Graph cut based mesh segmentation using feature points and geodesic distance

Lei Liu, Yun Sheng, Guixu Zhang
Department of Computer Science
East China Normal University
Shanghai, 200241, China
Email:ysheng@cs.ecnu.edu.cn

Hassan Ugail
Centre for Visual Computing
University of Bradford
Bradford BD7 1DP, UK

***Abstract*—Both prominent feature points and geodesic distance are key factors for mesh segmentation. With these two factors, this paper proposes a graph cut based mesh segmentation method. The mesh is first preprocessed by Laplacian smoothing. According to the Gaussian curvature, candidate feature points are then selected by a predefined threshold. With DBSCAN (Density-Based Spatial Clustering of Application with Noise), the selected candidate points are separated into some clusters, and the points with the maximum curvature in every cluster are regarded as the final feature points. We label these feature points, and regard the faces in the mesh as nodes for graph cut. Our energy function is constructed by utilizing the ratio between the geodesic distance and the Euclidean distance of vertex pairs of the mesh. The final segmentation result is obtained by minimizing the energy function using graph cut. The proposed algorithm is pose-invariant and can robustly segment the mesh into different parts in line with the selected feature points.**

I. Introduction

In cyberworlds, objects are usually represented by mesh models. These mesh models may be too large to manipulate, and thus need to be segmented before manoeuver. Mesh segmentation is to dispart a mesh into a series of sub-meshes which have their own practical meanings. Mesh segmentation is a significant ingredient in the researches of 3D modeling, texture mapping, 3D shape retrieval, mesh compression, and many other graphical applications. A majority of segmentation methods are based on the psychological minima rule in psychology, a theory that defines how human perception might decompose an object into its constituent parts [1]. This rule states that the boundaries for the constituent parts are along lines of the negative minima curvature. Many early mesh segmentation methods that have built on their image segmentation conterparts conform with the minima rule, such as 3D snake [2][3], watershed segmentation [4][5], Mean Shift [6], *etc*.

Many mesh segmentation approaches have been proposed in the past few decades, and these approaches can be divided into two categories, surface-type segmentation and part-type segmentation. Surface-type segmentation uses some geometry features such as curvature to dispart the mesh into surface patches, while part-type segmentation focuses on partitioning the mesh into meaningful components. The segmentation methods based on clustering and fitting primitives belong to the surface-type. Shlafma *et al*. utilized K-Means to segment meshes [7]. Fuzzy clustering was used by Katz *et al*. for mesh decomposition [8]. Yamauchi *et al*. also have presented a scheme for mesh segmentation based on norm clustering with Mean Shift [6]. A hierarchical segmentation method based on fitting primitives is proposed by Macro *et al*. [9] for mesh segmentation. In their method, the triangles faces are fitted by planes, spheres and cylinders by the parameters predefined.

Part-type segmentation methods are various. Some built on image segmentation. Lee [2][3] generalized 2D Snake proposed by Kass [10] to 3D, and demonstrated its feasibility in 3D mesh segmentation. The watershed segmentation method was also extended to 3D by Mangan *et al*. [4] and Page *et al* [5]. Other methods came with new ideas. For example, Katz's segmentation method employed feature points and core extraction [1]. Shapira *et al*. proposed the concept of shape diameter function with graph cut to segment meshes [11], which can also be used in skeletonization.

Graph cut was emerged from the max-flow and min-cut problem. It has been used for image segmentation in computer vision for a while. Graph cut can also be used for mesh segmentation. For example, Shapir *et al*. made use of the shape diameter function for their graph cut based segmentation. In their method, the Gaussian mixture model was used to model the distribution of the shape diameter function. From the result of the Gaussian mixture model, every vertex has a probability of belonging to a certain cluster. However, only using the shape diameter function cannot partition some models into sub-meshes clearly, because different parts may have similar diameter function values.

