International Journal of Computational Vision and Biomechanics
Vol.1 No. 2 (July-December, 2015)

Hierarchical 3D Segmentation Using Connected Face Structure

Sébastien Delest, Romuald Boné & Hubert Cardot

Université Franc¸ois Rabelais de Tours, Laboratoire Informatique, 64 avenue Jean Portalis, 37200 Tours, France

This paper describes a new approach for watershed segmentation on triangular mesh. Common watershed approaches use descending or flooding processes on a connected vertex structure. Watershed transformation is not related to a structure in particular: it only requires connected elements and a height function. This method is widely used on 2D, 3D images as well as on 3D meshes. Here, a connected face structure is implemented and adapted to the segmentation process. A connected face structure offers a different kind of curvature information and neighborhood. The waterfall algorithm based on the minimum spanning tree is used to compute the merging. Several segmentation schemes are built from the waterfall and they can easily be browsed by the user.

Keyword: *3D mesh segmentation, Connected face structure, Watershed, Waterfall*

1. INTRODUCTION

Polygonal meshes offer an efficient representation of 3D surfaces, in particular triangular meshes, which are used in many applications. In this paper, we deal with triangular meshes; yet, our method can be adapted to other types of polygonal meshes. Mesh segmentation has applications for major problems in visualization and modeling, metamorphosis, compression, 3D shape retrieval, collision detection, texture mapping, etc. The shape of the models is significant and can lead to different segmentation approaches depending on whether the problem concerns natural shapes or mechanical parts. Mesh segmentation methods are mainly classified into two groups, the patch-type and the part-type, the former being related to the creation of patches that are uniform with respect to some properties (e.g., curvature, distance to a fitting primitive, size or convexity, etc.) [7, 10, 36] and the latter aiming at identifying parts that correspond to relevant features of the shape [22, 15, 47].

In the following, we propose a patch-type segmentation based on watershed transformation. Our approach is bottom-up oriented and flooding is performed from the face curvature. In 2D images, most watershed algorithms are concerned with the structure of connected pixels and gray levels (or gradient magnitude) as a height function. On 3D meshes, the structure of connected vertices and vertex curvature are generally used to compute the watershed transformation [25, 33, 43, 1, 6]. The proposed approaches use a connected face structure (not the dual graph of the mesh, see figure 2) and the face curvature to create the first partition of the mesh. The watershed transformation produces an over-segmentation; techniques such as filtering, using markers or hierarchical processes, are usually used so as to avoid this problem. Here, we propose two hierarchical processes to merge regions of the watershed partition. The former involves successive merging according to the watershed depth order; the latter uses the waterfall scheme to create several segmentation levels. The remainder of the paper proceeds as follows: the next section deals with related works about the patch-type segmentation.

Section 3 addresses the curvature calculation from connected vertex and connected face structures. Section 4 presents the watershed transformation algorithm. Section 5 explains the methods to limit over-segmentation. Finally, results and discussions are presented in section 6.

2. RELATED WORK

A formulation of boundary mesh segmentation has been presented by Shamir in [38]. The mesh segmentation can be seen as an optimization problem where a three dimensional mesh is defined as a tuple $\{V, E, F\}$ of vertices V, edges E and faces F. Mesh segmentation algorithms partition the faces, the vertices or the edges of the mesh. Segmentation methods generally fall into two classes: region-based and boundary-based approaches. The latter use special features of local properties as candidate locations for boundaries; regions are deduced from these located boundaries. Region based approaches look for areas with similar properties that define the regions; the boundaries are deduced from them. The possible approximate solutions for patch-type segmentation are region growing [42, 46, 45, 28, 20, 29], watershed approaches [25, 43, 34, 31, 6], hierarchical [10, 36, 39, 2], iterative [40, 37, 16, 7, 32, 18, 17] or spectral clustering [22, 44]. Some methods are manual or semi-manual.

They involve techniques such as graph cut [13], shortest path algorithms [9], simplification [24] or snakes [14, 21, 23].

In this paper, we focus on hierarchical approaches which make it possible to incrementally build and merge the regions until all regions have been merged or when a threshold has been reached. Hierarchical clustering can start when each face is its own cluster or when a primary partition has been built from a pre-processing step such as region growing or watershed transformation. The dual graph of the partition (see Fig. 1) can be used to represent the region's connectivity and neighborhood. Each edge of the dual graph is assigned a cost for merging. Garland *et al.* [10] proposed a hierarchical face clustering using L_2 distance and orientation norms from representative planes as a planarity measurement. This measurement is formulated using the quadric error metric. The regions' shape can be controlled by adding a compactness heuristic which improves the regularity of the clusters. A similar segmentation scheme is proposed in [36] but additional tests are performed before merging two clusters to incorporate topology constraints, like the fact that, for example, each clustered patch must be homeomorphic to a disk. In the post processing step, smooth boundaries between the charts are created by calculating the shortest constrained path. Sheffer [39] worked on the dual graph of the mesh and contracted edges in respect to topological costs including size, shape, curvature and more. Attene *et al.* [2] proposed a hierarchical segmentation based on fitting primitives such as a plane, a sphere and a cylinder.

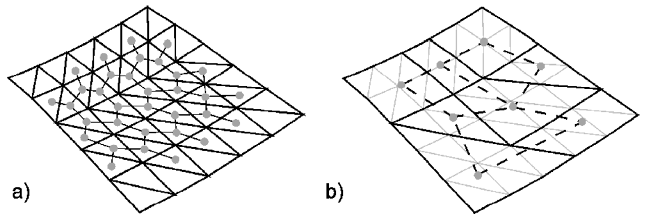

Figure 1: Dual graph of the mesh (a), dual graph of the partition (b)

The hierarchical segmentation involves an initial partition composed by faces or regions (built from a pre-process). The partitioning of 3D surface meshes has been explored by Mangan and Whitaker [25] who merge adjacent regions according to the watershed depth. Page *et al.* [31] developed a fast watershed algorithm using the minima rule [12] to compute the curvature and compared their approach to Mangan and Whitaker's method. They showed the advantages and drawbacks of using the minima rule, that is to say using negative curvature minima to define boundaries. The drawbacks, which appear as an ambiguity in the minima rule theory, are discussed in [41].

