Volume xxx, (1997) number yyy pp. 000–000

Skeletal Reconstruction of Branching Shapes

Eric Ferley †, Marie-Paule Cani-Gascuel †, Dominique Attali ‡

† iMAGIS † - GRAVIR / IMAG
BP 53, F-38041 Grenoble cedex 09, France
‡ Infodis - TIMC / IMAG
IAB, Domaine de la Merci, 38706 La Tronche cedex, France
Eric.Ferley@imag.fr, Marie-Paule.Gascuel@imag.fr, Dominique.Attali@imag.fr

Abstract

We present a new method to reconstruct an implicit representation of a branching object from a set of data points scattered on its surface. The method is based on the computation of a geometric skeleton inside the data set. This skeleton is simplified in order to filter noise and converted into skeletal elements – a graph of interconnected curves – that generate an implicit surface. We use Bézier triangles as extra skeletal elements to perform bulge free blends between branches while controlling the blend extent. The result is a smooth reconstruction of the object, that can be computed whatever its topology. The skeleton offers compact storage, and provides an underlying structure for the reconstructed object, making it easier to edit in a modeling or animation environment.

1. Introduction

With the recent developement of advanced range-imaging sensors, automatic reconstruction of real-world objects from scattered data points has become an important issue in Computer Graphics. Applications include medical imaging, inverse engineering, simulation, and semi-automatic modeling for design or animation. For the first set of applications, the aim is to provide an improved and enhanced visualization of the data. In other cases, the reconstructed object is going to be edited, deformed and animated. Here, providing a structured reconstruction is essential.

In the area of shape recognition, researchers structure a data set by computing its *skeleton*, defined as the locus of the centers of maximal spheres inside the data [1, 2]. This skeleton, a thin centered structure that throughout this paper will be called *geometric skeleton*, provides a compact representation of both the topology and the geometry of the shape. Such a notion of a geometric skeleton appears attractive in the area of implicit surface modeling. The questions to be asked are: does this skeleton represent the same notion as the skeletal elements used to generate an implicit surface [3, 4, 5]? Can we use it to construct a smooth implicit representation of an object?

Compared with previous approaches for reconstruction using implicit surfaces, the introduction of a geometric skeleton to generate the surface gives important benefits:

- The implicit surface can be computed in a purely geometric way, without the need of optimization processes such as in [6, 7]. These processes are often computationally intensive and difficult to control.
- The skeleton, a graph of interconnected curve segments or surface elements, defines a structure for the reconstructed object. It can be used to edit the shape in an natural way. Such a structure is not provided when objects are reconstructed by directly computing an implicit function [8, 9], or by merely generating a list of skeletal points [6, 7].

In consequence, an approach based on skeletal reconstruction seems more adapted to modeling and animation applications.

† iMAGIS is a joint project of CNRS, INRIA, Institut National Polytechnique de Grenoble and Université Joseph Fourier.

This paper focuses on the reconstruction of free form branching shapes from the geometric skeleton of data points scattered on the object's surface. Conversion from geometric skeleton to parametric representation has already been widely studied [10]. The aim of this paper is different since we want to build a smooth implicit surface representing the shape. We call "branching shapes" these objects for which the geometric skeleton can be represented as a graph of interconnected curves, without any surface element. The topology of these objects is arbitrary

Section 2 reviews the definition of the geometric skeleton of an object and presents the semi-continuous approach that was used to compute it from a data set. Section 3 points out that this skeleton cannot directly be used to generate an implicit surface, and lists the problems to solve. Section 4 presents a method to convert each branch of the geometric skeleton into a skeletal curve that generates a smooth implicit surface of varying radius in the cross-section. Section 5 develops a new method, based on Bézier triangular skeletal elements, to smoothly blend branches at joints without generating bulges or creases. Results are presented in Section 6. We conclude and focus on future work in Section 7.

2. Geometric Skeleton

2.1. Definitions

The **geometric skeleton** of an object is the locus of the centers of maximal spheres inside this object. A sphere included in an object is said to be **maximal** if it is not included in any other sphere included in the object.

The geometric skeleton is a thin structure, represented as a graph of interconnected curve segments or surface elements, centered in the object (see Figure 1). It stores in a compact way both the topological properties of the object – given by its graph structure – and the geometrical information, i.e. the distance to the surface.

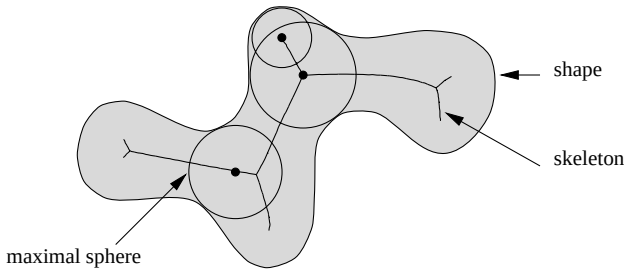


Figure 1: *The geometric skeleton of an object.*

Note that the geometric skeleton is based on the same principles as the "medial axis" used in discrete geometry [11]. However, the medial axis of the data set is built as a volume made of an unstructured set of voxels. In contrast, the graph structure of the geometric skeleton gives useful neighboring information.

