Pattern Recognition Letters xxx (2014) xxx–xxx

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Comparison of curve and surface skeletonization methods for voxel shapes ☆

André Sobiecki [a,*], Andrei Jalba [b], Alexandru Telea [a]

[a] Institute Johann Bernoulli, Univ. of Groningen, Nijenborgh 9, 9747 AG Groningen, The Netherlands
[b] Department of Mathematics and Computer Science, TU Eindhoven, Den Dolech 2, 5600 MB, Eindhoven, The Netherlands

ARTICLE INFO

Article history:
Available online xxxx

Keywords:
Surface and curve skeletons
Voxel shapes
Medial axes

ABSTRACT

Surface and curve skeletons are important shape descriptors with applications in shape matching, simplification, retrieval, and animation. In recent years, many surface and curve skeletonization methods for 3D shapes have been proposed. However, practical comparisons of such methods against each other and against given quality criteria are quite limited in the literature. In this paper, we compare 4 surface and 6 recent curve skeletonization methods that operate on voxel shapes. We first compare the selected methods from a global perspective, using the quality criteria established by a reference paper in the field. Next, we propose a detailed comparison that refines the gained insights by highlighting small-scale differences between skeletons obtained by different methods.

1. Introduction

Skeletons are shape descriptors with many applications in shape processing, registration, retrieval, matching, animation, and compression [1]. 3D shapes admit two types of skeletons: *Surface* skeletons are 2D manifolds formed by the loci of maximally-inscribed balls within a shape [1,2]. *Curve* skeletons are 1D curves which are locally centered in the shape and capture the shape's part-whole structure [3].

Since the early skeleton definition by Blum [4], many methods have been proposed to compute the two skeleton types. Such methods differ in theoretical aspects, *e.g.* the exact definition for curve skeletons, and practical aspects, *e.g.* space discretization (voxels *vs* meshes); the various approximations being used; and the actual skeleton extraction algorithm. These aspects, and the inherent sensitivity of skeletons to boundary noise, makes different methods produce widely different skeletons for the same input. This causes challenges for the users of skeletons in both research and practical contexts.

Recognizing these challenges, Cornea et al. [3] have presented a taxonomy of curve skeletonization methods and the way these satisfy a set of desirable skeletal properties, and illustrated these for four such methods. Since this publication, several new skeletonization methods have been proposed. A recent study extended the work in [3] by comparing methods of a particular class (contraction-based methods for meshed shapes) against Cornea's criteria [5]. These two studies however cover only a very limited fraction of the current skeletonization methods. Also, papers introducing new skeletonization methods typically present only few additional comparisons. Table 1 illustrates this for a selection of methods, which is by no means exhaustive. Finally, very few comparisons of surface skeletons with curve skeletons are presented, so there are still many open questions on the relationships of the two skeleton types. As computational advances allow implementing increasingly complex skeletonization methods, the challenge of understanding the relative pro's and con's of such new methods only grows.

In this paper, we address the above challenges by presenting a comparison of 4 surface and 6 curve skeletonization methods. In contrast to [5], we focus here on voxel-based methods. In addition to [3], we cover here methods having emerged after their study was published. We use in our comparison the same desirable criteria as in [3]. In addition, we also propose a detailed comparison that aims to provide a fine-grained detail view on the subtle differences between skeletons computed by different methods, including comparisons of curve with surface skeletons. Our results offer additional insight in limitations and challenges of current methods which, to our knowledge, have not been highlighted so far. These results represent further support for the quest of designing better skeletonization methods.

☆ This paper has been recommended for acceptance by Cris L. Luengo Hendriks.

* Corresponding author. Tel.: +31 503633711.
E-mail addresses: a.sobiecki@rug.nl (A. Sobiecki), a.c.jalba@tue.nl (A. Jalba), a.c.telea@rug.nl (A. Telea).

http://dx.doi.org/10.1016/j.patrec.2014.01.012
0167-8655/

Table 1
Recent 3D skeletonization papers. For each method, we show its type (**V**olume or **M**esh), and the surface- and/or curve-skeletonization methods it is compared with. Dashes show that a method does not compute the respective (curve or surface) skeleton type. Last three rows are survey papers.

Method		**Compared with**	
Type	Name	Surface skeleton	Curve skeleton
M	Jalba et al. [8]	[19]	[7,14]
M	Giesen et al. [19]	[12,47,48]	–
M	Huang et al. [18]	–	[15]
M	Au et al. [14]	–	[49,37,30,34]
M	Dey and Sun [34]	–	[30]
M	Tagliasacchi et al. [50]	–	[34,14]
V	Arcelli et al. [45]	–	0
V	Reniers et al. [7]	0	0
V	Hesselink et al. [26]	0	–
V	Siddiqi et al. [25]	0	–
V	Liu et al. [51]	–	[7,37]
V	Ju et al. [52]	[53]	[53]
M + V	Livesu et al. [35]	–	[30,34,14,51]
M	Sobiecki et al. [5]	–	[14–16,54]
M + V	Cornea et al. [3]	–	[37,49,30,13]
V	**Our contribution**	[25,26,7,52]	[25,7,52,51,45,37]

This paper is organized as follows. Section 2 reviews related work. Section 3 presents the compared methods and comparison criteria. Section 4 presents the comparison methodology. Section 5 presents our comparison results. Section 6 discusses these results. Section 7 concludes the paper.

2. Related work

2.1. Skeletonization methods

For a shape $\Omega \subset \mathbb{R}^3$ with boundary $\partial\Omega$, we first define its distance transform $DT_{\partial\Omega} : \mathbb{R}^3 \rightarrow \mathbb{R}^+$

$$DT_{\partial\Omega}(\boldsymbol{x} \in \Omega) = \min_{\boldsymbol{y}\in\partial\Omega} \|\boldsymbol{x} - \boldsymbol{y}\|. \quad (1)$$

The surface skeleton, also called the medial surface, S_Ω of Ω is next defined as

$$S_\Omega = \{\boldsymbol{x} \in \Omega \,|\, \exists \boldsymbol{f}_1, \boldsymbol{f}_2 \in \partial\Omega, \quad \boldsymbol{f}_1 \neq \boldsymbol{f}_2, \ \|\boldsymbol{x} - \boldsymbol{f}_1\| = \|\boldsymbol{x} - \boldsymbol{f}_2\| = DT_{\partial\Omega}(\boldsymbol{x})\} \quad (2)$$

where $\boldsymbol{f}_1$ and $\boldsymbol{f}_2$ are the contact points with $\partial\Omega$ of the maximally-inscribed ball in Ω centered at $\boldsymbol{x}$ [6,7], also called *feature transform* (FT) points [8]. When $\Omega \subset \mathbb{R}^2$, Eq. (2) yields the 2D skeleton, also called the medial axis, of the shape Ω. Surface skeletons contain several manifolds with boundaries which meet along a set of Y-intersection curves [9–11]. Curve skeletons are loosely defined as 1D structures locally centered within a shape $\Omega \subset \mathbb{R}^3$.

Surface and curve skeletons can be computed by geometric, distance field, general field, and thinning methods. **Geometric** methods include Voronoi diagrams [12] and subsets thereof [13], mesh contraction in normal direction [14–17], mean-shift-like clustering [18], and union-of-balls approaches [19,20,8]. Such methods use meshed shape representations and thus scale well to handle high-resolution models [20,8]. **Distance-field** methods find S_Ω along singularities of $DT_{\partial\Omega}$ [21–26] and can be efficiently done on GPUs [27,28]. **General-field** methods use fields smoother (with fewer singularities) than distance transforms [29–32]. Such methods are more robust for noisy shapes. Various regularization metrics, *e.g.* the angle between feature vectors [33,27], or the geodesic distance between feature points [34,7], are used to eliminate spurious skeleton details caused by noise on $\partial\Omega$. Field methods can also compute 3D curve skeletons by backprojecting 2D skeletons of 2D projections [35] or axis-aligned slices [36] of the shape back into 3D. Field methods are implemented for both voxel and mesh shapes. **Thinning** methods remove $\partial\Omega$ voxels while preserving connectivity [37,38]. Tools from mathematical morphology [39] were among the first used to compute curve skeletons by thinning. The residue of openings, based on Lantuéjoul's formula [40], usually leads to disconnected skeleton branches, whereas methods based on homotopic thinning transformations [40–42,37] yield connected skeletons. Constraining thinning by distance-to-boundary order [43–45] or flux-order [46] further enforces centeredness. Further details on related work and such methods are given in Section 3.2.

