Title:
Protrusion-oriented Point Cloud Semantic Segmentation

Authors:
Alexander Agathos, agatha@aegean.gr, University of the Aegean
Philip Azariadis, fazariadis@uniwa.gr, University of West Attica

Keywords:
Point Cloud, Semantic Segmentation, Protrusion Function, Elliptic Gabriel Graph

DOI: 10.14733/cadconfP.2023.95-100

Introduction:

In recent years the acquisition of point clouds using different 3D sensors (LIDAR, structured light, etc.) or 2D camera sensors using photogrammetry has led to the need to give a higher semantic meaning to the captured raw data so as to allow for a higher-level processing in CAD applications or to use the semantics for various applications like autonomous driving, robotics, urban and rural classifications etc. A way to do this is through *point cloud semantic segmentation*. Point cloud semantic segmentation is the technique to assign to each vertex of the 3D point cloud a semantic label. Point cloud semantic segmentation differs from the known general term *point cloud segmentation*. In point cloud segmentation the aim is to partition the point cloud into homogenous areas which have a geometric shape like the planar roof and walls of a building, while in semantic segmentation the aim is to give to the segmented parts a characteristic that makes sense in human perception, like the handle of a cup, the wing of an airplane the head, arms and feet of a human figure. The later parts have complicated geometry and are difficult to be described by geometric surfaces.

Semantic segmentation can be unsupervised using various techniques like convexity analysis [3] or using the protrusion function like in the work presented in this paper. Recently, supervised techniques have been introduced using machine learning techniques like for example Maximum Likelihood Classifiers [4], Support Vector Machines [5], Random Forests [2]. Though the most active field of machine learning in semantic segmentation is nowadays deep learning. In this field the point cloud is either (i) transformed into multi-view images [9], (ii) transformed into voxels by partitioning it into a volumetric grid [6], and (iii) used directly with its coordinates [8] or with its edge connectivity [11]. In all cases convolutional neural networks are used to process and generate features. These convolutions are either one-dimensional, two-dimensional or three-dimensional.

The main problem with supervised learning is that there is a need of a significant number of annotated point clouds with labels to train the neural networks, which is cumbersome and needs a lot of human labor. Also, to make things worse, for each new part there is a need to provide new annotations for the network to process it. Of course, a neural network has the ability to generalize so as to be able to label points belonging in a variant of the specified class, but it can't generalize to label a completely new part that it has not been trained to recognize. In contrast, unsupervised semantic segmentation does not have this limitation. It can be used to segment a variety of objects using general geometric characteristics that a wide variety of objects have. A class of point clouds that are of interest in this work is those that are sampled from articulated objects, i.e., having protrusions. This work, extents the work of semantic segmentation of polygonal meshes presented in [1] to unstructured point clouds. With the proposed methodology there is no need to reconstruct the polygonal mesh surface of the object, a task that can be quite challenging and time consuming, especially with the presence of noise.

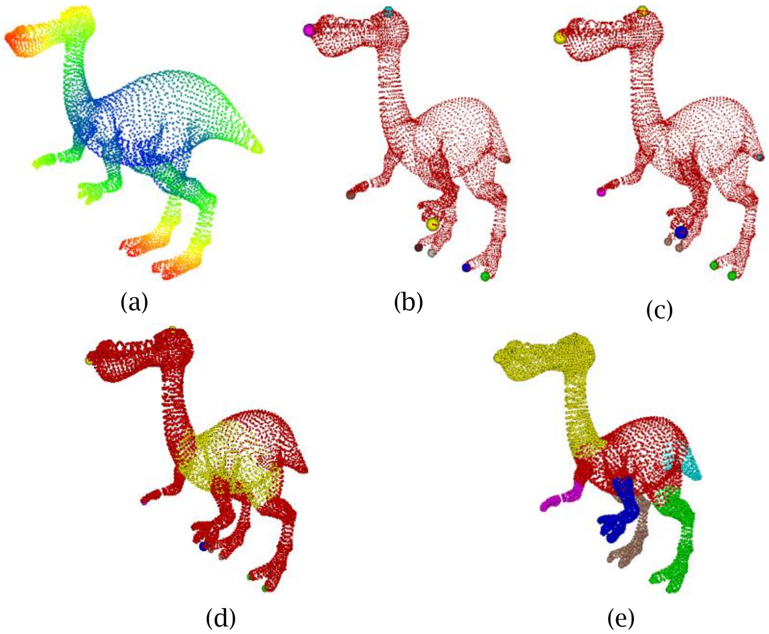

Fig. 1: (a) Visualization of the protrusion function calculated on the point cloud, blue to red depict small to large values. (b) The salient points of the object are illustrated with different colors. (c) The salient points are grouped according to the protrusion they belong to. (d) The core of the object is illustrated in yellow. (e) The final segmentation of the point cloud is illustrated with each part having a different color.

