Skeleton Extraction of Vertex Sets Lying on Arbitrary Triangulated 3D Meshes

Dimitri Kudelski[1,2,3], Sophie Viseur[1,3], and Jean-Luc Mari[1,2]

[1] Aix-Marseille University, France
jean-luc.mari@univ-amu.fr
http://www.dil.univ-mrs.fr/~mari/
[2] Information and System Science Laboratory (LSIS), UMR CNRS 7296,
[3] European Center for Research and Teaching in Environmental Geoscience (CEREGE)

Abstract. Complex models can be simply described by notions such as skeletons. These robust shape descriptors faithfully characterize the geometry and the topology of an object. Several methods have been developed yet to obtain the skeleton from regular object representations (*e.g.* 2D images or 3D volumes) but only a few attempt to extract the skeleton from unstructured 3D mesh patches. In this article, we extract a skeleton by topological thinning from vertex sets lying on arbitrary triangulated surface meshes in 3D. The key idea comes down to eroding a 2D set located on a discrete 2-manifold. The main difficulty is to transpose the notion of neighborhood from the classical thinning algorithms where the adjacency is constant (*e.g.* 26-adjacency in digital volumes, 8-adjacency in 2D images) to the mesh domain where the neighborhood is variable due to the adjacency of each vertex. Thus we propose a thinning operator dedicated to irregular meshes in order to extract the skeleton of a vertex set. To estimate the robustness of our technique, several tests and an application to the feature line detection are presented as a case-study.

Keywords: surface skeleton extraction, topological thinning, irregular mesh.

1 Introduction

The skeleton is a robust shape descriptor faithfully characterizing the topology and the geometry of an object. This notion is widely used for various applications such as video tracking [4], shape recognition [12], surface sketching [9], and in many other scientific domains. Several techniques have been proposed to extract the skeleton from binary 2D images [13], 3D closed meshes defining a volume [1], or 3D cubic grids [8]. However few have been dedicated to the extraction of skeletons from a binary information located on an arbitrary triangulated mesh. Rössl *et al.* [10] have presented a method in which some mathematical morphology operators have been ported to triangulated meshes. The main interest of this approach is to combine an efficient computation and a simple implementation. However, regarding the operator definitions and the underlying algorithm, several drawbacks have been pointed out which mainly lead to unexpectedly disconnected skeletons [7].

Contributions
In this article, we propose a novel method to extract the skeleton of unstructured mesh patches by a topological thinning process. To figure out the issues of skeletonization of heterogeneous and arbitrary triangulated meshes, we extend the concepts introducted in [10]. The presented approach herein strictly relies on the mesh connectivity to achieve the extraction of the final skeleton. Therefore, for the sake of understanding, the basic method of Rössl *et al.* is described in Section 2 with an assessment of its abilities and drawbacks. Section 3 details the proposed approach and introduces the additional definitions and the novel algorithm. The results of our method including tests on irregular meshes as well as on the performance of the algorithm are shown in Section 4. Finally, an application to the feature line detection is presented in Section 5.

2 Basic Notions and Definitions

2.1 Position of the Problem

Let $\mathcal{S}$ be an arbitrary manifold surface represented by an unstructured mesh patch $\mathcal{M}$ such as $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{T})$. The sets $\mathcal{V}, \mathcal{E}$, and $\mathcal{T}$ correspond, respectively, to the vertices, the edges, and the triangles composing $\mathcal{M}$, the piecewise linear approximation of $\mathcal{S}$. The vertices are denoted by p_i, with $i \in [0; n[$ and $n = |\mathcal{V}|$ being the total number of vertices of $\mathcal{M}$. The neighborhood $\mathcal{N}$ of a vertex p_i is then defined as following:

$$\mathcal{N}(p_i) = \{q_j \mid \exists \text{ a pair } (p_i, q_j) \text{ or } (q_j, p_i) \in \mathcal{E}\}. \tag{1}$$

In such a case, $m_i = |\mathcal{N}(p_i)|$ represents the total number of neighbors of p_i.