In this paper, we present a new mesh segmentation method based on graph cut using feature points and geodesic distance. The main procedures of our method are shown in Fig. 1. The mesh is first preprocessed using Laplacian smoothing. Then, we select candidate feature points from the mesh vertices with a predefined threshold value of curvature. The final feature points are obtained by making a DBSCAN (Density-Based Spatial Clustering of Application with Noise) on the candidate feature points. The graph is constructed by regrading the mesh faces as nodes and the feature points as labels. Graph cut is executed to obtain the final segmentation results.

The remaining of this paper is arranged as follows. Section II introduces the preprocessing of mesh. Section III is devoted

978-1-4673-9403-1/15 $31.00 © 2015 IEEE
DOI 10.1109/CW.2015.31

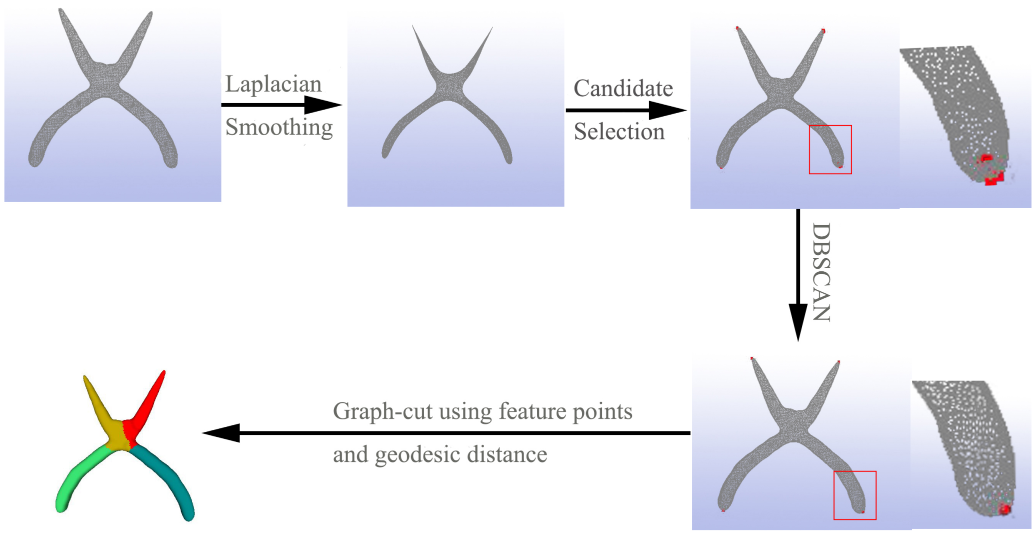


Fig. 1. The flow diagram of our method

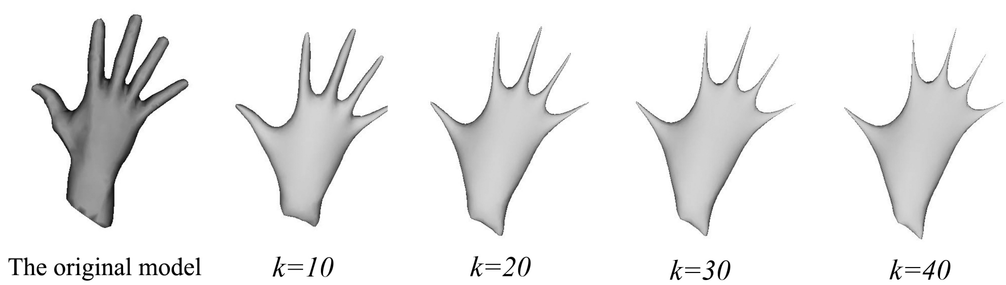


Fig. 2. Laplacian smoothing

to detection of feature points from candidates. In section IV, we present the graph construction for graph cut, and the experimental results are shown in section V. Section VI concludes this paper.