Most of the 3D mesh watershed algorithms derive from 2D methods. The main algorithms are described in [35]. The initial partition computed by watershed transformation can be considered as the first step of a hierarchical process such as the dynamics [30] or the waterfall [4]. The waterfall appears as a powerful tool to merge regions and to create several segmentation levels. Marcotegui and Beucher [26] proposed an implementation of waterfalls based on graphs. This process helps to create several segmentation levels quickly. The user can easily browse the different levels in order to choose the most suitable one for his application.

In this paper, we deal with watershed transformation and hierarchical segmentation processes on 3D meshes. The first hierarchical process provides a solution based on edge contraction of the dual graph in order to merge regions built from the watershed transformation. The second process corresponds to the waterfall algorithm on 3D meshes adapted from [26, 8]. Section 5 analyses the advantages and drawbacks of the two methods.

3. CURVATURE CALCULATION

To perform the watershed transformation, a structure associating a neighborhood and a criterion to each element is needed. In 2D images, a pixel has 8 neighbor pixels. In the case of polygonal meshes, vertex and faces may have a variable connectivity, as it is shown in Figure 2.

For 2D images, the criterion can correspond to a gray level, a gradient magnitude or a distance function relative to boundaries in an image based on brightness, color and texture cues [11]. For polygonal meshes, we associate this criterion to the vertex or face curvature. Several methods are proposed to calculate the vertex characteristics. Mangan and Whitaker have pointed out the covariance matrix efficiency in [25] to compute the curvature. Gaussian, mean and principal curvatures have been detailed in [27] and a recent approach [19] has been proposed to estimate local

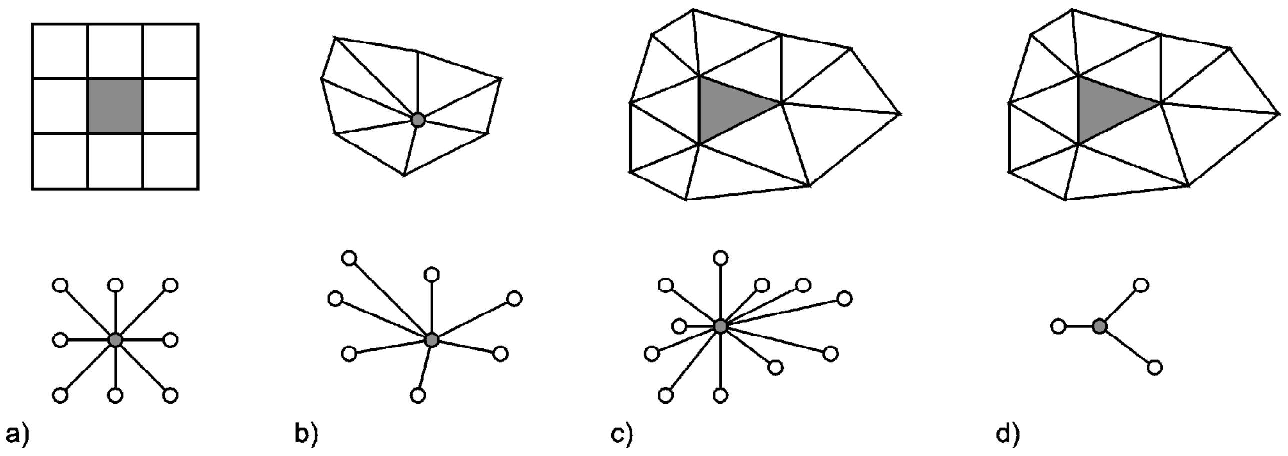

Figure 2: Neighborhood relationships for (a) pixels, (b) vertices, (c) triangles and (d) dual graph

characteristics of the surface by means of integral quantities. In order to compare the vertex and face connection approaches, we use the covariance matrix, which offers relevant curvature information. This method is less sensitive to noise than approaches using dihedral angle measurement only.

The covariance matrix is given by the variance and covariance in all three directions:

$$\sigma_{uu}^2 = \frac{1}{N}\sum_{t=0}^{N}(u_t - \overline{u})^2 \quad (1)$$

$$\sigma_{uv}^2 = \frac{1}{N}\sum_{t=0}^{N}(u_t - \overline{u})(v_t - \overline{v}) \quad (2)$$

$$u \in (x, y, z\} \quad v \in (x, y, z\} \quad (3)$$

where N represents the number of triangles associated with this vertex (or face) and $\{x_t, y_t, z_t\}$ are the components of the normal for triangle t. Curvature C is defined by the Euclidean norm of the covariance matrix M. In the case of a connected face structure, components of the central triangle normal can be used instead of $\overline{u}$ and $\overline{v}$.

$$C = \| M \| \;\; with \;\; M = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{bmatrix} \quad (4)$$

In the case of a connected face structure, there are more neighbors than in the case of a connected vertex structure; this allows us to easily differentiate the significant curvature, with a view to reducing noise. Neighbor triangles do not have the same type of connection; some are connected to the face by one of the edges while others are connected by one of the vertex as it is shown in Figure 2. Keeping all triangles connected gives the best curvature information and watershed computation while keeping a maximum of three triangles connected by one of the edges only, ensures the best time calculation but a less accurate curvature and watershed.

4. WATERSHED TRANSFORMATION

This section describes the approach used in this watershed algorithm and the features of the implementation. The bottom-up approach of watershed transformation is a segmentation technique which simulates water rising on the image relief from the local minima. Several kinds of height function can be used (height function considering gray level intensity is shown in figure 3).

The flooding starts from local minima and incrementally fills the basins until they connect to its neighbors. A watershed is generated where basin collision occurs (Figure 4).

