Information Sciences 614 (2022) 464–479

Contents lists available at ScienceDirect

Information Sciences

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

Mesh Oversegmentation with Segmentation-Aware Loss

Jibril Muhammad Adam [a,b], Muhammad Kamran Afzal [a], Zang Yu [a,*], Saifullahi Aminu Bello [a], Cheng Wang [a], Jonathan Li [a,c]

[a] Spatial Sensing and Computing Lab, Xiamen University, Xiamen, FJ 361005, China
[b] Department of Computer Science, Federal University Dutse, Nigeria
[c] Department of Geography and Environmental Management, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada

ARTICLE INFO

Article history:
Received 1 January 2022
Received in revised form 18 August 2022
Accepted 6 October 2022
Available online 11 October 2022

Keywords:
Superfacet
3D mesh
Oversegmentation
Semantic segmentation
Dual-primal graphs
Graph attention networks

ABSTRACT

Superfacets are generated by clustering adjacent mesh faces that share similar characteristics, which can serve as processing units in downstream mesh applications. While there are existing deep neural networks that generate superpixels and superpoints/supervoxels from images and point clouds respectively, the current oversegmentation methods in 3D meshes mostly rely on hand-crafted features that are extracted using non-differentiable algorithms to generate superfacets. Nevertheless, these methods cannot leverage the feature extraction abilities of deep neural networks to generate superfacets in an end-to-end fashion. Therefore, we propose an end-to-end trainable deep neural network that learns to generate boundary-aware superfacets from 3D meshes. Specifically, our network learns a soft face-superfacet association map from faces and their adjacency relationships. Moreover, we develop a segmentation-aware loss on the faces that train the network to predict their labels in an end-to-end manner. We evaluate the performance of our method using mesh adaptations of two well-known superpixel evaluation metrics where experimental results demonstrate that the performance of our proposed network surpasses that of other state-of-the-art mesh oversegmentation methods, and in doing so simultaneously improves superfacet-based semantic segmentation.

1. Introduction

In computer vision, the process of segmenting sets of contiguous elements that share common features into tractable and perceptual units is known as oversegmentation. These units serve as perceptual and semantic entities that can be used in subsequent downstream vision processes. They simplify the number of elements considered for a task from hundreds of thousands to some cases millions, to a few manageable numbers of clusters thereby reducing both the time and/or memory cost(s) of subsequent processes. The concept of oversegmentation has been applied to the different data formats that are involved in vision research *i.e.* image, point cloud, and 3D meshes.

One of the earliest methods to use the concept of oversegmentation in the image domain is SLIC [35], where the generated units are called superpixels. Subsequently, many methods have used superpixels for image processing tasks such as semantic segmentation (SCN) [48], stereo matching and optical flow (SSN) [16], and object detection [39]. In point clouds, oversegmentation methods are mostly used in pre-processing steps to generate superpoints and/or supervoxels which are subsequently used in downstream applications like semantic segmentation (SPG) [19](SSP) [18](BPSS) [22]. In 3D meshes,

* Corresponding author.
E-mail addresses: cwang@xmu.edu.cn (M.K. Afzal), zangyu7@126.com (Z. Yu), cwang@xmu.edu.cn (C. Wang), junli@uwaterloo.ca (J. Li).

https://doi.org/10.1016/j.ins.2022.10.039
0020-0255/

superfacets generated from oversegmentation methods like Fast and Scalable Mesh Superfacets (FSS) [40] have been used to achieve the tasks of segmentation [38] and co-segmentation [14,4]. In this work, we are concerned with the generation of superfacets from 3D meshes.

Most of the existing mesh oversegmentation methods *e.g.* FSS [40,36] rely on non-differentiable methods like the *k*-means algorithm to generate handcrafted features based on which similar faces or vertices are grouped. However, the performance of these methods is hindered by the inability of their features to capture complex geometric properties of 3D mesh elements. Also, the non-differentiability of these methods makes it difficult to use them in conjunction with end-to-end trainable deep networks. Similarly, the non-differentiable region-growing [32,29] algorithms that are used to group faces into planarity-sensible segments in PSSNet [9] make it not end-to-end trainable. Recently proposed methods like Primal–Dual Mesh Convolutional Neural Networks (PD-MeshNet) [27] and LaplacianNet [34] used mesh processing techniques like edge collapse [10] and spectral clustering to cluster similar faces and vertices respectively. These clustering techniques are a means (mostly as pooling operations) to achieve the main goal of these methods *i.e.* classification and segmentation as opposed to the generation of qualitative superfacets in oversegmentation methods. Therefore, generating high-quality superfacets from 3D meshes via an end-to-end trainable method is needed (Figs. 3 and 4).

In this paper, we propose an end-to-end trainable deep network for the oversegmentation of 3D meshes. We develop a deep hierarchical face clustering network that learns a face-superfacet association mapping through oversegmentation-driven edge collapse operations. Our approach draws inspiration from edge collapse methods in the context of face clustering. Our motivation is that edge collapse tends to cluster similar faces of a mesh into homogeneous superfacets forming a face-superfacet association mapping. Following the spirit of PD-MeshNet [27], we construct primal and dual graphs that encode the face and edge features of a 3D mesh respectively. Alternately, attention-based [43] convolution operations are applied to the two graphs across the layers of the network thereby computing complex and contextual features that encode relevant neighborhood information. Based on learned attention scores, edges of the primal graph are collapsed, which translates to the clustering of faces with similar geometric features into homogeneous superfacets. By repeating these operations across the layers of the network, hierarchical, soft association maps between faces and their parent superfacets are learned. We then construct a segmentation-aware loss on the faces and superfacets to train the network in an end-to-end manner. By training the network, our method *i.e. Mesh Oversegmentation with SEgmentation-Aware Loss* (MO-SEAL) learns to cluster similar faces into high-quality superfacets. To the best of our knowledge, our network is the first superfacet generation method that is end-to-end trainable. Fig. 2 is an illustration of the proposed MO-SEAL network.

To evaluate the performance of our method, we adapt two well-known superpixel evaluation metrics [41] to superfacets, where we obtain state-of-the-art results. We also validate our method in superfacet semantic segmentation where we report superior performance. Compared to existing oversegmentation methods in meshes, our method has the following contributions:

- We propose an end-to-end learning-based method, MO-SEAL, for superfacet generation. To the best of our knowledge, this is the first purposefully-built learning-based superfacet algorithm;
- We present a segmentation-aware loss that guides the network to generate boundary-aware superfacets by reconstructing semantic affinities of face-superfacet associations;
- With our learned superfacets, experiments demonstrate significant improvement of state-of-the-art results of 3D mesh oversegmentation. Further experiments demonstrate improvement in superfacet-based semantic segmentation task.

The sections of this paper are organized as follows: Section 2 contains a review of oversegmentation-based methods in images, point cloud, and 3D meshes. In Section 3, the network's theoretical foundations are discussed while Section 4 presents its implementation details. Section 5 presents experiments conducted followed by an analysis and discussion of the results obtained. Section 6 finally concludes the paper while suggesting areas for future improvements.

2. Related work

Oversegmentation methods have been applied to 2D images, 3D point clouds, and 3D meshes. We focus here on methods that prioritize oversegmentation as a goal, have used at least one of the metrics in [41] to evaluate the process, and/or used a purposefully built loss function to train the method. Fig. 1 shows an overview of the history of oversegmentation methods in different data formats.

2.1. Superpixels/superpoints/supervoxels

The main categories of superpixel algorithms are cluster-based and graph-based. The latter represents pixels as vertices of a graph and based on similarity criteria; usually edge-weights, the graph is partitioned. Some of the most popular algorithms in this category include SLIC [35] and ERS [24]. Cluster-based algorithms adapt well-known clustering techniques to partition images into superpixels. SLIC [35] is a well-known superpixel algorithm that adapted *k*-means to cluster images. Subsequently, other works such as Manifold SLIC [25], SNIC [2], and LSC [21] emerged as its variants.

All the above-mentioned methods used handcrafted features to generate superpixels and consequently suffer from the inherent problems of not leveraging the feature extraction powers of deep neural networks and non-differentiability. SSN

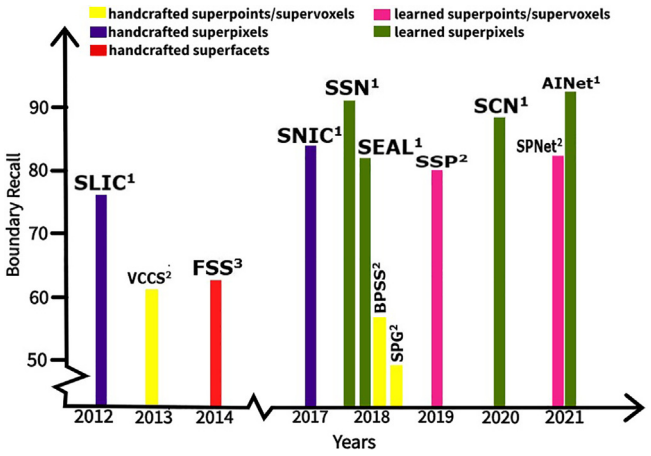


Fig. 1. The timeline for oversegmentation methods on different types of data. The X-axis indicates the years the methods were proposed and the Y-axis represents the boundary recall metric. See (1) From Fig. 4(a) in AINet [45], BP=0.11, $*10^{-2}$. (2) From Fig. 3(b) in SPNet [15], number of superpoints = 1,000. (3) Approximated value from FSS [40] for more information.