Let now consider a binary attribute F on each vertex of $\mathcal{V}$. The set $R \subseteq \mathcal{V}$ is then written as follows:

$$\forall p_i \in R \iff F(p_i) = 1. \tag{2}$$

The attribute F may be defined from beforehand process such as a manual selection, or a thresholding based on geometrical properties (triangle area, principal curvatures, *etc.*). Then, an edge $e = (p, q)$ belongs to R if and only if $p, q \in R$. Similarly, a triangle $t = (p, q, r)$ belongs to R if and only if $p, q, r \in R$.

The main objective is to finally develop a technique to extract the skeleton of the set R by using a topological thinning based on the mesh connectivity.

2.2 The Existing Approach

The skeletonization algorithm introduced by Rössl *et al.* consists in an iterative constraint thinning. This relies on a classification of each vertex of R. The authors proposed then three vertex types and $c(p_i)$, the *complexity* of the vertex p_i such as:

$$c(p_i) = \sum_{j=0}^{m_i-1} |F(q_j) - F(q_k)|, \tag{3}$$

where $k = j + 1 \bmod m_i$ and $q_j, q_k \in \mathcal{N}(p_i)$.

Definition 1. *A vertex p_i is considered as* complex *if and only if $c(p_i) \geq 4$. The set of all* complex *vertices is named C.*

A *complex* vertex p_i thus potentially corresponds to a part of a skeleton branch if $c(p_i) = 4$, or a connection through several branches if $c(p_i) > 4$.

Definition 2. *A vertex p_i is marked as* center *if and only if and $\mathcal{N}(p_i) \subseteq R$. The set of all* center *vertices is named E.*

Definition 3. *A vertex p_i is called* disk *if and only if $\exists q_j \in \mathcal{N}(p_i), q_j \in E$ that is a* center*. The set of all* disk *vertices is named D.*

A *disk* vertex corresponds to a *simple* point: a point that does not modify the expected skeleton topology if it is removed [3]. We denote $\overline{X}$ the complementary of the set X in the region R.

Definition 4. *The* skeleton operator *of R is defined as a constrained thinning:*

$$skeletonize(R) = R \setminus (D \cap \overline{C \cup E}). \tag{4}$$

After applying the skeleton operator until idempotence on R, the set of the remaining vertices, corresponding to the final *skeleton*, is called Sk_R. During each pass, the skeleton operator removes the boundary *disk* vertices. Figure 1 illustrates the execution of the algorithm. After obtaining the skeleton Sk_R of R, it is possible to remove the smallest branches. This last operation is called *pruning* and defined as follows:

$$prune(Sk_R) = Sk_R \setminus \overline{C}. \tag{5}$$

This pruning step is shown by Figure 1 (d).

2.3 Result Assessment

Due to the simplicity of the used operators, the computational time of the Rössl *et al.* method is very low, and the skeleton extraction is thus almost instantaneous on meshes composed of 50K triangles. However, the accuracy and the continuity of the obtained skeleton deeply depends on the mesh configuration. In other words, a same set R defined on two different triangulations of S could lead to skeletons with two topologies drastically different. Moreover, the lack of continuity also occurs in the case of particular configurations that are shown in

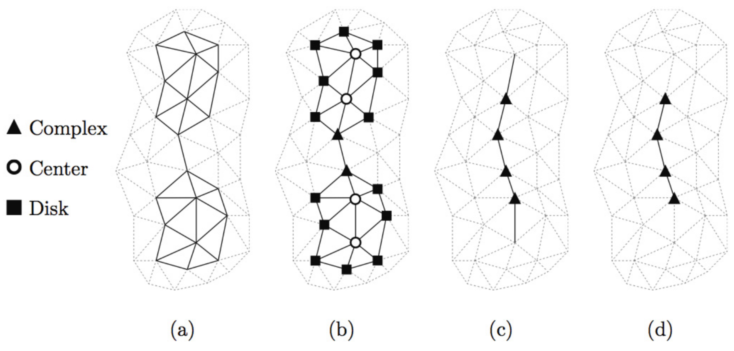

Fig. 1. Illustration of the Rössl *et al.* algorithm. From left to right: (a) a set of vertices R, (b) classification of R, (c) thinning until idempotence, and (d) resulting skeleton after pruning.

