# Meshing implicit algebraic surfaces: the smooth case

L. Alberti* & G. Comte° & B. Mourrain*

* GALAAD, INRIA
BP 93
06902 Sophia Antipolis
France

° Lab. J. A. Dieudonné
Parc Valrose
06034 Nice cedex
France

**Abstract.** In this paper, we present a new algorithm to mesh surfaces defined by an implicit equation, which is able to isolate the singular points of the surface, to guaranty the topology in the smooth part, while producing a number of triangles which is related to geometric invariants of the surface. We prove its termination and correctness and give complexity bounds, based on metric entropy analysis. The method applies to surfaces defined by a polynomial equation or a spline equation. We use Bernstein bases to represent the function in a box and subdivide this representation according to a generalization of Descartes rule, until the problem in each box boils down to the case where either the implicit object is isotopic to its linear approximation in the cell or the size of the cell is smaller than a parameter $\varepsilon$. This ensures that the topology of the implicit surface is caught within a precision $\varepsilon$. Experimentations on classical examples from the classification of singularities show the efficiency of the approach.

## §1. Introduction

Several methods have been developed over the last decades to visualize or to mesh an implicit surface. We mention in particular the following approaches:

- ray tracing,


xxx and xxx (eds.), pp. 1–4.

ISBN 0-9728482-x-x

- marching cube,
- marching polygonizer,
- deformation methods,
- subdivision methods.

Used to detect the visibility of objects, "ray tracing" methods [16] compute the intersection between the ray from the eye of the observer and the first object of the scene, for each pixel of the image. The rendering is very good, but the computing time is significant. It depends on the resolution of the scene to be viewed, and can produce only images in 2 dimensions. Moreover, computation cannot (a priori) be re-used for other views. Other techniques, such as particle sampling, also use clouds of points lying on the surface, to visualize it. See for instance [2]. However, it only yields an approximation of the surface [28] without the topological structure, nor with guaranty on the result.

The "marching cube" algorithm [19] developed in order to reconstruct images in 3 dimensions starting from medical data, is very much used for visualization of level sets of functions. The principle of this algorithm is simple: the domain of interest is divided in several cells, generally boxes, of the same size. At the corners of each cell, the values of the function $f$ are calculated and a triangular mesh is then deduced according to the sign of the function at these corners. This triangulation may not capture the topology of the surface, if it is not supported by additional calculation. Several triangulations are possible for the same combination of signs. Some partial solutions exist to avoid some of these ambiguities [18]. The covering of all the space of study increases the computing time considerably. Indeed the boxes not cut by the surface are not useful. Despite its defects, the "marching cube" methods remains a reference for its simplicity and its easiness of adaptation.

The marching polygonizer method brings a significant improvement to the marching cube method. The principal idea of this method is to calculate only the "useful" cells, that is, those which cut the surface. The algorithm starts from a valid cube (or tetrahedron), and propagate toward the connected cells, which cut the surface [5], [6], [14], [1]. Thus it is necessary to start from a cell intersecting the surface. Again, for self intersecting surfaces or surfaces with singularities, the result might be erroneous. The algorithm is rather effective, but it does not make it possible to mesh any surface correctly, if this one has, for example, several connected components, without using external tools such as "topological skeleton " [4].

Deformation methods exploit results from Morse theory, in order to correctly follow the transformation of the level-sets $f(x) = t$ [1]. See also [7] for a connected approach, which applies for smooth surfaces. These

methods assume implicitly that the function is a Morse function, ie. the critical points that are traversed during the deformation are not degenerate. This is not always the case nor is straightforward to check.

We describe here a new subdivision algorithm to mesh an implicit surface, which is able to isolate the singular points of the surface, to guaranty the topology in the smooth part, while producing a number of linear pieces, related to the Vitushkin variations of the surface. It applies to surfaces defined by a polynomial equation or a B-spline equation. Our method has similarities with the one presented in [22], but we go further by describing a new and guaranteed subdivision criterion. Moreover, we analyze in detail the complexity of the subdivision algorithm in terms of the entropy of the surface, which yields a bound on the number of cells produced by the method in terms of geometric invariants of the surface. The termination and correctness of the algorithm are proved, in the case of a smooth surface. Its extension for the treatment of singularities, using a local conic structure theorem, is briefly mentioned, but the extended details of this treatment are postponed to another paper.

Regarding the technical aspects, we use Bernstein bases to represent the function in a box and subdivide this representation according to a generalization of Descartes' rule, until the problem in each box boils down to the case where either the implicit object is proved to be homeomorph to the computed linear approximation in the cell or the size of the cell is smaller than $\epsilon$. This ensures that the topology of the implicit surface is caught within a precision $\epsilon$, where $\epsilon$ is a tunable parameter. Experimentations on classical examples from the classification of singularities show the efficiency of the approach.

## §2. Algebraic ingredients

For any point $p \in \mathbb{R}^3$, and any set $A \subset \mathbb{R}^3$, $\mathrm{dist}(p, A)$ denotes the minimal Euclidean distance between $p$ and points $q \in A$. We define $\mathrm{dist}_x(p, A)$ as the minimal Euclidean distance between $p$ and a point $q \in A$ with the same $(y, z)$-coordinates, if it exists and $+\infty$ otherwise. The distances $\mathrm{dist}_y(p,\ \mathrm{A})$, $\mathrm{dist}_z(p, A)$ are defined similarly.