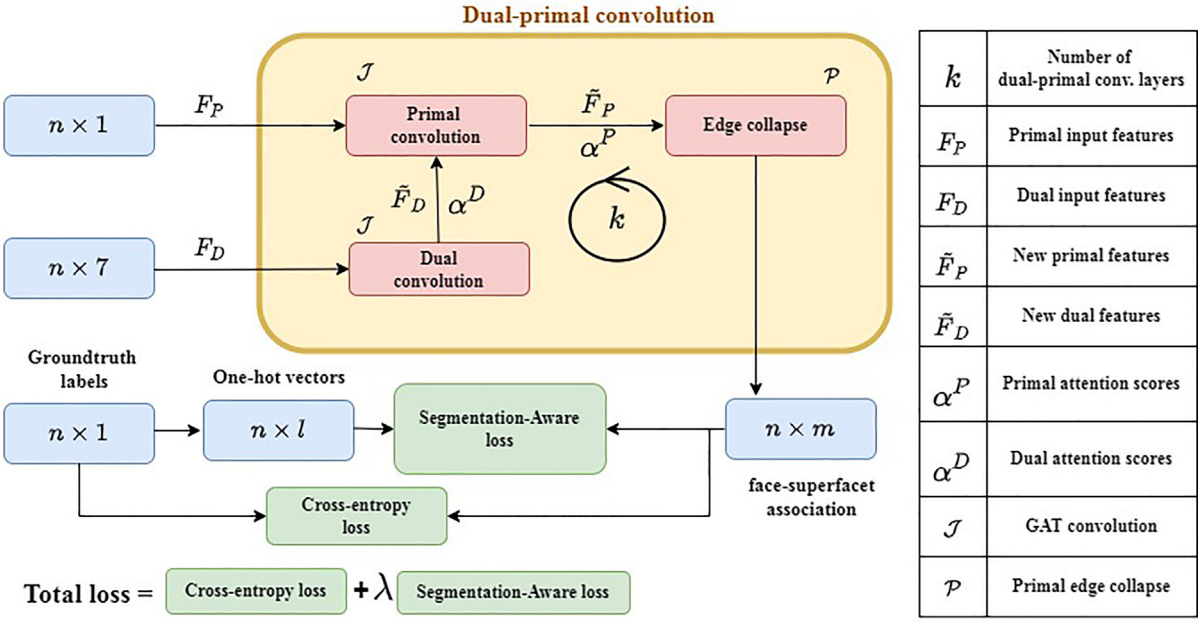


Fig. 2. Overview of the proposed oversegmentation network (MO-SEAL). **Dual-primal convolutions and edge collapse:** Dual and primal GAT convolutions are applied on dual and primal input features to generate new features on the dual and primal graphs' nodes. Edge collapse operation is then applied on the primal graph to generate face-superfacet association maps. These operations are repeated in k layers of the network. **Segmentation-aware loss:** The final face-superfacet mapping is used in segmentation-aware loss to reconstruct the one-hot vectors representing face labels.

[16] addressed the non-differentiability problem of SLIC [35] by softening the nearest neighbor computation in the k-means algorithm. They also formulated two task-specific reconstruction losses from pixel labels and optical flow maps to train the network to learn superpixels that adhere to semantic and optical flow boundaries respectively. The authors in SEAL [42] used neural networks to learn pixel affinities for superpixel segmentation. The SEAL method made use of segmentation errors to learn pixel affinities through a segmentation-aware loss. In SCN [48], a fully convolutional neural network is adopted to learn pixel-superpixel mapping for superpixel segmentation. The mappings are then passed on to a flexible loss function that uses either cross-entropy or l_2 norm as a distance measure to train the network. Inspired by the lack of context when inferring pixel-superpixel association due to limited receptive fields of convolution operations, AINet [45] utilized an association implantation module to adequately depict the mapping between a pixel and its neighbors. The method uses a boundary-perceiving loss to aid the network capture the context of the pixel-superpixel relationship.

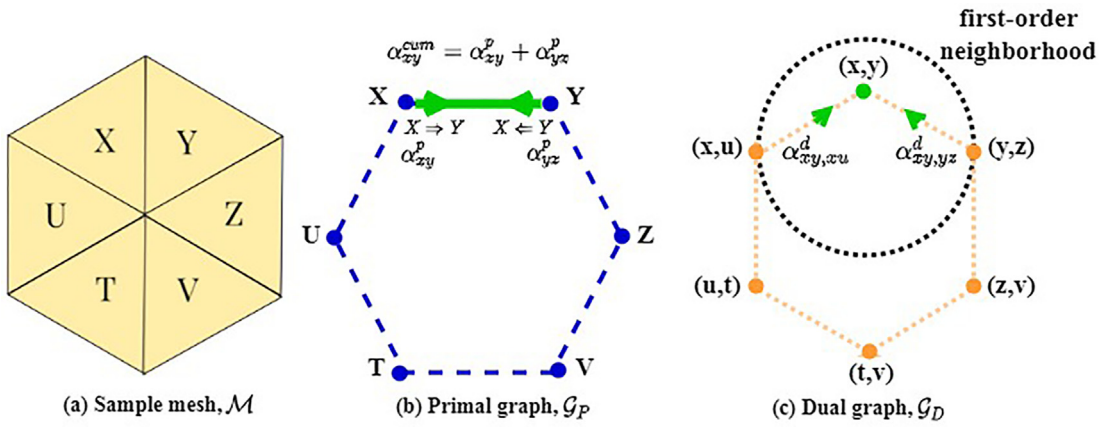


Fig. 3. The construction of primal, $\mathcal{G}_P$ and dual, $\mathcal{G}_D$ graphs from sample mesh, $\mathcal{M}$. (b) depicts the addition of attention scores, α_{xy}^{p} and α_{yx}^{p} associated with adjacent primal nodes, X and Y respectively. (c) shows the aggregation of first-order neighborhood features for dual node, (x, y).

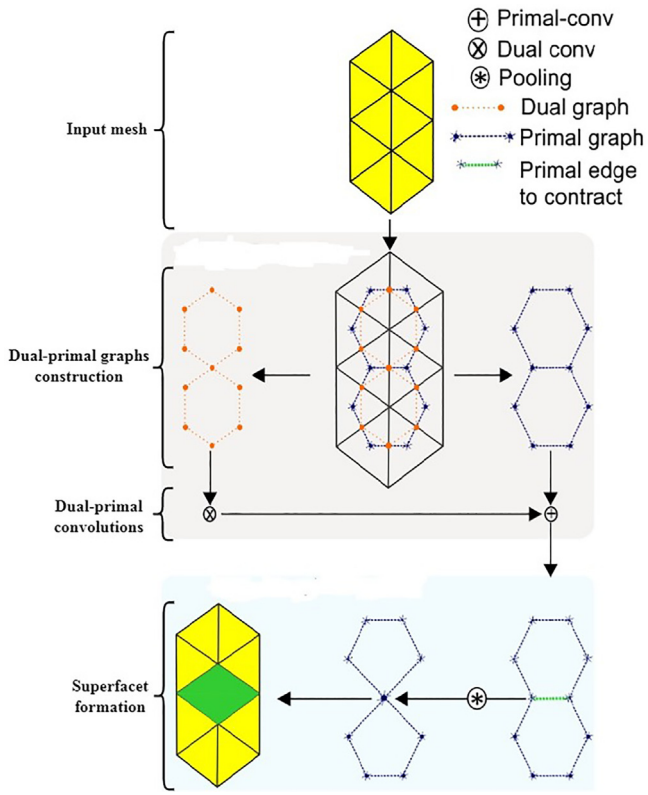


Fig. 4. Overview of the different operations in the MO-SEAL network. **Input mesh:** The input to the graph construction phase is a 3D mesh model. **Dual-primal graphs construction:** Dual and primal graphs (undirected) are constructed from the input mesh as illustrated in Fig. 3. The initial nodal features of the graphs are also constructed in this phase. **Dual-primal convolutions:** New nodal features and attention scores are learned from the graphs by applying GAT convolutions on them. **Superfacet formation:** Primal edges are collapsed based on the sum of their learnt attention scores to produce clusters of faces (superfacets).

Superpoints and/or supervoxels are the product of oversegmentation in point clouds. Initially, some of the early algorithms like VCCS [30] used *k*-means clustering to segment voxels generated from a point cloud. PCLV [3] adapted [5] to the 3D domain to oversegment point clouds. BPSS [22] proposed a method that does not require seed initialization but uses the local information of points, a dissimilarity criterion, and a user-defined number of clusters for supervoxel segmentation. Geom-Graph [12] addressed the task of urban scene classification by partitioning a graph that represents the input point

cloud. SPG [19] groups geometrically similar points into units called superpoints and subsequently constructs a graph, a superpoint graph, using the units as nodes. A graph convolutional network (GCN) is then trained to classify the nodes of the graph which translates to the task of semantic segmentation of the input point cloud.