2.2. The challenge of comparison

Unsurprisingly, the wealth of existing skeletonization methods makes an exhaustive comparison hard. Aspects which contribute to this challenge are (a) different shape representations (voxels *vs* meshes *vs* point clouds), (b) the unavailability of several implementations, and (c) different skeleton definitions. The last aspect is particularly important: For *surface* skeletons, one could argue that Eq. (2) is a unique definition against which all methods can be checked. However, both spatial discretizations of Eq. (2) and heuristic regularizations that remove small-scale 'noise' details allow multiple weak forms of Eq. (2) [26,7,19]. For *curve* skeletons, the problem is even harder, as these have no unique definition, not even in the continuous $\mathbb{R}^3$ space.

Such aspects make it hard to analytically compare, and reason about, the properties of the produced skeletons. As such, qualitative comparisons have been proposed. In 2007, Cornea et al. compared four curve-skeletonization methods, one from each class listed in Section 2.1. To facilitate the comparison, they also propose several quality criteria that skeletons should obey. Six years later, this comparison was extended for six other contraction-based curve-skeletonization methods [5]. Schaap et al. proposed a quantitative comparison of 13 centerline extraction algorithms for coronary artery datasets [55]. As reference, they use a centerline constructed by manual annotation by expert users. However, in contrast to the tubular artery shapes considered in [55], manual construction of curve, and even more so of surface, skeletons for general 3D shapes not feasible. A cursory scan over many skeletonization papers shows that such method comparisons are very limited (see Table 1). As such, more comparison studies are strongly needed to better understand the strengths and limitations of existing methods.

3. Methods

We next describe a study that adds 4 surface and 6 curve skeletonization methods for voxel shapes to the existing comparison surveys mentioned in Section 2. Section 3.1 presents the desirable criteria that we compare against. Section 3.2 introduces the methods selected for comparison.

3.1. Comparison criteria

Following [3,1,5,8], we focus on the following well-known quality criteria for curve and surface skeletons:

Homotopy: The skeleton is topologically equivalent to the input shape (same number of connected components, cavities, and tunnels).

Thin: The skeleton should be as thin as the sampling model used allows it. Voxel-based skeletons should be one voxel thick, *i.e.*, no 2×2 foreground-voxel configurations should exist.

Centered: For surface skeletons, this is equivalent to Eq. (2). For curve skeletons, no unique centeredness definition exists. An

Please cite this article in press as: A. Sobiecki et al., Comparison of curve and surface skeletonization methods for voxel shapes, Pattern Recognition Lett. (2014), http://dx.doi.org/10.1016/j.patrec.2014.01.012

useful weak form of curve-skeleton centeredness says that the curve skeleton should be a subset of the surface skeleton, since the latter is by definition centered in the shape [50,54,8].

Smoothness: As centeredness, smoothness is hard to formally define. Surface skeleton manifolds are at least $\mathcal{C}^2$ continuous [2,1]. Curve-skeletons are centered subsets thereof [50,54]. Hence, it is arguable that curve skeletons should be also per-branch $\mathcal{C}^2$. In any case, curve skeletons should not exhibit curvature discontinuities induced by the sampling of either the input surface or curve skeleton representation.

Regularization: Skeletons should capture fine-scale details, such as bumps or edges, of the input shape. Users should be able to select the scale of significant details which the skeleton should capture. All smaller-scale details are regarded as noise, and should thus be eliminated. This criterion subsumes the so-called noise robustness and detail preservation criteria. Since the definition of noise *vs* details is an application-dependent scale issue, we chose to use here instead the criterion of *regularization*, defined as the ability of user-controlled skeleton simplification [20,8,54].

Sampling robustness: The difference between skeletons computed for two voxel samplings of a shape should be proportional with the input-sampling differences.

Scalability: Skeletonization methods should be able to extract skeletons of large (1024^3 or similar) voxel volumes in (tens of) seconds on a modern PC with 16 GB RAM.

3.2. Selected methods for comparison

To select actual methods, we used the following criteria:

- *model:* We study only voxel-based methods. Mesh-based methods were recently separately covered in [5];
- *type:* We chose methods in each class (distance-field, general-field, and thinning). Geometric methods are not studied, as these typically use a mesh representation;
- *coverage:* We chose to compare several methods not surveyed by [3];
- *quality:* We chose methods whose advertised features match the quality criteria in Section 3.1, in particular, methods which can handle large voxel volumes;
- *generality:* We chose methods that handle any shapes, regardless of form, complexity, or genus. In particular, this eliminates methods that cannot handle shapes with tunnels, methods that only work for tubular shapes and/or branch-less shapes;
- *availability:* We chose methods with a public (or easily replicable) implementation, so our results can be verified.

Using these criteria, we selected 6 curve skeletonization (*CS*) and 4 surface skeletonization (*SS*) methods, as follows (note that some methods produce both curve and surface skeletons):

Integer Medial Axis transform (IMA, *CS*): Roerdink et al. proposed IMA, a distance-field method that computes one feature point per voxel (also called single-point feature transform [7]) of $\partial\Omega$. Regularized surface skeletons are found as those voxels whose 27-neighborhoods contain feature points located on $\partial\Omega$ further apart than a user-given value γ, similar to the distance-and-angle regularization in [27]. IMA has a time complexity that is linear in the number of input voxels, is very simple to implement, and can be easily parallelized.

Multiscale Skeletons (MS, *SS* and *CS*): Reniers et al. proposed MS [7], a general-field method that computes the surface skeleton following Eq. (2) using the feature transform of [56]. Surface skeletons are regularized by an importance metric equal to the length of the shortest path on $\partial\Omega$ between feature points. Curve skeletons are detected as those surface-skeleton points admitting two different such shortest paths, following [34], and regularized based on the area enclosed by the two shortest paths mentioned above. Thresholding the importance delivers a hierarchy of nested skeletons describing the shape at different scales. Skeleton connectivity is implied by the conjectured, but not proved, monotonicity of the importance metric.

Hamilton–Jacobi Skeletons (HJ, *SS* and *CS*): Siddiqi et al. proposed HJ, one of the first general-field methods. HJ detects medial points, which coincide with the shocks of the grassfire flow, as points where the average outward flux of the distance-transform gradient is non-zero [25]. A flux-ordered homotopy-preserving thinning is used to simplify the surface skeleton. With sufficient simplification, curve skeletons are obtained. The idea has been further enhanced with subpixel flux calculations andimproved with error correction in areas of large curvature [57]. To our knowledge, this enhancement has only been tested for 2D shapes. We implemented the curvature-correction enhancement for 3D shapes for our comparison.

Distance-Driven Skeletonization (DDS, *CS*): To extract surface skeletons, Arcelli et al. combine iterative thinning and distance-field methods to remove non-skeletal voxels while updating the distance transform based on a $\langle 3,4,5\rangle$ scheme [45]. From these, curve skeletons can be further extracted by a similar thinning based on several heuristics that approximate the distance transform $DT_{\partial S}$ of the surface skeleton boundary ∂S over the surface skeleton S. Both skeleton types are further regularized by an importance metric similar in concept with, but implemented differently from, the collapse metric in [7]. DDS guarantees connected, voxel-thin, and rotation-invariant skeletons.

Thinvox (TV, *CS*): TV implements the 3D directional thinning proposed by Pálagyi and Kuba [37] to compute curve skeletons. Several computational optimizations are added, including GPU acceleration. TV is part of the *binvox* package [58]. Given the wide popularity and usage of *binvox*, we included TV in our comparison.

Iterative Thinning Process (ITP, *SS* and *CS*): Ju et al. compute, with ITP, skeletons of volumetric models by alternating thinning and a novel skeleton pruning routine [52]. ITP creates a family of skeletons parameterized by two user-specified values that determine respectively the size of curve and surface features on the skeleton. The method was, however, tested mainly on tubular-and-thin-plate shapes.

Robust Thinning (RT, *CS*): In [51], Liu et al. propose RT, a thinning method that works on cell-complex representations built using voxelization techniques. RT uses a 'medial persistence' importance metric that discriminates object parts with different anisotropic elongations, *e.g.*, tubes or plates, similar to ITP [52]. Based on medial persistence, RT can produce a continuum between surface-and-curve skeletons and 'pure' curve skeletons, and is claimed to be more robust to noise than ITP. In our comparison, we used only the curve skeletons produced by RT, as the other tested methods do not produce mixed skeletons.

