervised-learning algorithms.

Hessigheim 3D [32, 33] is a small-scale semantic urban dataset consisting of highly dense LiDAR point clouds and high resolution texture meshes. Particularly, the mesh is generated from both LiDAR point cloud and oblique aerial images in a hybrid way. The labels of point clouds are manually annotated in CloudCompare [34], and the labels of the mesh are transferred from the point clouds by computing the majority votes per triangle. However, if the mesh triangle has no corresponding points, some faces may remain unlabelled which resulted in about 40% unlabelled area. In addition, this dataset contains non-manifold vertices, which makes it difficult to use directly.

2.2. LiDAR Point Clouds

Unlike photogrammetric point clouds, LiDAR point clouds usually do not contain colour information. To annotate them properly, additional information is often required, e.g. images or 2D maps. LiDAR point cloud benchmark datasets are more common than photogrammetric ones.

2.2.1. Street-view Datasets

The *Oakland 3D* [17] is one of the earliest mobile laser scanning (MLS) point cloud datasets, which was designed for the classification of outdoor scenes. It has five hand-labelled classes with 44 sub-classes, but without colour information and semantic categories like roof, canopy, or interior building block, which are typical for all street-view captured datasets.

Compared to *Oakland 3D*, *Paris-rue-Madame* [18] is a relatively smaller dataset which used the 2D semantic segmentation results for 3D annotation. Specifically, the point clouds were projected onto images to extract the objects hierarchically with several unsupervised segmentation and classification algorithms.

Although the 2D pre-labelled generation is fully automatic, different semantic categories require different segmentation algorithms resulting in difficulties in the classification of multiple classes.

The *iQmulus dataset* [19] is a 10 *km* street dataset annotated based on projected images in the 2D space. Specifically, the user first needs to extract objects by editing the image with a polyline tool and then assigns labels to the extracted object regions. Some automatic functions are made for polyline editing in this framework, but the entire annotation pipeline is still complicated.

Unlike other street view datasets, *Semantic3D* [2] is a dataset consisting of terrestrial laser scanning (TLS) point clouds (the scanner is not moving and scans are made from only a few viewpoints). It has eight classes and colours were obtained by projecting the points onto the original images. There are two annotation methods: (1) annotating in 3D with an iterative model-fitting approach on manually selected points; (2) annotating in a 2D view by separate background from a drawn polygon in CloudCompare [34]. Although it covers many urban scenes and includes RGB information, the acquired objects are incomplete because of the limited viewpoints and occlusions.

The other three typical MLS point cloud datasets that were manually labelled are *Paris-Lille-3D* [20], *SemanticKITTI* [21], and *Toronto-3D* [22].

2.2.2. Aerial-view Datasets

As for ALS benchmark point clouds, representative datasets are *ISPRS* [23], *DublinCity* [24], and *LASDU* [26] covering various scales of city landscapes and were annotated manually with off-the-shelf software. Instead of fully manual annotation, the *Dayton Annotated LiDAR Earth Scan (DALES)* [25] used digital elevation models (DEM) to distinguish ground points with a certain threshold, the estimated normal to label the building points roughly, and satellite images to provide contextual information as references for annotators to check and label the rest of data. Similarly, the AHN3 dataset [5] was semi-manually labelled by different companies with off-the-shelf software. Besides, since the ALS measurement is conducted in the top view direction, unlike oblique aerial cameras, the obtained point clouds often miss facade information to a certain degree.

3. The Semantic Urban Mesh Dataset

3.1. Dataset Specification

We have used Helsinki's 3D texture meshes as input and annotated them as a benchmark dataset of semantic urban meshes. The Helsinki's raw dataset covers about 12 km^2, and it was generated in 2017 from oblique aerial images that have about a 7.5 cm ground sampling distance (GSD) using an off-the-shelf commercial software namely ContextCapture [36]. The source images have three colour channels (i.e., red, green, and blue) and are collected from an airplane with five cameras that have 80% length coverage and 60% side coverage. To recover the 3D water bodies that do not fulfil the Lambertian hypothesis, 2D vector maps and ortho-photos are used when performing the surface reconstruction. Furthermore, processing like aerial triangulation, dense image matching, and mesh surface reconstruction were all performed with ContextCapture. It should be noticed that the entire region of Helsinki is split into tiles, and each of them covers about 250 m^2 [37]. As shown in Figure 3, we have selected the central region of Helsinki as the study area, which includes 64 tiles and covers about 4 km^2 map area (8 km^2 surface area) in total.

3.2. Object Classes

We define the semantic categories for urban meshes by the most common objects in the urban environment with unambiguous geometry and texture appearance. Moreover, each triangle face is assigned to a label of one of the six semantic classes. Ambiguous regions (which account for about 2.6% of the total mesh surface area), such as shadowed regions or distorted surfaces, are labelled as unclassified (see Figure 4). The object classes we consider in the benchmark dataset are:

- **terrain**: roads, bridges, grass fields, and impervious surfaces;
- **building**: houses,high-rises, monuments, and security booths;
- **high vegetation**: trees, shrubs, and bushes;
- **water**: rivers, sea, and pools;
- **vehicle**: cars, buses, and lorries;
- **boat**: boats, ships, freighters, and sailboats;
- **unclassified**: incomplete objects like buses and trains, distorted surfaces like tables, tents and facades, construction sites, underground walls.

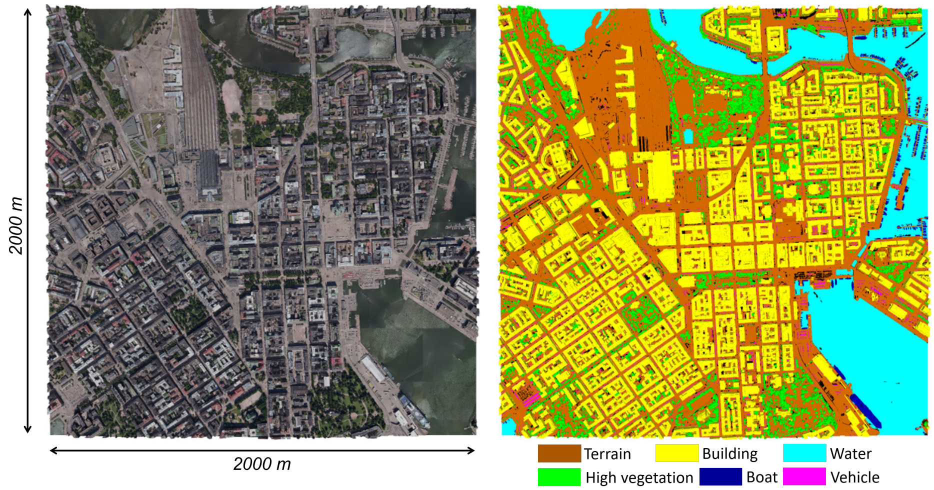


Figure 3: Overview of the semantic urban mesh benchmark. Left: the texture meshes covering about 4 km^2 map area. Right: the ground truth meshes. More views of the same scene (with different visualization styles) are shown in Figures 1 and 2.

