ISPRS Journal of Photogrammetry and Remote Sensing 196 (2023) 32–44

Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

journal homepage: www.elsevier.com/locate/isprsjprs

PSSNet: Planarity-sensible Semantic Segmentation of large-scale urban meshes

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

ARTICLE INFO

Keywords:
Texture meshes
Semantic segmentation
Over-segmentation
Urban scene understanding

ABSTRACT

We introduce a novel deep learning-based framework to interpret 3D urban scenes represented as textured meshes. Based on the observation that object boundaries typically align with the boundaries of planar regions, our framework achieves semantic segmentation in two steps: planarity-sensible over-segmentation followed by semantic classification. The over-segmentation step generates an initial set of mesh segments that capture the planar and non-planar regions of urban scenes. In the subsequent classification step, we construct a graph that encodes the geometric and photometric features of the segments in its nodes and the multi-scale contextual features in its edges. The final semantic segmentation is obtained by classifying the segments using a graph convolutional network. Experiments and comparisons on two semantic urban mesh benchmarks demonstrate that our approach outperforms the state-of-the-art methods in terms of boundary quality, mean IoU (intersection over union), and generalization ability. We also introduce several new metrics for evaluating mesh over-segmentation methods dedicated to semantic segmentation, and our proposed over-segmentation approach outperforms state-of-the-art methods on all metrics. Our source code is available at https://github.com/WeixiaoGao/PSSNet.

1. Introduction

Recent advances in photogrammetry and 3D computer vision have enabled the generation of textured meshes of large-scale urban scenes that contain buildings, trees, vehicles, etc. (City of Helsinki, 2019; Google, 2012; Gao et al., 2021). Deriving semantic information from the mesh models is critical to allowing the use of these meshes in diverse applications, e.g., energy estimate, noise modeling, and solar potential (Biljecki et al., 2015; Saran et al., 2015; Besuievsky et al., 2018).

There exists a large volume of machine learning-based algorithms for the semantic segmentation of 3D data, and they are designed mainly for 3D point clouds (Demantké et al., 2011; Hackel et al., 2016; Qi et al., 2017a,b; Thomas et al., 2018, 2019). A few recent works also address deep learning for surface meshes (Hanocka et al., 2019; Gao et al., 2019; Selvaraju et al., 2021; Fu et al., 2021) but are limited to individual objects or small indoor scenes (e.g., living room, kitchen). Unlike point clouds that are usually obtained as the raw input from typical data acquisition devices, textured meshes (see Fig. 2(a)) provide topological information, have continuous surfaces, yield better visualization, and are lightweight, which makes them an ideal representation for urban scenes. Surprisingly, the semantic segmentation of urban meshes has rarely been investigated, Verdie et al. (2015), Rouhani et al. (2017), Gao et al. (2021) are exceptions.

In this work, we address the semantic segmentation of urban meshes by introducing a two-step framework using deep learning. Our framework is designed to improve the following three aspects of semantic segmentation:

(1) Segmentation quality. Urban scenes typically contain piecewise regions, which can already inspire the separation of man-made objects (e.g., roads, buildings) from organic objects (e.g., trees). We observe that semantic segmentation algorithms usually perform well in the interior of large smooth surfaces (including planar surfaces), but that they perform poorly for the identification of object boundaries. Given the fact that object boundaries typically align with the boundaries of planar regions (see Fig. 1), our framework achieves semantic segmentation by first exploiting a planarity-sensible over-segmentation step that separates *planar* and *non-planar* surface patches.

(2) Descriptiveness of geometric features. Existing methods for semantic segmentation of 3D data commonly rely on features defined on local primitives (i.e., points or triangles) (Weinmann et al., 2013, 2015; Qi et al., 2017a; Huang et al., 2019; Li et al., 2019; Schult et al., 2020) or segments (i.e., a group of points or triangles) (Lin et al., 2018;

* Corresponding author.
E-mail addresses: w.gao-1@tudelft.nl (W. GAO), liangliang.nan@tudelft.nl (L. Nan), bas.boom12@gmail.com (B. Boom), h.ledoux@tudelft.nl (H. Ledoux).

https://doi.org/10.1016/j.isprsjprs.2022.12.020
Received 22 April 2022; Received in revised form 21 November 2022; Accepted 22 December 2022
Available online 2 January 2023

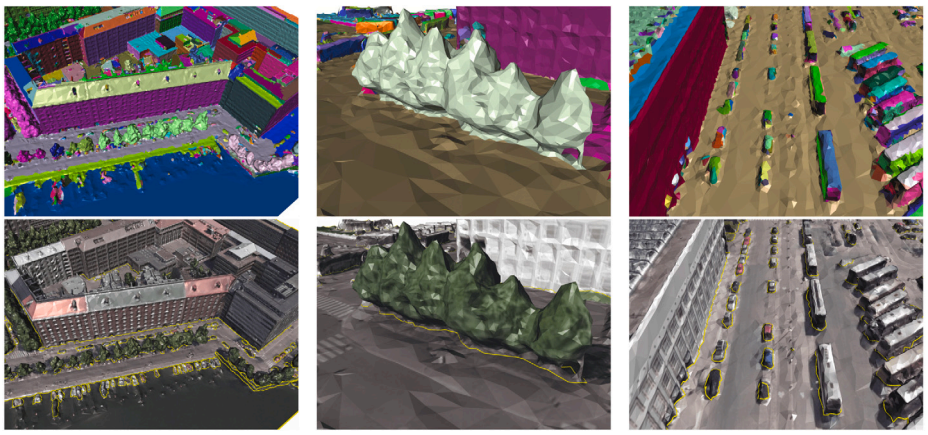


Fig. 1. Object boundaries often align with the boundaries of planar regions. Top: planar segmentation results; Bottom: the corresponding ground truth object boundaries (shown as yellow lines). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

(a) Input mesh (b) Over-segmented mesh (c) Semantic mesh

Fig. 2. The workflow of our method. We first decompose the input mesh (a) into a set of *planar* and *non-planar* segments (b). Then we classify the segments using graph convolutional networks to obtain the results of semantic segmentation (c). In (b), the segments are randomly colorized. In (c), the colors are: terrain, building, high vegetation, water, vehicle, boat. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Landrieu and Simonovsky, 2018; Cohen-Steiner et al., 2004; Lafarge and Mallet, 2012; Verdie et al., 2015; Rouhani et al., 2017). Features from local primitives are limited to a certain distance in the local neighborhood, while features used in existing segment-based approaches do not effectively capture the contextual relationships between segments. Thus, they are less descriptive in representing the complex shapes of diverse objects and in revealing the relationships between objects. In our work, by initially decomposing a mesh model into *planar* and *non-planar* segments, both local geometric features of individual segments and global relationships between segments can be captured.

(3) Efficiency. Existing deep learning-based methods for processing 3D data are limited by the data size, especially for large-scale urban scenes. This has already motivated over-segmentation for semantic segmentation (Weinmann et al., 2015; Landrieu and Simonovsky, 2018; Landrieu and Boussaha, 2019; Hui et al., 2021). Following the spirit of the previous work for improving efficiency, our over-segmentation facilitates better object boundaries and strengthens semantic segmentation by distinctive local and non-local features, which is suitable for the subsequent classification using graph convolutional networks (GCN).

Besides the two-step semantic segmentation framework, we also introduce several new metrics for evaluating mesh over-segmentation techniques. We believe the proposed metrics will further stimulate the improvement of over-segmentation for semantic segmentation.

Experiments on two benchmarks show that our approach outperforms recently developed methods in terms of boundary quality, mean IoU (intersection over union), and generalization ability.

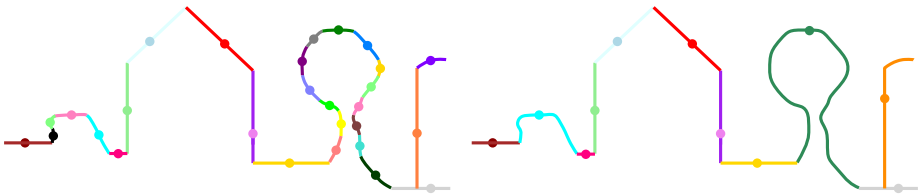
In summary, our contributions are: (1) a novel mesh over-segmentation approach for extracting planarity-sensible segments that are dedicated for GCN-based semantic segmentation; (2) a new graph structure that encodes both local geometric and photometric features of segments, as well as global spatial relationships between segments; (3) several novel metrics for evaluating mesh over-segmentation techniques in the context of semantic segmentation.

2. Related work

While there is a large volume of research on the over-segmentation and semantic segmentation of urban images (Cordts et al., 2016; Yang et al., 2018), we focus in this sole section on methods designed to process large-scale 3D data, i.e., point clouds and meshes of urban scenes. Methods specially designed for handling individual objects or small scenes (Nan et al., 2012; Hanocka et al., 2019; Gao et al., 2019; Selvaraju et al., 2021; Fu et al., 2021) usually do not scale to large-scale urban scenes and thus are not covered.