A large amount of work has been done in image analysis on the computation of the skeleton [1, 2, 12]. Skeletonization methods are quite different depending on the way objects are represented. If the object is defined as a collection of voxels (or pixels in 2D), then the skeleton can be computed either by simulating the propagation of a wire initialized at the boundary of the object, by topological thinning, or by first extracting the medial axis from a distance map. In this paper, as the object is only known through a set of unorganized points $\{p_i\}_{i=1}^{n}$ located on its boundary, a different approach must be considered.

2.2. Computation of the geometric skeleton

Our approach exploits the relationship between the skeleton of an object and the Voronoi graph of its boundary points. The Voronoi graph of a set of points (called *seeds*) divides the space into regions [13]. Each region is the set of points closer to a particular seed than to any other seed. The skeleton may be approximated as the subset of the Voronoi elements (the edges of the Voronoi diagramm) that are completely included in the object [12] (see Figure 2). The better the sample of the object, the better the approximation of the skeleton.

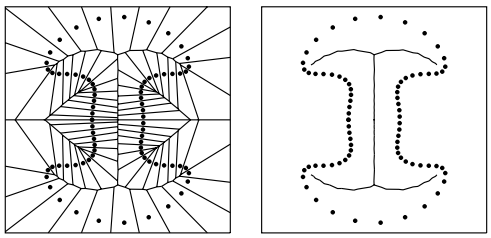


Figure 2: *Relation between the Voronoi graph (a) and the geometric skeleton (b) of a set of point.*

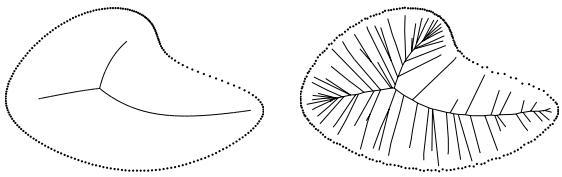
The computation of the skeleton requires to determine which part of the Voronoi graph is located inside the object. As the initial data is a mere set of boundary points with no structure, there is no immediate way to decide whether a point is inside or outside the object. Therefore, an heuristic must be used. We first compute the Delaunay tetrahedrization of the boundary points p_i, which is the dual of the Voronoi graph. Then, Delaunay tetrahedra are progressively merged until space is partitioned into two sets that represent

the interior and the exterior of the object [14] (see Figure 4(a)). After this step, the skeleton can easily be retrieved from the tesselation of the object.

The result is a thin shape made up of straight-line segments in $2D$ and polygons in $3D$ (Figures 3(a) and 4(b)). Each Voronoi vertex located on the geometric skeleton is the center of a sphere passing through at least three seeds and having no seed in its iterior. The radius of this sphere approximates the distance to the surface at each vertex.

2.3. Pruning branches due to noise

A drawback of the skeleton transformation is its sensibility to noise. Noise on the boundary of an object may significantly change its skeleton (Figure 3(b) and 4(b)). A simplification algorithm is therefore needed to remove peripheral elements – branches in $2D$, polygons in $3D$ – having no perceptual relevance.

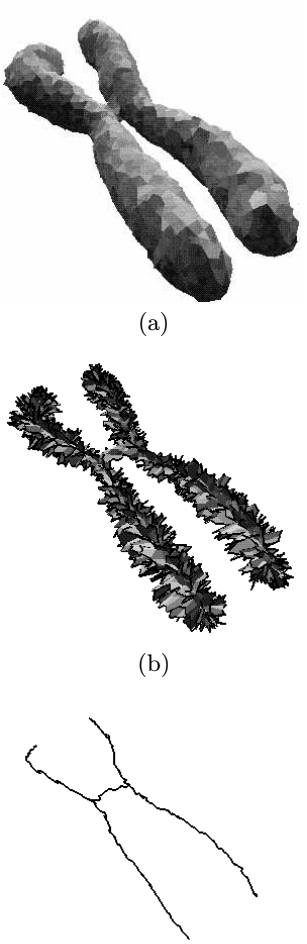


(a) (b)

Figure 3: *Sensitivity of the geometric skeleton to noise in the data.*

The skeleton is simplified by iteratively removing those Voronoi vertices that are located on the border of the skeleton and verify a *removing criterion*. By construction, the simplified skeleton is a subset of the initial skeleton. Moreover, the simplification method does not change the homotopy type of the skeleton, and thus the topology of the object remains unchanged.

In this paper, we apply a removing criterion based on the dual relationship between the object and its skeleton [15] [16]. Since the center and the radius of each maximal sphere is known, the skeleton stores the information on the boundary of the object. Hence, the amount of change in this boundary gives a criterion to guide the simplification process: we remove a Voronoi vertex from the skeleton if the volume of its dual Delaunay simplex is less than a given threshold. This prunes branches due to noise, and in the $3D$ case, can lead to wireframe skeletons, which are "natural skeletons" for branching shapes – made up of generalized cylinders (see Figure 4(c)). By construction, the order of all branching points is three.



(a) (b) (c)

Figure 4: *The data set of a chromosome, its geometric skeleton (including surface elements), and the wireframe skeleton obtained at the end of the simplification process.*

3. Problems Raised by Skeletal Implicit Modelling

People working in the area of implicit modeling use a set of geometric primitives – a skeleton – associated with distance varying field functions to generate a surface. Ideally, a smooth surface representing an object should be directly computed from any skeletal description. Unfortunately, using the geometric skeleton described in Section 2 for iso-surface generation is not so trivial, since this skeleton is a graph of very small line segments.