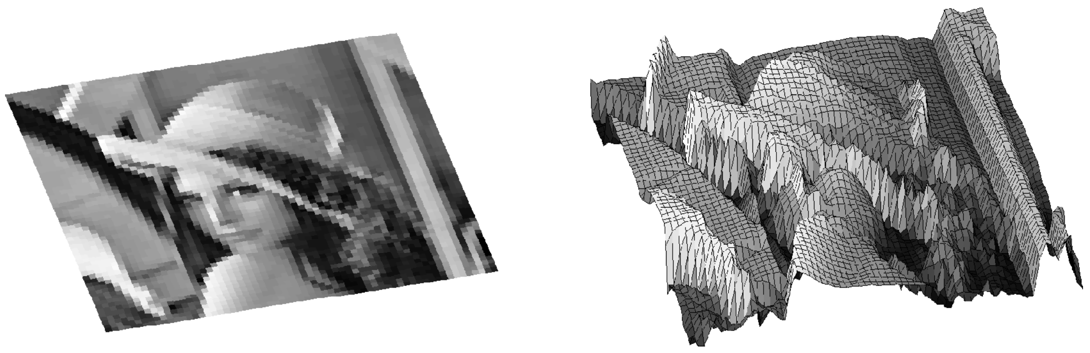

Figure 3: Lena image map generated from its gray level intensities

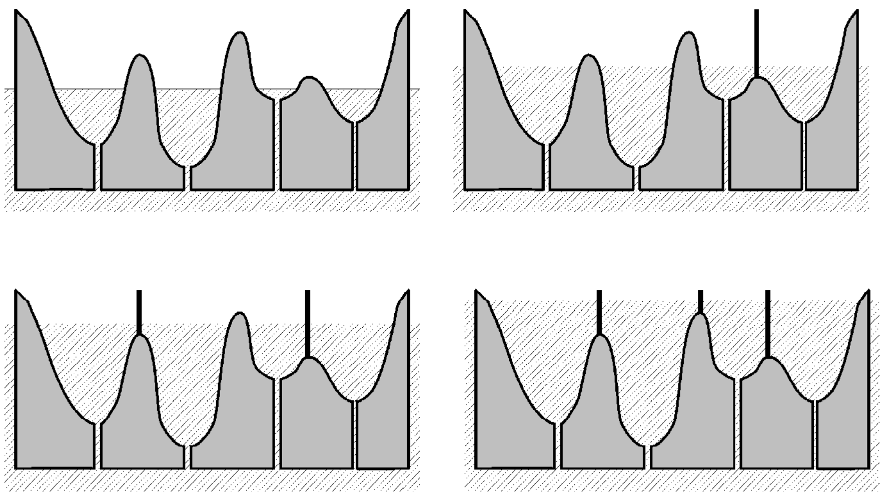

Figure 4: A one dimensional example of watershed transformation. The different levels of flooding. Flooding start from minima; basin collisions generate a watershed

The use of hierarchical queues is one of the best solutions to build a watershed on 2D images quickly. The hierarchical queue [5] is made up of several FIFO queues and each queue corresponds to a level (gray level or gradient magnitude on 2D images, curvature or roughness measurement on 3D meshes, etc.). Queues are sorted per level and a queue can be unstacked only when the previous queues have been emptied (Figure 5).

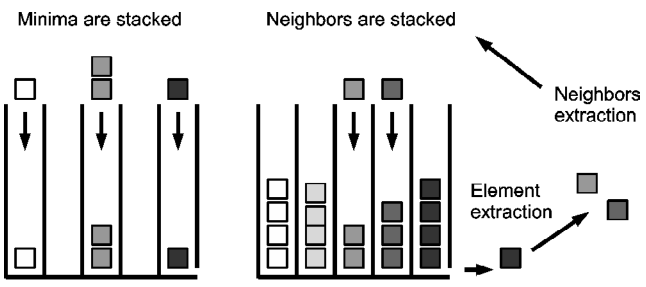

Figure 5: Watershed transformation based on hierarchical queue

The process uses the following steps. First, the minima are stacked into their corresponding queue. Each minimum (element or plateau) obtains a label. The first element of the first queue is unstacked, then its non conflicting neighbors are stacked in their corresponding queues and receive their root element label. The conflicting elements are labelled as watershed. In figure 6, the results are obtained by calculating the hierarchical queue watershed algorithm of 3D models with the two approaches. Section 6 offers a comparison of these methods.

5. HIERARCHICAL SEGMENTATION

As it can be seen in figure 6, the output of watershed transformation quite badly suffers from over-segmentation. To avoid this problem, two methods are used: forcing specific regions with markers and hierarchical segmentation. The former is very accurate but supposes that the characteristics of the object are known, the latter offers the choice between several levels of segmentation. The general case is considered in this article and two hierarchical segmentation algorithms are proposed in the following.

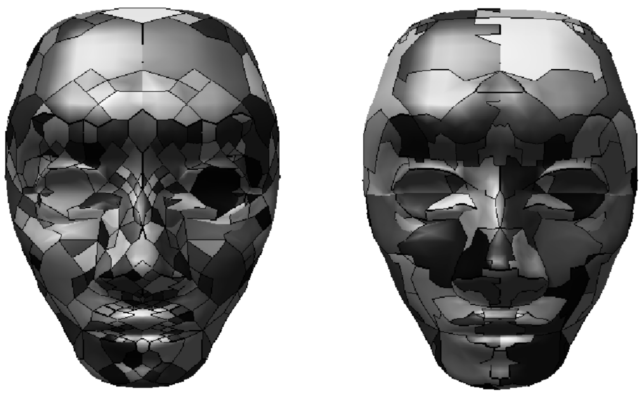

Figure 6: Hierarchical queue watershed transformation on connected vertex structure on the left and on connected face structure on the right

