Complexity of Discrete Surfaces in the Dividing-Cubes Algorithm

Fatima Boumghar*
LASIE-Alger
U.S.T.H.B. Bab-Ezzouar
16011 Alger (Algeria)

Serge Miguet**
Laboratoire ERIC
Bât. L, Université Lyon-2
5 av. Pierre Mendès-France
69676 Bron (France)

Jean-Marc Nicod
LIP, URA CNRS 1398
Ecole Normale Supérieure de Lyon
46 allée d'Italie
69364 Lyon Cedex 7 (France)

[fboumgha,miguet,jmnicod]@lip.ens-lyon.fr
miguet@eric.univ-lyon2.fr

Abstract. The main result of this paper is to exhibit a complexity model for discrete surfaces obtained by regular subdivisions of cells. We use it for estimating the number of points that will be generated by the Dividing-Cubes algorithm to represent the surface of 3D medical objects. Under the assumption that surfaces have uniform orientations in the space, and can be locally compared to planes, we show that their average number of points is a quadratic function of the subdivision factors. We give analytical expressions for the coefficients of the quadratic form.

1 Introduction

Medical volume images can be obtained by many medical imaging devices like *Computerized Tomography Scanners*, *Magnetic Resonance Imagers*, *Positron or single photon Emission Tomography scanners* or even *ultrasound scanners*. These 3D images contain detailed information about 3D internal structures (human organs) and need appropriate extraction and visualization methods. Many techniques have been designed [6]. They present a compromise between the precision of the representation and the computing time of the images. A classification of these methods can be proposed according to the dimensionality of the graphic primitives that are employed:

- The oldest methods are based on 1D primitives. The objects contours appear on each slice and are extracted by 2D image processing techniques. The surfaces can then be visualized using wireframe representation. Two adjacent contours can then be joined by a polygon mesh to generate an approximation of the external surface of the objects [5, 7]. The main problems to solve with this approach are the closing of the surfaces and the variable number of contours on the adjacent slices generated by complex structures.

* Invited at the LIP of the ENS Lyon during spring 1996.
** This research has been done while the author was member of the LIP, ENS-Lyon.

- Lorensen and Cline extract directly 2D primitives (polygon meshes) from the 3D images: they consider, in the Marching-Cubes algorithm [9], the cubic cells composed by eight adjacent samples of the image. For a value of the threshold, they determine the intersection of the surface with each cell and thus generate a polygonal approximation.
- Many researchers are working on direct visualization techniques of 3D images without intermediate representation of data [8, 4]. Algorithms based on ray-tracing or on direct projection of voxels (3D primitives) give high quality images, with relative transparency or color of some materials being taken into account. On the other hand, these methods are very time consuming and are not compatible with real-time constraints.
- Cline and al, notice in [3], that the polygons generated by the Marching-Cubes algorithm are projected on only some pixels of the screen. Without a specialized graphics hardware to perform the rendering, they suggest to represent the surface by a cloud of points. They propose in the Dividing-Cubes algorithm a regular subdivision of the voxels so that the size of the projection of a subdivided voxel on the screen is at most equal to the size of a pixel.

The work presented here is a part of a project developed at the *LIP*, whose aim is the elaboration of a parallel environment dedicated to 3D medical imaging [10]. The objectives of our research group is to study the implementation, on distributed memory parallel machines, of 3D imaging methods. The processing nodes are general-purpose processors, not specialized in graphic operations.

Therefore, we want to study a parallel version of the Dividing-Cubes algorithm on these kind of platforms. To have a balanced execution of the algorithm, we have to estimate the workload, as a function of the algorithm parameters, which are directly related to the number of points generated on the iso-surface. The remainder of the paper is organized as follows:

Section 2 presents the basic principles of the Dividing-Cubes algorithm. In section 3 we summarize previous results concerning the optimization of subdivision parameters, for the visualization of still or animated images as well as a technique for quick production of lower quality images. We explain then how to determine the number of points generated on the iso-surface as a function of the subdivision factors. A statistical study in 2D and in 3D allows to get a model of the complexity of the extracted surface. Our relations are validated by experimentation on artificial and medical images.

