Three dimensional reconstruction of complex shapes based on the Delaunay triangulation

Jean-Daniel Boissonnat, Bernhard Geiger

UNITÉ DE RECHERCHE
INRIA-SOPHIA ANTIPOLIS

Rapports de Recherche

N°1697

Programme 4
Robotique, Image et Vision

THREE DIMENSIONAL RECONSTRUCTION OF COMPLEX SHAPES BASED ON THE DELAUNAY TRIANGULATION

Jean-Daniel BOISSONNAT
Bernhard GEIGER

April 1992

Institut National
de Recherche
en Informatique
et en Automatique

Sophia Antipolis
B.P. 109
06561 Valbonne Cedex
France
Tél : 93 65 77 77

Three Dimensional Reconstruction Of Complex Shapes Based On The Delaunay Triangulation

La Triangulation de Delaunay appliquée à la Reconstruction 3D d'Objets Complexes

Jean-Daniel Boissonnat[†]
Bernhard Geiger[†]

April, 17 1992

Abstract

We propose a solution to the problem of 3D reconstruction from cross sections, based on the Delaunay triangulation of object contours. Its properties—especially the close relationship to the medial axis—enable us to do a compact tetrahedrisation resulting in a nearest-neighbor connection. The reconstruction of complex shapes is improved by adding vertices on and inside contours.

Résumé

Nous présentons ici une solution au problème de la reconstruction tridimensionnelle d'objets à partir de coupes parallèles, qui utilise la triangulation de Delaunay des contours des objets. Nous construisons une tétraédrisation de l'objet reliant les contours à leurs plus proches voisins, grâce aux propriétés de la triangulation de Delaunay, en particulier sa relation étroite avec l'axe médian. Dans le cas d'objets complexes (branches, trous), nous améliorons la reconstruction en ajoutant des sommets sur et à l'intérieur des contours.

Keywords: 3D reconstruction, Delaunay triangulation, Voronoi diagrams, medial axis.

[†]Institut National de Recherche en Informatique et Automatique
BP 93 — 06902 Sophia Antipolis Cedex (France)
Phone : +33 93 65 77 60— Telex : 970 050F — Fax : +33 93 65 76 43
E-mail : boissonn@sophia.inria.fr geiger@sophia.inria.fr

Contents

1 Introduction

An important problem in medical imaging is the creation of three dimensional models from a set of tomographic cross sections. Such reconstructions of human organs are already used for

- surgery planning
- prothesis milling
- radiation therapy planning
- volumetric measurements

According to the high interest, there have been a great number of approaches, which may roughly be classified into two groups: surface reconstruction and volume reconstruction. The first group contains the classical solutions of [Kep75] and [FKU77]. In the second we find the today's dominant voxel technique.

Another volume approach is the Delaunay reconstruction proposed in [Boi88]. In this article, we consider the Delaunay reconstruction from a different point of view that

allows us to compare its heuristic to that of the voxel technique. We show its close relationship to the medial axis and finally add some new extensions.

In the following section, we will give a brief description of the surface- and volume-approach and discuss some theoretical questions. In section 3 we explain the principal terms Voronoi diagram and Delaunay triangulation and give a range of successive refinements, from Voronoi diagram of point sites to the medial axis. Finally, we describe the Delaunay reconstruction in section 4.

2 Previous Solutions

2.1 Surface reconstruction

Historically, surface reconstruction has been the first approach. Running along the contours, surface triangles are constructed between adjacent planes. Various conditions may be imposed: maximize the volume, minimize the surface, minimize the edge length or angles (See Keppel [Kep75] and Fuchs et al. [FKU77]). These solutions were limited to one single contour on each cross section. In the suite, there were a number of methods trying to solve the branching problem when the number of contours varies from one plane to another ([CS78] [Sha81] [MSS91] [Toe89]).

2.2 Voxel reconstruction

A voxel is the spatial equivalent to a pixel. Since scanner data comes on a regular grid, the pixels, it is quite natural to extend them to a volume element (Barillot [BGLS85]). However, to avoid excessive radiation, cross section distances often are significantly greater than the 2D resolution. This makes interpolation necessary in order to avoid jaggies. To visualize such a heap of cubes, it is necessary to fit a surface on it (Marching Cube [LC87]) or to deduce surface normals with other methods [HB86], [Lev88]). The main disadvantage however is the huge amount of data, which has to be processed. Still, real time motion requires special hardware. For an exhaustive survey on reconstruction methods and systems see also [SFF91].

2.3 The reconstruction problem

A 3D object is intersected by a number of parallel planes. Since concave objects are admitted, we get one or several regions in each plane. The contours of those regions may be approximated by polygons, possibly some of them lying inside others.

We now want to link these regions (or to connect the contours) in a way to obtain the "best" approximation to our original object.

One condition the reconstructed object has to satisfy is: If we cut it along the original planes, we must get the actual regions.

If we consider figure 1, we have to admit that both solutions b and c satisfy the condition specified above. Selecting a particular solution is a heuristic decision, if no a priori information is present.

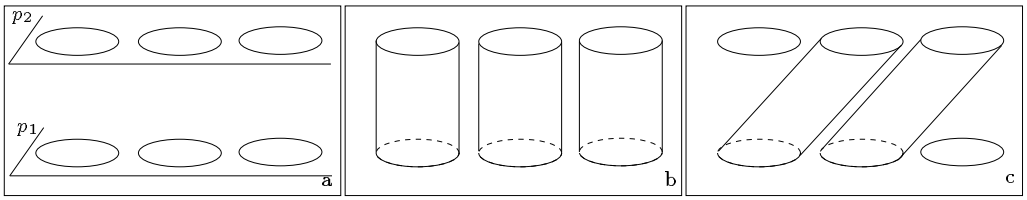

Figure 1: A set of 6 contours, pairwise superposed in two horizontal, parallel planes p_1 and p_2. (b) and (c) show two possible connections

All reconstruction methods make such heuristic assumptions, except the interactive ones. In voxel reconstruction, the heuristic consists in extending all object pixel to a parallelepiped of height equal to the slice distance, thus favorizing "vertical" connections. It seems to be quite a simple, natural assumption, and at an adequate relation of object size and slice distance, the results are fully satisfying.

In our approach, we subdivide the planar regions into triangles, and extend these triangles to tetrahedra by linking them to the contours in adjacent cross sections. Our attempt hereby is to achieve an almost vertical, "voxel-like" connection (heuristic), but in the same time to reduce the data, avoid anti-alias or interpolation steps and get directly a polyhedral representation which is suitable for rendering on standard graphics workstations.