### 2.1. Representation of polynomials

Let us recall that a univariate polynomial $f(x)$ of degree $d$ can be represented in the Bernstein basis by:

$$f(x) = \sum_{i=0}^{d} b_i\ B_d^i(x),$$

where $B_d^i(x) = \binom{d}{i} x^i (1-x)^{d-i}$. The sequence $\mathbf{b} = [b_i]_{i=0,\ldots,d}$ is called the set of control coefficients on $[0, 1]$. The polynomials $B_d^i$ form the Bernstein

basis on $[0,1]$. Similarly, we will say that a sequence $\mathbf{b}$ represents the polynomial $f$ on the interval $[a,b]$ if:

$$f(x) = \sum_{i=0}^{d} b_i \binom{d}{i} \frac{1}{(b-a)^n} (x-a)^i (b-x)^{d-i}.$$

The polynomials

$$B_d^i(x;a,b) := \binom{d}{i} \frac{1}{(b-a)^n} (x-a)^i (b-x)^{d-i}$$

form the Bernstein basis on $[a,b]$. Hereafter, we are going to consider the sequence of values $\mathbf{b}$ together with the corresponding interval $[a,b]$. A first property of this representation is that the derivative $f'$ of $f$, is represented by the control coefficients:

$$d\Delta \boldsymbol{b} := d(b_{i+1} - b_i)_{0 \leqslant i \leqslant d-1}.$$

Another fundamental algorithm that we will use on such a representation is the de Casteljau algorithm [10]:

$$\begin{aligned} b_i^0 &= b_i & i &= 0, \ldots, d \\ b_i^r &= (1-t)\, b_i^{r-1} + t\, b_{i+1}^{r-1}(t) & i &= 0, \ldots, d-r \end{aligned}$$

It allows us to subdivide the representation of $p$ into the two sub-representations on the intervals $[a,(1-t)a+tb]$ and $[(1-t)a+tb,b]$. For a complete list of methods on this representation, we refer for instance to [10].

By a direct extension to the multivariate case, any polynomial $f(x,y,z)$ of degree $d_1$ in $x$ ,$d_2$ in $y$, $d_3$ in $z$, can be decomposed as:

$$f(x,y,z) = \sum_{i=0}^{d_1} \sum_{j=0}^{d_2} \sum_{k=0}^{d_3} b_{i,j,k} B_{d_1}^i(x;a_1,b_1)\, B_{d_2}^j(y;a_2,b_2)\, B_{d_3}^k(z;a_3,b_3),$$

where $(B_{d_1}^i(x;a_1,b_1)\, B_{d_2}^j(y;a_2,b_2)\, B_{d_3}^k(z;a_3,b_3))_{0\leq i\leq d_1, 0\leq j\leq d_2, 0\leq k\leq d_3}$ is the tensor product Bernstein basis on the domain $\mathcal{D} := [a_1,b_1] \times [a_2,b_2] \times [a_3,b_3]$. The polynomial $f$ is represented in this basis by the third order tensor of control coefficients $\mathbf{b} = (b_{i,j,k})_{0\leq i\leq d_1, 0\leq j\leq d_2, 0\leq k\leq d_3}$.

Hereafter, we will denote by a *cell*, the pair of the box $[a_1,b_1]\times[a_2,b_2]\times[a_3,b_3]$ together with the control coefficients $\boldsymbol{b}$, representing $f$. The size of a cell will be $\max\{|b_1-a_1|, |b_2-a_2|, |b_3-a_3|\}$.

De Casteljau algorithm also applies in the $x$, $y$ or $z$-direction. Because of this tensor product representation, the control coefficients of the derivative $\partial_x f(x,y,z)$ are given by:

$$d\,(b_{i+1,j,k} - b_{i,j,k})_{0\leq i\leq d_1-1, 0\leq j\leq d_2, 0\leq k\leq d_3}$$

and similarly for the derivatives $\partial_y f$, $\partial_z f$.

Notice that the univariate Bernstein representation also extends to a so-called triangular Bernstein basis. This representation can also be used in our approach, but we will concentrate on the tensor product one.

### 2.2. Univariate solver

The subdivision criterion that we are going to use, is based on Descartes' rule for a univariate polynomial with control coefficients $\mathbf{b}$ in the Bernstein basis. The number of sign changes of a sequence $\mathbf{b}$, also called the sign variation of $\mathbf{b}$, is denoted hereafter by $V(\mathbf{b})$.

**Proposition 1.** *[10], [26] The number of sign changes $V(\mathbf{b})$ of the control coefficients $\mathbf{b} = [b_i]_{i=0,\ldots,d}$ of a univariate polynomial on $[0,1]$ bounds its number of real roots in $[0,1]$ and is equal to it modulo 2.*

Thus, by this proposition,

- if $V(\mathbf{b}) = 0$, the number of real roots in $[0,1]$ is 0;
- if $V(\mathbf{b}) = 1$, the number of real roots in $[0,1]$ is 1.

