Strong Thinning and Polyhedrization of the Surface of a Voxel Object*

Jasmine Burguet and Rémy Malgouyres

GREYC, ISMRA
6 bd Maréchal Juin 14000 Caen, France
{Burguet,Remy.Malgouyres}@llaic.u-clermont1.fr

Abstract. We first propose for digital surfaces a notion analogous to the notion of strong homotopy which exists in 3D [1]. We present an associated parallel thinning algorithm. The surface of an object composed of voxels is a set of surfels (faces of voxels) which is the boundary between this object and its complementary. But this representation is not the classical one to visualize and to work on 3D objects, in frameworks like Computer Assisted Geometric Design (CAGD). For this reason we propose a method for passing efficiently from a representation to the other. More precisely, we present a three-step algorithm to polyhedrize the boundary of a voxel object which uses the parallel thinning algorithm presented above. This method is specifically adapted to digital objects and is much more efficient than such existing methods [12]. Some examples are shown, and a method to make the reverse operation (discretization) is briefly presented.

Keywords: digital surface, thinning, strong homotopy, parallel algorithm, polyhedrization.

Introduction

To study a three-dimensional object, its surface is often used because it is less costy (in particular concerning the memory cost) than studying the whole 3D volume [2,3,4,7,8]. For this reason, the discrete representation of digital surfaces of 3D objects is of practical interest. Indeed, it allows us to compute easily the union, intersection and difference (i.e. boolean set operations) of two discrete objects represented by their surfaces. On the other hand, classical modeling techniques use a continuous representation for surfaces. More precisely, to encode surfaces, a representation is often used to vizualize and to work on 3D objects: polyhedrons. This representation has a lot of advantages, particulary in Computer Assisted Geometric Design (CAGD) applications. However, floating points computations are much less relevant to perform boolean set operations. Indeed, there are some problems when the objects we want to work on are tangeant to

* The authors' new permanent address is: LLAIC1 - IUT Département Informatique, BP 86, 63172 AUBIERE Cédex

G. Borgefors, I. Nyström, and G. Sanniti di Baja (Eds.): DGCI 2000, LNCS 1953, pp. 222–234, 2000.

each other. This leads us to find a way to pass efficently and quickly from a discrete representation to a polyhedron representation, and conversely.

The existing methods to polyhedrize the surfaces, except in [12], are not especially adapted to our initial data structure for the discrete surfaces, namely a set of surfels.

The purpose of this paper is to introduce an algorithm which enables us to compute a *polyhedrized surface* from a *discrete surface*.

The classical methods, using for exemple the Voronoï diagram and the Delaunay triangulation [17], are not effective for the non-euclidien spaces like surfaces. So, we use a topological approach for the discrete surfaces (see [5] and [6]). This leads us to define a "topological Voronoí diagram" on these surfaces which allows us to built the triangular faces of our polyhedrized surface.

In the first part, we present a parallel thinning algorithm to compute the skeleton of a set of surfels. This algorithm is based on an analogue for the surfaces of the notion of *strong homotopy* existing in 3D and introduced by G. Bertrand ([1,13]).

Then, we present a polyhedrization method which is based on the use of a germ-obtained skeleton. The germs are particular surfels chosen, with respect to the maximal curvature, in different locations of our surface; the higher is the curvature, the more numerous are the germs. Then, using the parallel algorithm previously presented, we compute the skeleton of the set of surfels composed of our surface from which the germs are removed. The obtained skeleton enables us to draw triangles which will be the faces of the polyhedrized surface. These triangles are built during a cover of the skeleton. Since the germs are more numerous where the surface is very curved, the number of faces is higher in such locations. Some examples of the polyhedrization algorithm results are given.

The presented method, which can also be used for (non conservative) data compression of binary 3D images, is much faster than the method introduced in [12]. Moreover, it produces more regular faces, which are all triangles, which makes it much more convenient for CAGD applications.

A method to make the reverse operation (discretization) is briefly presented. It is based on a particular data structure: an array the cells of which contain chained lists of surfels. Note that the boolean set operations can easily be performed using this data structure.