3 Definitions

3.1 Voronoi Diagram - Delaunay Triangulation

Given a set S of N sites $S = \{p_i \in E^2 | i = (1..N)\}$ such that no four sites lie on a common circle.

We define $V(i)$ as the set of points closer to site p_i than to any other site in S. $V(i)$ is called the **Voronoi cell** associated with p_i (See fig. 2). The union over all $V(i)$, $V(S)$, is called the **Voronoi diagram** of S. It can be considered as a subdivision of the plane in convex polygonal regions—the Voronoi cells. The boundaries of the regions are referred to as **Voronoi edges**, the joints of (always) three Voronoi edges are called **Voronoi vertices**. Each Voronoi edge is associated with two adjacent Voronoi cells, and is a part of the perpendicular bisector of their sites, thus denoting the locus of equal distance. A Voronoi vertex is equidistant to three sites.

If we draw a line segment between each pair of sites whose Voronoi cells share an edge, we obtain a triangulation of the points in S called the Delaunay triangulation (See fig. 3).

The Delaunay triangulation is the straight-line-dual of the Voronoi diagram :

- each Voronoi vertex is the circumcenter of a Delaunay triangle.

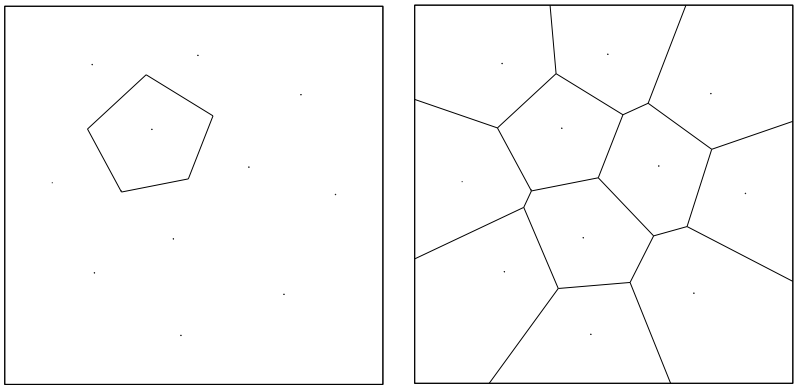

Figure 2: A Voronoi cell (left) and the Voronoi diagram (right) of a number of sites

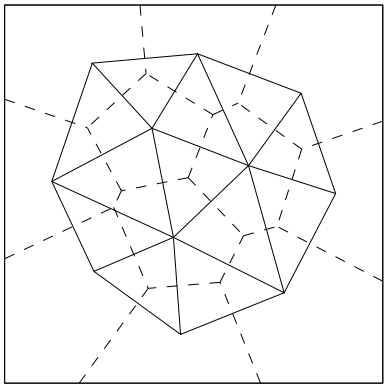

Figure 3: A Voronoi diagram (dashed) and its straight-line-dual, the Delaunay triangulation .

- each Voronoi edge corresponds to an edge in the Delaunay triangulation , despite the fact that they may not even intersect. This geometric difference between Voronoi diagram and Delaunay triangulation becomes important in the reconstruction issue.

Some further properties of the Delaunay triangulation are :

- The border of the Delaunay triangulation is the convex hull of its sites.
- The Delaunay triangulation maximizes the minimum angle over all triangulations.
- The number of triangles in the Delaunay triangulation is at most $2N - 5$, where N is the number of vertices in the triangulation.
- A Delaunay triangle does not contain any other site in its circumcircle (empty circle property). This implies that any triangulation without obtuse angles must be Delaunay.

The definition of Voronoi diagrams and Delaunay triangulation in 3D space is straightforward (Refer to [PS85]). Instead of triangles we will get tetrahedra, besides Voronoi edges there will also be Voronoi faces, and the empty *circle* property translates in empty *sphere* property.

3.2 Contour containment

In our application, we have to deal with contour lines, approximated by simple closed polygons. Just triangulating the polygons vertices may result in a triangulation where contour segments are not guaranteed to be edges of the triangulation. Since our issue is to get a 3D polyhedron whose intersection with the given cross sections yields in the original contours, our triangulation has to satisfy the following requirement:

All contour segments have to appear as Delaunay edges in the Delaunay triangulation (contour containment condition).

An easy solution is to insert new points on the concerned contour edges, so that the contour shape is not changed (cf. figure 4).

3.3 Internal and external Voronoi skeleton

Once the containment condition is satisfied, we can divide the Delaunay triangles in two groups: internal triangles lying inside the contours and external triangles lying outside the contours.

We call **internal Voronoi skeleton** the edges of V(S) which are dual to an internal Delaunay edge (that is an edge shared by two adjacent internal Delaunay triangles).

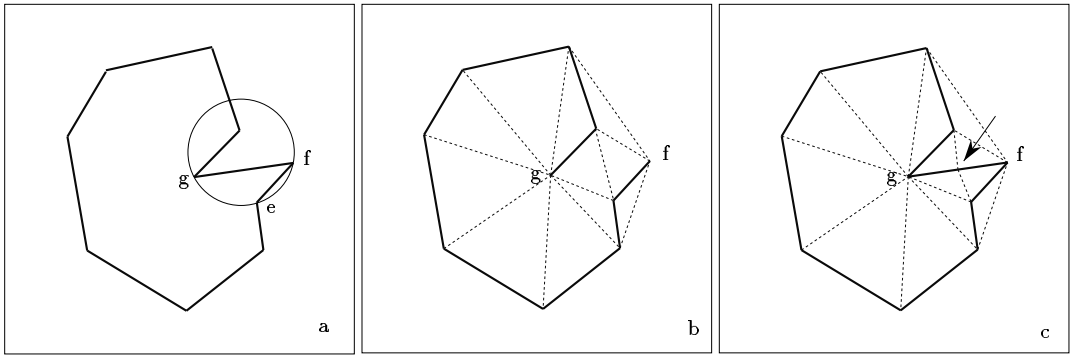

Figure 4: (a) A contour. It is evident, that triangle e,f,g cannot be part of a Delaunay triangulation, since its circumcircle would contain another site. (b) shows the Delaunay triangulation of the contour points. Contour segments which are incident with Delaunay edges are drawn bold. Contour edge f,g is contained after insertion of a vertex on e,f (c).