4. Comparison methodology

We used the selected methods to extract curve and surface skeletons from a set of 38 shapes available in PLY mesh format. Shapes range from simple to very complex in terms of topology (branches and tunnels), surface detail, number of triangles (30 K to over 1 M), and cover both synthetic and natural objects. Fig. 1 (left column) shows a selection. First, we used *binvox* [58] to voxelize the shapes to several resolutions ranging from 128^3 to 1024^3 voxels. Next, we ran each method on each voxel shape, tuning the method's regularization parameters (if any) so as to (a) eliminate spurious (noise) branches but (b) keep detail branches. In total, several hundreds of skeletonization runs were performed. Finally, for each method, and based on the values found in the previous pass, we chose a

Please cite this article in press as: A. Sobiecki et al., Comparison of curve and surface skeletonization methods for voxel shapes, Pattern Recognition Lett. (2014), http://dx.doi.org/10.1016/j.patrec.2014.01.012

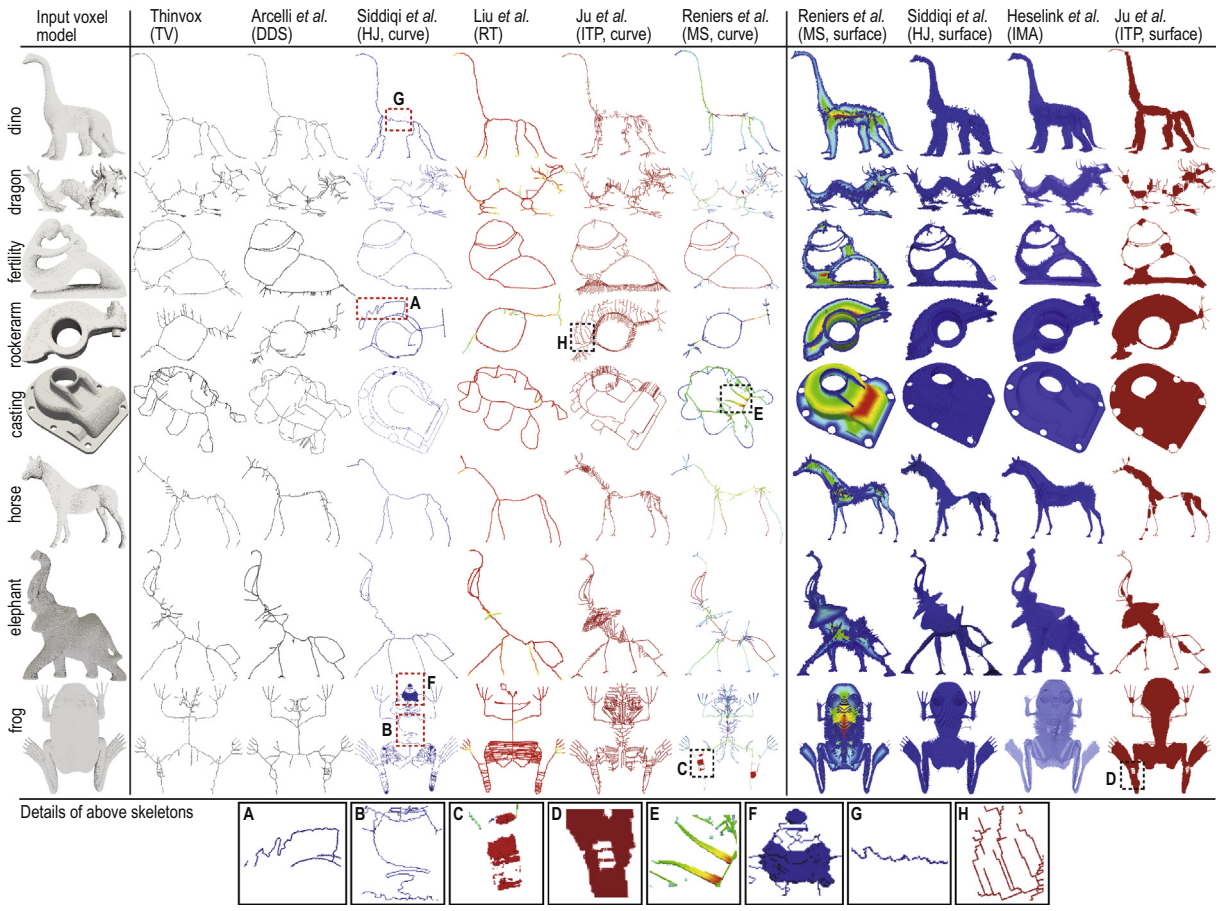

Fig. 1. Curve and surface skeletons of different shapes, computed, from left to right, by 6 curve and 4 surface skeletonization methods (see Section 4). The input volume has a resolution of 512^3 voxels. Dotted markers indicate various problems of the produced skeletons and underlying methods. Details thereof are shown below the main image. (see Section 5.1).

fixed set of parameter values that gives overall good results. These values were next used for all shapes treated by the respective method. Obtaining such results was quite easy for all studied methods, except ITP-curve, where we could not always eliminate spurious branches *and* keep important ones intact at the same time for all studied shapes. For testing, we used two 3.5 GHz, PC with 32 GB RAM, running Windows 7 and Linux respectively, depending on the requirements of each method's implementation. The datasets and software implementations used are publicly available at [59].

5. Results

Our results include two parts: A global comparison (Section 5.1) and a detailed comparison (Section 5.2). Both are described next.

5.1. Global comparison

Homotopy: All studied SS methods captured well the input shape topology, including protrusions and tunnels, and delivered connected skeletons (Fig. 1, 4 right columns). The only method that had issues here was ITP. The *frog* model (Fig. 1, bottom row) is an example of such issues: Although the input is of genus 0, the ITP surface skeleton exhibits a number of small spurious holes in the leg regions (Fig. 1, detail *D*). For curve skeletons, we see more variation: For most tubular shapes, all CS methods produce skeletons which match the input's topology. For non-tubular shapes, like *rockerarm* and *casting*, the topology of the produced results varies widely. The *frog* model is also the most challenging for CS methods: Although the model is relatively smooth and has, in most areas, a tubular structure, the computed curve skeletons vary strongly between all methods.

Thin: For our methods, this criterion implies one-voxel-thin skeletal manifolds and curve skeletons. Visual inspection shows that not all methods satisfy this. For surface skeletons, MS and HJ exhibit small-size thick clusters of several voxels around the manifold Y-intersection curves. IMA and ITP-surface perform the best. For curve skeletons, all methods produce voxel-thin curves, except HJ and MS, which create spurious thick *surface* fragments (see Fig. 1, details *E*, *F* on *casting* and *frog*). This is explained by the fact that, in contrast to the other studied methods, HJ and MS do not have an explicit thinning step or similar postprocessing to guarantee voxel-thin skeletons.

Centered: Visual comparison of the four SS types computed shows that these appear well centered within their input shapes. This may appear less evident for ITP (Fig. 1, rightmost column). However, closer inspection shows that ITP differs from the other skeletons mainly around the boundaries of the skeletal manifolds, *i.e.*, in regions where different methods use different degrees of simplification. In 'core' areas, ITP skeletons are very similar to the other surface skeletons computed. For CSs, centeredness differences range from small for simple models (*horse*, *fertility*) to visibly large ones (*rockerarm*, *frog*). This is partially expected, since all studied methods use different CS definitions. However, we also found CS fragments which arguably cannot be centered within

Please cite this article in press as: A. Sobiecki et al., Comparison of curve and surface skeletonization methods for voxel shapes, Pattern Recognition Lett. (2014), http://dx.doi.org/10.1016/j.patrec.2014.01.012

any reasonable centeredness definition – see *e.g.* details A, B on *rockerarm* and *frog*, Fig. 1. To better assess centeredness, we propose detailed skeleton comparison further in Section 5.2.

