Computing Multiscale Curve and Surface Skeletons of Genus 0 Shapes Using a Global Importance Measure

Dennie Reniers, Jarke J. van Wijk, *Member, IEEE,* and Alexandru Telea

***Abstract*— We present a practical algorithm for computing robust, multiscale curve and surface skeletons of 3D objects of genus zero. Based on a model which follows an advection principle, we assign to each point on the skeleton a part of the object surface, called the *collapse*. The size of the collapse is used as a uniform importance measure for the curve and surface skeleton, so that both can be simplified by imposing a single threshold on this intuitive measure. The simplified skeletons are connected by default, without special precautions, due to the monotonicity of the importance measure. The skeletons possess additional desirable properties: They are centered, robust to noise, hierarchical, and provide a natural skeleton-to-boundary mapping. We present a voxel-based algorithm that is straightforward to implement and simple to use. We illustrate our method on several realistic 3D objects.**

***Index Terms*— Surface skeletons, curve skeletons, importance measure**

I. Introduction

SKELETONS are compact shape descriptors. The skeleton, or medial axis, of a 2D object can be defined as the centers of maximally inscribed discs [1]. In 3D, this definition can be extended to the centers of maximally inscribed balls. Alternative definitions are the set of interior points that have at least two closest points on the object surface, or the first-order singularities of the distance-to-boundary field, or distance transform (DT). Using these definitions for a 3D object, we obtain the so-called *surface skeleton* $\mathcal{S}$, or medial surface, which consists of a set of 2D manifolds, or skeletal sheets, and a set of 1D curves. The skeleton points, together with their distance to the 3D object boundary, define the medial surface transform (MST), which can be used for volumetric animation [2], surface smoothing [3], or topological analysis used in shape recognition, registration, simplification, or feature tracking.

A different type of skeleton is the *curve skeleton* $\mathcal{C}$, which consists, for a 3D object, of 1D curves locally centered with respect to the object boundary. Curve skeletons have a low dimensionality which makes them suitable for "stick-figure" object representations in 3D animation, morphing, and geometric processing [4]. In contrast to surface skeletons, curve skeletons lack a formal and unanimously accepted definition.

It is widely accepted that 2D skeletons and 3D surface and curve skeletons should fulfill several desirable properties [3], [4]: homotopic, invariant under isometric transformations, reconstruction, thin, centered, junction detective, robust, efficient to compute, and hierarchical. While 2D skeletonization is a well-studied problem, computing 3D skeletons with all of the above properties is still an open problem.

The robustness and hierarchical properties deserve further attention. The *robustness* property means that the skeleton is insensitive to small-scale boundary noise. This is needed in all practical applications, as the boundary discretization, whether voxel or polygon based, inevitably introduces noise. The *hierarchical* property means computing a multiscale of nested skeletons, each one representing the shape details on a different spatial scale. Hierarchical skeletons are typically also robust, since boundary noise can be seen as small-scale object details. To the above, we add the *uniformity* and *inclusion* properties. The uniformity property requires that both curve and surface skeletons are computed and treated similarly. For example, if a method provides a parameter for multiscale skeleton simplification, this parameter should be the same for both curve and surface skeletons. The inclusion property requires that $\mathcal{C}$ is a subset of $\mathcal{S}$, motivated by the fact that curve skeletons can be considered a limit case of surface skeletons for objects with local circular symmetry (e.g. a tube).

In this paper, we introduce the *collapse measure*, a novel importance measure for computing both curve and surface skeletons of 3D shapes. Our measure possesses all the desirable properties mentioned earlier: It is robust to compute on complex, noisy 3D shapes, always delivers connected skeletons, its thresholding produces a hierarchy of nested skeletons describing the shape at different scales, treats surface and curve skeletons uniformly, and is simple to implement. Additionally, our collapse measure has a geometrically-motivated definition which makes its use in practice intuitive. Conceptually, the collapse measure is based on an advection process which generalizes to 3D the global feature-distance measure used to compute 2D skeletons by several authors [5]–[7]. Intuitively, this can be explained as follows. Mass, initially located on the object boundary, is advected onto and then along the skeleton. The collapse measure of an object point is the amount of mass advected through that point. All object points are treated equally, yielding a uniform collapse measure for non-skeleton, surface skeleton, and curve skeleton points. Hence, our method merges the skeleton detection and simplification in a single step. We present a voxel-based algorithm that computes an approximation of the collapse measure. The algorithm does not simulate the advection process explicitly, but computes

The authors are with the Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, PO Box 513, 5600 MB Eindhoven, The Netherlands. E-mail: d.reniers@tue.nl, vanwijk@win.tue.nl, alext@win.tue.nl.

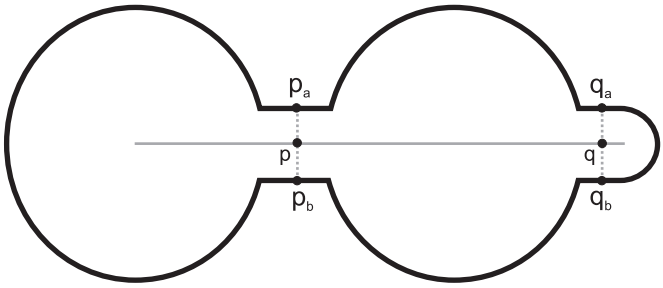


Fig. 1. A 2D shape and its skeleton. Points $p, q \in \mathcal{S}$ and their feature points p_a, p_b and q_a, q_b respectively have locally identical configurations. Globally however, p is considered more important than q.

the collapse measure directly. We demonstrate the algorithm on several real-world examples.

The structure of this paper is as follows. In Section II we give an overview of related work. In Section III we discuss our informal model in 2D, show its equivalence to the feature-distance measure, and propose an extension to 3D. The informal 3D model motivates our algorithm, which is presented in Section IV. Section V presents the results of the algorithm. Section VI discusses how our method fulfills the desirable skeleton properties, and compares it with existing methods. Section VII concludes this paper.

II. Related work

Methods for computing medial axes and skeletons are commonly classified into four groups [8]: thinning or erosion [9], geometric methods [5], [10], [11], distance-field methods [7], [12]–[14], and general-field methods [15], [16]. Such methods usually work either on a raster or voxel-based object representation, or on a boundary sampling. Thinning methods iteratively remove boundary voxels as long as connectivity is not violated. Removing voxels in order of a true distance-to-boundary fashion enforces the centeredness criterion [17]. Geometric methods often use the Voronoi-diagram of a boundary sampling and deliver a polygonal skeleton description [5]. These methods are quite complex to implement in 3D and the resulting surface skeletons can be difficult to handle. Other geometric methods compute skeletons using edge collapses [10], from a mesh segmentation [18], or by sphere sweeping [11]. Distance-field methods detect skeletons as singularities of various types of the distance transform of the object [3], [13] and have been efficiently implemented using graphics hardware [14], [19], [20]. In general-field methods alternatives for the distance transform are used [15], [16]. These methods are generally less sensitive to noise, due to the averaging effect that the field has.

A key aspect of any skeletonization method is its ability to produce robust, hierarchical skeletons. At the core of any such method is an importance measure for the skeleton points. Such measures have the desirable property that small values correspond to skeleton points that represent object details, whereas large values correspond to points that represent the object's main structure. Simplified skeletons can be obtained by pruning the skeleton using the importance measure [21].

We distinguish between *local* and *global* measures. Local measures use only object properties from a small, limited vicinity of every object point, and are thus incapable of distinguishing between locally identical configurations, such as those sketched in 2D in Fig. 1. Thresholding local measures usually delivers disconnected skeletons. Connectivity can be enforced, but this requires extra work [13], [22], [23], and often makes the simplification non-continuous [21], thus less intuitive. In contrast, global measures monotonically increase from the skeleton boundary to its center, and describe a skeleton point's importance to representing the whole object. Pruning global measures by simple thresholding always delivers connected skeletons.

In 2D, the *feature-distance* measure, also called collapsed boundary measure, is a simple, robust and efficient to compute global importance measure used both for polygonal and pixel object representations by several authors [5]–[7]. Fig. 1 sketches this measure: The importance of the skeleton point p equals the length of the smaller boundary curve delimited by points p_a and p_b, where p_a, p_b are the points at minimum distance of p: the so-called *feature points*. By this measure, the importance of q is regarded lower, which agrees with intuition. In 3D, the θ-SMA method uses a local measure based on the angle between the feature (or anchor) points [19], [24]. Combining the angle with the distance-to-boundary value yields a more robust measure [22]. Siddiqi et al. propose a range of methods that use a local divergence-based measure to detect and classify skeleton points in 3D [13], [25], [26]. A similar approach using first-order moments is given by Rumpf and Telea [3]. Pizer et al. [27] give a comparison of methods that compute multiscale skeletons. The general-field method of Cornea et al. [16] delivers smooth curve skeleton hierarchies. However, only a limited number of hierarchy levels is generated and the different levels are not treated uniformly. Recently, Dey and Sun [28] presented a method that computes robust curve skeletons. Similar to Prohaska et al. [29], they use shortest-geodesic length between feature points as an importance measure for the surface skeleton points. This is a global importance measure, used to find the middle of the surface skeleton. They obtain a curve skeleton by using the local divergence-measure of [13] on this geodesic-length measure.

Related to the curve skeleton is the Reeb graph: a 1D structure encoding geometrical and topological properties of 3D shapes [30], [31]. It is constructed by following the evolution of the level sets of a real-valued function on the object boundary. The critical points of this function are represented by nodes in the Reeb graph, and the edges represent connections between critical points. The real-valued function, defined on the object boundary, should be chosen carefully to obtain a Reeb graph that is suitable for the application at hand. In contrast, our approach uses curve and surface skeleton definitions to obtain a real-valued function on the skeleton, the importance measure, which is then used to simplify it.

In this paper, we propose a global importance measure for computing multiscale surface and curve skeletons, by generalizing the 2D feature-distance measure to 3D. At the core of our measure is an advection principle. In the next

section we present the conceptual model.