1 Notations and Basic Notions

Let X and E be two sets such as $X \subset E$. We denote by $\overline{X}$ the complement of X in E and by $card(X)$ the number of elements of X. In the sequel, an *adjacency relation* on X is an antireflexive symmetric binary relation on X. Given α an adjacency relation, an *α-path* in X with length l is a sequence $(x_0, x_1, \ldots, x_l)$ in X and such as x_{i-1} is α-adjacent to x_i, for $i = 1, \ldots, l$. An α-path is called *closed* if $x_0 = x_l$ and *simple* if the x_i are pairwise distinct. Two elements y and y' of X are said to be α-connected in X if there exists an α-path $(x_0, x_1, \ldots, x_l)$ contained in X such as $y = x_0$ and $y' = x_l$. The α-connectedness is an equivalence

relation and its equivalence classes are called α-*connected components*. We denote by $C_{alpha}(X)$ the set of connected components of X and X is said to be α-*connected* if $card(C_\alpha(X)) = 1$.

1.1 Surface of a 3D Object

A 3D object O is a set of points in $\mathbb{Z}^3$. These points, which we can represent as unit cubes centered at a point $(i, j, k) \in \mathbb{Z}^3$, are called *voxels*. Two voxels are said to be *6-adjacent* if they share a face, *18-adjacent* if they share a face or an edge. We derive from this as above the classical notions of an *n-path* and *n-connected component*, with $n \in \{6, 18\}$, and we denote by $C_n(O)$ the set of n-connected components of the object O. We define the n-neighborhood of a voxel x denoted by $N_n(x)$ by $N_n(x) = \{y \in \mathbb{Z}^3 / y$ is n-adjacent to $x\}$.

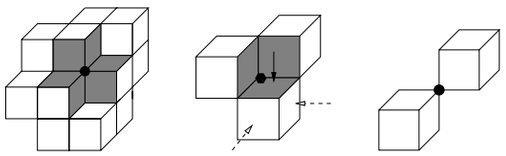


Fig. 1. Examples

Now, a *surfel* is a couple (v_1, v_2) of 6-adjacent voxels v_1 and v_2. A surfel may be seen as an unit square shared by two 6-adjacent voxels, and then we call *surface* of an object $O \subset \mathbb{Z}^3$ the set of all surfels (v_1,v_2) such as $v_1 \in O$ and $v_2 \in \overline{O}$. We can define adjacency relations between surfels, and these relations depend on the one we consider on voxels. We consider the surface Σ of an object O. Now, we call *e-adjacency* the adjacency relation on surfels considering the edges. Under this one, a surfel has exactly four neighbors, one per edge. For example, let call x the grey surfel pointed by a full arrow on the object of the figure 1, (b). If we consider the 6-adjacency, the object is composed of three 6-connected sets (the three voxels), and the e-neighbors of x are the four surfels of the lower voxel which share an edge with x. By contrast, if we consider the object with the 18-adjacency, the three voxels are 18-connected and the four e- neighbors of x are the two other grey surfels and the two surfels pointed by the dotted arrow. We are now going to define an adjacency relation associated with the vertices of the surfels. First, we define a *loop* in Σ as an e-connected component of the set of all surfels in Σ which share a given vertex (figure 1). One vertex may define two loops (see figure 1, b) and c)), so the loops are a way to duplicate formaly vertices. Thus, two surfels are called *v-adjacent* if they belong to a common loop. Then, e and v are adjacency relations on Σ from which follow the classic notions of *n-paths*, *n-neighborhoods*, and *e-connected components*, with

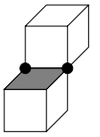


Fig. 2. Counter-example

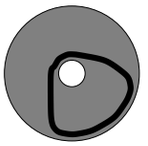


Fig. 3. A decentred skeleton

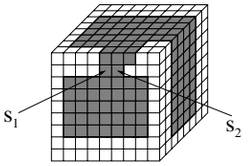


Fig. 4. No parallel strategy

$n \in \{e, v\}$. The kind of surfaces defined above satisfies the Jordan property [11], namely a e-connected component of a surface separates the space into two parts: a 6-connected and a 18-connected one [16].

Afterwards, we will assume that each loop of our surface is a topological disk. In other words, two v-adjacent surfels which are not e-adjacent cannot belong simultaneously to two distinct loops. The object of the figure 2 does not satisfy our hypothesis because the two loops defined by the marked vertices contain surfels which are v-adjacent but not e-adjacent. (see [9] for a short discution on this hypothesis).