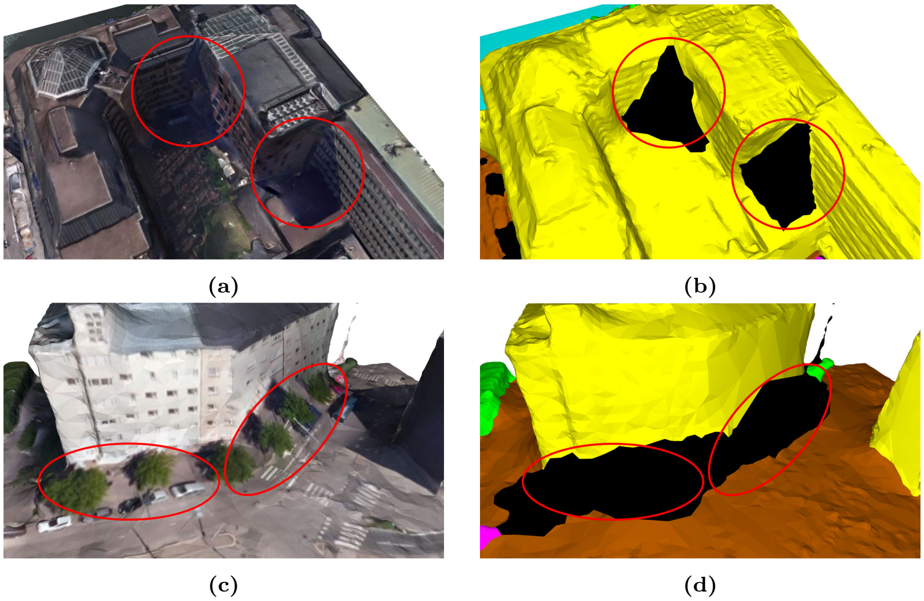


Figure 4: Ambiguous regions are labelled as unclassified (in black). (a) Shadow region with texture. (b) Shadow region with semantic colour. (c) Distorted region with texture. (d) Distorted region with semantic colour.

3.3. Semi-automatic Mesh Annotation

Rather than manually labelling each triangle face of the raw meshes, we design a semi-automatic mesh labelling framework to accelerate the labelling

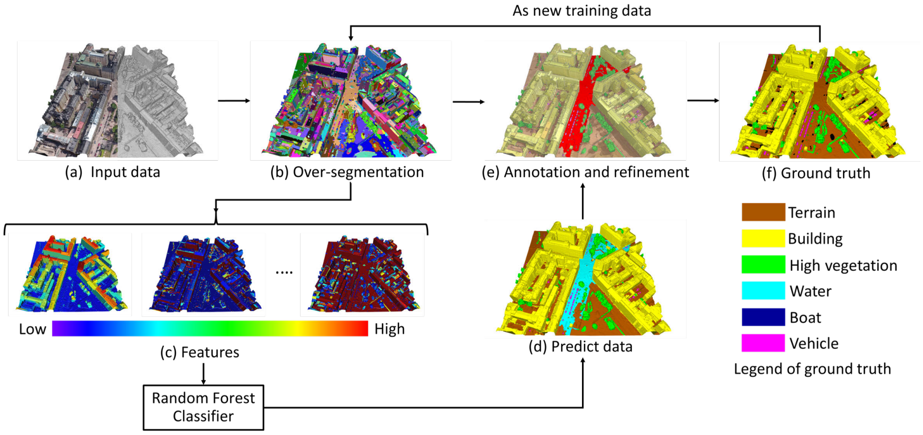


Figure 5: The pipeline of the labelling workflow.

process. Figure 5 shows the overall pipeline of our labelling workflow.

Given the fact that urban environments consist of a large number of planar regions in the data, we opt to label the data at the segment level instead of individual triangle faces. Specifically, we over-segment the input meshes into a set of planar segments. These segments can enrich local contextual information for feature extraction and serve as the basic annotation unit to improve annotation efficiency.

Instead of randomly choosing a mesh tile as input for annotation and refinement, which is insufficient for manual annotation progress, we favour picking a mesh tile that is more difficult to classify. Similar to active learning, we first compute the feature diversity (see Equation 1) to optimally select a mesh tile containing a variety of classes and objects at different scales and complexity. The feature diversity F_m of tile m is computed as

$$F_m = \frac{\sum_{i=1}^{N_f} \left(f_i - \bar{f}\right)^2}{N_f} \tag{1}$$

where f_i represents each handcrafted feature which describe in Section 3.3.1, and $\bar{f}$ is mean value of a N_f dimensional feature vector. To acquire the first ground truth data, we manually annotate the mesh (with segments) that is selected with the highest feature diversity. Then, we add the first labelled mesh into the training dataset for the supervised classification. Specifically, we use the segment-based features as input for the classifier, and the output is a pre-labelled mesh dataset. Next, we use the mesh annotation tool to manually refine the pre-labelled mesh according to the feature diversity. Finally, the new refined mesh will be added to the training dataset to improve the automatic classification accuracy incrementally.

3.3.1. Initial Segmentation

To avoid redundant computations of numerous triangles, we first apply mesh over-segmentation (i.e., linear least-squares fitting of planes) based on region growing on the input data to group triangle faces into homogeneous regions [38]. Such grouped regions are beneficial for computing local contextual features. We then extract both geometric and radiometric features from those mesh segments as follows:

- *Eigen-based features* are computed from the covariance matrix of the triangle vertices with respect to the average centre within each segment, which is beneficial for identifying urban objects with various surface distributions. The linearity $= (\lambda_1 - \lambda_2)/\lambda_1$, sphericity $= \lambda_3/\lambda_1$ and change of curvature $= \lambda_3/(\lambda_1 + \lambda_2 + \lambda_3)$ are computed based on the three eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0$. The local eigenvectors $\mathbf{n}_i$ and the unit normal vector $\mathbf{n}_z$ along Z-axis are used to compute the verticality $= 1 - |\mathbf{n}_i \cdot \mathbf{n}_z|$ [39]. Note that many eigen-based features have been studied in literature [39, 40, 41], and some of them were designed for and tested on LiDAR point clouds. These eigen-based features are mostly computed per point based on its spherical neighbourhood, which often contains noise and does not form a surface. Our chosen eigen-based features are defined on a segment representing the surface of a mesh, and thus they can capture non-local geometric properties of an object. Additionally, in this work, we have tested all eigen-based features from the literature [39], and we only present the ones that are effective for texture meshes.

- *Elevation* is divided into absolute elevation z_a, relative elevation z_r and multiscale elevations z_m. Where z_a is the average elevation of the segment; the relative elevation is computed as $z_r = z_a - z_{r_{min}}$; the multiscale elevation [42, 43] $z_m = \sqrt{\frac{z_a - z_{min}}{z_{max} - z_{min}}}$. And $z_{r_{min}}$ denotes the lowest elevation of the local largest ground segment computed within a cylindrical neighbourhood with 30 meters radius around the segment centre. z_{min} and z_{max} represent the local minimum and maximum elevation values of a cylindrical neighbourhood within the scale of 10 meters, 20 meters, and 40 meters. Such large cylindrical neighbourhoods allow to find the local ground considering the resilience to hilly environments, and the square root ensures that small relative height values (i.e., values smaller than 1 m) get a larger elevation attribute to enlarge elevation differences between small objects and the local ground (e.g., cars against the ground, boats against the water surfaces). More importantly, due to the influence of terrain fluctuations and various scales of urban objects, the elevation of these three categories can complement each other.