This section first reviews methods used in skeletal implicit modeling, and then points out the difficulties

raised by the use of the geometric skeleton computed in Section 2.

3.1. Implicit surfaces generated by skeletons

Implicit surfaces have been defined as iso-surfaces of field functions (Sometimes called "implicit function" although they are defined by an explicit equation) generated by a set of geometric primitives that define the "skeleton" of the object [3, 4].

The field contribution f_i of each skeletal primitive S_i is a decreasing function of the distance to this element. The implicit surface $\mathcal{S}$ is defined as an iso-surface of isovalue c for the sum of skeletal contributions:

$$\mathcal{S} = \left\{ P \mid \sum_i f_i(P) - c = 0 \right\} \quad (1)$$

This formulation is a generalization of the original Blinn's objects [17], metaballs [18] and soft objects [19], where only skeletal points were used.

If the summation is replaced by a maximum in equation (1), the implicit volume is the union of the volumes created by each skeletal primitive (see Figure 5(c)). The advantage of summation is that it defines a smooth blend between these contributions.

However, the way each f_i varies affects the blend so that generating a smooth shape is not an easy task. In particular, bulges (see Figure 5(b)), characterized by a *"cross section that exhibits negative, then positive, then negative curvature with respect to the underlying skeleton"* [5], should be avoided.

The next paragraphs review the different models that have been proposed as field functions.

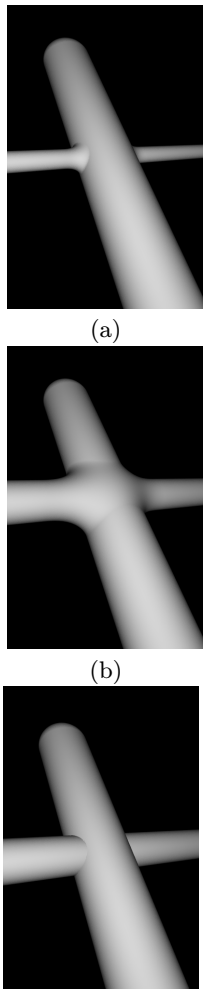
Distance surfaces

Distance surfaces [3] define f_i as a decreasing C^1 continuous function F of the distance to S_i (defined as the distance to the closest point on S_i):

$$f_i(P) = F(d(P, S_i)) \quad (2)$$

The choice of a function F with local support limits the influence of each skeletal component, thus providing local control of the shape and reducing computations. Anisotropic field functions depending on the location of the closest point on S_i may be used to generate more complex shapes [20].

Distance surfaces can be safely used only with convex skeletons, otherwise creases may appear. Dividing non-convex skeletons into convex parts does not give any solution, since bulges at joints would be produced in this case. Consequently, modeling a ramification without any bulge or crease appears difficult (see Figure 5).



(a)

(b)

(c)

Figure 5: *Distance surface blend for a skeleton made of two segments:*
(a) The blend is smooth when the thicknesses are very different.
(b) A bulge appears when the branches have about the same thickness.
(c) Union surface produced by replacing sum by maximum in equation (1).

Convolution surfaces

Convolution surfaces [4] define f_i as the integral over S_i of contributions from individual points. Here, the resulting surface does not depend on the way a skeleton

is divided into skeletal elements, which is more satisfactory. Moreover, the resulting surface is smooth, even for a non-convex skeleton.

However, J. Bloomenthal showed that the convolution method generates bulges at a joint of one dimensionnal skeletal pieces [5]: one must thicken the skeletal pieces and handle $2D$ (or $3D$ for circular cross-sections) skeletons to prevent that phenomenon.

Procedural methods

Procedural field functions were introduced as an example of complex implicit definition [3], in the specific case of a skeleton made up of two branching curves S_1 and S_2. In order to compute the field contribution at a given point P, the points P_1 and P_2 closest to P on each of the two skeletal curves S_i are computed, and then $f_i(P)$ is defined by the distance to the closest point on the segment $[P_1, P_2]$:

$$f_i(P) = F(d(P, [P_1, P_2]))$$

Unfortunately, this approach that seems to offer an elegant solution to the bulge problem cannot be applied in the general case:

- It does not work for sharp angles between branches, as shown in Figure 6, since the surface tangent changes suddenly when a projected point P_1 or P_2 is located on the extremity of the line segment.
- There is no control of the blend extent: it is restricted by the projective domains of points P on S_i. In consequence, two consecutive ramifications can not be modeled with this method.

3.2. Problems to be solved

Applying one of the previous skeletal modeling techniques to reconstruct a smooth shape from the geometric skeleton of a data set is not straightforward:

Each branch of the geometric skeleton computed in Section 2 is made up of many segments. Due to noise in the data, the resulting polyline is not smooth. The distance surface generated by this polyline would exhibit creases, while the summation of each segment's contribution would generate bulges. As we want to exploit the simplicity of the $1D$ skeletal parts computed, the use of convolution surfaces does not fit our problem. None of the previous methods of field computation can directly be applied to the skeleton we obtained. We have to find a way to smoothly blend consecutives line-segments and smooth junctions between several branching curves.