2.1. Over-segmentation of 3D data

Many methods for over-segmentation of 3D data are inspired by image over-segmentation algorithms (Liu et al., 2011) and can be divided into four categories: (1) primitive-based fitting (Vosselman et al., 2004; Schnabel et al., 2007; Lafarge and Mallet, 2012), (2) graph-based partitioning (Landrieu and Simonovsky, 2018; Ben-Shabat et al., 2018), (3) local region expansion (Cohen-Steiner et al., 2004; Melzer, 2007; Lafarge and Mallet, 2012; Papon et al., 2013; Rouhani et al., 2017; Vosselman et al., 2017; Papon et al., 2013; Lin et al., 2018), and (4) learning-based methods (Landrieu and Boussaha, 2019; Hui et al., 2021). Over-segmentation often serves as pre-processing for tasks such as semantic segmentation, instance segmentation, or reconstruction, and aims at reducing the complexity of subsequent tasks by using fewer segments having local homogeneity. Due to the complexity of real-world scenes and the irregularity of the data, it is challenging to obtain over-segmentation results with a desired number of segments and clear object boundaries. The aforementioned methods are either limited by the primitive types (e.g., plane, sphere, and cylinder) or suffer from severe under-segmentation errors when the number of segments is reduced or by the type of available labels in the training data (e.g., a few methods require instance labels (Landrieu and Boussaha, 2019; Hui et al., 2021)). We propose to partition the input



(a) Planar segments (b) Planarity-sensible segments

Fig. 3. 2D illustrative comparison between planar and planarity-sensible segments. Each dot and its line denote a segment.

meshes into a relatively small number of homogeneous regions with clear object boundaries based on both geometric and photometric characteristics (see Section 3), which is beneficial to semantic segmentation (see Section 4).

2.2. Semantic segmentation of 3D data

An important step in semantic segmentation is feature extraction. Based on the methods used for feature extraction, semantic segmentation approaches can be roughly categorized into three groups: handcrafted-feature-based (Demantké et al., 2011; Weinmann et al., 2013; Verdie et al., 2015; Weinmann et al., 2015; Hackel et al., 2016; Rouhani et al., 2017; Vosselman et al., 2017; Thomas et al., 2018), learning-based (Qi et al., 2017a,b; Thomas et al., 2019; Hu et al., 2020; Lei et al., 2021), and hybrid methods (Landrieu and Simonovsky, 2018; Wu et al., 2021). Handcrafted features are often effective when with limited training data. In contrast, deep-learning techniques are more effective when sufficient training data is available (Guo et al., 2020). These methods usually require contextual information to compute or learn features. However, it is difficult to capture effective global contextual features. Inspired by SPG (Landrieu and Simonovsky, 2018), our graph structure encodes various local geometric, photometric, and contextual features, and we apply a GCN for semantic segmentation. Our method exploits enriched spatial relationships in the graph at both local and global scales, which greatly facilitates the GCN model to capture contextual information and learn distinctive features for semantic segmentation.

3. Methodology

Our framework for semantic segmentation of urban meshes has two steps (as shown in Fig. 2):

Planarity-sensible over-segmentation. This step decomposes the urban mesh into a set of *planar* and *non-planar* surface patches because object boundaries often align with the boundaries of planar regions. This step not only enhances the descriptiveness of the features learned through local context but also significantly reduces the number of segments to be classified.

Segmentation classification. We construct a graph with its nodes encoding the local geometric and photometric features of the segments and its edges encoding global contextual features. We achieve semantic segmentation of the mesh by classifying the segments using a graph convolutional network.

3.1. Planarity-sensible over-segmentation

This step aims to decompose the urban mesh into a set of homogeneous segments in terms of geometric and photometric characteristics, see Fig. 3. Compared with *planar* segments generated by classical region growing methods (Lafarge and Mallet, 2012), our segments can accommodate more complex surfaces (i.e., trees and vehicles). Our over-segmentation, further detailed below, is achieved in two steps: (1) *planar* and *non-planar* classification, and (2) incremental segmentation.

Planar and non-planar classification. We classify the triangle faces of a mesh as either *planar* or *non-planar*. Following Gao et al. (2021), we design a set of features including Eigen-based (i.e., *linearity*, *planarity*, *sphericity*, *curvature*, and *verticality*), elevation-based (i.e., *absolute*, *relative*, and *multi-scale*), scale-based (i.e, *InMAT radius* (Ma et al., 2012; Peters and Ledoux, 2016): interior shrinking ball radius of 3D medial axis transformation), density-based (i.e., the number of vertices and the density of triangle faces), and color-based (i.e., *greenness* and *HSV histograms*) features, and we concatenate these features into a feature vector $\mathbf{F}_i$. We then use random forest (RF) (Geurts et al., 2006) to learn the probability of a face being *non-planar* as

$$G_i(L) = \frac{1}{|\tau|}\sum_{t\in\tau}\log\big(P_t(l_i \mid \mathbf{F}_i)\big), \tag{1}$$

where τ is a set of decision trees, and the predicted probability from decision tree t is denoted by $P_t \in [0,1]$. $L = \{0,1\}$ represents the potential labels of a face i (i.e., $l_i = 0$ for *planar* and $l_i = 1$ for *non-planar*). We learn a probability map (instead of binary classification) for the subsequent segment aggregation.

Incremental segmentation. Grouping all triangles into segments in one step using graph cuts would require the total number of segments, which is often not a priori. Therefore, we use the learned *planar* and *non-planar* probability maps to incrementally aggregate the mesh faces into a set of locally homogeneous segments. Inspired by Lafarge and Mallet (2012), we accumulate faces for a segment by solving a binary labeling problem. Starting from the face with the highest *planar* probability (i.e., the current region r has only a starting face at the beginning), we incrementally gather its neighboring face i to the current region r based on the labeling outcome of face i. The growing process is illustrated in Fig. 4. Our idea is to grow a region r if its neighboring face i receives the same label. We exploit a Markov Random Field (MRF) formulation to select the most suitable face for the aggregation in each growing iteration. The energy function $U(X)$ is defined as the sum of a unary term $\psi_i(x_i)$ and a pairwise term $\varphi_{i,r}(x_i, x_r)$, i.e.,

$$U(X) = \lambda_d \cdot \sum_{i\in A}\psi_i(x_i) + \lambda_m \cdot \sum_{i\in A}\varphi_{i,r}(x_i, x_r), \tag{2}$$

where A denotes the neighboring faces of the current growing region (i.e., the faces directly connected to r). x_i and x_r denote the binary labels that will be received by face i and region r, respectively. A neighboring face can be added to the current region only if it receives the same label as the current region. In our implementation, we fix the label of the current region to 0 (i.e., $x_r \equiv 0$) before minimizing the energy function. The face i is added to r only when $x_i = 0$ after the optimization. $\lambda_d \geq 0$ and $\lambda_m \geq 0$ are the weights balancing the unary and pairwise terms. A larger λ_d can lead to an excessive number of segments with smaller under-segmentation errors (see Figs. 5(a) and 5(b)). In contrast, a larger λ_m can result in fewer segments but may introduce larger under-segmentation errors (see Fig. 5(c)).

The **unary term** $\psi_i(x_i)$ measures the penalty of assigning a label x_i to a face i. To define this term, we consider the geometric distance (for *planar* regions) and the probability map (for *non-planar* regions), which is formulated as

$$\psi(x_i) = \begin{cases} \min\{\mathrm{d}(f_i, p_r), C_i\}, & \text{if } x_i = 0 \\ 1 - \min\{\mathrm{d}(f_i, p_r), C_i\}, & \text{if } x_i = 1 \end{cases}, \tag{3}$$

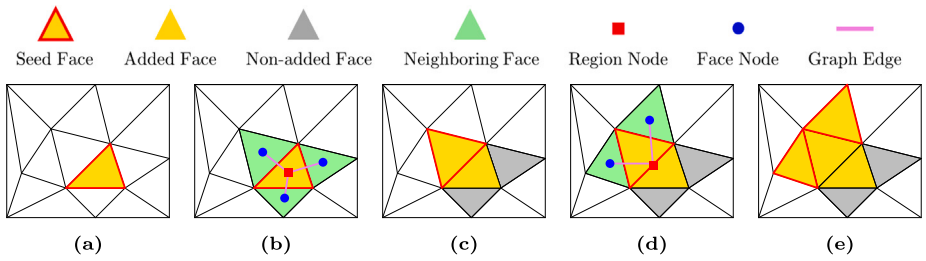


Fig. 4. An illustration of the first few steps of incremental segmentation. (a) The seed face with the highest *planar* probability is shown in gold color. (b) The local graph is constructed on the seed face (represented by the region node) and its three neighboring faces (represented by the face node). (c) The labeling outcome of the Markov random field (MRF): one newly added face is used as a seed face, and two non-added faces will be labeled as visited faces for the current growth step. (d) A local graph is constructed based on the growing region (represented by the region node) and two neighboring faces (represented by the face node). (e) The new labeling outcome, where the newly added two faces will be used as seed faces for the next growing step. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

$$C_i = \begin{cases} 1 - \lambda_g \cdot G_i, & \text{if } l_i = 1 \wedge l_r = 1 \\ \infty, & \text{otherwise} \end{cases}, \tag{4}$$

where $\mathrm{d}(f_i, p_r)$ measures the Euclidean distance between the farthest vertex of face i and the fitted plane p_r of the region r. The plane is obtained by linear least squares fitting using all the vertices of the region and dynamically updated when a new face has been added. During growing, when $x_i = 0$, $\min\{\mathrm{d}(f_i, p_r), C_i\}$ measures the cost of assigning face i the same label as the current region r (i.e., the cost of adding face i to the current region r). On the contrary, the cost is measured by $1 - \min\{\mathrm{d}(v_i, p_r), C_i\}$ when $x_i = 1$. In particular, for the *planar* case, since $C_i = \infty$, the geometric distance $\mathrm{d}(f_i, p_r)$ is actually used as the cost measure. For the *non-planar* case, the prior term G_i (see Eq. (1)) is considered to define the cost C_i. $\lambda_g \geq 0$ is a weight that controls the relative numbers of *planar* and *non-planar* segments (see Figs. 5(a) and 5(d)).