II. Mesh preprocessing

Laplacian smoothing is employed to preprocess the meshes before subsequent operations, because the original meshes have abundant geometric details which may challenge the detection of feature points. Mesh smoothing has been studied for a long period, and there has been a variety of smoothing methods. Among them, Laplacian smoothing is the simplest smoothing method, which is described as follows. A vertex v_i can be smoothed out as

$$v_i \leftarrow \frac{1}{|N_i|} \sum_{v_j \in N_i} v_j \tag{1}$$

where N_i is the neighborhood set of vertex v_i. If we add the coordinate value of the original vertex with a weight to control the proportion, then the smoothing equation becomes

$$v_i \leftarrow (1-\lambda) v_i + \frac{\lambda}{|N_i|} \sum_{v_j \in N_i} v_j \tag{2}$$

Laplacian smoothing has a flaw that the mesh will be shrunken after smoothing. As shown in Fig. 2, the hand model shrinks after iterations of Laplacian smoothing. To avoid shrinking, Taubin has proposed an improved smoothing method called $\lambda | \mu$ smoothing [12]. In this paper, the purpose of smoothing is to discover feature points. The shrinking will not affect the topological structure of meshes, and the feature points still need to be mapped back to the original mesh. So, we employ Laplacian smoothing for preprocessing. Our experiments show that 40 iterations of Laplacian smoothing is enough for detection of feature points in this paper.

III. Feature points detection

Feature points can be used in various applications, such as mesh deformation, mesh retrieval, parameterization, and mesh segmentation. Katz et al. have proposed a method for feature points detection based on geodesic distance. In their method, for a vertex v with its neighborhood set N_v, if v is a feature point, it must satisfy the condition

$$\sum_{v_i \in M} Gd(v, v_i) > \sum_{v_i \in M, v_n \in N_v} Gd(v_n, v_i) \tag{3}$$

where Gd represents the geodesic distance between two vertices. In order to obtain the local minimal value of geodesic distance, geodesic distances of a number of vertex pairs need to be computed. The computation of geodesic distances is, however, time consuming.

In this paper, we propose to detect feature points based on the Gaussian curvature. When the mesh is smoothed, the minor details are removed, with only prominent parts left. As a result, the curvature values of the vertices in the prominent parts are higher than others. The curvatures of all vertices in the smoothed mesh need to be normalized firstly.

$$k_i = \frac{k_i - \overline{k}}{k_{max} - k_{min}} \tag{4}$$

where $\overline{k}$ is the curvature mean, k_{max} and k_{min} are the maximal and minimal values of curvature respectively.

We set a threshold value as T. When $k_i < T$, we regard the vertex v_i as a candidate feature point. The threshold value should depend on the character of the mesh. The more complex the mesh, the lower the threshold value, and vice versa. There is a problem for the candidate feature points selected by the threshold value, where there may exist more than one feature point on every prominent part, as shown in Fig. 3. However, we need only one feature point for each prominent part. Thus, the candidates need further culling. Since the candidate feature points of one prominent part are concentrated, the Euclidean distance between them are small. When clustering the candidate feature points, the points close to each other can be regarded as one cluster.

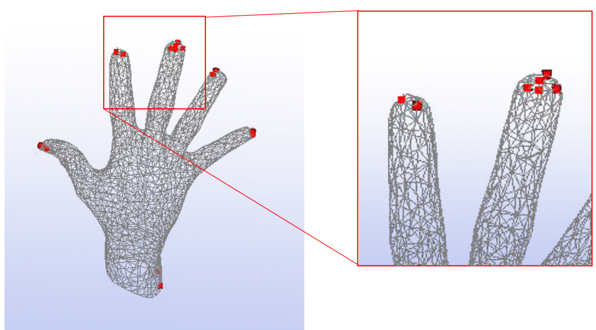


Fig. 3. Candidate feature points

K-Means cluster is the most widely used method for clustering. However, it does not suit for this circumstance because the value of K need to be assigned beforehand. We cannot foreknow how many parts a mesh will be segmented into. Futhermore, K-Means is sensitive to noise. So we resort to another clustering method in this paper. We use DBSCAN [13] proposed by Matin *et al.* for clustering. As for a dataset $D = \{d_1, d_2, d_3, \ldots, d_n\}$, DBSCAN can separate it into k clusters with noise data. The clustering result is marked with an array m, where the noise data are set to -1.