Figure 2 because the removal of *disk* vertices can modify the topology of the skeleton. Figure 3 illustrates the unexpected results and disconnections generated by the execution of the skeletonization. Once the vertices P_1 and P_2 are removed (b), the skeleton becomes disconnected at this location (c). However, some vertices would change to *complex* if a new classification step was applied. This kind of vertices represents relevant points in a topological point of view and thus, should not be deleted.

Another issue occurs since pruning is applied: the ending vertices of the skeleton are removed. As a matter of fact, when the set R contains no *center* and no *complex* vertex, the pruning operator removes all the vertices. This case is illustrated by Figure 4.

3 A Skeletonization Method for Any Arbitrary Triangulated Mesh

Both a new definition of particular vertices and a new algorithm have been elaborated to solve the disconnection issues previously raised up in Section 2. These two key points of the approach we propose are successively presented below.

3.1 Additional Definitions

The different classes of vertices proposed by Rössl *et al.* aim at describing the topology of R. However, they are not sufficient as there are still vertices that are unmarked and that are then not considered in the skeletonization. For this reason, we introduce the *outer* class.

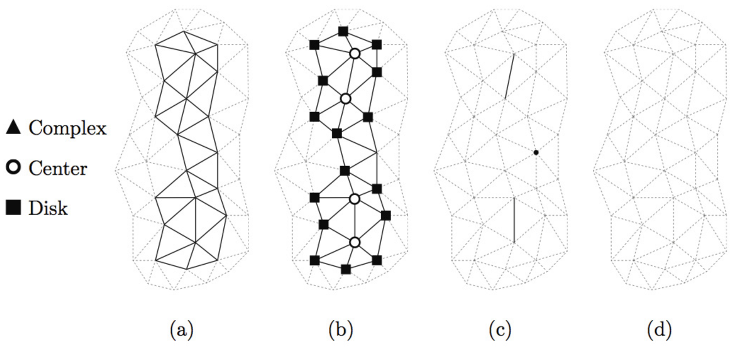

Fig. 2. Example of unexpected results by applying the Rössl *et al.* method. From left to right: (a) the set of feature points R, (b) classification of R, (c) skeletonization of R, (d) resulting skeleton after pruning.

Definition 5. *A vertex p_i is marked as* outer *if and only if $F(p_i) = 1$ and $p_i \notin (C \cup D \cup E)$. The set of* outer *vertices is named O and is defined as follows:*

$$O = R \setminus (C \cup D \cup E) \tag{6}$$

As it has been shown previously, a vertex may change from one class to another and, as a side-effect, this may lead to potential disconnections during the skeletonization. To counteract this issue, we propose to define a priority between the classes.

Definition 6. *The* disk *class has a lower priority over the other classes.*

If a vertex is already classified as *disk*, it can change to *complex*, *center* or *outer* if necessary.

3.2 Algorithm

If the skeleton operator defined by Rössl *et al.* is directly applied to an unstructured patch, the final result may suffer from disconnections as some *disk* vertices are deleted while they characterize the topology of the object. To correct this issue, the algorithm we propose does not remove *all* the *disk* vertices but only those that will not be converted to a different priority class after the operator application. This requires to add an additional step in the algorithm: at each application of the skeleton operator, the class of a vertex is recomputed before its deletion. For example, if a *disk* vertex becomes a *complex* vertex, the vertex is not removed.