The **pairwise term** $\varphi_{i,r}(x_i, x_r)$ is designed to control the smoothness degree during the growing process,

$$\varphi_{i,r}(x_i, x_r) = \angle(\mathbf{n}_i, \mathbf{n}_r) \cdot \mathbb{1}(x_i \neq x_r), \tag{5}$$

where $\mathbf{n}_i$ and $\mathbf{n}_r$ denote the normals of a neighboring triangle face i and the region r, respectively. This term encodes the angle between these normal vectors, to reduce the normal deviation within the local neighborhood in the segmentation. $\mathbb{1}(x_i \neq x_r)$ is an indicator function that measures the coherence between x_i and x_r.

The energy $U(X)$ is minimized using the $\alpha - \beta$ swap graph cut algorithm (Boykov et al., 2001) to accumulate a face for the current segment. The growth of a segment stops if no more faces can be accumulated. We then restart growing a new segment from the face with the highest *planar* probability in the remaining set of faces. The growing of segments is repeated until all mesh faces have been processed.

3.2. Classification

We construct a graph whose nodes encode features of the segments and edges encode interactions between segments. With this graph, the semantic segmentation of the mesh is achieved by classifying the segments using GCN.

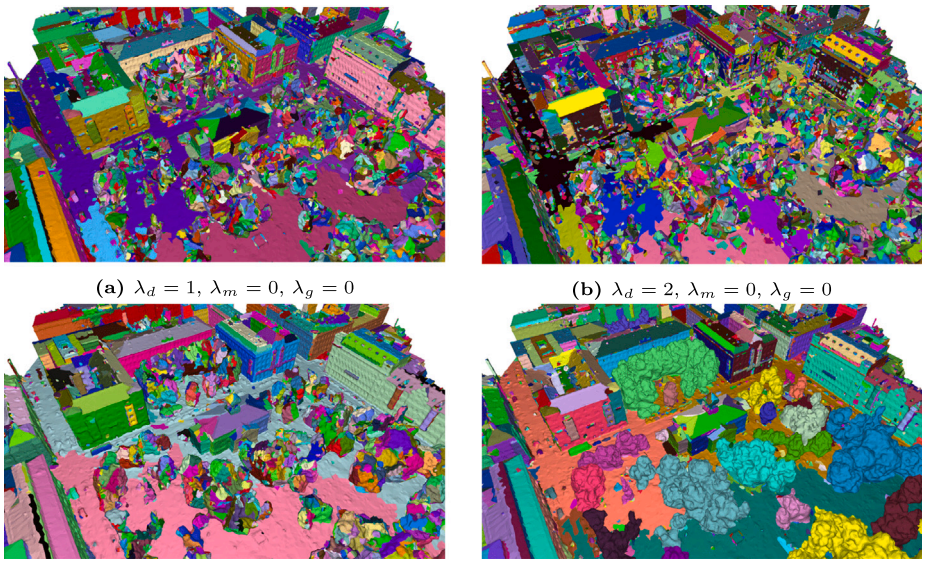
Node feature embedding. In our graph, each node represents a segment and it encodes two types of features generated based on the vertices and face centroids of the segment: (1) $F_l(s_k)_{256}$ is learned using PointNet (Qi et al., 2017a), and the input to it is a point cloud of randomly sub-sampled points from mesh vertices and face centroids. The size of the feature vector is 128×6 and consists of XYZ and RGB; (2) $F_h(s_k)_{48}$ is generated from the handcrafted feature generator (HFG), and it contains the same type of features used for *planar and non-planar classification* (see in Section 3.1) and four additional shape-based features capturing local geometric differences (see Table 1).

Table 1
Shape-based features defined on segments. $C(s_k)$ is the circumference of a segment s_k. λ_2 and λ_3 are the eigenvalues derived from the linear fitting line of m boundary points in 3D (Karl Pearson, 1901). *Avg Distance* measures the average distance from n mesh vertices p_i to the supporting plane P_k of the segment.

Compactness	$CP_k = \frac{4 \cdot \pi \cdot area(s_k)}{C(s_k)^2}$	Shape Index	$SI_k = \frac{C(s_k)}{\sqrt[4]{area(s_k)}}$
Straightness	$SD_k = \frac{\sum_{i=1}^{m} \frac{\lambda_2}{\lambda_3}}{m}$	Avg Distance	$D_k = \frac{\sum_{n}^{i=1} dist\left(p_i, P_k\right)}{n}$

Edge feature embedding. In contrast to graphs defined on 3D points or triangle faces, graphs defined on certain segmentation of the data can better capture global contextual relationships than simple adjacency connections. The idea behind the designed graph is to give more prominence to the segment differences that are usually present in urban scenarios. To this end, we intend to include global features that fulfill two conditions. First, the global features should be generalizable, i.e., they can be captured in different scenarios. Second, the global features should be established between graph nodes that have large feature differences. To make full use of the *planar* and *non-planar* segments and establish meaningful relationships between the segments, we propose a graph consisting of the following four types of edges (see Fig. 6):

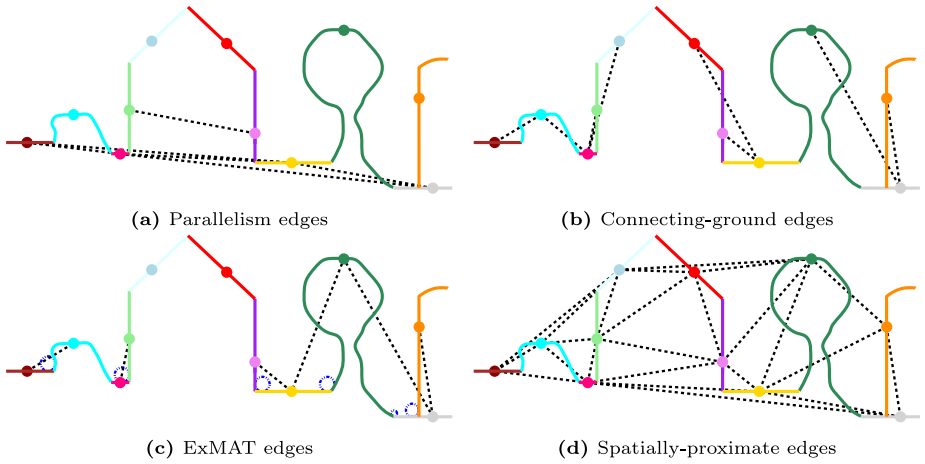
(1) *parallelism edges*: edges connecting parallel *planar* segments. Two *planar* segments are considered parallel if the angle between their supporting planes is smaller than a threshold (5° in our experiments). These edges mainly connect the *planar* segments belonging to man-made objects.
(2) *connecting-ground edges*: edges connecting segments and their local ground planes. A local ground plane is identified as the lowest and largest *planar* segment in a cylindrical neighborhood (30 m in our experiments) around the boundary vertices of the segment. These edges primarily capture the relationship between the ground and all non-ground objects.
(3) *exterior medial axis transform (ExMAT) edges*. We first build the ExMAT (i.e., exterior shrinking ball radius of 3D medial axis transformation) (Ma et al., 2012; Peters and Ledoux, 2016) on the segments, and we introduce graph edges that link the segments connected by the exterior shrinking ball (see Fig. 6(c)). Since the external skeleton usually corresponds to the joints between objects, ExMAT edges allow connecting segments that are adjacent but belong to different objects.
(4) *spatially-proximate edges*. We first build a 3D Delaunay triangulation (Jaromczyk and Toussaint, 1992) with the input mesh vertices and the centroids of mesh faces. Two segments are connected by an edge if at least one pair of points from the two segments are connected by a Delaunay edge. This type of edges allow the encoding of contextual information on different scales. Particularly, these edges contribute to capturing the relationships between urban objects from short-range (for objects



(a) $\lambda_d = 1,\ \lambda_m = 0,\ \lambda_g = 0$ (b) $\lambda_d = 2,\ \lambda_m = 0,\ \lambda_g = 0$

(c) $\lambda_d = 1,\ \lambda_m = 1,\ \lambda_g = 0$ (d) $\lambda_d = 1,\ \lambda_m = 0,\ \lambda_g = 1$

Fig. 5. The effect of the parameters λ_d, λ_m, and λ_g on the over-segmentation. These parameters provide control over the size and boundary smoothness of the segments.



(a) Parallelism edges (b) Connecting-ground edges

(c) ExMAT edges (d) Spatially-proximate edges

Fig. 6. An illustration of the four types of edges in our graph. Each color indicates a segment encoded as a node (i.e., the colored dot on each segment) in the graph. The dash lines denote the graph edges. In (c), the blue circles represent the exterior shrinking balls. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

that are close to the ground) to long-range (for objects that are far away from the ground).

