Pacific Graphics 2008
T. Igarashi, N. Max, and F. Sillion
(Guest Editors)

Volume 27 (2008), Number 7

Patch-type Segmentation of Voxel Shapes using Simplified Surface Skeletons

Dennie Reniers[1] and Alexandru Telea[2]

[1]Department of Mathematics and Computing Science, Eindhoven University of Technology, The Netherlands
[2]Institute of Mathematics and Computing Science, University of Groningen, The Netherlands

Abstract

We present a new method for decomposing a 3D voxel shape into disjoint segments using the shape's simplified surface-skeleton. The surface skeleton of a shape consists of 2D manifolds inside its volume. Each skeleton point has a maximally inscribed ball that touches the boundary in at least two contact points. A key observation is that the boundaries of the simplified fore- and background skeletons map one-to-one to increasingly fuzzy, soft convex, respectively concave, edges of the shape. Using this property, we build a method for segmentation of 3D shapes which has several desirable properties. Our method segments both noisy shapes and shapes with soft edges which vanish over low-curvature regions. Multiscale segmentations can be obtained by varying the simplification level of the skeleton. We present a voxel-based implementation of our approach and illustrate it on several realistic examples.

1. Introduction

Shape segmentation is an important pre-processing step in many applications ranging from shape analysis, computer vision, compression, and collision detection. The type of segments produced depend on the target application, so a wealth of methods exist. One can distinguish between patch-type and part-type segmentation methods [Sha04]. *Patch-type* methods are geometry-oriented, typically using local shape information such as surface curvature, and produce segments that are quasi-flat and separated by high-curvature creases. *Part-type* methods, on the other hand, are more semantically-oriented, i.e., they try to find segments that a human user would intuitively perceive a distinct logical parts of the shape. Such segments are not necessarily separated by high-curvature creases.

The skeleton of a 3D shape provides a compact, hierarchical, description of the parts a shape is made of. In 3D, one can compute both curve and surface skeletons of a shape. The curve skeleton and the related Reeb graph are higher-level abstractions that describe the shape using only 1D curves. Curve skeletons are typically used in part-type segmentation methods that can handle smooth, organic shapes. However, they are ill-suited for obtaining patch-type segmentations, since they do not capture the shape's edges. For example, the curve skeleton of a box cannot be readily used to detect its six faces. The surface skeleton, or medial surface, is a structure consisting mainly of 2D manifolds, and 1D curves for degenerate cylindrical parts. The surface skeleton is more suitable for patch-type segmentations, and has additional advantages as compared to non-skeleton-based segmentations. As far as we know, the surface skeleton has not yet been used for 3D shape segmentation, probably because of its infamous instability to boundary noise.

In this paper, we propose a patch-type segmentation method based on a shape's simplified surface-skeleton. Since the simplified skeleton encodes the shape's entire volume, our method is more robust against boundary noise than typical surface-curvature-based segmentation methods. Furthermore, our approach can find weak and disappearing edges, produce multiscale segmentations, and also captures the shape's symmetry in its segmentation. Our method produces segmentations of voxel shapes, in which the shape is sampled on a regular grid: a representation often used in the discrete-geometry and medical-imaging communities. Segmentation of voxel data brings its own difficulties. In contrast with mesh representations, the notion of an edge is implicit in the voxel representation. Furthermore, the resolution of the data is typically low, the data contains discretization

Published by Blackwell Publishing, 9600 Garsington Road, Oxford OX4 2DQ, UK and 350 Main Street, Malden, MA 02148, USA.

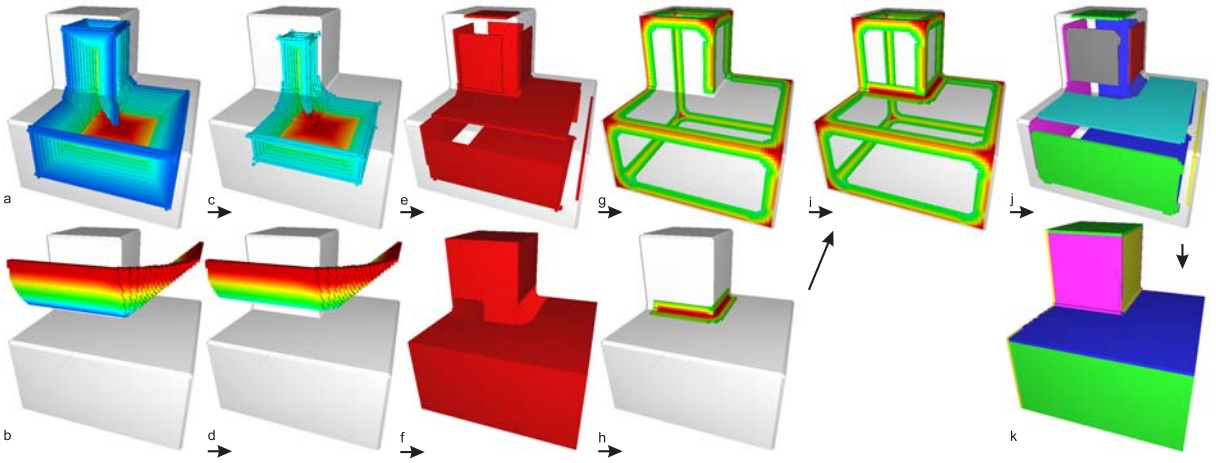

Figure 1: *Overview of our approach. Fore- and background skeletons (a,b), rainbow color-map encodes importance measure. Simplified fore- and background skeletons (c,d). Gaps in feature points (e,f). Convex edges (g). Concave edges (h). Combined edges (i). Connected components (j). Final segmentation (k).*

artifacts and other noise, and boundary normals are not readily available.