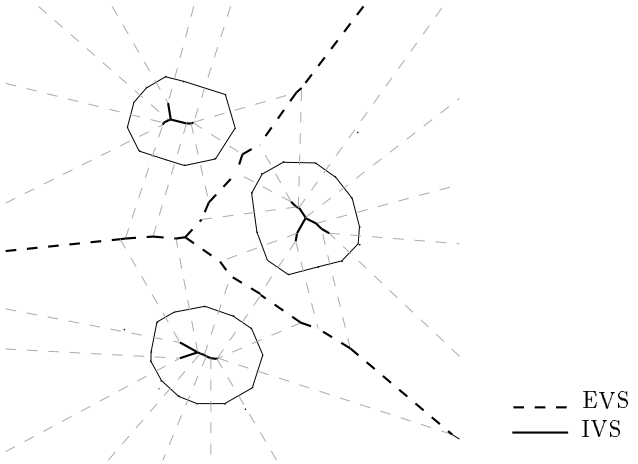

Figure 5: Internal (IVS) and external (EVS) Voronoi skeleton of three contours

Similarly we define **external Voronoi skeleton** the edges of V(S) which are dual to an edge shared by two external Delaunay triangles (See fig. 5).

The internal Voronoi skeleton represents in some sense the contour polygon, but as we mentioned above, the Delaunay edges and the Voronoi edges of a point set may be quite different. The *internal* Voronoi skeleton is not guaranteed to lie *inside* the polygon boundaries. Even worse, polygons with same shape but different distribution of vertices may have totally different Voronoi skeletons (See fig. 6). Thus the quality of shape representation is not very high at this stage.

3.4 Increasing the quality of shape representation

To guarantee that the internal Voronoi skeleton lies inside its contour polygon, we add new vertices on contour segments involved in obtuse Delaunay triangles (See fig. 6):

For each contour segment, we consider the opposite angle of the triangle it belongs to. If it is greater than 90 degrees, its corresponding Voronoi vertex lies outside the triangle. In this case we add a new point at the normal projection of the opposite vertex onto the contour segment. This will divide the obtuse triangles in two right-angled triangles (See fig. 6). It is to point out that this procedure doesn't eliminate all obtuse angles in the Delaunay triangulation, only obtuse angles opposed to contour segments are detected. But it is sufficient to guarantee that the internal Voronoi skeleton stays inside its contour.

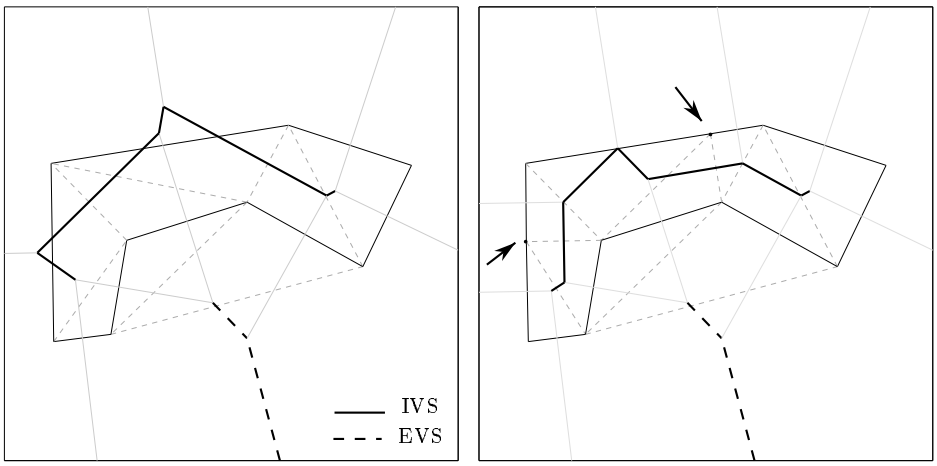

Figure 6: Two vertices have been inserted in the right image to avoid obtuse angles

In general, increasing the number of polygon vertices makes the Voronoi skeleton tending towards the center of the polygon (see fig. 7). Suppose we add n vertices, equally spaced, onto the contour. If we increase n, we get an approximation of the **medial**

axis. The medial axis of a polygon is the locus of points with equal distance to at least two contour points. It is a well studied object in pattern recognition, because it highly represents the metric and topological properties of shapes (cf. [Blu67] [KKOK89]). If several contours are present in one layer, the external Voronoi skeleton or external medial axis is forming a kind of "macro" cells. The points inside such a "macro" Voronoi cell are closer to its contour than to any other contours. Hence the external Voronoi diagram is the locus of equidistance from at least two contours (see fig. 5).

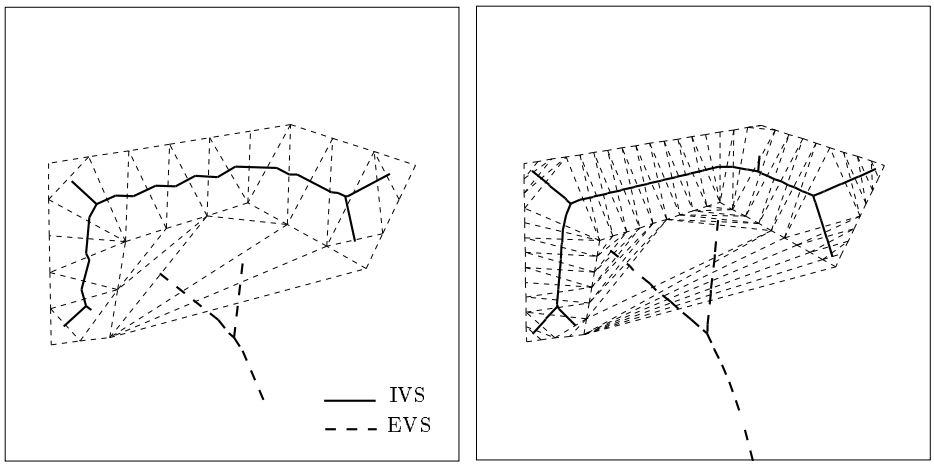

Figure 7: Two approximations of the medial axis, with 27 and 72 vertices on the contour.

The next step would be to calculate the real medial axis, not just its approximation. This can be done by constructing a generalization of the Voronoi diagram - the edge Voronoi diagram (EVD) - where sites are no longer points but entire contour edges (see [Kir79], [Yap87], [For85]). Once again we would get both an internal and external Voronoi skeleton, which now is built of straight line segments and arcs of parabola. The internal Voronoi skeleton is the medial axis representing the contour, the external Voronoi skeleton in fact is the generalized Voronoi diagram where the sites are simple closed polygons (the contours).