- *Segment area* is computed as $area(S_k) = \sum_{i=1}^{N} area(f_i)$, where f_i denotes a triangle of the segment S_k, and N denotes the total number of triangles in S_k.

- *Triangle density* is defined as $density(S_k) = \frac{N}{area(S_k)}$, which reveals the object complexity, especially for adaptive urban meshes.
- *Interior radius of 3D medial axis transform (InMAT)* [44, 45] of a segment S_k is formulated as $r_k = \frac{\sum_{i=1}^{M} r_i}{M}$, where M denotes the total number of triangle vertices of S_k, and r_i denotes the interior radius of the shrinking ball that touches the vertex v_i within the segment S_k. It is designed to distinguish objects with different scales.
- *HSV colour-based features* are derived from the RGB channel of the entire texture map. We use the HSV colour space since it can better differentiate different objects than RGB. We compute the average colour, the variance of the colour distribution of all pixels within each segment, and we further discretize it into a histogram that consists of 15 bins of the hue channel, five bins of the saturation channel, and five bins of the value channel.
- *Greenness* a_g is used to classify objects that are similar to green vegetation. Specifically, it is computed according to the averaged RGB colour of each segment via $a_g = G - 0.39 \cdot R - 0.61 \cdot B$ [46].

All the above features are concatenated into a 44-dimensional feature vector used by our random forest (RF) classifier in the initial segmentation.

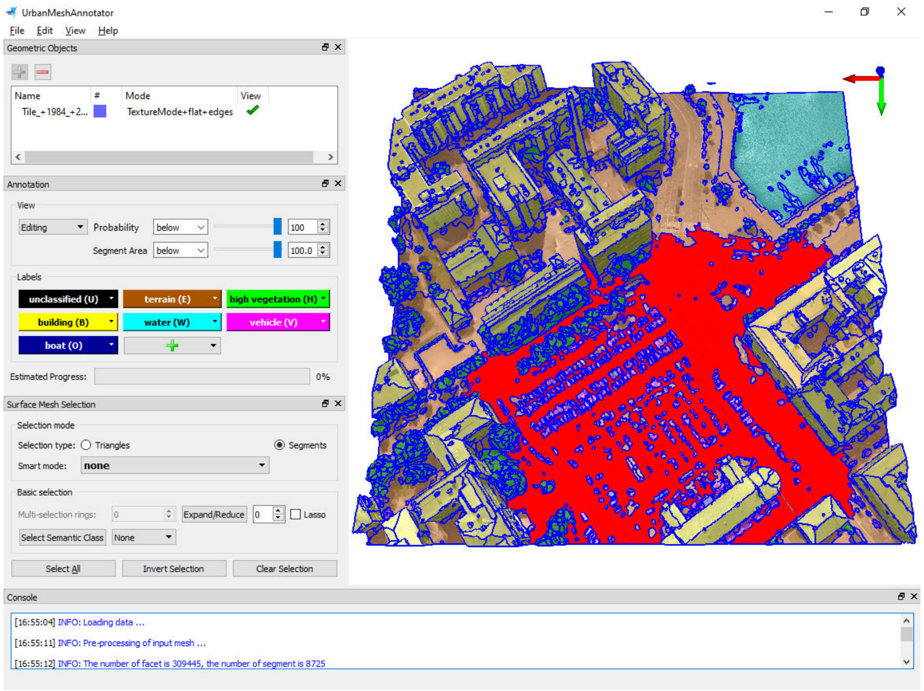
3.3.2. Annotation Tool for Refinement

Because of the under-segmentation errors and the imperfect results of the semantic mesh segmentation process, we design a mesh annotation tool (see Figure 6) to manually correct the labelling errors. Our mesh annotation tool is developed based on the labelling tool of CGAL [47].

As shown in Table 2, it consists of three operation categories: view, selection, and annotation. The view operations provide essential functions for the user to manipulate the scene camera, such as translate, rotate, zoom, or set the new pivot for the scene. In addition, to use textures as a reference for labelling, we map texture and face colour with a certain degree of transparency, and we visualize the segment border to differentiate each segment.

The selection operations allow the user to select or deselect either triangle faces (see Figure 7) or segments (see Figure 8) freely via a brush or a lasso. Specifically, the face selection operation is used to fix the under-segmentation errors and generate new segments, and the segment selection operation is to fix incorrect segment labels.

We also allow the user to edit the selection of each individual segment with splitting functions (see Figure 9) and automatic extraction of the most planar region (see Figure 10). As for splitting, we first detect the potential planar and non-planar segments marked by user strokes, and then the non-planar one is split according to the vertex-to-plane distance. It allows generating candidate non-planar regions (with respect to the detected planar segment) for the user to edit, and it is useful to split a segment that covers large non-planar regions or contains more than one dominant planar area. To extract the most planar



Name	#	Mode	View
Tile_+1984_+2...	🟦	TextureMode+flat+edges	✓

View: Editing ▾ Probability: below 100 · Segment Area: below 100.0

Estimated Progress: 0%

Selection type: ○ Triangles ● Segments

Smart mode: none

Multi-selection rings: 0 · Expand/Reduce 0 · □ Lasso

Select Semantic Class: None

```
[16:55:04] INFO: Loading data ...
[16:55:11] INFO: Pre-processing of input mesh ...
[16:55:12] INFO: The number of facet is 309445, the number of segment is 8725
```

Figure 6: The interface of our annotation tool for 3D texture meshes.

Categories	Operations	Objects
View	Translate	Camera
	Rotate	Camera
	Zoom in / out	Camera
	Set pivot	Camera
Selection	Multi-selection / Lasso	Triangles / Segments
	Expand / Reduce	Triangles / Segments
	Semantic selection	Segments
	Split region	Segments
	Planar region extraction	Triangles
	Split mesh	Triangles
Annotation	Probability slider	Segments
	Segment area slider	Segments
	Progress bar	Triangles
	Switch semantic view	Triangles
	Labelling	Triangles / Segments

Table 2: Basic operations in our annotation tool.

region, we apply the region growing algorithm [38] within the selected segment to

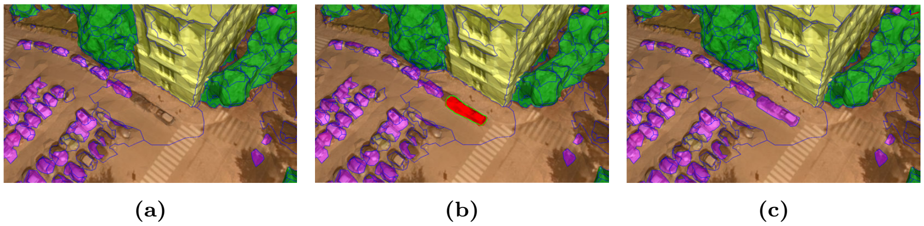


Figure 7: An example of labelling by selecting triangles using the lasso tool (blue edges: segment boundaries). (a) Before selection. (b) Lasso selection result (in red). (c) The correct label has been assigned to the selected region. In this example, the label of the selected region has been changed from ‘ground’ to ‘vehicle’.