Methodology:

The work of Agathos et al. [1] segments semantically the triangle mesh of an object containing protrusions. Its extension to point clouds that capture the geometry of an object with protrusions, is not trivial and its constituent steps to achieve a successful segmentation require methods and algorithms suitable for point clouds. The development of such algorithms is a challenging task since, contrarily to polygonal meshes, there is no definition of any kind of surface representing the object. Furthermore, all the proposed algorithms are not using any other information, like normal vectors, except the raw coordinates of the point cloud.

The steps to achieve semantic segmentation to a point cloud are: (i) Computation of the protrusion function on the point cloud, (ii) Computation of the salient points, (iii) Grouping of the salient points, (iv) Boundaries extraction, (v) Semantic parts extraction with the aid of graph-cut. A brief description of all steps is given in the sequel.

Computation of the protrusion function

The protrusion function $f_{prot} : \mathbb{R}^3 \to \mathbb{R}$ is defined for each vertex p of the point cloud P as:

$$f_{prot}(p) = \sum_{p' \in P} g(p, p') \qquad (1)$$

where $g(p, p')$ the geodesic distance of point p to point p'. The geodesic distance is found by applying the Dijkstra algorithm [10] using the Elliptic Gabriel Graph to provide connectivity between the raw points [7]. The protrusion function receives small values at the center of the object and high values at its extrema (protrusions), see Fig. 1(a).

The problem with the computation of the protrusion function is its high complexity since it has to be calculated for each vertex of the point cloud making it intractable to point clouds consisting of thousands of points. In [1] an approximation to the protrusion function is proposed and it is followed also in this work. Specifically, the point cloud is divided into non overlapping geodesic patches. These patches are created by applying the Dijkstra algorithm constrained with a distance threshold on randomly chosen points of the point cloud. Each geodesic patch has a center b and an area, $area(b)$, computed by finding the mean-radius of the 1-ring neighborhood of each point contained in the patch and taking the sum of the disk areas defined by these radii. The k-ring neighborhood of a point is defined with the aid of the Elliptic Gabriel Graph. By considering the set B of all the centers of the geodesic patches the protrusion function can be approximated as:

$$f_{prot}(p) \simeq \sum_{b \in B} g(p,b) area(b) \tag{2}$$

Computation and grouping of the salient points

The points where the protrusion function receives a local maximum are the salient points. In order to find the local maxima, for each vertex p of the point cloud P a local neighborhood N_p needs to be defined. This neighborhood can be (i) a k-ring neighborhood, or (ii) a geodesic neighborhood calculated by constraining the Dijkstra algorithm with a threshold distance. In this work the k-ring is used. In Fig. 1(b) the salient points (local maxima) of the model are shown with different colors. As can be observed, for each semantic part of the object more than one salient point may be found. In order to distinguish the salient points according to the part they belong to they need to be clustered (grouped). Assuming the set of salient points is defined as $S = \{s_i, i = 1...N_s\}$, a value T_s is defined:

$$T_s = \frac{\sum_{i=1}^{Ns-1} \sum_{j=i+1}^{Ns} g(s_i, s_j)}{N_s(N_s - 1)} \tag{3}$$

The salient points are clustered in such a way so as each cluster contains the salient points whose geodesic distance with each other is smaller than the threshold value T_s. The geodesic distance between the salient points is also calculated with the Dijkstra algorithm. Fig. 1(c) shows this clustering of the salient points. The *representative* salient point of each cluster is the one with the highest value of protrusion function.

Computation of the core of the object

The core of the object are the vertices of the point cloud that belong to the center of the object and its boundaries define the boundaries of the protrusions. Assuming that the representative salient points are defined as $\hat{S} = \{\hat{s}_i, i = 1...N_c\}$, where N_c the number of clusters then the geodesic paths between the salient points can be found. Again, the geodesic paths can be found with the Dijkstra algorithm. The vertices of the point cloud are inserted in a priority queue having as key the protrusion function value. The vertices are then extracted from the priority queue one by one until a sufficient portion of the geodesic paths is covered. For further details of the algorithm applied the reader is referred to [1] (Fig. 4). These extracted vertices constitute the core of the object and are utilized for the establishment of the protrusions boundaries, Fig. 1(d).