III. Conceptual Model

Let Ω be a d-dimensional object with boundary $\partial\Omega$. The distance transform $D : \Omega \rightarrow \mathbb{R}$ is defined as $D(p) = \min_{k\in\partial\Omega}\ dist(p,k)$, where $dist$ is the Euclidean distance in $\mathbb{R}^d$. The feature transform $F : \Omega \rightarrow \mathcal{P}(\partial\Omega)$, where $\mathcal{P}$ is the power set, assigns to a point p in Ω the set of points on $\partial\Omega$ that are at minimum distance of p, the *feature points* of p:

$$F(p \in \Omega) = \{x \in \partial\Omega \mid dist(p,x) = D(p)\}. \qquad (1)$$

The skeleton $\mathcal{S}$ of Ω is defined as those points in Ω that have at least two feature points:

$$\mathcal{S}(\Omega) = \big\{x \in \Omega \;\big|\; |F(x)| \geq 2\big\}. \qquad (2)$$

In the generic case, skeleton points have exactly two feature points. In the non-generic cases, which are limit cases, skeleton points have three or more feature points. When $d = 3$, $\mathcal{S}$ is called the surface skeleton. An equivalent definition of the skeleton is as the points where D is non-differentiable. D and ∇D can be computed using several methods, e.g., by solving the eikonal equation $\|\nabla D\| = 1$ using level set methods such as the fast marching method [32], graphics hardware techniques [14], [33], or a variety of discrete propagation methods (for an overview of these see [34]).

In the remainder of this section we present a conceptual model that appeals to intuition, in order to motivate the algorithm that we further present in Section IV. At all times, we assume the generic case of a skeleton point having exactly two feature points. The non-generic cases of points having three or more feature points are dealt with in Section IV-C. We proceed as follows. In Section III-A we first describe an informal 2D model using an advection principle that yields a measure equivalent to the feature-distance measure, and as such has the desirable property of monotonicity. In Section III-B we extend this model to 3D, such that we obtain a monotonic 3D importance measure.

A. Collapse Measure in 2D

Let Ω be a 2D shape. The skeleton of Ω can be defined as those points where the distance transform D is non-differentiable. Governed by the eikonal equation, the trajectory of each boundary point follows the gradient field ∇D and ends at the skeleton $\mathcal{S}$, where ∇D is undefined. We aim to define a flow vector field $\mathbf{F}$ that extends ∇D to the skeleton, such that trajectories do not end at $\mathcal{S}$. On non-skeleton points, $\mathbf{F}$ is equal to the distance field gradient:

$$\mathbf{F} = \nabla D \qquad \text{on } \Omega \setminus \mathcal{S}. \qquad (3)$$

Instead of ending at $\mathcal{S}$, we let the trajectories end in a unique point on $\mathcal{S}$, called the *root* R. The precise definition of R follows later (Eq.6). If we assume that the object is of genus 0, the skeleton is also of genus 0, because the skeleton is homotopic to the shape [35]. Assuming genus 0, the skeleton of Ω is a tree, it has no loops, so that the continuation of the trajectories on $\mathcal{S}$ toward R is predetermined. The flow field $\mathbf{F}$ on $\mathcal{S}$ can be defined in only one way, namely tangent to the skeleton:

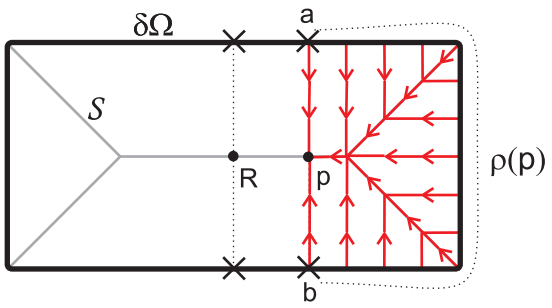


Fig. 2. A rectangle shape Ω, its skeleton $\mathcal{S}$, the root R, a skeleton point p, and p's feature points a, b. The collapse of p is formed by the origins of the trajectories (red) through p.

$$\mathbf{F}: \quad \mathbf{F} \cdot \mathbf{n}_{\mathcal{S}} = 0 \qquad \text{on } \mathcal{S} \setminus R, \qquad (4)$$

where $\mathbf{n}_{\mathcal{S}}$ represents the normal to the skeleton, and $\mathbf{F}$ is chosen in the direction of the root R. All trajectories end in the root, where we define $\mathbf{F}(R) = \vec{0}$.

Consider a discretization of the object boundary into a large but finite number of particles evenly distributed on the boundary $\partial\Omega$. The advection of the particles is governed by $\mathbf{F}$. The trajectories of these particles together form a *trajectory tree* of which R is the root. When the number of particles approaches infinity, the tree covers the whole of Ω, so that each point $p \in \Omega$ is on the trajectory tree. On non-skeleton points, the trajectories do not intersect because they follow ∇D. On skeleton points the trajectories necessarily overlap, as the skeleton of a 2D shape is only a 1D structure. Consequently, each object point $p \in \Omega$ has one or more trajectories going through it that originate on the object boundary. That part of the boundary that is formed by the origins of the trajectories (tree leaves) going through a point p is called the *collapse* of p. The collapse at a non-skeleton point is just its single feature point. The collapse at a skeleton point are its two feature points, plus the origins of the trajectories incoming from the skeleton and passing through that point. Because the skeleton is a tree structure, a point $p \in \mathcal{S}$ divides $\mathcal{S}$ into two subtrees. The collapse at p is the sum (or integration in the limit) of the feature points of the subtree not containing R, which means that the collapse at p is a compact boundary arc.

Based on the construction presented above, we can now define our importance measure ρ on Ω. The importance measure tells us how important an object point p is in representing the object boundary: the more trajectories through p, the more important p is. In case of an infinite number of particles this presents a problem, as the importance for a skeleton point would be infinite, so we use the size of the collapse instead:

$$\rho(p) : \Omega \rightarrow \mathbb{R} \qquad \rho(p) = |\text{ collapse at } p\ |. \qquad (5)$$

We call this importance measure ρ the *collapse measure*. We can now define the root R as that point on $\mathcal{S}$, having two neighboring points $x, y \in \mathcal{S}$, such that the collapse measure at x is equal to the collapse measure at y:

$$R(\Omega): \quad \rho(x) = \rho(y). \qquad (6)$$

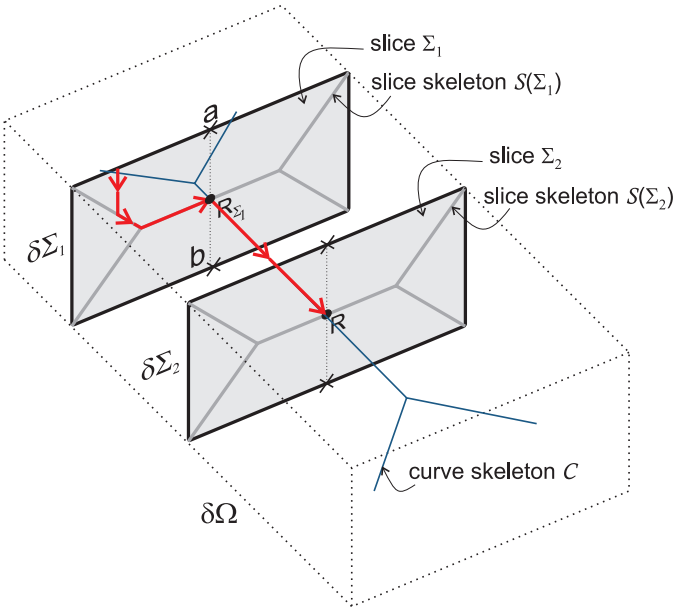


Fig. 3. A box shape Ω, (part of) the curve skeleton $\mathcal{C}$, and two slices Σ_1, Σ_2. One trajectory is shown in red.

Hence, the root is in the middle of the skeleton with respect to the amount of incoming trajectories. In Fig. 2, the feature points of the root R can be seen to divide the boundary exactly into two components of equal length.

Intuitively, the above field $\mathbf{F}$ defines an advection of particles, or mass, from the boundary onto and then along the skeleton, up into the root R. The importance of a point is the amount of mass that passes through that point on its way to the root. The collapse measure has a low value at the non-skeleton points $\Omega \setminus \mathcal{S}$. On the skeleton, ρ increases while approaching the root, as more and more trajectories overlap. The collapse measure is monotonic, because the trajectories form a tree. It can be easily seen that the value of ρ on $\mathcal{S}$ is equal in the limit to the well-known feature-distance measure [5]–[7], which assigns to each skeleton point the smaller boundary-arc length between two feature points. Using our construction, we obtain the same measure. In Fig. 2, a sparse subset of the incoming trajectories for a point p is shown. These trajectories clearly originate from the smaller boundary arc between the feature points of p. Indeed, our collapse measure $\rho(p)$ is equal to the length of this arc.

B. Extension to 3D

Our goal is to construct an importance measure in 3D that is monotonic. Then, we can compute robust 3D skeletons simply by thresholding this measure, just as is done in 2D using the feature-distance measure. We extend our informal 2D advection model from the previous section to 3D. This informal model for 3D shapes motivates our algorithm from Section IV.

Again, we want to construct a flow vector field $\mathbf{F}$, yielding a trajectory tree that covers the whole volume of Ω. As in 2D, we require that the trajectories first follow ∇D:

$$\mathbf{F} = \nabla D \qquad \text{on } \Omega \setminus \mathcal{S}. \tag{7}$$

The definition of $\mathbf{F}$ should be extended to $\mathcal{S}$ and there should be a single root $R \in \mathcal{S}$. In contrast to the 2D case, there are more possibilities to define $\mathbf{F}$ as $\mathcal{S}$ is now a 2D structure (Eq. 2): the surface skeleton, generally consisting of 2D curved manifolds. We present a construction of $\mathbf{F}$ that yields a measure that is not only monotonic, but has the nice property that it differentiates between the surface and curve skeleton.