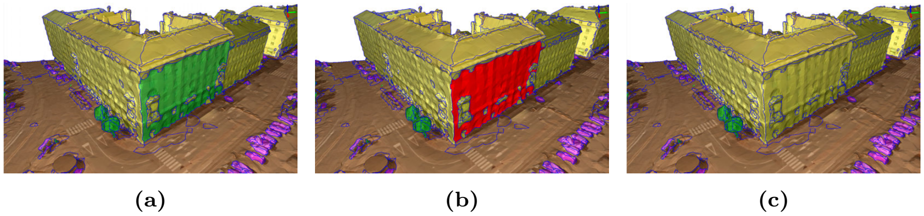


Figure 8: An example of segment labelling. (a) Part of a wall of the building was previously labelled as ‘high vegetation’ (in green). (b) Segment selection result (in red). (c) The label of the selected segment has been corrected with the new label ‘building’.

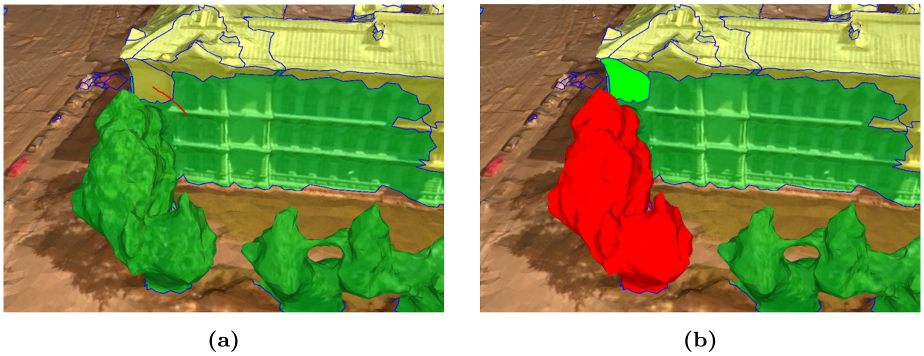


Figure 9: An example splitting planar and non-planar regions. (a) The user draws a stroke (in red) across the border of the non-planar segment and the planar segment. (b) The detected non-planar segment has been split into two parts (i.e., a non-planar region shown in red and a planar segment shown in green).

automatically generate the candidate triangle faces with user-defined thresholds (i.e., the maximum distance to the plane, the maximum accepted angle, and the minimum region size). Such an operation allows the user to filter out some small bumpy regions of the selected segment.

Besides, probability and area-based sliders and a progress bar are provided in the annotation panel to improve annotation efficiency and experience,

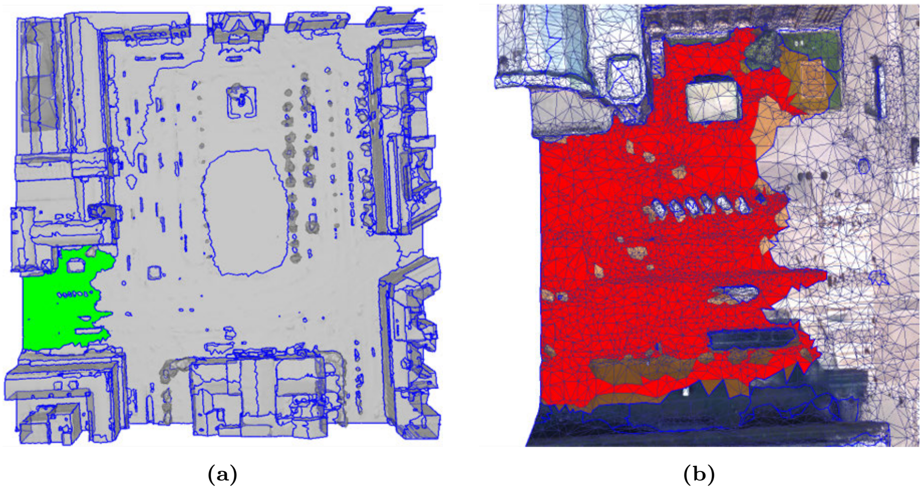


Figure 10: Editing an individual segment. (a) A segment is selected (highlighted in green) for splitting. (b) Automatic extraction of the most planar region (shown in red) within the selected segment according to user-defined thresholds.

respectively. Specifically, the probability slider is introduced for the user to visually inspect the segments that are most likely misclassified. Moreover, the user can further use it to inspect a specific class by switching the view to highlight a specific semantic class. The segment area slider is used to identify isolated tiny segments, which commonly appear as errors. The progress bar is used to indicate the estimated labelling progress during the annotation. After performing the selection, the user can easily assign the corresponding label to the selected area.

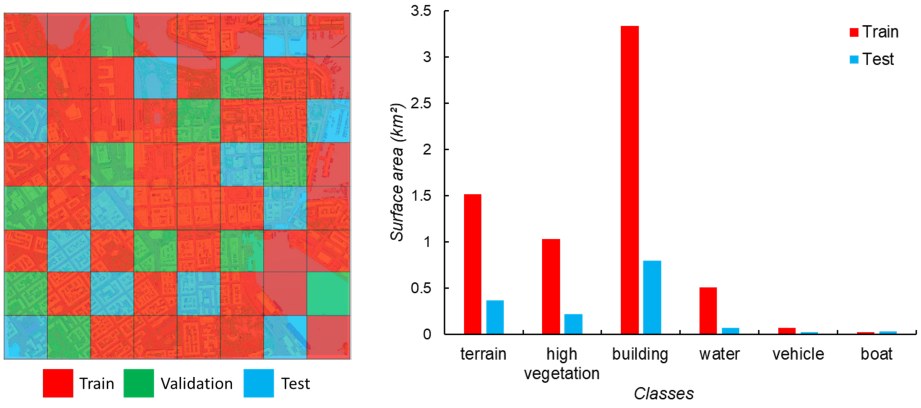
4. Experiments

4.1. Data Split

To perform the semantic segmentation task, we randomly select 40 tiles from the annotated 64 tiles of Helsinki as training data, 12 tiles as test data, and 12 tiles as validation data (see Figure 11 (a)). For each of the six semantic categories, we compute the total area in the training and test dataset to show the class distribution. As shown in Figure 11 (b), some classes, like vehicles and boats, only account for less than 5% of the total area, while the building and terrain together comprise more than 70%. The unbalanced classes impose significant challenges for semantic segmentation based on supervised learning.

4.2. Evaluation Metric

Since the triangle faces in the meshes have different sizes, we compute the surface area for semantic evaluation instead of using the number of triangles. The performance of semantic mesh segmentation is measured in precision, recall,



(a)

(b)

Figure 11: Overview of the data used in our experiment. (a) The distribution of the training, test, and validation dataset. (b) Semantic categories of training (including validation dataset) and test dataset.

F1 score, and intersection over union (IoU) for each object class. The evaluation of the whole test area is applied with overall accuracy (OA), mean per-class accuracy (mAcc), and mean per-class intersection over union (mIoU).

4.3. Evaluation of Initial Segmentation

We have implemented the semantic mesh segmentation and annotation tool in C++ using the open-source libraries include CGAL [47], Easy3D [48], and ETHZ random forest [49].