2 The Dividing-Cubes algorithm

2.1 Subdivision and Surface Reconstruction

The basic idea of the Dividing-Cubes algorithm is that the original voxels of the 3D images are subdivided into $a \times b \times c$ elementary cells. The subdivision factors a, b and c are chosen so that the size of one projected elementary cell is at most equal to the pixel size on the raster display. Then the algorithm generates

one point of the surface for each elementary cell intersected by the iso-surface. A point is represented by three coordinates (the center of the small cell) and a color calculated from the Phong model. The normal on a small cube is obtained by a trilinear interpolation of the normals to the vertices of the cells which are themselves determined by a gradient computation. The Dividing-Cubes algorithm eliminates the scan conversion step used when rendering surfaces extracted by the Marching-Cubes algorithm.

The choice of the threshold on an iso-surface determines which organ we want to visualize. In figure 1, we present two iso-surfaces extracted from the same 3D image. The left and the right image are processed for the threshold corresponding respectively to the skin tissue and to the bone surface.

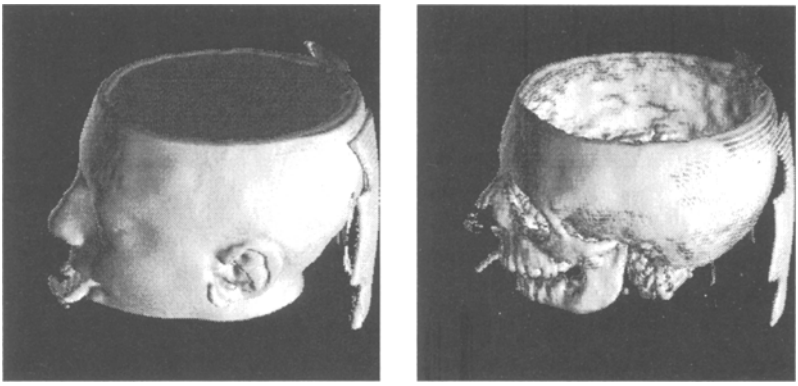

Fig. 1. Two iso-surfaces of a same 3D medical image

2.2 Visualization of the iso-surface

Once the iso-surface is extracted, the visualization is straightforward: each point is projected according to the visualization parameters. Its color is stored in the frame buffer and the elimination of the hidden parts is done by the depth buffer algorithm (ZBuffer).

Figure 2 presents the visualization parameters used when the points of the surface are projected in perspective. The voxels coordinates are normalized between -1 and +1. A view is defined by 6 parameters which are the distance r of the observer to the 3D image center, the focal distance f from the observer to the raster display center, the two Euler angles θ and φ, and the raster display resolution (x, y).

3 Optimization of the subdivision parameters

We explain in [1] how to optimize the subdivision parameters, both for the production of a single image and for the production of an animation, where vi-

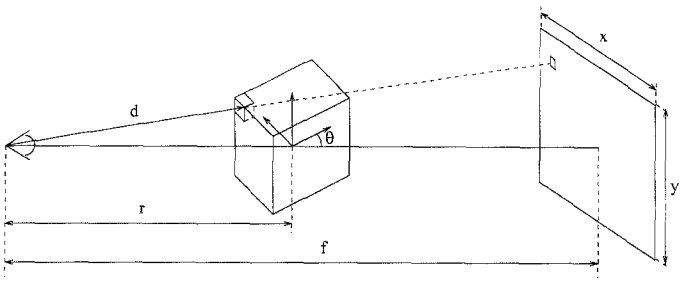

Fig. 2. Visualization parameters

sualization parameters vary. We give sufficient conditions to garantee that the surface does not present holes. The number of points generated by this algorithm can nevertheless be very important, around one million for the visualization of a medical image in 512×512 resolution.

We also explain in [1] a mechanism for fast production of lower quality images. The idea is to generate less points than the theoretically minimal number and to render each point by a rectangle instead of a single pixel.

The following section presents an estimation of the number of points generated on the surface, as a function of the subdivision parameters. This modelisation allows to predict the memory allocations needed to store the iso-surface. As part of a parallelization of the algorithm, this model also allows to predict precisely the workload of the algorithm which can then be balanced.