The rest of this paper presents our solutions to these problems:

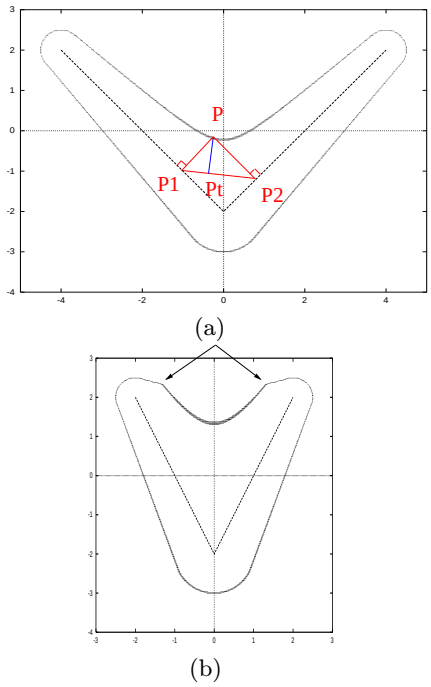


Figure 6: *Effects of the procedural method to blend the contributions of two line segments:*
(a) The method works when the angle between the segments is greater than $\frac{\pi}{2}$.
(b) Strange features appear for sharp angles.

In Section 4, we approximate each polyline by a spline curve, and define a field function of varying strength along it in order to generate an implicit generalized cylinder of varying radius.

Our solution to the branching problem is described in Section 5. It was inspired by the procedural method reviewed above: in this method, the locus of points P_t (see Figure 6(a)) can be seen as an extra skeletal element – a curve in $2D$, a surface in $3D$ – that is used to compute a distance surface. The procedural approach did not offer any chance of controlling the shape of this element nor the smoothness of its junction with the skeletal curves, which resulted in the problems listed above. Our basic idea is thus to define such an element explicitly. We use a triangular patch whose corners fit with the branching curves. The non-manifold skeleton made of the union of this patch and of the curves generates a junction without any bulge or crease.

4. Implicit Reconstruction along a Branch

The method for computing the geometric skeleton described in Section 2 outputs a graph of connected line segments, the distance to the surface being stored at

each vertex (this distance is the radius of the maximal sphere associated with the vertex).

The first step for converting the geometric skeleton into a set of skeletal elements that can be used in iso-surface generation is to split the graph into a set of polylines (or "branches"), connected by joints. Then, we simplify each polyline and define the field function that generates an implicit generalized cylinder around it.

4.1. Smoothing a polyline

The algorithm of Section 2 computes a skeleton whose number of vertices is of the same order as the number of initial data points. Consequently, each branch is an irregular polyline made up of a large number of very small segments (see Figure 4).

As the easiest way to generate an implicit generalized cylinder is to use the distance surfaces along a smooth curve, we approximate the polyline by a spline curve. This is done in two steps:

- We first reduce the number of vertices along the polyline using Pavlidis algorithm [21]: the polyline is first approximated by the line segment between its extremities, and then this coarse representation is progressively refined by adding some of the inner initial vertices. The algorithm to refine a segment is:

  1. Compute the distance between the segment and each of the initial vertices along this branch;
  2. Stop the refinement if the maximal distance is smaller than the level of detail threshold;
  3. Otherwise, split the segment into two parts by adding the inner vertex which is the farther from the line segment.

  As we need to take into account the variations of the radius of the cylinder as well as the geometry of its axis, we use the above algorithm in a $4D$ space, where the fourth coordinate of a vertex is the distance to the surface (i.e. the radius of the associated maximal sphere).
- Then, we use the few remaining vertices as control points for a spline curve that approximates the initial polyline (we currently use cubic B-Spline curves).

An important remark is that the distance surface generated by a curve is smooth if and only if the radius of the cross section at each point of the curve is smaller than the local radius of curvature (see Figure 7). Otherwise, a crease will be generated. In our current implementation, we apply more smoothing to the polyline when the crease problem appears (our hypothesis is that the shape to reconstruct was smooth).

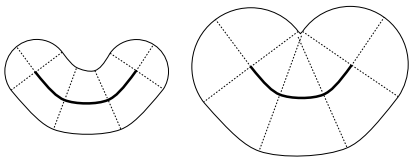


Figure 7: *Constraint on the curvature of a skeletal curve: it should remain smaller than the distance to the surface. Otherwise, a crease will appear.*

In future work, we plan to automatically compute the level of detail threshold to be used in step 2 of the above algorithm from the local constraint of curvature.

4.2. Field contribution along a curve

The skeleton gives precise information on the geometry of the object, since the distance to the surface is stored for each vertex. The reconstructed implicit surface should take this information into account. In

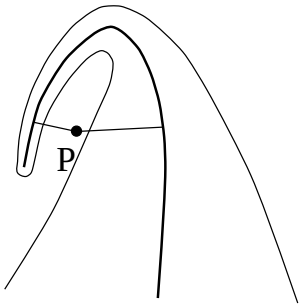


Figure 8: *Implicit contribution along a curve.*

particular, using distance to the closest point on the curve to define the implicit contribution at point P, as in equation (2), no longer works (see Figure 8). Euclidean distance must be replaced by a generalized distance that takes the radius along the curve into account.