1.2 Neighborhood Graph of Surfels

Let $x \in \Sigma$, $n \in \{e, v\}$ and $X \subset \Sigma$ be a set of surfels. We introduce the *neighborhood graph of surfels* of x as follows. We have assumed that each loop in Σ is a topologic disk, but there exist some v-neighborhoods $N_v(x)$ which are not topological disks (see the surfels of the two loops defined by the marked vertex on figure 1 (b). Then we have to define an other "topology" under which $N_v(x) \cup \{x\}$ is a topological disk. Two surfels s_1 and s_2 in $N_v(x) \cup \{x\}$ are said to be *e_x-adjacent* (respectively *v_x-adjacent*) if they are e-adjacent (respectively *v-adjacent*) and are contained in a common loop which contains x. Then, we denote by *$G_e(x,X)$* (respectively *$G_v(x,X)$*) the graph whose vertices are the surfels of $N_v(x) \cap X$ and whose edges are the pairs of e_x-adjacent (respectively v_x-adjacent) surfels of $N_v(x) \cap X$. The set of all connected components of *$G_n(x,X)$* which are n-adjacent to x is denoted by $C_n^x(G_n(x, X))$. Note that $C_n^x(G_n(x, X))$ is a set of sets of surfels and not a set of surfels.

A surfel x is said to be *n-isolated* in X if $N_n(x) \cap X = \emptyset$ and to be *n-interior* in X if $N_{\overline{n}}(x) \cap \overline{X} = \emptyset$.

2 Topology Preservation and Thinning

2.1 P-Simple Surfels

First, we introduce the notion of *n-simpleness* of surfels [10]:

Definition 1. *[9]. Let $X \subset \Sigma$ be a set of surfels of a digital surface Σ. A surfel $x \in X$ is said to be* n-simple *in X if and only if x is not n-interior in X and if $card(C_n^x(G_n(x, X)) = 1$.*

Intuitively, a surfel is n-simple in X if its deletion from X does not change the topology of X.

Definition 2. *Let $Y \subset X \subset \Sigma$. The set Y is said to be (lower) n-homotopic to X if and only if Y can be obtained from X by sequential deletion of n-simple surfels.*

Thus, we can obtain a skeleton from a set of surfels by a sequential erosion analogous to classical 2D sequential thining [10,9]. As in the planar 2D case [10], this method presents some drawbacks : there is no parallel way to test the n-simplicity of surfels (on figure 4, the surfels s_1 and s_2 of the set X composed of grey surfels are n-simple in X, but we cannot remove simultaneously these surfels without changing the topology). Moreover, the shape of the skeleton depends on the order with which the n-simplicity of surfels is tested (see figure 3 if we use the lexicographical order to test the n-simplicity of surfels). Finally, the strategy using end surfels (surfels which have only one n-neighbor) causes the apparition of a lot of undesirable branches (see figure 6).

For these reasons, we are now interested in a new thinning method of sets of surfels using a parallel strategy. This method is an analogue of a parallel thinning algorithm existing in 3D introduced by G. Bertrand [1], and using the notion of *P-simple points*.

First we must state some theoretical definitions and results.

Definition 3. *Let $Y \subset X \subset \Sigma$. The set Y is called* strongly n-homotopic to X *if for any Z such that $Y \subset Z \subset X$, the set Z is n-homotopic to X. If Y is strongly n-homotopic to X, then we call the set $S = X \backslash Y$ a* strongly n-simple set *in X.*

Note that if Y is strongly n-homotopic to X, then for any set $S \subset X \backslash Y$, the set $X \backslash S$ is always n-homotopic to X.

Definition 4. *Let $X \subset \Sigma$, $P \subset X$ and $x \in X$. The surfel x is called P_n-simple in X if, for any set S such that $S \subset P \backslash \{x\}$, the surfels x is n-simple for $X \backslash S$. We denote by $S_n(P)$ the sets of the P_n-simple surfels. A set $E \subset X$ is said* P_n-simple *in X if $E \subset S_n(P)$.*

The P_n-simple surfels satisfy a strong constraint, namely a P_n-simple surfel can be removed whatever the set contained in P already removed. Now, we establish a relationship between the notions of strongly n-simplicity and P_n-simplicity which is directly deduced from the definitions.

Theorem 1. *Let $Y \subset X \subset \Sigma$. The set Y is strongly lower n-homotopic to X if and only if the set $P = X \backslash Y$ is P_n-simple for X.*

Now, we propose a local caracterization of the P_n-simple surfels, which makes effective the notion of a P_n-simple surfel.

