Eurographics Symposium on Geometry Processing (2007)
Alexander Belyaev, Michael Garland (Editors)

Shape Reconstruction from Unorganized Cross-sections

Jean-Daniel BOISSONNAT Pooran MEMARI

INRIA Sophia-Antipolis, France

Abstract

In this paper, we consider the problem of reconstructing a shape from unorganized cross-sections. The main motivation for this problem comes from medical imaging applications where cross-sections of human organs are obtained by means of a free hand ultrasound apparatus. The position and orientation of the cutting planes may be freely chosen which makes the problem substantially more difficult than in the case of parallel cross-sections, for which a rich literature exists. The input data consist of the cutting planes and (an approximation of) their intersection with the object. Our approach consists of two main steps. First, we compute the arrangement of the cutting planes. Then, in each cell of the arrangement, we reconstruct an approximation of the object from its intersection with the boundary of the cell. Lastly, we glue the various pieces together. The method makes use of the Delaunay triangulation and generalizes the reconstruction method of Boissonnat and Geiger [BG93] for the case of parallel planes. The analysis provides a neat characterization of the topological properties of the result and, in particular, shows an interesting application of Moebius diagrams to compute the locus of the branching points. We have implemented our algorithm in C++, using the [CGAL] library. Experimental results show that the algorithm performs well and can handle complicated branching configurations.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling.

Introduction

The reconstruction of an object from a set of cross-sections has intrigued computer science researchers for the last decades. The need for such reconstructions is a result of the advances in medical imaging technology. From data obtained by CT, MRI, or other systematic scanning devices, contours representing the boundaries of the organs may be extracted on slices, and then interpolated in order to reconstruct and visualize human organs. 3D reconstructions of organs are widely considered to be an important diagnostic aid in the medical world.

Since the work of Keppel [Ke75] on tiling between parallel polygonal contours, numerous algorithms were introduced for parallel inter-slice interpolation. The problem is considered to be quite difficult because the topology of the contours may change between slices. Some progress was made with the introduction of the Delaunay-based technique of Boissonnat [Bo88], and the method of Bajaj et al. [BCL96]. Both approaches attempt to handle the most general case, in which the geometries and topologies of the contours in every slice are totally unrestricted. Subsequently, Barequet and Sharir [BS96] suggested an interpolation method based on geometric hashing. In this method similar sub-contours are identified first and stitched together, while the remaining contour portions are triangulated so as to minimize the surface area of the reconstructed solid. Later, Barequet et al. [BGLS04] suggested another interpolation algorithm that uses the medial axis of the overlay of the two slices. This method generates a smooth and intuitive reconstruction since it inherently captures the differences between the slices.

Now that 3D reconstruction from parallel slices, as produced by CT and MRI, is more or less solved, we are given the more difficult problem of 3D reconstruction from non-parallel slices (see Figure 1). This need arises in the use of freehand 2D ultrasound. Information on the orientation of each individual cutting plane is available through the use of a navigation device coupled with the probe, producing 6 degrees of freedom. This allows the registration of the contours extracted from the images to a global coordinate system, but certainly does not indicate how the contours should be interpolated in order to reconstruct the full object geometry.

The 2D analog of the problem has been previously considered by Coll and Sellares [CS01a] and Sidlesky, Barequet and Gotsman [SBG06]. The incremental algorithm proposed in [CS01a], processes the cutting lines sequentially. At each step, the reconstruction is updated in such a way that the time complexity of the overall algorithm is logarithmic in the total number of cutting lines. In [SBG06], the authors make use of the arrangement of the cutting lines and consider the portions of one-dimensional cross-sections, lying on the boundary of each cell of the arrangement. Then, using some inter-cell relations, they enumerate all possible reconstructions that are conform with the given cross-sections.

To the best of our knowledge, the 3D problem has only been considered by Payne and Toga [PT94], and an unpublished work [DP97], by Dance and Prager. The method of Payne and Toga is restricted to simple shapes and does not allow any branching. It produces poor results for shapes with complex topologies. An approach with some similarities to the one reported here, is presented in an unpublished report [DP97]. This method builds also the arrangement of the cutting planes and makes use of the Delaunay triangulation of the section vertices. New contributions of our paper are : a neat characterization of the notions of branching and cutting loci, additional guarantees on the reconstructed shape and experimental results showing that the method works well on objects with complex topologies.

In this paper, we present a Delaunay-based method to solve the general 3D reconstruction problem. Given some cutting planes (whose positions and orientations may be arbitrary) and a polygonal approximation of their cross-sections with an object, we propose a method to construct an approximation of the object whose intersection with the cutting planes coincides with the input sections. This approach is a generalization of the algorithm proposed by Boissonnat and Geiger [Bo88] for parallel cross-sections.

The approach consists of three main steps. First, we compute the arrangement of the cutting planes (i.e. the subdivision of space induced by the cutting planes). Then, in each cell of the arrangement, we reconstruct an approximation of the object from its intersection with the boundary of the cell. Lastly, we glue the various pieces together. As the first step is standard in Computational Geometry and the last one is easy, our main contribution is a method for reconstructing a 3d-solid shape from its intersection with the boundary of a given convex polyhedron $\mathcal{C}$. The method is general and can handle any input section including, in particular, complex branching configurations and nested contours.

Of utmost importance is the characterization of the branching locus, i.e. the set of points of the sections where the topology of the sections changes when we slightly perturb the cutting planes. We characterize the branching locus in a face of $\mathcal{C}$ as a (subset of a) Moebius diagram, an extension of the Euclidean Voronoi diagram [BK03]. This characterization extends on the work of Boissonnat and Geiger [BG93] for the case of two parallel cutting planes where the branching locus was defined as (a subset of) a Euclidean Voronoi diagram, and on the work of Boissonnat et al. [BCDT96] for the case of two *non parallel* cutting planes where the branching diagram was shown to be a (subset of) a hyperbolic Voronoi diagram (a special case of Moebius diagram).

The algorithm consists in constructing the 3-dimensional Delaunay triangulation of the union of the contour vertices and of the branching points. We then prune this triangulation so as to obtain a 3-dimensional triangulated manifold (with boundary) that does not intersect the cutting planes outside the given sections. Some geometric and topological properties of the reconstructed manifold are established. Moreover, the algorithm has been implemented in C++ using the [CGAL] library and experimental results are discussed.