Computing the field contribution generated at a given point P by a cubic curve $S(u)$ can be done as follows:

1. Compute the set of curve parameters $(u_i)_{i=1..p}$ that define local minima of the distance to the point P, by finding the roots of a 5th degree univariate polynomial [22, 23].
2. For $i = 1..p$, Compute the object thickness $r(u_i)$ from the distances to the surface at the curve's control points (we use again a cubic B-spline approximation to interpolate the radii associated with the vertex).

3. Select the parameter u_k, for $k \in \{1..p\}$, that minimizes the generalized distance:

$$D(P, S(u)) = (d(P, S(u)) - r(u))$$

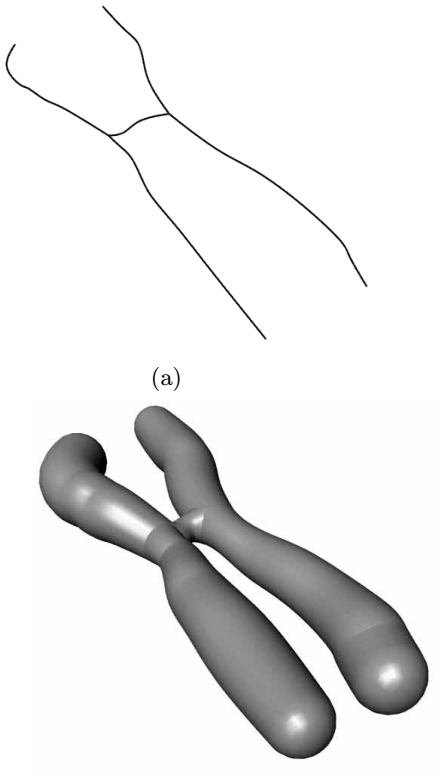
4. Then, equation 2 is replaced by:

$$f_i(P) = F(D(P, S(u_k))) \quad (3)$$

where F is a C^1 continuous function of local support satisfying $F(0) = c$.

This definition of the field function could be used for directly ray-tracing the implicit surface [17]. In practice, the images illustrating this paper have rather been rendered from a polygonalisation of the implicit surface, computed from the parametrization of the skeletal curves [10].

Figure 9 shows the conversion of the geometric skeleton of the chromosome into a graph of B-spline curves, and the surface – with creases – obtained as the union of the generalized cylinders defined along each branch.



(a)

(b)

Figure 9: *Implicit reconstruction along branches for the chromosome example:*
(a) Conversion of the polylines into cubic B-spline curves of 51 control points.
(b) Union surface composed of implicit surfaces of varying radius generated along each curve.

5. Bulge Free Blend Between Implicit Branches

The last step of the implicit reconstruction algorithm is to generate a smooth blend between the field contributions of connected skeletal curves.

As noticed in Section 3.1, using a mere summation of field contributions gives satisfactory results when the thicknesses of connected branches are very different (see Figure 5). In that case, the blend extent can be controlled by carefully choosing the support of the function F used for each branch. However, if the branches have approximately the same size, a bulge appears near the junction. Since none of the previous approach solves this problem (see Section 3), we propose a new technique, specifically designed to model a joint between three branches (as explained in Section 2, only joints of order three are generated during the computation of the geometric skeleton). Possible extensions to higher order joints are discussed below.

5.1. Blending three branches

Our solution consist in smoothing the joint by locally replacing the three curves by a small surface patch. Then, the union of the curves and this patch, which is a smooth non-manifold entity, will be considered as a single skeletal element generating a distance surface.

In the case of three skeletal curves, we use a cubic Bézier triangle [24] to smooth the joint. This triangle, defined by 10 control points, is represented by the parametric equation:

$$\begin{aligned} S(u,v,w) &= u^3P_0 + v^3P_1 + w^3P_2 + 6uvwP_9 \\ &+ 3u^2vP_3 + 3uv^2P_4 + 3v^2wP_5 \\ &+ 3vw^2P_6 + 3uw^2P_7 + 3u^2wP_8 \end{aligned}$$

Points P_0, P_1 and P_2 are interpolated, while points P_3 to P_8 define the tangent directions (see Figure 10).

To generate a bulge/crease-free blend between three curves, we use the Bézier triangle as follows: we put P_0, P_1 and P_2 on the curves to blend. Their position along the curve control the blend extent. We choose them according to the thickness of the solid at the joint. The same property as in Section 4.1 must be satisfied: at each point on the Bézier triangle the local radius of curvature must be larger than the surface radius, otherwise no smooth surface can be defined.

Since the three corners of the Bézier triangle must

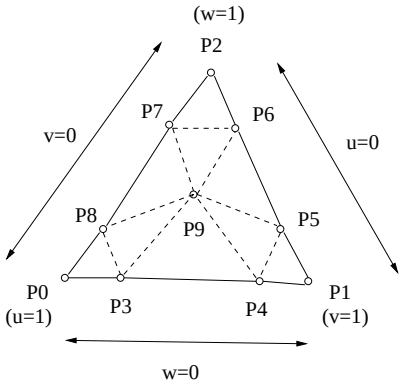


Figure 10: *Definition of a Bézier triangular patch: P_0, P_1 and P_2 are the primary control points; P_3, P_4, P_5, P_6, P_7 and P_8 are the secondary control points.*