We briefly describe our algorithm. Each point on the 3D skeleton has two points on the shape's boundary at minimum distance, which we call *feature points*. To simplify the skeleton, an importance measure is defined on the skeleton as the shortest geodesic length between each pair of associated feature points [PH02, DS06]. This measure is lowest near the periphery of the skeleton, and increases toward its center. When simplifying the skeleton by thresholding this importance measure, feature points near convex shape edges disappear. The skeleton of the shape's outside volume, or background, can also be computed. Its simplification removes feature points near concave shape edges. The key idea is that these gaps in the feature points form the shape's edges, and induce a segmentation of the shape surface. Figure 1 illustrates our approach.

Section 2 presents related work. In Section 3, we discuss preliminaries, including the computation of multiscale skeletons [RVT08] which is a key component of our segmentation approach. In Section 4, we discuss the core of our approach. Section 5 present results and discussion, Section 6 concludes this paper.

2. Related Work

We now briefly overview some of the patch-type segmentation methods that relate to our approach. Garland et al. [GWH01] perform a hierarchical clustering of the mesh faces to produce a patch-type segmentation consisting of planar segments. Clarenz et al. [CGST04] perform a fuzzy multiscale segmentation of a 3D shape, based on surface curvature. However, this method often generates noisy edges in low-curvature regions. Borgefors et al. [BdBS09] compute local thickness in combination with a multiresolution structure to produce a hierarchical segmentation of 3D voxel shapes. Hisada et al. [HBK01] use the skeleton in combination with denoising and filter techniques to detect salient shape features of polygonal shapes. Mangan and Whitaker [MW99] partition a surface into patches of similar curvature using a watershed algorithm that uses curvature as its height function. Zuckerberger et al. [ZTS02] give an improved watershed method and give numerous segmentation applications. Provot et al. [PDR08] segment voxel shapes by detecting discrete planes with variable width.

Edges can be seen as local features, for which various definitions and detectors have been proposed in the past, some of which also use the skeleton definition. The *local feature size* [AB98] is defined as the distance from a boundary point to the skeleton. In contrast, our detector (Sec. 4.2) also incorporates global information, by using shortest paths on the boundary, and as such can distinguish between locally similar, but globally different boundary configurations.

3. Preliminaries

Let Ω be a 3D shape with closed boundary $\partial\Omega$. Let $D : \Omega \rightarrow \mathbb{R}_+$ be the distance transform, assigning to each point inside the shape the minimum distance to the boundary. Let $F : \Omega \rightarrow \mathcal{P}(\partial\Omega)$, where $\mathcal{P}$ is the power set, be the feature transform [RVT08], assigning to each point inside the shape the set of boundary points at minimum distance, called the *feature points*:

$$F(p \in \Omega) = \left\{ q \in \partial\Omega \;\middle|\; \|p - q\| = D(p) \right\}. \qquad (1)$$

The *skeleton* $\mathcal{S}$ of Ω can be defined as the locus of centers of maximally inscribed balls. At each point p on the skeleton, a maximally inscribed ball can be placed that touches

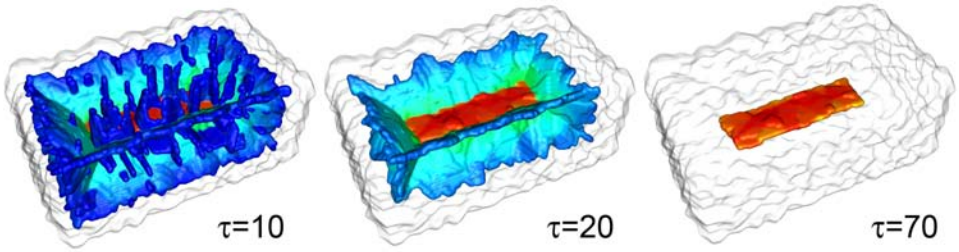

Figure 2: *Simplified skeletons $\mathcal{S}_\tau$ of a noisy box. The color map encodes the importance measure ρ (blue=low, red=high). Whereas $\mathcal{S}_{\tau=10}$ contains some spurious sheets, $\mathcal{S}_{\tau=20}$ is robust. In $\mathcal{S}_{\tau=70}$, only the center sheet is retained, which can be seen as a coarse-scale representation of the box.*

the boundary in at least two points, the feature points $F(p)$:

$$\mathcal{S}(\Omega) = \{p \in \Omega \mid |F(p)| \geq 2\}. \quad (2)$$

This definition can be used both in 2D and 3D. In 3D, $\mathcal{S}$ is also called the medial surface, or *surface skeleton*, to distinguish it from the curve skeleton. The surface skeleton consists of sheets, that intersect in so-called Y-curves. Whereas sheet points have two feature points, Y-curve points have three or more feature points which are the union of the feature-point pairs of the intersecting sheets. In addition, a surface skeleton can also contain isolated curves in some degenerate cases, such as a cylinder. Such objects are better handled by part-type segmentation method, e.g. [RT07]. We assume for now that the skeleton contains no such degeneracies.