The key idea is to slice the 3D shape into 2D simply-connected and possibly curved manifolds, called *slices*, and use our 2D model on these slices to define $\mathbf{F}$ in a divide-and-conquer fashion. Let Σ denote a slice with compact boundary $\partial\Sigma \subset \partial\Omega$. We define $\mathbf{F}$ on a slice Σ using the 2D model. Because $\mathbf{F}$ is already defined on $\Omega \setminus \mathcal{S}$ by Eq. 7, the slices must be parallel to ∇D, so that trajectories starting on a slice boundary stay in their respective slice. If we denote by $\mathbf{n}_\Sigma$ the normal to slice Σ, then we require:

$$\mathbf{n}_\Sigma \cdot \nabla D = 0 \qquad \text{on } \Sigma \setminus \mathcal{S}. \tag{8}$$

This implies that the 1D skeleton $\mathcal{S}(\Sigma)$ of a slice Σ is at the intersection of the slice with the surface skeleton $\mathcal{S}(\Omega)$ of the object:

$$\mathcal{S}(\Sigma) = \Sigma \cap \mathcal{S}(\Omega), \tag{9}$$

because each point $p \in \mathcal{S}(\Sigma)$ is connected to its two feature points by two trajectories originating on the slice boundary $\partial\Sigma$. Now, $\mathbf{F}$ on the slice skeleton, and thus on $\mathcal{S}(\Omega)$, can be defined according to our 2D model (Eq. 4), namely tangent to the slice skeleton, $\mathbf{F}(\mathcal{S}(\Sigma)) \cdot \mathbf{n}_\Sigma = 0$, in the direction of the slice root R_Σ. The slice root R_Σ is defined similar to the root in the 2D model: its two feature points divide $\partial\Sigma$ into two boundary arcs of equal length.

Applying the 2D model to each slice, we obtain one trajectory tree per slice, yielding a trajectory forest. In order to merge this forest into a single trajectory tree for the whole volume Ω, the trajectories starting on the slice boundary $\partial\Sigma$ should not stop in the slice root R_Σ, but should continue to a single global root R, which we will define later. This poses the final requirement on the slices, namely that the roots of all the slices together form a 1D connected structure. We call this structure the curve skeleton $\mathcal{C}$, as it is connected and can be considered in the center of the object, as it consists of individual slice roots:

$$\mathcal{C} = \{R_{\Sigma_i}\}_i. \tag{10}$$

We now define a slice set satisfying the requirements above. The key idea is that we choose each slice Σ such that its root R_Σ has two shortest geodesics on the object surface $\partial\Omega$ between its two feature points $F(R_\Sigma)$. The two shortest geodesics form the boundary $\partial\Sigma$ of the slice. In this way we are sure that the slice roots form a connected 1D structure, satisfying Eq. 10. Indeed, we use here the result of Dey and Sun [28]: They used as definition for their connected curve skeleton those points on $\mathcal{S}$ that have two shortest geodesics between their two feature points. A second reason why this definition fits our model nicely is that R_Σ is indeed the root of its slice: the two shortest geodesics between $F(R_\Sigma)$ forming the slice boundary $\partial\Sigma$ are necessarily of equal length. A

slice is completely characterized by its boundary consisting of two shortest geodesics, together forming a Jordan curve, and the fact that it is parallel to ∇D. What remains to be shown is that this definition yields simply-connected slices, in specific, that there are no holes at $\mathcal{S}(\Sigma)$. Extensive empirical study suggests this, as checked by us on several tenths of different 3D voxelized objects, sampled on various resolutions. However, we cannot prove this, and leave it as a conjecture.

Our recursive approach of first slicing the object and then applying our 2D model to the slices yields a hierarchically structured trajectory tree. Consider a trajectory starting on the object boundary $\partial\Omega$ and slice boundary $\partial\Sigma$. It first goes straight to the slice skeleton while staying in its slice Σ. The collapse at such a non-skeleton point is a single point on the boundary. After arriving on the slice skeleton, located on the surface skeleton $\mathcal{S}$, the trajectory follows the 1D slice skeleton toward the slice root $R_\Sigma \in \mathcal{C}$, overlapping more and more trajectories from the same slice on its way. The collapse at a slice skeleton point is similar to the collapse at a skeleton point in the 2D model: a boundary arc on the slice boundary. Without considering trajectories coming from other slices, the collapse at the slice root R_Σ is the slice boundary $\partial\Sigma$. After trajectories arrive at the slice root, located on the curve skeleton, they continue along the curve skeleton toward the global root $R \in \mathcal{C}$, joining the trajectories coming from other slices. Hence, the collapse at a curve skeleton point is an integration of slice boundaries: an object-surface area. Indeed, the Jordan curve theorem states that a Jordan curve on a surface of genus 0 divides that surface into two components. In our construction, the collapse at a curve skeleton point R_Σ is the area of the smaller component generated by the slice boundary and Jordan curve $\partial\Sigma$. We still need to define the global root R. Analogous to the 2D root, which is defined to be in the middle of the skeleton with respect to the collapse measure (Eq. 6), the global root R is the slice root $R_\Sigma \in \mathcal{C}$ of the slice whose boundary divides the object surface into two components of equal area.

Figure 3 exemplifies our construction. It shows a box shape, two slices Σ_1, Σ_2, and one trajectory starting on $\partial\Sigma_1$. The trajectory first goes to $\mathcal{S}(\Sigma_1)$, and then to R_{Σ_1} while staying in the slice. Point R_{Σ_1} is the root of Σ_1, as the feature points a, b of R_{Σ_1} divide the slice boundary $\partial\Sigma_1$ into two components of equal length: each component is a shortest geodesics between between a, b. The trajectory leaves the slice in R_{Σ_1} and continues along the curve skeleton $\mathcal{C}$. Finally, the trajectory ends in the global root $R \in \mathcal{C}$, which coincides with the root of slice Σ_2. Point R is chosen as the global root because its slice boundary divides the boundary into two components of equal area.

As in 2D, the tree structure of the trajectories ensures that the collapse measure is also monotonic in 3D by construction. It is furthermore important to note that the collapse measure of a curve skeleton point p can be computed independently from other points simply by taking the area of the smaller object surface component generated by the slice boundary, consisting of two shortest geodesics between the two feature points a, b of p. Likewise, the collapse measure of a surface skeleton point can be directly computed by taking the length of the shortest geodesic between a, b on the object surface. We use these properties in our algorithm in Section IV.

C. Skeleton simplification

After we have computed ρ for all points in the object Ω, we obtain a simplified skeleton by thresholding ρ with a desired importance value τ. To easily handle objects of different sizes, we first normalize ρ to $[0..1]$ by dividing it by its maximal value, i.e. half the object surface area (see Sec. III-A):

$$\mathcal{S}_\tau = \{p \in \Omega \mid \rho(p) > \tau \frac{1}{2}\text{area}(\partial\Omega)\}. \tag{11}$$

These simplified skeletons are connected by default, as the collapse measure ρ is monotonic.

One observation must be made concerning the interpretation of the collapse measure ρ. This measure is essentially of a higher dimensionality on the curve skeleton $\mathcal{C}$ than on the remainder of the surface skeleton $\mathcal{S} \setminus \mathcal{C}$. Whereas $\rho(\mathcal{C})$ denotes a collapsed area, $\rho(\mathcal{S} \setminus \mathcal{C})$ denotes a collapsed curve length. This means that, when we increase τ, the surface skeleton typically disappears completely even before the curve skeleton starts to get simplified. This is desirable in applications where the curve skeleton is considered more important. However, in other applications, this behavior may not be desired, as we shall show in Section VI. For such applications, we can "equalize" ρ by reducing the dimensionality of $\rho(\mathcal{C})$ from an area to a length by taking its square root. We denote the *equalized* skeleton by $\mathcal{S}'_\tau$. In contrast to $\mathcal{S}_\tau$, the equalized skeleton gets simplified uniformly both in its curve as in its surface components when τ is increased. Note that this is just one of the possible ways to equalize ρ. Other options leading to other applications are open to further study.

IV. Algorithm

We present now a voxel-based algorithm that computes simplified skeletons, by first computing the collapse measure and then thresholding it (Eq. 11). A strong point of our algorithm is that it works on the object-boundary voxels only and not on any derived structures. In particular, there is no need to compute $\mathcal{S}$ first. It is important to note that we do not need to explicitly compute the flow vector field $\mathbf{F}$ and simulate an advection process, which would be unstable. Instead, we compute the collapse measure at surface and curve skeleton voxels directly as a length and object surface area, respectively. In Section IV-A we present an outline of the algorithm. Section IV-B discusses the implementation into more detail. Section IV-C explains how the algorithm deals with the non-generic cases.

A. Outline

Our voxel-based algorithm computes the collapse measure for each object voxel p independently from the other voxels. From the curve skeleton definition (Sec. III-B) follows that we can detect a curve skeleton point p by checking whether p admits two shortest geodesics on the object surface between its two feature points. However, in discrete space this definition is problematic. On a voxelized object surface, the length of a

1: compute F on Ω
2: **for** each $p \in \Omega$ **do**
3: $\quad \overline{F} \leftarrow \bigcup_{x,y,z \in \{0,1\}} F(p_x + x, p_y + y, p_z + z)$
4: $\quad P \leftarrow \bigcup_{a \neq b \in \overline{F}}$ shortestpath(a, b)
5: $\quad C \leftarrow \{$connected components in graph $\partial\Omega \setminus P\}$
6: $\quad$ **if** $|C| \geq 2$ **then** {curve skeleton voxel}
7: $\quad\quad \rho(p) \leftarrow$ area$(\partial\Omega) - \max_{c \in C}$ area(c)
8: $\quad$ **else** {surface or non-skeleton voxel}
9: $\quad\quad \rho(p) \leftarrow \max_{l \in P}$ length(l)
10: $\quad$ **end if**
11: $\quad \rho(p) \leftarrow \frac{\rho(p)}{\frac{1}{2}\text{area}(\partial\Omega)}$
12: **end for**
13: $\mathcal{S}_\tau = \{p \in \Omega \mid \rho(p) > \tau\}$

Fig. 4. Pseudo code of the complete algorithm.

shortest path, as discrete equivalent of a shortest geodesic, is only an approximation for the geodesic length on the original continuous surface. In practice, we will thus find only one shortest path between the two feature voxels of a curve skeleton voxel. Because of this, and in order to handle the non-generic cases as will be explained in Sec. IV-C, we opt to detect the curve skeleton differently. Instead of computing the shortest path(s) between the two feature voxels of a voxel p, we also consider the feature voxels of p's neighbors, basically "extending" the feature set of p. Shortest paths are computed between each pair of voxels in the extended feature set, yielding multiple shortest paths. For curve skeleton voxels, these shortest paths form a band around the object (see e.g. Fig. 6a), which can be considered the discrete version of the Jordan curve that the slice boundary is (Sec. III-B). Assuming that the object is of genus 0, such a band splits the object surface into two connected components if and only if p is a curve skeleton voxel, which fact we use as the detector. The collapse measure at p is the area of the smaller component generated by the band. If the band does not divide the surface into multiple components, the voxel is considered a surface skeleton voxel, and we take as collapse measure the maximum over the shortest-path lengths. After computing the collapse measure for all voxels, the simplified skeleton is computed by thresholding ρ. We detail on the implementation in the next section.