Our proposed pipeline for initial segmentation only takes a few input parameters, which are shown in Table 3. The over-segmentation is intended to find all planar regions in the model, for which we set the distance threshold to 0.5 meters. This threshold value specifies the minimum geometric features we would like the over-segmentation method to identify. In other words, the region growing-based over-segmentation method will not be able to distinguish two parallel planes with a distance smaller than this threshold. We set the angle threshold to 90 degrees, which is large enough to cope with high levels of noise (e.g., the distance value is small, but the angle between the triangle normal and the plane normal is large). Moreover, the minimum area is set to zero to allow planar segments of any arbitrary size. As for the random forest classifier, we set the parameters initially to those of Rouhani et al. [43] followed by fine-tuning using the validation data. Specifically, using 100 trees is sufficient to guarantee the stability of the model, and using the depth of 30 is adequate to avoid over-fitting and under-fitting for training.

Method	Parameters	Value
Region Growing	Minimum area	0 m^2
	Distance to plane	0.5 m
	Accepted angle	90°
Random Forest	Number of trees	100
	Maximum depth	30

Table 3: Parameters used in our approach.

Rather than classifying about 19 million triangle faces (i.e., the entire dataset), we use 515,176 segments that are clustered during over-segmentation. Although both semantic segmentation and labelling refinement can benefit from mesh over-segmentation, the degree of the under-segmentation error cannot be avoided. Since our mesh over-segmentation does not intend to retrieve the individual objects and the purpose is to perform semantic segmentation, we measure the maximum achievable performance by calculating the IoU instead of using under-segmentation errors to evaluate it. The upper bound IoU of each class we could achieve for semantic segmentation is presented in Table 4, and the upper bound mean IoU (mIoU) over all classes is about 90.9% as shown in Table 5. In addition, the results of our experiment in Tables 4 and 5 are reported based on the average performance of ten times experiments with the same configuration.

Class	Precision (%)	Recall (%)	F1 scores (%)	IoU (%)	Upper bound IoU (%)
Terrain	87.7	94.3	90.9	83.3	93.9
High Vegetation	96.3	93.8	95.0	90.5	96.2
Building	94.6	97.7	96.1	92.5	99.0
Water	97.0	88.3	92.5	86.0	92.7
Vehicle	77.9	41.7	54.4	37.3	73.2
Boat	77.9	7.5	13.7	7.4	90.5

Table 4: Overall evaluation of our method. The *Upper bound IoU* refers to the maximum achievable IoU in theory.

For semantic segmentation, a detailed evaluation of each class is listed in Table 4, and we achieve about 93.0% overall accuracy and 66.2% mIoU as shown in Table 5. The qualitative evaluation of it is shown in Figure 12. As shown in Figure 12 (e), most of the prediction errors occur at small-scale objects such as vehicles and boats due to fewer training samples and errors from over-segmentation.

Model	OA (%)	mAcc (%)	mIoU (%)	ΔmIoU (%)
Upper bound (Perfect)	98.1	91.6	90.9	——
Ours (best)	93.0	70.6	66.2	0.0
Without sphericity	93.0	70.5	66.1	-0.1
Without segment area	92.9	70.5	66.0	-0.2
Without triangle density	92.9	70.4	66.0	-0.3
Without variance HSV	92.9	70.3	65.9	-0.3
Without absolute elevation	93.0	70.2	65.9	-0.3
Without relative elevation	92.9	70.3	65.8	-0.4
Without curvature	92.9	70.2	65.8	-0.4
Without multiscale elevations	92.8	69.8	65.1	-1.1
Without linearity	91.8	66.6	62.0	-4.2
Without greenness	91.9	66.6	61.9	-4.3
Without InMat	91.6	66.4	61.6	-4.6
Without average HSV	91.7	66.1	61.4	-4.8
Without verticality	91.4	66.1	61.3	-4.9
Without HSV histogram bins	91.5	66.0	61.1	-5.1

Table 5: Ablation study of the features in our approach. The *Upper bound (Perfect)* refers to the maximum achievable performance in theory.

To better understand the relevance of the features, we measure the feature importance and perform ablation studies (see Table 5). We can observe that the radiometric features (which account for 62.8%) are more important than geometric ones (which account for 37.2%). Moreover, after removing individual feature vectors, the performance will decline, indicating each feature contributes to the best results.

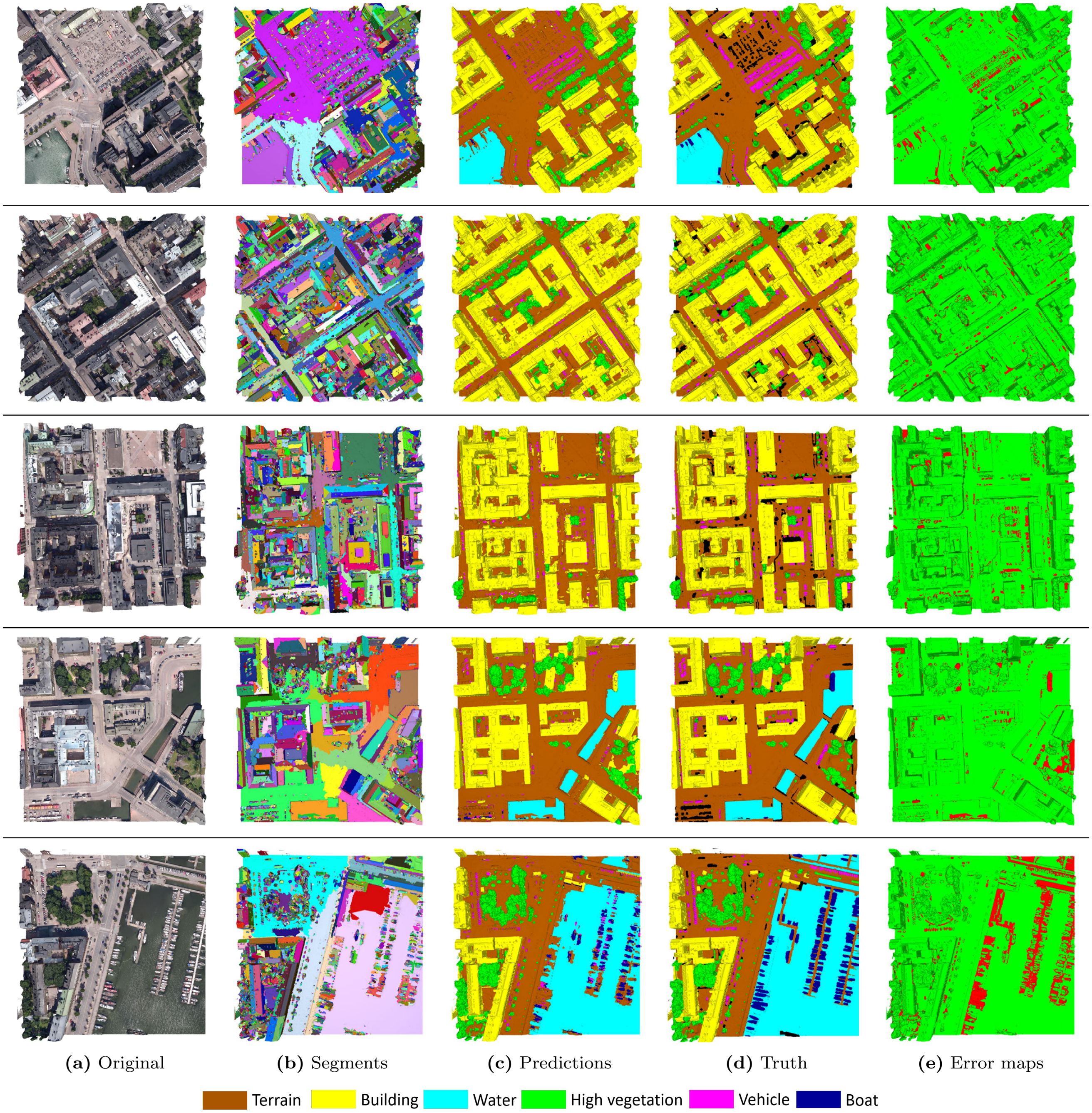
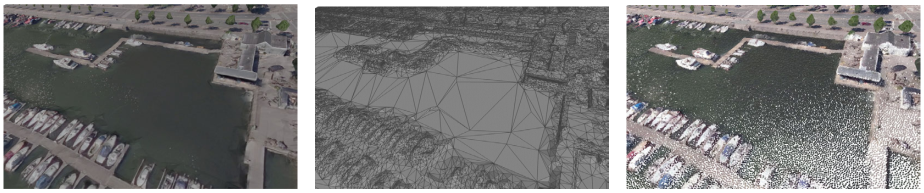