However, the resulting skeleton may be too thick using this technique (*e.g.* if it is composed of only *outer* vertices). For this reason, a final cleaning step is added to obtain the expected skeleton. At this stage, the skeleton must be composed of *complex* vertices (*i.e.* the skeleton branches or nodes) and *outer*

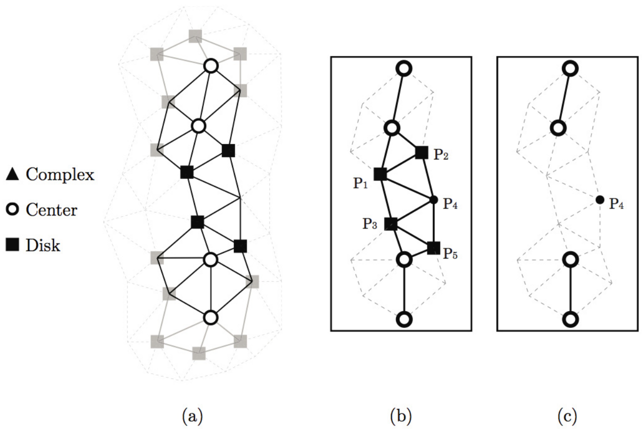

Fig. 3. Execution of the skeletonization operator [10]: (a) vertex classification, (b) execution of the algorithm, (c) final skeleton with a broken topology

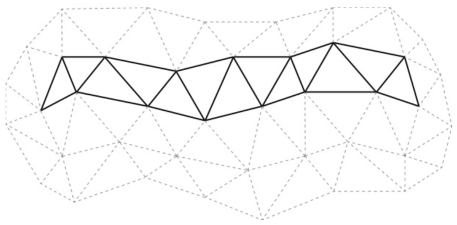

Fig. 4. Example of a particular configuration: while the vertices of R are not classified, they will be deleted by the pruning operator of Rössl *et al.*

vertices, the ending points of the branches with only one *complex* vertex in their neighborhood. Thus, to obtain the final skeleton, a two steps process is applied:

– the *outer* vertices that have more than two neighbors belonging to R are removed;
– the *outer* vertices with at most one neighbor belonging to R are kept.

Moreover, as for the skeleton operator, each vertex complexity change is checked before removing this vertex. Examples of resulting skeletons are shown in Figure 6 and the impact of the algorithm modification with the update step is presented in Figure 7: *disk* vertices are deleted (b) after checking their classes (c). During the deletion of P_1 and the update step, the class of P_2 changes from *disk* to *complex* and P_4 from *outer* to *complex*. Thus, these vertices are not removed and the extracted skeleton is fully connected and faithfully characterizes the topology of R (d). The complete method of skeleton extraction is summarized by the algorithm presented on Figure 5.

```
**repeat**
    **forall the** *vertices* $p_i \in R$ **do**
        **if** $p_i$ *is a* disk *vertex* **then**
            compute the complexity $c(p_i)$ of the vertex
            **if** *the priority of* $p_i$ *does not change* **then**
                delete $p_i$
**until** *idempotence*

**repeat**
    **forall the** *vertices* $p_i \in R$ **do**
        **if** $p_i$ *is an* outer *vertex* **then**
            compute the complexity $c(p_i)$ of the vertex
            **if** *the priority of* $p_i$ *does not change and if* $|\mathcal{N}(p_i)| > 2$ **then**
                delete $p_i$
**until** *idempotence*

**repeat**
    **forall the** *vertices* $p_i \in R$ **do**
        **if** $p_i$ *is an* outer *vertex* **then**
            compute the complexity $c(p_i)$ of the vertex
            **if** *the priority of* $p_i$ *does not change and if* $|\mathcal{N}(p_i)| > 1$ **then**
                delete $p_i$
**until** *idempotence*
```

Fig. 5. Extraction of the skeleton

4 Results

Some results of skeleton extraction on meshes are presented in Figures 8, 9 and 10. The obtained skeletons describe the geometry and the topology of the original set R. The used meshes are relatively homogeneous in Figure 8 while, in Figures 9 and 10, the algorithm has been tested on irregular meshes to show the robustness of the proposed approach to unstructured meshes. It may be noticed that the resulting skeletons are the expected ones and reflect correctly the topology and geometry of the original set R in a proper way.