Smoothness: The studied four SS methods exhibit negligible differences in terms of the smoothness of the extracted skeletal manifolds. In contrast, CS methods show a wide variation here. DDS, TV, and MS produced overall very smooth CS branches for all studied models. RT creates small-scale staircase effects, slightly larger than the voxel resolution used. However, these can be easily eliminated by increasing resolution. HJ creates several unexpected wiggles of various scales in the CS (*e.g.* Fig. 1, details A, B, G on *rockerarm*, *frog*, and *dino* respectively). We could not correlate the wiggles' appearance with shape properties such as curvature, thickness, smoothness, or voxel resolution. A possible explanation of these effects is that HJ is mainly designed to compute (simplified) *surface* skeletons. Highly simplifying these skeletons *can* produce curve skeletons, but the simplification order is *only* constrained by homotopy preservation. As such, HJ cannot guarantee that a skeletal manifold gets simplified with equal speed from all its boundaries inwards. When this does not happen, CS structures will still be contained by the SS, but not centered with respect to the SS boundaries. Finally, ITP creates coarse-scale staircase artifacts along CS branches (*e.g.* Fig. 1, *rockerarm*, detail H). This can explained by the fact that, unlike *e.g.* DDS or TV, ITP does not impose any geometry-based voxel removal order order during the thinning process.

Regularization: All studied SS and CS methods, except TV and DDS, offer one or more parameters to eliminate skeleton branches corresponding to small-scale noise on the input shape. TV does not need such a parameter for its curve skeletons, as these are noise-resistant by construction [37]. The DDS implementation we obtained from its authors uses the fixed simplification values indicated in [45]. To assess, in a global manner, the ease of noise elimination, we showed in Fig. 1 the results obtained by using a fixed set of regularization parameters, determined as described in Section 4.

Further increase of the simplification level should keep the most important skeletal branches, eliminating less important ones. To study the effect of the regularization parameters on simplification, we next varied these parameters for each method in order to progressively simplify the produced skeletons. Fig. 2 shows this for four progressively simplified instances of the *dragon* skeleton for seven of the studied methods. Here, as we increase simplification, we notice growing differences between the studied methods. Among CS methods, MS and ITP offered the most intuitive way to eliminate small-scale details and keep the main skeleton structure. HJ-curve was the hardest to control: Too little simplification preserves spurious branches, such as the clearly not-centered branch on the dragon's back (Fig. 2, detail A). Increasing simplification removes this branch, but also looses important skeletal structures such as the dragon's legs and tail, which is an undesired effect (Fig. 2, details B, C). Among SS methods, MS and IMA simplify quite

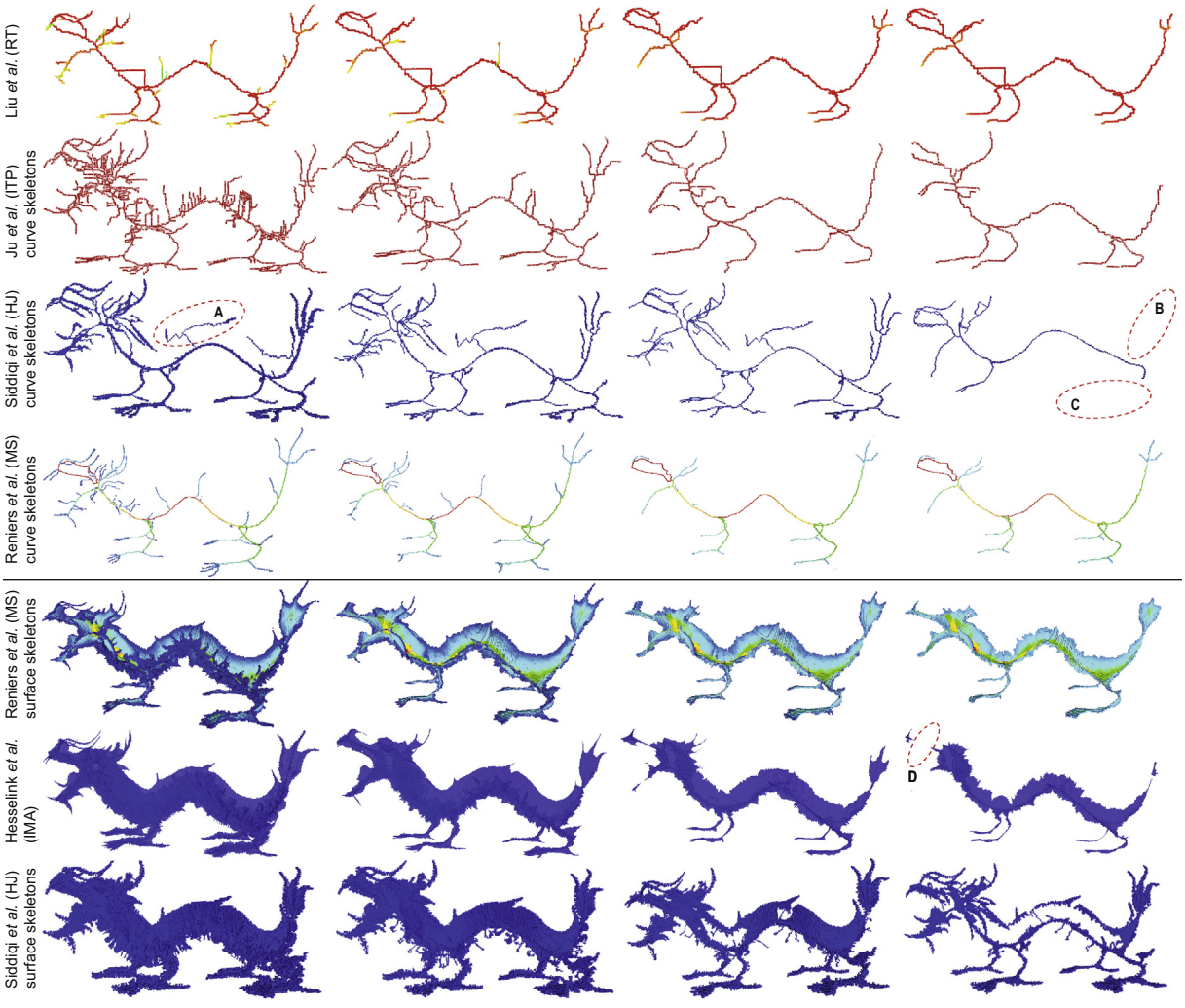

Fig. 2. Regularization of curve skeletons (top 4 rows) and surface skeletons (bottom 3 rows). See Section 5.1.

Please cite this article in press as: A. Sobiecki et al., Comparison of curve and surface skeletonization methods for voxel shapes, Pattern Recognition Lett. (2014), http://dx.doi.org/10.1016/j.patrec.2014.01.012

similarly, the effect being of removing SS voxels in increasing distance from the SS boundary. However, higher simplification values disconnect the IMA SS (Fig. 2, detail *D*). In contrast, MS never disconnected the SS, even for high simplification values. Simplifying HJ-surface skeletons also proved quite challenging: The divergence-based importance used by HJ is good for removing small-scale noise, but produces arguably incorrect results, like genus changes, when higher values are used (Fig. 2, bottom row). The explanation resides in the fact that the divergence metric does not have a monotonic evolution from the SS boundary inwards, but can exhibit various local maxima [57].

Sampling robustness: Fig. 3 shows CS and SS results from the studied methods, each ran on several instances of *dragon*, sampled at resolutions between 200^3 and 512^3 voxels. For RT, we only used 256^3 and 512^3 voxels, since this method admits only power-of-two resolutions. We chose *dragon* for testing, as it is the shape having the highest amount of complex surface detail from our studied model collection. Regularization is tuned so as to obtain the visually most similar results for all methods. For both SS and CS methods, we see that, as resolution increases, progressively more skeletal details get captured, as expected. For SS methods and also for TV, MS-curve, and RT, as resolution increases, the overall skeleton shape does not change significantly. However, methods that not enforce homotopy (IMA and MS-curve) show skeleton disconnections at lower resolutions (Fig. 3, details *A-C*). These disappear when resolution is increased. The situation is more subtle for the remaining CS methods: HJ-curve, DDS, and TV show small-scale branch twists, and even topological events (loops) which appear and next disappear as resolution changes (Fig. 3, details D, E, F). Upon closer inspection, we see that these loops are correct, as they correspond to a tunnel in the input shape. In contrast, the lack of such a tunnel for MS is incorrect. For ITP-surface and ITP-curve, a large part of the SS, respectively CS structure varies with resolution in a hard-to-control manner. We also note that we could not run all methods for all resolutions – the empty places for MS and ITP in Fig. 3 correspond to resolutions for which the respective implementations crashed for the *dragon* model.