With the above graph edges, we define the edge feature $F_h(e_{k,k+1}) = log(F_h(s_k)/F_h(s_{k+1}))$, where s_k and s_{k+1} are the two segments connected by an edge $e_{k,k+1}$. We also introduced two additional edge features defined as the mean and standard deviation of the vertex offsets of the segment boundaries, in which the offset is defined using the closest point pair between two segments.

Segment classification. Based on the graph and feature embedding (see Fig. 7), we exploit a GCN (Li et al., 2016) to classify the segments. The node features learned from PointNet (Qi et al., 2017a) and the handcrafted features are concatenated and fed to MLP (Multilayer perceptron) to output a 64D feature vector that serves as the hidden state of the Gated Recurrent Unit (GRU: a gating mechanism in recurrent neural networks for updating and resetting hidden states to capture short-term and long-term dependencies in sequence) (Cho et al., 2014). We apply ReLU activation (Nair and Hinton, 2010) and batch normalization (Ioffe and Szegedy, 2015) for each hidden layer of all MLPs. The computed edge features are used as the input to the

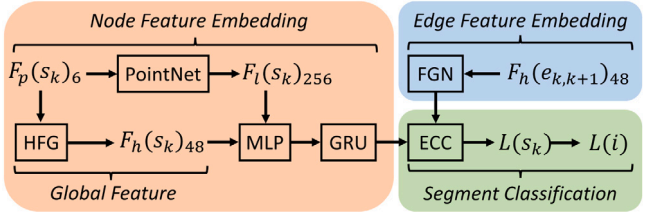


Fig. 7. The feature embedding components for segment classification. Our network takes mesh vertices and face centers (with XYZ and RGB) of each segment, denoted as $F_p(s_k)_6$, as input. PointNet (Qi et al., 2017a) is used to learn features $F_l(s_k)_{256}$ for each segment. The handcrafted features $F_h(s_k)_{48}$ are computed using the feature generator (HFG). These two types of features are then processed jointly by the MLP and then refined in the GRU. The $F_h(e_{k,k+1})_{48}$ are handcrafted edge features and input to a filter generating network (FGN). The final classification is obtained using edge-conditioned convolution (ECC) that takes both node and edge features as input. The output segment labels $L(s_k)$ are then transferred to face labels $L(i)$.

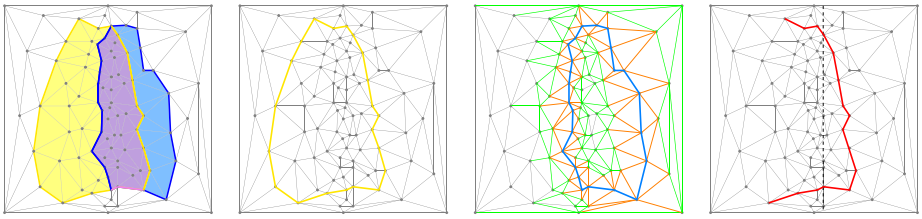


Fig. 8. 2D schematic of *object purity*, *boundary precision*, and *boundary recall*. In (a), the yellow region represents the ground truth segment G. The blue region represents the generated segment S. The pink region represents the largest overlapping region between a segment and the ground truth segments. In (b), the yellow edges represent the border B_G of the ground truth segment. In (c), the blue edges represent the border B_S of the generated segment. The orange edges represent the first ring of B_S. The green edges represent the second ring of B_S. In (d), the red edges are the intersection of B_G and B_S (as well as its first two rings). The black dashed line represents the true border between objects. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Filter Generating Network (FGN: a sequence of MLPs with widths of 32, 128, and 64 to output a 64D edge feature vector) (Landrieu and Simonovsky, 2018). The output edge weights are then used to update the hidden state and refine the GRUs via Edge-Conditioned Convolution (ECC: a dynamic edge-conditioned filter that computes element-wise vector–vector multiplication for each edge and averages the results over respective nodes) (Simonovsky and Komodakis, 2017). To alleviate class imbalance, we apply a standard cross-entropy loss (Szegedy et al., 2016) $l_n = -w_{y_n} \log \frac{\exp(x_{n,y_n})}{\sum_{c=1}^{C} \exp(x_{n,c})}$ weighted by $w_{y_n} = \sqrt{N/n_c}$, where x is the input, and y is the target. N denotes the total number of segments, and n_c represents the number of segments in each class c. The final output is per-segment labels that are then transferred to the faces of the input mesh.

4. Evaluation

4.1. Data split

We have implemented our mesh over-segmentation with CGAL (The CGAL Project, 2020) and Easy3D (Nan, 2021), and the semantic classification with PyTorch (Paszke et al., 2019). All experiments were carried out on a desktop PC with a 3.5 GHz CPU and a GTX 1080Ti GPU.

We have used the SUM dataset (Gao et al., 2021) and H3D dataset (Kölle et al., 2021) to evaluate our method. To the best of our knowledge, SUM is the largest benchmark dataset for semantic urban meshes, which covers about 4 km^2 of Helsinki (Finland) with six object classes: *terrain* (*terra.*), *high vegetation* (*h-veg.*), *building* (*build.*), *water*, *vehicle* (*vehic.*), and *boat*. The whole dataset contains 64 tiles each covering an 250 m × 250 m area. Following the SUM baseline, we used 40 tiles (62.5% of the whole dataset) for training, 12 tiles (18.75%) for the test, and 12 tiles for validation. The H3D dataset covers about 0.19 km^2 area of the village of Hessigheim (Germany) with 11 classes: *Low Vegetation*, *Impervious Surface*, *Vehicle*, *Urban Furniture*, *Roof*, *Facade*, *Shrub*, *Tree*, *Soil/Gravel*, *Vertical Surface*, and *Chimney*. We follow the data splits in H3D (Kölle et al., 2021), and we further merge small mesh tiles into a large one to obtain more contextual information.

4.2. Evaluation metrics

Metrics for over-segmentation. Our over-segmentation aims to produce homogeneous segments to better facilitate semantic segmentation. We propose three novel evaluation metrics focusing on the impact of the over-segmentation on the final semantic segmentation: *object purity (OP)*, *boundary precision (BP)*, and *boundary recall (BR)*.

Since our goal is semantic segmentation, the best achievable over-segmentation is identical to the ground truth semantic segmentation. In this ideal situation, each segment covers exactly an individual object, and its boundaries perfectly align with the object boundaries. Thus, similar to *intersection over union*, we define *object purity* as

$$OP(S,G) = \frac{\sum_k purity(s_k, G)}{area(G)}, \tag{6}$$

where $S = \{s_k\}$ denotes the set of segments in our over-segmentation, and $G = \{g_k\}$ are the segments extracted as connected components from the ground truth semantic segmentation. $purity(s_k, G)$ measures the surface area of the largest overlapping region between a segment and the ground truth segments (see Fig. 8).

Boundary precision measures the correctness of the segment boundaries. Thus, it is defined to quantify how much the segment boundaries overlap with the boundaries of the ground truth semantic segmentation,

$$BP(B_S, B_G) = \frac{length(B_S \cap B_G)}{length(B_S)}, \tag{7}$$

where B_S and B_G denote the boundaries of the over-segmentation and those of the ground truth of the semantic segmentation, respectively. The function $length(\cdot)$ quantifies the total length of a set of segment boundaries. To handle noisy and dense meshes, we allow a tolerance when looking for overlapping boundary edges. Specifically, two edges e_1 and e_2 are considered overlapping if the two endpoints of e_2 fall within the 2-ring neighborhood of the endpoints of e_1 (see Fig. 8).

Boundary recall measures the completeness of the segment boundaries, defined as

$$BR(B_S, B_G) = \frac{length(B_S \cap B_G)}{length(B_G)}. \tag{8}$$

Metrics for semantic segmentation. To evaluate semantic segmentation results, we measure the precision, recall, F1 score, and intersection over union (IoU) for each object class, and we also record the overall accuracy (OA), mean accuracy (mAcc), and mean intersection over union (mIoU) of all object classes.