Boundary extraction of the protrusions

The segmentation boundaries are defined as the boundaries between the core of the object and the established protrusions. It is assumed that a representative salient point represents a protrusion (part) whose segmentation boundary needs to be found. Let this representative be denoted as s_{rep}, and let $C_{\min}$ be the nearest point of the core to s_{rep}. Next, a distance function $D_{s_{rep}}(p)$ is defined for each point p of the point cloud as the distance from the representative s_{rep} to point p. This distance function is defined for each connected pair of points (u,v) in the Elliptic Gabriel Graph as

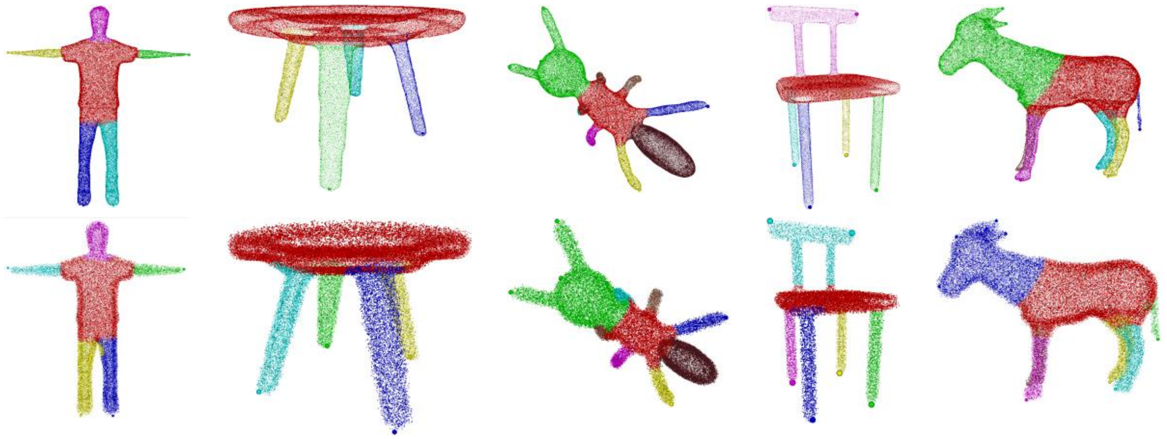

Fig. 2 Semantic segmentation of various point clouds. First row without noise. Second row with noise (1% random displacement of the points from their initial positions).

Model/Method	*Human* *62590 points*	*Table* *74623 points*	*Ant* *43454 points*	*Chair* *64548 points*	*Donkey* *73287 points*
Point-based	2.935	1.225	2.625	1.316	3.470
Polygon-based	3.686	1.316	3.099	1.225	5.354

Tab. 1 Execution time in secs for the segmentation of the models of Fig. 2.

$$dist(u,v) = \delta \frac{\left\| u - v \right\|}{average\_dist} + (1-\delta) \frac{\left| f_{prot}(u) - f_{prot}(v) \right|}{average\_prot} \tag{4}$$

where $\delta \in [0,1]$, $average\_dist$ and $average\_prot$ the average value of all pair differences in length and protrusion respectively. In order to find the value of $D_{s_{rep}}(p)$ for every point p the Dijkstra algorithm is applied on the Elliptic Gabriel Graph using for edge weights the distance function defined in equation (4). The region defined by the points with distance value in the range $D_{s_{rep}} \in [(1-d_1)D_{s_{rep}}(C_{\min}),(1-d_2)D_{s_{rep}}(C_{\min})]$ is considered, because the segmentation boundary is most likely to consist of points belonging in this interval. In order to find it, thresholding is applied on the distance function to create successive region rings and their successive area ratio is examined. The area of the region ring is found by using the 1-ring neighborhood of the points belonging in the region. When this ratio is above a value then the perimeter that is examined is defined as the segmentation boundary.

Similarly to [1], after the boundary is found a graph-cut algorithm is applied to refine it so that it passes through the concavities of the object. The flow network graph is defined by the elliptic Gabriel graph and the edge-capacities required for the graph cut algorithm are defined on the midpoints of each of its edges. Specifically, for each edge $e = (u,v)$ of the elliptic Gabriel graph the middle point p_e is found by averaging the vertices of the point cloud corresponding to its endpoints. On this point the Gaussian curvature is employed. After boundary refinement the part is extracted with region growing from the representative salient point, accumulating points until the segmentation boundary is reached.