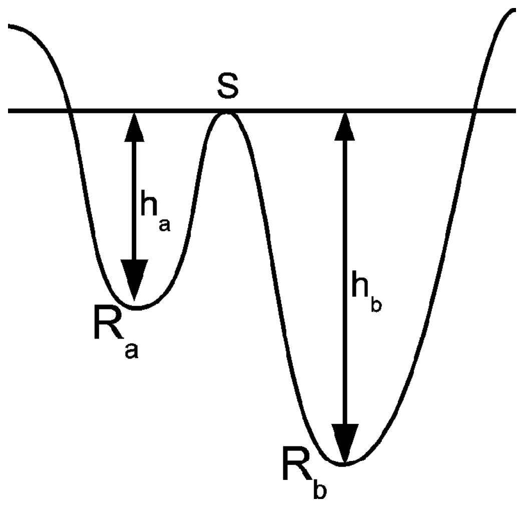

Figure 7: Watershed depth

The result of the watershed is used as an input partition of the hierarchical segmentation algorithm. This partition is composed of regions separated by watersheds (see figure 1b). Our algorithms are based on the contraction of dual graph edges: two neighbor catchment basins are considered as an edge with a depth value (Figure 7). Depth P of the watershed saddle point S between regions a and b corresponds to the difference between the saddle point curvature and the minimum curvature of the regions a and b:

$$\begin{aligned} P(S_i(R_a, R_b)) &= C(S_i) - \min(R_a, R_b) \\ &= \max(h_a, h_b) \end{aligned}$$

The edges are composed of the element (vertex or face) list of the watershed. Each region contains its element list, its minimum curvature and its watershed list. A tree node is used to represent the region merging. For each region, a node is created from the region information. The merging can now be computed.

5.1 Successive Merging

The first algorithm corresponds to the successive contractions of the dual graph edges in respect to edge value order. The value of each edge corresponds to the watershed depth. The region merging from the watershed depth has been introduced by Mangan and Whitaker in [25]; here we propose a new formulation based on the minimum spanning tree (MST). In order to allow a quick browsing of the different segmentation levels, each node created by an edge contraction receives the following information:

- the node child list
- a new label
- the external edge watershed list
- the internal edge watershed list

Each merging involves the watershed depth calculation of the new region and a new sorting of watershed depth list. When nodes are merged, their shared watersheds are removed, however, they must be stored in the node as internal watershed because they exist at an inferior level of the tree (an internal watershed belongs to a single node only); then label attribution is quickly carried out. Figure 8 shows the partition computed from the watershed transformation and the minimum spanning tree built from the successive dual graph edge contractions. Figure 10 provides the region minima value and the saddle point curvature of the watersheds on the left and the corresponding dual graph of the partition on the right. The connectivity of the eight regions is represented by the dual graph. Merging can be done considering only one or several dual edge contractions at the same time. If two minimum edges have the same value, the one whose the saddle point curvature is the lowest can be chosen first. In this example, we allow multiple contractions when several minimum edges are detected.

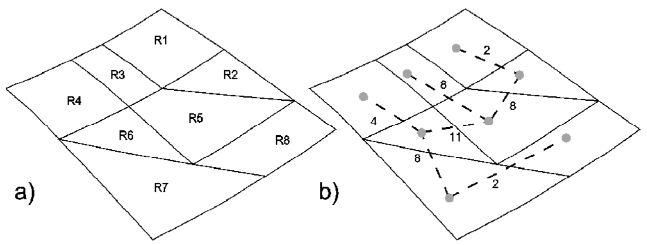

Figure 8: Partition created by the watershed transformation (a) and minimum spanning tree built from the successive region merging (b)

Our method involves the dynamic construction of the minimum spanning tree. At the first iteration, edge with a value of 2 are contracted which allows the merging of regions $\{R_1, R_2\}$ in R_9 and regions $\{R_7, R_8\}$ in R_{10}. These edges are the lowest and the first ones to be contracted. The creation of new regions requires the computation of new minima and edge value. Region R_9's minimum corresponds to $\min(R_1, R_2)$. The old value $(10 - 3 = 7)$ becomes $(10 - 2 = 8)$. One or several dual graph edges can be removed during the creation of a region. At the second iteration, the edge with a value of 4 is contracted.

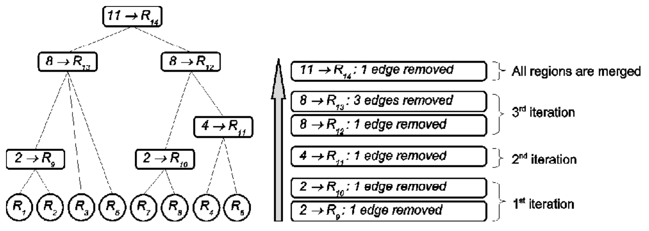

Figure 9: Regions merging tree. Node information corresponds to edge values and to the merging order from node {2 → R_9}.

Region R_{11} is created and the values $(13 - 4 = 9)$ and $(10 - 4 = 6)$ become $(13 - 2 = 11)$ and $(10 - 2 = 8)$ respectively. The process stops when only one region remains. Figure 9 shows the different steps of the merging. Flooding always follows the minimum height path. The minimum spanning tree is built from the successive edge contractions (see figure 8b) and cannot be obtained directly as it can be seen by comparing edge values on the MST and the initial partition. New partitions are obtained by contracting all MST edges whose values are below a threshold. The MST appears as a very condensed way to store the information and allows very efficient implementation of hierarchical segmentation approaches.