This yields the following simple but efficient algorithm:

**Algorithm 1.**
INPUT*: A precision $\epsilon$ and a polynomial $f$ represented in the Bernstein basis of an interval $[b,a]$: $f = (\mathbf{b}, [a,b])$.*

- *Compute the number of sign changes $V(\mathbf{b})$.*
- *If $V(\mathbf{b}) > 1$ and $|b-a| > \epsilon$, subdivide the representation into two sub-representations $\mathbf{b}^-, \mathbf{b}^+$, corresponding to the two halves of the input interval and apply recursively the algorithm to them.*
- *If $V(\mathbf{b}) > 1$ and $|b-a| < \epsilon$, output the $\epsilon/2$-root $(a+b)/2$ with multiplicity $V(\mathbf{b})$.*
- *If $V(\mathbf{b}) = 0$, remove the interval $[a,b]$.*
- *If $V(\mathbf{b}) = 1$, the interval contains one root, that can be isolated within the precision $\epsilon$.*

OUTPUT*: list of subintervals of $[a,b]$ containing exactly one real root of $f$ or of $\epsilon$-roots with their multiplicities.*

In the presence of a multiple root, the number of sign changes of a representation containing a multiple root is bigger than 2, and the algorithm splits the box until its size is smaller than $\epsilon$.

In order to analyze the behavior of the algorithm, we used a partial inverse of Descartes' rule [23] (see also [21]), to show that if $f(x) = 0$ has

only simple roots on $[a, b]$, an upper bound of the number of recursion steps of the algorithm 1 is

$$l = \lceil \log_2 \left( \frac{1+\sqrt{3}}{2s} \right) \rceil,$$

where $s$ is the minimal distance between the complex roots of $f$.

Notice that this localization algorithm extends naturally to B-splines, which are piecewise polynomial functions [10].

## §3. Toward a guaranteed method

The aim of this section is to describe the method, which allows us to build a mesh of the surface $f(x, y, z) = 0$ in a domain $\mathcal{D} = [a_1, b_1] \times [a_2, b_2] \times [a_3, b_3] \subset \mathbb{R}^3$, having the same topology as the surface. The set of points $(x, y, z)$ in $\mathcal{D}$ such that $f(x, y, z) = 0$ will be denoted by $S := Z(f) \cap \mathcal{D}$. The set of singular points of $S$ (where $f = \partial_x f = \partial_y f = \partial_z f = 0$) will be denoted by $S_{\text{sing}}$, the set of smooth (non-singular) points of $S$ by $S_{\text{smooth}}$.

### 3.1. Description of the algorithm

The general scheme of the meshing algorithm is as follows:

- Represent the polynomial $f(x, y, z)$ in the Bernstein basis adapted to the domain $\mathcal{D} = [a_1, b_1] \times [a_2, b_2] \times [a_3, b_3]$ as follows:

$$f(x, y, z) = \sum_{i=0}^{d_1} \sum_{j=0}^{d_2} \sum_{k=0}^{d_3} b_{i,j,k}\, B_{d_1}^i(x)\, B_{d_2}^j(y)\, B_{d_3}^k(z),$$

- Subdivide the box into smaller boxes (using de Casteljau algorithm) until the topology in these boxes can be certified or the size of the box is smaller than $\epsilon$.

It leads to the following scheme:

**Algorithm 2.**

```
L := [Cell(f,D)];
while (L is not empty)
{
 C := first_cell_of(L);
 if(topology_guaranty(C) &&  size(C) < epsilon_smooth )
   insert C at the head in the list of solutions;
 else if(not(topology_guaranty(C)) && size(C)< epsilon_sing)
   insert C at the head in the list of (unmeshed) solutions;
 else {
```

```
   subdivide the cell C;
   insert  the new generated cells at the tail of L;
   remove  C from  L;
 }
}
```

The two parameters involved here are:

- $\varepsilon_{\text{smooth}}$ which is the maximal size of the cells where the topology is guaranteed,
- $\varepsilon_{\text{sing}}$, which is the minimal size after which we consider that the cell contains a singular point.

Hereafter, to simplify the analysis of the algorithm, we will take $\varepsilon = \varepsilon_{\text{smooth}} = \varepsilon_{\text{sing}}$. In practice, it could be interesting to have $\varepsilon_{\text{smooth}} > \varepsilon_{\text{sing}}$, in order to compute large boxes in the smooth part and small boxes around the singularities. This explain why we consider these two parameters.

This subdivision scheme produces a sequence of boxes $F$, which size is decreasing. It corresponds to the construction of an octree, level by level. An advantage of the octree data-structure is the fast localisation of points and of faces or edges shared by several cells [27].

The subdivision criterion that we are going to use, is based on a extension of Descartes' rule for a polynomial with control coefficients $\mathbf{b} = (b_{i,j,k})_{0 \leqslant i \leqslant d_1, 0 \leq j \leq d_2, 0 \leq k \leq d_3}$ in the Bernstein basis.

In order to test whether we have to split a cell, we check if the number of sign changes in one of the directions $x, y, z$ is 0 or 1 and that the sign variation of the control coefficients of the derivative in this direction is 0. More precisely, the sign variation of $f$ in the $x$ direction is the maximum for all $j, k$ with $0 \leq j \leq d_2, 0 \leq k \leq d_3$, of the sign variations of the sequences $\mathbf{b}_{j,k} = (b_{i,j,k})_{0 \leq i \leq d_1}$.

The stopping criterion that we use is the following:

**Definition 2.** *The cell $C$ is $x$-regular (resp. $y$, $z$-regular) for $f$, if the sign variation of $\boldsymbol{b}$ in the $x$ (resp. $y$, $z$) direction is 0 or 1 and if the coefficients of the derivative in this direction have a constant sign.*

A similar definition applies to control coefficients of polynomials in two variables on two-dimensional boxes.