$$m_i = \begin{cases} j & \text{if } d_i \text{ belongs to cluster } j \\ -1 & \text{if } d_i \text{ is subject to noise data} \end{cases} \tag{5}$$

There are two values to be set beforehand, the neighborhood radius ε and the core point threshold T_0. The procedure of DBSCAN is as follows

Algorithm 1 DBSCAN

```
set ε = a, m_k = 0, n = 0, T_0 = b
repeat
  take out an unvisited data d_i from the dataset D;
  compute the neighborhood density ds_i from the circular
  area with the radius of ε;
  if ds_i < T_0 then
    mark the data d_i as noise, m_i = −1;
  else
    let m_i = n and mark it as core point;
    repeat
      take out a data point d_j from the neighborhood of
      d_i;
      compute the neighborhood density ds_j of d_j;
      if ds_j < T_0 then
        mark the data d_i as core point, m_j = n;
      end if
    until all the neighborhood points have been visited
  end if
  n = n + 1;
until every element in the dataset has been visited.
```

DBSCAN only needs to predefine two parameters, the neighborhood radius ε and the core point threshold value T_0. They can be either set for every mesh or reused with same values for different meshes. The coordinates of mesh vertices are normalized before processing in this paper. So the different meshes can use a common radius. Distance used for clustering is the Euclidean distance. When the candidate feature points are seperated into several clusters, we pick up the vertex with a maximum Gaussian curvature in every cluster as the final feature point.

Fig. 4 shows the final feature points detected by our method. In the results, there are six feature points for the hand model which are the tips on each finger and wrist. There is a feature point on the wrist because it become a cusp when this model is smoothed by Laplacian. The right model in Fig. 4 has four feature points which are successfully located on the prominent positions.

IV. Graph construction

We regard the feature points as labels, and construct a graph from the mesh with the faces as nodes. Then the mesh can readily be segmented by graph cut. Given an undirected weighted graph G, there are two terminal nodes in the graph (the number of terminal nodes can be more than two, here we take two as example). A cut denoted as C is defined by the

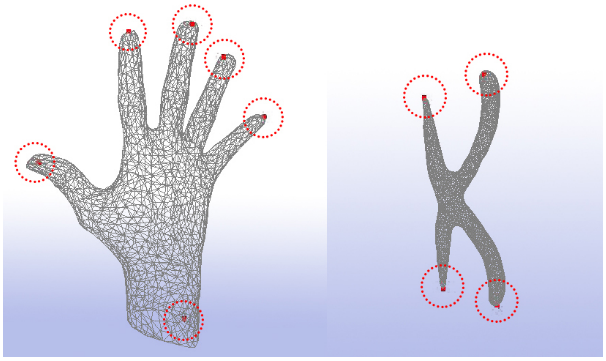


Fig. 4. Finally selected feature points

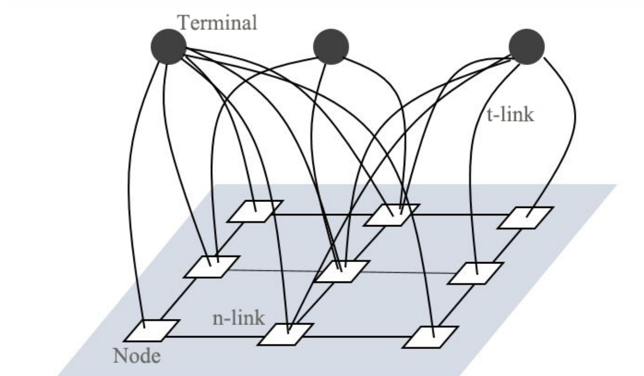


Fig. 5. Graph cut

two terminal nodes is separated, and satisfies the condition that an arbitrary subset of the cut can not separate the two terminal nodes. Because every edge in the graph has a weight, we consider the sum of the weights in the cut C as the cost of cut, marking it as $|C|$. The problem left behind is how to find a cut with a minimum cost.