Definition 5. *We denote by $A_n(x, X)$ the set of surfels defined by*

$$y \in A_n(x, X) \Leftrightarrow \exists \text{ a } n_x\text{-path in } N_v(x) \cap X \text{ from } y \text{ to an } n\text{-neighbor of } x$$

and we denote by $T_n(x, X)$ the number of n_x-connected components of $A_n(x, X)$.

In other words, $A_n(x, X)$ is the union of all connected components of $G_n(x, X)$ which are n-adjacent to x.

Theorem 2. *Let $P \subset X \subset \Sigma$ and $x \in P$. We denote $R = X \backslash P$. Then the surfel x is P_n-simple in X if and only if the four following properties are satisfied:*

- $T_n(x, R) = 1$;
- $T_{\overline{n}}(x, \overline{X}) = 1$;
- $\forall y \in N_n(x) \cap P$, y *is n-adjacent to* $A_n(x, R)$;
- $\forall y \in N_{\overline{n}}(x) \cap P$, y *is $\overline{n}$-adjacent to* $A_{\overline{n}}(x, \overline{X})$.

This theorem enables us to test the P_n-simpleness of surfels by an algorithm using only local considerations, and enables us to define an algorithm using a parallel strategy.

2.2 Parallel Algorithm and Results

First, we present a sequential thinning algorithm of a subset X of a surface Σ, implemented by A. Lenoir [9,4]. The two-step principle is the following one: while there exist some n-simple surfels in X,

- for all x in X, if x is n-simple, mark x with REMOVABLE;
- for all surfels in X, if x is marked with REMOVABLE, and x is n-simple in the remaining set, remove x from X.

To preserve the general shape of our initial set of surfels, we impose on the end surfels to be unremovable (i.e. surfels of X with exactly one neighbor in X).

Now let us present our parallel algorithm (see figure 5). Only the surfels $\overline{n}$-adjacent to X can be be removed from a set X at each step. So, to test the P_n-simplicity of the surfels in X, our set P will be the set of the surfels in X which are $\overline{n}$-adjacent to $\overline{X}$. Now, a surfel which is not P_n-simple may be n-simple. So, after using the parallel algorithm, we will use the sequential one to remove all n-simple surfels which are not end surfels. The parallel algorithm is presented on figure 5.

The results of this algorithm (see figure 6) are, as in the 3D case, quite better than the ones obtained by the sequential algorithm. First, there are less undesirable branches on the final squeleton obtained by the parallel algorithm. Next, the symmetry of this skeleton is closer to the symmetry of the initial set of surfels.

Parallel Algorithm

```
M = array of all surfels;
P = set of the surfels on the border;
R = set of the interior surfels;
Repeat
    made := FALSE;
    For all x in P
            If (x is P-simple)
              Then mark x with REMOVABLE;
                   made := TRUE;
    End For
    For all x in P
            If (x is marked with REMOVABLE)
               Then remove x from M;
    End For
    Update P;
While made;
```

Fig. 5. Parallel algorithm

3 Polyhedrization

The method of polyhedrization presented here consists of three steps.

3.1 Step One: The Choice of the Germs

The first step consists in choosing some special surfels on the surface of the object we want to polyhedrize. These surfels, called *germs*, are distributed according to the curvature of the surface: the more curved is the surface, the more numerous are the germs. First, using the method of A. Lenoir [3,4], we compute the maximal curvature of all the surfels of the surface, and we choose a surfel at random, which is our first germ, on the surface. Then:

1. determination of a maximal width l depending on the maximal curvature of the surfel;
2. breadth-first exploration (using a FIFO structure) of the e-adjacency graph from the germ until the width l is reached; we mark the covered surfels;
3. if there are some non-marked surfels, choice of a new germ among the non-marked neighbors of the marked surfels, and back to the step 1. Otherwise, STOP.

At the third step, for better results, the next germ is the one which has the greater maximal curvature. Indeed, with this choice, we obtain more faces at the more curvated places.