**Lemma 1.** *Let $(u, v, w)$ be any permutation of $(x, y, z)$. Assume that a $(u, v)$-facets $F$ of a cell is $u$-regular for $f$. Then the topology of the surface $f = 0$ on the face $F$ is uniquely determined by its intersection points (counted with multiplicity) with the edges of the face.*

**Proof.** Since $f = 0$ has no singular point on $F$, the trace of $f = 0$ on $F$ is a set of arc segments (possibly of length 0) intersecting the edges of the

face. They project along the $u$-direction on the other axis (say the $v$-axis) as a set of non-overlapping intervals. Consequently, the topology of $f = 0$ on $F$, is the same as those of the set of segments connecting the points of $S$ on the edges, sorted according to their $v$ coordinates and taken by pairs. This proves that the topology of the surface $f = 0$ on the facets of the cell is determined by the its points on the edges. □

**Proposition 3.** *Let $(u, v, w)$ be any permutation of $(x, y, z)$. Assume that $C$ is $u$-regular and that the topology of $f = 0$ on the two $(v, w)$-facets of $C$ is known. Then the topology of the surface $f = 0$ in the box $C$ is uniquely determined by its intersection points (counted with multiplicity) with the edges of the box.*

**Proof.** We can assume without loss of generality, that $f$ is $u$-regular on $F$, for $u = x$ or $u = y$ or $u = z$. According to the previous lemma, the topology of $f = 0$ on all the facets of the cell $C$ is determined. As inside the cell $C$ the surface $S$ is the graph of a function in the $u$-direction, and as there are no singular points of $f = 0$ in $C$, $S \cap C$ is topologically homeomorphic to a set of discs which are determined by the projection of the segments of the facets, on a $(v, w)$-plane along the $u$ -direction. This concludes the proof. □

These lemma and proposition imply that checking the regularity of $f$ in the box $B$ and on faces, and computing the points of the surface on the edges of the box allows us to deduce the topology of the surface in the box. To compute the mesh in a regular cell, we need to compute the points of $S$ (counted with their multiplicity) on the edges of the boxes. This is performed by the univariate solver (see algorithm 1).

This criterion implies that in the valid cells, the derivative of $f$ in one direction is of constant sign and on the two faces transversal to this direction, another derivative is of constant sign. This may be difficult to obtain, when a point of the surface where two derivatives vanish is on (or near) the border of the cell. In order to avoid this situation, we weaken the criterion and improve the subdivision in the following way:

- We check that a derivative of $f$ in one of the directions $x, y, z$ has a constant sign in the cell $C$. If not, the cell is subdivided.
- For the two faces transversal to this direction, we apply the same algorithm on the faces (in 2 dimensions), in order to get polygons representing the trace of $f = 0$ on these faces.

In such a case, the topology of the set $S$ in the cell $C$ is guaranteed: It is the graph of a function, say in the direction $u$ for which the derivative has a constant sign. The polygons of $f = 0$ on all the faces define closed

curves on the border of $C$. Applying proposition 3, we are able to compute the topology of $f = 0$ in $C$.

Notice that if precautions are not taken, the trace of $f = 0$ on the border of $C$ might be a singular curve. To avoid this situation, we simply precompute the critical points of $S$ for the projection in the directions $(x, y), (x, z), (y, z)$. These points are defined by the equations $f = \partial_x f = \partial_y f = 0$, $f = \partial_x f = \partial_z f = 0$, $f = \partial_y f = \partial_z f = 0$. In the case of a smooth surface, after a generic change of coordinates (or simply a generic translation) the number of such points is finite (it is bounded by $3\,d(d-1)^2$ where $d = \deg(f)$, by Bezout's theorem). We avoid these points when the cells are subdivided, by choosing adequately the position of subdivision (applying de Casteljau algorithm for a value of $t$ in between critical values). In order to apply recursively the algorithm in dimension 2, we take the parameters $\varepsilon_{smooth} = \varepsilon_{sing} = \varepsilon$.

These adaptations allow us to prove that for a smooth surface and $\varepsilon$ small enough, the algorithm stops with the correct topology. By the structure of the algorithm, we are able to detect if $\varepsilon$ is not small enough.

To prove termination and correctness, we need the following definition and result on the approximation of a function by the control polygon. Let $K_2(f) = \max_{p\in\mathcal{D}} \|H_2(f)(p)\|$ where $H_2(f)(p)$ is the Hessian of $f$ at $p$. Let $C$ be a cell of size $\varepsilon$.

Let $s_{i,j,k}$ be the points of the regular subdivision associated with the control coefficients $c_{i,j,k}$ of $f$ on $C$. Then there exists $\gamma_2(\boldsymbol{d}) = \gamma_2(d_1, d_2, d_3)$ depending of $d_1 = \deg_x(f), d_2 = \deg_y(f), d_3 = \deg_z(f)$ such that

$$|f(s_{i,j,k}) - c_{i,j,k}| < \gamma_2(\boldsymbol{d}) K_2(f)\varepsilon^2. \tag{1}$$

See eg. [24], [25], [20] for a proof and more details on this result. We denote $\kappa_2(f) = \gamma_2(\boldsymbol{d})K_2(f)$.

First, we analyze the cells which are rejected by the algorithm. We denote $\Gamma_f(r) = \{p \in \mathcal{D}, |f(p)| \leqslant r\}$.