B. Implementation

The pseudo code of the algorithm is shown in Figure 4. The algorithm takes as input a binary object Ω voxelized on a regular grid. The output is the simplified skeleton $\mathcal{S}_\tau$, represented on the same voxel grid. The voxelized boundary $\partial\Omega$ is represented as a graph in which nodes are voxels. An edge in the graph corresponds to two neighboring voxels which are 26-connected, i.e share at least one corner. The object voxels Ω are 6-connected, i.e. they are connected if sharing a face. The algorithm consists of four stages: computing the (extended) feature transform, the shortest path sets, the collapse measure, and finally the simplified skeleton.

In the first stage (Fig. 4 line 1) we compute the feature transform F using [36]. Next, we compute the *extended feature set* $\overline{F}$ of a voxel p by merging the feature set of p with the feature sets of its 26-neighbors that lie in the first octant (line 3). The purpose of $\overline{F}$ is to combat two discretization problems, as outlined in Sec. IV-A. The first problem is that, while surface skeleton points always have at least two feature points in $\mathbb{R}^3$, this is not necessarily so in discrete ($\mathbb{Z}^3$) space [34]. In a box of even height, no voxels on the center surface skeleton sheet contain two feature voxels. Second, curve skeleton points have two shortest geodesics between their feature points in $\mathbb{R}^3$, but in $\mathbb{Z}^3$ the geodesic length cannot be computed exactly, and we may find only one shortest path in practice.

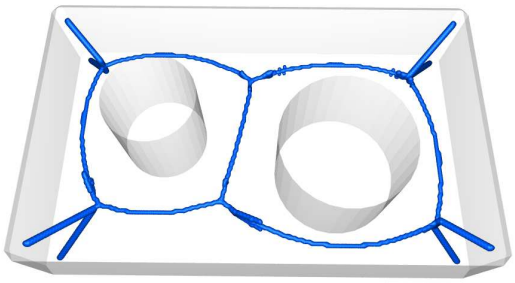


Fig. 5. Curve skeleton of a box with two tunnels computed using a variation of our algorithm.

In the second stage, we compute the set of shortest paths P between each pair of feature voxels in $\overline{F}$ (line 4). The shortest path between two feature voxels a, b is computed as a 3D chain code [37], using the A* shortest-path algorithm [38] on the boundary graph, with Euclidean distance as the search heuristic. The A* algorithm computes only one shortest path between two feature voxels a, b. This is not a problem, because we compute the set of shortest paths between each pair of feature voxels in $\overline{F}$. Although computing shortest paths between each pair of feature voxels seems computationally expensive, the size of $\overline{F}$ is typically small (≤ 8), and half of the paths are between neighboring feature voxels.

In the third stage, the collapse measure $\rho(p)$ is computed. We compute the set of connected components C in the boundary graph in which the voxels of the shortest-path set P are removed. A simple spatial subdivision scheme on the boundary is used to speed up the flood fills used. Moreover, we prevent computing the collapse for voxels that certainly are non-skeleton voxels, i.e. have only few voxels in P. If C contains two or more connected components, p is a curve skeleton voxel and the collapse at p is $\partial\Omega \setminus c$, where $c \in C$ is the largest connected component in terms of voxel count. The collapse measure is equal to the number of voxels in $\partial\Omega \setminus c$ (line 7). We could use a surface-area estimator to better approximate the collapse area on the original continuous $\partial\Omega$, but taking the cardinality of the collapse yields good results in practice. If C contains only one connected component, p is either a surface skeleton or non-skeleton voxel. In that case, the collapse at p is the longest shortest-path in P. As collapse measure we could take the amount of voxels in this longest path. However, using the length estimator of [37] instead gives better results in practice (line 9). Although the

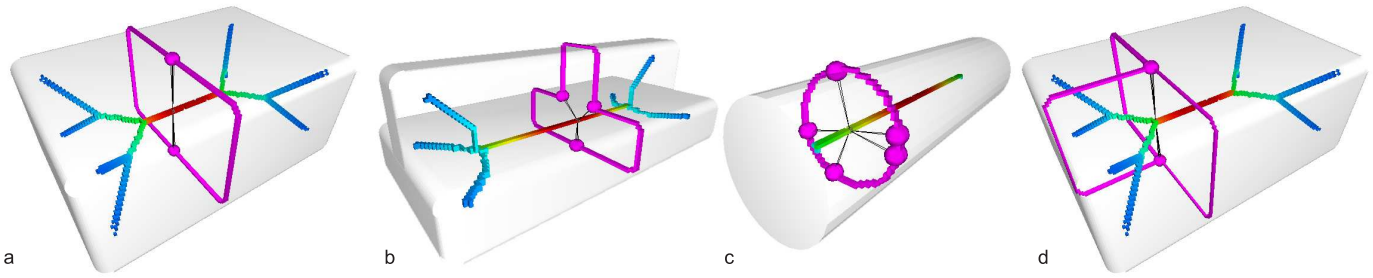


Fig. 6. Generic curve skeleton point (**a**), non-generic cases (**b-d**). The curve skeletons are shown with a rainbow color map encoding the collapse measure. In each image, the shortest path set for a selected curve skeleton voxel is shown in magenta. Feature voxels are shown as spheres and are connected to the selected voxel using line segments. These images are screenshots made using our implementation.

algorithm does not differentiate between surface skeleton and non-skeleton voxels, for which $|C|$ is also 1, the length of the longest shortest-path for non-skeleton voxels will be between neighboring voxels, resulting in a very small ρ. The object voxels will be of low importance and will disappear first when increasing the threshold. Next, ρ is normalized as explained in Section III-C (line 11).

The above method of detecting curve skeleton points and computing their collapse measure works well only for genus 0 objects, i.e. objects without tunnels, whose skeleton is a tree. Detecting curve skeleton points in this manner for objects with genus greater than zero does not work, because the Jordan curve theorem does not hold: for points where the shortest paths P are adjacent to a tunnel, P does not divide the boundary into two or more connected components. A simple variation of the algorithm for detecting curve skeleton points for objects with tunnels is as follows. We replace the curve skeleton detection (lines 5 and 6, Fig. 4) with a "Jordan curve test" on P. See [39] for details. The choice of a meaningful importance measure for the points situated on $\mathcal{C}$-loops is subject to future research. Yet, the variation suggested above allows a simple and robust computation of curve skeletons for objects of arbitrary genus. Fig. 5 shows such a curve skeleton extracted for an object with two tunnels.

In the fourth and final stage, we simplify the skeleton with the desired τ value by applying Eq. 11 (line 13). In order to compute the equalized skeleton $\mathcal{S}'_\tau$, the algorithm can be slightly modified by taking the square root of ρ on $\mathcal{C}$ (Fig. 4, line 7), and normalizing by $\sqrt{\frac{1}{2}\text{area}(\partial\Omega)}$ (line 11). Depending on the value of τ, we obtain for $\mathcal{S}_\tau$ or $\mathcal{S}'_\tau$ surface and curve skeletons simplified to different levels, as demonstrated in Section V.

C. Non-generic cases

Our model from Section III-B assumed the generic case of a skeleton point having exactly two feature points and admitting one or two shortest geodesics between them. An example of such a generic curve skeleton point is shown in Figure 6a. However, our algorithm from the previous section can also deal with the non-generic cases, as explained in this section. We can distinguish two cases: 1) points having three or more feature points, and 2) points having three or more shortest geodesics between their two feature points.

We begin explaining the first case. Surface skeletons consist of manifolds with boundaries, called *sheets* [40], [41]. Sheets intersect in curves. Points on these curves have more than two feature points. Fig. 6b shows such a configuration in a box with a vertical ridge. The selected curve skeleton point lies on the intersection curve of three sheets and has three feature voxels. This is a limit case. A point lying on one sheet has one pair of feature points, but a point lying on the intersection of three sheets has three pairs of feature points. Some feature points are shared among pairs, yielding three distinct feature points. No two feature points among these three admit two shortest geodesics between them, so that the curve skeleton cannot be detected using this criterion. However, our algorithm combines all shortest geodesics, so that we now obtain a slice boundary that splits the object surface into two components, correctly detecting the curve skeleton point. Another example of a point having three or more feature points is that of a curve skeleton point whose inscribed ball has a finite contact with the boundary. This yields a continuum of feature points, which in voxel space results in a finite number of feature voxels. Combining the shortest paths between all feature voxels again resolves the issue as it splits the object surface into two components. An example of this situation in a cylinder shape is shown Fig. 6c.

Combining all shortest geodesics might result in slightly wrong collapses in the uncommon occasion that a curve skeleton point lies on the intersection curve of more than three sheets. In this case, not every two feature points and the associated shortest geodesic between them correspond to a sheet. At those points, the collapse measure might differ slightly from what it should be, violating monotonicity. However, we did not find this to be a detectable issue in any real-world example (see Sec. V).

The second non-generic case is that of a point $p \in \mathcal{C}$ having three or more shortest geodesics between its two feature points, which happens at junction points of the curve skeleton (Fig. 6d). This is a limit case, since each curve skeleton point next to the junction does admit two shortest geodesics. The difference with the generic case is that we obtain more than two connected components in case of a junction point. The algorithm deals with this by always taking $\partial\Omega \setminus c$ as the collapse at p (Fig. 4, line 7), where c is the largest component in C, essentially taking the largest collapse among

the collapses of p's neighbors.