Finally we can imagine—we will see the reason later—to fill up the interior of the contours with nearly equally spaced vertices to achieve a regular triangular grid without obtuse angles. The construction of such a mesh is often required in finite element calculation, a solution by successive refinement is shown in Hermeline [Her82]. Because such a mesh satisfies the empty circle condition, it is a Delaunay triangulation. The internal Voronoi skeleton of a filled polygon is a mesh of Voronoi cells, its density depends on that of the internal vertices. In case of infinite density, it would represent the entire area inside the contour.

4 Volume reconstruction

We reduce the problem to find an overall reconstruction to computing a solid slice between each pair of adjacent cross sections. In each cross section, our input consists of one or several closed simple polygons, which possibly may lie one inside another. The contours are oriented in such a way that the inside of the object they are describing is on its right side, the outside on its left.

Let P_1 and P_2 be two adjacent cross sections with C_1 and C_2 the set of contours in each plane. $C_1 = \{c1_i | i = 1..n_1\}$ and $C_2 = \{c2_i | i = 1..n_2\}$, with ci_j being simple closed polygons.

Let S_i be the set of vertices of $C_i (i = 1, 2)$.

4.1 The subdivision of the plane

We first make a subdivision of the planes P_1 and P_2, but unlike to the voxel method, where the plane is divided into a regular grid (pixels), we choose the Delaunay triangulation of the contour vertices. If necessary, additional vertices are added on the contours, to satisfy the contour containment condition and to avoid obtuse angles, as described in the previous sections (see fig. 8).

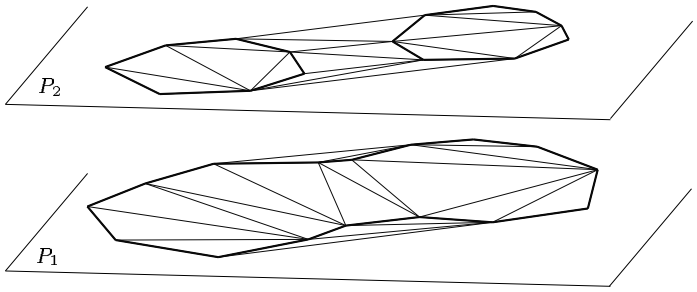

Figure 8: Delaunay triangulation of contours in two adjacent planes

The first advantage is the data reduction, since the number of triangles is linearly related to the number of contour vertices. To represent the object regions by a regular grid would cost space in quadratic terms.

There are several reasons to take the Delaunay triangulation :

- There exists several well known algorithms to calculate it.
- The shape of a Delaunay triangle is in general compact (maximizing the minimal angle).
- Due to its close relationship to Voronoi diagrams, we can perform fast point location in it

4.2 2D to 3D mapping

In the next step, for each triangle $t \in P_1$ we search the vertex $v \in P_2$ which lies closest to the circumcircle of t. We connect t with v to obtain a tetrahedron. The same is done with P_1 and P_2 inversed (see fig. 9).

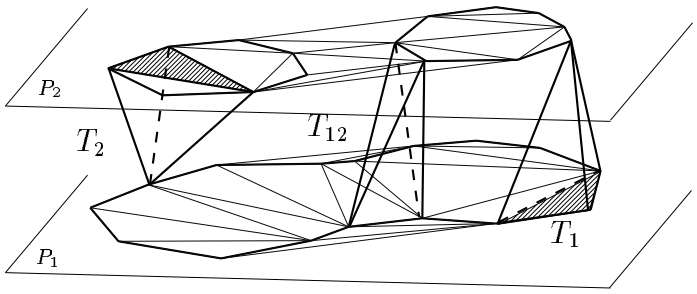

Figure 9: Connecting triangles to tetrahedra

Up to now our triangulation consists of tetrahedra which we call of type t_1 or t_2, depending on whether they have a facet in P_1 and a vertex in P_2, or a facet in P_2 and a vertex in P_1. The third kind of tetrahedra, called t_{12}, having one edge in P_1 and one edge in P_2, is still missing. We define the graph $G = V(S_1) \cup proj(V(S_2))$ the union of the Voronoi diagram of S_1 and the orthogonal projection of the Voronoi diagram of S_2 onto the Voronoi diagram of S_1. All information is then contained in that graph: A Voronoi vertex of $V(S_1)$ is representing a t_1 tetrahedron , a Voronoi vertex of $V(S_2)$ is representing a t_2 tetrahedron , and each intersection of two Voronoi edges corresponds to a t_{12} tetrahedron . We call the vertices in G resp. t_1, t_2 or t_{12} vertices.

As shown in [Boi88], the resulting triangulation is in fact the 3D Delaunay triangulation of the vertices of $S_1 \cup S_2$. Its border is the convex hull of $S_1 \cup S_2$.

4.3 Elimination step

We have to eliminate all tetrahedra having an edge in P_1 or P_2 outside the contours (see fig. 10). This corresponds to remove all vertices from G lying on external Voronoi skeletons.

There may still remain *non solid connections*, tetrahedra which are only connected along an edge or at one single point to one of the two planes (see fig. 11). We can find and remove these portions in our graph G.

If we discuss the behavior of the reconstruction on our graph G, we observe the following facts:

- External Voronoi skeletons cut the object in parts. We call E_i the regions (cells) formed by the intersection of the external Voronoi skeletons in G ($i = 1..k$).

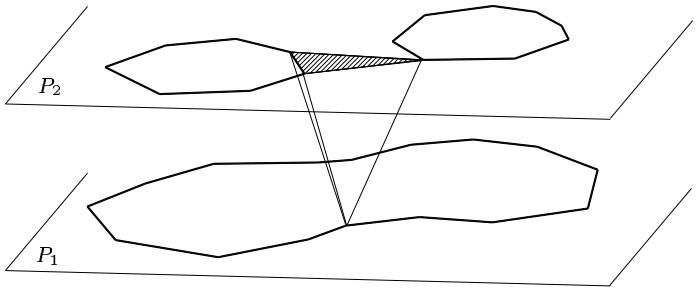

Figure 10: Tetrahedra lying outside the contours

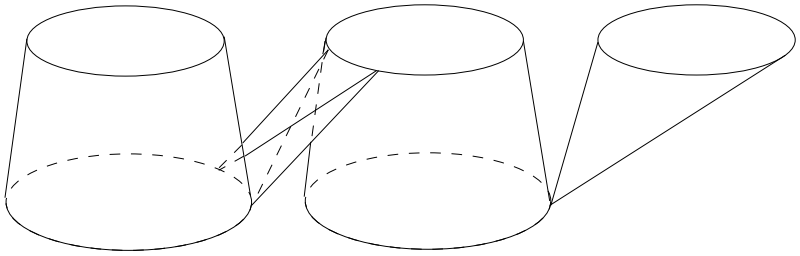

Figure 11: Non solid connections

- If an external Voronoi skeleton crosses an internal Voronoi skeleton, its corresponding contour is split along the Delaunay edge dual to the intersected Voronoi edge (See fig. 12).
- Those parts of internal Voronoi skeletons lying in the same region E_j will correspond to the connected contour parts in the final reconstruction.
- If the internal and external skeletons are good approximations of the medial axis (that is, the number of vertices on the contour polygons is sufficiently high), contours will split along the external medial axis. Contour parts will be connected to their nearest-neighbor-contour parts.