3.1. Simplified Skeletons

Following from Eq. 2, skeletons are inherently sensitive to small boundary perturbations. Sampling or acquisition noise produces spurious skeleton sheets. To handle this, some skeletonization methods define an importance measure $\rho : \mathcal{S} \rightarrow \mathbb{R}_+$ indicating the importance for each skeleton point to the shape representation. Together with a subsequent pruning strategy, this delivers a simplified skeleton.

In [PH02], an importance measure ρ was proposed for 3D skeletons, as the length of the shortest path on the surface $\partial\Omega$ between the two feature points $F(p)$ for each point $p \in \mathcal{S}$. This measure, which we shall use in our method, smoothly evolves over skeletal sheets, may contain jumps at Y-curves (cf. Fig. 2), and has a local maximum ridge that forms a 1D connected structure on $\mathcal{S}$. This last property has been shown in [DS06] and was used to formally define a curve skeleton.

We compute both the distance and feature transform using [Mul92]. Let $\overline{F}(p)$ be the *extended* feature transform, defined as $\overline{F} = \bigcup_{x,y,z\in\{0,1\}} F(p_x + x, p_y + y, p_z + z)$, which ensures that, unlike the normal feature transform, skeleton voxels contain at least one feature voxel on each side of the skeleton. Now, the importance measure $\rho : \Omega \rightarrow \mathbb{R}_+$ is defined on the object voxels as the maximum shortest-path length between the points in $\overline{F}(p)$:

$$\rho(p) = \max_{a,b\in\overline{F}(p)} g(a,b)\ , \quad (3)$$

where g is the shortest-path length between a, b. Shortest paths are computed using Dijkstra's method on the boundary graph, in which voxels are nodes and two nodes are connected when their voxels share at least one corner. We use the length estimator of [KS93] instead of using simply the Euclidean distance between voxel centers. Using ρ, the definition of the discrete simplified-skeleton $\mathcal{S}_\tau$ becomes (cf. Eq. 2):

$$\mathcal{S}_\tau(\Omega) = \{p \in \Omega \mid \rho(p) \geq \tau\}. \quad (4)$$

Empirical study suggests that the threshold τ should be at least 5 voxel lengths to detect the skeleton $\mathcal{S}_\tau$ so that it is robust to discretization artifacts [PH02]. The threshold τ functions as a continuous scale-parameter controlling the simplification level. Small τ values eliminate less important skeleton parts that are due to small-scale surface features or noise. Larger τ values can be used to retain the most salient parts of the skeleton. Figure 2 illustrates the use of τ for a noisy box. Please refer to [RVT08] for more implementation details.

The simplified skeletons are connected because the measure ρ is monotonic on $\mathcal{S}_\tau$ except on the local maximum ridge [DS06]. For completeness, we note that the definition of ρ (Eq. 3) can be extended to the curve skeleton points in order to obtain a fully monotonic measure for *all* skeleton points [RVT08], but this is outside this paper's scope.

4. Segmentation Method

We use the simplified skeleton of the shape's interior Ω (foreground), denoted $\mathcal{S}_\tau(\Omega)$, to detect convex shape edges, and the skeleton of the exterior $\bar{\Omega}$ (background), denoted $\mathcal{S}_\tau(\bar{\Omega})$, to detect concave edges.

4.1. Background Skeleton

Computation of the background skeleton $\mathcal{S}_\tau(\bar{\Omega})$ is slightly different from the computation of $\mathcal{S}_\tau(\Omega)$, as follows. The background volume $\bar{\Omega}$ is enclosed in a bounding box around

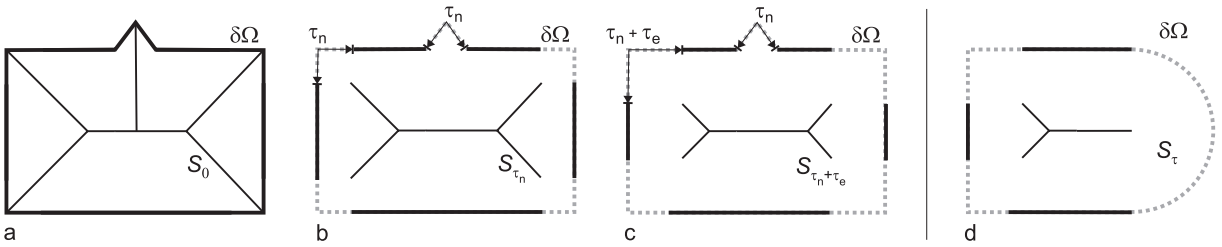

Figure 4: *Non-simplified skeleton (a). Simplified skeleton at scale τ_n (b). Simplified skeleton at scale $\tau_n+\tau_e$ (c). Large gap due to round part (d). Thick lines are feature collections V^{Ω}.*