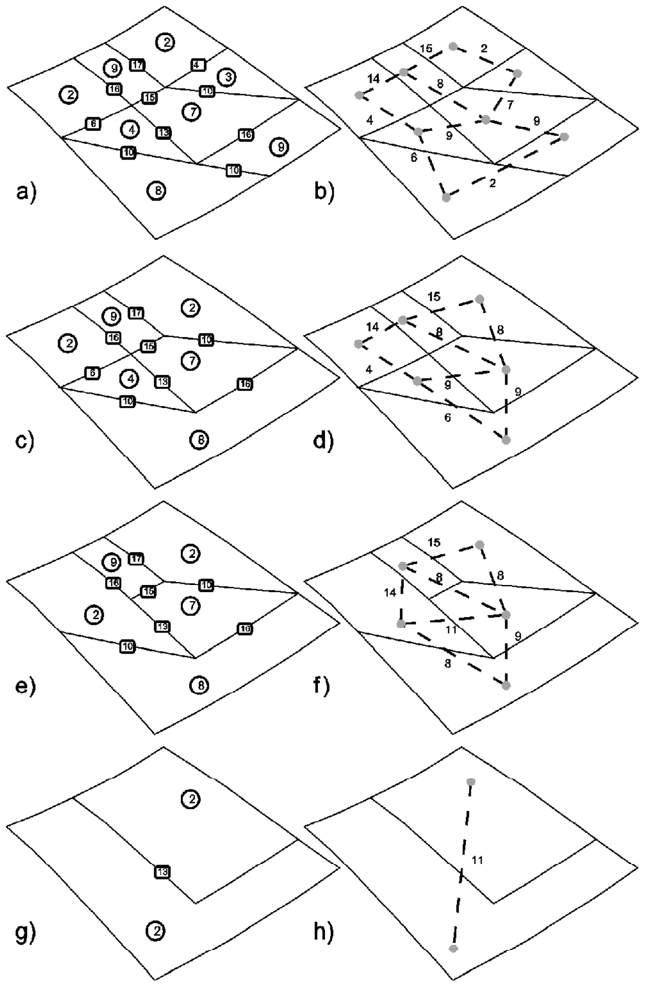

Figure 10: The successive contractions of the dual graph edges. Region minima and saddle point curvatures on the left, value (watershed depth) of the dual graph edges on the right.

5.2 The Waterfall

The Waterfall has been introduced by Beucher in [4] and corresponds to a hierarchical approach selecting among all the contours of the watersheds those which are completely surrounded by higher contours. A simplified partition is obtained by removing these contours. The process may be iterated until a single region covers the whole mesh. The Waterfall algorithm presented in [26] uses the minimum spanning tree of the partition. All local minimum edges of the MST obtain a different label. Other MST edges are browsed in the increasing order and are labeled according to the label of the lowest neighbor edge. An edge is not labelised if more than one neighbor edge is assigned to a different label. Figure 11 shows the first waterfall iteration. The MST edges with values of 2 and 4 are considered as local minima and obtain a different label; thereby, the partition of the first waterfall iteration possesses three regions. Two other MST edges can be contracted at this iteration. At the second iteration, only one local minimum is found. This involves the merging of all remaining regions. The Waterfall technique helps to extract the main regions at each iteration which considerably reduces the search for the best segmentation level as it is shown in figure 12.

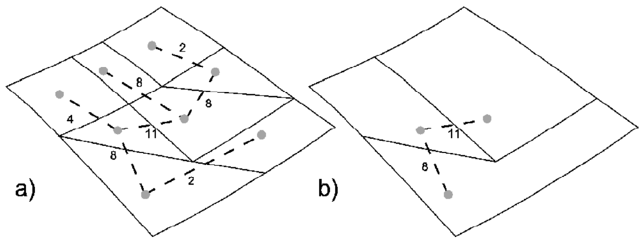

Figure 11: Minimum spanning tree (a) and the first waterfall iteration (b)

The waterfall process finds only one segmentation level in this example whereas the successive merging process builds three levels at least. The successive merging process is interesting in the case of low resolution models but is not appropriate for high resolution models. The main advantage of the waterfall process is to create some partitions containing the main regions at different levels. Only few segmentation levels are proposed which eases the search for the best one. The waterfall process can merge too many regions between two iterations but additional function can be used to browse intermediary merging between two waterfall iterations.

6. DISCUSSION AND RESULTS

We have presented two ways to characterize the mesh surface by calculating the vertex curvature and the face curvature. The same watershed process is used to compute the flooding on a connected vertex structure or a connected face structure. We provide two merging methods relative to the tradeoff flexibility / efficiency. Most of the segmentation methods are sensitive to the shape of the 3D model. Some methods

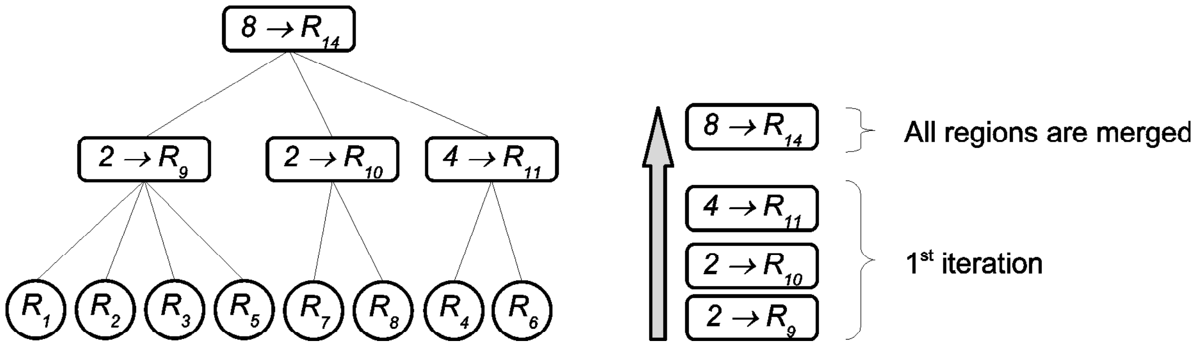

Figure 12: Regions merging tree built by the waterfall process. Node information corresponds to local minimum edge values and to the region number.

are adapted to mechanical parts whereas others help segmenting smooth patches on natural shapes. The faces curvature provides a good surface characterization in either case and avoids focusing on the *n*-neighborhood of vertices. Figure 13 shows several models segmented at the best segmentation level and a comparison of waterfall segmentation carried out on connected face structure and on connected vertex structure.

Faces are nearly twice as many as vertices and they have a larger neighborhood; then, connected face structure gives a proportionately important computation time. In a noise filtering way, a greater number of faces makes it possible to easily differentiate the significant curvature. Connected face structures lead to longer watershed computation but also to a lower number of regions and a better characterization of the different segmentation levels as shown in figure 13 and table 1. Each model's segmentation feature appears on table 1. All models have been segmented by the waterfall process which each time gave the best segmentation results. Pre-processing and post-processing are not addressed here; they constitute complementary approaches to improve segmentation. Attene et al. [3] recently proposed a comparative study of the latest mesh segmentation methods; they defined several criteria in order to compare the five methods they dealt with. In the following, we describe these criteria and discuss the efficiency of our method.