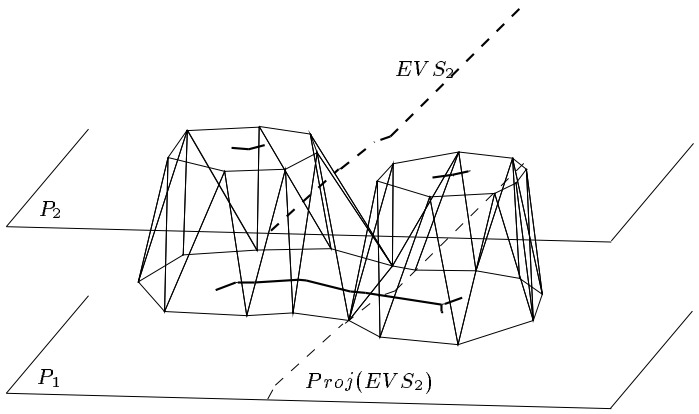

Figure 12: Contour split

4.4 Introducing internal vertices

So far, this reconstruction algorithm works well for simple contours. With complex contours, especially for complicated branching and birth of holes, there may rest some unsatisfying results, which arise from the fact that there are no vertices inside the contours (see fig. 13). Contour splits only can occur along one single edge which is crossing the entire contour.

What would happen if we fill up the interior of the contours in a way to achieve a regular, non obtuse triangle mesh described in the previous section, and apply our reconstruction algorithm on this configuration?

In figure 14, the external Voronoi skeleton EVS_2 in P_2 cuts the contour in P_1 in 3 regions: a', b' and c'. Since all small triangles have non-obtuse angles, their circumcenters are lying inside the triangles. So all the triangles lying in region a' are connected to contour a, b' to b and c' to c. The splitting line in P_1 would approximate more or less the projection of EVS_2, dependent on the size of the small triangles and hence on the number of inserted vertices. Fortunately, it is not necessary to fill the

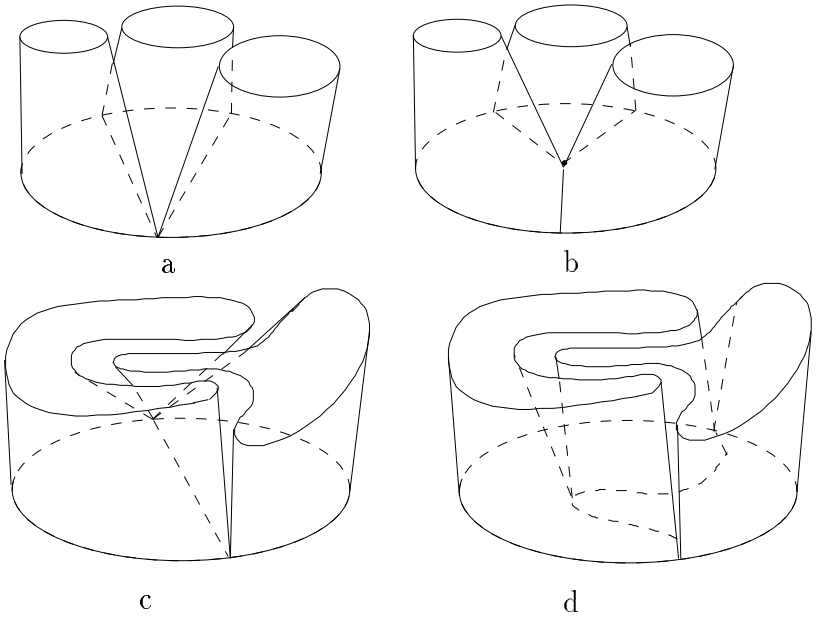

Figure 13: Internal points: (a),(c) solutions without internal vertices, (b),(d) would be better solutions

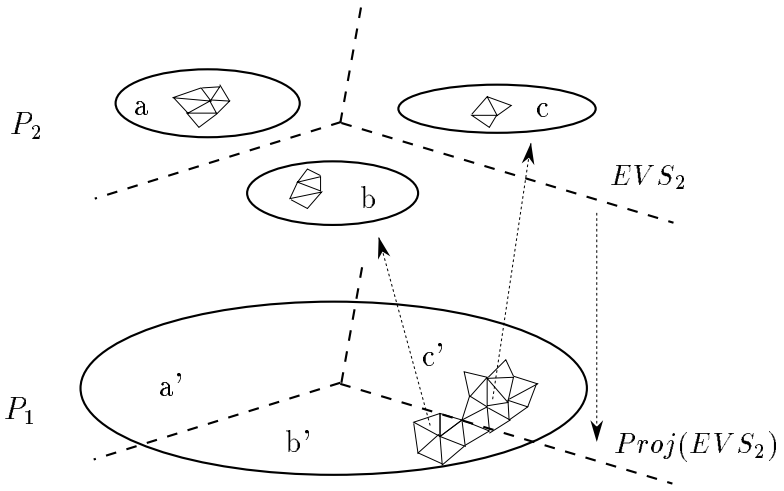

Figure 14: Contours with small triangular mesh inside

entire contours to get this result. It is sufficient to put vertices onto the projections of the external Voronoi skeletons in the opposite planes.

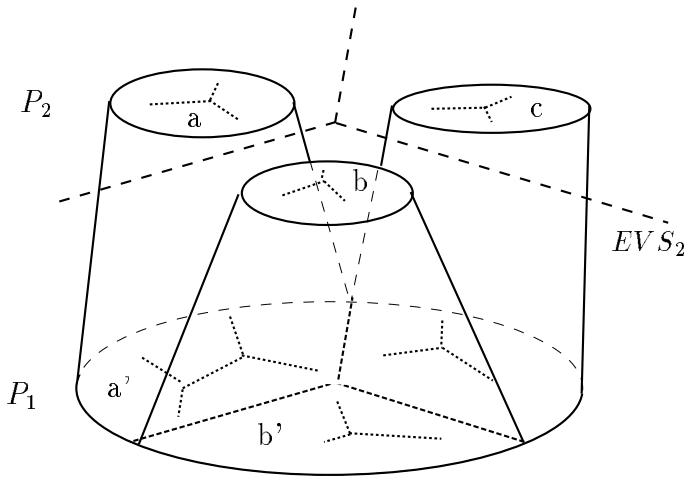

Figure 15: Internal vertices along the projection of the external Voronoi skeleton

In figure 15, vertices have been inserted inside the contour in P_1, along the projection of EVS_2. The contour contains now three internal Voronoi skeletons, which lie inside the regions a', b' and c', if the density of inserted points is sufficient, and the procedure to avoid obtuse angles is applied. Thus the region a' is connected to contour a, b' to b and c' to c. The splitting line now follows the projection of EVS_2, which itself is the approximation of the medial axis (see fig. 16).