V. Results

We have implemented the algorithm from the previous section in C++ and have run it on a Pentium 4, 3GHZ with 1GB of RAM. As input we used several complex polygonal meshes from [42], [43], voxelized using binvox [44] for various resolutions. We used object resolutions ranging up to 512^3 voxels. For each object the resolution that we used is appended to the object's name. The Mobile object was created such that it incorporates several synthetic, non-natural shapes. Various measurements are shown in Table I. Column "dim" indicates the dimension of the object. Columns "$|\Omega|$" and "$|\partial\Omega|$" indicate the number of boundary and object voxels respectively. Columns "#paths" indicate the number of computed shortest paths, and "#comp." indicates how many times the connected components are computed (Fig. 4, line 5), whereas "paths t" and "comp. t" show the total wall-clock time for both. The last column shows the total wall-clock time for the whole algorithm to complete. As can be seen, all objects are completed within 10 minutes. We use a simple caching scheme for shortest paths to prevent computing the same shortest path more than once as much as possible. The size of the cache is a user parameter and presents a tradeoff between speed and memory usage. In these measurements a cache size of 50 MB was used, which presents a speed-up factor of 3 to 4 approximately. For these resolutions, larger cache sizes do not improve speed significantly. Column "mem" shows the peak memory usage, which is below 800 MB for all the objects we considered.

Fig. 7 shows the simplified skeletons $\mathcal{S}_\tau$ of three objects as computed by our algorithm, with the indicated values of τ. The importance measure is visualized using a rainbow color map, mapping 0 to blue (i.e. unimportant skeleton points) and 1 to red (i.e. central, important skeleton points). We observe that the non-generic cases in the Mobile object, such as the cylinder and extruded star, are handled well. The surface skeleton is mainly blue because its importance measure is significantly smaller than that of the curve skeleton, as explained in Section III-C. In order to highlight the variation of ρ on $\mathcal{S} \setminus \mathcal{C}$ we can better use the equalized skeleton $\mathcal{S}'_\tau$. Fig. 8 shows $\mathcal{S}'_\tau$ for the Bird object. It is interesting to see how $\mathcal{S}_\tau$ and $\mathcal{S}'_\tau$ progress differently when increasing τ. With $\mathcal{S}'_\tau$, the surface and curve skeleton are simplified more simultaneously, especially near the periphery of the skeleton, which can be useful for some applications, as we will see in the next section.

Fig. 9 shows the curve skeleton in isolation for various objects. Note that we modified the original Rockerarm object so that it is genus 0. We observe that the extracted curve skeletons reach into fine structures like the tentacle tips of the Octopus object and the Cow's tail, are centered with respect to the object surface, and exhibit very little wiggle noise. Cornea et al. [8] made a comparison between four curve skeleton methods: a basic thinning, distance field, geometric, and potential-field method. Among the objects tested was the Plane object. We observe that our approach delivers superior results on this object (compare Fig. 9 and [8, Fig.12]). The price we pay is that our method is slower than the first three methods mentioned in [8], although it is the same order as the potential-field method. More results of our skeletonization method are available online at [45], including animations of $\mathcal{S}_\tau$'s progression.

VI. Discussion

Because curve and surface skeletonization methods differ in the precise skeleton definition that they use, the object representation they work on, and the applications they target, they are usually compared by their results based on various desirable properties as mentioned in Section I. In Section VI-A we discuss how our approach satisfies these desirable properties. In Section VI-B, we discuss how several key aspects of our approach relate to other approaches.

A. Desirable properties

The skeleton is *invariant* under isometric transformations (modulo, of course, the voxel discretization). The skeleton is up to four voxels thick due to the discretization; a simple post-processing step could be added to make it one voxel *thin*, if desired. The surface skeleton is *centered* because it is defined as those points having at least two feature points. The curve skeleton is centered on the surface skeleton with respect to the shortest-geodesic length function, and thus it is also centered within the object. Since the importance measure is monotonically increasing (Sec. III-B), the simplified skeletons $\mathcal{S}_\tau$ are *connected* by default. No special homotopy-preserving provisions are needed to ensure this, unlike e.g. [13], [23], [28], due to the global nature of the importance measure. The simplification is continuous for the vast majority of shapes. That is, small changes in τ result in small changes in $\mathcal{S}_\tau$. This is a very important property, given the infamous unstable behavior of skeletal structure. The continuity can be ascribed to the fact that the shortest geodesic of a point $q \in \mathcal{S} \setminus \mathcal{C}$ evolves smoothly over $\mathcal{S} \setminus \mathcal{C}$, so that ρ is continuous on $\mathcal{S} \setminus \mathcal{C}$. The slice boundaries, each consisting of two shortest geodesics, evolve smoothly over $\mathcal{C}$ except at curve skeleton junctions, so that ρ is continuous on $\mathcal{C}$ and only contains jumps at $\mathcal{C}$-junctions. The skeletons are *hierarchical* or multiscale, because ρ represents a continuous hierarchy of nested skeletons in which each $\mathcal{S}_\tau$ represents a separate level.

Our approach satisfies the two additional desirable properties we proposed in Section I for methods that compute both curve and surface skeletons. First, the curve skeleton is *included* in the surface skeleton, because the curve skeleton is considered as a special case of the surface skeleton, namely as those points having more than one shortest geodesic. Second, our importance measure treats the non-skeleton, surface and curve skeleton points in a *uniform* manner: All points are characterized by their importance given by the advected mass model.

Our simplification method has a single parameter τ; no other hacks or settings are needed. The meaning of the importance ρ which is thresholded by τ is quite simple and intuitive: for a given skeleton point p, $\rho(p)$ represents the fraction of the object's boundary which is described by that point.

TABLE I

TABLE WITH MEASUREMENTS. SEE THE TEXT FOR DETAILS.

object	dim	$\|\Omega\|$	$\|\partial\Omega\|$	#paths	#comp.	paths t	comp. t	mem	total t
Bird512	262x132x488	717k	95k	750k	73k	126s	59s	303M	196s
Cow256	78x146x244	549k	44k	357k	47k	9s	14s	125M	28s
Cow384	115x217x366	1,904k	102k	811k	151k	32s	102s	319M	148s
Cow512	154x289x488	4,575k	183k	1,441k	345k	75s	302s	592M	411s
Dino512	159x453x487	3,218k	177k	1,413k	231k	61s	156s	590M	246s
Dragon512	214x336x488	5,656k	320k	2,664k	491k	152s	254s	747M	475s
Hand512	488x205x342	3,137k	167k	1,348k	211k	132s	171s	570M	332s
Homer512	162x488x276	3,947k	191k	1,517k	324k	62s	207s	561M	299s
Horse512	488x421x228	4,900k	214k	1,698k	349k	117s	289s	716M	449s
Plane	217x304x98	545k	110k	912k	71k	257s	57s	306M	323s
Mobile384	366x224x348	1,211k	175k	1,418k	123k	71s	83s	482M	171s
Octopus384	366x259x335	610k	76k	680k	64k	16s	24s	321M	54s
Rockerarm256	246x126x76	528k	67k	541k	58k	68s	31s	176M	105s
Rockerarm384	366x188x112	1,860k	154k	1,197k	156k	301s	142s	399M	461s

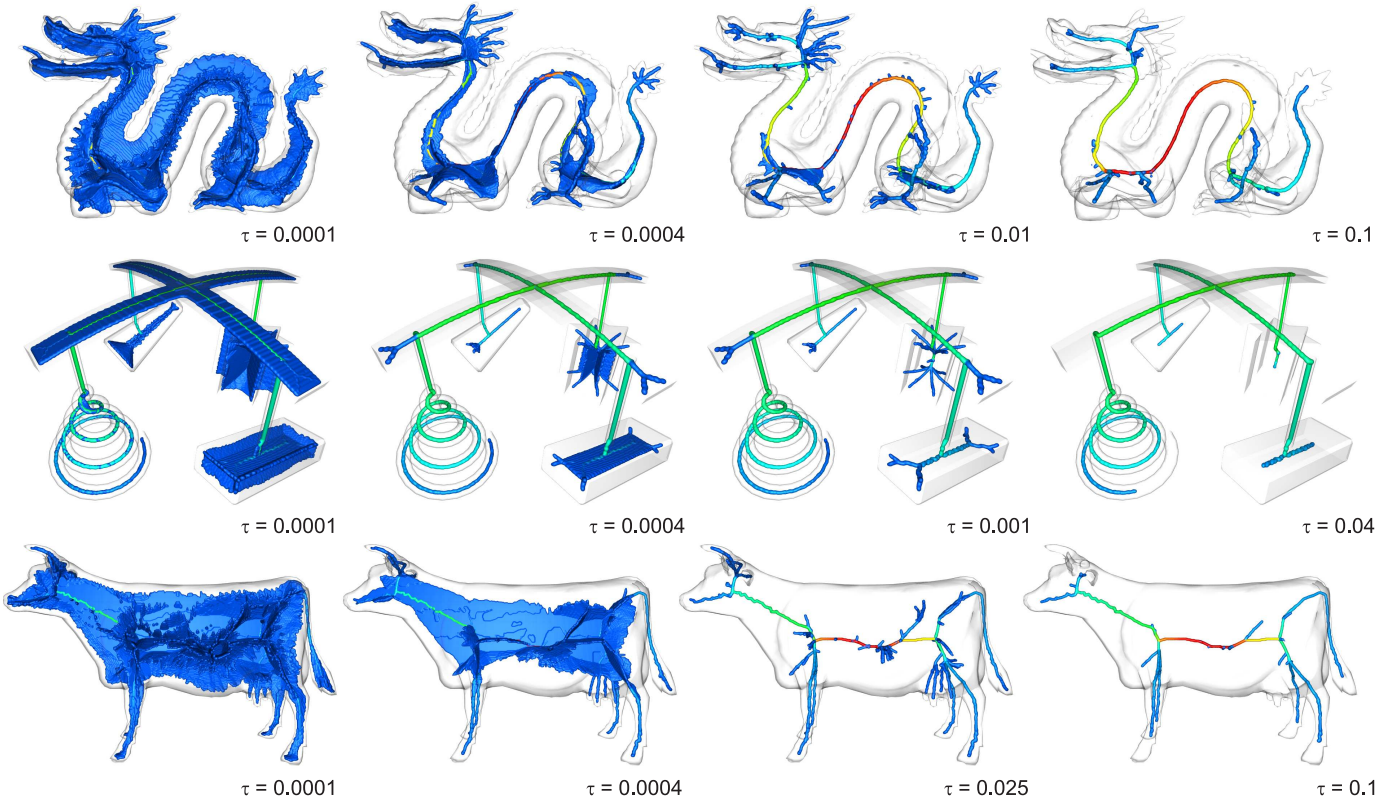


Fig. 7. Simplified skeletons $\mathcal{S}_\tau$ of the Dragon, Mobile, and Cow objects at four thresholds τ. The importance measure is visualized using a rainbow color map.