We denote the terminal nodes as t and s. The edges connecting the terminal node and common nodes are t-link, and the edges connecting the common nodes are n-link. When we consider these edges as pipes, finding the max flow from s to t is a MaxFlow problem. The Ford-Fulkerson theory [14] shows that when the n-link edges be situated in the max flow, and when they can separate the two terminal nodes, the cut consisting of these edges is the minimum cut. Thus the MaxFlow and MinCut problems are equivalent.

The energy of graph cut can be written as

$$E(f) = \sum_{\{p,q\}\in N} V_{p,q}(y_p, y_q) + \sum_{p\in P} D_p(y_p) \tag{6}$$

where $V_{p,q}(y_p, y_q)$ is the smoothing energy, and $D_p(y_p)$ is the data energy that measures the degree of p belonging to the label y.

Fig. 5 shows a graph with three terminal nodes. It is notable that the terminal node is not connected to every common node, and that the weights of these edges can be set to zero in practice. When the number of terminal nodes is more than two, the MinCut problem becomes a NP hard problem. Because solving the global optimal is time consuming, many researchers compute the local optimal rather than the global optimal. However, in some situations, the difference between the global optimal and local optimal is considerable. We assign the label value to every node for constituting a label array. As for a label array Y, if the energy is a local optimum, with a neighborhood array Y', there exists the correlation

$$E(Y) \leq E(Y') \tag{7}$$

The neighborhood array Y' can be computed by changing the label value of a vertex. This process is called standard moves. If the label array is in local optima, the standard moves can not decrease the energy, the result may fall into the local optima. Considering drawbacks of the standard moves, Boykov *et al*. proposed two move strategies [15][16], α-expansion and α-β-swap. Given a label α, a move from a label array Y to a new label array Y' is called an α-expansion if $Y_\alpha \subset Y'_\alpha$ and $Y'_l \subset f_l$ for any label $l \neq \alpha, \beta$. This means that an α-expansion move allows any set of labels to change to α. Given a pair of labels α and β, a move from a label array f to a new label array Y' is called an α-β-swap if $Y_l = Y'_l$ for any label $l \neq \alpha, \beta$. This means that the difference between Y and Y' is that some labels α in y become labels β in Y', and some labels β in Y become labels α in Y'.

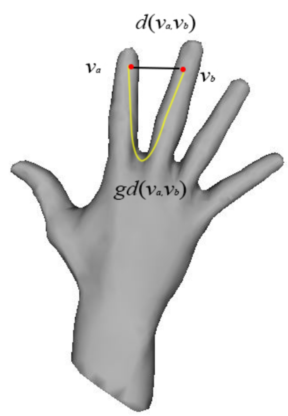


Fig. 6. Geodesic distance

It can be observed from many mesh models that the geodesic distance between vertices located at different prominent parts can be much larger than the Euclidean distance. As shown in Fig. 6, we take out a vertex from the index finger denoted as v_a, and a vertex from the middle finger denoted as v_b. The yellow line is the geodesic distance $gd(v_a, v_b)$, and the black line is the Euclidean distance $d_(v_a, v_b)$. Let the ratio between them be ϵ, and $\epsilon = \frac{gd(v_a,v_b)}{d(v_a,v_b)} > 1$. When the