All the aforementioned methods use handcrafted features in generating the superpoints/supervoxels and, as such, suffer from the same problems that affected the initial superpixel algorithms. SSP [18] is the first learning-based oversegmentation method in point clouds. The method uses a point embedding [33] network to compute point features which are reconstructed as a weighted graph. A graph-based contrastive loss is then used to train the network to identify boundary points as high contrast embeddings. Recently, SPNet [15] proposed a point cloud version of differentiable SLIC as an end-to-end superpoint generation method. It is an iterative clustering method that learns a point-superpoint association map from spatial coordinates and embedded features of a point cloud. The network is trained with the help of a label consistency loss to accurately assign points to their respective superpoints.

2.2. Superfacets

In FSS [40], superfacets are sets of contiguous faces generated using a k-means style adaptation of SLIC [35] from the face graph of 3D meshes. It is one of the few algorithms that directly generates superfacets from 3D meshes and considers the performance of the oversegmentation process as a goal. The method uses as a feature, a combination of angular and geodesic weights to extract superfacets from meshes. In contrast to FSS's restrictive expansion strategy when computing the distances between a centroid and the faces in its radius, methods like [38] try to compute the distances to all vertices in every iteration. This approach is computationally expensive and cannot be scaled to large meshes. Other methods, especially feature and shape descriptors [36,23], use oversegmentation as a means to a goal *i.e.* shape/feature description, rather than the goal itself. Other approaches [47,46] adapted normalized cuts [37,11] algorithm to polygonal meshes for partitioning purposes. As a consequence, they also inherited the problems of computational and memory intractability. [10] is one of the earliest methods to create hierarchical clusters of faces by merging edges of a dual graph that is constructed on faces of a polygonal surface. The method uses a composite error metric to evaluate the dual merging process. Apart from the problems inherent in these methods, they either use handcrafted features or a hard face-superfacet association for oversegmentation. The latter makes it difficult to train these methods in an end-to-end manner.

While there is an increase in the use of deep networks to achieve various mesh-based vision tasks like feature retrieval [6], classification [1], and semantic segmentation [13], more attention needs to be given to the area of oversegmentation. Methods like PD-MeshNet [27] and LaplacianNet [34] involve the grouping of similar faces and vertices using edge contraction and spectral clustering techniques respectively, but they mainly used them as pooling operations towards achieving their main goals *i.e.* shape classification and semantic segmentation. As such, the metrics used to evaluate these methods' performances are per the networks' task. In contrast, our method focuses on superfacet generation and uses relevant evaluation metrics to assess the oversegmentation process. Recently, a region-growing approach [32,29] was used to oversegment 3D meshes to planarity-sensible segments in PSSNet [9]. Afterward, the segments are classified using a GCN. However, the oversegmentation technique employed by PSSNet [9] is non-differentiable. Thus, the method is not end-to-end trainable.

3. Related frameworks

In this section, we present the major theoretical underpinnings that serve as foundational frameworks for the most relevant parts of our method.

3.1. Definitions

Let $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ be an undirected graph with nodes, $\mathcal{V} = \{1, .., n\}$ and edges, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ s.t. $(a, b) \in \mathcal{E}$ iff $(b, a) \in \mathcal{E}$. $\mathcal{N}_a$ and $\mathcal{N}_a^n$ denotes the neighborhood and n-order neighborhood of node $A \in \mathcal{V}$ respectively.

Our concern is to get a function, $\mathcal{J} : \mathcal{V} \rightarrow \tilde{\mathcal{V}}$ that operates on the nodes, $\mathcal{V}$ by considering the relevance of first-order neighborhood nodes, $\mathcal{N}_V^1$ of each node in $\mathcal{V}$ to produce $\tilde{\mathcal{V}}$ as new nodal features. $\mathcal{V}$ and $\tilde{\mathcal{V}}$ can be denoted as vectors, $f \in \mathbb{R}^n$ and $\tilde{f} \in \mathbb{R}^{n \times c}$ respectively. We are also interested in an operation, $\mathcal{P} : \mathcal{E} \rightarrow \tilde{\mathcal{E}}$, where $\tilde{\mathcal{E}}$ represents a reduced version of $\mathcal{E}$ due to the collapse of edges in $\mathcal{G}$.

3.2. Graph attention networks (GAT)

GAT [43] is an attention technique that is used in convolutions to learn the relevance of each node in a neighborhood. Using attention scores that are computed from input features of neighboring nodes, the GAT convolution computes new features for nodes.

For arbitrary adjacent nodes $A, B \in \mathcal{V}$, with input feature vector f_a, the GAT convolution generates new a feature vector, $\tilde{f_a}$ using attention score α_{ab}, where $B \in \mathcal{N}_A^1$. The operations involved in the convolution operation are given below:

$$\tilde{f}_a = \xi\left(\sum_{r\in\mathcal{N}_a^1}\alpha_{ab}f_b\mathcal{W}\right) \tag{1}$$

$$\alpha_{ab} = \frac{e^{\eta(t([f_a\|f_b]))}}{\sum_{v\in\mathcal{N}_a^1}e^{\eta(t([f_a\|f_v]))}} \tag{2}$$

where ξ and η are activation functions, $\mathcal{W}$ denotes learnable weights for transforming features, and t denotes learned attention parameters. t can have multiple variants (*heads*) depending on the type of $\|$ that is used to transform the features, f_a and f_v.

In this paper, we use GAT convolution as $\mathcal{J}$. We use it to compute new features for nodes of graphs.

3.3. Dual-primal convolutions

Only the incident nodes are visible to the GAT technique (3.2) when computing the relevance of an edge in a convolution. This hinders the technique to consider the context in the neighborhood, thereby disregarding the contribution of each node. To address this limitation, Dual-Primal Graph CNN (DPGCNN) [28] constructed an additional dual graph with nodes representing the edges of the initial graph *a.k.a* primal graph. Alternating GAT convolutions are then operated on the primal and dual graphs. This way, both the node and edge features of the primal graph contribute to the neighborhood context during a convolution operation. PD-MeshNet [27] extended this GAT generalization architecture to 3D meshes. In their configuration, primal graph nodes represent faces while dual graph nodes are denoted as edges between adjacent faces of the mesh. Using this architecture, the network learns relevant features from the faces and edges of an input mesh.

In this paper, the dual-primal convolutions architecture is used to learn complex features from input mesh that are suitable for superfacet generation. These features reflect the relevance of all faces and edges in a neighborhood, thereby adequately representing a mesh's geometric properties.

3.4. Edge collapse operation

Edge collapse is an operation applied on graphs to merge two or more nodes by contracting the edges between them. Based on a selection criterion, edges are contracted. In most graph neural network architectures [8,20], edge collapse is utilized as a pooling technique. In [10], the operation is used to merge similar faces of a mesh in the course of achieving the task of the methods.

In this paper, we use edge collapse operation as $\mathcal{P}$. We use it to hierarchically establish the relationship between superfacets and their constituent faces.

4. The MO-SEAL network

Here, we describe the steps taken to implement the various frameworks in the previous section, and how their combination culminates in our proposed end-to-end trainable network: MO-SEAL.

4.1. Dual-primal graphs construction

Given an input mesh, $\mathcal{M}$ Fig. 3(a) we construct the dual and primal graphs in the following steps:

Let $\mathcal{G}_P = \{\mathcal{F}, \mathcal{E}_p\}$ (Fig. 3b) be an undirected, primal graph created on mesh $\mathcal{M}$ (Fig. 3a), with faces $\mathcal{F} = \{1,\ldots,n\}$ representing primal nodes and primal edges, $\mathcal{E}_p \subseteq \mathcal{F}\times\mathcal{F}$ s.t. $(x,y)\in\mathcal{E}_p$ iff $(y,x)\in\mathcal{E}_p$, denoting adjacencies between neighboring faces. We denote f_x^p as the input feature of primal node $X\in\mathcal{F}$ and $F_P = \{f_n^p : n = 1,\ldots,|\mathcal{F}|\}$ as set of input primal features. $\mathcal{G}_P^t$ denotes the primal graph at the t-th layer of the network. We denote the input vector for every face, $X\in\mathcal{M}$ as:

$$f_x^p = X_{area}/\mathcal{F}_{area} \tag{3}$$

where $f_x^p \in F_P, X_{area}$ represents the area of face X and $\mathcal{F}_{area}$ denotes the summation of all areas of faces in $\mathcal{M}$.