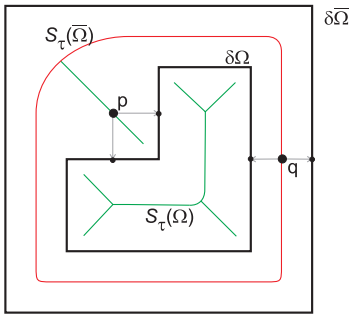

Figure 3: *Background skeleton $\mathcal{S}_\tau(\bar{\Omega})$ of exterior volume bounded by $\partial\bar{\Omega}$, points $p,q \in \mathcal{S}_\tau(\bar{\Omega})$, and feature points $F(p),F(q)$.*

the entire shape. The background volume needs to be bounded, otherwise, the maximally inscribed balls would be of infinite size. The voxels on the surface of this bounding box form an additional boundary $\partial\bar{\Omega}$ which is disjoint from the object boundary $\partial\Omega$. Background voxels that have feature voxels only on $\partial\bar{\Omega}$ are discarded, voxels that have feature voxels only on $\partial\Omega$ are assigned the same importance as foreground voxels would. To voxels that have feature voxels on both $\partial\Omega$ and $\partial\bar{\Omega}$ we assign an infinite importance (ρ=∞), so that they will never be simplified regardless the setting of τ. This is the desired behavior: the background skeleton should only be simplified for concave edges, not for any other parts. Note that in Figure 1d, the background skeleton is not shown for the sake of clarity. A cross-section of the background skeleton of that object is shown in Figure 3. Point p is a point having both feature points on $\partial\Omega$, whereas point q has one feature point on $\partial\Omega$ and one on $\partial\bar{\Omega}$. Point q will never be simplified ($\rho(q)=\infty$), regardless the setting of τ.

4.2. Edge Detection

Let V^{Ω} be the set of feature points associated with the simplified skeleton, called the foreground *feature collection*:

$$V^{\Omega}(\mathcal{S}_\tau) = \bigcup_{p\in\mathcal{S}_\tau(\Omega)} \overline{F}(p). \tag{5}$$

The feature collection of the background skeleton is denoted $V^{\bar{\Omega}}$. The key idea of our approach is that, by increasing the threshold τ on the importance measure ρ, gaps will appear in V on and near shape edges (Fig. 1e,f), which we can detect. However, the parameter τ is also used to prune spurious skeleton parts that are due to boundary noise. Setting τ to the noise level τ_n opens V on the edges, but also on noisy parts. Therefore, we further increase τ to $\tau_n+\tau_e$: V is opened further on edges, but not on boundary noise. This is illustrated in Fig. 4 in the 2D case (for the sake of clarity). In Fig. 4a, the non-simplified skeleton $\mathcal{S}_0$ of a box with a noise bump is shown. The feature collection (thick lines) covers the whole boundary. When τ is set to the noise level τ_n (Fig. 4b), the openings in V^{Ω} on the bump and near the non-noisy convex corners have the same size, so that we cannot differentiate between the two situations. By further increasing τ to $\tau_n+\tau_e$ (Fig. 4c), V^{Ω} is further opened on the corners, but not on the bump.

Hence, we can detect convex edges by computing for each boundary point $q\in\partial\Omega$ the geodesic distance to $V^{\bar{\Omega}}(\mathcal{S}_{\tau_n+\tau_e})$: points at a distance of at least $\frac{1}{2}\tau_n$ are detected as edge points. The term τ_e controls the minimum detected edge width. In a voxel-based implementation, we verified that setting $\tau_e \geq 4$ gives good results. The convex edges E^{Ω} are given by:

$$E^{\Omega} = \{q\in\partial\Omega \mid \min_{v\in V^{\Omega}} g(q,v) > \frac{1}{2}\tau_n\}, \tag{6}$$

where g is again the geodesic distance on $\partial\Omega$. Concave edges $E^{\bar{\Omega}}$ are detected by replacing V^{Ω} by $V^{\bar{\Omega}}$ in Eq. 6. All in all, the combined edges $E=E^{\Omega}\cup E^{\bar{\Omega}}$ (Fig. 1i) divide the boundary into a set of connected components C (Fig. 1j).

The edge width parameter τ_e controls the minimum width of the detected edges, but not the maximum. In case of round parts of the shape, e.g. as shown in Fig. 4d, the openings in V and thus the edges might become thicker than τ_e. Both thick and thin edges are handled by the edge erosion step, as detailed in the next section. Note that in the continuous case, there would not be a gap in the feature collection at all on the curved part, because the right-most skeleton point has the whole curve as feature points. However, since we keep for each skeleton point only the two feature points that are farthest apart, we do not have this problem.

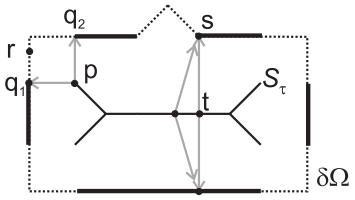

Figure 5: *Normal computation*

4.3. Normal-sensitive edge erosion

The set of connected components C are separated from each other by the borders in E of at least width τ_e. In order to produce thin segment borders, we erode the edge voxels in E, thereby dilating the components in C. Instead of a geodesic-distance-ordered erosion, we do a *normal-sensitive erosion* of the edges. For this, we need to compute a normal on every boundary point first.