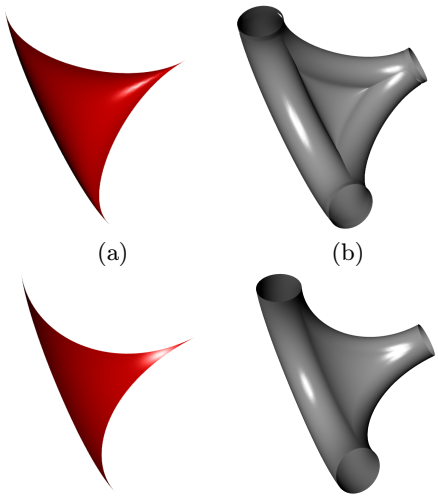


Figure 11: *Positioning of a Bézier triangle.*
(a) and (b) with secondary control points merged.
(c) and (d) with compostion of two Bézier triangles.

be sharp to fit the curves, our first attempt consisted in merging pairs of secondary control points. These points were positioned along the tangent direction to the curve at the nearest primary control point, in order to satisfy the skeleton G^1 continuity (same tangent direction). However, when the three tangents to the curves are not coplanar, the triangle radius of curvature reaches zero near the primary control points (see Figure 11(a)); so the curvature condition (similar to Section 4.1) can never be satisfied, which causes the creases in Figure 11(b).

To avoid this problem, we use a composition of two

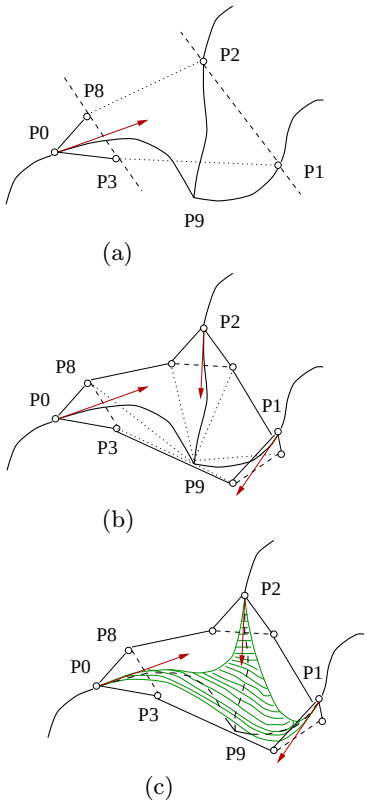


Figure 12: *Definition of an extra-skeletal element.*

Bézier triangles, one in the $2D$ parametric domain of the other one (which is in $3D$):

- As before, points P_0, P_1 and P_2 of the 3D triangle are positioned on the curves to blend. Then, pairs of adjacent secondary control points are set in the plane that passes through the closest primary control point and includes both the tangent to the curve at that point and the vector formed by the two other primary control points (see Figure 12(a) and (b)). Since the corners of this triangle tend to be located in a plane, there is no more problem with high curvature.
- Then, we restrict the parametric domain of this patch to ensure the G^1-continuity with the curves: we use a $2D$ Bézier triangle whose pairs of secondary control points are merged to map the parametric domain of the patch into a smaller domain having sharp angles at corners (see Figure 12(c)).

The resulting offset surface presents no more creases, as depicted on Figures 11(c) and 11(d).

Implicit contribution along the Bézier triangle

When the surface has a varying radius along the three skeletal curves, the distance to the implicit surface

must vary smoothly along the Bézier triangle $\mathcal{T}$. We use the same approach as in Section 4.2: the equation of the Bézier triangle is used again to interpolate the distances to the surface at control points, in order to compute the desired radius $r(u, v, w)$ at the point of barycentric coordinates u, v, w in $\mathcal{T}$. Then, the field contribution along the surface can be computed in the same way than along a curve (Section 4.2), using the generalized distance:

$$D(P, S(u, v, w)) = (d(P, S(u, v, w)) - r(u, v, w))$$

in the field computation.

Again, our implementation is based on a parametric method to display the corresponding *offset* surface (inspired from [10]).

The G^1 continuity of the resulting surface is ensured by:

1. the C^1 variation of both the skeletal curves and triangles, and the offset radius along these elements.
2. the G^1 connection between these elements. (this holds also for the radius, considered as the fourth dimension of the skeleton)
3. the observance of the curvature condition.

Figure 13 shows the results for 3D curves along which the thickness is varying.

5.2. Generalizations

Although not yet implemented, extensions of this blending approach to more general branchings seems promising.

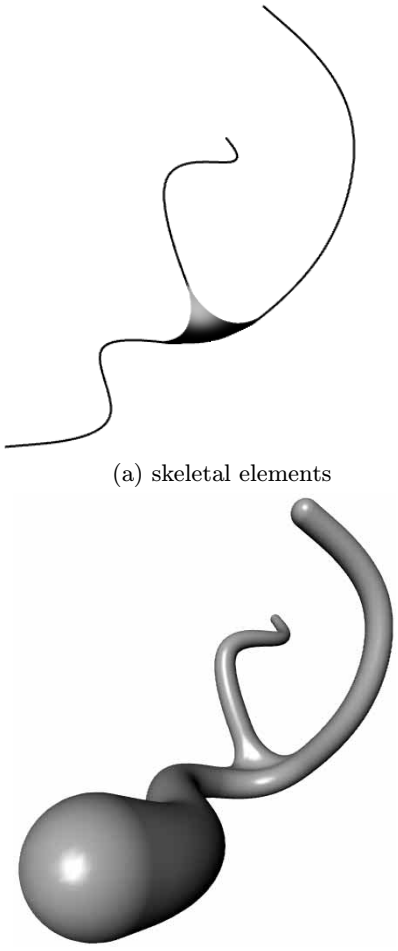
In the case of a flat branching between n curves, the method would first project the curves to a plane that is "tangent" to the joint, in order to define an order between the curves. Then, an n-sided Bézier patch [25] could be used instead of a Bézier triangle to thicken the junction.