Our complete reconstruction algorithm looks now as follows:

1. In each cross section do:
   (a) Compute the 2D Delaunay triangulation of the vertices
   (b) Add vertices to satisfy the contour containment condition
   (c) Add vertices to avoid obtuse angles
   (d) Add vertices along the external Voronoi skeleton inside the contours of the two adjacent sections
2. With each pair of adjacent cross sections do:
   (a) Extend the two 2D triangulations to one 3D Delaunay triangulation
   (b) Remove external tetrahedra and non-solid connections

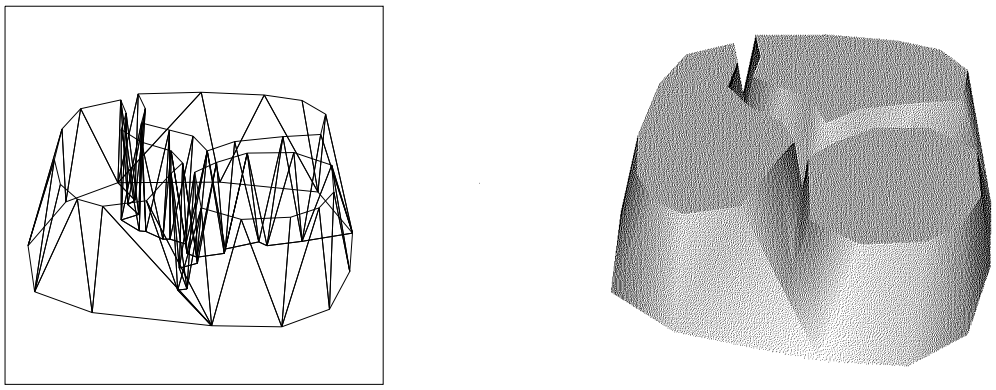

Figure 16: An example of splitting one contour into three branches.

	cross sections	contours	points	added points	cpu time	time/ point	surface tiles
pelvis	23	84	2059	622	6.50s	3.15ms	5518
knee	42	106	2028	191	5.33s	2.62ms	4386
heart	29	58	1222	243	3.60s	2.94ms	2988
head	17	25	856	155	2.35s	2.74ms	2032

Figure 17: Some experimental results

5 Experimental Results

The algorithm has been implemented in C and tested with various medical data. In all practical cases, the cpu time is almost linearly related to the number of contour vertices, because we are exploiting the fact that input vertices are sorted along contours. This allows a constant time point location. The cpu time on a DECstation 5000 is less than 4ms per vertex. Experience has shown that the number of added vertices depends on the contour complexity and does rarely exceed 30%. Fig. 17 shows some typical results. The pelvis in fig. 20 has been rendered on a Personal Iris.

6 Conclusion

We have shown a solution to the reconstruction problem using the 3D Delaunay triangulation. The heuristics are based on the notion of the medial axis, thus realizing a *nearest neighbor* connection between adjacent planes. Especially, we showed how to calculate an approximation of the medial axis, whose quality is adjustable by adding automatically vertices on and inside the contours.

Our method provides a 3D polygonal representation, composed of tetrahedra. The

surface triangles are used for visualization. The tetrahedra structure is useful for further applications, like simulation of motion or finite element methods.

Acknowledgment

We are grateful to Frank Muennemann from Resonex Inc., Sunnyvale CA for providing the MR data of the pelvis. The figures in this document have been created with JPdraw.