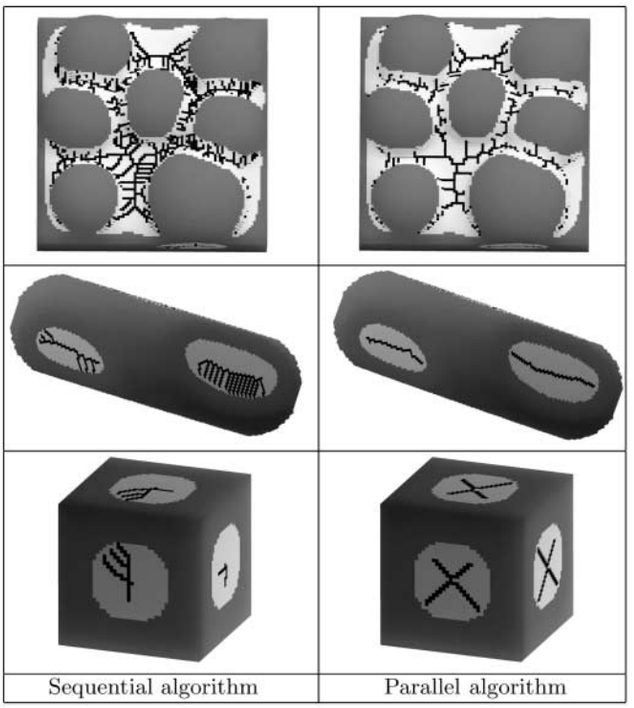


Fig. 6. Comparison of results

An interesting point is that we can choose a limit width L for the breadth-first exploration: if the computed width l is larger than L, then we replace l by L. Moreover, we use a slowdown strategy at the more curved places during the breadth-first exploration. So we can have germs more or less dense on the less curved places of the surface.

We denote by G the set of all germs thus obtained. Some results of this strategy are presented on figure 7.

3.2 Step 2: Obtaining of a "Topological Voronoï Diagram"

Let Σ be the surface of the object. To obtain the "topological Voronoï diagram" (i.e. a skeleton in which there are no more simple surfels), we use the thinning algorithm presented in the first section on the set $\Sigma \backslash G$, without the end surfels conditions. We obtain the skeletons visible on figure 8 for the three previous presented objects.

The skeletons are obviously more dense where the germs are more numerous.

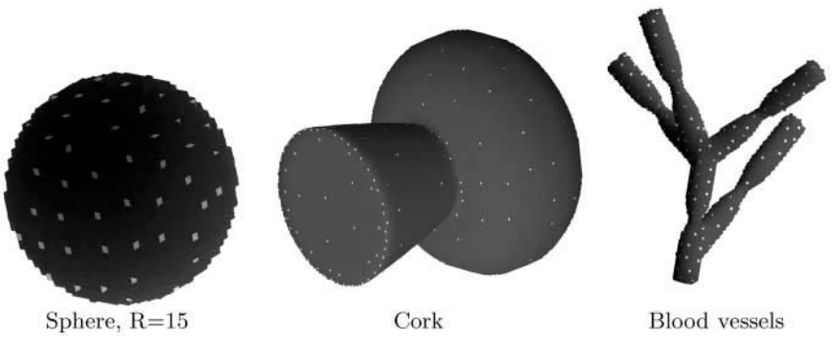


Fig. 7. The choosen germs (in white) on the surfaces of three voxels objects

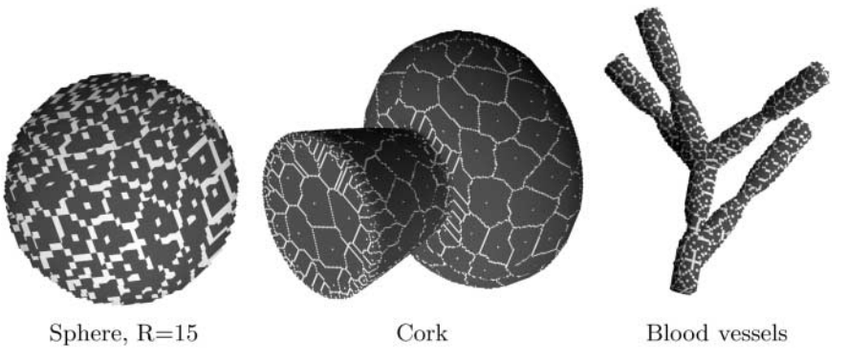
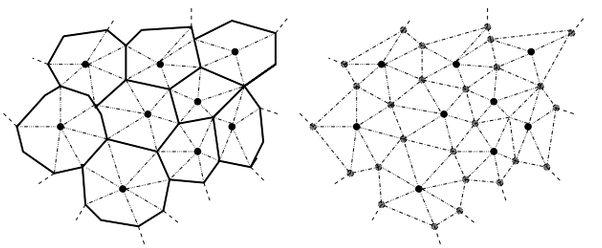


Fig. 8. The chosen germs and the "topological Voronoï diagrams"

3.3 Step 3: The Obtaining of the Faces

Now, from the skeleton and the germs, we are going to cover the triangular faces which will form our polyhedrized surface.