4.3. Evaluation of over-segmentation

We evaluate over-segmentation on SUM dataset (Gao et al., 2021) because it covers a larger area and contains fewer unlabeled areas compared to H3D dataset (Kölle et al., 2021). Fig. 9 presents our planarity-sensible over-segmentation result and comparison with seven other commonly used over-segmentation techniques, namely region growing (RG) (Lafarge and Mallet, 2012), efficient RANSAC (RA) (Schnabel et al., 2007), geometric partition (GP) (Landrieu and Simonovsky, 2018), supervized superpoint generation (SSP) (Landrieu and Boussaha, 2019), variational shape approximation (VSA) (Cohen-Steiner et al., 2004), supervoxel generation (SPV) (Lin et al., 2018), voxel cloud connectivity segmentation (Vccs) (Papon et al., 2013), superface clustering (SC) (Verdie et al., 2015), and superface partitioning (SP) (Rouhani et al., 2017). RG, VSA, SC, SP, and our method use meshes as input, and the other methods (originally developed for point clouds) perform over-segmentation on points that we densely sampled (10 pts/m^2) from the input mesh. We can see from Fig. 9 that the segment boundaries of our method are largely aligned with object boundaries. RG and VSA perform similarly but generate excessive segments for non-planar objects such as trees. Our over-segmentation generates segments that

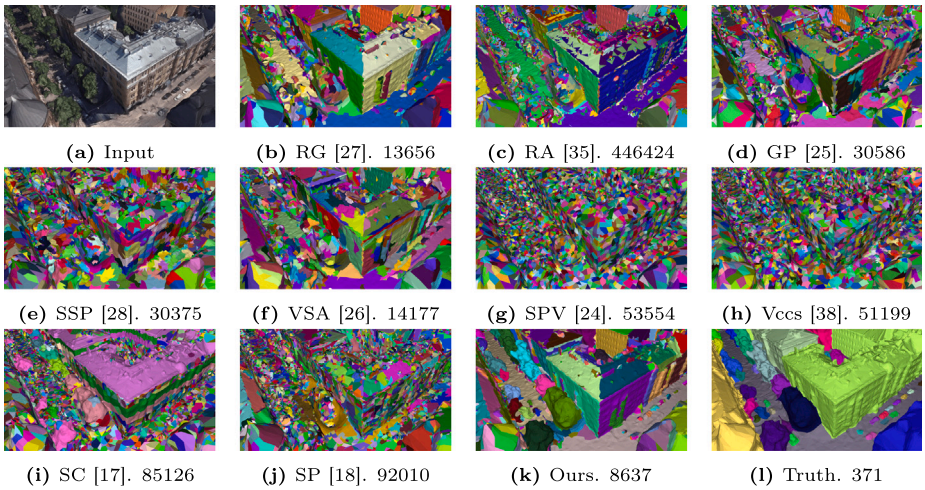


Fig. 9. Comparison of mesh over-segmentation methods on a tile of the SUM dataset (Gao et al., 2021). (b) to (k) show the over-segmentation results with the same *object purity* (around 92%) for all methods. The number below each result denotes the number of segments required to achieve the desired *object purity*. (l) shows the connected components extracted from the ground truth semantic segmentation.

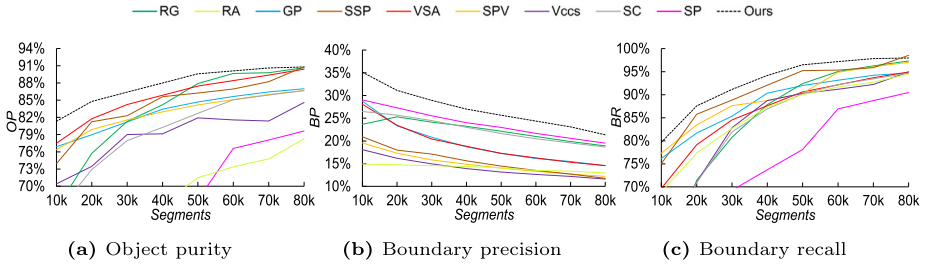


Fig. 10. Comparison of different over-segmentation method in terms of *OP*, *BP*, and *BR* on the SUM dataset (Gao et al., 2021). For each method, we tuned their parameters such that all methods generated a similar number of segments. The data was recorded at different numbers of segments for all methods.

are closer to semantically meaningful objects. In terms of the number of segments, RA, SC, SP, Vccs, and SPV generate relatively large numbers of segments, which are unfavorable to the subsequent classification using GCN. In contrast, our method generates the least segments, which is a strong advantage for the subsequent classification step in terms of efficiency.

We have used the entire SUM dataset (Gao et al., 2021) to evaluate our over-segmentation method in terms *OP*, *BP*, and *BR*, and we have compared them with those of the other seven over-segmentation methods. In the comparison, we tuned the parameters of each method such that all methods generated a similar number of segments, and we then computed the *OP*, *BP*, and *BR* for each method. We recorded the performance of all methods for different numbers of segments, and the results are shown in Fig. 10. It can be observed that our method outperforms the others for all three metrics. Specifically, as the number of segments increases, the *OP* of VSA, RG, and SSP get closer to ours. However, VSA underperforms our method in terms of *BP* and *BR*, which indicates that our method generated segments with better boundary qualities. For RG and SSP, its *OP*, *BP*, and *BR* are rather low when the number of segments is small, indicating that our method is more robust with a relatively small number of segments. Other methods like RA, SP, GP, SPV, Vccs, and SC also require a larger number of segments to produce satisfactory results.

To understand the potential of each method, Table 2 provides the maximum achievable performance of semantic segmentation for each over-segmentation method. The maximum achievable performance is measured by the maximum IoU and mIoU that can be achieved in theory. We can see that our method significantly outperforms the other methods with a considerable margin ranging from 4.8% to 30.1% in terms of mIoU (which reflects the under-segmentation errors). It is also worth noting that VSA has slightly better results on very few object classes (e.g., *water* and *boat*) while our method can better distinguish small non-planar objects such as vehicles which are very common in urban textured meshes.

Performance analysis. For the impact of planarity-sensible over-segmentation in different configurations, we experimented with different settings for each weight term based on the default parameters (i.e., $\lambda_d = 1.2$, $\lambda_m = 0.1$, and $\lambda_g = 0.9$). In each experiment, only one weight is tuned while the others remain unchanged. Fig. 11 shows its results in terms of *OP* and the number of *segments*. We can observe that increasing λ_d leads to a larger *OP* but also encourages the splitting of segments into smaller planar ones. In contrast, increasing λ_m results in over-smoothed segments (i.e., the smaller segments are merged into a larger one). However, λ_g performs differently from the other two since our aim is to use λ_g to reduce the number of *segments* without decreasing *OP* (as demonstrated in Fig. 11). The performance of applying λ_g is limited to the results of *planar* and *non-planar* classification, while the performance of the other two terms is limited to the quality of the mesh.

4.4. Evaluation of semantic classification

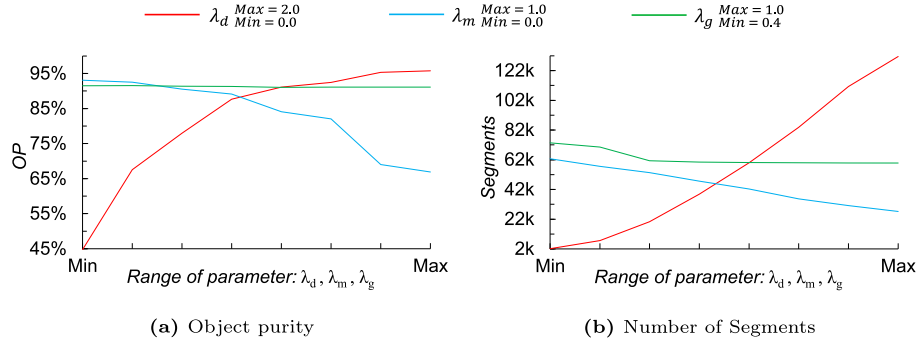
We have tested our semantic classification method on both the SUM (Gao et al., 2021) and the H3D (Kölle et al., 2021) datasets.

4.4.1. Evaluation on SUM

Fig. 12 shows the results for two tiles from the SUM dataset. From the extensive experiments, we also observed that our method is robust against non-uniform triangulation of mesh, for which an example is

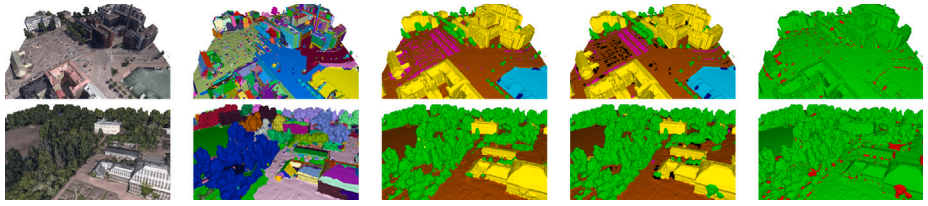
Table 2
Comparison of different over-segmentation methods in terms of maximum achievable performance of semantic segmentation on test data from the SUM dataset (Gao et al., 2021), with 50,000 segments. Evaluation metrics are reported as per-class IoU (%) and mean IoU (mIoU, %).

Methods	Terra.	H-veg.	Build.	Water	Vehic.	Boat	mIoU
SP (Rouhani et al., 2017)	73.4	88.1	96.8	12.5	15.7	68.1	59.1
RANSAC (Schnabel et al., 2007)	82.3	88.6	97.0	57.3	29.8	69.7	70.8
Vccs (Papon et al., 2013)	77.4	86.3	94.3	87.4	35.9	84.5	77.6
SC (Verdie et al., 2015)	86.8	92.5	97.8	75.5	46.1	79.3	79.7
SPV (Lin et al., 2018)	83.7	91.5	97.0	87.1	37.5	84.5	80.2
GP (Landrieu and Simonovsky, 2018)	86.5	91.2	96.6	87.1	46.9	84.6	82.1
RG (Lafarge and Mallet, 2012)	90.9	93.9	98.4	84.5	53.9	75.6	82.9
SSP (Landrieu and Boussaha, 2019)	85.3	91.2	96.1	91.1	49.6	88.3	83.6
VSA (Cohen-Steiner et al., 2004)	91.1	95.1	98.7	**93.4**	39.3	**88.9**	84.4
Ours	**93.3**	**95.6**	**98.9**	91.6	**67.1**	88.8	**89.2**



(a) Object purity (b) Number of Segments

Fig. 11. Comparison of over-segmentation with different parameter configurations in terms of object purity and the number of segments on the SUM dataset (Gao et al., 2021). Note that the range of parameters depends on the quality of the input data.