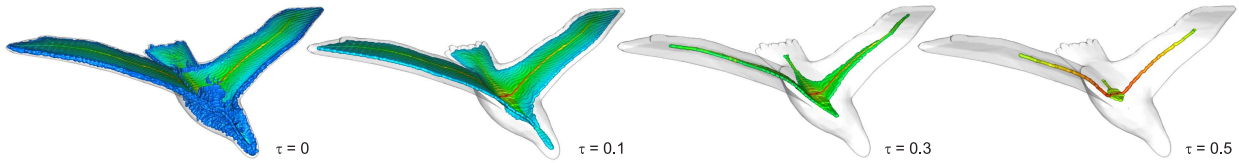


Fig. 8. Simplified equalized skeletons $\mathcal{S}'_\tau$ for the Bird object.

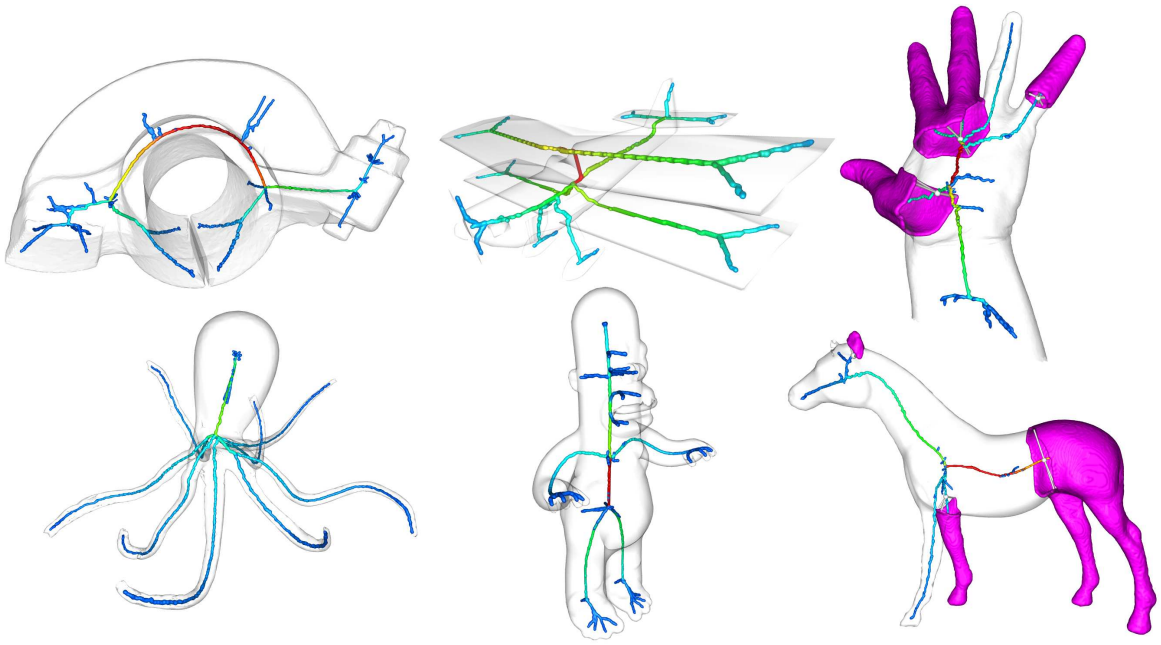


Fig. 9. The curve skeletons of the Rockerarm, Plane, Hand, Octopus, Homer, and Horse objects. The importance measure is visualized using a rainbow color map. For the Hand and Horse objects we show the collapses (magenta) for several curve skeleton voxels (white). The line segments indicate feature voxels of the white voxels.

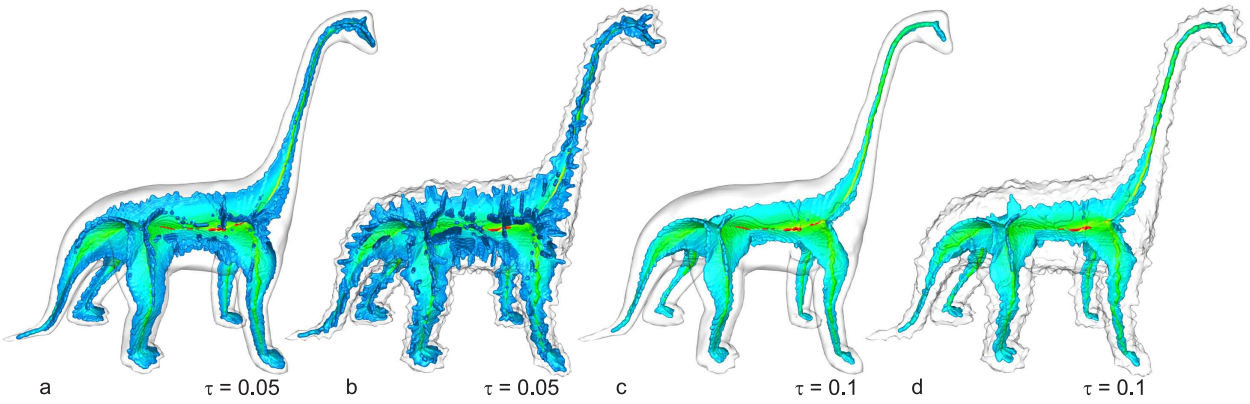


Fig. 10. The skeletons can be made robust by increasing the threshold τ.

The *robustness* property requires the skeleton of a noisy surface to be close to that of the corresponding smooth surface. We can achieve this behavior by setting τ such that skeleton parts due to noise are filtered out. Fig. 10 shows the Dino object with and without surface noise and its skeleton for two thresholds τ. As expected, we observe that $\mathcal{S}'_\tau$ of the noisy Dino (**b**) is much noisier than $\mathcal{S}'_\tau$ of the Dino without noise (**a**) for small values of τ: $\tau < 0.05$. The skeleton of the noisy Dino can be made robust (**d**) by increasing τ. At $\tau = 0.1$, the structure of the simplified skeleton of the noisy Dino (**d**) is comparable to the skeleton of the non-noisy Dino (**c**). The importance measures on both are also very similar as indicated by the colors. In the above, we used the simplified equalized skeleton $\mathcal{S}'_\tau$. If we use the non-equalized skeleton $\mathcal{S}_\tau$, the spurious branches of the curve skeleton remain, until τ is so high that $\mathcal{S}$ is completely removed.

The curve skeleton is said to be *junction detective*, or allowing for component-wise differentiation, when different logical parts of the object can be inferred from the curve skeleton [4]. In fact, the collapses that we compute provides us

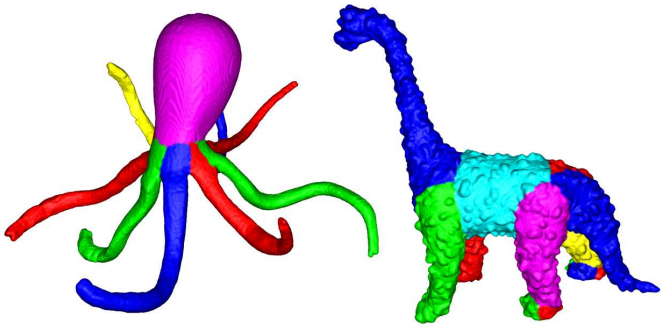


Fig. 11. Shape segmentations using the skeleton-to-boundary mapping.

with a natural skeleton-to-boundary mapping. Surface skeleton points map to curves on the object surface, whereas curve skeleton points map to contiguous areas. Figure 9 shows the skeleton-to-boundary mapping for several curve skeleton points for the Hand and Horse objects. We observe that the collapses correspond to logical parts of the boundary, such as the legs or ear of the horse and fingers of the hand. The skeleton-to-boundary mapping can be used in various applications, e.g., for selection purposes in geometric modeling applications. Another application is shape segmentation. The collapses associated with the curve skeleton branches coming together in a junction provide us with meaningful components: distinct logical object parts. Combining the meaningful components that are associated with all curve skeleton junctions yields a shape segmentation having desirable properties, as follows. Because the skeleton-to-boundary mapping is based on shortest geodesics, the borders between segments are smooth, minimally twisting, and robust to noise. Being based on the curve skeleton, the segmentation respects circular shape symmetry and is pose invariant. A shape segmentation method from the first and third author exploiting this skeleton-to-boundary mapping is described in [39]. Figure 11 shows two segmentations obtained using this approach.

The original object, or a simplified version thereof, can be *reconstructed* from the surface skeleton by placing at each voxel its maximally inscribed ball. In general, reconstruction from the curve skeleton is not possible by using only balls. One simple application of our simplified skeletons is that of surface smoothing. By reconstructing a surface from the simplified skeleton $\mathcal{S}_\tau$ for a small τ value, small-scale surface noise is replaced by sphere segments. For the purpose of surface smoothing, the simplified equalized skeleton $\mathcal{S}'_\tau$ is used, so that the spurious curve and surface skeleton parts due to the noise near the boundary are simplified simultaneously. Fig. 12 shows two examples. Of course, τ cannot be too high, otherwise the reconstruction becomes too inaccurate, or "spherified". Reducing noise in this manner works best at thick object parts, such that the inscribed balls of this part are large in comparison to the inscribed balls in the perturbations. Indeed, noise at thin parts can be considered as object features, more so than noise at thick parts. Noise at the thin parts, namely the ridges of the box and the neck of the Dino, is less reduced than the thick parts. A nice feature of using the collapse measure is that the reconstructed object cannot become disconnected due to the simplification, because the collapse measure on the curve skeleton has such a high value. In contrast, if we would simplify the Dino in Fig. 12 using only the collapse measure as defined on $\mathcal{S} \setminus \mathcal{C}$, the neck would become disconnected.