Scalability: Table 2 presents computational aspects of the tested methods for the models in Fig. 1. To make results independent of the input shape and its sampling, we computed *Speed* as the number of processed input foreground voxels per millisecond. This way, background voxels, which do not request computations in the tested methods, are ignored. In contrast, *Memory* gives the peak memory usage divided by the input volume size (voxels) for normalization, since the studied methods allocate data arrays of the size of the entire input volume (foreground and background voxels). Intuitively, *Speed* can be seen as the throughput of a given method, while *Memory* can be seen as the space cost per input voxel. The tested volumes range between 128^3 and 1024^3 voxels. For HJ, we include a single measurement for both the curve and surface skeletons, since both are computed using the same algorithm, the difference being only the simplification level applied as postprocessing. Absolute memory usage ranges from 1.5 GB (IMA, *fertility*) to 23.7 GB (HJ, *elephant*).

Several observations can be made. First, TV and IMA are the fastest methods. Even if factoring the possible lack of optimizations in the slower methods, this can be explained by the fact that TV and IMA use quite simple algorithms. More interestingly, we see that the throughputs of all methods are relatively uniform, less so for HJ and MS. This indicates that most studied methods scale computationally well. For HJ, this can be explained by the particular homotopy-preserving thinning used, whose complexity depends on the number of detected skeleton end-points [57]. For MS, this is explained by the regularization metric used, which requires computation of geodesics between skeletal feature points. Both above operations are, indeed, strongly dependent on the input's shape, and not only on its size. Memory-wise, we see quite large differences: Here, again, TV and IMA require less than one order of magnitude less memory than the most expensive method, HJ. As for speed, this is explained by the relative simplicity of TV and IMA as compared to general-field methods – the former need only two such fields (input volume and skeleton), while the latter typically need to store many 3D floating-point fields over the entire volume.

A different outlier is visible for the *dragon* dataset, which is markedly slower to process than the other considered models. As *dragon* is the model having by far the most amount of surface detail, it will generate the most complex skeletons in terms of number of medial sheets or curves. Our observation that speed is related to skeleton complexity matches the similar separate observations in [7,30]. However, we also see that the DDS and RT methods are less affected by the *dragon* model complexities in terms of performance. Although these methods are, on average, slower than the fastest considered methods, their throughput is much more *stable*, *i.e.* fluctuates less, for different models. This stability is an advantage in practice, as it allows one to estimate upfront the computational time required by for skeletonizing a given model.

5.2. Detailed comparison

The global comparison presented in Section 5.1 outlines differences between the studied methods in terms of all criteria in Section 3.1, except centeredness. Assessing centeredness differences from image pairs is harder, since such differences can be small-scale, local, and subtle. We next propose a visualization method that addresses the following centeredness questions:

- Given two surface skeletons SS_1 and SS_2, or two curve skeletons CS_1 and CS_2, which are the differences?
- Given a surface skeleton SS_1 and a curve skeleton CS_2, how well is CS_2 contained in SS_1?

Given two (curve or surface) skeletons S_1 and S_2, sampled over the same volume, we first define the scalar distance field

$$D_{12}(\mathbf{x} \in \mathbb{R}^3) = \begin{cases} \min_{\mathbf{y} \in S_2} \|\mathbf{x} - \mathbf{y}\| = DT_{S_2}(\mathbf{x}) & \text{if } \mathbf{x} \in S_1 \\ 0 & \text{if } \mathbf{x} \notin S_1 \end{cases}$$

To compare two skeletons of the same kind (CS_1 *vs* CS_2, or SS_1 *vs* SS_2), we draw the field $D_{12} + D_{21}$ over the voxel union $S_1 \cup S_2$, normalized by its maximum value, using a rainbow (blue-to-red) colormap. Voxels in a skeleton which are close to the other skeleton are blue. Note that $D_{12}(\mathbf{x}) = D_{21}(\mathbf{x}) = 0, \forall \mathbf{x} \in S_1 \cap S_2$. Voxels in a skeleton which are far away from the other skeleton are red (see Fig. 4, inset). Comparing a curve skeleton CS_1 with a surface skeleton SS_2 is done differently, since we now want to show how well is CS_1 *contained* within SS_2. For this, we color voxels in CS_1 with D_{12}, and voxels in $SS_2 \setminus CS_1$ with gray. Voxels in CS_1 which are included in SS_2 are blue. Voxels in CS_1 far from SS_2 become red.

Fig. 4 shows a subset of the performed comparisons, for several method-pairs, for the *dragon* model, at 500^3 resolution. Each row i or column j corresponds to a method; the image (i,j) shows the comparison of the skeleton-pair (S_i, S_j). Fig. 5 shows the aggregated maximum and average distances between the considered skeleton pairs (S_i, S_j) for five different sampling resolutions. All distances are given as functions of the voxel size at resolution 100^3, *e.g.*, the size of a voxel at resolution 200^3 is $1/2$, the size of a voxel a resolution 300^3 is $1/3$, and so on, so that we can assess how distances change with resolution. From Figs. 4 and 5, the following observations can be made.

Surface-*vs*-surface: Surface skeletons are very similar with each other, except at tips – see *e.g.* Fig. 4 (HJ *vs* IMA), which has

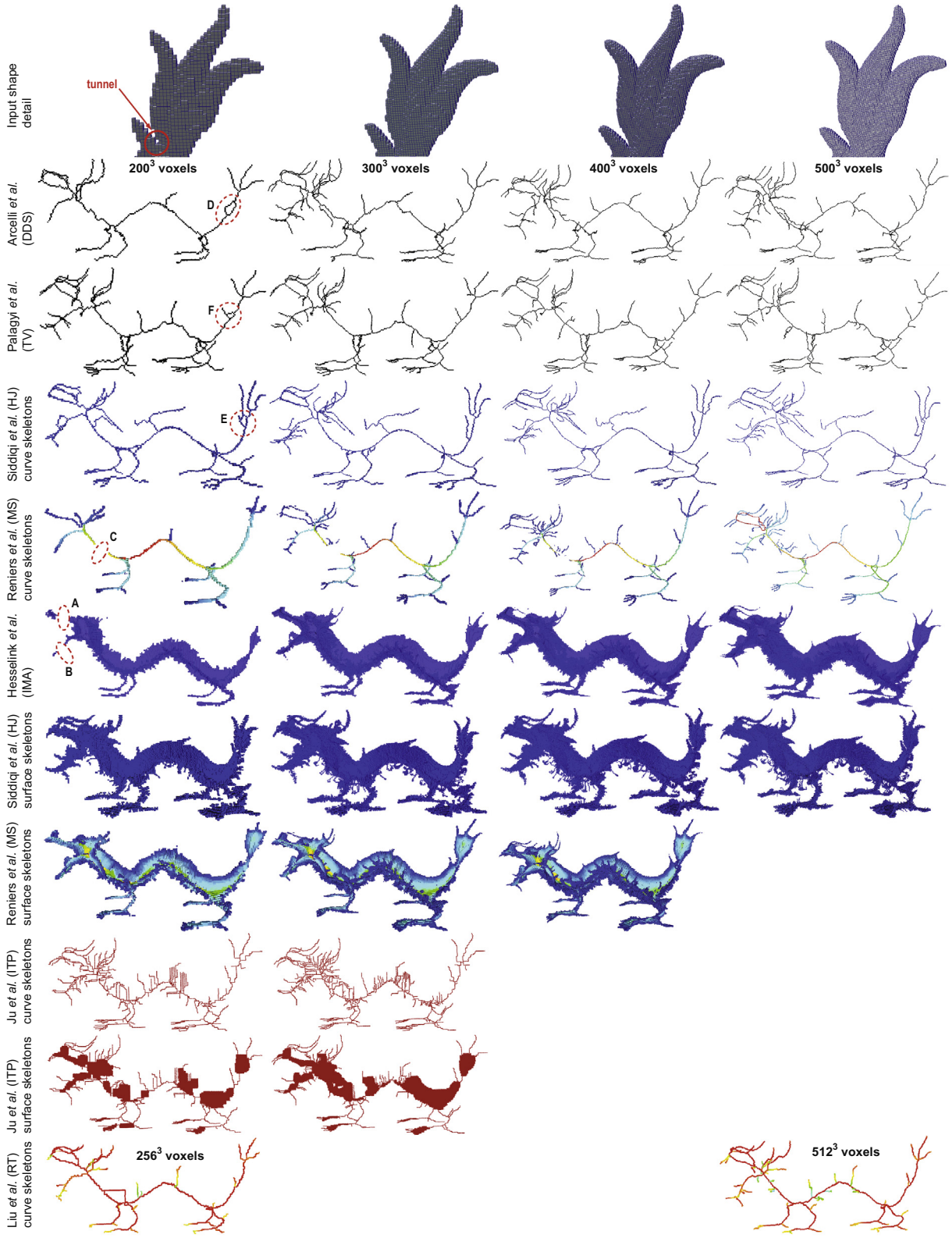

Fig. 3. Sampling robustness of all studied methods for different voxel resolutions (see Section 5.1).

an overall (dark) blue color. We noticed the same similarity in the other *SS-vs-SS* comparisons we did (not shown here for sake of space). The red tips in Fig. 4 (HJ *vs* IMA) indicate minor differences in terms of a few skeleton boundary voxels. These are expected given the different pruning heuristics of the considered methods. The fact that surface skeletons are quite similar strengthens (but does not prove) our hypothesis that centeredness is well captured – indeed, it would be surprising that methods using fundamentally different principles would yield the *same* errors.