In Fig.2, the semantic segmentation of various kinds of objects is illustrated with and without noise. The first row presents the results of the proposed method applied to several articulated objects.

The second row is using the same point clouds contaminated with noise to simulate the input from a real-world scanning procedure. It can be observed that the segmentation is consistent in both cases. In fact, the proposed method succeeded in providing meaningful semantic segmentation in all tested objects using as input the raw scanning data. In Tab.1 the execution time of the proposed algorithm is shown in comparison with the execution time of the mesh-based approach of [1] performed on the triangulation of the point clouds of the models of Fig.2. It can be observed that the proposed method performs competitively to the mesh-based methodology.

Conclusions:

In this work, a new methodology for the semantic segmentation of point clouds has been presented based on the main workflow implemented for polygonal meshes by Agathos et al. [1]. New algorithms have been devised for each step of the methodology which are applicable to point clouds. With the new methodology a point cloud of an articulated object can be successfully segmented to semantic parts without the need to produce a polygonal mesh which can be a tedious process in cases of complicated topology and/or noise. Also, the new methodology does not need the normals of the vertices of the point cloud making it more resistant to noise. This work has an advantage over machine learning methods because it can segment a vast variety of articulated objects, as Fig. 2 shows, without the need of preprocessing as in neural networks. The limitation of this methodology is that it needs sufficient sampling so as to create proper connectivity between the vertices of the point cloud.

Acknowledgement:

This research has been financially supported by the Initiative "Regional Excellence", under the Operational Program “Competitiveness, Entrepreneurship & Innovation” (EPAnEK), which is co-funded by Greece and European Union (MIS 5047046).

References:

[1] Agathos, A.; Pratikakis, I., Perantonis, J. S.; Sapidis S.N.: Protrusion-oriented 3D mesh segmentation, Vis. Comput, 26, 2010, 63–81. https://doi.org/10.1007/s00371-009-0383-8.

[2] Chehata, N.; Guo, L.; Mallet, C.: Airborne lidar feature selection for urban classification using random forests, International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, 2009, 207–212.

[3] Kaick, O.V.; Fish, N.; Kleiman, Y.; Asafi, S.; Cohen-Or, D.: Shape Segmentation by Approximate Convexity Analysis, Transactions on Graphics, 34, 2014, 1-11. https://doi.org/10.1145/2611811.

[4] Lalonde, J.-F.; Unnikrishnan, R.; Vandapel, N.; Hebert, M.: Scale Selection for Classification of Point-Sampled 3-D Surfaces, Fifth International Conference on 3D Digital Imaging and Modeling (3DIM 2005), 13-16 June 2005, Ottawa, Ontario, Canada. IEEE Computer Society. 285–292. https://doi.org/10.1109/3DIM.2005.71.

[5] Li, Z.; Zhang, L.; Tong, X.; Yuebin. B.D.; Zhang, W.L.; Zhang, Z.; et al.: A Three-Step Approach for TLS Point Cloud Classification, IEEE Trans. Geosci. Remote. Sens., 54, 2016, 5412–5424. https://doi.org/10.1109/TGRS.2016.2564501.

[6] Maturana, D.; Scherer, A.S.: VoxNet: A 3D Convolutional Neural Network for real-time object recognition, IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2015, Hamburg, Germany, September 28 - October 2, 2015, 922–928. https://doi.org/10.1109/IROS.2015.7353481.

[7] Park, J.C.; Shin, H.; Choi, K B.: Elliptic Gabriel graph for finding neighbors in a point set and its application to normal vector estimation, Comput. Aided Des., 38, 2006, 619–626. https://doi.org/10.1016/j.cad.2006.02.008.

[8] Qi, C.R.; Mo S.H.; Guibas J.L.: PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation, CVPR, 2016. http://arxiv.org/abs/1612.00593.

[9] Su, H.; Maji S.; Kalogerakis E.; Learned-Miller G.E.: Multi-view Convolutional Neural Networks for 3D Shape Recognition, ICCV, 2015. http://arxiv.org/abs/1505.00880.

[10] Tenenbaum, J.B.; de Silva, V.; Langford C.J.: A Global Geometric Framework for Nonlinear Dimensionality Reduction, Science, 290, 2000, 2319-2323. https://doi.org/10.1126/science.290.5500.2319.

[11] Wang, Y.; Sun Y.; Liu, Z.; Sarma E.S.; Bronstein M.M.; Solomon M.J.: Dynamic Graph CNN for Learning on Point Clouds, ACM Trans. Graph., 38(5), 2019, 1–12. https://doi.org/10.1145/3326362.