Moreover, since the definitions and the operators used to extract the skeleton are very simple, the computational time of the proposed approach is also very low, even if an additional checking step has been added. It is possible to process a mesh with 100K vertices in 1 second. The tests have been ran on an Intel Core 2 Duo 2.8 Ghz.

To complete the algorithm tests and to evaluate the robustness of the proposed approach, an application dedicated to the feature line detection is presented in the following section.

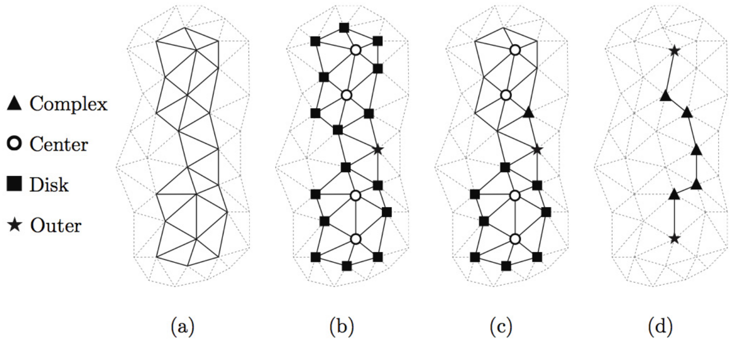

Fig. 6. Illustration of the proposed approach: (a) region R, (b) vertex classification, (c) execution of the thinning algorithm with update, (d) final skeleton fully connected

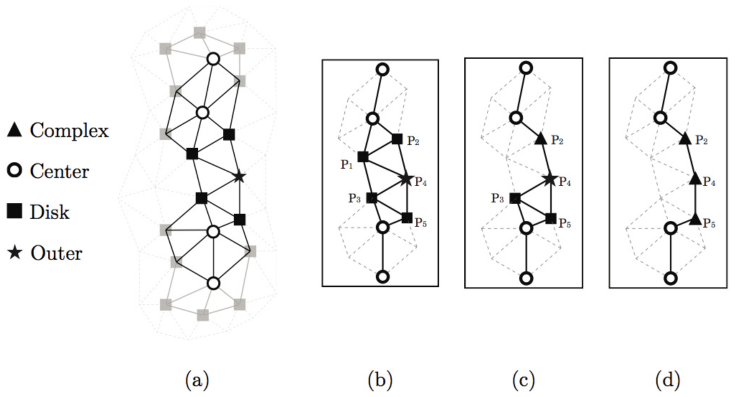

Fig. 7. Detailed view of the thinning process: (a) vertex classification, (b) execution of the skeleton operator, (c) update of vertex classes after deletion, (d) final skeleton

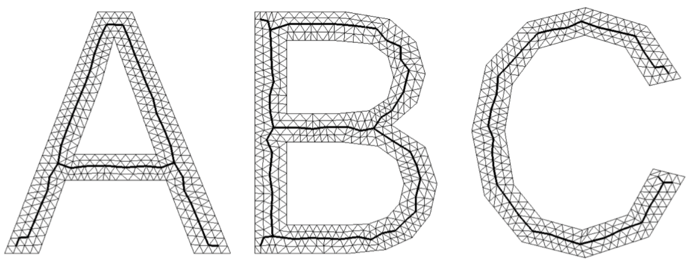

Fig. 8. Application of the skeletonization algorithm on regular triangulated 3D meshes

5 Application to the Feature Line Detection

The detection of features within 3D models is a crucial step in shape analysis. It is possible to extract from the surface of an object simple shape descriptors such as lines (drawn on the surface). Generally, the methods of feature line detection focus on the estimation of differential quantities and the research of curvature