The paper is organized as follows. The two first sections of this paper contain a short review of the problem and the basic concepts and definitions which will be used further on. The core of the reconstruction is studied in the third section, by introducing the *branching diagram*. Then, the reconstruction algorithm whose complexity is that of the 3D Delaunay triangulation, is presented. In the last section, we give some experimental results of the proposed algorithm.

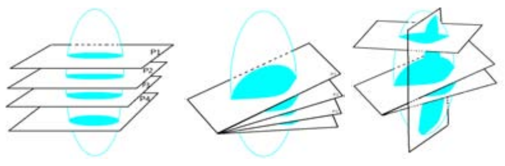

Figure 1: *Different cutting planes positions: (a) Parallel serial sequence of planes (b) non-parallel serial sequence (c) arbitrary cutting planes.*

1. Problem Description

Let $P_1, \ldots, P_k$ be a set of planes, called the *cutting planes*, that intersect a compact set $\mathcal{O}$ of $\mathbb{R}^3$. We call *section* of $\mathcal{O}$ by P_i the intersection $A_i = P_i \cap \mathcal{O}$, $i = 1, \ldots, k$. Given the sections $A_1, \ldots, A_k$, our goal is to reconstruct an approximation $\mathcal{R}$ of $\mathcal{O}$ whose intersection with the cutting planes is identical to the sections, i.e. $\mathcal{R} \cap P_i = A_i$, $i = 1, \ldots, k$.

The reconstructed object will consist of line segments (called *fibers*) joining the sections. It will satisfy two main conditions : first, no fiber will intersect a cutting plane outside a section (*conformity condition*); second, the reconstructed object $\mathcal{R}$ will be a 3-dimensional manifold with boundary (*solidity condition*).

1.1. Conformity condition and the arrangement

Since no fiber can intersect a cutting plane outside a section (conformity condition), we can decompose the problem into several subproblems as follows. Consider the arrangement $\mathcal{P}$ of the cutting planes, i.e. the subdivision of $\mathbb{R}^3$ into convex polyhedral cells induced by the P_i (see Figure 2). Any fiber that conforms to the cutting planes either is contained in a cell of $\mathcal{P}$ or can be decomposed into at most $k-1$ subsegments, each contained in a cell of $\mathcal{P}$. Conversely, any fiber with its two endpoints on the boundary of a cell of $\mathcal{P}$ does

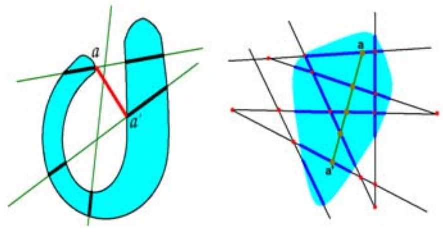

Figure 2: *Conformity condition and the arrangement of the cutting planes:(Left) Fiber aa' violates the conformity condition. (Right) Fiber aa' can be decomposed into subsegments, each contained in a cell.*

not cross any cutting plane except at its endpoints and therefore fulfills the conformity condition. Hence, without loss of generality, we can restrict our attention to fibers inside a cell of $\mathcal{P}$ and reduce the reconstruction of $\mathcal{O}$ to the reconstruction of $\mathcal{O} \cap \mathcal{C}$ for all cells $\mathcal{C}$ of $\mathcal{P}$. Since the various reconstructed pieces will conform to the given sections, it will be easy to glue them together in the end and their union will constitute the overall reconstructed object $\mathcal{R}$.

In the sequel, we focus on a cell $\mathcal{C}$ of the arrangement and describe how to reconstruct the corresponding portion $\mathcal{R}_\mathcal{C} = \mathcal{R} \cap \mathcal{C}$ of the object. On each face f of $\mathcal{C}$, we are given a set of polygonal regions that approximate the intersection of the object $\mathcal{O}$ with f. This set is called the *section* of f and we denote by $\mathcal{A}_\mathcal{C}$ the union of the sections of all the faces of $\mathcal{C}$. $\mathcal{A}_\mathcal{C}$ is the only information we know about $\mathcal{O}$ and constitutes the input of our algorithm.

Notations

Section: A set of polygonal regions, possibly with holes, that approximate the intersection of $\mathcal{O}$ with a face of $\mathcal{C}$.
Section-contour: A connected component of the boundary of a section. Note that the section-contours form a union of non-intersecting polygons, some possibly lying inside others.
Section-point: Any point of a section, which could itself be a *contour-point* or an *interior-point*.
Contour-vertex: Any vertex of a polygonal region of $\mathcal{A}_\mathcal{C}$.
$\mathcal{V}_\mathcal{C}$ denotes the set of contour-vertices.

1.2. Solidity Condition in a cell of the arrangement

The solidity condition forbids connections between sections along zero or one dimensional features, as shown in the 2D example of Fig 3*c*. To state this condition more formally, we introduce the following definitions.

Definition 1 (Extended $\mathcal{A}_\mathcal{C}$, Extended $\mathcal{R}_\mathcal{C}$) To any section-point $p \in \mathcal{A}_\mathcal{C}$, we associate the *normal cone* of $\mathcal{C}$ at p, i.e. the set of rays issued from p, lying outside $\mathcal{C}$ and directed along the normals to the supporting planes of $\mathcal{C}$ at p. The extended $\mathcal{A}_\mathcal{C}$ is the union of all such rays, noted $\widetilde{\mathcal{A}_\mathcal{C}}$. $\widetilde{\mathcal{R}_\mathcal{C}}$ is then defined as $\mathcal{R}_\mathcal{C} \cup \widetilde{\mathcal{A}_\mathcal{C}}$, see Fig 3.

Definition 2 (Singularity) Points that do not have ball or half-ball neighborhoods in $\widetilde{\mathcal{R}_\mathcal{C}}$ are called singular. In this case, we say that $\widetilde{\mathcal{R}_\mathcal{C}}$ has a singularity at such a point.