Type of segmentation: the models presented here often appear in the mesh segmentation literature. They are used by the two main segmentation families which are the part-type segmentation and the patch-type segmentation. Among the different kinds of segmentation, our approach focuses on patch-type segmentation of 3D meshes from curvature information.

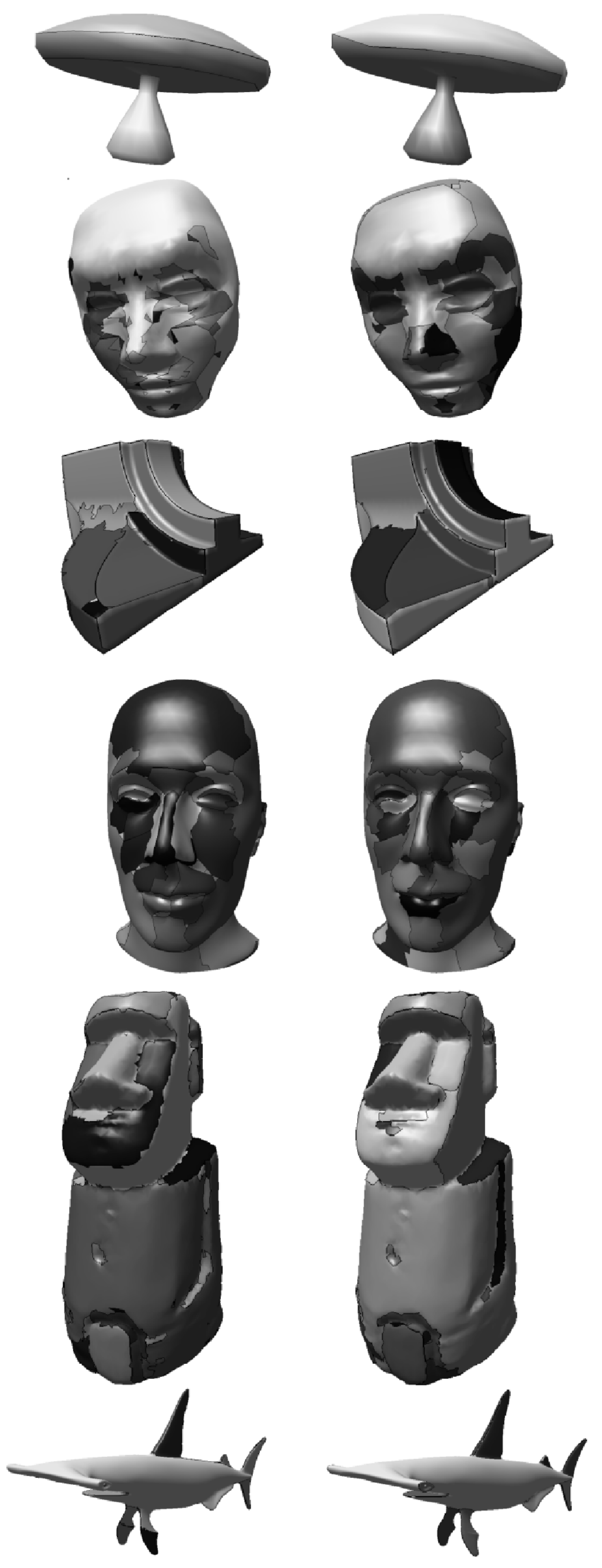

Figure 13: **Comparison of the two approaches (connected vertex structure on the left, connected face structure on the right)**

Extracting the correct patches: as explained in [3], defining the correct component of a given model is impossible. The right segmentation depends on the application, the viewer's perspective and the knowledge of the world. In most papers, the correctness of the segmentation is only obtained by looking at the images themselves. Our method uses the watershed transformation and local minima in order to create patches. The boundaries and patch minima are used to compute the watershed depth of each boundary. The minimum spanning tree of the partition can be built from successive contraction of dual graph edges as explained in section 5.1. The waterfall algorithm based on the MST helps to generate several segmentation schemes and find the most suitable segmentation level of the application very quickly.

Boundaries quality: the boundaries are made in respect to the scheme of the watershed process and regions meet at their highest curvature area. Our merging process successively removes some boundaries at each segmentation level. Additional process could be used to control some geometric properties such as boundary smoothness, boundary length and its location along concave features, etc.

Hierarchical segmentation: some methods open the possibility of browsing several segmentation levels. In [2], the iterative clustering is computed several times depending on the maximum number of clusters or on a threshold error. The user may interactively move a slider which sets the desired number of clusters (or the threshold error). We propose here a slider which makes it possible to select the different partitions computed from the waterfall.

Sensitivity to the pose: in the case of part-type methods, the sensitivity to the pose is important. A retrieval method can involve a segmentation process which must decompose the same parts whatever the pose of the 3D model [15]. In the case of patch-type methods, the geometric properties of the model can be very different depending on the pose. This criterion, defined in [3], is not concerned directly with the patch-type segmentation.

Calculation time: all performance measurements for the segmentation algorithm were made on a 2.8 Ghz Intel Pentium IV system. The running time of the watershed transformation and the merging process vary between 1 ms for model Mushroom and 72.4 s for model Shark. This time span strongly depends on the number of vertices and faces. Table 1 shows the number of regions created by the watershed transformation and the reduction obtained with the chosen level of segmentation. We estimate the best level of segmentation in terms of patch-segmentation and we can see that this level can easily be found because of the low number of levels to browse. Table 1 shows the number of regions for each model and each type of structure after the watershed transformation; it gives the last number level (where all regions are merged) and the chosen segmentation level.