In the general branching case, the n branches may go in any direction in space, so there is no way of flattening the junction. Then, the generalization of our method would mean embedding the joint in a Bézier volume interpolating each of the n branches (see Figure 14).

The faces of this Bézier volume would be Bézier triangles joining three of the n curves, so computing the closest point of this volume would involve closest point computations for Bézier triangles only.

6. Results

Figure 15 shows the final reconstruction for the chromosome example of Figure 4. Figure 16 illustrates the different steps of the reconstruction of a more complex object.



(a) skeletal elements

(b) resulting implicit surface

Figure 13: *Bulge free blend between three skeletal curves*

7. Conclusion and Future Work

This paper has developed a new approach to the reconstruction of 3D free-form surfaces, based on the combination of geometric skeletons and implicit surfaces. The main features of the method are:

- a purely geometric way to perform the reconstruction, without any optimization process to define the skeletal elements that generate the surface.
- a structured reconstruction: the skeleton that generates the implicit surface is a graph of interconnected curves, so it gives an intuitive and compact representation of the object. Editing this skeleton to model deformations or articulations also appears

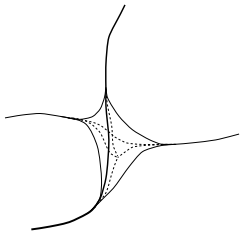
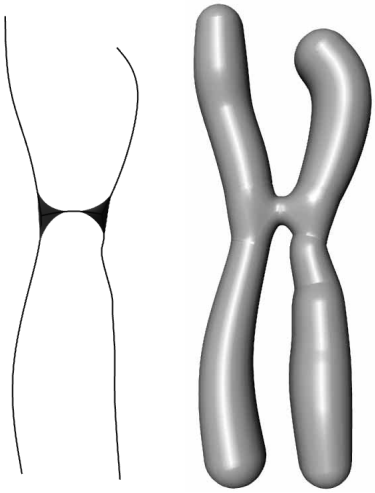


Figure 14: *Embedding of a joint in a Bézier volume*



(a) (b)

Figure 15: *Smooth implicit reconstruction of the chromosome.*

feasible. Therefore, the method seems promising for further applications in modeling and animation.

Perfecting the reconstruction method has led us to solve a problem inherent in skeletal implicit design: the bulge free blending of surfaces generated by skeletal curves. Instead of trying to define more complex field contributions, as was done with convolution surfaces or with procedural field functions, our solution is based on the local thickening of the skeleton near a joint to prevent the creases on the corresponding distance surface. This approach offers the advantage of an explicit control on the blend extent near the joint. In a modeling system, the surface or volume element used to thicken the skeleton could be computed in an automatic way, thus being transparent to the user.

Future work includes the implementation of a smooth blend between any number of skeletal curves, using Bézier volumes. We are also looking for bounds on the curvature of Bézier triangles in order to fix their size automatically according to the local thickness of the object at a joint.

This paper has studied the reconstruction of "branching shapes" for which the skeleton was a graph of interconnected curve segments. In the general case the geometric skeleton of an object includes both curves and surface elements. We are planning to use either interconnected surface patches or curve segments with anisotropic implicit contributions to model these surface elements.

Acknowledgements

Our understanding of the bulge free blend problem was really deepened by discussions with Mathieu Desbrun, Agata Opalach and Jules Bloomenthal. Many thanks to Mathieu for his help during the implementation and to Agata for carefully rereading this paper.

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.), (Cambridge, MA), pp. 362–380, M.I.T. Press, (1967).
2. C. Arcelli and G. Sanniti di Baja, "Euclidean skeleton via centre-of-maximal-disc extraction", *Image and Vision Computing*, **11**(3), pp. 163–173 (1993).
3. J. Bloomenthal and B. Wyvill, "Interactive techniques for implicit modeling", *Computer Graphics*, **24**(2), pp. 109–116 (1990). Proceedings of Symposium on Interactive 3D Graphics.
4. J. Bloomenthal and K. Shoemake, "Convolution surfaces.", *Computer Graphics*, **25**(4), pp. 251–256 (1991). Proceedings of SIGGRAPH'91 (Las Vegas, Nevada, July 1991).
5. J. Bloomenthal, "Bulge elimination in convolution surfaces", *Computer Graphics Forum*, **16**(1), pp. 31–41 (1997). An early version of this paper appeared in *Implicit Surfaces'95*, Grenoble, France, apr 1995.
6. S. Muraki, "Volumetric shape description of range data using blobby model", *Computer Graphics*, **25**(4), pp. 227–235 (1991).
7. E. Bittar, N. Tsingos, and M. Gascuel, "Automatic reconstruction of unstructured 3d data: Combining medial axis and implicit surfaces", in *Eurographics'95*, (Sept. 1995).
8. R. Whitaker, "Algorithms for implicit deformable models", in *The International Conference of Computer Vision*, (Boston, Mass), (1995).
9. L. Velho and J. Gomez, "Approximate conversion of parametric to implicit surfaces", *Computer Graphics Forum*, **15**(5), pp. 327–337 (1996). A preliminary version of this paper appeared in *Implicit Surfaces'95*, Grenoble, France, may 1995.
10. S.-M. Gelston and D. Dutta, "Boundary surface recovery from skeleton curves and surfaces", *Computer Aided Geometric Design*, **12**, pp. 27–51 (1995).