Curve-*vs*-curve: Curve skeletons exhibit largest differences in terminal and central regions. In terminal regions, one skeleton can be longer than the other, as shown by the red tips of several

Please cite this article in press as: A. Sobiecki et al., Comparison of curve and surface skeletonization methods for voxel shapes, Pattern Recognition Lett. (2014), http://dx.doi.org/10.1016/j.patrec.2014.01.012

Table 2
Performance comparison for models in Fig. 1. *Speed* gives the number of processed foreground voxels per millisecond. *Mem.* gives the peak memory usage (bytes/voxel) per input volume size (see Section 5.1).

Methods	TV		DDS		HJ curve/surface		RT		ITP, curve		MS, curve		MS, surface		IMA		ITP, surface	
	Speed	Mem.	Speed	Mem.	Speed	Mem.	Speed	Mem.	Speed	Mem.	Speed	Mem.	Speed	Mem.	Speed	Mem.	Speed	Mem.
Dino	109.5	28.8	9.9	45.12	36.9	192	14.7	250	60.9	127	30.6	148	38.32	148	62.5	14.2	66.5	127
Dragon	61.2	24.6	10.6	46.8	14.8	597	10.8	175	25.7	127	7.2	135	27.2	120	45.8	11.9	25.7	127
Fertility	133.0	27.7	13.2	43.3	30.9	507	16.8	204	51.9	144	15.4	165	25.6	171	158.0	14.3	51.9	144
Rockerarm	111.4	28.4	14.3	50.7	52.3	270	22.3	203	51.9	114	17.9	161	13.5	161	99.4	13.9	49.3	120
Casting	109.8	31.7	12.6	45.6	49.7	322	19.7	203	48.6	114	9.3	124	29.1	124	87.5	14.1	53.2	115
Horse	120.5	29.2	10.2	47.3	52.1	270	25.9	204	54.9	125	13.7	145	32.9	157	94.3	14.0	54.9	127
Elephant	112.8	28.9	10.6	47.6	42.3	190	9.1	203	43.8	120	20.8	153	30.4	174	109.7	14.3	42.2	120
Frog	123.0	31.8	12.9	47.12	54.1	240	16.0	183	48.1	126	10.5	168	63.3	125	148.5	12.8	65.6	122

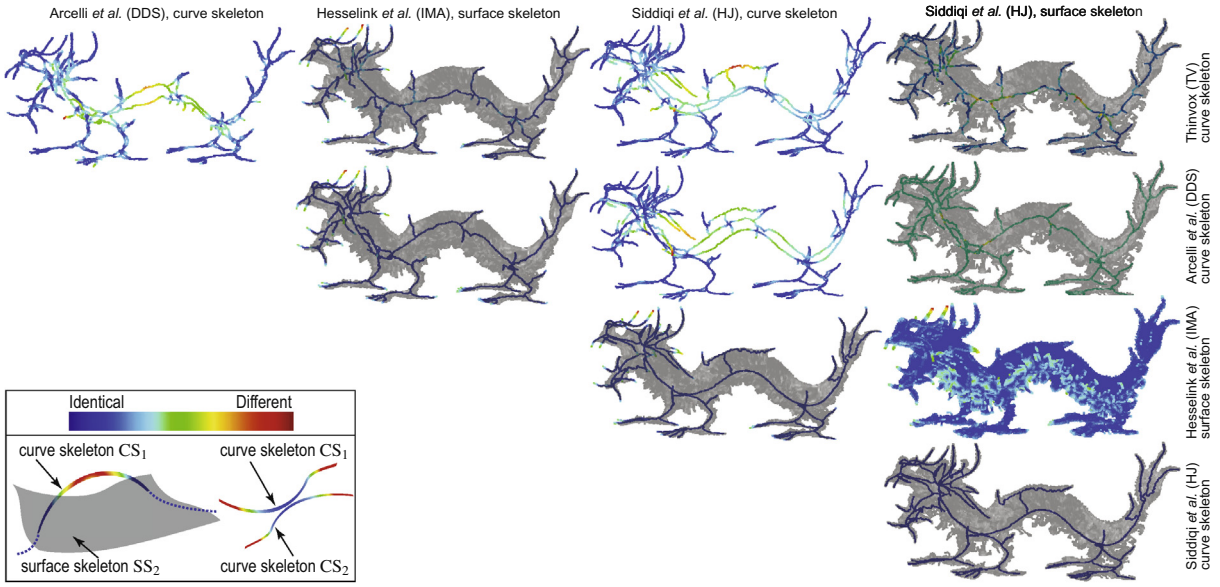

Fig. 4. Detailed pair-wise comparison of curve *vs* curve, curve *vs* surface, and surface *vs* surface skeletons. Inset shows color mapping method.

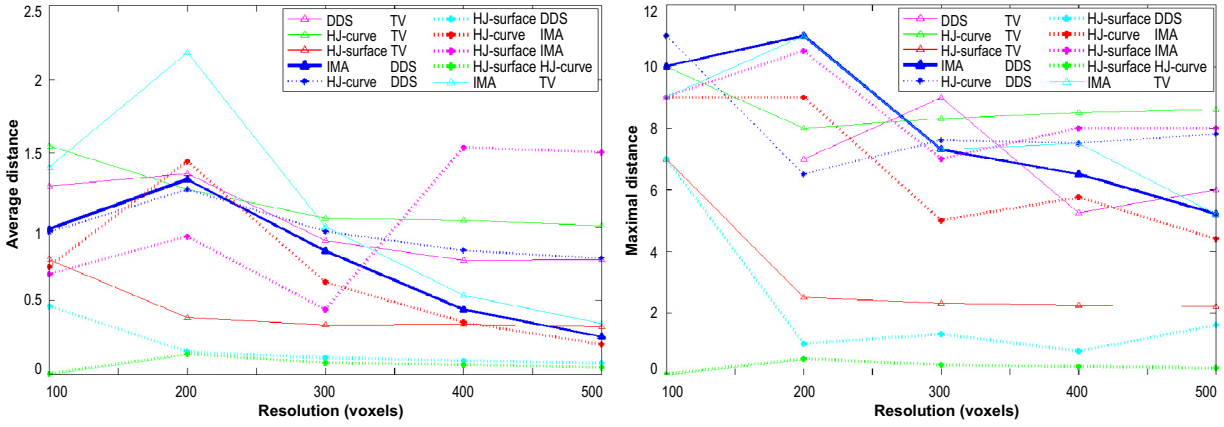

Fig. 5. Average and maximal distances between skeletons computed by different methods at different resolutions (see Fig. 4).

skeleton branches. As for the *SS-vs-SS* case, this is due to the different pruning heuristics of the studied methods, and is an expected result which does not show lack of centeredness. However, along the dragon's central rump region, curve skeletons follow parallel, but quite different, paths. This means that *at least* one of the studied curve skeletons is poorly centered. This is a less expected insight, which cannot be inferred easily from the typical curve-skeleton images shown in typical skeletonization papers.