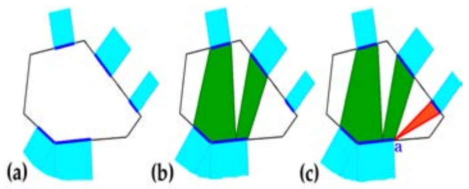

Figure 3: *Extended reconstruction and the solidity condition: (a) $\widetilde{\mathcal{A}_\mathcal{C}}$ (colored in blue) is the union of the normal cones. (b) For this reconstruction $\mathcal{R}_\mathcal{C}$ (in green), $\widetilde{\mathcal{R}_\mathcal{C}} = \mathcal{R}_\mathcal{C} \cup \widetilde{\mathcal{A}_\mathcal{C}}$, is a 3-manifold (with boundary). (c) This extended reconstruction has a singularity at point a.*

In the sequel, we will ensure that $\widetilde{\mathcal{R}_\mathcal{C}}$ has no singularity and therefore is a 3-manifold (with boundary).

2. Empty Spheres and Delaunay Simplices

Our method is based on the Delaunay triangulation and the related notion of empty spheres.

Definition 3 (Empty sphere) Let K be a set of points. The sphere S is called K-empty, if the open ball bounded by S does not include any point of K.

Definition 4 (Delaunay simplex) Let K be a set of points. A simplex with vertices in K is called Delaunay, if there exists a K-empty sphere passing through all its vertices.

To give some intuition behind our method, consider the maximal $\mathcal{A}_\mathcal{C}$-empty spheres. These spheres pass through at least two section-points. We call *Delaunay fibers*, the line segments joining the pairs of points lying on two different cutting planes. The Delaunay fibers are candidates to be fibers of the reconstructed object.

Definition 5 (ϕ and ϕ^{-1}) Let f be a face of $\mathcal{C}$, and P be its supporting cutting plane. To any section-point a of f, we associate S_a, the maximal $\mathcal{A}_\mathcal{C}$-empty sphere tangent to P at a. We call $\phi(a)$ the set of section-points (distinct from a) lying on S_a, see Fig 4a. For a section-point a, $\phi^{-1}(a)$ denotes the set of section-points b, such that $a \in \phi(b)$.

By definition, for any a, if $b \in \phi(a)$ then $[ab]$ is a Delaunay fiber. Similarly, if $b \in \phi^{-1}(a)$ then $[ab]$ is a Delaunay fiber.

2.1. Branching Diagram

Generically, $\phi(a)$ consists of a single point. An important situation occurs when for some a, $\phi(a)$ contains more than one point. This defines the branching locus of the reconstruction.

Definition 6 (Branching diagram) Let f be a face of $\mathcal{C}$. A section-point $a \in f$ for which S_a passes through two other section-points (distinct from a) is called a *branching point*, see Fig 4c. The *branching diagram* of f, B_f, is defined as the

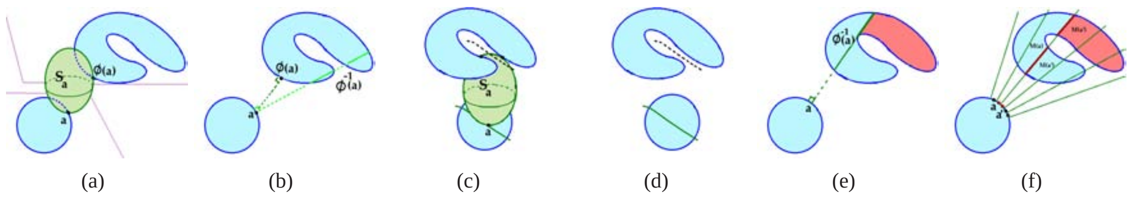

Figure 4: *(a) $\phi(a)$ is on the maximal $\mathcal{A}_\mathcal{C}$-empty sphere tangent to P at a. (b) ϕ and ϕ^{-1} (in the case of two parallel planes). (c) a is a branching point if $\phi(a)$ contains more than one point. (d) Branching diagram example (in green). In the case of parallel planes, it is the orthogonal projection of the external medial axis of the adjacent section. (e) Flat zone (in red) and Cutting diagram (in green). ϕ^{-1} looses connectivity at a. (f) Approximation of the cutting diagram using the Moebius diagram of the contour-vertices. $M(a)\cap M(a')$ is on the cutting diagram if $M(a)$ is connected and $M(a')$ is not.*

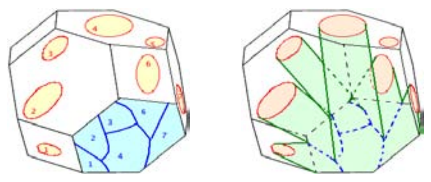

Figure 5: *Left: Branching diagram of a face f (in blue) of $\mathcal{C}$. Right: The set of Delaunay fibers (in green) with an endpoint on f.*

locus of the branching points on f. The branching diagram of $\mathcal{C}$, is the union of B_f, for all faces f of $\mathcal{C}$, i.e.

$$B_\mathcal{C} = \{a \in \mathcal{A}_\mathcal{C},\ |\phi(a)|\ \geq 2\}.$$

In the case of parallel cutting planes, $\mathcal{C}$ is formed by two consecutive planes P_1 and P_2. We write A_1 and A_2 for the sections of P_1 and P_2 respectively. The branching diagram of P_i is then the restriction to A_i of the orthogonal projection onto P_i of the external medial axis of A_{i+1} (indices being taken modulo 2). See Fig 4*d* for an example.

In the case or arbitrary cutting planes (Fig 5), the restriction of the branching diagram to a face of $\mathcal{C}$ is a variant of a known 2-dimensional diagram, called *Moebius diagram*. This diagram is a generalization of both power diagrams and multiplicatively weighted Voronoi diagrams. Its edges are circular arcs and it can be computed efficiently [BK03], [BD05]. Although an exact construction of the branching diagram is possible, we suggest, as a simpler solution, to approximate its curved edges by polygonal lines. Hence, in addition to the contour-vertices, we add new vertices inside the sections along the branching diagram.

2.2. Cutting Diagram

For a contour-point a, there may be several $\mathcal{A}_\mathcal{C}$-empty spheres passing through a and tangent to other cutting planes. Hence $\phi^{-1}(a)$ may contain several points. An important situation occurs when $\phi^{-1}(a)$ is not connected, Figure 4b shows an example. In this case, there is a branching along the contour which creates a singularity. To fulfill the solidity condition, some of the fibers $[ab]$, $b \in \phi^{-1}(a)$, will be eliminated. As a consequence, a *flat* zone, i.e. not linked to any other section, may appear in the section containing b. See Figs 4e and 3b. A flat zone is either an entire section or a portion of a section. In the latter case, the boundary of the flat zone is contained in the so-called *cutting diagram*. This diagram is formally defined as the locus of points in $\phi^{-1}(a)$, such that ϕ^{-1} looses connectivity at a.