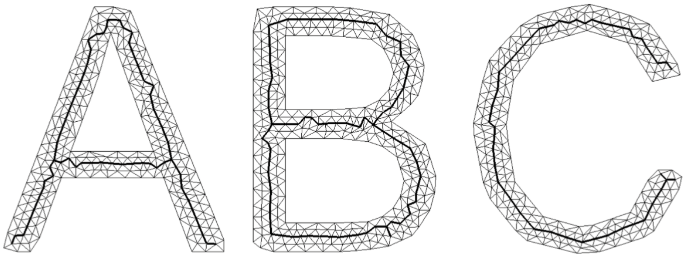

Fig. 9. Skeleton extraction on irregular 3D meshes

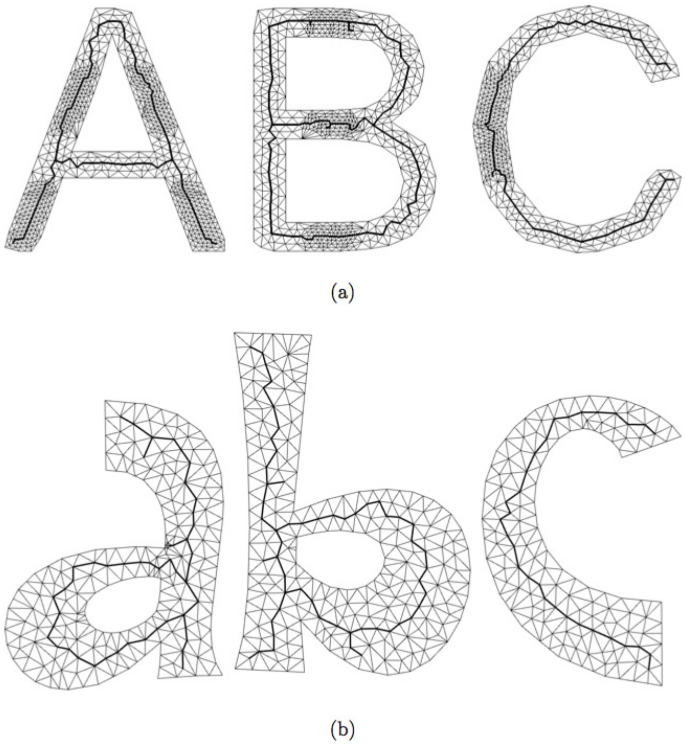

Fig. 10. Extraction of the skeletons on meshes with mixed and unstructured meshes

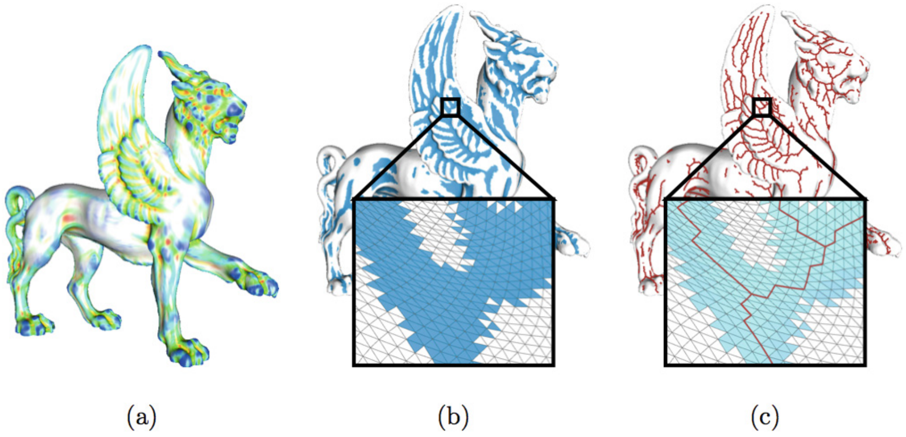

Fig. 11. Algorithm of feature lines extraction: (a) curvature estimation, (b) definition of the set R, (c) extraction of lines from R by the proposed thinning approach