4 Complexity of iso-surfaces

We can estimate the number of the points $S(a, b, c)$ generated on the given iso-surface as a function of the subdivision parameters a, b and c. The simple algorithm given in pseudo-C notation in Figure 3 first explains how S(1,1,1) can be computed. S(1,1,1) represents the number of intersections with the iso-surface when cells are not subdivided. Obviously, it depends on the threshold value. The method we give here scans only once the whole 3D image, and precomputes this number for any possible threshold.

It is natural to think that $S(a, b, c)$ is proportional to $S(1, 1, 1)$, the proportionality factor being a quadratic function of a, b and c. Indeed, only those cells that are intersected by the iso-surface are subdivided into abc small cubes. But in the volume composed by these abc small cubes, only a discrete surface will be preserved. We have then searched for a relation of the following form:

$$\frac{S(a, b, c)}{S(1, 1, 1)} = \alpha abc + \beta ac + \gamma ab \tag{1}$$

```
for i = 0 to 255 do
    W[i] = 0;
for each cell (i,j,k) do
    m = min(V(i + ε1, j + ε2, k + ε3)); /* εl = 0 or 1 */
    M = max(V(i + ε1, j + ε2, k + ε3)); /* εl = 0 or 1 */
        /* this cell is intersected by each iso-surface σ
         * of threshold σ such as m ≤ σ < M */
    W[m]++;
    W[M]- -;
for i = 1 to 255 do
    W[i] += W[i-1];
```

Fig. 3. Threshold-independant computation of $S(1,1,1)$: at the end of this procedure, $W[k]$ holds the number of (non-divided) cells intersected by surface of threshold k

The experimentations carried out in section 4.3 show a good adequation of this formula with the reality. We have also observed that a very simple relation links together the parameters α, β and γ:

$$\alpha + \beta + \gamma \approx 1 \tag{2}$$

The following section presents a statistical proof of these observations and gives mathematical formulation of α, β and γ parameters. We suppose that the cells size are small enough to be able to assimilate locally the object's surface to planes. We suppose also that the orientations and positions of these planes have equal probabilities. We determine then an average number of subdivided cells intersected by the iso-surface.

We lead in a first step the statistical study in 2D. In this case, the planes become straight lines, the cells rectangles and the iso-surfaces iso-contours. We generalize afterwards this study in 3D.

4.1 Statistical study in 2 dimensions

The cell is a rectangle of dimension $d_x \times d_y$, subdivided into $a \times b$ small rectangles. We consider the set of the straight lines $\mathcal{D}$ of the plane defined by equation:

$$x\cos(\theta) + y\sin(\theta) = r \tag{3}$$

Among this set of straight lines we choose those ones that have an intersection with the cell (see figure 4). We search to establish the average number of small

rectangles intersected by the straight line $\mathcal{D}$ as a function of the parameters a and b. By symmetry we consider the following intervals: $\theta \in [0, \pi/2]$ and $r \in [0, r_{max}(\theta)]$ where $r_{max}(\theta)$ is the distance to the origin of the straight line beyond which there is no more intersection ($r_{max}(\theta) = \cos(\theta)d_x + \sin(\theta)d_y$).

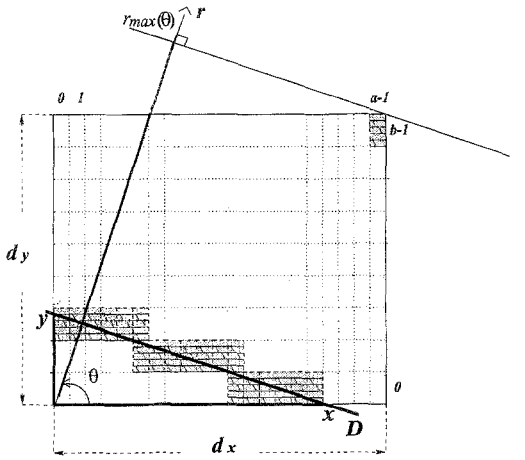

Fig. 4. Cell intersected by a straight line $\mathcal{D}(r, \theta)$

Given $f(r, \theta)$ the number of small rectangles intersected (discrete length) and $l(r, \theta)$ the euclidean length of the intersection. We have the following relation:

$$f(r, \theta) = \rho(\theta) l(r, \theta) \tag{4}$$

where $\rho(\theta)$ is a factor allowing to convert the euclidean length to the discrete length along the straight line perpendicular to the direction given by the angle θ.