Let consider the figure 9, (a). The germs are represented by the black points, and the squeleton by the thick black lines. There exist two kinds of surfels on the skeleton: the surfels of the branches (surfels which have exactly two neighbors in the skeleton) and the surfels of the intersections (surfels which have strictly more than two neighbors). So, we cover the skeleton by covering of the branches from an intersection to an other one. Let I_1 and I_2 be the barycenters of two intersections connected by a branch. The segment I_1I_2 is the edge of two triangles, and the other vertices of these triangles are the germs which the branch separates (see figure 9, (b)).



(a) (b)

Fig. 9. Obtaining of triangular faces from the skeleton

3.4 Results

The results of the algorithm for the sphere with radius 15, the cork and the blood vessels are presented on figure 10. The initial numbers of surfels of these surfaces are respectively 4254, 81066, 10792 and the chosen limit widths L 4, 8 and 4. The numbers of triangular faces obtained by the algorithm are 504, 3914 and 1316. It takes a few seconds to compute the polyhedrized surfaces (for the bigger example, the cork, it takes about 30 seconds on Ultra Sparc Sun^{tm} work stations with bi-processors).

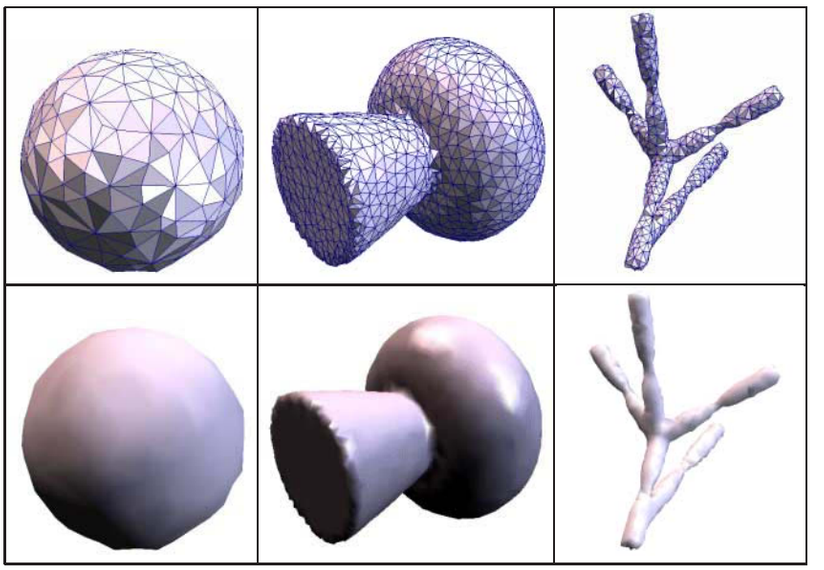


Fig. 10. The polyhedrized surfaces

As we can see on the figure, these faces are regular, well located and one gets good quality surfaces, even on thin objects (see the result on the blood vessels). The worst results are near the angles (see base of the cork), but they remain satisfying.

4 Discretization

Now, we have an effective tool to obtain a polyhedrized surface from a discrete one. Here, we present briefly a method to realize the reverse operation.

First, we choose a step of discretization P (smaller than the size of the surface S we want to discretize). Then, we are going to work according to the first axis of the classical coordinates space. We define a 2-dimensionnal array $(kP) * (kP)$, with $k \in \mathbb{N}*$ according to the plane containing the second and the third axis. Each cell of this array contains a chained list of surfels (figure 11). The method to fill up this array is to study the intersection between the line D_{ij} (the line coming from the cell (i, j) and parallel to the first axis) and S (figure 12). During the recover of D_{ij}, for each intersection we have to determine by a parity rule if we enter into the surface or if we get out. Then, we can compute the coordinates of the deduced surfels and obtain the discretized surface.

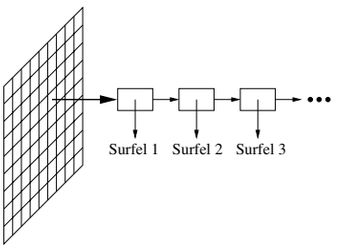


Fig. 11. Data structure

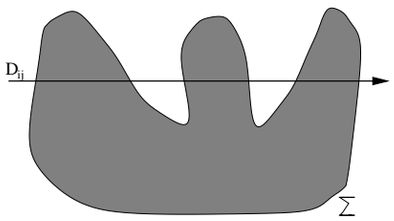


Fig. 12. Method

5 Conclusions and Perspectives

Following section 2, we have obtained an analogue of the notion of P_n-simplicity existing in 3D for discrete surfaces. Moreover, the results of thinning using the parallel strategy are far better than the ones deriving from the sequential method.