Let $\mathcal{G}_D = \{\mathcal{E}_d = \mathcal{E}_p, \tilde{\mathcal{E}_d}\}$ (Fig. 3c) be a dual graph created from primal graph $\mathcal{G}_P$, with each dual node, $(x,y)\in\mathcal{E}_d$ corresponding to a primal edge, $(x,y)\in\mathcal{E}_p$ with undirected edges $\tilde{\mathcal{E}_d} = \{1,\ldots,|\tilde{\mathcal{E}_d}|\}$, where $|\tilde{\mathcal{E}_d}| = \frac{1}{2}\sum_{i=1}^{n} d_x^2 - |\mathcal{E}_d|$ and d_x is the degree of primal node X. $(xy,xu)\in\tilde{\mathcal{E}_d}$ represents an edge between dual nodes $(x,y),(x,u)\in\tilde{\mathcal{E}_d}$. We denote $f_{(x,y)}^d$ as the input feature of dual node $(x,y)\in\mathcal{E}_d$ and $F_D = \left\{f_n^d : n = 1,\ldots,|\mathcal{E}_d|\right\}$ as set of input dual features. $\mathcal{G}_D^t$ denotes the dual graph at the t-th layer of the network. We represent the input vector for dual node (x,y) for every edge, $X,Y\in\mathcal{M}$ (Fig. 3a) as follows:

$$f_{(x,y)}^d = \left[\frac{|(x,y)|}{|(x,u)|}, \frac{|(x,y)|}{|(x,z)|}, \frac{|(x,y)|}{|(y,u)|}, \frac{|(x,y)|}{|(y,z)|}, \frac{|(x,y)|}{h_X}, \frac{|(x,y)|}{h_Y}, \gamma_{XY}\right]^{\mathsf{T}} \tag{4}$$

where $f^d_{(x,y)} \in F_D, |(a,b)|$ is the length of edge $(a,b), h_A$ denotes height of face A and γ_{AB} is the dihedral angle between faces A and B.

Our choice for the configuration in (3) is motivated by the need to cluster faces as opposed to other elements like vertices or edges of the mesh while the setting in (4) follows the strategy of similar versions in MeshCNN [13] and PD-MeshNet [27]. The latter has proven to be an efficient feature selection criteria for vision tasks related to 3D meshes.

4.2. Dual-primal convolutions

After constructing the dual and primal graphs from the input mesh (4.1), alternating dual and primal convolutions are operated on them respectively.

4.2.1. Dual convolution

Starting with the dual convolution, GAT is operated on the dual graph $\mathcal{G}^t_D$ (for brevity $\mathcal{G}_D$) to compute features $\widetilde{F}_D$ on the edges of the primal graph. This is because every dual node in the dual graph corresponds to an edge in the primal graph (4.1). These features are subsequently used to compute the attention scores for primal edges in the GAT convolutional layer (Fig. 3c).

To highlight this operation, let us consider the dual node (x,y) *a.k.a* primal edge (x,y) in Fig. 3c. Given f^d_{xy} as its input feature vector, we generate a new dual feature vector, $\tilde{f}^d_{xy}$ by applying (1). The attention scores, $\alpha^d_{xy,xu}$ and $\alpha^d_{xy,yz}$ associated with the first-order edges of the node are also generated through (2). As illustrated in the figure, these scores are computed for edges, (xy,xu) and (xy,yz) in the directions, $(x,u) \Rightarrow (x,y)$ and $(y,z) \Rightarrow (x,y)$ respectively. The new dual feature, $\tilde{f}^d_{xy}$ is subsequently used in computing the primal feature, $\tilde{\mathbf{f}}^p_x$ in the ensuing primal convolution layer.

In this work, we use the dual convolution operation to generate primal edge features which correspond to features on the edges of the mesh. The operation corresponds to the communication of contextual information between the edges of the mesh. It helps to generate features that reflect the behavior of edges in a neighborhood rather than considering the behavior of only incident faces. This leads to the generation of better attention scores for the edges. Most importantly, it helps the network to identify edges and faces that are located on the boundaries of the mesh surface.

4.2.2. Primal convolution

The output features, $\widetilde{F}_D$ from the dual convolution layer are used in generating primal attention scores, α^P for edges in the primal graph (2). These scores are used by GAT convolution on the primal graph to produce primal node features, $\widetilde{F}_P$ (1).

To illustrate this operation, let us consider primal node X and edge, (x,y) in Fig. 3(b). Given f^p_x as the input feature vector of the node, we generate a new primal feature vector, $\tilde{f}^p_x$ by applying (1). The attention scores, α^p_{xy} and α^p_{yx} along the directions $X \Rightarrow Y$ and $Y \Rightarrow X$ are computed using (2) respectively. The key difference with the GAT technique in (1) is the use of dual features, $\tilde{F}_D$ in generating the primal attention scores.

In this work, primal convolution is equivalent to generating features on the faces of the mesh. The features generated not only reflect the geometric qualities of the faces, but also the behavior and contextual information of their neighboring edges. This property is very important in predicting the strength of relationships between faces in the mesh and correctly identifying the mapping between faces and their corresponding superfacets.

4.3. Edge collapse (Superfacet formation)

Similar to [27], we use the sum of primal attention scores as edge selection criteria for the edge collapse operation. This is because the attention scores represent relationship strength between faces in the mesh. Contracting primal edges with higher scores is equivalent to merging closely-related faces into superfacets.

To illustrate this concept, let us consider the edge (x,y) in Fig. 3b. The value of α^{cum}_{xy} compared to other scores, determines whether the edge will be collapsed or not.

$$\alpha^{cum}_{xy} = \alpha_{xy} + \alpha_{yx} \tag{5}$$

In this work, we use the edge collapse operation on the primal edges to cluster faces with similar features. By applying the operation multiple times in progressive primal convolution layers, hierarchical mappings are established between faces and their corresponding superfacets. As the network is trained, it learns to collapse primal edges that will lead to more accurate representations of these association maps. This way the network learns to predict high-quality superfacets. Another advantage of the hierarchical face-superfacet association is, the ability to trace the mappings between faces and superfacets at each layer of the network. This gives the property of *connectivity enforcement*, which is an important characteristic of oversegmentation methods in both images [48,16] and point clouds [30,15].

In contrast to other oversegmentation methods [16,19], the number of superfacets to generate is not explicitly given to the network. Rather, our network determines the number of superfacets based on the value of $\mathcal{N}_{preserve}$ that is given by the user. Therefore, the number of superfacets to be generated by the network can be controlled using this parameter.

The formulation for determining the number of new primal edges to collapse is as follows:

$$\mathcal{E}_{collapse} = |\mathcal{E}_P|(1 - \mathcal{N}_{preserve}) \tag{6}$$

where $\{\mathcal{N}_{preserve} \in \mathbb{Q} : 0 > \mathcal{N}_{preserve} < 1\}$ is user-defined.

4.4. Dual-primal graphs reconstruction

After collapsing the primal edges, a new primal graph $\mathcal{G}_P^{t+1}$ has to be reconstructed to serve as input to the next layer of the network. To compute a node value of the new graph, the initial values of the nodes that were merged into the new node are aggregated together. To illustrate the two aggregation methods we use, let us consider arbitrary adjacent faces $X_1, X_2, .., X_n \in \mathcal{F}$ that are merged to form a superfacet, $\{x_1 x_2 .. x_n\} \in \mathcal{F}^n$. The feature, $f^p_{\{x_1 x_2 .. x_n\}}$, of the superfacet can be computed using a mean *mean* $\left(\underset{mean}{\Leftarrow}\right)$ aggregation:

$$f^p_{\{x_1 x_2 .. x_n\}} = \frac{\sum_{i=1}^{n} f^p_{x_i}}{n} \tag{7}$$

or an addition (*add*: $\underset{add}{\Leftarrow}$) aggregation:

$$f^p_{\{x_1 x_2 .. x_n\}} = \sum_{i=1}^{n} f^p_{x_i} \tag{8}$$

After computing the values of the new nodes, they are connected using undirected edges $\mathcal{E}_P^{t+1}$ to complete the reconstruction of $\mathcal{G}_P^{t+1}$. Similarly, the dual graph $\mathcal{G}_D^{t+1}$ is reconstructed. These new graphs are the input to the next layer in the network. After the last layer (k in Fig. 2 and Algorithm 1), the parent values of vectors $\widetilde{F}_P^k \in \mathbb{R}^{n \times m}$ are assigned to constituent children nodes. This is possible due to the hierarchical mappings that are established between parent nodes (superfacets) and their child nodes (faces that comprise superfacet). This soft face-superfacet association map, $Q \in \mathbb{R}^{n \times m}$ serves as input to our loss function.