(a) Input (b) Segments (c) Semantic (d) Truth (e) Error (red)

Fig. 12. Semantic segmentation results of our method on two tiles from the SUM dataset (Gao et al., 2021).

Table 3
Comparison with state-of-the-art semantic segmentation methods on the SUM dataset (Gao et al., 2021). Per-class IoU (%), mean IoU (mIoU, %), Overall Accuracy (OA, %), mean class Accuracy (mAcc, %), mean F1 score (mF1, %), and the running times for training (over-seg: 1.8 h, graph: 2.7 h, classification: 11.8 h, total: 16.3 h) and testing (over-seg: 15 min, graph: 40 min, classification: 7 min, total: 62 min) are included. Note that each method was run ten times and the mean performance is reported here.

Methods	Terra.	H-veg.	Build.	Water	Vehic.	Boat	mIoU	OA	mAcc	mF1	Training (h)	Testing (min)
PointNet (Qi et al., 2017a)	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	**1**
RandLaNet (Hu et al., 2020)	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	52
SPG (Landrieu and Simonovsky, 2018)	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	26
PointNet++ (Qi et al., 2017b)	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	3
RF-MRF (Rouhani et al., 2017)	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**	15
SUM-RF (Gao et al., 2021)	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	18
KPConv (Thomas et al., 2019)	**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	42
Ours	84.9	**90.6**	**93.9**	84.3	50.9	**32.3**	**72.8** ± 2.0	**93.8** ± 0.4	**79.2** ± 3.0	**81.6** ± 2.3	16.3	62

shown in Fig. 15. We have also compared our method with several state-of-the-art semantic segmentation approaches, among which RF-MRF (Rouhani et al., 2017) and SUM-RF (Gao et al., 2021) directly consume meshes. To compare with methods originally developed for semantic segmentation of point clouds, e.g., PointNet (Qi et al., 2017a), PointNet++ (Qi et al., 2017b), SPG (Landrieu and Simonovsky, 2018), KPConv (Thomas et al., 2019), and RandLA-Net (Hu et al., 2020), we sampled points from the meshes by following Gao et al. (2021). For each deep learning method designed for point clouds, we feed it with the colored point clouds densely sampled from the texture meshes. We tune the hyper-parameters starting with their default setting. For a fair comparison, we use the same weight for all the competing methods involved in the comparison. Due to the randomness of deep learning, we ran each method ten times with the same settings to record its average performance. We report the results in Table 3, and we can see the results of the top three best methods in Fig. 13. Our method achieves the highest per-class IoU on the majority of object classes, and it outperforms all other methods in all overall metrics, with a margin from 4% to 35.9% in terms of mIoU. Compared to KPConv and SPG, our method requires less training time, and our results are more stable (i.e., with smaller standard deviations).

4.4.2. Evaluation on H3D

We also attempted to train and test on the H3D dataset (Kölle et al., 2021) (see for an overview of the results in Fig. 14). It should be noted that the H3D dataset is much smaller (0.19 km^2) than SUM (4 km^2).

Fig. 13. Semantic segmentation results of SUM-RF (Gao et al., 2021), KPConv (Thomas et al., 2019), and our method on five tiles from the SUM dataset (Gao et al., 2021).

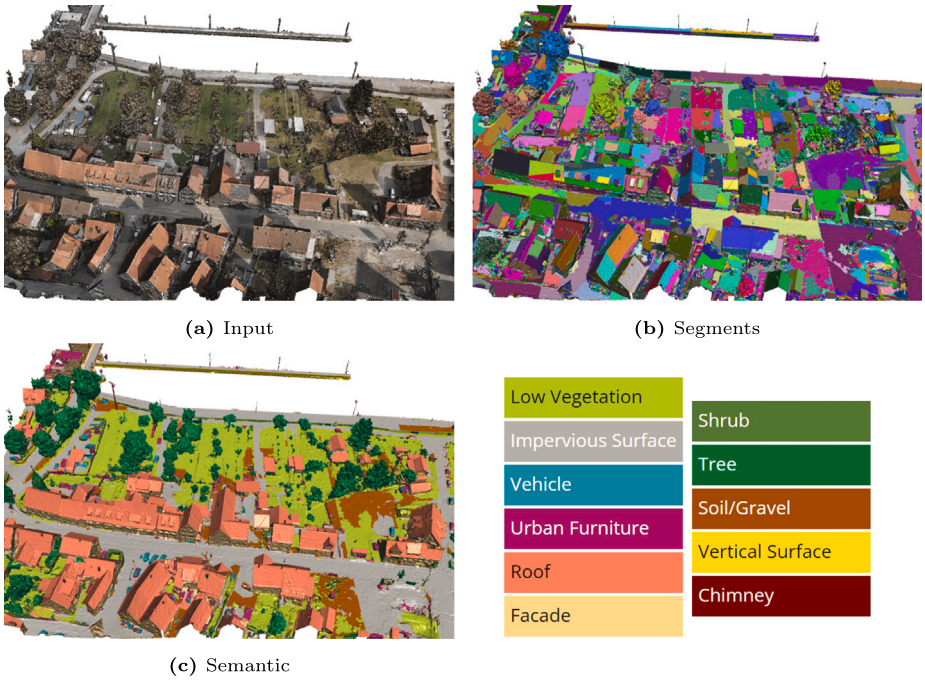
In addition, 40% of the area of the mesh in H3D is unlabeled and many triangles have incorrect labels, which was due to the limitation in their cloud-to-mesh labeling process where the correspondence was either not completed or has ambiguities when transferring the labels from the points to the mesh faces. This prevents H3D from being an ideal training dataset for us, as both our method and other deep learning methods require a large amount of labeled data. Besides, to distinguish between different classes that are geometrically coplanar in H3D (e.g., Low Vegetation, Impervious Surface, Soil, and Gravel), the *planar* class is divided into sub-classes and used as a prior for over-segmentation in our approach. The test results show that our method achieves about 52.2% mIoU, outperforming the KPConv (45.5% mIoU), SPG (29.9% mIoU), and PointNet++ (15.6% mIoU).

4.5. Ablation study

To understand the effect of several design choices made in the graph construction, and the contributions of the hand-crafted features and the learned features, we have conducted an ablation study on the SUM dataset (Gao et al., 2021).

Table 4 summarizes the ablation result of the graphs and the features. The upper part of Table 4 reveals the impact of removing each type of graph edges (i.e., connections between segments) on the final semantic segmentation. It reveals that every type of graph edges contributes to the performance, and removing any of them results in a drop in mIoU in the range $[2.9\%, 4.3\%]$. This implies that both local and global interactions between segments provide useful information and play an important role in semantic segmentation. To understand the effectiveness of the designed graph, we compared it to using a graph with random connections (with the same number of edges as in our designed graph). Although these connections may overlap with the edges we have designed, the presence of randomness significantly reduces the capability of the network. This is because the random set of edges does not convey the effective features captured by our carefully designed edges. Besides, the orthogonal edges were also tested (see Table 4), from which we can see that they are less useful than the other types of edges. This is because the orthogonal relationship is more often incident to man-made structures within the same object (such as building parts) than between segments from different objects.

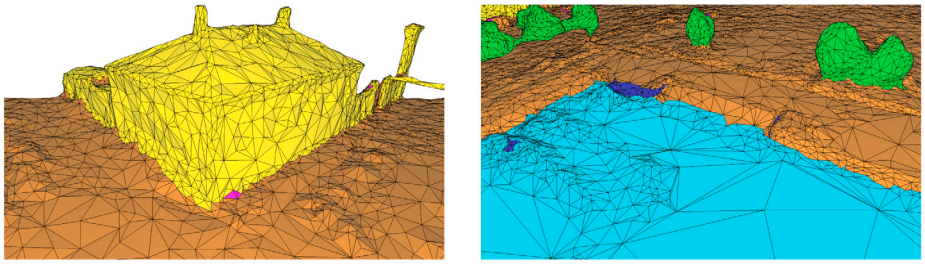
The lower part of Table 4 details the ablation analysis of different features. We have evaluated the importance of each feature by removing it from the experiment and recording the performance of the semantic segmentation. These experiments show that the combination of all the features outperforms all degraded features with a margin



(a) Input (b) Segments

(c) Semantic

Fig. 14. Semantic segmentation results of our method on the test area of the H3D dataset (Kölle et al., 2021).



(a) Near-uniform triangulation (b) Non-uniform triangulation

Fig. 15. Robustness against triangulation.

Table 4
Ablation study of graph edges and features on the SUM dataset (Gao et al., 2021). *PointNet* denotes features learned from PointNet (Qi et al., 2017a) with XYZ and RGB as input, and " No *RGB with PointNet*" means RGB is not used as input.