(a) Original (b) Segments (c) Predictions (d) Truth (e) Error maps

Figure 12: Part of our semantic segmentation results. The first column shows the input texture meshes; the second column shows the over-segmentation results; the third column shows the predicted semantic meshes; the fourth column shows the ground truth meshes; the last column shows the error maps (red: errors; green: correct labels).



(a) Texture mesh **(b)** Wireframe **(c)** Sampled point cloud

Figure 13: Sampling point cloud from texture meshes. Our sampled points preserve both geometric and radiometric information of the original mesh.

4.4. Evaluation of Competition Methods

To the best of our knowledge, none of the state-of-the-art deep learning frameworks of 3D semantic segmentation can directly be used on large-scale texture meshes. Additionally, although the data structures of point clouds and meshes are different, the inherent properties of geometry in the 3D space of the urban environment are nearly identical. In other words, they can share the feature vectors within the same scenes. Consequently, we sample the mesh into coloured point clouds (see Figure 13) with a density of about 10 pts/m^2 as input for the competing deep learning methods. In particular, we use Montecarlo sampling [50] to generate randomly uniform dense samples, and we further prune these samples according to Poisson distributions [51] and assign the colour via searching the nearest neighbour from the textures.

To evaluate and compare with the current state-of-the-art 3D deep learning methods that can be applied to a large-scale urban dataset, we select five representative approaches (i.e., PointNet [14], PointNet++ [52], SPG [15], KPConv [16], and RandLA-Net [53]). We perform all the experiments on an NVIDIA GEFORCE GTX 1080Ti GPU. Note that these deep learning-based methods downsample the input point clouds significantly as a pre-processing step. In our experiments, the point sampling density is limited by the GPU memory, and increasing or decreasing the sampling density within a reasonable range may lead to slightly different performance. It should be noted that no matter how dense the input point clouds are, almost all state-of-the-art deep learning architectures (such as PointNet, PointNet++, RandLaNet, KPConv, and SPG, etc.) downsample the input point clouds significantly, and they are still able to learn effective features for classification. Besides, different deep learning-based point cloud classification frameworks exploit different strategies for downsampling the input points. In addition, we also compare with the joint RF-MRF [43], which is the only competition method that directly takes the mesh as input and without using GPU for computation.

The hyper-parameters of all the competing methods are tuned according to the validation data to achieve the best results we could acquire. Besides, the results of each competitive method (see Table 6) are demonstrated in average performance based on ten times experiments with the same setting. From the comparison results, as shown in Table 6, we found that our baseline method

	Terrain	High Vegetation	Building	Water	Vehicle	Boat	mIoU	OA	mAcc	mF1	t_{train}
PointNet [14]	56.3	14.9	66.7	83.8	0.0	0.0	36.9 ± 2.3	71.4 ± 2.1	46.1 ± 2.6	44.6 ± 3.2	1.8
RandLaNet [53]	38.9	59.6	81.5	27.7	22.0	2.1	38.6 ± 4.6	74.9 ± 3.2	53.3 ± 5.1	49.9 ± 4.8	10.8
SPG [15]	56.4	61.8	87.4	36.5	34.4	6.2	47.1 ± 2.4	79.0 ± 2.8	64.8 ± 1.2	59.6 ± 1.9	17.8
PointNet++ [52]	68.0	73.1	84.2	69.9	0.5	1.6	49.5 ± 2.1	85.5 ± 0.9	57.8 ± 1.8	57.1 ± 1.7	2.8
RF-MRF [43]	77.4	87.5	91.3	83.7	23.8	1.7	60.9 ± 0.0	91.2 ± 0.0	65.9 ± 0.0	68.1 ± 0.0	**1.1**
KPConv [16]	**86.5**	88.4	**92.7**	77.7	**54.3**	**13.3**	**68.8** ± 5.7	**93.3** ± 1.5	**73.7** ± 5.4	**76.7** ± 5.8	23.5
Baseline	83.3	**90.5**	92.5	**86.0**	37.3	7.4	66.2 ± 0.0	93.0 ± 0.0	70.6 ± 0.0	73.8 ± 0.0	1.2

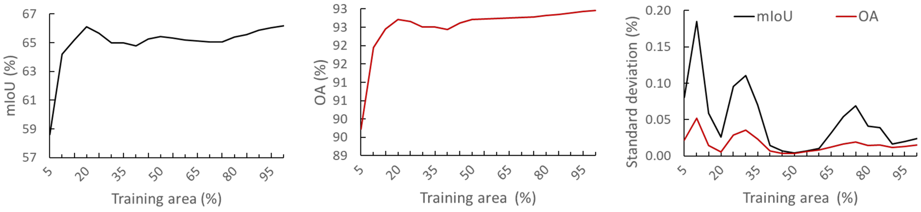
Table 6: Comparison of various semantic segmentation methods on the new benchmark dataset. The results reported in this table are per-class IoU (%), mean IoU (mIoU, %) ± standard deviation, Overall Accuracy (OA, %) ± standard deviation, mean class Accuracy (mAcc, %) ± standard deviation, mean F1 score (mF1, %) ± standard deviation, and the time cost of training (t_{train}, hours). The running times of SPG include both feature computation and graph construction, and RF-MRF and our baseline method include feature computation. We repeated the same experiment ten times and presented the mean performance.

outperforms other methods except for KPConv. Specifically, our approach outperforms RF-MRF with a margin of 5.3% mIoU, and deep learning methods (not including KPConv) from 16.7% to 29.3% mIoU. Compared with the KPConv, the performance of our method is much more robust, which can be observed from Table 6 that the standard deviation of our method is close to zero (i.e., the standard deviation of mIoU of our method is about 0.024%). The reason is that in our method, we set 100 trees in the random forest to ensure the stability of the model, but in KPConv, the kernel point initialization strategy may not be able to select some parts of the point cloud, which leads to the instability of the results. Furthermore, compared with all deep learning pipelines, our method is conducted on a CPU and uses much less time for training (including feature computation). This can be explained by the fact that we have fewer input data (triangles versus points), and the time complexity of our handcrafted features computation is much lower than the features learned from deep learning.