Algorithm 1: The MO-SEAL Network

Input: $\mathcal{G}_D^t = \{\mathcal{E}_d, \tilde{\mathcal{E}}_d\}$ with node features $F_{D\ n\times 7}^t$

$\mathcal{G}_P^t = \{\mathcal{F}, \mathcal{E}_p\}$ with node features $F_{P\ n\times 1}^t$

$\mathcal{N}_{preserve}$

Output: face-superfacet association $Q_{n\times m}$

1: **for** each layer t from 1 to k **do**
2: $\alpha^D \leftarrow F_D^t$ //Compute dual attention scores
3: $\tilde{F}_D^t = \mathcal{J}(\mathcal{G}_D{}^t, \alpha^D)$ //Compute new features on primal edges
4: $\alpha^P \leftarrow \tilde{F}_D^t$ //Compute primal attention scores
5: $\tilde{F}_P^t = \mathcal{J}(\mathcal{G}_P{}^t, \alpha^P)$ //Compute new primal features
6: $\mathcal{E}_{collapse}^t = |\mathcal{E}_P|(1 - \mathcal{N}_{preserve})$ //Determine number of primal edges to collapse
7: $\mathcal{E}_P^{t+1} = \mathcal{P}\left(\mathcal{G}_P^t, \mathcal{E}_{collapse}^t\right)$ //Collapse primal edges
8: $F_P^{t+1} \underset{mean/add}{\Leftarrow} F_P^t$ //Aggregate constituent collapsed nodes
9: $G_P^{t+1} = \left\{\mathcal{F}^{t+1}, \mathcal{E}_P^{t+1}\right\}$ //Reconstruct primal graph
10: $\mathcal{E}_d^{t+1} \underset{mean/add}{\Leftarrow} F_D^t$
11: $\widetilde{\mathcal{E}}_d^{t+1} = \Longleftrightarrow$ //connect nodes using undirected edges
12: $G_D^{t+1} = \left\{\mathcal{E}_d^{t+1}, \widetilde{\mathcal{E}}_d^{t+1}\right\}$
13: **if** $t < k$
14: Repeat steps 2–12 using G_D^{t+1} and G_P^{t+1} as input dual and primal graphs respectively
15: **end if**
16: **end for**
return $\mathcal{F}^k$ //$\mathcal{F}^k \equiv Q_{n\times m}$

4.5. Segmentation-aware loss

We construct a segmentation-aware loss to optimize the network. We leverage the ground-truth semantic labels of faces in segmentation datasets (e.g., COSEG [44]) as semantic affinities, and try to consistently reconstruct them by training the network in an end-to-end fashion.

The expectation is that reconstructed labels of faces should be the same as their counterparts in the dataset. Given $R = \{r_i \in \mathbb{R}^l | i = 1, \ldots, n\}$ as initial label vector of faces, where r_i is a one-hot vector and $R \in \mathbb{R}^{n \times l}$, the network tries to predict $W = \{w_v \in \mathbb{R}^l | v = 1, \ldots, n\}$, with w_v as a one-hot vector and $W \in \mathbb{R}^{n \times l}$, as reconstructed face labels. Using Q (k^{th} face-superfacet association map) from (4.3), the formulation of the segmentation-aware loss is as follows:

To map the face semantics affinities, R onto their corresponding superfacet representations, Q, we use a simple matrix multiplication between the former and the latter's transposed version, $Q^\top$,

$$S = Q^\top \bullet R$$

$$S = \mathbb{R}^{m \times n} \bullet \mathbb{R}^{n \times l}, \quad \text{where } S \in \mathbb{R}^{m \times l}$$

This mapping process enables downstream applications to seamlessly substitute faces with their corresponding superfacets as basic elements. The soft face-superfacet mappings, Q as opposed to hard associations, combined with the simple matrix multiplications make it possible to create a differentiable loss function. And a commensurate differentiable loss function is what is required to train the network in an end-to-end fashion.

The prior mapping operation is reversed by multiplying the row-normalized association map, $\widetilde{Q}$ with the superfacet representations, S i.e.,

$$W = \widetilde{Q} \bullet S$$

$$W = \mathbb{R}^{n \times m} \bullet \mathbb{R}^{m \times l}, \quad \text{where } W \in \mathbb{R}^{n \times l}.$$

The result, W is the network's prediction/reconstruction of R. The aim is to train the network to consistently reconstruct R as W. This way, the network learns to accurately predict the hierarchical face-superfacet association maps which naturally translates to boundary-aware superfacets.

Mathematically, the segmentation-aware loss is denoted as

$$\begin{aligned} L_{seal} &= \mathcal{L}(R, W) \\ &= \mathcal{L}\left(R, \widetilde{Q} Q^\top R\right) \end{aligned} \tag{9}$$

We choose cross-entropy loss for $\mathcal{L}(.,.)$. Our choice is partly influenced by the nature of face affinities and the downstream application of semantic segmentation. We use the task of semantic segmentation to demonstrate the quality of our generated superfacets. Cross-entropy loss has been shown [16,48,15] to work well with these configurations. It is important to note that R is only visible to the network during training, the network predicts W in an unsupervised way in the test phase. The superfacets that are generated during testing are used for evaluation purposes.

For the task of superfacet semantic segmentation, we also choose cross-entropy loss which is computed on the labels of the input mesh. Mathematically, the total loss that is optimized for training the network is:

$$L_{\text{total}} = L_{ce} + \lambda L_{seal} \tag{10}$$

We realized as in [15,16,49], that multiplying the loss with a weight λ, helps in the optimization process. We set λ to 10^{-1} for all our experiments.

4.6. The pseudocode of the MO-SEAL network

This section presents pseudocode describing the implementation of the superfacet generation part of our network. The definitions of notations we use are shown in Table 1, and the pseudocode is described in Algorithm 1.

5. Implementation details, experiments, and results analysis

5.1. System details

In this section, we describe the network configurations (5.1.2) for our experiments and the datasets (5.1.1) we use to evaluate (5.1.3) the performance of our method.

Table 1
The meaning of notations used in the MO-SEAL method.

Notation	Definition	Notation	Definition
$\mathcal{G}_D^t$	Dual graph	$\mathcal{G}_P^t$	Primal graph
$\mathcal{E}_d$	Dual nodes	$\mathcal{F}$	Primal nodes
$\tilde{\mathcal{E}}_d$	Dual edges	$\mathcal{E}_p$	Primal edges
$F_{D\ n\times 7}^t$	Dual node features	$F_{P\ n\times 1}^t$	Primal node features
α^D	Dual attention scores	α^P	Primal attention scores
$\tilde{F}_D^t$	New dual features	$\tilde{F}_P^t$	New primal features
$\mathcal{N}_{preserve}$	Fraction of primal edges to preserve	$\mathcal{J}$	GAT convolution
$\mathcal{E}_{collapse}^t$	Number of primal edges to collapse	$\lvert\mathcal{E}_P\rvert$	Number of primal edges
$\mathcal{P}$	Primal edges collapse operation	$\Longleftrightarrow$	Undirected edges
$\Leftarrow_{mean/add}$	Nodes aggregation method	$\mathcal{F}_P^k$	Final primal nodes

5.1.1. Datasets

In this paper, we evaluate our method on two datasets: COSEG [44] and Human Body [26]. They are mostly used to evaluate the task of semantic segmentation in 3D meshes. The labeling is on the faces of the meshes.

The COSEG dataset has three categories of objects: *aliens*, *chairs,* and *vases*. The first category contains 198 models with 4 classes of labels, the second contains 397 models with 3 classes labeled, and the third category is composed of 297 models with 4 class labels. We use models that are downsampled to 1,500 faces as in PD-MeshNet [27] for both training and testing, to have fair comparisons of results.

The Human Body dataset contains 381 training models and 18 test models. Unlike in [13,27] where the resolution of the models is down-sampled to 2,250 edges and 1,500 faces respectively, we use full resolution meshes for our evaluations. This additionally increases the robustness of the model and provides the advantage of better qualitative results for visualization.

5.1.2. Network settings

The network is implemented using the Pytorch [31] deep learning framework and Pytorch Geometric [7], a graph neural network library. We use Adam [17] optimiser for all experiments with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We train the model for 1000 epochs with a learning rate of 0.001.

The model comprises 5 (k in Fig. 2) stacked layers, each consisting of primal convolution, dual convolution, and edge collapse layers. 3 attention heads are used for each convolution. The layers are connected by a skip connection. The input channels for the dual and primal convolution layers are 1 (3) and 7 (4), as described in (4.1). The output channels for both dual and primal convolutions in each layer of the stack are 16, 32, 64, 128, and 256 respectively. For each pooling layer, the value of $\mathcal{N}_{\sqrt{}^{\nabla\rceil\int\lceil\nabla\sqsubseteq\rceil}}^{\sqcup}$ is given, where t denotes the position of the layer in the stack.

5.1.3. Evaluation metrics

In [41], standard metrics for evaluating oversegmentation methods in images are proposed. These metrics were adapted to evaluate the performance of point cloud oversegmentation methods by [18]. Similarly, [40] extended this approach to 3D meshes by adapting two of these metrics. In this section, we define the two metrics we use to evaluate the quality of the learned superfacets by our network.

Undersegmentation error: It measures the boundary adherence of the superfacets by punishing the algorithm if a segment overlaps with ground-truth boundaries. Because pixels are structured on a grid, sampling is uniform in images but applying the same technique on meshes is not prudent due to the non-uniformity of triangular faces. Instead, areas of the faces are used to compute the mesh-equivalent by determining the fraction of the area that overlaps with the ground-truth borders. The formula is given below:

$$\mathrm{U}(s) = \frac{\sum_i \sum_{j:A(s_j \cap g_i) > threshold} A(s_j - g_i)}{\sum_i A(g_i)} \tag{11}$$

$A(.)$ is the area of a given number of faces, g_i denotes the area of faces in the i-th ground-truth segment, s_j denotes the sum of face areas in the j-th superfacet and *threshold* is the tolerable fraction of areas that overlaps with the ground-truth. We use $(5/100) \cdot A(s_j)$ as a *threshold*, the same value in [40]. $s_j - g_i$ refers to all faces that are in j-th superfacet but not in i-th ground-truth segment.