Similarly to the branching diagram, this cutting diagram can be extracted from the Moebius diagram of the contour-vertices. Let a be a contour-vertex and $b \in \phi^{-1}(a)$ be on face f of $\mathcal{C}$. b belongs to the cell $M(a)$ of a in the Moebius diagram of f. Let a' be a neighbor of a along the section-contour. If $M(a)$ is connected while $M(a')$ is not, then $M(a)\cap M(a')$ is on the cutting diagram, Fig 4f. In addition to the contour-vertices and the added branching points, we add also some points on the cutting diagram.

3. The Initial Delaunay Triangulation

We have characterized some significant points of the sections which are either on the branching diagram or on the cutting diagram. Instead of sampling the whole polygonal regions of the sections, we consider the contour-vertices and a sample of the branching and cutting diagrams.

Let $\mathcal{B}_\mathcal{C}$ be a sample of points on the branching diagram of $\mathcal{C}$ and $\mathcal{M}_\mathcal{C}$ be a sample of points on the cutting diagram. We consider now $DT(\mathcal{V}_\mathcal{C} \cup \mathcal{B}_\mathcal{C} \cup \mathcal{M}_\mathcal{C})$, the 3D Delaunay triangulation of the union of the contour-vertices and these added points. This triangulation, called R_0 in the sequel, contains the reconstructed object $\mathcal{R}_\mathcal{C}$ as a subcomplex. All the considered vertices are in the sections and called *section-vertices*. Let us fix some notations for the simplices of R_0.

Grounded simplex is a simplex whose vertices all lie in a same cutting plane. We distinguish three types of grounded edges depending on the relative position of the edge with the corresponding section

1. **Contour-edge** A grounded edge which is an edge of the section-contour.
2. **Internal-edge** A grounded edge which lies inside the section (but not on its contour).
3. **External-edge** A grounded edge which lies outside the section.

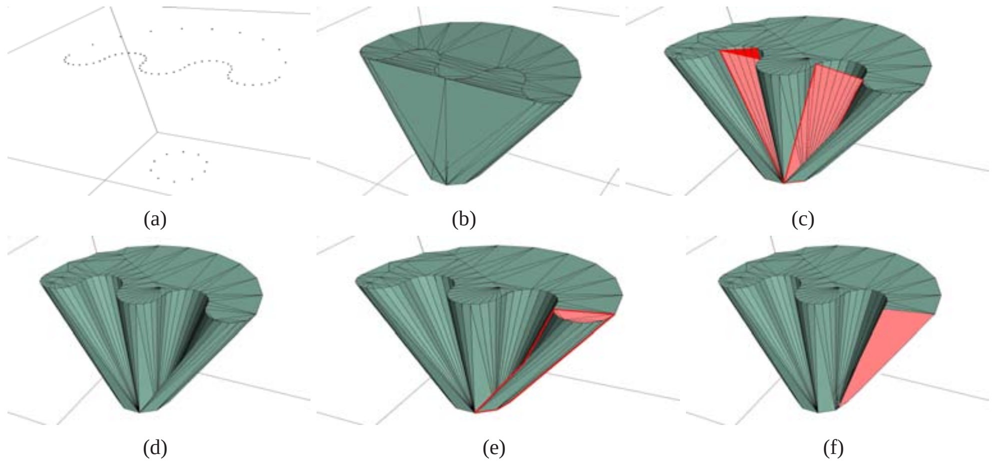

Figure 6: ***Reconstruction Algorithm:*** *(a) Input points (contour-vertices of the sections). (b) The initial Delaunay triangulation. (c) In red, some external tetrahedra which are eliminated in Step 1. (d) The result of Step 1, which has a singularity along a section-contour. (e) In red, the set of eliminated tetrahedra during Step 2. (f) The reconstructed object.*

We distinguish two types of grounded triangles:

1. **Internal-triangle** A grounded triangle which lies inside the section.
2. **External-triangle** A grounded triangle which lies outside the section.

Ground of an edge To each grounded edge e, we associate the two grounded triangles $t_r(e)$ and $t_l(e)$ that are incident to e. The ground of e is the intersection of $\{t_r(e),\ t_l(e)\}$ with the sections. For example, if e is a contour-edge then the ground of e contains a single triangle.

Ground of a section-vertex To each section-vertex v, we associate $G(v)$ the set of grounded triangles incident to v. The *ground* of v is the intersection of $G(v)$ with the section.

Types of tetrahedra The tetrahedra of R_0 are of four types, see Fig 7. A tetrahedron of the first type, T_{vf}, has a grounded face (triangle) on a cutting plane. The tetrahedra of type T_{ee} have two grounded edges on two different cutting planes. The tetrahedra of type T_{evv} have a single grounded edge and intersects two other cutting planes at a single vertex. The tetrahedra of type T_{vvvv} have a vertex on four different cutting planes.

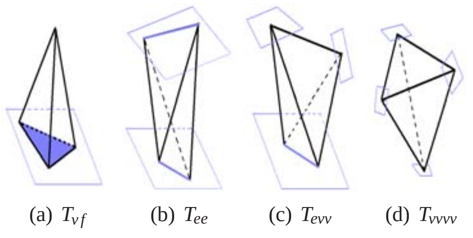

Figure 7: *Types of tetrahedra .*

Sampling condition on $\mathcal{A}_\mathcal{C}$, $\mathcal{B}_\mathcal{C}$ and $\mathcal{M}_\mathcal{C}$: We will assume the following condition to be satisfied, which can always be ensured by adding finitely many new vertices on the section contours and on the branching diagram [ET92].

Delaunay conformity condition: We assume that the edges of the section-contours are edges of R_0. Similarly, we assume that any two branching (or cutting) vertices that are consecutive along $\mathcal{B}_\mathcal{C}$ ($\mathcal{M}_\mathcal{C}$) are joined by an edge in R_0.

According to this condition, any grounded edge is either a contour-edge, an internal-edge, or an external-edge, and any grounded triangle is either internal or external. In the sequel, a tetrahedron of R_0 with an external grounded edge or triangle will be called *external*, see Fig 6c.