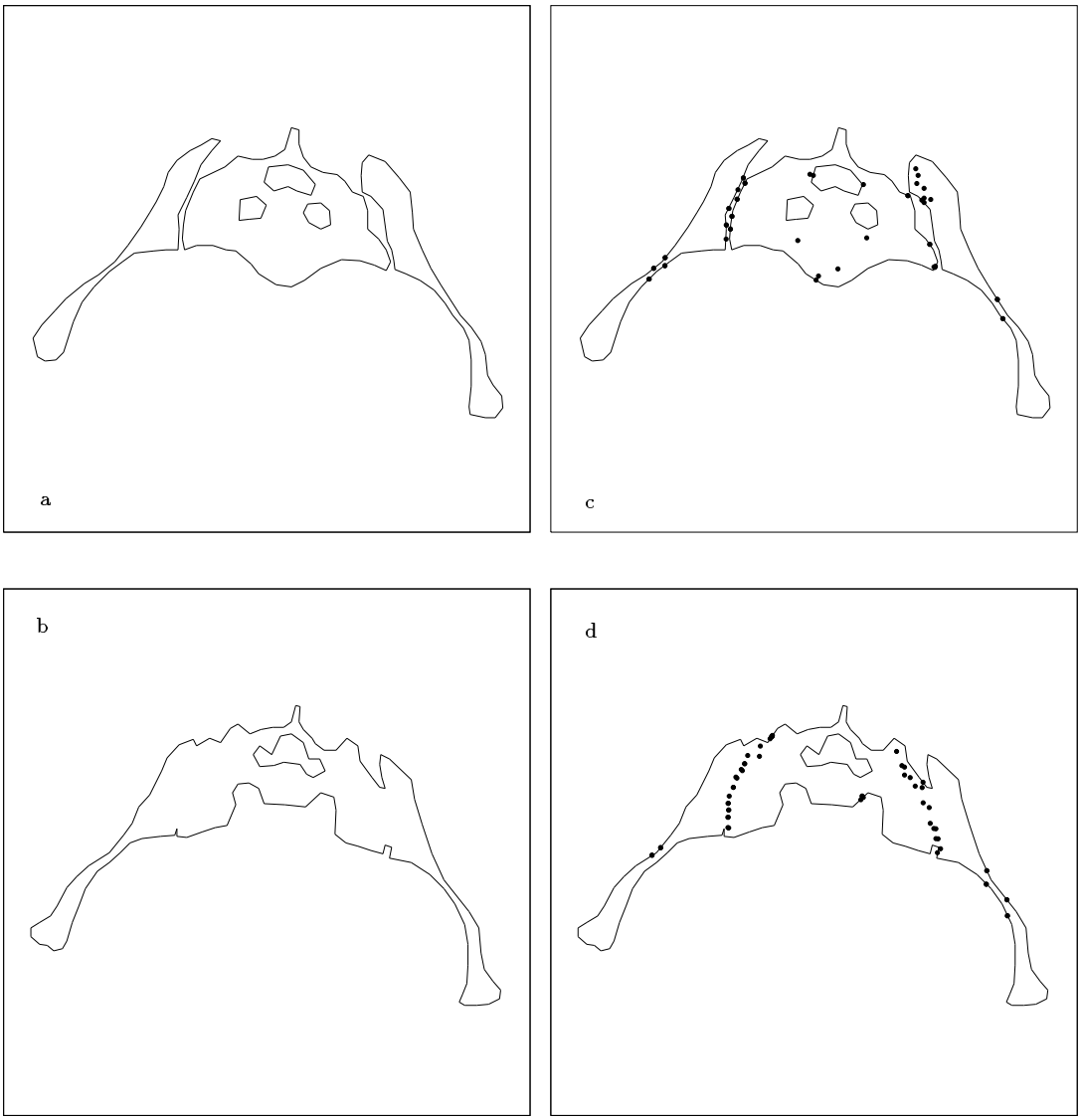

Figure 18: a) and b) show two adjacent cross sections of a pelvis. Slice distance is about 8mm. The vertices inserted during the reconstruction are shown in c) and d).

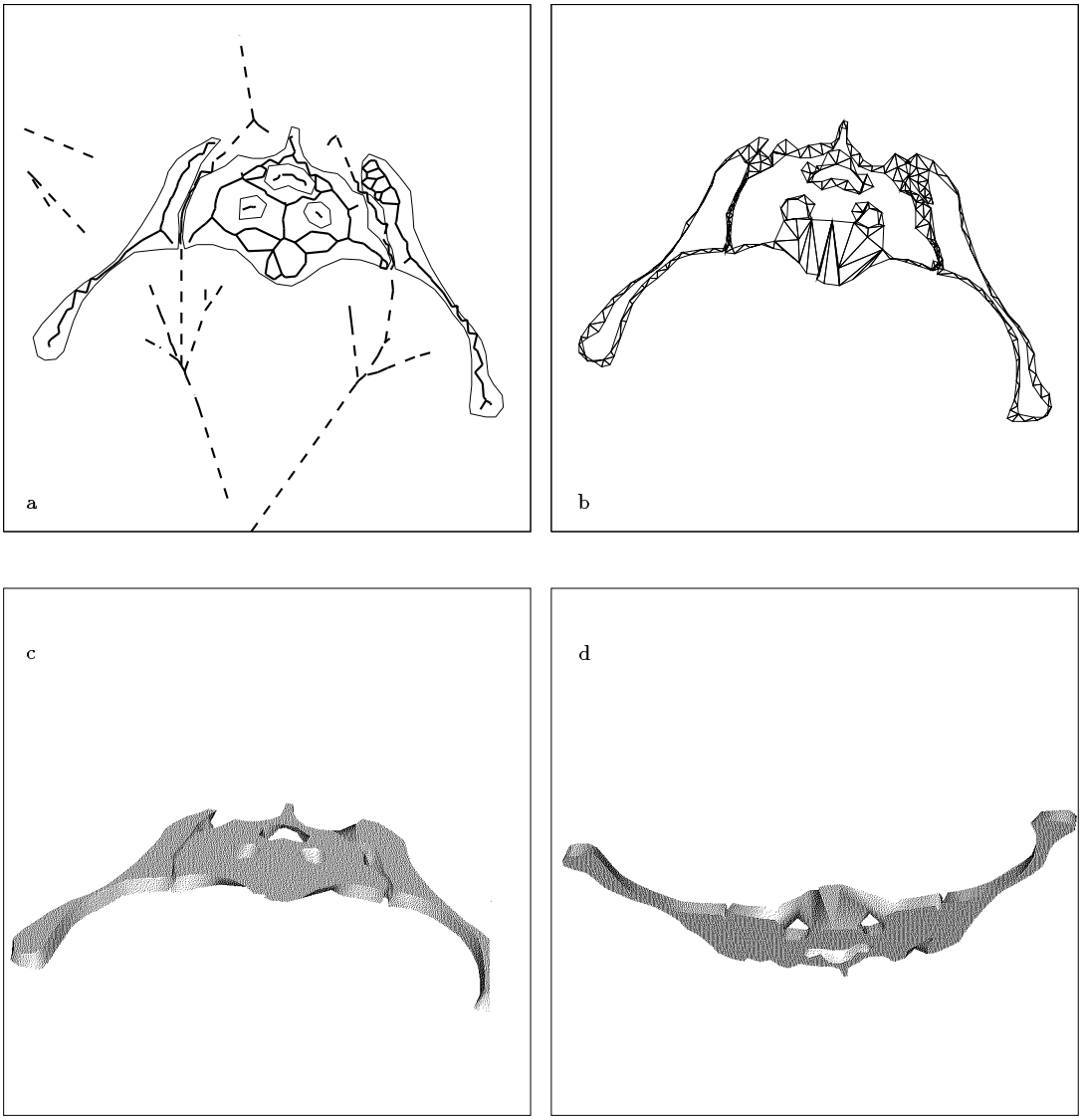

Figure 19: a) The upper cross section of fig. 18 with internal (bold) and external (bold dashed) Voronoi skeleton. b) The reconstructed surface between the two sections, seen from above. c) and d) are two shaded perspective views of the slice.