**Proposition 4.** *Let $C$ be a cell of size $\varepsilon$, outside $\Gamma_f(\kappa_2(f)\varepsilon^2)$. Then the control coefficients of $f$ on $C$ are of constant sign.*

**Proof.** As $C$ is outside $\Gamma_f(\kappa_2(f)\varepsilon^2)$, $f$ does not vanish in $C$, so that it has a constant sign. Assume, without loss of generality, that $f > 0$ so that $f > \kappa_2(f)\varepsilon^2 > 0$ in $C$. Then by (1), we have

$$c_{i,j,k} = f(s_{i,j,k}) - (f(s_{i,j,k}) - c_{i,j,k}) > \kappa_2(f)\varepsilon^2 - \kappa_2(f)\varepsilon^2 = 0.$$

□

In consequence, such a cell will not be kept by the algorithm.

**Theorem 1.** *If the surface $S$ defined by $f(x, y, z) = 0$ is smooth in $\mathcal{D}$, then the algorithm 2 stops for $\varepsilon_{smooth} > \varepsilon_{sing}$ small enough, and output a mesh homeomorphic to $S$.*

**Proof.** By equation (1), for $\varepsilon := \varepsilon_{smooth}$ small enough, the cells $C$ which are kept by the algorithm intersect $\Gamma_f(\kappa_2(f)\varepsilon^2)$.

Let us denote by $x_0$ a point of $\Gamma_f(\kappa_2(f)\varepsilon^2) \cap C$. For any $x \in C$, we have

$$|f(x) - f(x_0)| \leq \kappa_1(f)\, ||x - x_0||_\infty \leq \kappa_1(f)\varepsilon$$

where $\kappa_1(f) = \max_{p\in\mathcal{D}} ||(\partial_x f(p), \partial_y f(p), \partial_z f(p))||_1$. As $x_0 \in \Gamma_f(\kappa_2(f)\varepsilon^2)$, we have

$$|f(x))| \leq \kappa_1(f)\varepsilon + \kappa_2(f)\varepsilon^2,$$

which implies that $C \subset \Gamma_f(\kappa_1(f)\varepsilon + \kappa_2(f)\varepsilon^2)$. As $S$ is smooth, for $\varepsilon$ small enough, we have

$$\begin{array}{ll} & \Gamma_f(\kappa_1(f)\varepsilon + \kappa_2(f)\varepsilon^2) \\ \cap & \Gamma_{\partial_x f}(\kappa_2(\partial_x f)\varepsilon^2) \cap \Gamma_{\partial_y f}(\kappa_2(\partial_y f)\varepsilon^2) \cap \Gamma_{\partial_z f}(\kappa_2(\partial_z f)\varepsilon^2) = \emptyset. \end{array}$$

This implies, for $\varepsilon$ small enough, for any cell $C$ of size $\varepsilon$ kept by the algorithm and for all $x \in C$, either $|\partial_x f(x)| > \kappa_2(\partial_x f)\varepsilon^2$ or $|\partial_y f(x)| > \kappa_2(\partial_y f)\varepsilon^2$ or $|\partial_z f(x)| > \kappa_2(\partial_z f)\varepsilon^2$. By equation (1), either $\partial_x(f)$ or $\partial_y(f)$ or $\partial_z(f)$ has its Bernstein coefficients of the same sign in $C$. A similar proof applies for the trace of $f$ on the transversal faces, since we have avoided the critical sections, for which the trace of $f$ on the face is singular. Consequently, for $\varepsilon_{smooth}$ and $\varepsilon_{sing}$ small enough the algorithm stops on cells, in which the topology of $f$ is guaranteed. □

### 3.2. Complexity analysis

In this section, we analyze the behavior of the algorithm as the size of the cells goes to 0. Let $A$ be a subset of the surface $S$ in the domain $\mathcal{D}$.

**Definition 5.** *We denote by $C(\varepsilon, A)$ the minimal union of cells of size $\leqslant \varepsilon$ in the octree, covering $A$. Let $N(\varepsilon, A)$ be the number of cells involved in $C(\varepsilon, A)$.*

In order to analyze the number of boxes $N(\varepsilon, A)$, we connect it to the following notion [11], [29]:

**Definition 6.** ***($\varepsilon$-entropy)*** *For any set $A$ in $\mathbb{R}^3$, let $E(\varepsilon, A)$ be the minimum number of closed balls of radius $\varepsilon$, covering $A$.*

We will first show that $N(\varepsilon, A)$ is of the same order than the *entropy* $E(\varepsilon, A)$ of $A \subset \mathbb{R}^3$:

**Proposition 7.** *$E(\varepsilon, A) \leqslant N(\varepsilon, A) \leqslant \gamma_0 E(\varepsilon, A)$ where $\gamma_0 = \mu(4\sqrt{3})$ where $\mu(r)$ is the minimal number of balls of radius 1, covering a ball of radius $r$.*

**Proof.** Since a cell of size $\varepsilon$ is covered by a ball of radius $\varepsilon$, and $C(\varepsilon, A)$ covers $A$, we have $E(\varepsilon, A) \leqslant N(\varepsilon, A)$.