4. The Reconstruction Algorithm

We will prune the 3-dimensional Delaunay triangulation R_0, to obtain a complex which verifies the conformity and solidity conditions, Fig 6. The pruning of R_0 is done in three steps. We call the resulting subcomplexes R_1, R_2 and R_3 respectively. R_1 is the maximal subset of R_0 that conforms to the given sections. R_1 may not satisfy the solidity condition and further tetrahedra have to be removed until the solidity condition is verified. This procedure is done in two steps: in Step 2, we compute the maximal set of tetrahedra of types T_{vf} and T_{ee} (each linking two cutting planes), which verifies the solidity condition, see Section 4.2.3. Then in Step 3, we add to R_2, as many tetrahedra of type T_{evv} and T_{vvvv} as possible while maintaining the solidity condition.

4.1. Step 1 : Removal of the external tetrahedra

We remove from R_0 all the external tetrahedra . The resulting subcomplex of R_0 is noted R_1.

4.2. Step 2 : Solidity check around grounded edges

Let us make precise the solidity condition in this context. For this, we need the following definitions.

Definition 7 (Shell of a grounded edge (section-vertex)) For $R' \subset R_0$, the shell of a grounded edge (section-vertex) e is defined as the tetrahedra of R' which contain e. In the sequel, we write $Sh_{R'}(e)$ for the shell of e in R'. When the context is unambiguous, we may occasionally drop R' and write $Sh(e)$.

Definition 8 (Face-to-face connectivity) Two tetrahedra T and T', are said to be face-to-face connected in R', if there exists a sequence of tetrahedra $T = T_0, \ldots, T_k = T'$ such that for all i, $T_i \in R'$, and T_{i-1} and T_i share a face.

Definition 9 (Solidity at a grounded edge (section-vertex)) R' is solid at a grounded edge or section-vertex e, if any tetrahedron in $Sh(e)$ is face-to-face connected (in R') to at least one tetrahedron containing a grounded triangle incident to e. Such a shell is called *solid*.

The goal of this step is to prune R_1 to obtain a solid complex, whose tetrahedra are all incident to a pair of cutting planes. To consider the connections between the pairs of planes, we remove the tetrahedra of type T_{evv} and T_{vvvv}. The goal is now to prune around the grounded edges. We prune the incident tetrahedra of a grounded edge until we obtain a solid shell. Fig 8*b* shows all the possible configurations of a solid shell, resulting from Step 2.

Shell pruning description

Let T and T' be two tetrahedra incident to a grounded edge e. A *pivot* from T to T' around e is a sequence of face-to-face connected tetrahedra incident to e joining T to T', see Fig 9.

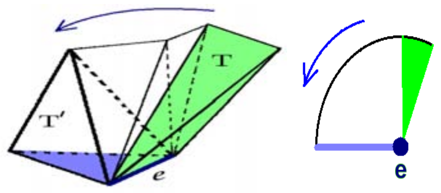

Figure 9: *A pivot from T to T' around e, that contains four tetrahedra . On the right side a 2D view of these tetrahedra .*

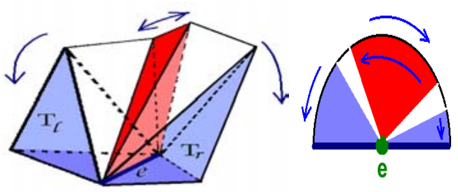

Figure 10: *A disconnected tetrahedron from the ground of e is between two eliminated tetrahedra . On the right side a 2D view of $Sh(e)$.*

During the pruning process, a tetrahedron $t \in Sh(e)$ is disconnected from the ground of e iff it lies between two eliminated tetrahedra incident to e, see Fig 8*a*. We compute these tetrahedra as follows. For an eliminated tetrahedron $T \in Sh(e)$, we start pivoting at T around e until we find another eliminated tetrahedron or a T_{vf} tetrahedron incident to e, called T'. The pivot sequence consists of T_{ee} tetrahedra all sharing edge e. To any such T_{ee} tetrahedron , we can associate a direction corresponding to the direction of the pivot. In our algorithm, we store this orientation as an orientation on the other grounded edge of the tetrahedron (distinct from e).

We apply the same procedure to all the eliminated tetrahedra of $Sh(e)$. A T_{ee} tetrahedron incident to e is disconnected from the ground of e if it is visited twice with different directions, Fig 10. Note that if during a pivot we reach a tetrahedron already oriented with the same orientation as the pivot, we can stop the propagation.

We are now in a position to describe Step 2 of the algorithm:

```
1. We put all the eliminated tetrahedra in a list $EL$.
2. **While** $EL \neq \emptyset$ do
    a. **For** a tetrahedron $T \in EL$ do
    b. **For** any grounded edge $e$ of $T$ do
        • Start pivoting at $T$ around $e$ and orient the visited
          tetrahedra until an eliminated or $T_{vf}$ tetrahedron is reached.
        • If a tetrahedron $t$ receives two opposite orientations,
          mark $t$ as eliminated and add it to $EL$.
    c. Remove $T$ from the list ($EL = EL \backslash T$).
3. Eliminate the face-to-face connected subsets of $T_{vf}$
   which share no face with the non-eliminated $T_{ee}$ tetrahedra .
4. $R_2$ is the set of non-eliminated tetrahedra .
```

4.2.1. Complexity of Step 2

Let s be the total number of tetrahedra in R_0. We claim that Step 2 can be performed in $O(s)$ time. Indeed, a tetrahedron of type T_{ee} has at most two pivoting edges and can receive at most two (opposite) orientations for each edge before being eliminated.

4.2.2. Solidity of R_2

R_2 is a set of tetrahedra of type T_{vf} and T_{ee}, which satisfies the following properties.

Property 1 (Solidity at grounded edges) Let e be a grounded edge. Any tetrahedron $t \in Sh_{R_2}(e)$ is face-to-face connected (in R_2) to the ground of e.