The monotonicity of the collapse measure comes at a price: Our measure is not so efficient to compute as purely local importance measures, due to the global operations involved. Computing the feature transform using [36] takes $O(n)$, where $n = |\Omega|$. Computing the set of shortest paths for an object voxel in the boundary graph using A* takes $O(b \log b)$ in the worst case, where $b = |\partial\Omega| \approx \log n$. The worst case is a sphere, as the shortest-path algorithm visits practically all boundary voxels for diametrically-opposed feature voxels. However, shortest paths are not computed between arbitrary boundary voxels, but always between associated feature-voxels, so that for objects that consist of distinct parts, the algorithm visits only a small subset of the boundary voxels. Computing the connected components using a hierarchical spatial subdivision scheme on the surface takes $O(\log b)$. In total, the worst case of the algorithm is $O(n(b \log b))$, but practical cases are far below this limit. Table I shows the relation between n, b, and the running times for several practical objects. In particular, we note that the speed of our implementation compares favorably to [28]. On a comparable machine, their approach took half an hour to compute a curve skeleton for the Rockerarm. Our approach took under 10 minutes, using a voxel resolution of 384^3 for a full curve and surface skeleton hierarchy, while observing that the complexity of the curve skeleton is comparable (compare Fig. 9 and [28, Fig.5]). Our approach is faster than the potential-field method of Cornea et al. [16], which reportedly takes up to half an hour for voxel resolutions in the order of 200^3 on a standard PC.

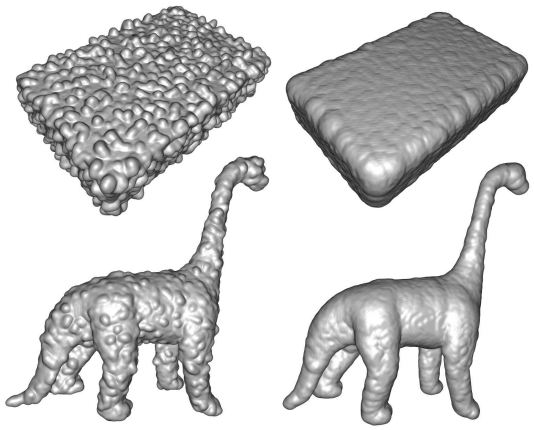


Fig. 12. Surface smoothing of the Box and Dino objects by reconstruction from $\mathcal{S}'_\tau$, with a small τ.

When discussing the speed, we should stress the fact that our approach is, to our knowledge, the only one that generalizes a global importance measure for 2D objects, namely the feature-distance measure, to 3D. This measure is the cornerstone of our method, as it guarantees the satisfaction of all the desirable properties considered, similar to its analogous 2D counterpart.

B. Comparison with other methods

In this section we compare our method with existing methods on five main aspects of our implementation: use of geodesics, local versus global detection, gradient field extension, the use of advection, and discrete versus continuous space methods.

Our approach resembles the recently published approach of Dey and Sun [28] in the sense that the shortest-geodesic length between feature points is used. There are some important differences however. Most importantly, we produce a skeleton hierarchy, while their approach does not. Second, their approach is more of a hybrid method, in which first

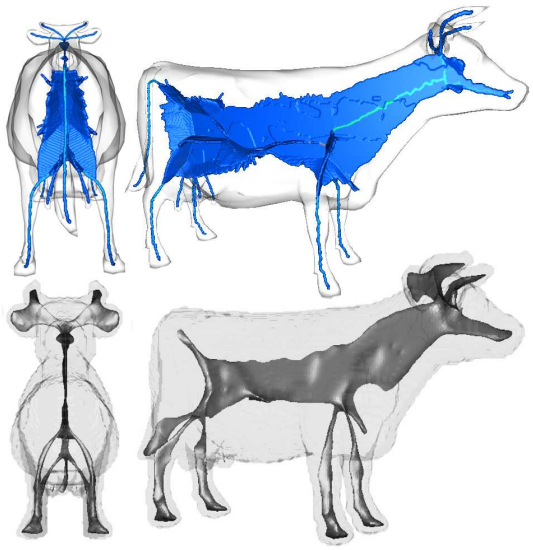


Fig. 13. A comparison between the surface skeleton produced by using our global measure (top) and a typical local measure (bottom).

the surface skeleton and global geodesic-length measure are explicitly determined, after which a local divergence-based measure [13] is used to find the curve skeleton on the derived surface skeleton structure. Because this latter measure is local, an erosion is needed to enforce connectedness, as well as a sampling resolution high enough to accurately compute the involved divergence. In contrast, our approach uses a *single* global measure for both the curve and surface skeleton. The measure can be computed independently on a point-by-point fashion, making our algorithm simple and allowing parallel processing. Furthermore, all our computations are based on a non-derived structure, namely the object surface. This allows us to use a voxel representation, requiring simple data-structures and algorithms, keeping our algorithm straightforward and efficient. A second advantage of using only non-derived structures and integral quantities, such as geodesics and collapsed areas, is that our approach is very robust, even for coarsely sampled volumetric objects.

We would further like to compare the surface skeleton produced by our global importance measure with the skeleton resulting from a purely local measure. See Fig. 13. The top images show the surface skeleton $\mathcal{S}_\tau$ of the Cow object computed by our method. The bottom images (taken from [23]) show the surface skeleton resulting from the moment-based measure from [3]. The threshold used for the latter images was carefully chosen such that the skeleton is both connected and noise free. We chose τ for our method such that the skeletons are most similar. Yet, we observe that our skeleton at this comparable simplification level is able to capture more details, such as the horizontal skeleton sheets and the skeleton in the udder and tail. This can be explained by the fact that local measures cannot distinguish between these fine structures and noise, and will eliminate them together, whereas global measures can. Our skeleton is connected regardless of τ.

At a conceptual level, our advection-based importance measure is related to the flow complex of Giesen et al. [46], [47]. We briefly outline next the similarities and differences. Both approaches define a vector field inside a 3D object, in which advection of points situated on the object boundary is considered. In both approaches, the vector field is equal to the gradient of the distance-to-boundary function in the non-critical points, i.e. away from the skeleton. Both approaches extend the vector field to the critical (skeleton) points, however in different ways, and for different purposes. In both cases, the vector field defines implicitly a dynamical system, or induced flow, which describes the motion, or advection, of particles. Typically, such particles originate from a (dense) sampling of the surface of the considered 3D object. Giesen et al. extend the gradient field on the skeleton in the direction of steepest ascent, and end the trajectories, called orbits in [46], in critical points where the steepest ascent cannot be determined, called fixed points of the flow complex. In contrast, our advection model extends the vector field on the skeleton such that it is tangent to the trajectory tree, constructed as described in Sec. III-B, and oriented toward the unique root. Hence, the advection model of [46] admits several fixed points or "sinks", even for genus 0 objects, whereas our model admits one unique sink, or root. Our notion of collapse of a point is similar to the set of start points, on the object surface, of all orbits contained in what in [47] is called a "stable manifold" of that point. They define, and compute, the core of a 3D object as the set of unstable manifolds of critical points c, which are the set of points to which the neighbors of c flow. The core is shown to be homotopic to the medial axis, or skeleton. At an implementation level, the object and skeleton are manipulated in a computational geometry setting based on polygonal descriptions, using tools such as the Voronoi and Delaunay diagrams. In contrast, we define and compute the skeleton using the (extended) feature transform (Sec.IV), and we perform all computations in a discrete voxel setting. All in all, the goal of our advection-based model is to define a monotonic importance measure which allows robust simplification of complex 3D skeletons, whereas the model in [46] is used to fulfill topological guarantees for medial axes [47] and surface reconstruction [48] computation methods.

Skeletons can be computed by direct numerical simulation of advection. For example, Torsello and Hancock [49] compute skeletons of 2D shapes by simulating an advection process based on momentum conservation. This yields a scalar density field which captures the local contractions and dilations of the boundary evolution in the distance field gradient. Hence, their method is a skeleton detector, but not a skeleton simplifier. In contrast, our collapse measure has overall low values away from the skeleton, but has a high variation on the skeleton itself, which allows us to simplify the skeleton.

Finally, let us mention that we have opted for a voxel-based approach for its ease of implementation. A disadvantage of this is that polygonal models first need to be voxelized. The resulting skeletons are influenced by the grid's orientation, causing a slight loss of rotation invariance, and the grid's resolution, which may cause a loss of detail for objects containing small features. Methods acting directly on continuous geometrical data do not have these issues. However, we would like to

indicate that the definition of our global importance measure is not limited to discrete space. As long as the key ingredients of this measure are available, namely, computation of the feature points, shortest geodesics, and connected components, our approach can be adapted to other representations.

VII. Conclusion

We have proposed the collapse measure, a novel importance measure that enables the robust computation of multiscale curve and surface skeletons of 3D objects. To our knowledge, this is the first truly global 3D importance measure that can be used to obtain both surface and/or curve skeleton hierarchies in a uniform manner. Underpinning the collapse measure is a physical process in which mass from the object boundary is advected or "collapses" onto the skeleton. This gives the measure an intuitive meaning, and allows us to reason about it and deduce several properties of interest. For the practical implementation of the proposed measure, we do not explicitly simulate the advection process, as this can be computationally unstable and complex. Instead, we propose a practical algorithm that is straightforward to implement, does not need any post-processing steps, and is robust, as it uses only integral operations. Our algorithm delivers good results on a wide range of real-world, complex objects. A limitation of our measure is that objects with tunnels cannot be handled. However, this is a known limitation for the analogous 2D feature-distance measure [5]. In detail, we can compute curve skeletons for such objects, but not simplify them, since we have no importance measure for their loop-parts.

In future work, we want to apply our skeletons in various applications, such as the simplification of complex 3D objects. Also, we consider an extension of the collapse measure that can handle objects with tunnels. The recursive nature of our approach makes it interesting to look at generalizations and applications beyond 3D, e.g. for time-dependent surfaces. Finally, on a more theoretical ground, we plan to investigate our conjecture (see Sec. III-B) in more depth to acquire more insight into the relation of shapes with their skeletal counterparts.

Acknowledgement

This work was supported by the Netherlands Organization for Scientific Research (NWO) under grant number 612.065.414. We would like to thank the anonymous reviewers for their helpful comments.

References

[1] H. Blum, "A transformation for extracting new descriptors of shape," in *Models for the Perception of Speech and Visual Form*, W. Wathen-Dunn, Ed. MIT Press, 1967, pp. 362–380.