Since the Hausdorff distance between $A$ and $C(\varepsilon, A)$ is at most the length $\sqrt{3}\,\varepsilon$ of the diagonal of the cube, we have (see [29])

$$E(2\sqrt{3}\varepsilon, C(\varepsilon, A)) \leqslant E(\sqrt{3}\,\varepsilon, A).$$

On the other hand, $N(\nu, A) \leqslant E(\frac{\nu}{2}, C(\nu, A))$ since a cell of size $\nu$, cannot be covered by a single ball of radius $\frac{\nu}{2}$, so that we have:

$$N(\varepsilon, A) \quad \leqslant \quad E(\frac{\varepsilon}{2}, C(\varepsilon, A)) \leqslant \mu(4\sqrt{3})E(2\sqrt{3}\,\varepsilon, C(\varepsilon, A))$$

since $E(\frac{\nu}{\lambda}, A) \leq \mu(\lambda)E(\nu, A)$ for $\lambda > 0, \nu > 0$. We deduce that

$$N(\varepsilon, A) \quad \leqslant \quad \mu(4\sqrt{3})E(\sqrt{3}\,\varepsilon, A) \leqslant \mu(4\sqrt{3})E(\varepsilon, A),$$

since $E(\sqrt{3}\,\varepsilon, A) \leq E(\varepsilon, A)$ □

Next we will use the relations between the $\varepsilon$-entropy and the Vitushkin variations, defined as follows:

**Definition 8.** *For any set $S \subset \mathbb{R}^3$, let $V_0(S)$ be the number of connected components of $S$, and*

$$V_i(S) = c(i) \int_{L \in \mathcal{G}_{3-i}} V_0(S \cap L)\, \mathrm{dL},$$

*where $\mathcal{G}_k$ is the Grassmannian of affine spaces of dimension $k$ in $\mathbb{R}^3$, $\mathrm{dL}$ is the canonical measure on $\mathcal{G}_{3-i}$, and $c(i) = \dfrac{1}{\displaystyle\int_{L \in \mathcal{G}_{3-i}} V_0([0,1]^i \cap L)\, \mathrm{dL}}$, (so that $V_i([0,1]^i) = 1$ and $c(3) = 1$).*

Our aim is now to relate the number of boxes produced by the algorithm to geometric invariants of the surface, such as the variations $V_i(S)$:

**Theorem 2.** *Suppose that the surface $S \subset \mathcal{D}$ defined $f(x, y, z) = 0$, is smooth in $\mathcal{D}$. Then the number $N$ of cells produced by the algorithm for $\varepsilon = \varepsilon_{smooth}$ is bounded by*

$$N \leqslant \gamma_0 \left( V_0(S) + \frac{1}{\varepsilon} V_1(S) + \frac{1}{\varepsilon^2} V_2(S) \right). \tag{2}$$

*where $\gamma_0 \in \mathbb{R}_{>0}$ is a universal constant.*

**Proof.** We use the following property [15], [29]:

$$E(\varepsilon, S) \leqslant \left( V_0(A) + \frac{1}{\varepsilon} V_1(S) + \frac{1}{\varepsilon^2} V_2(S) + \frac{1}{\varepsilon^3} V_3(S) \right),$$

and the property that $V_3(S) = 0$ since $S$ is of dimension 2.

Now by proposition 7, we have

$$N(\varepsilon, S) \leqslant \gamma_0' \left( V_0(S) + \frac{1}{\varepsilon} V_1(S) + \frac{1}{\varepsilon^2} V_2(S) \right).$$

By proposition 4, in every cell outside $\Gamma_f(\kappa_2(f)\varepsilon^2)$, the Bernstein coefficients of $f$ have the same sign. Thus such a cell is not kept by the algorithm. Consequently, we have

$$N \leq N(\varepsilon, \Gamma_f(\kappa_2(f)\varepsilon^2)).$$

As $S$ is smooth in $\mathcal{D}$, the function $x \in \mathcal{D} \mapsto \frac{\operatorname{dist}(x,S)}{|f(x)|}$ is well defined and bounded by a constant $\kappa_1(f)$.

Thus, for $\epsilon$ small enough

$$\Gamma_f(\kappa_2(f)\varepsilon^2)) \subset S_\epsilon = \{x \in \mathcal{D}; \operatorname{dist}(x, S) < \varepsilon\}$$

We deduce that

$$N \leq N(\varepsilon, S_\epsilon) \leq 27 N(\varepsilon, S),$$

since by surrounding each of the cells covering $S$, by its 26 neighbors cells we cover the points of $S_\epsilon$ at distance $\epsilon$ from $S$. This proves inequality (2), with $\gamma_0 = 27\gamma_0'$. □

Notice that we can link $V_1(S)$ to the curvature of $S$, since there exists a universal constant $c_1$ such that

$$V_1(S) \leqslant c_1 \int_{S_{smooth}} |k_1(p)| + |k_2(p)| dp,$$

where $k_1(p), k_2(p)$ are the principal curvatures of $S$ at $p$.

Similarly, we have

$$V_2(S) = \operatorname{Area}(S).$$

See [29], [17] for more details.

### 3.3. Singularities

In our framework the treatment of singular points is a delicate task, although certifying the topology of a tame set (ie. algebraic, semi-algebraic, subanalytic or more generally a set definable in some o-minimal structure) near one of its points is formally possible, since such a set has locally around each of its points a cone-like topology. In some small enough ball

centered at the special point $p$, our set as the same topology as the cone of vertex $p$ constructed on the intersection of the set and the boundary of the ball.

This is a consequence of the existence of Whitney stratifications for these sets and of general properties of topological uniform finiteness. Let us be more precise.