It is well known that the *feature vectors*, which point from a skeleton point $p \in \mathcal{S}$ to its feature points $q \in F(p)$ are normal to the boundary $\partial\Omega$ [PSS*03]. Hence, we use $\frac{q-p}{\|q-p\|}$ as our normals. This is shown in Figure 5 in 2D for illustrative purposes. By using our robust simplified skeleton $\mathcal{S}_\tau$, which does not contain any spurious parts due to noise, the resulting normals are also robust to noise. Normals may be ambiguous at a few boundary points in the case of skeleton ligatures, e.g. point s in Fig. 5. Multiple skeleton points have point s as feature points. In such cases, we use the normal of that skeleton point which has the largest angle between its two feature vectors.

However, as explained already, the feature collection of the simplified surface-skeleton does not completely cover the shape's surface, and hence the normals are not available everywhere. For the other boundary point r, we get the nearest known point q and get its associated skeleton point p. The normal of r is then the feature vector at p whose dot product with $r-p$ is maximal, which is $\frac{q_1-p}{\|q_1-p\|}$ in case of Fig. 5. The erosion of the edges is then performed in the order of most similar normals, that is, each voxel e on the edge is assigned to the component of voxel p for which the the cumulative normal-difference along the boundary path between p and e is minimal.

4.4. Handling corners

Although Eq. 6 is well-suited to detect the shape's edges, problems occur at corners, where edges of different strength come together. The reason is that for the same setting of τ, weak edges have larger inscribed balls than strong edges (see Fig. 6). At corners where a strong and weak edge come together, only the smallest ball fits inside the shape. Hence, the feature collection will contain feature points on opposite sides of the strong edge, making the weak edge undetected near the corner.

Figure 7 illustrates this for the two stacked boxes from

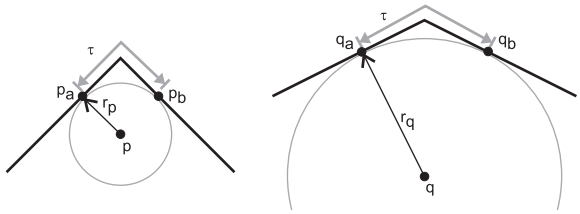

Figure 6: *Cross-section of a sharp (left) and weak edge (right). Inscribed ball centered at a point p has feature points p_a and p_b and radius r_p. Feature points p_a, p_b are at geodesic distance τ.*

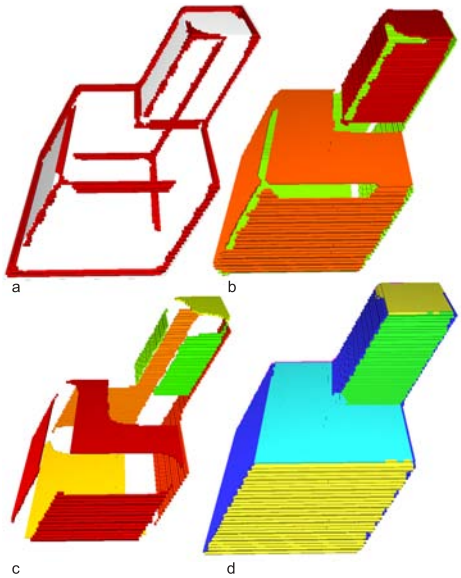

Figure 7: *Handling corners. The detected edges E (a). Connected components have thin connections (b). Splitting components (c). Final segmentation after edge erosion (d).*

Fig. 1, but now skewed. Fig. 7a shows the set of detected edge voxels E, which has disconnections near some of the corners. The resulting segmentation in Fig. 7b shows thin connections between components. A straightforward solution is to dilate the edges E so that these connections between components disappear. The edge erosion step (Sec. 4.3) will then remove the dilated edges. Let H_ε be the set of points at a geodesic distance of at least ε of the detected edge voxels E, so that $H_0 = \partial\Omega \setminus E$. For high enough ε there will be no thin connections left in the connected component of H_ε. However, for high values of ε some of the smaller connected components in H_ε may be completely removed, or some of the components in H_0 may inadvertently be split into multiple components. We proceed as follows. Starting with $\varepsilon = 0$, we iteratively increase ε with small increments, and consider the connected components in H_ε. We check whether a component c in a level H_i is split into multiple components d, e in level H_j $(i < j)$. In that case, we let component c split only if the resulting components

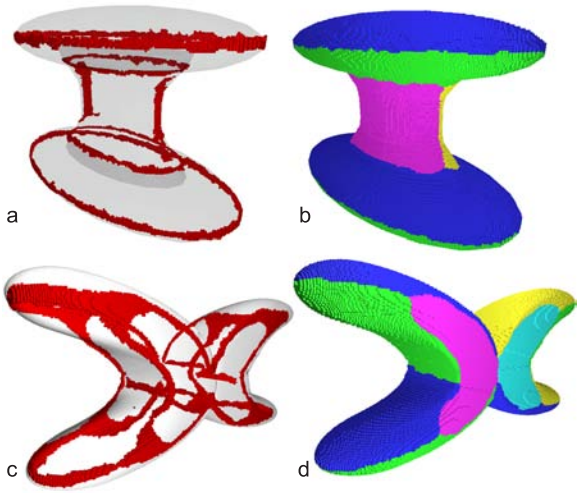

Figure 9: *The detected edges (a,c) and segmentation (b,d) of a smooth H-shape and smooth X-shape respectively.*

d, e are not too small and have a different orientation (using the normal as computed in Sec. 4.3). The first constraint is to prevent small components from completely disappearing in the final segmentation. The second is to prevent components with thin parts from being split up into multiple segments. Fig. 7c shows the result after splitting of components, whereas Fig. 7d shows the final segmentation after edge erosion. Note that the presented solution for handling corners is not necessarily the only one, but gives satisfactory results in practice.