Curve-*vs*-surface: A perfect *CS*-in-*SS* inclusion is achieved only by the HJ curve and surface skeletons (*CS* voxels are all dark blue in Fig. 4, HJ-curve-*vs*-HJ-surface). This is expected, as both the *CS* and *SS* are computed by the same base method (HJ). For all other studied cases, we see a number of warm-colored curve-skeleton voxels (*e.g.* Fig. 4: TV-*vs*-HJ-surface, TV-*vs*-IMA, DDS-*vs*-IMA). Highest differences occur at skeleton tips, which is expected, as already explained. However, the pair HJ-surface-*vs*-TV shows such red voxels

Please cite this article in press as: A. Sobiecki et al., Comparison of curve and surface skeletonization methods for voxel shapes, Pattern Recognition Lett. (2014), http://dx.doi.org/10.1016/j.patrec.2014.01.012

also deep inside the curve skeleton. Overall, we conclude that curve skeletons are generally well centered with respect to the medial surface, but less well centered *within* this surface.

Resolution: Fig. 5 shows the variation of the maximal and average distances between the compared skeleton pairs for sampling resolutions ranging between 100^3 and 500^3 voxels. Several observations can be made. On average, the compared skeletons are quite close to each other (1 to 2 voxels), but the maximal distances can be large (up to 11 voxels). The minimal distances (not shown in the figure) are zero, for all considered skeleton-pairs. As resolution increases, most (but not all) compared skeletons tend to become closer, both in terms of average and maximal distances. This is not surprising, since higher resolutions should allow a finer-grained placement of skeleton voxels, thus a better approximation of the actual skeletal locus. However, as the measured differences do not monotonically decrease with resolution in all cases, this suggests that there exist structural differences between the skeletons computed by the studied methods, which cannot be solved simply by a finer sampling. Interestingly, HJ-surface is the method which participates in the most similar skeleton-pairs, both in terms of average and maximal distances, for all resolutions. Conversely, IMA participates in the least-similar skeleton pairs. If we assume that all methods are, statistically speaking, equally valid with respect to centeredness, this implies that HJ-surface delivers a well-centered 'consensus' skeleton, and IMA delivers an outlier, far less well centered, skeleton.

6. Discussion

Several points emerge from our comparison, as follows:

Best method: First, let us stress that the aim of our comparison was not to designate the 'best' skeletonization method, but to highlight pro's and con's of the studied methods with respect to a set of accepted quality criteria. From this viewpoint, no CS or SS method can be seen as optimal with respect to *all* considered criteria. For surface skeletons, all methods create well-centered, noise-free, skeletons. Noise (or detail) removal is the most intuitive with MS. The other studied methods can remove noise or details, but are less intuitive to control (produce disconnections, modify the topology, or remove skeletal parts which one may regard as 'core' to the skeleton). For curve skeletons, DDS and TV are simple to use (require no parameters), and produce clean, noise-free, but still detailed, and reasonably smooth, curve-skeletons. However, they offer less freedom for skeleton simplification. MS-curve arguably produces the smoothest curve-skeletons, but does not guarantee voxel-thickness. IMA produces centered, smooth, skeletons, and is very simple to use, but has less intuitive simplification parameters. HJ can produce high-quality surface skeletons. However, HJ cannot produce centered curve skeletons for more complex shapes, and also has a less intuitive simplification control. Performance-wise, TV and IMA are the fastest methods, and the least demanding in memory terms. However, the differences with the other studied methods are not that large so as to warrant a clear winner.

Criteria: The covered comparison criteria are clearly not exhaustive. Additional ones exist, *e.g.*, skeleton invariance to isometric transformations of the input shape [3,45] and input reconstructibility from the skeleton [45,8]. Given the available space, we chose to focus in more depth on a smaller number of criteria. Studying how CS and SS methods perform on additional criteria is subject for separate future work.

Methods: Our selection of compared methods cannot cover all existing CS and SS techniques in existence, so our findings cannot be directly generalized to any method. However, our comparison outlined several non-evident challenges of a good representative subset of recent methods. Most of the studied methods could extract curve and surface skeletons from a large variety of complex shapes. However, we also discovered several problems with respect to all considered quality criteria (except scalability), which are visible only for certain combinations of complex input shapes and method-specific parameter settings. As such, we argue that more comparative studies are required for a better understanding of the added value and limitations of skeletonization methods.

Standard: The various discretization and regularization techniques used in skeletonization algorithms, together with the lack of a unique formal CS definition, make comparisons of a given algorithm with a 'gold standard' difficult. Evident problems, such as thick or disconnected skeletons, are easy to check for. However, checking criteria such as smoothness, noise robustness, and CS centeredness, is much harder. As such, the question of what is the 'correct' skeleton of a given shape is very hard to answer in general. Rather than trying to answer this question, we advocate a comparative approach that highlights *differences* between several skeletonization methods with respect to input shape, input resolution, and simplification parameters.

7. Conclusion

In this paper, we presented a comparison of six curve-skeletonization and four surface-skeletonization methods using voxel models. Compared to existing surveys in the area [3,5], we extend insights by discussing ten methods (not covered by previous surveys) with respect to established quality criteria for curve and surface skeletons. We compare methods on a range of 3D shapes ranging from simple to complex, covering both natural and synthetic forms, and consider the effects of several parameters such as simplification level and input resolution on the obtained skeletons. We include also a quantitative performance in terms of speed and memory requirements. Finally, we propose a detail visualization able to highlight small-scale centeredness differences between curve and surface skeletons. Our work highlights challenges of, and differences between, existing 3D skeletonization methods which to our knowledge have not been highlighted in the literature. On a higher level, our results expose several limitations of current skeletonization methods and underline the need for future work towards extending such comparisons and also towards creating better methods.

Future work in skeletonization comparison involves including additional methods and quality criteria in this comparison. On a more theoretical level, a promising direction is to devise new metrics for the quantitative comparison of the desired quality criteria in ways that help algorithm designers pinpoint and next solve causes for current limitations of such methods.

Acknowledgement

This work was financially supported by the grant 202535/2011-8 offered by CNPq, Brazil.

References

[1] K. Siddiqi, S. Pizer, Medial Representations: Mathematics, Algorithms and Applications, Springer, 2009.
[2] S. Pizer, K. Siddiqi, G. Szekely, J. Damon, S. Zucker, Multiscale medial loci and their properties, Int. J. Comput. Vision 55 (2–3) (2003) 155–179.
[3] N. Cornea, D. Silver, P. Min, Curve-skeleton properties, applications, and algorithms, IEEE Trans. Visual Comput. Graphics 13 (3) (2007) 87–95.
[4] H. Blum, Biological shape and visual science, J. Theor. Biol. 38 (1973) 205–287.
[5] A. Sobiecki, H. Yasan, A. Jalba, A. Telea, Qualitative comparison of contraction-based curve skeletonization methods, in: Proc. International Symposium on Memory Management, Springer, 2013, pp. 425–439.
[6] P. Giblin, B. Kimia, A formal classification of 3D medial axis points and their local geometry, IEEE Pattern Anal. Mach. Intell. 26 (2) (2004) 238–251.