	Model	OA (%)	mAcc (%)	mIoU (%)	ΔmIoU (%)
Graph edges	No *parallelism*	92.9	75.3	68.5 ± 1.9	−4.3
	No *ExMAT*	93.1	76.0	69.3 ± 1.3	−3.5
	No *spatial-proximity*	93.1	76.3	69.4 ± 1.9	−3.4
	No *connecting-ground*	93.3	76.3	69.9 ± 1.6	−2.9
	Only *random*	92.5	74.9	67.8 ± 2.3	−5.0
	With *orthogonal*	93.0	75.3	68.9 ± 1.8	−3.9
Features	No *All Handcrafted*	89.7	75.9	65.0 ± 3.8	−7.8
	No *Offset*	92.4	75.0	67.2 ± 1.5	−5.6
	No *PointNet*	92.8	74.5	68.2 ± 1.3	−4.6
	No *Eigen*	93.0	75.3	68.3 ± 2.3	−4.5
	No *Color*	93.1	75.0	68.9 ± 1.4	−3.9
	No *Density*	93.1	76.4	69.4 ± 2.0	−3.4
	No *Scale*	93.0	76.1	69.7 ± 0.9	−3.1
	No *Shape*	93.2	75.6	69.7 ± 0.8	−3.1
	No *RGB with PointNet*	93.5	76.4	70.4 ± 0.4	−2.4
	Ours	**93.8**	**79.2**	**72.8** ± 2.0	–

of mIoU from 2.4% to 7.8%, which means every feature contributes to the performance. It is also interesting to notice that the results are more stable without RGB information as input to PointNet for feature learning, which indicates the low quality (e.g., distortion, shadow) of mesh textures in our training datasets.

4.6. Generalization ability

We have conducted experiments to test the generalization ability of our method and the competing methods. Such tests can indicate the applicability of models trained by different methods on practical test datasets. SUM (Gao et al., 2021) and H3D (Kölle et al., 2021) are well

Table 5
Generalization ability comparison. All methods are trained on four classes of the SUM dataset (Gao et al., 2021). The top eight rows show the scores on the testing area of the SUM dataset, while the bottom eight records show the testing results on the four classes H3D dataset. Per-class IoU (%) and mean IoU (mIoU, %) are reported here.

	Methods	Terra.	H-veg.	Build.	Vehic.	mIoU
SUM	PointNet (Qi et al., 2017a)	66.8	13.8	65.7	0.0	36.6
	PointNet++ (Qi et al., 2017b)	77.7	76.7	86.3	1.3	60.5
	SPG (Landrieu and Simonovsky, 2018)	86.0	73.9	88.5	13.9	65.6
	RF-MRF (Rouhani et al., 2017)	86.8	86.7	90.5	20.7	71.2
	RandLaNet (Hu et al., 2020)	83.0	91.6	90.1	22.0	71.7
	KPConv (Thomas et al., 2019)	89.4	84.7	91.5	**34.1**	75.0
	SUM-RF (Gao et al., 2021)	88.0	90.2	92.3	30.4	75.2
	Ours	**89.5**	**92.0**	**93.9**	33.0	**77.1**
H3D	PointNet++ (Qi et al., 2017b)	0.0	0.0	0.0	1.1	0.3
	KPConv (Thomas et al., 2019)	0.0	0.0	0.0	1.1	0.3
	SPG (Landrieu and Simonovsky, 2018)	0.0	0.0	18.3	0.0	4.6
	RandLaNet (Hu et al., 2020)	0.0	0.0	20.7	0.0	5.2
	PointNet (Qi et al., 2017a)	56.9	24.1	0.0	0.0	20.3
	RF-MRF (Rouhani et al., 2017)	73.9	46.0	40.0	4.7	41.1
	SUM-RF (Gao et al., 2021)	**80.2**	44.0	41.6	9.2	43.8
	Ours	74.2	**66.2**	**44.5**	**13.4**	**49.6**

qualified for generalization ability tests as they both represent urban scenes. As the original classes of these two datasets do not match, we merged them into four common classes that are typical of urban scenarios for testing, i.e., terrain (including water, low vegetation, impervious surface, soil, and gravel), high-vegetation (including shrub and tree), building (including roof, facade, and chimney), and vehicle (including car and boat). For all methods, we trained the model on the SUM dataset and perform the testing on the H3D dataset. In particular, for training and validation, we have used the training and validation splits of the SUM dataset as they cover a larger area than H3D. For testing, we have used training and validation splits of the H3D dataset that has publicly available ground truth labels. To compare the difference in results, we also tested the same model on the test area of SUM.

As shown in the top eight rows of Table 5, the mIoU of all methods has improved compared to the previous six classes in the SUM test area because the task of classification has become relatively easy due to the reduction in the number of classes. The bottom eight records of Table 5 demonstrate the generalization ability of different methods. We can see that our method outperforms all competing methods with a margin from 5.8% to 49.3% in terms of mIoU. Except for our approach, almost all other deep learning-based methods (i.e. PointNet++ (Qi et al., 2017b), SPG (Landrieu and Simonovsky, 2018), KPConv (Thomas et al., 2019), and RandLA-Net (Hu et al., 2020)) failed to predict the classes in a new urban scene. This is because these methods all learn global features by having a large receptive field, and these global features can lead to overfitting of the model to the training data and result in degradation of generalization ability. In contrast, the global features of PointNet (Qi et al., 2017a) are based on aggregated local features and do not correspond to a larger receptive field, which does not have the overfitting problem. In other words, training with only local features avoids the degradation of model generalization ability, which is better illustrated by RF-MRF (Rouhani et al., 2017) and SUM-RF (Gao et al., 2021) as they only use features extracted on local segments. Whereas our approach includes global features derived from the local features, our proposed graph is based on the spatial distribution of the object components in the urban scene, which facilitates the generalization of the global features.

From Tables 4 and 5, we can conclude that compared with features learned by the neural network, the proposed handcrafted features and edges lead to better semantic segmentation results and contribute to a stronger generalization ability, especially for data with domain gaps. In particular, the domain gaps are attributed to the differences between training data and test data (i.e., different feature distributions), and the reasons for these differences can be grouped into two main categories: (1) same data acquisition and processing pipeline, but covering different urban scenarios (e.g., from dense urban area to rural or forest area); (2) covering the same urban scenarios but with different data acquisition (e.g., using different sensors or different parameters for data collection) and processing pipelines (e.g., using different approaches or parameter configurations for generating the 3D data). Nevertheless, the features learned by the existing neural network architectures cannot cope with such differences without adding new training samples from the test area. Our segment-based handcrafted features and edges capture the intrinsic characteristics of the object (e.g., the facade is usually perpendicular to the ground or the surface of the tree is undulating and non-planar) and can better cope with the variance in feature distribution.

4.7. Limitations

Our method is based on the observation that urban scenes consist of objects demonstrating both planar and non-planar regions, and that object boundaries lie in the connections between the planar and non-planar regions. In special cases where adjacent objects contain only non-planar regions (e.g., vehicles underneath trees), our method will not be able to differentiate them. In addition, our approach generates segments for semantic classification, which reduces memory consumption as well as the number of samples. Specifically, if the training data covers a small area (e.g., H3D (Kölle et al., 2021)) and only a few segments are generated after over-segmentation, the number of samples may not be sufficient to train a competent model. Possible solutions include data augmentation of segments and graph connections or adding more labeled data. Besides, our method requires adjacency information of the mesh for incremental region growing. Additional preprocessing is necessary for meshes that are non-manifold or contain duplicated vertices, as they destroy the topological adjacency information. Potential solutions can be to split non-manifold vertices or to reconstruct adjacency information of duplicated vertices. In our experiments, the meshes in the SUM dataset (Gao et al., 2021) are 2-manifold, while the meshes in the H3D dataset (Kölle et al., 2021) are not.

5. Conclusion

We have presented a two-stage supervised framework for semantic segmentation of large-scale urban meshes. Our planarity-sensible over-segmentation algorithm favors generating segments largely aligned with object boundaries, closer to semantically meaningful objects, can deliver descriptive features, and can represent urban scenes with a smaller number of segments. A thorough analysis reveals that our planarity-sensible over-segmentation plays a key role in achieving superior performance in semantic segmentation. We have also shown that

exploiting multi-scale contextual information better facilitates semantic segmentation. Furthermore, we have demonstrated that our proposed approach achieves better generalization abilities in comparison with other methods, owing to the segment-based local features and unique connections in graphs. Our proposed new metrics are effective for evaluating mesh over-segmentation methods dedicated to semantic segmentation. We believe the proposed metrics will further stimulate improving other over-segmentation techniques. In future work, we would like to extend our framework to part-level (e.g., dormers, balconies, roofs, and facades of buildings) urban mesh segmentation. In addition, we will also investigate how the semantics learned from multi-view images can be used for semantic segmentation of urban meshes.

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.

References

Ben-Shabat, Y., Avraham, T., Lindenbaum, M., Fischer, A., 2018. Graph based over-segmentation methods for 3D point clouds. Comput. Vis. Image Underst. 174, 12–23.

Besuievsky, G., Beckers, B., Patow, G., 2018. Skyline-based geometric simplification for urban solar analysis. Graph. Models 95, 42–50.

Biljecki, F., Stoter, J., Ledoux, H., Zlatanova, S., Çöltekin, A., 2015. Applications of 3D city models: State of the art review. ISPRS Int. J. Geo-Inf. 4 (4), 2842–2889.

Boykov, Y., Veksler, O., Zabih, R., 2001. Fast approximate energy minimization via graph cuts. IEEE Trans. Pattern Anal. Mach. Intell. 23 (11), 1222–1239. http://dx.doi.org/10.1109/34.969114.