Table 1
Summary of the relative features of the different approaches applied to the models in Figure 13

Models	*Type*	*Merging level / Max*	*Number of regions / max*	*Computation time (ms)*
Mushroom	Vertex	2 / 3	4 / 27	1
(226 vertices, 448 faces)	Face	1 / 2	3 / 15	2
Head	Vertex	5 / 10	48 / 423	53
(2363 vertices, 4640 faces)	Face	2 / 4	33 / 130	146
Fandisk	Vertex	3 / 5	26 / 367	738
(6475 vertices, 12946 faces)	Face	3 / 5	16 / 237	2738
Mannequin	Vertex	2 / 9	81 / 325	788
(6743 vertices, 13424 faces)	Face	2 / 8	67 / 259	3028
Moai	Vertex	6 / 13	91 / 1109	2423
(10002 vertices, 20000 faces)	Face	4 / 10	50 / 573	7178
Shark	Vertex	10 / 19	48 / 747	2239
(10054 vertices, 20104 faces)	Face	5 / 10	30 / 377	7240

Control parameter: a slider is provided to set the segmentation level rather than a threshold. All segmentation schemes can be stored efficiently in the merging tree presented in figure 12.

7. CONCLUSIONS

Our approach uses the face curvature information to compute the 3D segmentation. We have presented a comparison of segmentations with different structures and showed that connected face structure entailed the best segmentation. We use the watershed transformation and the waterfall algorithm based on the minimum spanning tree to create several partition schemes and open the possibility to browse the different segmentation levels quickly. Future studies will involve the adaptation of our method to specific applications. We are studying the efficiency of different criteria such as the distance to crest line, vertex and face curvature, roughness measurement, etc. in order to build a more specific height function for the watershed transformation and the waterfall.

ACKNOWLEDGEMENTS

We are grateful to AIM@SHAPE (www.aimatshape.net) and 3D Cafe (www.3dcafe.com) for making their models available online.

REFERENCES

[1] P. N. Atkar, A. Greenfield, D. C. Conner, H. Choset, and A. A. Rizzi. Hierarchical segmentation of surfaces embedded in R3 for auto-body painting. In ICRA '05: International Conference on Robotics and Automation, pages 572–577, Barcelona, Spain, April (2005).

[2] M. Attene, B. Falcidieno, and M. Spagnuolo. Hierarchical mesh segmentation based on fitting primitives. The Visual Computer: *International Journal of Computer Graphics*, 22(3): 181–193, (2006).

[3] M. Attene, S. Katz, M. Mortara, G. Patané, M. Spagnuolo, and A. Tal. Mesh segmentation-a comparative study. In SMI '06: Proceedings of the IEEE International Conference on Shape Modeling and Applications, pages 14–25, Washington, DC, USA, (2006). IEEE Computer Society.

[4] S. Beucher. Watershed, hierarchical segmentation and waterfall algorithm. In J. Serra and P. Soille, editors, *ISMM '94: Proceedings of the 2nd International Symposium on Mathematical Morphology and its Applications to Image Processing*, pages 69–76. Kluwer Academic Publishers, 1994.

[5] S. Beucher and F. Meyer. The morphological approach of segmentation: The watershed transformation. In E. R. Dougherty, editor, Mathematical Morphology in Image Processing, chapter 12, pages 433–481. New York (1993).

[6] L. Chen and N. D. Georganas. An efficient and robust algorithm for 3D mesh segmentation. Multimedia Tools and Applications, 29(2):109–125, (2006).

[7] D. Cohen-Steiner, P. Alliez, and M. Desbrun. Variational shape approximation. ACM Transactions on Graphics, 23(3): 905-914, (2004).

[8] S. Delest, R. Boné, and H. Cardot. Fast segmentation of triangular meshes using waterfall. In VIIP '06: International Conference on Visualisalization, Imaging and Image Processing, pages 308–312, Palma De Mallorca, Spain, August 2006.

[9] T. Funkhouser, M. Kazhdan, P. Shilane, P. Min, W. Kiefer, A. Tal, S. Rusinkiewicz, and D. P. Dobkin. Modeling by example. *ACM Transactions on Graphics*, 23(3): 652–663, (2004).

[10] M. Garland, A. Willmott, and P. S. Heckbert. Hierarchical face clustering on polygonal surfaces. In *SI3D '01: Proceedings of the 2001 Symposium on Interactive 3D graphics*, pages 49–58, New York, NY, USA, 2001. ACM Press.

[11] A. Hanbury and B. Marcotegui. Waterfall segmentation of complex scenes. In *ACCV '06: Proceedings of the 7th Asian Conference on Computer Vision*, volume 3851, pages 888–897, Hyderabad, India, January 2006.

[12] D. D. Hoffman and W. A. Richards. Parts of recognition. *Readings in Computer Vision: Issues, Problems, Principles and Paradigms*, pages 227–242, (1987).

[13] Z. Ji, L. Liu, Z. Chen, and G. Wang. Easy mesh cutting. *Computer Graphics Forum*, 25(3): 283–291, (2006).

[14] M. Jung and H. Kim. Snaking Across 3D Meshes. In *PG '04: Proceedings of the 12th Pacific Conference on Computer Graphics and Applications*, pages 87–93, Washington, DC, USA, 2004. IEEE Computer Society.

[15] S. Katz, G. Leifman, and A. Tal. Mesh segmentation using feature point and core extraction. *The Visual Computer*, 21(8-10): 649–658, (2005).

[16] S. Katz and A. Tal. Hierarchical mesh decomposition using fuzzy clustering and cuts. *ACM Transactions on Graphics*, 22(3): 954–961, July 2003.

[17] V. Krayevoy and A. Sheffer. Variational, meaningful shape decomposition. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Sketches*, page 50, New York, NY, USA, 2006. ACM Press.

[18] Y. K. Lai, Q. Y. Zhou, S. M. Hu, and R. R. Martin. Feature sensitive mesh segmentation. In *SPM '06: Proceedings of the 2006 ACM Symposium on Solid and Physical Modeling*, pages 17–25, New York, NY, USA, 2006. ACM Press.

[19] Y. K. Lai, Q. Y. Zhou, S. M. Hu, J. Wallner, and H. Pottmann. Robust feature classification and editing. *IEEE Transactions on Visualization and Computer Graphics*, 13(1): 34–45, (2007).

[20] G. Lavoué, F. Dupont, and A. Baskurt. A new CAD mesh segmentation method, based on curvature tensor analysis. *Computer-Aided Design*, 37(10): 975–987, (2005).