Expression of $\rho(\theta)$: given $\boldsymbol{v}(x, y)$ a direction vector of the straight line $\mathcal{D}$. The length l or euclidean norm of $\boldsymbol{v}$ as well as its discrete length f (number of intersected cells) are given by the following relations:

$$l = \|\boldsymbol{v}\| \tag{5}$$

$$= \sqrt{x^2 + y^2} \tag{6}$$

$$f \approx \frac{xa}{d_x} + \frac{yb}{d_y} \tag{7}$$

according to (4) the expression of $\rho(\theta)$ is the following:

$$\rho(\theta) = \frac{f}{l} \tag{8}$$

$$= \frac{a}{d_x}\sin(\theta) + \frac{b}{d_y}\cos(\theta) \tag{9}$$

Determination of the average value f_{avg} of $f(r,\theta)$ for all the orientations and positions of the straight lines $\mathcal{D}$ having an intersection with the cell. f_{avg} can be expressed as a double integral on r and θ. See [2] for more detailed computations.

$$f_{avg} = \frac{2}{\pi}\int_0^{\frac{\pi}{2}} \frac{1}{r_{max}(\theta)} \int_0^{r_{max}} f(r,\theta) dr d\theta \tag{10}$$

$$= \frac{2}{\pi}\int_0^{\frac{\pi}{2}} \frac{bd_x\cos(\theta) + ad_y\sin(\theta)}{d_x\cos(\theta) + d_y\sin(\theta)} d\theta \tag{11}$$

after integration, we obtain the following relation, where $\varepsilon = \frac{d_y}{d_x}$:

$$f_{avg} = \frac{\pi + 2\varepsilon\ln(\varepsilon)}{\pi(1+\varepsilon^2)} b + \frac{\varepsilon^2\pi - 2\varepsilon\ln(\varepsilon)}{\pi(1+\varepsilon^2)} a \tag{12}$$

The equation (12) is in form $\alpha b + \beta a$ and corresponds to a 2D formulation of the relation (1). We can verify that we have $\alpha + \beta = 1$, which corroborate the relation (2) in 2D.

4.2 Generalization in 3D

We take again the initial definition of a tridimensionnal cell. It is a rectangular parallelepiped of dimensions $d_x \times d_y \times d_z$, subdivided into $a \times b \times c$ small cells.

Like in 2D, we search the average number of small cells intersected by all possible planes. One of these planes is defined by its equation:

$$x\cos(\varphi)\cos(\theta) + y\cos(\varphi)\sin(\theta) + z\sin(\varphi) = r \tag{13}$$

where θ and φ are the Euler angles of the normal to the plane and r its distance to the origin. We can limit the integration space to the first octant for θ and φ and to the interval $[0, r_{max}(\theta,\varphi)]$ for r.

Like in 2D we express the number of small cells intersected by a discrete plane surface as a function of the area of the corresponding continuous surface element. We have the relation:

$$f(r,\theta,\varphi) = \rho(\theta,\varphi)s(r,\theta,\varphi) \tag{14}$$

where $\rho(\theta,\varphi)$ is the factor to convert the continuous area $s(r,\theta,\varphi)$ to the corresponding discrete surface $f(r,\theta,\varphi)$. We show in [2] that $\rho(\theta,\varphi)$ is expressed as:

$$\rho(\theta,\varphi) = \frac{ab}{d_x d_y}\sin(\varphi) + \frac{c}{d_z}\cos(\varphi)\left(\frac{b}{d_y}\cos(\theta) + \frac{a}{d_x}\sin(\theta)\right) \tag{15}$$