Let $p \in S$ be a singular point of $S$ and let $\beta_p : S \ni q \to \|q-p\|_\infty$ where $\|\cdot\|_\infty$ is the infinity norm. Note that the set $\beta_p^{-1}(r) = \{q \in S, \beta_p(q) = r\}$ is the intersection of $S$ with the faces of a box centered at $p$ and of size $2r$. This map $\beta_p$ is semi-algebraic, so that the following theorem applies:

**Theorem 3.** *[12]; [13] There exists $\varepsilon_0 \in \mathbb{R}_{>0}$ such that for each $0 < r \leq \varepsilon_0$, $(\beta_p^{-1}(\varepsilon_0), \beta_p^{-1}([0,\varepsilon_0]))$, $(\beta_p^{-1}(r), \beta_p^{-1}([0,r]))$ and $(\beta_p^{-1}(r)\times\{1\}, \beta_p^{-1}(r)\times[0,1]/\equiv)$ are homeomorphic, where $\equiv$ is defined on $\beta_p^{-1}(r) \times [0,1]$ by $(x,t) \equiv (y,s) \Leftrightarrow t = s = 0$.*

In other words, for a sufficiently small cell $C$ centered at a singular point $p$ of $S$, the topology of $S$ in the cell $\beta_p^{-1}(r)$, $r \leq \varepsilon_0$ is the same as those of the cone of vertex $p$, "lying" on the curve $\beta_p^{-1}(\varepsilon_0)$ given by the intersection of $S$ with the facets of C. In order to compute a coherent mesh in such a case, we compute the star-triangulation whose vertex is a singular point of the surface in the cell, and which basis is the piecewise-linear approximation of the curve of intersection of the surface with the boundary of the box. We will describe this meshing method near singular points in a forthcoming paper.

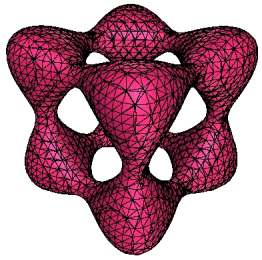
## §4. Experimentation

We present here some experimentations on surfaces related to the classification of singularities [3]. The implementation is available in the library AXEL (Algebraic Software-Components for gEometric modeLing)[1]. The table reports on the number of triangles, of cells including the singular one denote by **nt** (represented by boxes in the pictures) and the timing. The tests have been run on a Pentium IV 2.4 Ghz workstation. We consider a smooth case, a case with finitely many singular points, with a self intersection curve and with the singular points containing an isolated curve arc[2]. The parameters used for the subdivision are $\varepsilon_{smooth} = 2^{-5}\,|\mathcal{D}|$ and $\varepsilon_{sing} = 2^{-8}\,|\mathcal{D}|$. The pictures show the corresponding mesh. In order to get a better rendering we could compute the normal at points on the surface. This is direct from the implicit equation, but is not done in the following visualization. Notice also that once the topology is certified, the triangulation can be improved in the smooth boxes, according to geometric criteria [9], [8].

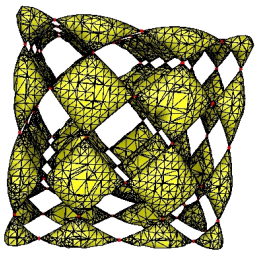
[1] `http://www-sop.inria.fr/galaad/software/axel`

[2] More examples can be found at `http://www-sop.inria.fr/galaad/data/surface/`

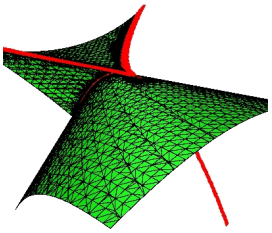
**Equation:**
$x^4 - 5\,x^2 + y^4 - 5\,y^2 + z^4 - 5\,z^2$
$+11.8 = 0$
**Nb of triangles**: 8881
**Nb of cells**: 4165
**Time (s)**: 1.53 s



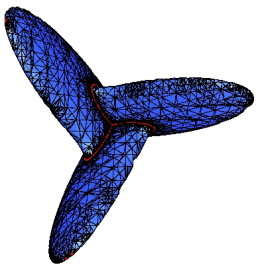
**Equation**:
$32\,x^8 - 64\,x^6 + 40\,x^4 - 8\,x^2 + 1 + 32\,y^8$
$-64\,y^6 + 40\,y^4 - 8\,y^2 + 32\,z^8 - 64\,z^6$
$+40\,z^4 - 8\,z^2 = 0$
**Nb of triangles**: 45680
**Nb of cells**: 14555 + 594 nt
**Time (s)**: 8.13 s



**Equation:**
$-4\,z^3\,y^2 - 27\,y^4 + 16\,x\,z^4 - 128\,x^2\,z^2$
$+144\,x\,y^2\,z + 256\,x^3 = 0$
**Nb of triangles**: 21354
**Nb of cells**: 7752 + 4684 nt
**Time (s)**: 53.39 s



**Equation:**
$-2749.231165\,x^3\,z\,y^2 - 1832.820776\,y\,z^2\,x^2$
$+648\,z\,x^2\,y^2 - 1620\,z^2\,x^2\,y^2$
$+1832.820776\,y\,z^3\,x^2 - 4123.846747\,y^4\,x\,z$
$+916.4103882\,z^3\,x\,y^2 + 64\,z^3 + 432\,z^5 - 216$
$-729\,x^6 - 729\,y^6 - 144\,z^2\,x^2 - 288\,z^4$
$-144\,z^2\,y^2 + 324\,z^4\,x^2 + 324\,z^4\,y^2$
$+324\,z\,x^4 + 324\,z\,y^4 + 610.9402588\,y^3\,z^2$
$-810\,z^2\,y^4 - 610.9402588\,y^3\,z^3 - 2187\,x^4\,y^2$
$-2187\,x^2\,y^4 + 1374.615582\,x^5\,z$
$-305.4701294\,z^3\,x^3 = 0$
**Nb of triangles**: 26184
**Nb of cells**: 7924 + 1616 nt
**Time (s)**: 9.66 s



**Acknowledgments:** We would like to thanks specially L. Deschamps, for his help in the implementation of a first prototype of this algorithm.