[2] N. Gagvani, D. Kenchammana-Hosekote, and D. Silver, "Volume animation using the skeleton tree," in *Proc. IEEE Symposium on Volume Visualization*, 1998, pp. 47–53.

[3] M. Rumpf and A. Telea, "A continuous skeletonization method based on level sets," in *IEEE TCVG Symposium on Visualization*, D. Ebert, P. Brunet, and I. Navazo, Eds., 2002, pp. 151–158.

[4] N. Cornea, D. Silver, and P. Min, "Curve-skeleton applications," in *Proc. IEEE Visualization*, 2005, pp. 95–102.

[5] R. Ogniewicz and O. Kübler, "Hierarchic voronoi skeletons," *Pattern Recognition*, vol. 28, no. 3, pp. 343–359, 1995.

[6] L. Costa and R. Cesar, Jr, *Shape analysis and classification*. CRC Press, 2001.

[7] A. Telea and J. J. van Wijk, "An augmented fast marching method for computing skeletons and centerlines," in *Proc. of the Symposium on Data Visualisation (VisSym'02)*, 2002, pp. 251–259.

[8] N. Cornea, D. Silver, and P. Min, "Curve-skeleton properties, applications and algorithms," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 3, pp. 530–548, 2007.

[9] K. Palagyi and A. Kuba, "Directional 3d thinning using 8 subiterations," in *Proc. Discrete Geometry for Computer Imagery (DGCI), Lecture Notes in Computer Science*, vol. 1568, 1999, pp. 325–336.

[10] X. Li, T. W. Woon, T. S. Tan, and Z. Huang, "Decomposing polygon meshes for interactive applications," in *Proc. of symposium on Interactive 3D graphics*, 2001, pp. 35–42.

[11] M. Mortara, G. Patanèt, M. Spagnuolo, B. Falcidieno, and J. Rossignac, "Plumber: a method for a multi-scale decomposition of 3d shapes into tubular primitives and bodies," in *Proc. of the ACM Symposium on Solid Modeling and Applications*, 2004, pp. 339–344.

[12] M. Wan, F. Dachille, and A. Kaufman, "Distance-field based skeletons for virtual navigation," in *Proc. IEEE Visualization*, 2001, pp. 239–246.

[13] K. Siddiqi, S. Bouix, A. Tannenbaum, and S. Zucker, "Hamilton-jacobi skeletons," *Int. Journal of Computer Vision*, vol. 48, no. 3, pp. 215–231, 2002.

[14] R. Strzodka and A. Telea, "Generalized distance transforms and skeletons in graphics hardware," in *Proc. of EG/IEEE TCVG Symposium on Visualization (VisSym '04)*, 2004, pp. 221–230.

[15] N. Ahuja and J. Chuang, "Shape representation using a generalized potential field model," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19, no. 2, pp. 169–176, 1997.

[16] N. Cornea, D. Silver, X. Yuan, and R. Balasubramanian, "Computing hierarchical curve-skeletons of 3d objects," *The Visual Computer*, vol. 21, no. 11, pp. 945–955, 2005.

[17] C. Pudney, "Distance-ordered homotopic thinning: A skeletonization algoritm for 3D digital images," *Computer Vision and Image Understanding*, vol. 72, no. 3, pp. 404–413, 1998.

[18] S. Katz and A. Tal, "Hierarchical mesh decomposition using fuzzy clustering and cuts," *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 954–961, 2003.

[19] A. Sud, M. Foskey, and D. Manocha, "Homotopy-preserving medial axis simplification," in *Proc. of the ACM Symp. on Solid and Physical modeling (SPM'05)*, 2005, pp. 39–50.

[20] M. van Dortmont, H. van de Wetering, and A. Telea, "Skeletonization and distance transforms of 3d volumes using graphics hardware," in *Proc. Discrete Geometry for Computer Imagery (DGCI)*, 2006, pp. 617–629.

[21] D. Shaked and A. Bruckstein, "Pruning medial axes," *Computer Vision and Image Understanding*, vol. 69, no. 2, pp. 156–169, 1998.

[22] G. Malandain and S. Fernández-Vidal, "Euclidean skeletons," *Image and Vision Computing*, vol. 16, no. 5, pp. 317–327, 1998.

[23] A. Telea and A. Vilanova, "A robust level-set algorithm for centerline extraction," in *Proc. of the Symposium on Data Visualisation (VisSym'03)*, 2003, pp. 185–194.

[24] M. Foskey, M. Lin, and D. Manocha, "Efficient computation of a simplified medial axis," in *Proc. of ACM Symposium on Solid Modeling and Applications*. ACM Press, 2003, pp. 96–107.

[25] K. Siddiqi and B. Kimia, "A shock grammar for recognition," in *Proc. of the Conference on Computer Vision and Pattern Recognition (CVPR'96)*, 1996, pp. 507–513.

[26] S. Bouix, K. Siddiqi, and A. Tannenbaum, "Flux driven automatic centerline extraction," *Medical Image Analysis*, vol. 9, no. 3, pp. 209–221, 2005.

[27] S. M. Pizer, K. Siddiqi, G. Székely, J. N. Damon, and S. W. Zucker, "Multiscale medial loci and their properties," *Int. Journal of Computer Vision*, vol. 55, no. 2-3, pp. 155–179, 2003.

[28] T. Dey and J. Sun, "Defining and computing curve-skeletons with medial geodesic function," in *Proc. of Eurographics Symposium on Geometry Processing*, 2006, pp. 143–152.

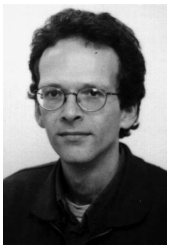
[29] S. Prohaska and H.-C. Hege, "Fast visualization of plane-like structures in voxel data," in *Proc. IEEE Visualization*, 2002, pp. 29–36.

[30] Y. Shinagawa, T. Kunii, and Y.-L. Kergosien, "Surface coding based on Morse theory," *IEEE Computer Graphics and Applications*, vol. 11, no. 5, pp. 66–78, 1991.

[31] M. Hilaga, Y. Shinagawa, T. Kohmura, and T. L. Kunii, "Topology matching for fully automatic similarity estimation of 3D shapes," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM Press, 2001, pp. 203–212.

[32] J. Sethian, *Level set methods and fast marching methods*, 2nd ed. Cambridge University Press, 1999.
[33] C. Sigg, R. Peikert, and M. Gross, "Signed distance function using graphics hardware," in *Proc. IEEE Visualization*, 2003, pp. 83–90.
[34] D. Reniers and A. Telea, "Quantitative comparison of tolerance-based feature transforms," in *First Int. Conference on Computer Vision Theory and Applications (VISAPP'06)*, 2006, pp. 107–114.
[35] A. Lieutier, "Any open bounded subset of $\mathbb{R}^3$ has the same homotopy type as its medial axis," *Computer-Aided Design*, vol. 36, no. 11, pp. 1029–1046, 2004.
[36] J. Mullikin, "The vector distance transform in two and three dimensions," *CVGIP: Graphical Models and Image Processing*, vol. 54, no. 6, pp. 526–535, 1992.
[37] N. Kiryati and G. Székely, "Estimating shortest paths and minimal distances on digitized three-dimensional surfaces," *Pattern Recognition*, vol. 26, pp. 1623–1637, 1993.
[38] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.
[39] D. Reniers and A. Telea, "Skeleton-based hierarchical shape segmentation," in *Proc. of the IEEE Int. Conf. on Shape Modeling and Applications (SMI'07)*, 2007, pp. 179–188.
[40] P. Giblin and B. B. Kimia, "A formal classification of 3d medial axis points and their local geometry," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 26, no. 2, pp. 238–251, 2004.
[41] J. Damon, "Determining the geometry of boundaries of objects from medial data," *International Journal of Computer Vision*, vol. 63, no. 1, pp. 45–64, 2005.
[42] AIM@SHAPE repository http://shapes.aim-at-shape.net.
[43] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser, "The Princeton shape benchmark," in *Shape Modeling International*, 2004.
[44] P. Min, "Binvox, a 3d mesh voxelizer," http://www.google.com/search?q=binvox.
[45] D. Reniers, personal website, http://www.win.tue.nl/~dreniers/skeletons.
[46] J. Giesen and M. John, "The flow complex: a data structure for geometric modeling," in *Proc. of the 14th ACM-SIAM symposium on Discrete algorithms*, 2003, pp. 285–294.
[47] J. Giesen, E. A. Ramos, and B. Sadri, "Medial axis approximation and unstable flow complex," in *Proc. of the 22th annual symposium on Computational geometry*, 2006, pp. 327–336.
[48] J. Giesen and M. John, "Surface reconstruction based on a dynamical system," *Computer Graphics Forum*, vol. 21, no. 3, p. 363, 2002.
[49] A. Torsello and E. R. Hancock, "Correcting curvature-density effects in the hamilton-jacobi skeleton," *IEEE Trans. on Image Processing*, vol. 15, no. 4, pp. 877–891, 2006.

Dennie Reniers received his MSc in Computer Science from the Eindhoven University of Technology (TU/e), the Netherlands, in 2004. Currently, he is a PhD student in the Department of Mathematics and Computer Science at TU/e. His research interests include shape representation and segmentation, and discrete geometry.



Jarke J. van Wijk received the MSc degree in industrial design engineering in 1982 and the PhD degree in computer science in 1986, both with honors. He worked at a software company and at the Netherlands Energy Research Foundation ECN before he joined the Technische Universiteit Eindhoven in 1998, where he became a full professor of visualization in 2001. He is a member of the IEEE, ACM SIGGRAPH, and Eurographics. He was a paper cochair for IEEE Visualization in 2003 and 2004, and a paper cochair for IEEE InfoVis 2006. His main research interests are information visualization and flow visualization, both with a focus on the development of new visual representations.

Alexandru Telea graduated in 1996 in Computer Science at the Polytechnics Institute of Bucharest, Romania, and received his Ph.D. in computer science in 2000 from the TU/e in the Netherlands. Since then he has been an assistant professor in visualization and computer graphics in the Department of Mathematics and Computer Science at TU/e. His research interests are in multiscale modeling and analysis for scientific and information visualization, shape analysis and representation, and component and object-oriented software architectures.