Determination of the average value f_{avg} of $f(r,\theta,\varphi)$ for all the orientations and positions of the planes in the space which have an intersection with the cell; f_{avg} represents the average number of subdivided cells intersected by the iso-surfaces. f_{avg} can be expressed as the triple integral on the domain occupied by the cell:

$$f_{avg} = \frac{2}{\pi}\int_0^{\frac{\pi}{2}}\int_0^{\frac{\pi}{2}} \frac{1}{r_{max}(\theta,\varphi)} \int_0^{r_{max}(\theta,\varphi)} f(r,\theta,\varphi)dr\cos(\varphi)d\varphi d\theta \tag{16}$$

where:

$$r_{max}(\theta,\varphi) = cos(\varphi)cos(\theta)d_x + cos(\varphi)sin(\theta)d_y + sin(\varphi)d_z \tag{17}$$

We show in [2], with similar computations as in the 2D space, that f_{avg} can be expressed as follows:

$$f_{avg} = \frac{2}{\pi}\int_0^{\frac{\pi}{2}}\int_0^{\frac{\pi}{2}} \frac{bc\cos(\varphi)\cos(\theta)d_x + ac\cos(\varphi)\sin(\theta)d_y + ab\sin(\varphi)d_z}{d_x\cos(\varphi)\cos(\theta) + d_y\cos(\varphi)\sin(\theta) + d_z\sin(\varphi)}\cos(\varphi)d\varphi d\theta$$

The relation above has the form:

$$f_{avg} = \frac{2}{\pi}\int_0^{\pi/2}\int_0^{\pi/2} \frac{Abc + Bac + Cab}{A + B + C}\cos(\varphi)\ d\varphi\ d\theta$$

We deduce immediatly that the relations (1) and (2) are validated.

Softwares specialized in formal computation did not succeed in integrating symbolically these formulae. Thus, we cannot give the analytical expressions of α, β and γ for arbitrary values of d_x, d_y and d_z. On the other hand, a numerical integration gives a good approximation when the cell size is given. For example we will use for α the following expression:

$$\alpha = \frac{2}{\pi}\int_0^{\pi/2}\int_0^{\pi/2} \frac{(\cos(\varphi))^2\cos(\theta)\ d_x\ d\varphi\ d\theta}{d_x\cos(\varphi)\cos(\theta) + d_y\cos(\varphi)\sin(\theta) + d_z\sin(\varphi)}$$

4.3 Experimental results

The results presented in this section compare the number of points extracted from our images to the estimation obtained by our model. We choose four different discrete images for our experiments. The first two are synthetic examples mathematically generated: a sphere discretized with regular samples leading to cubic voxels, and a torus with anisotropic discretization, where voxels have $2 \times 3 \times 1$ ratio. The last two are CT scans of a head ($256 \times 256 \times 113$) and of a wrist ($512 \times 512 \times 20$). The voxels of these two medical image have respectively $1 \times 1 \times 2$ and $1 \times 1 \times 11.6$ ratio. The numerical evaluations of our formulae are presented in the table 1.

	sphere	torus	CT skull	CT wrist
α	1/3	≈ 0.34	≈ 0.26	≈ 0.12
β	1/3	≈ 0.47	≈ 0.26	≈ 0.12
γ	1/3	≈ 0.19	≈ 0.48	≈ 0.76

Table 1. Numerical evaluation of α, β and γ.