4.5. Evaluation of Annotation Refinement

Following the proposed framework, a total of 19,080,325 triangle faces have been labelled, which took around 400 working hours. Compared with a triangle-based manual approach, we estimate that our framework saved us more than 600 hours of manual labour. Specifically, we have measured the labelling speed with these two different approaches on the same mesh tile consisting of 309,445 triangle faces and 8,033 segments. It took around 17 hours for manual labelling based on triangle faces, while with our segment-based semi-automatic approach, it took only 6.5 hours.

We also evaluate the performance of semantic segmentation with different amounts of input training data on our baseline approach with the intention of understanding the required amount of data to obtain decent results. Specifically, we use ten sets of different training areas with ten times experiments with the same configuration of each set, and we linearly interpolate the results as shown in Figure 14. From Figures 14a, 14b, and 14c, we can observe that our initial



(a) mIoU **(b)** OA **(c)** Standard deviation

Figure 14: Effect of the amount of training data on the performance of the initial segmentation method used in the semi-automatic annotation. We repeated the same experiment ten times for each set of training areas and presented the mean performance.

segmentation method only requires about 10% (equal to about 0.325 km^2) of the total training area to achieve acceptable and stable results. In other words, using a small amount of ground truth data, our framework can provide robust pre-labelled results and significantly reduce the manually labelling efforts.

5. Conclusion

We have developed a semi-automatic mesh annotation framework to generate a large-scale semantic urban mesh benchmark dataset covering about 4 km^2. In particular, we have first used a set of handcrafted features and a random forest classifier to generate the pre-labelled dataset, which saved us around 600 hours of manual labour. Then we have developed a mesh labelling tool that allows the users to interactively refining the labels at both the triangle face and the segment levels. We have further evaluated the current state-of-the-art semantic segmentation methods that can be applied to large-scale urban meshes, and as a result, we have found that our classification based on handcrafted features achieves 93.0% overall accuracy and 66.2% of mIoU. This outperforms the state-of-the-art machine learning and most deep learning-based methods that use point clouds as input. Despite this, there is still room for improvement, especially on the issues of imbalanced classes and object scalability. For future work, we plan to label more urban meshes of different cities and extend our Helsinki dataset to include parts of urban objects (such as roof, chimney, dormer, and facade). We will also investigate smart annotation operators (such as automatic boundary refinement and structure extraction), which involve more user interactivity and may help reduce further the manual labelling task.

6. Acknowledgements

We would like to thank EuroSDR for providing the funding for this project. The authors appreciate the people who have helped the project, especially Ziqian Ni for the development and the testing of the annotation platform, and Mels Smit and Charalampos Chatzidiakos for assisting with the annotation of the meshes.

References

[1] F. Matrone, A. Lingua, R. Pierdicca, E. S. Malinverni, M. Paolanti, E. Grilli, F. Remondino, A. Murtiyoso, T. Landes, A benchmark for large-scale heritage point cloud semantic segmentation, The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XLIII-B2-2020 (2020) 1419–1426. `doi:10.5194/isprs-archives-XLIII-B2-2020-1419-2020`.

[2] T. Hackel, N. Savinov, L. Ladicky, J. D. Wegner, K. Schindler, M. Pollefeys, SEMANTIC3D.NET: A new large-scale point cloud classification benchmark, in: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. IV-1-W1, 2017, pp. 91–98.

[3] Google, 3D imagery in google earth, `https://earth.google.com/web/`, accessed: 2021-01-16 (dec 2012).

[4] C. of Helsinki, Helsinki's 3D city models, `https://www.hel.fi/helsinki/en/administration/information/general/3d`, accessed: 2020-11-25 (Dec. 2019).

[5] Actueel Hoogtebestand Nederland (AHN), `https://www.ahn.nl/`, accessed: 2021-04-16 (2019).

[6] C. Ran, The development of 3D city model and its applications in urban planning, in: 2011 19th International Conference on Geoinformatics, IEEE, 2011. `doi:10.1109/geoinformatics.2011.5981007`.

[7] K. Czyńska, P. Rubinowicz, Application of 3D virtual city models in urban analyses of tall buildings: today practice and future challenges, Architecturae et Artibus 6 (1) (2014) 9–13.

[8] C. Cappelle, M. E. El Najjar, F. Charpillet, D. Pomorski, Virtual 3D city model for navigation in urban areas, Journal of Intelligent & Robotic Systems 66 (3) (2012) 377–399.

[9] S. Peyraud, D. Bétaille, S. Renault, M. Ortiz, F. Mougel, D. Meizel, F. Peyret, About non-line-of-sight satellite detection and exclusion in a 3D map-aided localization algorithm, Sensors 13 (1) (2013) 829–847. `doi:10.3390/s130100829`.
URL `https://www.mdpi.com/1424-8220/13/1/829`

[10] H. Li-Ta, G. Yanlei, K. Shunsuke, NLOS correction/exclusion for GNSS measurement using RAIM and city building models, Sensors 15 (7) (2015) 17329–17349. `doi:10.3390/s150717329`.

[11] Y. Reda, Y. Mabrouk, K. Abdullah, K. Walid, HybVOR: A voronoi-based 3D GIS approach for camera surveillance network placement, ISPRS International Journal of Geo-Information 4 (2) (2015) 754–782. `doi:10.3390/ijgi4020754`.

[12] D. Yichuan, C. P. C. Jack, A. Chimay, A framework for 3D traffic noise mapping using data from BIM and GIS integration, Structure and Infrastructure Engineering 12 (10) (2016) 1267–1280. doi:10.1080/15732479.2015.1110603.

[13] C. García-Sánchez, D. Philips, C. Gorlé, Quantifying inflow uncertainties for CFD simulations of the flow in downtown oklahoma city, Building and Environment 78 (2014) 118–129. doi:10.1016/j.buildenv.2014.04.013.

[14] C. R. Qi, H. Su, K. Mo, L. J. Guibas, Pointnet: Deep learning on point sets for 3D classification and segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 652–660.

[15] L. Landrieu, M. Simonovsky, Large-scale point cloud semantic segmentation with superpoint graphs, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4558–4567.

[16] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, L. J. Guibas, Kpconv: Flexible and deformable convolution for point clouds, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 6411–6420.

[17] D. Munoz, J. A. Bagnell, N. Vandapel, M. Hebert, Contextual classification with functional max-margin markov networks, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 975–982.

[18] A. Serna, B. Marcotegui, F. Goulette, J.-E. Deschaud, Paris-rue-Madame database: a 3D mobile laser scanner dataset for benchmarking urban detection, segmentation and classification methods, in: 4th International Conference on Pattern Recognition, Applications and Methods ICPRAM 2014, 2014.

[19] B. Vallet, M. Brédif, A. Serna, B. Marcotegui, N. Paparoditis, TerraMobilita/iQmulus urban point cloud analysis benchmark, Computers & Graphics 49 (2015) 126–133.