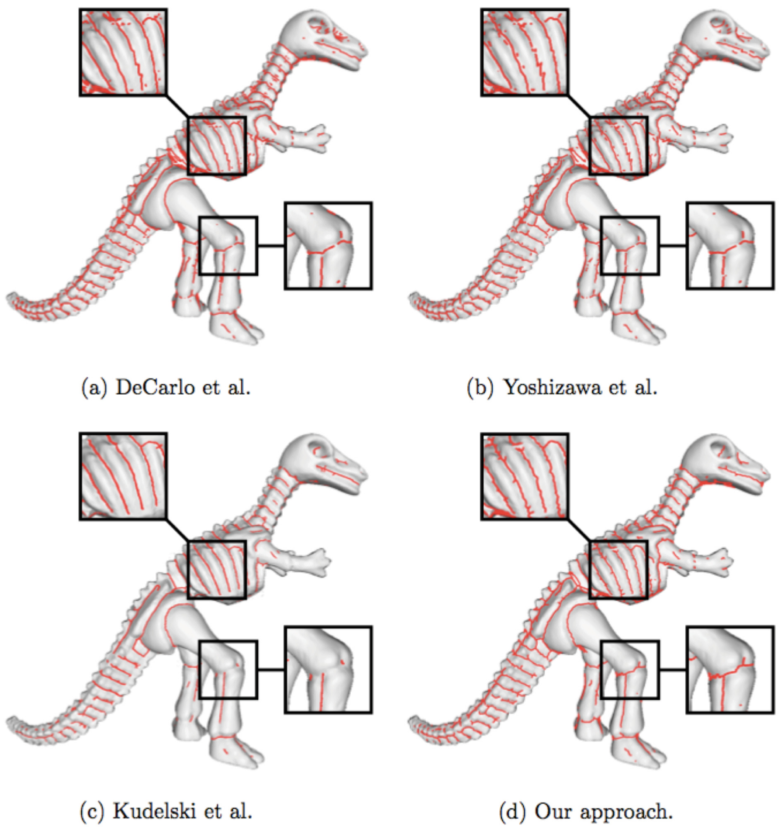

Fig. 12. Comparison of results obtained from feature detection applied on *Dinosaur*

extrema. However, these techniques are based on *third-order* differential properties and it leads to a common issue: they produce disconnected feature lines because of flat and spherical areas and because of the noise present in data sets. Thus, it is particularly difficult to generate intersections between feature lines. To overcome these recurrent issues, we propose to apply our method to extract salient lines of a model.

In order to define sets over triangulated 3D meshes, we use the algorithm proposed by Kudelski *et al.* [6]. We compute the mean curvature H through a local polynomial fitting in the least-squares sense [5]. The binary attribute F is then defined at each vertex p_i as follows:

$$H_{p_i} > 0 \implies F(p_i) = 1. \tag{7}$$

Finally, the objective is to thin the set, corresponding to potential feature parts of the mesh, in order to obtain lines describing the geometry and the topology of the object.

Figure 11 illustrates the process of feature line detection. The obtained characteristic lines are fully connected and describe accurately the topology of the sets. Then, due to the use of second-order differential properties (*i.e.*, the mean curvatures), the feature extraction is more robust. Moreover, this type of approach allows to generate intersections between feature lines which it is not possible with classical approaches (Figure 12).

6 Conclusion

In this article, we have proposed an efficient and general new algorithm to extract in a robust way the skeleton of a set R defined on a triangulated mesh by topological thinning. This approach relies on the definitions presented by Rössl *et al.* [10]. However, the latter generates, for some mesh configurations, unexpected skeletons that are generally more disconnected than they should. To overcome this issue, an additional definition of vertex categories has been added. Then, we have improved the thinning process by integrating a priority between vertex classes. Tests have been applied on different categories of meshes (homogeneous and heterogeneous) and set configurations. These tests and the application of feature line extraction, presented in the end, illustrate the efficiency of the approach.