Next, as for the polyhedrization, the obtained surfaces are rather satisfying. The computing time for results is quite short (a few seconds against several days for the method of [12]) and the obtained faces are much more regular, in comparison to what would be obtained by any other pre-existing method [12].

Note that the boolean set operations between the objects surrounded by the discretized surface can easily be performed using the obtained discrete data structure.

In future works, we shall include this work into a *multi-scale* context. Indeed, if we have to work with a set of multi-scaled objects, it is more relevant to consider different scales related to these objects. Thus, we will be able to represent the whole multi-scaled set with one method.

Acknowledgment

We thank A. Lenoir for providing us the source code of his computer program, including curvature computation and display for digital surfaces.

References

1. G. BERTRAND. *On P-simple points*, no. 321, C.R Académie des Sciences, 1995. 222, 223, 226
2. T. J. FAN, G. MEDIONI, R. NEVATA. *Recognising 3D Objects Using Surface Description*, IEEE Trans. Pattern Anal. Mach. Intelligence **11**(11),1140-1157, 1989. 222
3. A. LENOIR. *Fast Estimation of Mean Curvature on the Surface of a 3D Discrete Object*, Proceedings of DGCI'97, Lecture Notes in Computer Science, Vol. 1347, pp. 213-222, 1997. 222, 228
4. A. LENOIR. *Des Outils pour les Surfaces Discrètes*, PhD Thesis, 1999. 222, 227, 228
5. R. MALGOUYRES. *Presentation of the fundamental group in digital surfaces*, Proceedings of DGCI'99, Lecture Notes in Computer Science, Vol. 1568, pp. 136-150, March 1999. 223
6. S. FOUREY, R. MALGOUYRES. *Intersection number of paths lying on a digital surface and a new Jordan theorem*, Proceedings of DGCI'99, Lecture Notes in Computer Science, Vol. 1568, pp. 104-117, March 1999. 223
7. T. J. FAN, G. MEDIONI, R. NEVATA. *Recognising 3D Objects Using Surface Descriptions*, IEEE Trans. Pattern Anal. Mach. Intelligence 11(11), pp. 1140-1157, 1989. 222
8. G. FARIN. *Computer Assisted Geometric Design (CGAD)*. Academic Press, Inc. 222
9. R. MALGOUYRES, A. LENOIR. *Topology Preservation Within Digital Surfaces* Graphical Models (GMIP) 62, 71-84, 2000. 225, 226, 227
10. T. Y. KONG, A. ROSENFELD. *Digital Topology: Introduction and Survey*. Computer Vision, Graphics, and Image Processing 48, 357-393, 1989. 226
11. G. T. HERMAN. *Discrete Multidimensional Jordan Surfaces*, CVGIP: Graph. Models Image Process. 54, 507-515, 1992. 225
12. J. FRANÇON, L. PAPIER. *Polyhedrization of the Boundary of a Voxel Object*, Proceedings of DGCI'99, Lecture Notes in Computer Science, Vol. 1568, pp. 425-434, Mars 1999. 222, 223, 232
13. G. BERTRAND. *Simple points, topological numbers and geodesic neighborhoods in cubics grids*, Patterns Recognition Letters, 15:1003-1011, 1994. 223

14. R. MALGOUYRES. *Homotopy in 2-Dimensional Digital Images* Theoretical Computer Science. Vol. 230, pp. 221-233, 2000.
15. R. MALGOUYRES, S. FOUREY. *Intersection Number and Topology Preservation Within Digital Surfaces*, Proceedings of DGCI'99, Lecture Notes in Computer Science, Vol. 1568, pp. 104-117, Mars 1999.
16. J. K. UDUPA, V. G. AJJANAGADDE. *Boundary and Object Labelling in Three-Dimensonal Images*, Computer Vision, Graphics, and Image Processing 51, 355-369, 1990. 225
17. R. KLEIN. *Concrete and Abstract Voronoï Diagrams* . Lecture Notes in Computer Science, Springer Verlag Ed., 1989. 223