[20] X. Roynard, J.-E. Deschaud, F. Goulette, Paris-Lille-3D: A large and high-quality ground-truth urban point cloud dataset for automatic segmentation and classification, The International Journal of Robotics Research 37 (6) (2018) 545–557.

[21] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, J. Gall, SemanticKITTI: A dataset for semantic scene understanding of lidar sequences, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 9297–9307.

[22] W. Tan, N. Qin, L. Ma, Y. Li, J. Du, G. Cai, K. Yang, J. Li, Toronto-3D: A large-scale mobile lidar dataset for semantic segmentation of urban roadways, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020, pp. 202–203.

[23] J. Niemeyer, F. Rottensteiner, U. Soergel, Contextual classification of lidar data and building object detection in urban areas, ISPRS journal of photogrammetry and remote sensing 87 (2014) 152–165.

[24] S. Zolanvari, S. Ruano, A. Rana, A. Cummins, R. E. da Silva, M. Rahbar, A. Smolic, Dublincity: Annotated lidar point cloud and its applications, in: BMVC 30th British Machine Vision Conference, 2019.

[25] N. Varney, V. K. Asari, Q. Graehling, Dales: A large-scale aerial lidar data set for semantic segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020, pp. 186–187.

[26] Z. Ye, Y. Xu, R. Huang, X. Tong, X. Li, X. Liu, K. Luan, L. Hoegner, U. Stilla, Lasdu: A large-scale aerial lidar dataset for semantic labeling in dense urban areas, ISPRS International Journal of Geo-Information 9 (7) (2020) 450.

[27] G. J. Brostow, J. Fauqueur, R. Cipolla, Semantic object classes in video: A high-definition ground truth database, Pattern Recognition Letters 30 (2) (2009) 88–97.

[28] H. Riemenschneider, A. Bódis-Szomorú, J. Weissenberg, L. Van Gool, Learning where to classify in multi-view semantic segmentation, in: European Conference on Computer Vision, Springer, 2014, pp. 516–532.

[29] X. Li, C. Li, Z. Tong, A. Lim, J. Yuan, Y. Wu, J. Tang, R. Huang, Campus3D: A photogrammetry point cloud benchmark for hierarchical understanding of outdoor scene, in: Proceedings of the 28th ACM International Conference on Multimedia, 2020, pp. 238–246.

[30] Q. Hu, B. Yang, S. Khalid, W. Xiao, N. Trigoni, A. Markham, Towards semantic segmentation of urban-scale 3d point clouds: A dataset, benchmarks and challenges, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 4977–4987.

[31] G. Can, D. Mantegazza, G. Abbate, S. Chappuis, A. Giusti, Semantic segmentation on swiss3dcities: A benchmark study on aerial photogrammetric 3D pointcloud dataset, arXiv preprint arXiv:2012.12996 (2020).

[32] D. Laupheimer, M. Shams Eddin, N. Haala, On the association of lidar point clouds and textured meshes for multi-modal semantic segmentation, ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences 5 (2) (2020).

[33] M. Kölle, D. Laupheimer, S. Schmohl, N. Haala, F. Rottensteiner, J. D. Wegner, H. Ledoux, 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 1 (2021) 100001. doi:https://doi.org/10.1016/j.ophoto.2021.100001.

[34] D. Girardeau-Montaut, CloudCompare, https://www.danielgm.net/cc/, accessed: 2021-01-16 (2016).

[35] B. Foundation, Blender, https://www.blender.org/, accessed: 2021-01-16 (2002).

[36] B. SYSTEMS, ContextCapture, https://www.bentley.com/zh/products/product-line/reality-modeling-software/contextcapture, accessed: 2021-01-16 (2016).

[37] KIGA-digi, The kalasatama digital twins project - the final report of the kira-digi pilot project, https://www.hel.fi/hel2/tietokeskus/data/helsinki/kaupunginkanslia/3D-malli/Helsinki3D_Kalasatama_Digital_Twins_020519.pdf, accessed: 2020-11-25 (May 2019).

[38] F. Lafarge, C. Mallet, Creating large-scale city models from 3d-point clouds: a robust approach with hybrid representation, International journal of computer vision 99 (1) (2012) 69–85.

[39] T. Hackel, J. D. Wegner, K. Schindler, Fast semantic segmentation of 3D point clouds with strongly varying density, ISPRS annals of the photogrammetry, remote sensing and spatial information sciences 3 (2016) 177–184.

[40] K. F. West, B. N. Webb, J. R. Lersch, S. Pothier, J. M. Triscari, A. E. Iverson, Context-driven automated target detection in 3d data, in: Automatic Target Recognition XIV, Vol. 5426, International Society for Optics and Photonics, 2004, pp. 133–143.

[41] M. Weinmann, B. Jutzi, C. Mallet, Feature relevance assessment for the semantic interpretation of 3D point cloud data, ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences 5 (W2) (2013) 1.

[42] Y. Verdie, F. Lafarge, P. Alliez, LOD generation for urban scenes, ACM Transactions on Graphics 34 (3) (2015) 1–14. doi:10.1145/2732527.

[43] M. Rouhani, F. Lafarge, P. Alliez, Semantic segmentation of 3D textured meshes for urban scene analysis, ISPRS Journal of Photogrammetry and Remote Sensing 123 (2017) 124–139. doi:10.1016/j.isprsjprs.2016.12.001.

[44] J. Ma, S. W. Bae, S. Choi, 3D medial axis point approximation using nearest neighbors and the normal field, The Visual Computer 28 (1) (2012) 7–19.

[45] R. Peters, H. Ledoux, Robust approximation of the medial axis transform of lidar point clouds as a tool for visualisation, Computers & Geosciences 90 (2016) 123–133.

[46] T. McKinnon, P. Hoff, Comparing rgb-based vegetation indices with NDVI for drone based agricultural sensing, Agribotix. Com 21 (17) (2017) 1–8.

[47] The CGAL Project, CGAL User and Reference Manual, 5.1.1 Edition, CGAL Editorial Board, 2020.
URL `https://doc.cgal.org/5.1.1/Manual/packages.html`

[48] L. Nan, Easy3d: a lightweight, easy-to-use, and efficient c++ library for processing and rendering 3D data, `https://github.com/LiangliangNan/Easy3D` (2018).

[49] S. Walk, Ethz random forest, `https://prs.igp.ethz.ch/research/Source_code_and_datasets.html`, accessed: 2020-11-25 (2014).

[50] P. Cignoni, C. Rocchini, R. Scopigno, Metro: measuring error on simplified surfaces, in: Computer graphics forum, Vol. 17, Wiley Online Library, 1998, pp. 167–174.

[51] M. Corsini, P. Cignoni, R. Scopigno, Efficient and flexible sampling with blue noise properties of triangular meshes, IEEE transactions on visualization and computer graphics 18 (6) (2012) 914–924.

[52] C. R. Qi, L. Yi, H. Su, L. J. Guibas, Pointnet++: Deep hierarchical feature learning on point sets in a metric space, Advances in neural information processing systems 30 (2017) 5099–5108.

[53] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, A. Markham, Randla-net: Efficient semantic segmentation of large-scale point clouds, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 11108–11117.