[7] D. Reniers, J.J. van Wijk, A. Telea, Computing multiscale skeletons of genus 0 objects using a global importance measure, IEEE Trans. Visual Comput. Graphics 14 (2) (2008) 355–368.
[8] A. Jalba, J. Kustra, A. Telea, Surface and curve skeletonization of large 3D models on the GPU, IEEE Trans. Pattern Anal. Mach. Intell. 35 (6) (2013) 1495–1508.
[9] J. Damon, Global medial structure of regions in $\mathbf{R}^3$, Geom. Topol. 10 (2006) 2385–2429.
[10] F. Leymarie, B. Kimia, The medial scaffold of 3D unorganized point clouds, IEEE Trans. Visual Comput. Graphics 29 (2) (2007) 313–330.
[11] M. Chang, F. Leymarie, B. Kimia, Surface reconstruction from point clouds by transforming the medial scaffold, Comput. Vision Image Underst. 113 (2009) 1130–1146.
[12] T. Dey, W. Zhao, Approximate medial axis as a Voronoi subcomplex, Comput. Aided Des. 36 (2) (2004) 195–202.
[13] N. Amenta, S. Choi, R. Kolluri, The power crust., in: Proc. SMA, 2001, pp. 65–73.
[14] O. Au, C. Tai, H. Chu, D. Cohen-Or, T. Lee, Skeleton extraction by mesh contraction, in: Proc. ACM SIGGRAPH, 2008, pp. 441–449.
[15] A. Tagliasacchi, H. Zhang, D. Cohen-Or, Curve skeleton extraction from incomplete point cloud, in: Proc. SIGGRAPH, 2009, pp. 541–550.
[16] J. Cao, A. Tagliasacchi, M. Olson, H. Zhang, Z. Su, Point cloud skeletons via Laplacian based contraction, in: Proc. SMI, 2010, pp. 187–197.
[17] X. Li, T. Woon, T. Tan, Z. Huang, Decomposing polygon meshes for interactive applications, in: Proc. I3D Symp., 2001, pp. 35–42.
[18] H. Huang, S. Wu, D. Cohen-Or, M. Gong, H. Zhang, G. Li, et al., l_1 Medial skeleton of point cloud, ACM Trans. Graphics 32 (4) (2013) 1–8.
[19] B. Miklos, J. Giesen, M. Pauly, Discrete scale axis representations for 3D geometry, ACM Trans. Graphics 29 (4) (2010) 394–493.
[20] J. Ma, S.W. Bae, S. Choi, 3D medial axis point approximation using nearest neighbors and the normal field, Visual Comput. 28 (1) (2012) 7–19.
[21] F. Leymarie, M. Levine, Simulating the grassfire transform using an active contour model, IEEE Trans. Pattern Anal. Mach. Intell. 14 (1) (1992) 56–75.
[22] R. Kimmel, D. Shaked, N. Kiryati, A. Bruckstein, Skeletonization via Distance Maps and Level Sets, Comput. Vision Image Underst. 62 (3) (1995) 382–391.
[23] Y. Ge, J. Fitzpatrick, On the generation of skeletons from discrete Euclidean distance maps, IEEE Trans. Pattern Anal. Mach. Intell. 18 (1996) 1055–1066.
[24] M. Wan, F. Dachille, A. Kaufman, Distance-field based skeletons for virtual navigation, In: Proc. IEEE Visualization, 2001, pp. 239–246.
[25] K. Siddiqi, S. Bouix, A. Tannenbaum, S. Zucker, Hamilton-Jacobi skeletons, Int. J. Comput. Vision 48 (3) (2002) 215–231.
[26] W. Hesselink, J. Roerdink, Euclidean skeletons of digital image and volume data in linear time by the integer medial axis transform, IEEE Trans. Pattern Anal. Mach. Intell. 30 (12) (2008) 2204–2217.
[27] A. Sud, M. Foskey, D. Manocha, Homotopy-preserving medial axis simplification, in: Proc. SPM, 2005, pp. 103–110.
[28] T. Cao, K. Tang, A. Mohamed, T. Tan, Parallel banding algorithm to compute exact distance transform with the GPU, in: Proc. SIGGRAPH I3D, 2010, pp. 134–141.
[29] N. Ahuja, J. Chuang, Shape representation using a generalized potential field model, IEEE Trans. Pattern Anal. Mach. Intell. 19 (2) (1997) 169–176.
[30] N. Cornea, D. Silver, X. Yuan, R. Balasubramanian, Computing hierarchical curve-skeletons of 3D objects, Visual Comput. 21 (11) (2005) 945–955.
[31] C. Aslan, A. Erdem, E. Erdem, S. Tari, Disconnected skeleton: shape at its absolute scale, IEEE Trans. Pattern Anal. Mach. Intell. 30 (12) (2008) 2188–2203.
[32] M. Hassouna, A. Farag, Variational curve skeletons using gradient vector flow, IEEE Trans. Pattern Anal. Mach. Intell. 31 (12) (2009) 2257–2274.
[33] M. Foskey, M. Lin, D. Manocha, Efficient computation of a simplified medial axis, in: Proc. SMA, 2003, pp. 135–142.
[34] T. Dey, J. Sun, Defining and computing curve skeletons with medial geodesic functions, in: Proc. SGP. IEEE, 2006, pp. 143–152.
[35] M. Livesu, F. Guggeri, R. Scateni, Reconstructing the curve-skeletons of 3D shapes using the visual hull, IEEE Trans. Visual Comput. Graphics 18 (11) (2012) 1891–1901.
[36] A. Telea, J.J. van Wijk, An augmented fast marching method for computing skeletons and centerlines, in: Proc. VisSym., 2002, pp. 251–259.
[37] K. Palagyi, A. Kuba, Directional 3D thinning using 8 subiterations, in: Proc. Discrete Geometry for Computer Imagery, Lecture Notes in Computer Science, vol. 1568, Springer, 1999, pp. 325–336.
[38] M. Couprie, D. Coeurjolly, R. Zrour, Discrete bisector function and Euclidean skeleton in 2D and 3D, Image Vision Comput. 25 (10) (2007) 1543–1556.
[39] J. Serra, Image Analysis and Mathematical Morphology, Acad. Press, 1982.
[40] Lantuéjoul C. La, Squelettisation et son Application aux Mesures Topologiques de Mosaiques Polycristallines, School of Mines, Paris, 1979.
[41] F. Meyer, Skeletons and perceptual graphs, Signal Proc. 16 (4) (1989) 335–363.
[42] S. Beucher, Digital skeletons in Euclidean and geodesic spaces, Signal Process. 38 (1) (1994) 127–141.
[43] C. Pudney, Distance-ordered homotopic thinning: a skeletonization algorithm for 3D digital images, Comput. Vision Image Underst. 72 (3) (1998) 404–413.
[44] S. Svensson, Reversible surface skeletons of 3D objects by iterative thinning of distance transforms, in: Proc. Discrete Geometry for Computer Imagery, Springer, 2001, pp. 400–411.
[45] C. Arcelli, G. Sanniti di Baja, L. Serino, Distance-driven skeletonization in voxel images, IEEE Trans. Pattern Anal. Mach. Intell. 33 (4) (2011) 709–720.
[46] P. Peter, M. Breu, Refined homotopic thinning algorithms and quality measures for skeletonisation methods, in: Innovations for Shape Analysis Mathematics and Visualization, Springer, 2013, pp. 77–92.
[47] A. Belyaev, S. Yoshizawa, H.P. Seidel, Skeleton-based variational mesh deformations, Comput. Graphics Forum 26 (3) (2007) 255–264.
[48] J. Chaussard, M. Couprie, H. Talbot, A discrete lambda-medial axis, in: Proc. Discrete Geometry for Computer Imagery, Springer, 2009, pp. 232–243.
[49] N. Gagvani, D. Silver, Parameter-controlled volume thinning, Graphical Models Image Process. 61 (3) (1999) 149–164.
[50] A. Tagliasacchi, I. Alhashim, M. Olson, H. Zhang, Mean curvature skeletons, Comput. Graphics Forum 31 (2012) 1735–1744.
[51] L. Liu, E. Chambers, D. Letscher, T. Ju, A simple and robust thinning algorithm on cell complexes, Comput. Graphics Forum 29 (7) (2010) 2253–2260.
[52] T. Ju, M. Baker, W. Chiu, Computing a family of skeletons of volumetric models for shape description, Comput. Aided Des. 39 (5) (2007) 352–360.
[53] G. Bertrand, A parallel thinning algorithm for medial surfaces, Pattern Recognit. Lett. 16 (9) (1995) 979–986.
[54] A. Telea, A. Jalba, Computing curve skeletons from medial surfaces of 3D shapes, in: Proc. TPCG, 2012, pp. 137–145.
[55] M. Schaap, C. Metz, T. van Walsum, A. van der Giessen, A. Weustinck, N. Mollet, et al., Standardized evaluation methodology and reference database for evaluating coronary artery centerline extraction algorithms, Med. Image Anal. 13 (5) (2009) 701–714.
[56] J. Mullikin, P. Verbeek, Surface area estimation of digitized planes, Bioimaging 1 (1993) 6–16.
[57] A. Torsello, E. Hancock, Correcting curvature-density effects in the Hamilton-Jacobi skeleton, IEEE Trans. Image Process. 15 (4) (2006) 877–891.
[58] F. Nooruddin, G. Turk, Simplification and repair of polygonal models using volumetric techniques, IEEE Trans. Visual Comput. Graphics 9 (2) (2003) 191–205. See also <www.cs.princeton.edu/min/binvox>.
[59] A. Sobiecki, A. Jalba, A. Telea, Skeletonization benchmarking resources, 2014. <www.cs.rug.nl/svcg/Shapes/SkelBenchmark>.

Please cite this article in press as: A. Sobiecki et al., Comparison of curve and surface skeletonization methods for voxel shapes, Pattern Recognition Lett. (2014), http://dx.doi.org/10.1016/j.patrec.2014.01.012