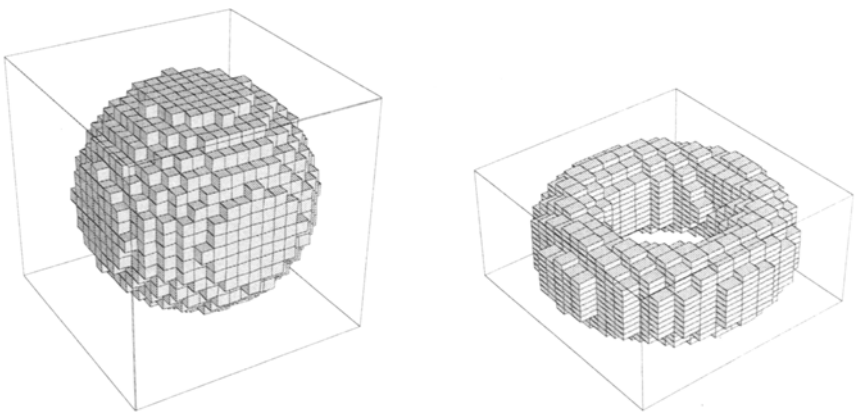

Fig. 5. Discrete sphere and torus used for our experimentations

The experimentations performed on our test images are presented in Table 2. The three columns contain respectively the subdivision factors used, the number of the extracted points on the iso-surface, and the estimation of this number given by relation (1). Our estimation is very accurate since the maximal relative error is about 2%.

sphere					torus					CT skull					CT wrist				
a	b	c	$S(a,b,c)$	$\tilde{S}(a,b,c)$	a	b	c	$S(a,b,c)$	$\tilde{S}(a,b,c)$	a	b	c	$S(a,b,c)$	$\tilde{S}(a,b,c)$	a	b	c	$S(a,b,c)$	$\tilde{S}(a,b,c)$
1	1	1	1448	1448	1	1	1	1512	1512	1	1	1	291005	291005	1	1	1	395642	395642
2	2	2	5792	5791	1	2	3	5632	5796	1	1	2	439080	432327	1	1	2	482625	485848
4	4	8	38368	38609	4	3	2	12132	12216	1	1	4	734993	744973	1	1	4	661062	666340
8	8	8	92120	92663	5	5	5	37228	37798	1	1	6	1030943	1047618	1	1	6	820327	812278
7	14	11	157780	158781	10	14	6	124076	126155	3	3	6	3920220	3980948	2	2	10	2979999	3026345

Table 2. Number of points on the iso-surface (subdivision parameters, measure and estimation)

These experiments validate our statistical study, which can predict precisely the memory space and the computing time associated to particular subdivision parameters.

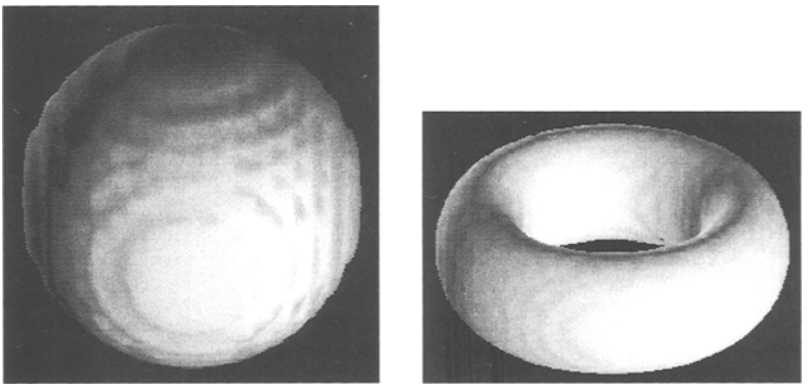

Fig. 6. Iso-surfaces extracted from a the sphere and a the torus.

5 Discussion

The expressions of the coefficients of our model are given only with integrals. Nevertheless these coefficients depend only on the voxel's proportions. They can be then tabulated for the most common proportions used in 3D imaging. The d_x and d_y dimensions are most often the same. The d_z dimension has typically a ratio between 1 and 10 as compared to d_x or d_y. In the case of $d_x = d_y = 1$ and $d_z \in [1, 10]$ we illustrate in figure 8 the evolution of the α, β and γ as functions of the cell height d_z. The evolution of these functions is smooth, so the tabulation method is very accurate.

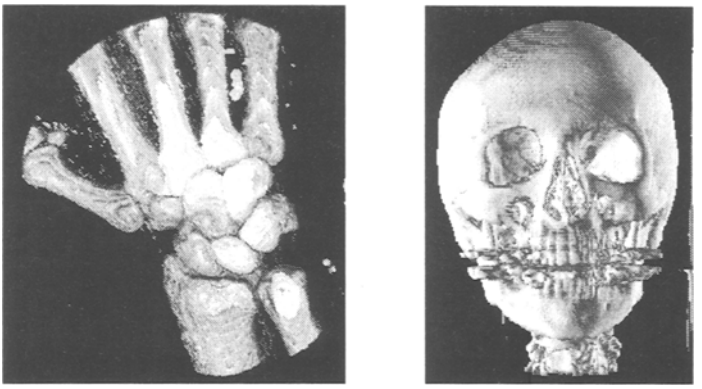

Fig. 7. Iso-surfaces extracting from the CT wrist and the CT skull.