A lower U is equivalent to having a small overlap with ground-truth segments. Hence, lower is better.

Compactness: It measures the contiguousness of faces that form a superfacet. The more tightly-packed the faces in a superfacet are, the higher the compactness value i.e. higher is better. Also, a higher value translates to higher connectivity enforcement between faces in a superfacet. The formula is given below:

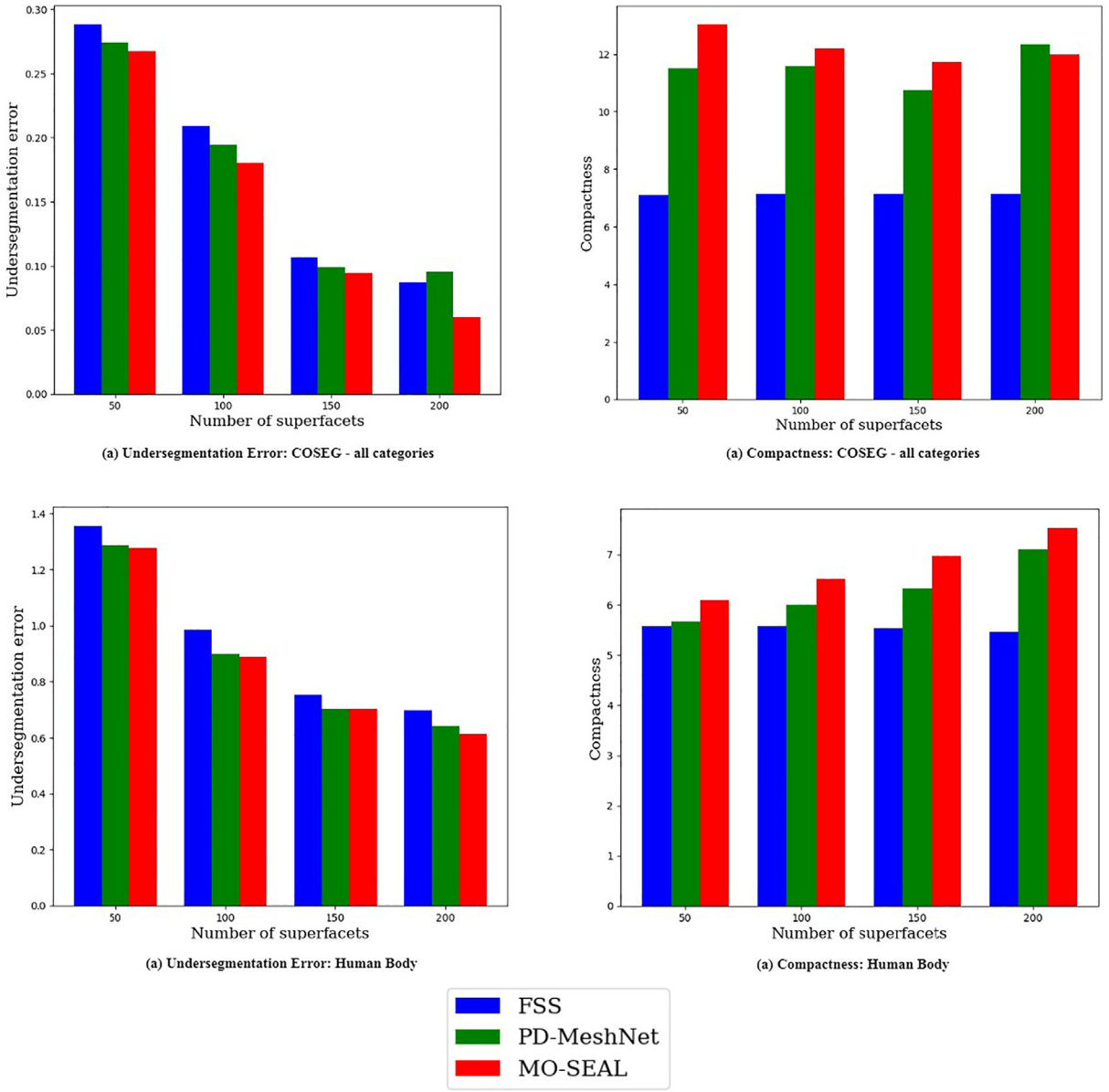


Fig. 5. Metric results of different methods for different number of superfacets on test models of COSEG and Human Body datasets. The COSEG results are averaged over the 3 categories of the dataset.

$$\text{compactness}(s) = \text{average}_j \left(\frac{P(s_j)}{\sqrt{A(s_j)}} \right) \tag{12}$$

where $P(.)$ denotes the perimeter of j-th superfacet.

5.2. Learned superfacets and comparison with state-of-the-art

The two competing algorithms we consider for comparison are FSS [40] and the superpixel-like configuration of [27]. We denote the latter as PD-MeshNet. FSS is a mesh oversegmentation algorithm that iteratively generates superfacets using a k-means algorithm and uses (5.1.3) as evaluation metrics. PD-MeshNet generates clusters of faces as a by-product of 3D semantic segmentation. Even though the network is not meant for generating superfacets, we extracted the face clusters to compare with our superfacets. As suggested by FSS, we use $\alpha = 200$ as the angular weight value to generate superfacets.

Note that superfacet evaluations (5.2.1 and 5.2.2) are carried out at the superfacet level, while semantic segmentation results (5.3) are obtained at the face level.

5.2.1. Quantitative results

We conducted experiments on COSEG [44] and Human Body [26] datasets to assess our method. The quantitative results are illustrated in Fig. 5. To have a fair comparison, the number of superfacets produced by methods on a dataset is just about the same. For instance, the number of superfacets generated on *shrec*_17 mesh model of the Human Body dataset is 207 for FSS, 206 for PD-MeshNet, and 203 for MO-SEAL (ours) respectively. The values in Fig. 5(a and b) are the average results taken across all 3 categories of the COSEG dataset. From the figures, it can be seen that our method outperforms the other methods. In contrast to the handcrafted features used in FSS [40] to generate superfacets, the features learned by our method represent the geometric information within and between faces better. Therefore, the superfacets generated by our method are of higher quality. Notice how the compactness results of FSS are almost the same in both Fig. 5(a) and (c), we believe this is due to a trade-off between compactness and boundary adherence to concavities. This trade-off is controlled by the angular weight term, α which tries to strike a balance between the two properties. Using a similar value ($\alpha = 200$) in all experiments gives the results in the figures. On the other hand, our method does not encounter such a restriction because it uses the relevant learned features to generate superfacets that are as compact as possible while respecting boundaries. The key difference between our method and PD-MeshNet [27] is the segmentation-aware loss that directs the network to generate boundary-aware superfacets. Consequently, our method has a better performance on both the metrics of compactness and undersegmentation error.

5.2.2. Visualization results

In Fig. 6, we present the visualizations of the superfacets generated by different methods. As seen from the figure, the superfacets generated by our method and PD-MeshNet [27] adhere more to boundaries than those generated by FSS [40].

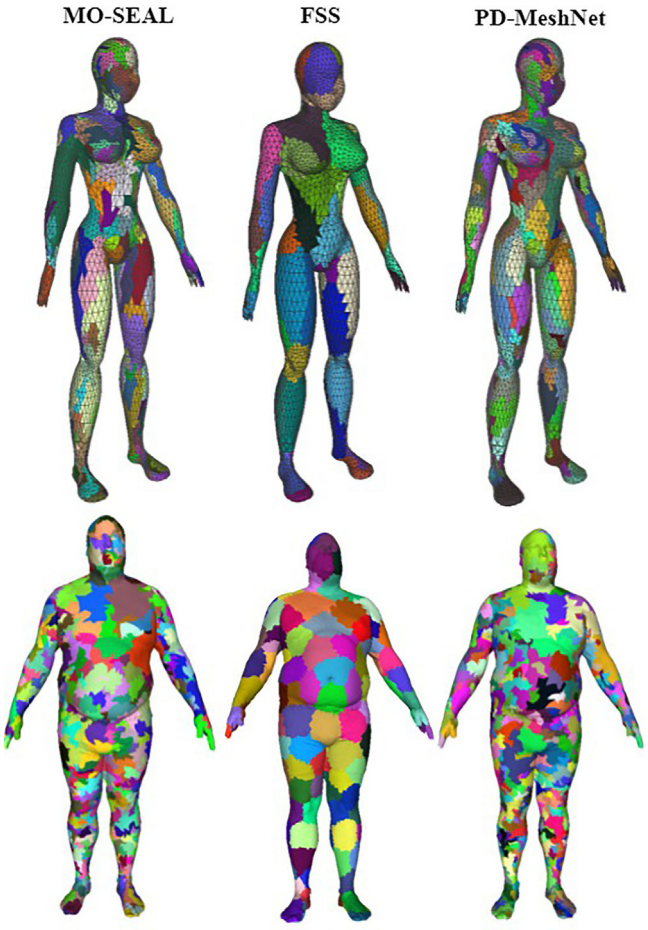


Fig. 6. Visual results of different methods on test models of the Human Body dataset. Different colors correspond to different clusters. The number of generated superfacets for each model is 200.