5. Results and Discussion

We have implemented our algorithm in C++ and ran it on an Intel Core 2 Extreme 3 GHz (using 1 CPU), with 2 GB of RAM. The input shapes have resolutions ranging up to 300^3 voxels. Figure 8 shows the resulting segmentations for several shapes. Most segments are placed as expected and no over-segmentations are produced. In the Plane shape each wing is split into an upper and lower part. On the tips of the wing no segments are placed because the resolution in the tips is too limited for the skeleton to reach into. We further observe that sharp and straight borders are produced for soft edges, such as in the Gear and Bird shapes.

Our segmentation approach has several desirable properties, as follows. First, we can detect very soft and vanishing edges, because we detect edges as gaps in the feature collection. These gaps arise when we threshold the skeleton's importance measure ρ, which represents geodesic distance between feature points. For both weak and strong edges, setting a threshold of τ ensures gaps of at least width τ, regardless of the edge strength. Figure 9 shows the detected edges E and the resulting segmentations of a smooth H-shape consisting of ellipsoids and a smooth X-shape consisting of two

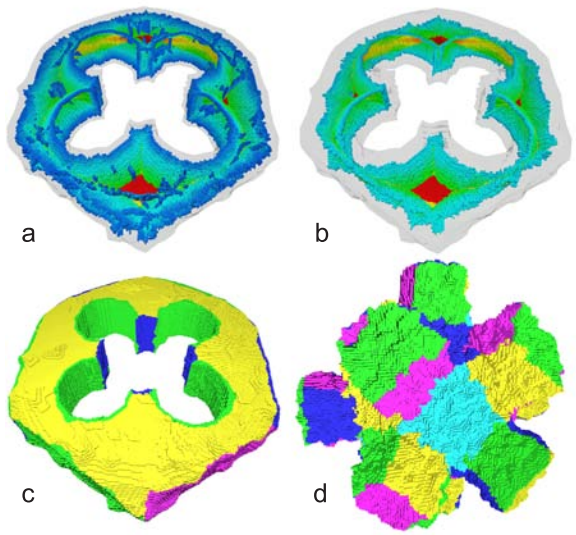

Figure 10: *Segmentation of two noisy shapes. $\mathcal{S}_0(\Omega)$ (a), $\mathcal{S}_{\tau=15}(\Omega)$ (b). Resulting segmentation (c). Segmentation of noisy Sea-mine (d).*

bent boxes with vanishing edges. We observe no spurious segments for both cases. For the H-shape, each ellipsoid is split in two. The vanishing edges of the X-shape are detected well, and sharp, straight, segment borders are generated for them by the edge erosion step.

Second, our method is capable of handling shapes with boundary noise, because it uses the simplified surface-skeleton. We assume the noise to be uniform, so that the scale parameter τ can be set to such a value that the skeleton does not contain any spurious parts due to noise. This parameter is intuitive as it is based on geodesic distance on the boundary: all feature points that are within a distance of at most τ to their associated feature points are removed. Figure 10c,d shows the segmentation of the noisy Tap Threads and Sea-mine shapes respectively. For the Tap Threads shape, both the non-simplified (Fig. 10a) skeleton, and the simplified skeleton (Fig. 10b) that is used in the segmentation are shown. This also shows that we can handle shapes with holes. These noisy shapes would be difficult to handle using traditional curvature-based segmentation approaches. Nevertheless, we should state that for very noisy objects the feature collections become too sparse, potentially resulting in over-segmentation.

Third, multiscale segmentations can be created by increasing the scale parameter τ beyond the noise level. In Figure 11 the "Sea-mine" object, consisting of a polyhedron with attached boxes, is simplified at three different simplification levels (τ=12,25,50). Fore- and background skeletons are shown in the left column (a rainbow color-map encodes the importance measure). The resulting segmentations are shown in the right column. At level τ=12, all patches of the