[21] Y. Lee, S. Lee, A. Shamir, D. Cohen-Or, and H. P. Seidel. Intelligent mesh scissoring using 3D snakes. In *PG '04: Proceedings of the 12th Pacific Conference on Computer Graphics and Applications*, pages 279–287, Washington, DC, USA, 2004. IEEE Computer Society.

[22] R. Liu and H. Zhang. Segmentation of 3D meshes through spectral clustering. In *PG '04: Proceedings of the 12th Pacific Conference on Computer Graphics and Applications*, pages 298–305, Washington, DC, USA, 2004. IEEE Computer Society.

[23] S. Liu, R. R. Martin, F. C. Langbein, and P. L. Rosin. Segmenting reliefs on triangle meshes. *In SPM '06: Proceedings of the 2006 ACM Symposium on Solid and physical modeling*, pages 7–16, New York, NY, USA, 2006. ACM Press.

[24] Z. Liyan, L. Shenglan, W. Xi, and Z. Laishui. Segmentation and parametrization of arbitrary polygon meshes. In *GMP '04: Proceedings of the Geometric Modeling and Processing 2004*, page 143, Washington, DC, USA, 2004. IEEE Computer Society.

[25] A. P. Mangan and R. T. Whitaker. Partitioning 3D surface meshes using watershed segmentation. *IEEE Transactions On Visualization and Computer Graphics*, 5(4): 308–321, (1999).

[26] B. Marcotegui and S. Beucher. Fast implementation of waterfall based on graphs. volume 30 of *Computational Imaging and Vision*, pages 177–186. Springer-Verlag, Dordrecht, (2005).

[27] M. Meyer, M. Desbrun, P. Schr¨oder, and A. H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In H. C. Hege and K. Polthier, editors, *Visualization and Mathematics III*, pages 35–57. Springer-Verlag, Heidelberg, (2003).

[28] J. Mitani and H. Suzuki. Making papercraft toys from meshes using strip-based approximate unfolding. *ACM Transactions on Graphics*, 23(3): 259–263, (2004).

[29] T. Mizoguchi, H. Date, S. Kanai, and T. Kishinami. Segmentation of scanned mesh into analytic surfaces based on robust curvature estimation and region growing. *In GMP' 06: Geometric Modeling and Processing*, pages 644–654, Pittsburgh, Pennsylvania, U.S.A., July 2006. Springer Berlin / Heidelberg.

[30] L. Najman and M. Schmitt. Geodesic saliency of watershed contours and hierarchical segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(12): 1163–1173, (1996).

[31] D. L. Page, A. Koschan, and M. A. Abidi. Perception-based 3D triangle mesh segmentation using fast marching watersheds. In *CVPR '03: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2, 27–32, (2003).

[32] G. Peyré and L. D. Cohen. Geodesic remeshing using front propagation. *International Journal of Computer Vision*, 69(1): 145–156, (2006).

[33] S. Pulla, A. Razdan, and G. Farin. Improved curvature estimation for watershed segmentation of 3-dimensional meshes. *IEEE Transactions on Visualization and Computer Graphics* (submitted for publication), 2001.

[34] A. Razdan and M. Bae. A hybrid approach to feature segmentation of triangle meshes. *Computer-Aided Design*, 35(9): 783–789, (2003).

[35] J. B. Roerdink and A. Meijster. The watershed transform: Definitions, algorithms and parallelization techniques. *Fundamenta Informaticae*, 41(1-2): 187–228, (2001).

[36] P. V. Sander, J. Snyder, S. J. Gortler, and H. Hoppe. Texture mapping progressive meshes. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 409–416, New York, NY, USA, (2001). ACM Press.

[37] P. V. Sander, Z. J. Wood, S. J. Gortler, J. Snyder, and H. Hoppe. Multi-chart geometry images. In *SGP '03: Symposium on Geometry Processing*, 146–155, (2003).

[38] A. Shamir. A formulation of boundary mesh segmentation. In *3DPVT '04: Proceedings of the second IEEE International Symposium on 3D Data Processing, Visualization and Transmission*, pages 82–89, Thessaloniki, Greece, (2004).

[39] A. Sheffer. Model simplification for meshing using face clustering. *Computer-Aided Design*, 33(13): 925–934, (2001).

[40] S. Shlafman, A. Tal, and S. Katz. Metamorphosis of polyhedral surfaces using decomposition. *Computer Graphics Forum*, 21(3): 219–228, (2002).

[41] M. Singh, G. D. Seyranian, and D. D. Hoffman. Parsing silhouettes: the short-cut rule. Perception and Psychophysics, 61(4): 636–660, (1999).

[42] O. Sorkine, D. Cohen-Or, R. Goldenthal, and D. Lischinski. Bounded-distortion piecewise mesh parameterization. In *VIS '02: Proceedings of the conference on Visualization*, pages 355–362, Washington, DC, USA, 2002. IEEE Computer Society.

[43] Y. Sun, D. L. Page, J. K. Paik, A. Koschan, and M. A. Abidi. Triangle mesh-based edge detection and its application to surface segmentation and adaptive surface smoothing. In *ICIP '02: Proceedings of the International Conference on Image Processing*, 3, 825–828, Rochester, NY, (2002).

[44] H. Zhang and R. Liu. Mesh segmentation via recursive and visually salient spectral cuts. In *VMV '05: Vision, Modeling and Visualization*, Erlangen, Germany, (2005).

[45] Y. Zhang, A. Koschan, and M. A. Abidi. Superquadric representation of automotive parts applying part decomposition. *Journal of Electronic Imaging, Special Issue on Quality Control by Artificial Vision*, 13(3): 411–417, (2004).

[46] Y. Zhang, J. K. Paik, A. Koschan, M. A. Abidi, and D. Gorsich. A simple and efficient algorithm for part decomposition of 3D triangulated models based on curvature analysis. In *ICIP '02: Proceedings of the International Conference on Image Processing*, 3, Rochester, (2002).

[47] Y. Zhou and Z. Huang. Decomposing polygon meshes by means of critical points. In *MMM'04: Proceedings of the 10th International Multimedia Modelling Conference*, page 187, Washington, DC, USA, 2004. IEEE Computer Society.