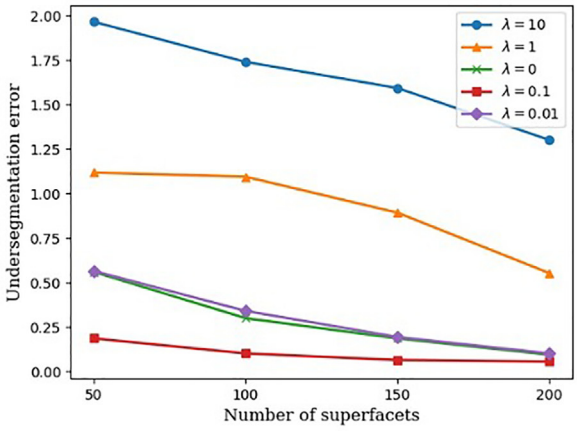
At the expense of trespassing boundaries, notice the regularity in the shape of superfacets generated by FSS. This is a result of the trade-off between the two adversarial properties of shape regularity (compactness) and concavity adherence (boundary-awareness) in the method. Even though the superfacets generated by our method are more irregular, their boundaries are more clear and accurate.

The superfacets in our method are generated without supervision during the testing period. Also, it can be seen from the visuals that our superfacets are bigger in flat regions that cover semantically consistent faces *e.g.* the chest area in the second-row models. This is due to our method's ability to cluster semantically similar faces without emphasizing the shape of the generated superfacet. However, the irregular superfacets are more consistent in covering semantic regions of the models than FSS's superfacets.

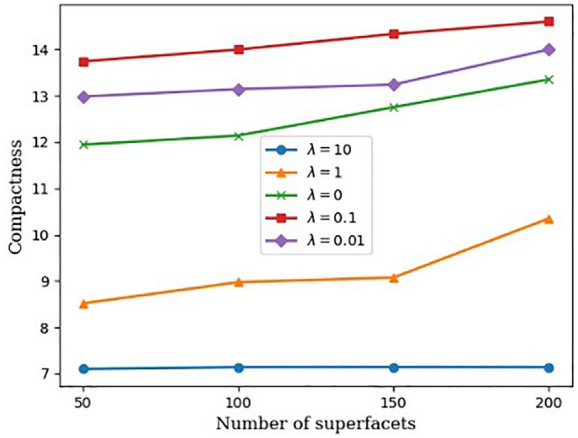
5.2.3. Ablation experiments

This section presents ablation studies to justify the settings we use in our experiments. We conduct ablation experiments to evaluate the performance of our method on superfacet generation using different settings.

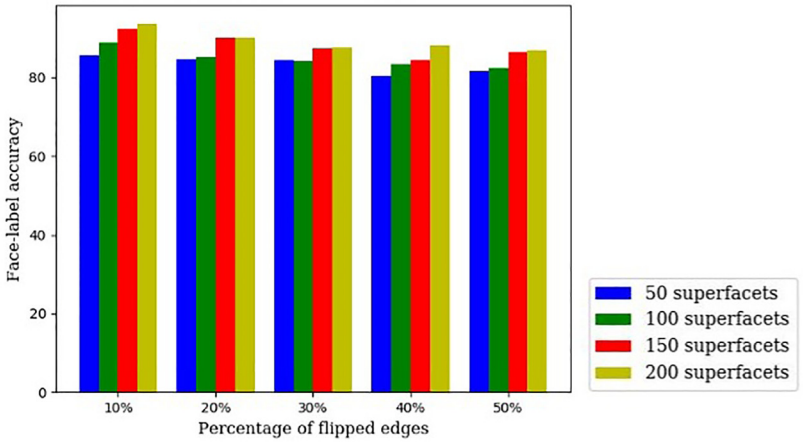
We conducted ablation experiments to justify the settings we use in our experiments, and illustrate the results in Fig. 7. Specifically, we explore the effect of the hyper-parameter λ, on the performance of our method. We show the curves of compactness and undersegmentation error on test models of the *aliens* category of the COSEG dataset. Only the value of the parameter is changed while the remaining parameter values are unchanged. It can be noticed that we obtained the best result when λ is 10^{-1}.



(a) Parameter λ : Undersegmentation error



(b) Parameter λ : Compactness



(c) Effect of flipping mesh edges on face-label accuracy

Fig. 7. Ablation experiment results investigating the effect of different parameter values of λ on **(a)** undersegmentation error and, **(b)** compactness metrics. Another ablation experiment to determine the **(c)** effect of flipping edges of the mesh on the face-label accuracy. Experiments **(a)** and **(b)** are conducted on test models of *aliens* while the experiments in **(c)** are done on *vases* categories of the COSEG dataset respectively.

We conducted additional experiments to investigate the choice of aggregation method ($\Leftarrow_{mean}$ (7) or $\Leftarrow_{add}$ (8)) on undersegmentation error and compactness for the different number of superfacets. The experiment is conducted on test models of the Human Body [26] dataset. We present the results in Table 2. From the table, the accuracy results demonstrate the superiority of $\Leftarrow_{add}$.

5.3. Semantic segmentation

This section presents superfacet semantic segmentation results obtained by our method on COSEG [44] and Human Body [26] semantic segmentation datasets.

5.3.1. Superfacet-based semantic segmentation

For a just comparison, we adopt the settings in PD-MeshNet (*superpixel – like segmentation*) [27] to get results on the Human Body dataset. For the COSEG dataset, we use the results as reported. We adopt the accuracy (Acc) metric on face labels to evaluate the results. Note that the results presented are superfacet-based.

According to Table 3, our method outperformed PD-MeshNet on the two datasets. The results demonstrate the effect our segmentation-aware loss has on generating high-quality superfacets that led to the improvement in semantic segmentation performance.

5.3.2. Ablation experiment

A good characteristic of an oversegmentation method is the ability of the user to specify the number of clusters to generate [41]. Our method fulfills this requirement using $\mathcal{N}_{preserve}$ (6), which is supplied by the user. To investigate the effect of $\mathcal{N}_{preserve}$ and $\mathcal{E}_{collapse}$ on superfacet segmentation performance, we conducted experiments on the test models of *vases* category of the COSEG dataset and present the results in Table 4. The results from the table demonstrate that higher values of $\mathcal{N}_{preserve}$ correspond to higher face-label accuracy. Also, from the table, our method is robust enough to perform competitively with a significant reduction in $\mathcal{E}_{collapse}$.

We conducted further experiments to understand the effect of flipping fractions of the edges of the input mesh on the face-label accuracy metric. An edge between two faces is considered flipped only if [27]: the flipping operation does not flip

Table 2
Undersegmentation error and compactness results base on aggregation method. The experiments are conducted on test models of Human Body dataset.

Metric	Number of superfacets			
	50	100	150	200
U (*add*)	1.27	0.89	0.70	0.61
U (*mean*)	1.31	0.96	0.87	0.81
compactness (*add*)	6.08	6.52	6.97	7.53
compactness (*mean*)	6.02	6.14	6.48	6.76

Table 3
Results of superfacet semantic segmentation on test models of Human Body and the 3 categories of COSEG datastes.

Dataset		Method	Face-label accuracy
Human Body		PD-MeshNet	82.45%
	MO-SEAL	84.59%	
COSEG	*aliens*	PD-MeshNet	94.75%
	MO-SEAL	95.24%	
	chairs	PD-MeshNet	93.74%
	MO-SEAL	96.23%	
	vases	PD-MeshNet	92.79 %
	MO-SEAL	95.09 %	

Table 4
Ablation study results of different $\mathcal{N}_{preserve}$ on test models of vases category of the COSEG dataset.

Fraction of Preserved Primal Edges	Number of Superfacets	Face-label accuracy
(0.75, 0.75, 0.75, 0.75, 0.75)	50–60	90.36%
(0.85, 0.85, 0.8, 0.7, 0.75)	100–110	90.69%
(0.85, 0.85, 0.75, 0.8, 0.8)	150–160	92.39%
(0.85, 0.85, 0.85, 0.8, 0.78)	200–210	92.78%

the face normals, the flipped edge is manifold and, the dihedral angle between the associated faces is between *min_dihedral* and $2 \setminus \pi - min\_dihedral$. We set $min\_dihedral = 0$ for our experiments. We conducted the experiments on the *vases* category of the COSEG dataset and present the results in Fig. 7(c). From the figure, it can be seen that our method has the resilience to perform competitively with an increasing number of flipped edges.

6. Conclusions

This paper developed an end-to-end deep oversegmentation network for generating superfacets. The network learns to generate boundary-aware superfacets with the guidance of a segmentation-aware loss we formulated. To make the network differentiable, we generate a soft-superfacet association map and combine it with the loss. Extensive experiments on two 3D mesh datasets demonstrate that the learned superfacets are not only better than those generated by existing methods but also improve the performance of superfacet semantic segmentation.

GAT [43] convolution involves holding both primal and dual graphs, their adjacency matrices, and node features in memory during training. This makes it prohibitive to scale the method to urban- and city-scale meshes. In the future, addressing this limitation using memory-efficient convolution strategies can be explored.