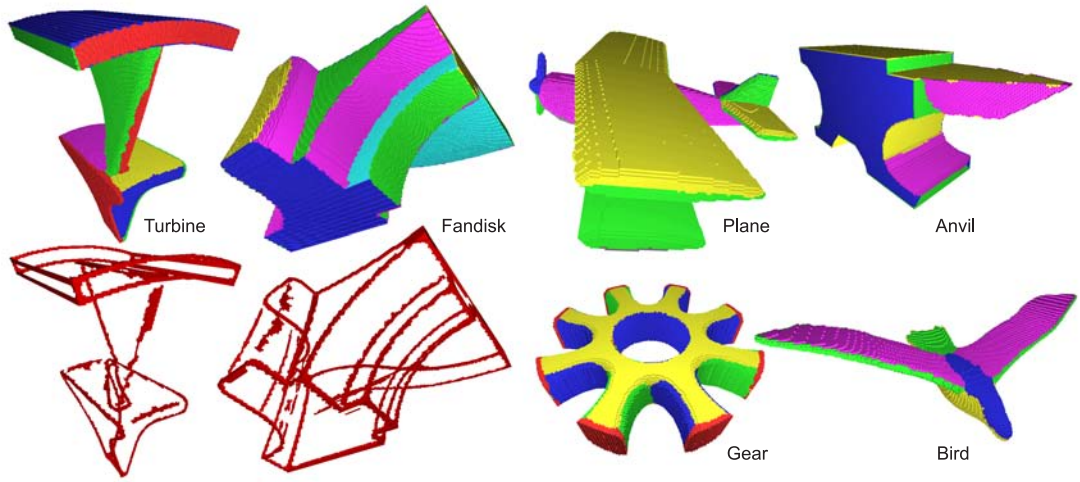

Figure 8: *Segmentations of several shapes. The detected edges are shown for the Turbine and Fandisk shapes.*

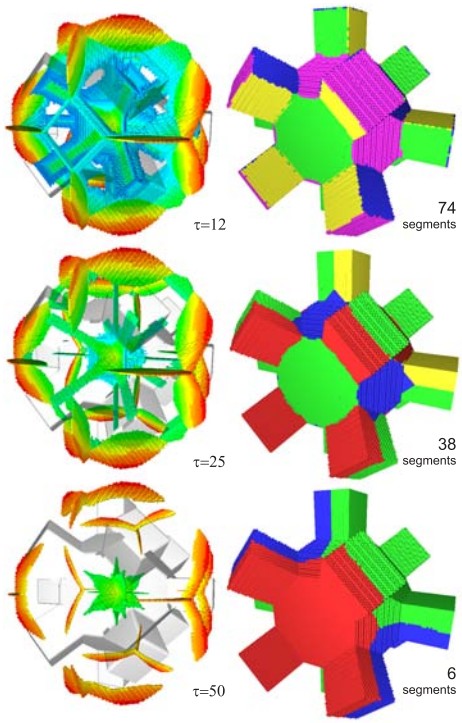

Figure 11: *Multiscale segmentations of the Sea-mine shape. Fore-and background skeletons (a). Corresponding increasingly coarse segmentations (b).*

Sea-mine are identified. At τ=25, only the foreground skeleton center sheets remain inside each box, producing two segments for each attached box. As we can see here, a property of our approach is that the borders between coarse segments not necessarily lie on curvature creases of the surface.

Table 1 show some indicative running times. Skeletonization takes the most time due to the large amount of shortest-path computations involved. Computing the background skeleton takes the longest, as the amount of background voxels is typically much larger than the amount of foreground voxels. The time needed for detecting the edges and performing the segmentation is comparatively small. Currently, the memory consumption is higher than necessary, because we store all intermediate results for debugging purposes. Because the skeleton can be computed in parallel for each voxel, only the voxel shape, its feature transform, and the feature collection need to be kept in memory, reducing the memory consumption up to an estimated 10 times.

Table 1: *Timing measurements in seconds for foreground skeleton (fg $\mathcal{S}$), background skeleton (bg $\mathcal{S}$), and segmentation time. Peak memory usage in megabytes.*

shape	dimension	fg $\mathcal{S}$	bg $\mathcal{S}$	segm	mem
Fandisk	295x177x276	187	233	30	1080
Sea-mine	203x204x176	21	115	29	640
X-shape	119x174x296	35	101	16	450

Currently, a limitation of our method is it works on voxel shapes. However, we believe an adaptation to polyhedral shapes is feasible. In [DS06], the importance measure that we use is computed for polyhedrons. The feature collection and their geodesics distances can also be computed on meshes. Another limitation is that patches should be completely bordered by convex and/or concave creases in order to be identified as segments, so our method is more suitable for geometric than for organic shapes, for which a part-type segmentation would be better choice. Finally, for parts of the object that are thin, we might not detect weak edges. As indicated in Sec. 4.4, inscribed balls for weak edges require larger radii than for strong edges. Unfortunately, thin parts of the object do not allow inscribed balls with a high radius, and thus we might not detect edges in such cases.