Cho, K., van Merrienboer, B., Gulcehre, C., Bougares, F., Schwenk, H., Bengio, Y., 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: Conference on Empirical Methods in Natural Language Processing (EMNLP 2014).

City of Helsinki, 2019. Helsinki's 3D city models. https://www.hel.fi/helsinki/en/administration/information/general/3d, Accessed: 2020-11-25.

Cohen-Steiner, D., Alliez, P., Desbrun, M., 2004. Variational shape approximation. In: ACM SIGGRAPH 2004 Papers. pp. 905–914.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B., 2016. The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3213–3223.

Demantké, J., Mallet, C., David, N., Vallet, B., 2011. Dimensionality based scale selection in 3D LiDAR point clouds.

Fu, H., Jia, R., Gao, L., Gong, M., Zhao, B., Maybank, S., Tao, D., 2021. 3D-future: 3d furniture shape with texture. Int. J. Comput. Vis. 1–25.

Gao, W., Nan, L., Boom, B., Ledoux, H., 2021. SUM: A benchmark dataset of semantic urban meshes. ISPRS J. Photogramm. Remote Sens. 179, 108–120. http://dx.doi.org/10.1016/j.isprsjprs.2021.07.008.

Gao, L., Yang, J., Wu, T., Yuan, Y.-J., Fu, H., Lai, Y.-K., Zhang, H., 2019. SDM-NET: Deep generative network for structured deformable mesh. ACM Trans. Graph. 38 (6), 1–15.

Geurts, P., Ernst, D., Wehenkel, L., 2006. Extremely randomized trees. Mach. Learn. 63 (1), 3–42.

Google, 2012. 3D imagery in google earth. https://earth.google.com/web/, Accessed: 2021-01-16.

Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L., Bennamoun, M., 2020. Deep learning for 3d point clouds: A survey. IEEE Trans. Pattern Anal. Mach. Intell. 43 (12), 4338–4364.

Hackel, T., Wegner, J.D., Schindler, K., 2016. Fast semantic segmentation of 3D point clouds with strongly varying density. ISPRS Ann. Photogram. Remote Sens. Spatial Inf. Sci. 3, 177–184.

Hanocka, R., Hertz, A., Fish, N., Giryes, R., Fleishman, S., Cohen-Or, D., 2019. Meshcnn: a network with an edge. ACM Trans. Graph. 38 (4), 1–12.

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

Huang, J., Zhang, H., Yi, L., Funkhouser, T., Nießner, M., Guibas, L.J., 2019. TextureNet: Consistent local parametrizations for learning from high-resolution signals on meshes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4440–4449.

Hui, L., Yuan, J., Cheng, M., Xie, J., Zhang, X., Yang, J., 2021. Superpoint network for point cloud oversegmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. ICCV, pp. 5510–5519.

Ioffe, S., Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning. PMLR, pp. 448–456.

Jaromczyk, J., Toussaint, G., 1992. Relative neighborhood graphs and their relatives. Proc. IEEE 80 (9), 1502–1517.

Karl Pearson, F., 1901. LIII. On lines and planes of closest fit to systems of points in space. Philos. Mag. Ser. 1 2 (11), 559–572.

Kölle, M., Laupheimer, D., Schmohl, S., Haala, N., Rottensteiner, F., Wegner, J.D., Ledoux, H., 2021. 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 J. Photogram. Remote Sens. 1, 100001. http://dx.doi.org/10.1016/j.ophoto.2021.100001.

Lafarge, F., Mallet, C., 2012. Creating large-scale city models from 3D-point clouds: a robust approach with hybrid representation. Int. J. Comput. Vis. 99 (1), 69–85.

Landrieu, L., Boussaha, M., 2019. Point cloud oversegmentation with graph-structured deep metric learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7440–7449.

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

Lei, H., Akhtar, N., Mian, A., 2021. Picasso: A CUDA-based library for deep learning over 3D meshes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13854–13864.

Li, S., Luo, Z., Zhen, M., Yao, Y., Shen, T., Fang, T., Quan, L., 2019. Cross-atlas convolution for parameterization invariant learning on textured mesh surface. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6143–6152.

Li, Y., Tarlow, D., Brockschmidt, M., Zemel, R., 2016. Gated graph sequence neural networks. In: Proceedings of the International Conference on Learning Representations.

Lin, Y., Wang, C., Zhai, D., Li, W., Li, J., 2018. Toward better boundary preserved supervoxel segmentation for 3D point clouds. ISPRS J. Photogramm. Remote Sens. 143, 39–47.

Liu, M.-Y., Tuzel, O., Ramalingam, S., Chellappa, R., 2011. Entropy rate superpixel segmentation. In: CVPR 2011. IEEE, pp. 2097–2104.

Ma, J., Bae, S.W., Choi, S., 2012. 3D medial axis point approximation using nearest neighbors and the normal field. Vis. Comput. 28 (1), 7–19.

Melzer, T., 2007. Non-parametric segmentation of ALS point clouds using mean shift. J. Appl. Geodesy Jag 1 (3), 159–170.

Nair, V., Hinton, G.E., 2010. Rectified linear units improve restricted boltzmann machines. In: Icml.

Nan, L., 2021. Easy3D: a lightweight, easy-to-use, and efficient C++ library for processing and rendering 3D data. J. Open Source Softw. 6 (64), 3255. http://dx.doi.org/10.21105/joss.03255.

Nan, L., Xie, K., Sharf, A., 2012. A search-classify approach for cluttered indoor scene understanding. ACM Trans. Graph. 31 (6), 1–10.

Papon, J., Abramov, A., Schoeler, M., Worgotter, F., 2013. Voxel cloud connectivity segmentation-supervoxels for point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2027–2034.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S., 2019. Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché Buc, F., Fox, E., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 32. Curran Associates, Inc., pp. 8024–8035.

Peters, R., Ledoux, H., 2016. Robust approximation of the medial axis transform of LiDAR point clouds as a tool for visualisation. Comput. Geosci. 90, 123–133.

Qi, C.R., Su, H., Mo, K., Guibas, L.J., 2017a. PointNet: Deep learning on point sets for 3D classification and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 652–660.

Qi, C.R., Yi, L., Su, H., Guibas, L.J., 2017b. PointNet++: Deep hierarchical feature learning on point sets in a metric space. Adv. Neural Inf. Process. Syst. 30, 5099–5108.

Rouhani, M., Lafarge, F., Alliez, P., 2017. Semantic segmentation of 3D textured meshes for urban scene analysis. ISPRS J. Photogramm. Remote Sens. 123, 124–139.

Saran, S., Wate, P., Srivastav, S., Krishna Murthy, Y., 2015. CityGML at semantic level for urban energy conservation strategies. Ann. GIS 21 (1), 27–41.

Schnabel, R., Wahl, R., Klein, R., 2007. Efficient RANSAC for point-cloud shape detection. In: Computer Graphics Forum. Vol. 26, (2), Wiley Online Library, pp. 214–226.

Schult, J., Engelmann, F., Kontogianni, T., Leibe, B., 2020. DualConvMesh-net: Joint geodesic and euclidean convolutions on 3D meshes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8612–8622.

Selvaraju, P., Nabail, M., Loizou, M., Maslioukova, M., Averkiou, M., Andreou, A., Chaudhuri, S., Kalogerakis, E., 2021. BuildingNet: Learning to label 3D buildings. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10397–10407.

Simonovsky, M., Komodakis, N., 2017. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3693–3702.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z., 2016. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2818–2826.

The CGAL Project, 2020. CGAL User and Reference Manual, 5.1.1 CGAL Editorial Board.

Thomas, H., Goulette, F., Deschaud, J.-E., Marcotegui, B., 2018. Semantic classification of 3D point clouds with multiscale spherical neighborhoods. In: 2018 International Conference on 3D Vision (3DV). IEEE, pp. 390–398.

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

Verdie, Y., Lafarge, F., Alliez, P., 2015. LOD generation for urban scenes. ACM Trans. Graph. 34 (3), 1–14. http://dx.doi.org/10.1145/2732527.

Vosselman, G., Coenen, M., Rottensteiner, F., 2017. Contextual segment-based classification of airborne laser scanner data. ISPRS J. Photogramm. Remote Sens. 128, 354–371.

Vosselman, G., Gorte, B.G., Sithole, G., Rabbani, T., 2004. Recognising structure in laser scanner point clouds. Int. Arch. Photogram. Remote Sens. Spatial Inf. Sci. 46 (8), 33–38.

Weinmann, M., Jutzi, B., Mallet, C., 2013. Feature relevance assessment for the semantic interpretation of 3D point cloud data. ISPRS Ann. Photogram. Remote Sens. Spatial Inf. Sci. 5 (W2), 1.

Weinmann, M., Schmidt, A., Mallet, C., Hinz, S., Rottensteiner, F., Jutzi, B., 2015. Contextual classification of point cloud data by exploiting individual 3D neigbourhoods. ISPRS Ann. Photogram. Remote Sens. Spatial Inf. Sci. II-3 2 (W4), 271–278.

Wu, S.-C., Wald, J., Tateno, K., Navab, N., Tombari, F., 2021. SceneGraphFusion: Incremental 3D scene graph prediction from RGB-D sequences. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7515–7525.

Yang, M., Yu, K., Zhang, C., Li, Z., Yang, K., 2018. DenseASPP for semantic segmentation in street scenes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3684–3692.