11. L. Lam, S.-W. Lee, and C. Y. Suen, "Thinning methodologies - a comprehensive survey", *IEEEP-AMI*, **14**(9), pp. 869–885 (1992).

12. J. W. Brandt and V. R. Algazi, "Continuous skeleton computation by Voronoi diagram", *CVGIP: Image Understanding*, **55**(3), pp. 329–337 (1992).

13. F. Aurenhammer, "Voronoi diagrams - a survey of a fundamental geometric data structure", *ACM Computing Surveys*, **33**(3), pp. 345–405 (1991).

14. D. Attali, *Squelettes et graphes de Voronoi 2D et 3D*. PhD thesis, Université Joseph Fourier- Grenoble I, (Oct. 1995).

15. D. Attali, G. Sanniti di Baja, and E. Thiel, "Pruning discrete and semicontinuous skeletons", in *Lecture Notes in Computer Science, Image Analysis and Processing* (L. D. F. C. Braccini and G. Vernazza, eds.), vol. 974, pp. 488–493, Springer-Verlag, (1995). Proceedings of the 8th ICIAP.

16. D. Attali and A. Montanvert, "Semicontinuous skeletons of 2D and 3D shapes", in *Aspects of Visual Form Processing* (C. Arcelli *et al.*, eds.), pp. 32–41, World Scientific, Singapore, (1994).

17. J. Blinn, "A generalization of algebraic surface drawing", *ACM Transactions on Graphics*, pp. 235–256 (1982).

18. H. Nishimura, M. Hirai, T. Kawai, T. Kawata, I. Shirakawa, and K. Omura, "Objects modeling by distribution function and a method of image generation (in japanese)", *The Transactions of the Institute of Electronics and Communication Engineers of Japan*, **J68-D**(4), pp. 718–725 (1985).

19. G. Wyvill, C. McPheeters, and B. Wyvill, "Data structure for soft objects", *The Visual Computer*, **2**(4), pp. 227–234 (1986).

20. Z. Kacic-Alesic and B. Wyvill, "Controlled blending of procedural implicit surfaces", in *Graphics Interface'91*, (Calgary, Canada), pp. 236–245, (June 1991).

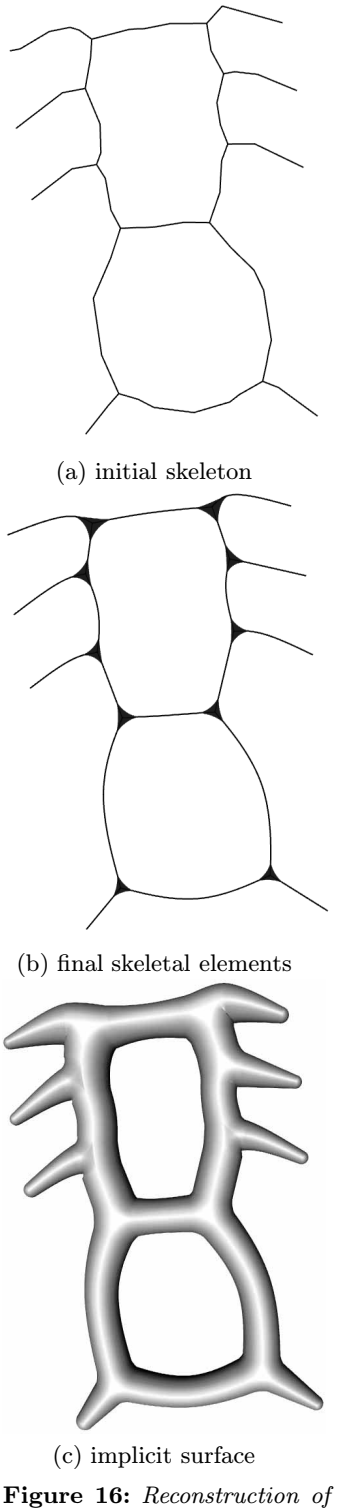
21. T. Pavlidis and S. L. Horowitz, "Segmentation of plane curves", *IEEE trans. in Computers*, **23**(8), pp. 860–870 (1974).

22. P. Schneider, *Solving the nearest-point-on-curve problem*. New York: Andrew Glassner, Academic Press, (1990).

23. J. Bloomenthal, "Skeletal design of natural forms", *PHD Thesis*, **The University of Calgary**, (1995).

24. G. Farin, *Curves and Surfaces for Computer Aided Geometric Design*. San Diego, California: Academic Press, (1988).

25. C. Loop and T. DeRose, "A multisided generalization of bézier surfaces", *ACM Transactions on Graphics*, **8**(3), pp. 204–234 (1989).



(a) initial skeleton

(b) final skeletal elements

(c) implicit surface

Figure 16: *Reconstruction of a free form branching shape.*