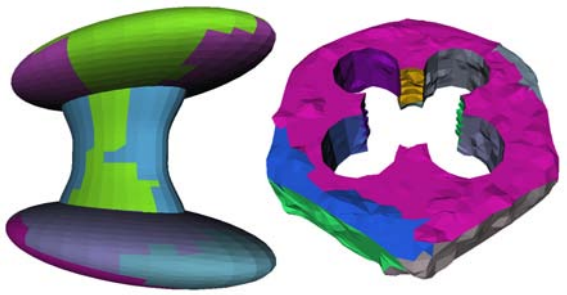

Figure 12: *Results of HFP (cf. Fig. 9b and Fig. 10c).*

5.1. Comparison

We compare results of our method with those of "HFP", a state-of-the-art segmentation method of Attene et al. [AFS06] for which the software is available on the internet. HFP clusters faces hierarchically based on fitting plane, sphere, and cylinder primitives. A binary tree of clusters is built in 10 seconds, after which the desired number of segments can be selected by the user. For each shape, we selected the amount of clusters in HFP such that it is equal to the amount of segments our method has produced. In our method the amount of segments cannot be controlled directly, but automatically follows from the chosen skeleton simplification level. For shapes which only have strong creases, HFP delivers equivalent results. Figure 12 shows HFP's clustering for the H-shape, which has soft edges, and the Tap Threads, which has boundary noise. For the H-shape, our method splits each of the ellipsoids into two symmetrical halves corresponding to the skeleton sheet's top and bottom sides (Fig. 9b). HFP's clusters on the other hand do not reflect the symmetry of the shape at all. For the Tap Threads, HFP produces some spurious segments due to the boundary noise. Our method does not suffer from the noise (Fig. 10c) as the simplified skeleton is unaffected by it. These examples indicates that using skeletal information can be beneficial in shape segmentation.

6. Conclusion

We presented a new edge-detection and patch-type segmentation method for voxel shapes. We use the simplified skeleton for detecting the edges, instead of using curvature information explicitly, which makes our method in principle more robust for voxelized and noisy shapes. The user sets a single user-parameter to specify the amount of noise. This parameter is intuitive, as it specifies geodesic length, and can also be used to create multiscale segmentations. Finally, sharp segment borders are generated for soft and vanishing edges.

Acknowledgement

This work was supported by the Netherlands Organization for Scientific Research (NWO) under grant number 612.065.414.

References

[AB98] AMENTA N., BERN M.: Surface reconstruction by voronoi filtering. In *SCG '98: Proceedings of the fourteenth annual symposium on Computational geometry* (1998), pp. 39–48.

[AFS06] ATTENE M., FALCIDIENO B., SPAGNUOLO M.: Hierarchical mesh segmentation based on fitting primitives. *The Visual Computer 22*, 3 (2006), 181–193.

[BdBS09] BORGEFORS G., DI BAJA G. S., SVENSSON S.: Decomposing digital 3d shapes using a multiresolution structure. In *Proc. of Discrete Geometry for Computer Imagery (DGCI'99)* (2009), vol. 1568, Springer Berlin / Heidelberg, p. 19.

[CGST04] CLARENZ U., GRIEBEL M., SCHWEITZER M. A., TELEA A.: Feature sensitive multiscale editing on surfaces. *The Visual Computer 20*, 5 (2004), 329–343.

[DS06] DEY T. K., SUN J.: Defining and computing curve-skeletons with medial geodesic function. In *Proc. of Eurographics Symposium on Geometry Processing* (2006), pp. 143–152.

[GWH01] GARLAND M., WILLMOTT A., HECKBERT P.: Hierarchical face clustering on polygonal surfaces. In *Proc. of ACM Symposium on Interactive 3D Graphics* (2001), pp. 49–58.

[HBK01] HISADA M., BELYAEV A. G., KUNII T. L.: A skeleton-based approach for detection of perceptually salient features on polygonal surfaces. *Computer Graphics Forum 21*, 4 (2001), 689–700.

[KS93] KIRYATI N., SZÉKELY G.: Estimating shortest paths and minimal distances on digitized three-dimensional surfaces. *Pattern Recognition 26* (1993), 1623–1637.

[Mul92] MULLIKIN J. C.: The vector distance transform in two and three dimensions. *CVGIP: Graphical Models and Image Processing 54*, 6 (1992), 526–535.

[MW99] MANGAN A. P., WHITAKER R. T.: Partitioning 3d surface meshes using watershed segmentation. *IEEE Transactions on Visualization and Computer Graphics 5*, 4 (1999), 308–321.

[PDR08] PROVOT L., DEBLED-RENNESSON I.: Segmentation of noisy discrete surfaces. In *Lecture Notes in Computer Science* (2008), vol. 4958, pp. 160–171.

[PH02] PROHASKA S., HEGE H.-C.: Fast visualization of plane-like structures in voxel data. In *Proc. IEEE Visualization* (2002), pp. 29–36.

[PSS*03] PIZER S. M., SIDDIQI K., SZÉKELY G., DAMON J. N., ZUCKER S. W.: Multiscale medial loci and their properties. *Int. Journal of Computer Vision 55*, 2-3 (2003), 155–179.

[RT07] RENIERS D., TELEA A.: Skeleton-based hierarchical shape segmentation. In *Proc. of the IEEE Int. Conf. on Shape Modeling and Applications (SMI'07)* (2007), pp. 179–188.

[RVT08] RENIERS D., VAN WIJK J. J., TELEA A.: Computing multiscale curve and surface skeletons of genus 0 shapes using a global importance measure. *IEEE Transactions on Visualization and Computer Graphics 14*, 2 (2008), 355–368.

[Sha04] SHAMIR A.: A formulation of boundary mesh segmentation. In *Proceedings of the 2nd International Symposium on 3DPVT* (2004).

[ZTS02] ZUCKERBERGER E., TAL A., SHLAFMAN S.: Polyhedral surface decomposition with applications. *Computers & Graphics 26*, 5 (2002), 733–743.