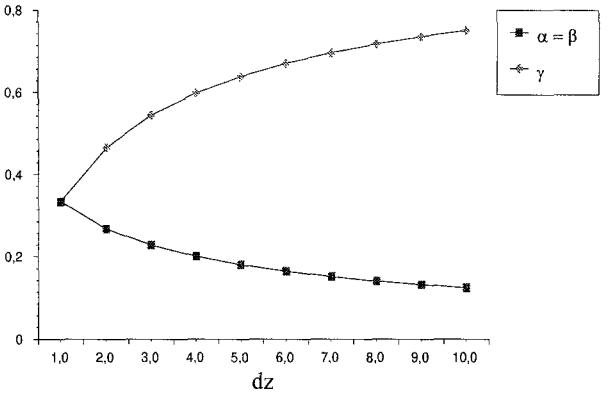

Fig. 8. Evolution of the α, β and γ coefficients as a function of the voxel's height d_z

6 Conclusion and Perspectives

We have shown how to predict the complexity of the produced iso-surface. Experiments on real and on synthetic images corroborate the validity of our formulations.

This work has been accomplished in the scope of a future parallelization of the algorithm on a distributed memory machine integrating dynamic load balancing. Indeed, the extremely simple data structure used to store the points of the surface allows to get a communication scheme much more effective than for surfacic algorithms. For an image animation, the visualization conditions vary slowly between two successive images, so we can take advantage of the frame-to-frame coherence: the points which have been handled by a processor for the production of an image are likely to be handled by the same processor for the following image, leading to a very low communication overhead.

References

1. Fatima Boumghar and Serge Miguet. Subdivision optimale dans l'algorithme des dividing-cubes. application à l'imagerie médicale 3D. In *3èmes Journées de l'Association Française d'Informatique Graphique (in french)*, pages 185–192. GAMSAU, Ecole d'architecture, Luminy, November 1995.
2. Fatima Boumghar, Serge Miguet, and Jean-Marc Nicod. Optimal subdivision and complexity of discrete surfaces in the dividing-cubes algorithm. Reseach report, Laboratoire de l'Informatique du Parallélisme, 46 Allée d'Italie 69364 Lyon Cedex 07, 1996. To appear in Sept.96.
3. Harvey E. Cline, C. R. Crawford, Wiliam E. Lorensen, S. Ludke, and B. C. Teeter. Two algorithms for the three-dimensional reconstruction of tomograms. *Medical Physics*, 15(3):320–327, may-june 1988.
4. D. Magid D.R. Ney, E.K. Fishman and R.A. Drebin. Volumetric rendering of computer tomography data: Principles and techniques. *IEEE Computer Graphics and Applications*, pages 24–32, March 1990.
5. H. Fuchs, Z. M. Kedem, and S. P. Uselton. Optimal surface reconstruction from planar contours. *Comm. of the ACM 20*, 10:693–702, Octobre 1977.
6. Arie Kaufman. *Volume Visualization.* IEEE Computer Society Press Tutorial, 1991.
7. E. Keppel. Approximation of complex surfaces by triangulation of contour lines. *IBM J. Res. Develop. 19*, pages 2–11, January 1975.
8. Marc Levoy. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, pages 29–37, May 1988.
9. William E. Lorensen and Harvey E. Cline. Marching cubes : A high resolution 3D surface construction algorithm. *ACM Computer Graphics*, 21(4):163–169, July 1987.
10. Serge Miguet. *Un Environnement Parallèle pour l'Imagerie 3D.* Mémoire d'habilitation à diriger des recherches (in french), Ecole Normale Supérieure de Lyon, December 1995.