This work was partially supported by the european projects GAIA IST-2001-35512 and Aim@Shape FP6 IST NoE 506766.

## §5. References

1. S Akkouche. and E. Galin. Adaptive implicit surface polygonization using marching triangles. *Comput. Graph. Forum*, 20(2):67–80, 2001.
2. Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T. Silva. Point set surfaces. In *Proceedings of the conference on Visualization '01*, pages 21–28. IEEE Computer Society, 2001.
3. V. Arnold, A. Varchenko, and S. M Gusein-Zade. *Singularités des applications différentiables*. Edition Mir, Moscou, 1986.
4. J. F. Blinn. A generalization of algebraic surface drawing. *ACM Transactions on graphics*, 1(3):235–256, july 1982.
5. J. Bloomenthal. Polygonization of implicit surfaces. *Computer-Aided Geometric Design*, 5(4):341–355, 1988.
6. J. Bloomenthal. An implicit surface polygonizer. In Paul Heckbert, editor, *Graphics Gems IV*, pages 324–349. Academic Press, Boston, MA, 1994.
7. Jean-Daniel Boissonnat, David Cohen-Steiner, and Gert Vegter. Meshing implicit surfaces with certified topology. Technical Report 4930, INRIA Sophia-Antipolis, 2003.
8. Jean-Daniel Boissonnat and Steve Oudot. An effective condition for sampling surfaces with guarantees. In *Proc. Symp. on Geometry Processing*, pages 9–18, 2004.
9. L. P. Chew. Guaranteed-quality mesh generation for curved surfaces. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 274–280, 1993.
10. G. Farin. *Curves and surfaces for computer aided geometric design : a practical guide*. Comp. science and sci. computing. Acad. Press, 1990.
11. H. Federer. *Geometric measure theory*, volume 153 of *Grund. Math. Wiss.* Springer-Verlag, 1969.
12. M. Goresky and R. MacPherson. *Stratified Morse theory*, volume 14 of *Ergebnisse Math.* Springer-Verlag, 1988.
13. R.M. Hardt. Semi-algebraic local-triviality in semi-algebraic mappings. *Amer. J. Math.*, 102(2):291–302, 1980. (Mather Notes on topological stability, Harvard University, 1970).
14. E. Hartmann. A marching method for the triangulation of surfaces. *Visual Computer*, 14(3):95–108, 1998.

15. L.D. Ivanov. *Variations of sets and functions (Russian).* Izdat. "Nauka", Moscow, 1975. Edited by A.G. Vituskin.

16. D. Kalra and A.H. Barr. Guaranteed ray intersections with implicit surfaces. In *Proc. of SIGGRAPH*, volume 23, pages 297–306, 1989.

17. A.M. Leontovic and M.S. melnikov. On the boundenness of variations of a manifold (russian). *Trudy Moskov. Mat. Obsc.*, pages 306–337, 1965.

18. T. Lewiner, H. Lopes, A. W. Vieira, and G. Tavares. Efficient implementation of marching cubes cases with topological guarantees. *Journal of Graphics Tools*, 8(2):1–15, 2003.

19. W. Lorensen and H. Cline. Marching cubes: a high resolution 3d surface construction algor ithm. *Comput. Graph.*, 21(4):163–170, 1987.

20. David Lutterkort and Jörg Peters. Optimized refinable enclosures of multivariate polynomial pieces. *Comput. Aided Geom. Design*, 18(9):851–863, 2001.

21. K. Mehlhorn. A remark on the sign variation method for real root isolation. Submitted for publication, report ECG-TR-123101-01.

22. R. Morris. A new method for drawing algebraic surfaces. In *Design and application of curves and surfaces (Edinburgh, 1992)*, volume 50 of *Inst. Math. Appl. Conf. Ser. New Ser.*, pages 31–47. Oxford Univ. Press, New York, 1994.

23. B. Mourrain, M. Vrahatis, and J.C. Yakoubsohn. On the complexity of isolating real roots and computing with certainty the topological degree. *J. of Complexity*, 18(2):612–640, 2002.

24. D. Nairn, J. Peters, and D. Lutterkort. Sharp, quantitative bounds on the distance between a polynomial piece and its Bézier control polygon. *Comput. Aided Geom. Design*, 16(7):613–631, 1999. Dedicated to Paul de Faget de Casteljau.

25. Ulrich Reif. Best bounds on the approximation of polynomials and splines by their control structure. *Comput. Aided Geom. Design*, 17(6):579–589, 2000.

26. J.J. Risler. *Méthodes mathématiques pour la CAO.* Masson, 1991.

27. H. Samet. *The Design and Analysis of Spatial Data Structures.* Addison-Wesley, Reading, MA, 1990.

28. A. Witkin and P. Heckbert. Using particles to sample and control implicit surface. In *Proc. of SIGGRAPH*, pages 269–277, 1994.

29. Y. Yomdin and G. Comte. *Tame geometry with applications in smooth analysis.* LNM 1834. Springer-Verlag, 2004.