Proof. If t is a T_{vf} tetrahedron in R_2, it is not external and is it-self in the ground of e. Let t be a T_{ee} tetrahedron disconnected from the ground of e. Hence, t is between two eliminated tetrahedra and has been visited with two opposite directions. Thus, t does not belong to R_2. □

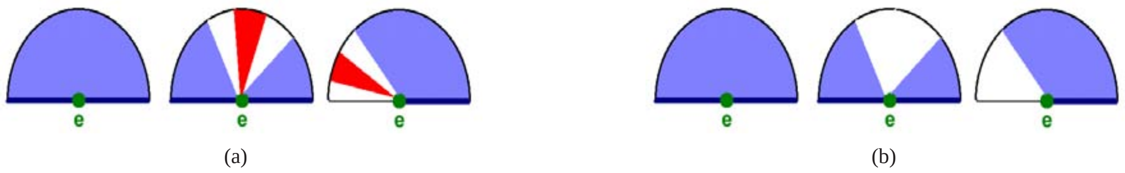

Figure 8: *A 2D view of the shell of a grounded edge e (e is perpendicular to the figure plane). (a) Disconnected tetrahedra from the ground are colored in red. They induce a singularity at e. (b) Configurations of a solid shell.*

Property 2 (Solidity at section-vertices) Let v be a section-vertex. Any tetrahedron $t \in Sh_{R_2}(v)$ is face-to-face connected (in R_2) to the ground of v.

Proof. Let t be a T_{ee} tetrahedron in R_2 which passes through an incident grounded edge of v. Thus, t is face-to-face connected to the ground of e and in particular to the ground of v.
Let t be a T_{vf} tetrahedron in R_2, with v as the fourth vertex (not in the plane of the grounded triangle of t). According to the algorithm, t is in a connected set of T_{vf} tetrahedra (with v as the fourth vertex), which shares a face with a T_{ee} tetrahedron in R_2. This T_{ee} tetrahedron is thus incident to a grounded edge e on the ground of v. Thus, t is face-to-face connected to the ground of e and in particular to the ground of v. □

4.2.3. Maximality of R_2

The following lemma shows that the order of elimination of the tetrahedra is not important.

Lemma 1 Let R' be subset of R_0. If T is a tetrahedron which creates a singularity in R', then T creates a singularity in any subset of R' containing T.

Proof. If t is disconnected from the ground of one of its grounded edges e, in R', then it is also disconnected from this ground in any subset of R'. □

In Step 2, the algorithm eliminates all the tetrahedra which create a singularity. According to the previous lemma, the elimination of some tetrahedron does not make disappear any other singularity, and so the order of elimination is not important. However, this elimination may make appear some other singularities in the resulting subset. Then, the algorithm removes these new singularities and so on until obtaining some subset of R_1 without singularity. Note that the number of the tetrahedra is finite and this process finishes. As at each step the order of the elimination is not important the resulting subset is maximal.

Proposition 1 R_2 is the maximal subset of T_{ee} and T_{vf} tetrahedra which has no singularity along the sections.

4.3. Step 3 : Connecting more than two planes

As said before, R_2 consists of Delaunay tetrahedra of type T_{ee} and T_{vf} exclusively. In this step, we add to R_2 as many tetrahedra of type T_{evv} and T_{vvvv} as possible provided that the only tetrahedra that can be added are those with exactly four, three or two faces in the current reconstruction. Clearly, as the intersection of such a tetrahedron with the current reconstruction is a topological disk or sphere, its insertion does not create singularities in the overall complex. Fig 11 provides some examples.

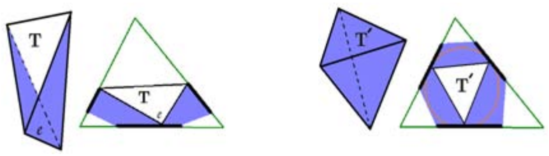

Figure 11: *Step 3 examples: T and T' do not create singularity and can be added to the reconstruction. A 2D view is also given.*

The algorithm maintains a list L of the tetrahedra of types T_{vvvv} and T_{evv} with such an intersection with the current reconstructed object. As long as this list is not empty, we add its first element in $\mathcal{R}$ and update L. We call R_3 the resulting complex which is , by construction, solid at any section-point and any grounded edge.

R_3 constitutes the reconstructed object (portion) $\mathcal{R}_\mathcal{C}$, corresponding to the cell of the arrangement $\mathcal{C}$. The overall reconstructed object is the union of these portions for all the cells of the arrangement.

5. Properties of the Reconstruction

Generality

There is no assumption on the position and orientation of the cutting planes neither on the geometry and topology of the sections. Complex section-contours with multiple branchings and holes can be handled, see Section 6 for some result examples.

Complexity

Let k be the number of cutting planes and n the total number of vertices, including the vertices of the section contours and the points on the branching diagram. Let C_i be a cell in the arrangement of the cutting planes and n_i the number of vertices in C_i, including the contour-vertices and the sample points on the branching diagrams of the faces of C_i. Assuming that the P_i are in general position, each point is on the boundary of $O(1)$ cells, and we have $\sum_i n_i = O(n)$.

Firstly, we compute the arrangement of k planes, which can be done in $O(k^3)$ time, see [E87]. In each cell C_i, we then compute the branching diagram of C_i by computing 2D Moebius diagrams in each face. Computing the 2D Moebius diagram of n points can be done in $O(n^2)$ time (see [BK03] and [BD05]), which is the same complexity as for computing the 3D Delaunay of n points in the worst case.
Once R_0 is computed, we eliminate the external tetrahedra . In this step, we have to visit each tetrahedron to decide if it is inside or outside the section-contours.
Since the size of 3D Delaunay is $O(n^2)$, Steps 2 and 3 can be done in $O(n^2)$ time. As each cell has at most k faces, the complexity of our algorithm is at most:

$$O(k^3) + \sum_{C_i}(k * Moeb_2(n_i) + DT_3(n_i) + O(n_i^2)) \leq \\ O(k^3) + k * \sum_{C_i} O(n_i^2) \leq O(k^3) + k * (\sum_{C_i} O(n_i))^2 \leq \\ O(k^3 + kn^2)$$

Consequently, for $k = O(1)$, the complexity is $O(n^2)$.