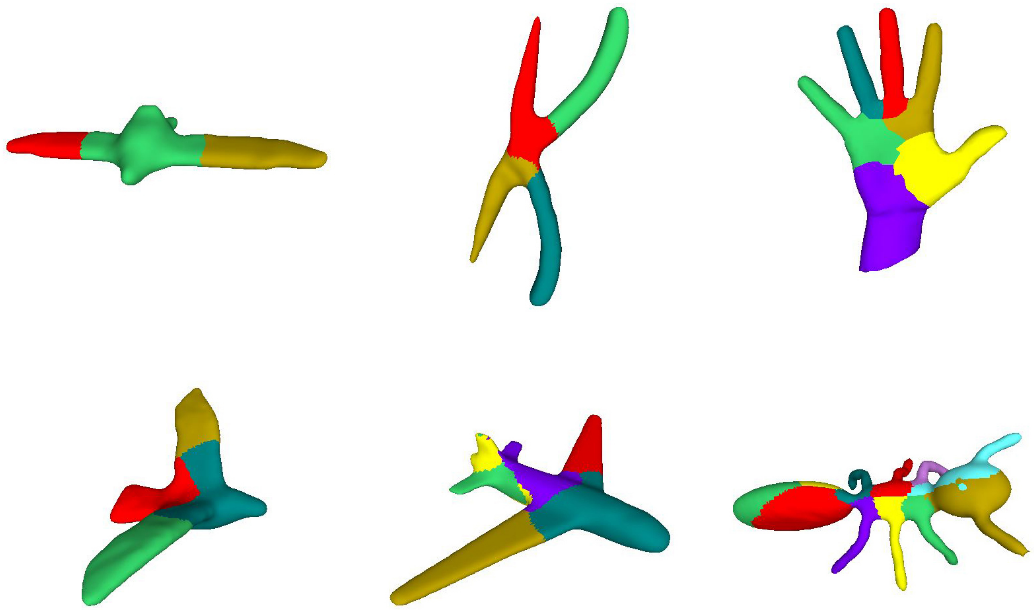


Fig. 7. Segmentation results using our method

two vertices come from the same prominent part, the ratio will approach 1; when they come from different prominent parts, the ratio will be much larger. According to this fact, we construct the energy term by this ratio. For the sake of segmentation, the faces of the mesh are regarded as the nodes of the graph. The sum of the ratios between three vertices in the face and the feature point is conducted as data energy. The geodesic distance between vertices is computed using Fast Marching [17].

$$D_f(i) = \sum_{v_j \in f} \frac{gd(v_j, v_i)}{d(v_j, v_i)} \tag{8}$$

where f denotes a face in the mesh, v_j is one of the vertices in face f, and v_i is a feature point. The smoothing energy is defined by the neighborhood information of faces. Iterating the α-expansion and α-β-swap moves obtains the global optimum of the energy. When the energy reaches the optimum, the values in the mark array are the segmentation result.

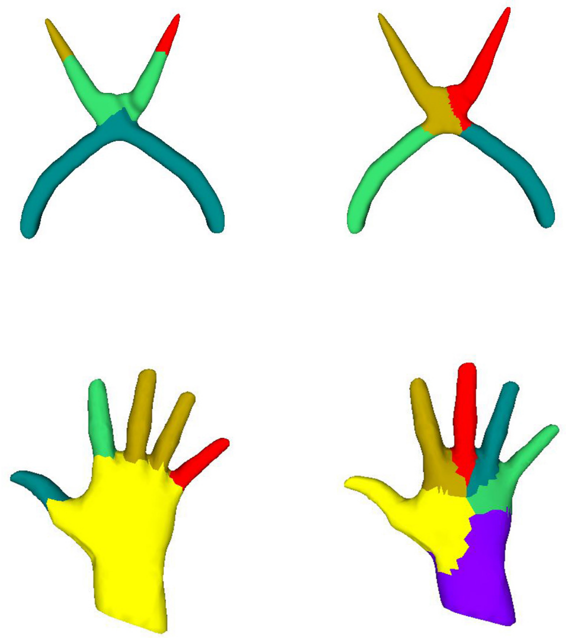
V. Experimental results

Fig. 7 shows some mesh segmentation results using our method, where different colors represent different parts segmented. For the model of pliers (the second model in Fig. 7), it is partitioned into four parts which conform with our human perception well. For the two bird models, their wings can be separated from the body. And the remaining three models can also be partitioned into several parts well.