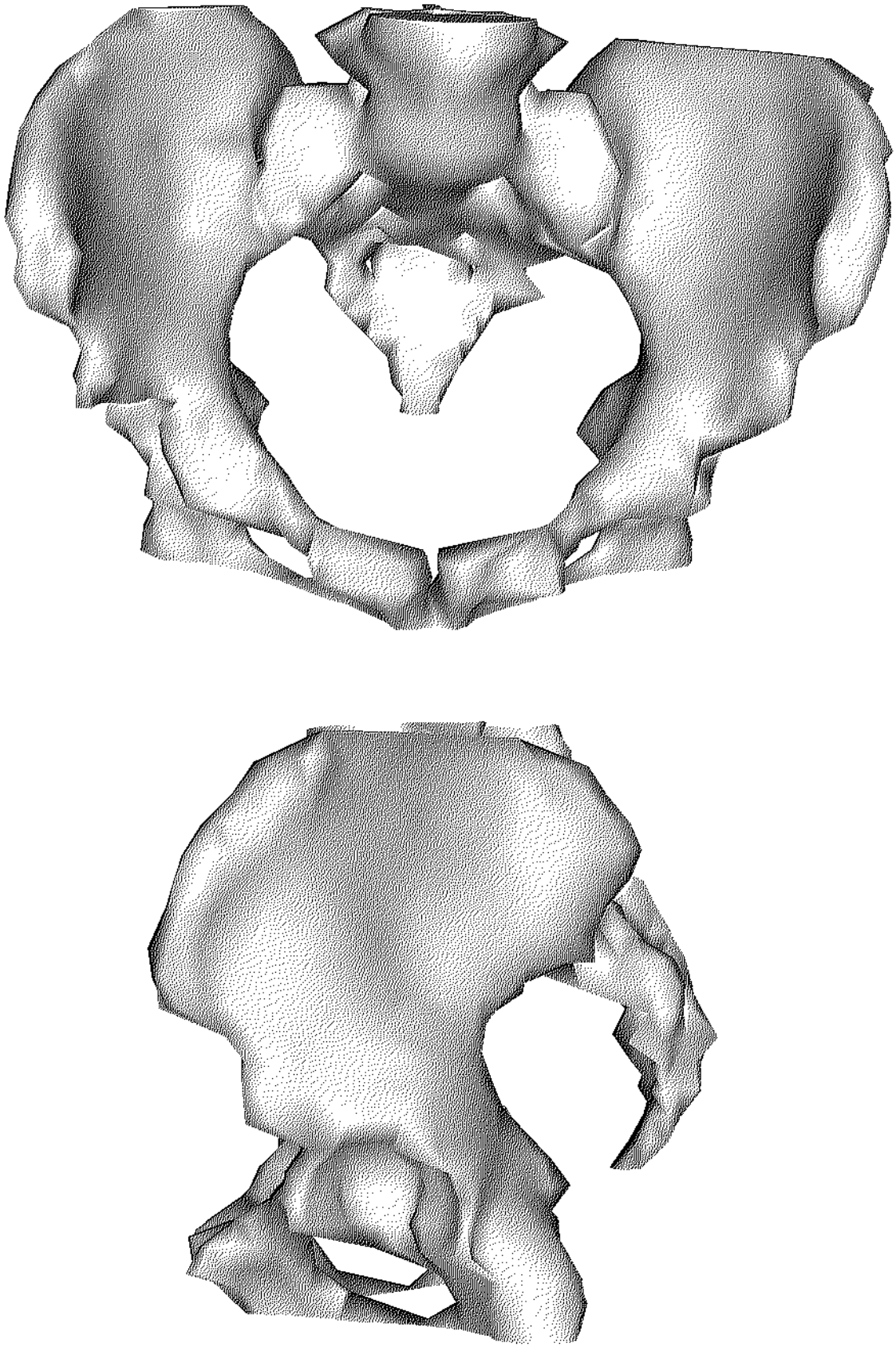

Figure 20: The overall reconstruction of a human pelvis from 23 MR images

References

[BGLS85] C. Barillot, B. Gilbaud, L. M. Luo, and J.M. Scarabin. 3-d representation of anatomic structures from CT examination. In *Biostereometrics*, pages 307–314, 1985.

[Blu67] H. Blum. A transformation for extracting new descriptors of shape. In W. Walthen-Dunn, editor, *Models for the perception of speech and visual form*. MIT Press, Cambridge, MASS, 1967.

[Boi88] J-D. Boissonnat. Shape reconstruction from planar cross-sections. *Computer Vision, Graphics, and Image Processing*, 44:1–29, 1988.

[CS78] H. N. Christiansen and T. W. Sederberg. Conversion of complex contour line definitions into polygonal element mosaics. *Computer Graphics*, 8:658–660, August 1978.

[FKU77] H. Fuchs, Z. Kedem, and S.P. Uselton. Optimal surface reconstruction from planar contours. *Communications of the ACM*, 20:693–702, 1977.

[For85] S. J. Fortune. Fast algorithms for polygon containment. In *LNCS 194 (12th Colloquium on Automata, Languages and Programming)*, pages 189–198. Springer-Verlag, 1985.

[HB86] K.H. Hoehne and R. Bernstein. Shading 3d-images from CT using gray-level gradients. *IEEE Trans. Medical Imaging*, pages 45–47, 1986.

[Her82] F. Hermeline. Triangulation automatique d'un polyèdre en dimension n. *RAIRO Modél. Math. Anal. Numer.*, 16(3):211–242, 1982.

[Kep75] E. Keppel. Approximating complex surfaces by triangulation of contour lines. *IBM Journal of res. and development*, 19:2–11, 1975.

[Kir79] D.G. Kirkpatrick. Efficient computation of continuous skeletons. In *IEEE Symposium on Foundations of Computer Science*, volume 20, pages 18–27, 1979.

[KKOK89] O. Kuebler, F. Klein, R. Ogniewicz, and U. Kienholz. Isolation and identification of abutting and overlapping objects in binary images. In *5th internat. Conference on Image Analysis and Processing*, Positano, Italy, September 1989. World Scientific.

[LC87] W. Lorensen and H. Cline. Marching cubes: a high resolution 3d surface construction algorithm. *Computer Graphics*, 21:163–169, 1987.

[Lev88] M. Levoy. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, pages 29–37, 1988.

[MSS91] D. Meyers, S. Skinner, and K. R. Sloan. Surface from contour: The correspondence and branching problems. In *Graphics Interface*, 1991.

[PS85] F.P. Preparata and M.I. Shamos. *Computational Geometry : an Introduction*. Springer-Verlag, 1985.

[SFF91] M. R. Stytz, G. Frieder, and O. Frieder. Three-dimensional medical imaging: Algorithms and computer systems. *ACM Computing Surveys*, 23(4):421–499, December 1991.

[Sha81] M. Shantz. Surface definition for branching contour defined objects. *Computer Graphics*, 15:242–270, July 1981.

[Toe89] K. D. Toennies. Surface triangulation by linear interpolation in intersection planes. In *Science and Engineering of Medical Imaging*, pages 98–105, 1989.

[Yap87] C.K. Yap. An $O(n \log n)$ algorithm for the Voronoi diagram of a set of simple curve segments. *Discrete and Computational Geometry*, 2:365–393, 1987.