Solidity and Topological correctness

We showed that the reconstructed object $\mathcal{R}_\mathcal{C}$ in each cell $\mathcal{C}$ of the arrangement has no singularity in the sections.
As said before, the boundary of the $\mathcal{R}_\mathcal{C}$ contains the section-contours, the branching and cutting diagrams, and some flat regions. When we glue the various $\mathcal{R}_\mathcal{C}$ together, some singularity may appear on the boundary.
Consider a face f of the arrangement and the two incident cells $\mathcal{C}$ and $\mathcal{C}'$. If the branching diagrams of $\mathcal{C}$ and $\mathcal{C}'$ intersect in f then intersection points are singular, see Fig 12a. We may have a similar situation, when two cutting diagrams intersect. In general position, the intersection points of the branching or cutting diagrams are isolated. The overall object is therefore locally a manifold except at some isolated points.
Note that this problem can be handled by displacing the branching and cutting points slightly *inside* the corresponding cells, or by thickening the cutting planes, Fig 12b.

Conformity

By definition, the reconstructed object does not intersect the cutting planes outside the given sections. Conversely, it should be observed that $\mathcal{R}_\mathcal{C}$ does not necessarily contain all the grounded triangles of the sections. This particular situation occurs when a region of a section is flat in the two adjacent reconstructed portions corresponding to the two cells. This only happens when the set of cutting planes is very sparse, Fig 13.

Thickening the cutting planes could be another solution to handle these situations. In this case, the thin regions which appear in the overall reconstruction guarantee its conformity with the sections, see Fig 12c.

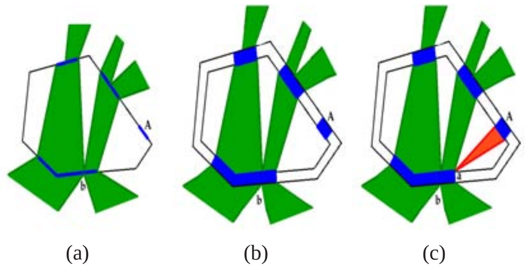

Figure 12: *(a) A singularity at point b which is on the intersection of the two branching diagrams, corresponding to the two adjacent cells. (b) This singularity disappears after thickening the cutting planes.(c) The flat region A gets also a thickness by this procedure. However, this thickening procedure does not make disappear the singularity at a.*

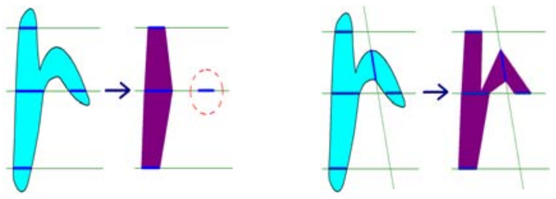

Figure 13: *A sufficiently dense sampling (right) of the cutting planes can guarantee that there is no flat region (left) in the overall reconstruction.*

6. Experimental Results

A preliminary version of the algorithm has been implemented in C++, using the [CGAL] library. The *input* of the algorithm consists of a cell of the arrangement (a convex polyhedron), and a set of polygonal contours lying on the boundary of this polyhedron. The contours are assumed to be Delaunay-conform and non self-intersecting, some possibly lying inside others. The contours are oriented in such a way that the interior of the section they bound is on the righthand side.

We present the results of the algorithm applied to a variety of input models. To gain more intuition, in addition to the 3D results, some figures of 2D reconstructions are also given. Fig 14 illustrates the reconstruction of a sphere cut by the faces of a cube. In this case, the reconstructed object is the convex hull of the contours.
Figure 15 shows how the method eliminates successfully

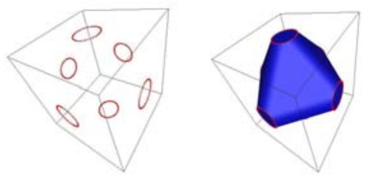

Figure 14: *The reconstructed object corresponding to a sphere, cut by the faces of a cube.*

the singularities of the Delaunay triangulation to guarantee the topological correctness of the reconstructed object.

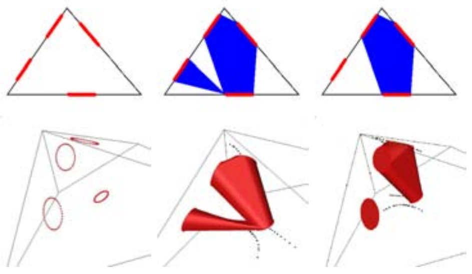

Figure 15: *Elimination of a singularity during the reconstruction: (Left) Input points. (Middle) Delaunay simplices of input points. (Right) Reconstructed object. A flat region appears in the reconstructed object.*

Some examples of branching between the sections are presented in Fig 16 and 17. These examples illustrate the role of the points added on the branching diagram. As we can see in Fig 16a, the branching diagram corresponds to the projection of the external medial axis of the sections. In the non-parallel case of planes, the branching is handled as well as the parallel case, see Fig 17.

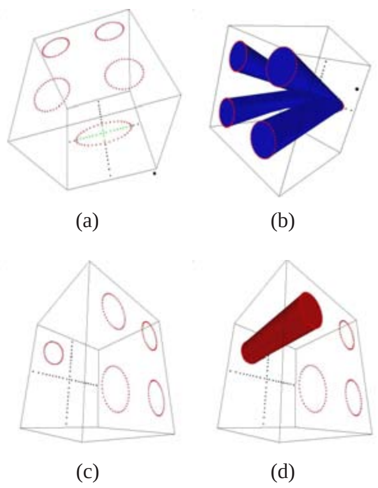

Figure 16: *(a) The single circle is at the same distance from all the other circles (b) There is a multiple branching between these four circles without any singularity. (c) As the circle is shifted, there is no branching anymore in the reconstructed object (d).*

The algorithm handles the complex branching of non-convex sections possibly with holes. The last figures provide some examples of these complex cases.