The shape diameter function based segmentation method proposed by Shapira et al. also employs graph cut. We compare our segmentation results with those produced by the shape diameter function based method in Fig. 8. Because of the cusps, the central and the handle parts of the plier have different shape diameter function values, resulting in an undesired segmentation result with Shapira's method as shown in Fig. 8. But in our method, the feature points are used for graph cut, and partition the faces based on the geodesic distance. Our method partitions the pliers into four parts as shown in Fig. 8. The same trouble occurs in the hand model with Shapira's method, the middle finger and ring finger can not be separated. While the fingers can be separated clearly in our result. On the other hand, in our segmentation results, the boundaries between different parts are not located along the negative minima curvature. This is because we focus on disparting different branches in the mesh rather than finding the best segmentation boundaries.

VI. Conclusion

In this paper, we introduce a new segmentation method for meshes based on graph cut using feature points and geodesic distance. The feature points are detected by the Gaussian curvature and DBSCAN. An energy function for graph cut is defined by the ratio between the geodesic distance and Euclidean distance. The idea of our method is to separate the model into branches. Compared with Shapira's method, ours can partition the mesh into sub-parts more clearly. Our method may benefit applications in cyberworlds. For instance, animation in cyberworlds usually demands that different branches of meshes move in different trails. Our method can segment



(a)Shapira's results (b)Our results

Fig. 8. Results comparison

meshes in cyberworlds according to branches, and thus, is available for use in cyberworlds.

VII. Acknowledgment

This work is supported by National Natural Science Foundation of China (61202291) and the Fundamental Research Funds for the Central Universities, China.

References

[1] Sagi Katz, George Leifman, and Ayellet Tal. Mesh segmentation using feature point and core extraction. *The Visual Computer*, 21(8-10):649–658, 2005.

[2] Yunjin Lee and Seungyong Lee. Geometric snakes for triangular meshes. In *Computer Graphics Forum*, volume 21, pages 229–238. Wiley Online Library, 2002.

[3] Yunjin Lee, Seungyong Lee, Ariel Shamir, Daniel Cohen-Or, and H-P Seidel. Intelligent mesh scissoring using 3d snakes. In *Computer Graphics and Applications, 2004. PG 2004. Proceedings. 12th Pacific Conference on*, pages 279–287. IEEE, 2004.

[4] Alan P Mangan and Ross T Whitaker. Partitioning 3d surface meshes using watershed segmentation. *Visualization and Computer Graphics, IEEE Transactions on*, 5(4):308–321, 1999.

[5] AF Koschan. Perception-based 3d triangle mesh segmentation using fast marching watersheds. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–27. IEEE, 2003.

[6] Hitoshi Yamauchi, Seungyong Lee, Yunjin Lee, Yutaka Ohtake, Alexander Belyaev, and H-P Seidel. Feature sensitive mesh segmentation with mean shift. In *Shape Modeling and Applications, 2005 International Conference*, pages 236–243. IEEE, 2005.

[7] Shymon Shlafman, Ayellet Tal, and Sagi Katz. Metamorphosis of polyhedral surfaces using decomposition. In *Computer Graphics Forum*, volume 21, pages 219–228. Wiley Online Library, 2002.

[8] Sagi Katz and Ayellet Tal. *Hierarchical mesh decomposition using fuzzy clustering and cuts*, volume 22. ACM, 2003.

[9] Marco Attene, Bianca Falcidieno, and Michela Spagnuolo. Hierarchical mesh segmentation based on fitting primitives. *The Visual Computer*, 22(3):181–193, 2006.

[10] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International journal of computer vision*, 1(4):321–331, 1988.

[11] Lior Shapira, Ariel Shamir, and Daniel Cohen-Or. Consistent mesh partitioning and skeletonisation using the shape diameter function. *The Visual Computer*, 24(4):249–259, 2008.

[12] G TaubinÝ. Geometric signal processing on polygonal meshes. *Eurographics State of the Art Reports*, 2000.

[13] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[14] LR Ford and Delbert Ray Fulkerson. *Flows in networks*, volume 1962. Princeton Princeton University Press, 1962.

[15] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(11):1222–1239, 2001.

[16] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(9):1124–1137, 2004.

[17] Ron Kimmel and James A Sethian. Computing geodesic paths on manifolds. *Proceedings of the National Academy of Sciences*, 95(15):8431–8435, 1998.