CRediT authorship contribution statement

Jibril Muhammad Adam: Conceptualization, Investigation, Methodology, Writing – original draft. **Muhammad Kamran Afzal:** Writing – review & editing. **Zang Yu:** Writing – review & editing. **Saifullahi Aminu Bello:** Writing – review & editing. **Cheng Wang:** Writing – review & editing, Supervision. **Jonathan Li:** Writing – review & editing, Supervision.

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.

Acknowledgement

This work was supported by the National Natural Science Foundation of China, number 61971363.

References

[1] Discriminative Feature Abstraction by Deep L2 Hypersphere Embedding for 3D Mesh CNNs, Inf. Sci. (2022) 1158–1173.
[2] R. Achanta, S. Susstrunk, Superpixels and Polygons using Simple Non-Iterative Clustering, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4651–4660.
[3] Y. Ben-Shabat, T. Avraham, M. Lindenbaum, A. Fischer, Graph Based Over-Segmentation Methods for 3D Point Clouds, Comput. Vis. Image Underst. 174 (2018) 12–23.
[4] Z. Chen, K. Yin, M. Fisher, S. Chaudhuri, H. Zhang, BAE-NET: Branched Autoencoder for Shape Co-Segmentation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 8490–8499.
[5] P.F. Felzenszwalb, D.P. Huttenlocher, Efficient Graph-based Image Segmentation, Int. J. Comput. Vision 59 (2) (2004) 167–181.
[6] Y. Feng, Y. Feng, H. You, X. Zhao, Y. Gao, Meshnet: Mesh Neural Network for 3D Shape Representation, Proceedings of the AAAI Conference on Artificial Intelligence vol. 33 (2019) 8279–8286.
[7] M. Fey, J.E. Lenssen, Fast Graph Representation Learning with PyTorch Geometric, arXiv preprint arXiv:1903.02428.
[8] H. Gao, S. Ji, Graph U-Nets, in: International Conference on Machine Learning, PMLR, 2083–2092, 2019.
[9] W. Gao, L. Nan, H. Ledoux, B. Boom, PSSNet: Planarity-sensible Semantic Segmentation of Large-scale Urban Meshes, arXiv preprint arXiv:2202.03209.
[10] M. Garland, A. Willmott, P.S. Heckbert, Hierarchical Face Clustering on Polygonal Surfaces, in: Proceedings of the 2001 Symposium on Interactive 3D graphics, 49–58, 2001.
[11] A. Golovinskiy, T. Funkhouser, Randomized Cuts for 3D Mesh Analysis, in: ACM SIGGRAPH Asia, 1–12, 2008.
[12] S. Guinard, L. Landrieu, Weakly Supervised Segmentation-Aided Classification of Urban Scenes from 3D LiDAR Point Clouds, in: ISPRS Workshop 2017, 2017.
[13] R. Hanocka, A. Hertz, N. Fish, R. Giryes, S. Fleishman, D. Cohen-Or, MeshCNN: A Network with an Edge, ACM Trans. Graphics 38 (4) (2019) 1–12.
[14] Q. Huang, V. Koltun, L. Guibas, Joint Shape Segmentation with Linear Programming, in: Proceedings of the 2011 SIGGRAPH Asia Conference, 1–12, 2011.
[15] L. Hui, J. Yuan, M. Cheng, J. Xie, X. Zhang, J. Yang, Superpoint Network for Point Cloud Oversegmentation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 5510–5519.
[16] V. Jampani, D. Sun, M.-Y. Liu, M.-H. Yang, J. Kautz, Superpixel Sampling Networks, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 352–368.
[17] D.P. Kingma, J. Ba, Adam: A Method for Stochastic Optimization.
[18] L. Landrieu, M. Boussaha, Point Cloud Oversegmentation with Graph-Structured Deep Metric Learning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 7440–7449.
[19] 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, 4558–4567, 2018.
[20] J. Lee, I. Lee, J. Kang, Self-Attention Graph Pooling, in: International Conference on Machine Learning, PMLR, 3734–3743, 2019.
[21] Z. Li, J. Chen, Superpixel Segmentation using Linear Spectral Clustering, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1356–1363.
[22] Y. Lin, C. Wang, D. Zhai, W. Li, J. Li, Toward Better Boundary Preserved Supervoxel Segmentation for 3D Point Clouds, ISPRS J. Photogrammetry Remote Sens. 143 (2018) 39–47.
[23] R. Litman, A.M. Bronstein, M.M. Bronstein, Stable Volumetric Features in Deformable Shapes, Comput. Graphics 36 (5) (2012) 569–576.

[24] M.-Y. Liu, O. Tuzel, S. Ramalingam, R. Chellappa, Entropy Rate Superpixel Segmentation, in: CVPR 2011, IEEE, 2097–2104, 2011.
[25] Y.-J. Liu, C.-C. Yu, M.-J. Yu, Y. He, S.L.I.C. Manifold, A Fast Method to Compute Content-Sensitive Superpixels, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 651–659.
[26] H. Maron, M. Galun, N. Aigerman, M. Trope, N. Dym, E. Yumer, V.G. Kim, Y. Lipman, Convolutional Neural Networks on Surfaces via Seamless Toric Covers, ACM Trans. Graphics 36 (4) (2017) 71.
[27] F. Milano, A. Loquercio, A. Rosinol, D. Scaramuzza, L. Carlone, Primal-Dual Mesh Convolutional Neural Networks, in: Proceedings of the 34th International Conference on Neural Information Processing Systems, Curran Associates Inc., 2020.
[28] F. Monti, O. Shchur, A. Bojchevski, O. Litany, S. Günnemann, M.M. Bronstein, Dual-Primal Graph Convolutional Networks, arXiv preprint arXiv:1806.00770.
[29] L. Nan, Easy3D: A Lightweight, Easy-to-use, and Efficient C++ Library for Processing and Rendering 3D Data, J. Open Source Software 6 (64) (2021) 3255.
[30] J. Papon, A. Abramov, M. Schoeler, F. Worgotter, Voxel Cloud Connectivity Segmentation-Supervoxels for Point Clouds, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2027–2034, 2013.
[31] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al, Pytorch: An Imperative Style, High-performance Deep Learning Library, Advances in Neural Information Processing Systems 32 (2019) 8026–8037.
[32] T.C. Project, CGAL User and Reference Manual, CGAL Editorial Board, 5.5 edn. URL: https://doc.cgal.org/5.5/Manual/packages.html, 2022.
[33] 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.
[34] Y.-L. Qiao, L. Gao, J. Yang, P.L. Rosin, Y.-K. Lai, X. Chen, Learning on 3D meshes with Laplacian Encoding and Pooling, IEEE Trans. Visualization Comput. Graphics.
[35] X. Ren, J. Malik, Learning a Classification Model for Segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, IEEE Computer Society, 2003.
[36] E. Rodola, S.R. Bulo, D. Cremers, Robust Region Detection via Consensus Segmentation of Deformable Shapes, in: Computer Graphics Forum, vol. 33, Wiley Online Library, 97–106, 2014.
[37] J. Shi, J. Malik, Normalized Cuts and Image Segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 888–905.
[38] S. Shlafman, A. Tal, S. Katz, Metamorphosis of Polyhedral Surfaces using Decomposition, in: Computer Graphics Forum, vol. 21, Wiley Online Library, 219–228, 2002.
[39] G. Shu, A. Dehghan, M. Shah, Improving an Object Detector and Extracting Regions using Superpixels, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 3721–3727.
[40] P. Simari, G. Picciau, L. De Floriani, Fast and Scalable Mesh Superfacets, in: Computer Graphics Forum, vol. 33, Wiley Online Library, 181–190, 2014.
[41] D. Stutz, A. Hermans, B. Leibe, Superpixels: An Evaluation of the State-of-the-art, Comput. Vis. Image Underst. 166 (2018) 1–27.
[42] W.-C. Tu, M.-Y. Liu, V. Jampani, D. Sun, S.-Y. Chien, M.-H. Yang, J. Kautz, Learning Superpixels with Segmentation-Aware Affinity Loss, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 568–576.
[43] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph Attention Networks, 2018.
[44] Y. Wang, S. Asafi, O. Van Kaick, H. Zhang, D. Cohen-Or, B. Chen, Active Co-Analysis of a Set of Shapes, ACM Trans. Graphics 31 (6) (2012) 1–10.
[45] Y. Wang, Y. Wei, X. Qian, L. Zhu, Y. Yang, AINet: Association Implantation for Superpixel Segmentation, in: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 7058–7067.
[46] Z. Wu, R. Shou, Y. Wang, X. Liu, Interactive Shape Co-Segmentation via Label Propagation, Comput. Graphics 38 (2014) 248–254.
[47] Z. Wu, Y. Wang, R. Shou, B. Chen, X. Liu, Unsupervised Co-Segmentation of 3D Shapes via Affinity Aggregation Spectral Clustering, Comput. Graphics 37 (6) (2013) 628–637.
[48] F. Yang, Q. Sun, H. Jin, Z. Zhou, Superpixel Segmentation with Fully Convolutional Networks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 13964–13973.
[49] H. Zhao, J. Shi, X. Qi, X. Wang, J. Jia, Pyramid Scene Parsing Network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 2881–2890.