7. Conclusions

We have presented a method to reconstruct a volumetric model of an object given its intersection with a given set of unorganized cutting planes. The method is an extension of the work of Boissonnat and Geiger which was limited to the case of parallel sections [BG93]. The result is given as a subcomplex of the Delaunay triangulation of the union of two sets of points: the first one is the set of vertices of a polygonal approximation of the sections; the second set is a sample of points taken on two diagrams, called branching and cutting diagrams. These diagrams can be computed from the first set of points and the cutting planes. Interestingly, they are a variant of a non-euclidean Voronoi diagram known as Moebius diagram. The reconstructed object $\mathcal{R}$ is obtained

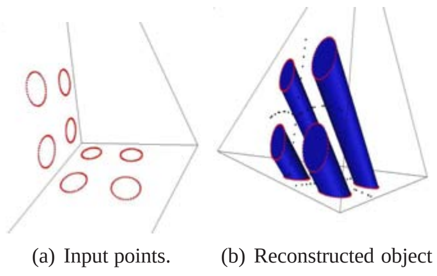

Figure 17: *An example with two non-parallel cutting planes.*

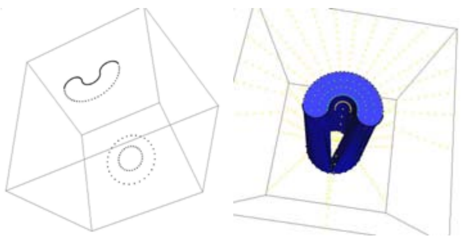

Figure 18: *A result for non-convex sections (given at left). There is no singularity in the reconstructed object (at right).*

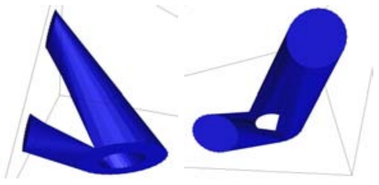

Figure 19: *A complex branching of a section with a hole.*

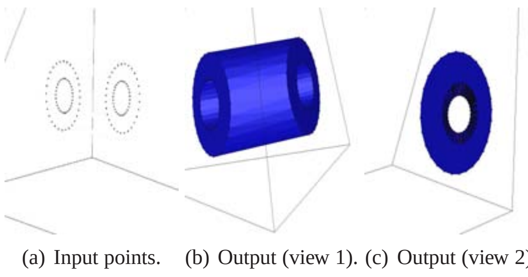

Figure 20: *An example with nested section-contours.*

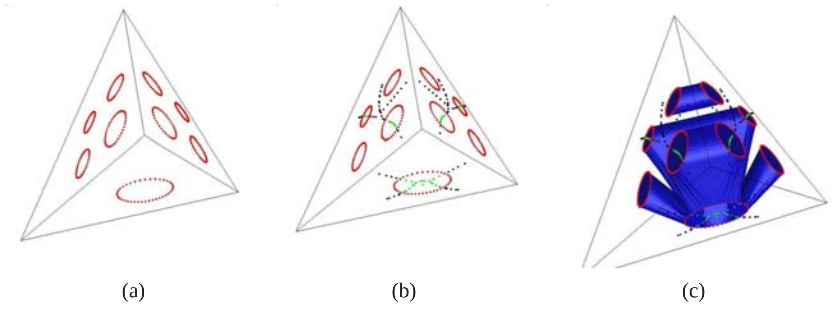

Figure 21: *Multiple branching between the sections:(a) Input points. (b) Input points in red and the added branching points in green. (c) Reconstructed object*

by pruning this Delaunay triangulation. We proved that $\mathcal{R}$ is a (triangulated) 3-manifold with boundary that conforms to the input sections. The overall construction has the same asymptotic complexity as the construction of the Delaunay triangulation.
Further questions we intend to consider in future are shape reconstruction from noisy cross-sections and reconstruction of moving objects.

Acknowledgments

We would like to thank an anonymous referee for providing us reference [DP97]. We acknowledge partial support from the EU Network of Excellence AIM@SHAPE (IST NoE No 506766) and from the High Council for Scientific and Technological Cooperation between France and Israel.

References

[Bo84] J.-D. Boissonnat, Geometric structures for three-dimensional shape representation, ACM Trans. Graph. , 4(3) (1984), 266-286.

[Bo88] J.-D. Boissonnat. Shape reconstruction from planar cross sections, Computer Vision, Graphics and Image Processing, 44 (1988).

[BG93] J.-D. Boissonnat, B. Geiger. Three dimensional reconstruction of complex shapes based on the Delaunay triangulation, Biomedical Image Processing and Biomedical Visualization, 964 (1993).

[BCDT96] J.-D. Boissonnat, A. Cérézo, O. Devillers, M. Teillaud, Output sensitive construction of the Delaunay triangulation of points lying in two planes. International Journal of Computational Geometry, Vol. 6 (1996).

[BK03] J.-D. Boissonnat, M. Karavelas, On the combinatorial complexity of euclidean Voronoi cells and convex hulls of d-dimensional spheres, 14th annual ACM-SIAM symposium on Discrete algorithms, 305-312 (2003).

[BD05] J.-D. Boissonnat, C. Delage, Voronoi Diagram of Additively Weighted Points, 13th European Symposium, (2005).

[BCL96] C.L. Bajaj, E.J. Coyle, and K.N. Lin, Arbitrary topology shape reconstruction from planar cross sections, Graphical Models and Image Processing, 58 (1996), 524-543.

[BGLS04] G. Barequet, M.T. Goodrich, A. Levi-Steiner, and D. Steiner, Contour Interpolation by straight skeletons, Graphical Models, 66 (2004), 245-260.

[BS96] G. Barequet and M. Sharir, Piecewise-linear interpolation between polygonal slices, Computer Vision and Image Understanding, 63 (1996), 251-272.

[CGAL] The Computational Geometry Algorithms Library (CGAL) Reference Manual, release 3.2.

[CS01a] N. Coll and J.A. Sellares. Planar shape reconstruction from random sections. Proc. 17th European Workshop on Computational Geometry, (2001), 121-124.

[DP97] C. R. Dance, R. W. Prager. Delaunay Reconstruction from Multiaxial Planar Cross-Sections, Technical Report, Cambridge University (1997), http://citeseer.ist.psu.edu/dance97delaunay.html.

[E87] H. Edelsbrunner. Algorithms in Combinatorial Geometry, Springer (1987).

[ET92] H. Edelsbrunner, T.S. Tan, An upper bound for conforming Delaunay triangulations, ACM Press NY, USA (1992).

[Ke75] E. Keppel, Approximating complex surfaces by triangulation of contour lines, IBM Journal of research and developement, 19 (1975), 2-11.

[PT94] B. A. Payne, A. W. Toga. Surface Reconstruction by Multiaxial Triangulation, IEEE Comput. Graph. Appl. 14, 6 (1994), 28-35.

[SBG06] A. Sidlesky, G. and Barequet, G. and C. Gotsman, Polygon Reconstruction from Line Cross-Sections, Canadian Conf. on Computational Geometry (2006).