As future work, a formal proof based on [2] and issued from the notion of *simple vertices* (by analogy to *simple points*) may need to be considered. Indeed, the Rössl *et al.* article does not include formal validations because the vertices classification is incomplete. With the changes we have made, the *disk* vertices truly correspond to simple points lying on a discrete 2-manifold. Thus it will be possible to transpose the notion of geodesic neighborhood in order to define topological numbers associated with simple vertices.

A second prospect is related to the position of the skeleton nodes. They may be still discussed and further works have to be dedicated to this subject. Indeed, the defined operators do not integrate any geometrical information and the extraction of the skeleton only relies on a one-ring neighborhood. However, as the position of the skeleton is generally easier to correct than the topology, post-processing steps could be envisaged to optimize the skeleton position. A possible improvement of our method will be to refine the node placement by energy minimization during the extraction to evolve like *active contours*. In this way, the resulting skeleton will describe in a better way both the topology and the geometry of the set lying on the mesh.

Acknowledgments. The authors would like to thank Jean Borgomano and Yves Guglielmi of the Geology of Carbonate Systems and Reservoirs laboratory for their precious help and pieces of advice. The models were provided courtesy of Caltech Multi-Res Modeling Group (Feline) and Cyberware (Dinosaur).

References

1. Au, O.K.C., Tai, C.L., Chu, H.K., Cohen-Or, D., Lee, T.Y.: Skeleton extraction by mesh contraction. ACM Transaction on Graphics 27(3), 1–10 (2008)
2. Bertrand, G.: Simple points, topological numbers and geodesic neighborhoods in cubic grids. Patterns Recognition Letters 15, 1003–1011 (1994)
3. Bertrand, G.: A boolean characterization of three-dimensional simple points. Pattern Recognition Letters 17, 115–124 (1996)
4. Gall, J., Stoll, C., De Aguiar, E., Theobalt, C., Rosenhahn, B., Seidel, H.: Motion capture using joint skeleton tracking and surface estimation. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), pp. 1746–1753. IEEE Computer Society (June 2009)
5. Goldfeather, J., Interrante, V.: A novel cubic-order algorithm for approximating principal direction vectors. ACM Transaction on Graphics 23(1), 45–63 (2004)
6. Kudelski, D., Mari, J.L., Viseur, S.: 3D feature line detection based on vertex labeling and 2D skeletonization. In: IEEE International Conference on Shape Modeling and Applications (SMI 2010), pp. 246–250. IEEE Computer Society (June 2010)
7. Kudelski, D., Mari, J.L., Viseur, S.: Extraction of feature lines with connectivity preservation. In: Computer Graphics International (CGI 2011 Electronic Proceedings) (June 2011)
8. Lee, T., Kashyap, R., Chu, C.: Building skeleton models via 3-D medial surface/axis thinning algorithms. Graphical Models and Image Processing 56(6), 462–478 (1994)
9. Mari, J.-L.: Surface Sketching with a Voxel-Based Skeleton. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) DGCI 2009. LNCS, vol. 5810, pp. 325–336. Springer, Heidelberg (2009)
10. Rössl, C., Kobbelt, L., Seidel, H.P.: Extraction of feature lines on triangulated surfaces using morphological operators. In: AAAI Spring Symposium on Smart Graphics, vol. 00-04, pp. 71–75 (March 2000)
11. Siddiqi, K., Pizer, S.: Medial Representations. Mathematics, Algorithms and Applications. Computational Imaging and Vision, vol. 37. Springer (2008)
12. Yu, K., Wu, J., Zhuang, Y.: Skeleton-Based Recognition of Chinese Calligraphic Character Image. In: Huang, Y.-M.R., Xu, C., Cheng, K.-S., Yang, J.-F.K., Swamy, M.N.S., Li, S., Ding, J.-W. (eds.) PCM 2008. LNCS, vol. 5353, pp. 228–237. Springer, Heidelberg (2008)
13. Zhang, T.Y., Suen, C.Y.: A fast parallel algorithm for thinning digital patterns. Communications of the ACM 27(3), 236–239 (1984)