Volumic Segmentation using Hierarchical Representation and Triangulated Surface

Jacques-Olivier Lachaud and Annick Montanvert

LIP, ENS-Lyon, 46, allée d'Italie, 69364 Lyon Cedex 7, France
e-mail: (jolachau, montanv)@lip.ens-lyon.fr

Abstract. This article presents a new algorithm for segmenting 3D images. It is based on a dynamic triangulated surface and on a pyramidal representation. The triangulated surface, which can as well modify its geometry as its topology, segments images into their components by altering its shape according to internal and external constraints. In order to speed up the whole process, the surface performs a coarse-to-fine approach by evolving in a specifically designed pyramid of 3D images.

1 Introduction

Volumic segmentation has become a major research topic in the last years. This is due to the appearance of 3D data in medical, geological or biological domains. This kind of data can come from either MR, tomography or confocal microscopy. Whereas 2D segmentation tries to mimic the vision process of the human being, volumic segmentation widens the detection of forms to the reconstruction of complex volumes, which is a difficult operation for our mind.

Unfortunately the analysis of 3D data and the detection of objects inside set a lot of additional problems, such as the control of objects with complex topology or the computational cost of the operations in a volumic space.

Hence the first purpose has been to evaluate the state of the art in order to develop a deformable and dynamical model that possesses a variable topology. Then, in order to overstep the scope of classical segmentation and to speed up the process, we associate a scalar continuous field with the image and we introduce the notion of pyramids composed of 3D images. We have tested our model and we have measured the savings given by the use of the hierarchical approach.

2 Deformable Surfaces

A deformable model is a model that follows a general principle: the object is deformed until it minimizes an energy function. This can be done either by a direct computation (least-square method for instance) or by the repeated application of constraints on the model. We have discarded models based on quadrics [12] and derivatives, for they can't manage complex topology objects, and models based on implicit surfaces (*blobs* [14] or front propagation [9]), because their computationnal cost is often heavy.

We prefer instead models of deformable meshes, such as cubic splines [7] or triangulated surfaces [10] which carry out segmentation by constraining the model on its vertices.

Considering the potentiality of the combination of meshes with some characteristics of bi-dimensionnal *snakes* [4], such as local minimization and inner constraints, we have chosen the last approach, the triangulated mesh, in order to segment 3D images. Quickness ($\mathcal{O}(N^2)$ vertices for a volumic image with edges of N voxels), rendering, are its main advantages. To these points we can add the opportunity of extracting features of the object, such as the area and the volume defined by the object, moments and topological informations.

3 Description of the Model

We define our surface as a closed oriented triangulated mesh. With this definition the surface always represents the boundary of a real volume. Our surface may thus be composed of several connected components, all closed and oriented.

Physical Aspect. The mesh is assimilated to a dynamic system of particles [11], which are the vertices of the triangles. Practically, interactions between particles occur only between direct neighbours. This neighbourhood allows us to define two internal constraints, the surface tension $\mathcal{F}_c$ and the surface elasticity $\mathcal{F}_e$, which follow the action/reaction principle. We transform the discrete image into a continuous scalar field (see section 4.1), in order to express the image influence through two external constraints: the force $\mathcal{F}_i$ which search for an isopotential surface and the force $\mathcal{F}_{di}$ which is a classical gradient descent.

The algorithm carrying out the displacement of the surface can be summarized into an iteration of the following steps:

1. Computing of internal and external forces for all vertices.
2. Re-sampling of the time scale to limit the vertex displacements.
3. Application of the Dynamic Fundamental Law for each vertex.
4. Effective displacement of the vertices.

We emphasize that this process expresses only the geometrical displacement of the surface and not the intrinsic topological modifications.

Geometrical Aspect. Meshes tend to intersect when they evolve. [5] introduces a global invariant δ which bounds the minimal and maximal sizes of each edge. By this way, local geometrical modifications are made easier and we can detect collisions by tests over vertex distance. Let us recall the two geometrical constraints induced:

$$\forall(U,V),\text{ if neighbours: } \delta \leq \|\overrightarrow{UV}\| \leq 2.5\,\delta,\ \text{ and if not: } \frac{2.5}{\sqrt{3}}\,\delta \leq \|\overrightarrow{UV}\| \qquad (1)$$

Topological breaks are controlled via Euler-Poincaré's characteristic χ [3]. Note that the different topology changes of the surface correspond to a modification of 2 or -2 of χ. Additionnal informations may be found in [6].

Initialization of the triangulated mesh. The triangulated surface is initialized with one icosahedron embracing the volumic image in real coordinates or with several icosahedra scattered in the image. The surface is then globally divided (see section 5.1) until it follows our geometrical constraints. After that, the surface is free to evolve according to its dynamic and geometrical rules.

4 Image Workspace and Pyramids

4.1 Transformation toward a Continuous Scalar Field

Let $I(i,j,k)$ be our discrete image of size $M \times N \times P$. Let μ, ν, π be its real size: it corresponds to the volume really occupied by the image in space. We determine the continuous potential function $\Pi_I(x,y,z)$, $(x \in [0,\mu[, y \in [0,\nu[, z \in [0,\pi[)$, by a first degree interpolation of $I()$ from space M, N, P to space μ, ν, π.

In this way we obtain a continuous scalar field that allows the computation of $\mathcal{F}_i$, but which is not derivable everywhere. Therefore $\mathcal{F}_{di}$ is computed by interpolating the discrete gradient deduced from the image. The computed scalar field can be interpreted as a stack of isopotential surfaces. Note that the interpolation degree has no influence about this fact and first degree is sufficient for an isopotential surface tracking.

4.2 Multi-scale Approach with 3-D Pyramids

Direct approach of image segmentation is not totally satisfactory. The influence of the image is indeed localized around vertices and makes sense only if the mesh has the same preciseness than the resolution of the 3D image. Common solutions either use a mesh with a refinement comparable to the one of the image, or take an interest in a wider area around each vertex. Hence the computational cost is deeply proportionnal to the image size. Therefore, our approach is to compute once and for all the influence of the image areas at different scales. This mixed solution can be done by the computation of a 3D image pyramid.

We take a particular interest in pyramids of frequency decomposition [1]: they provide a set of images at decreasing resolutions and details. These pyramids create no wrong contours, hence the model can exploit the results obtained at coarse levels in order to start the calculation on a finer level with more efficiency.

Their successive levels are computed by the convolution of a Gaussian kernel of side 5 voxels. It guarantees a low cost filtering without phase translation linked to a reduction factor of two for each image dimension [2]. Let G_0 be the initial 3D image and the base of the pyramid. The computation of G_{h+1} according to G_h (image of level h in the pyramid) is given by the discrete convolution formula:

$$G_{h+1}(i',j',k') = \sum_{m=-2}^{2} \sum_{n=-2}^{2} \sum_{p=-2}^{2} \omega(m,n,p) \cdot G_h(2i'+m, 2j'+n, 2k'+p) \quad (2)$$

where ω is a Gaussian convolution kernel of size 5 voxels: $(\frac{1}{16}[1\ 4\ 6\ 4\ 1])^3$.

4.3 3-D Image Pyramids of any Reduction Factor

The previous formulation (2) is not usable as it is, because we have to take into account two major constraints within our context:

- voxels are not bound to be cubic (sampling frequencies are highly dependent of the acquisition means and are not identical in the general case),
- the reduction factor of the re-sampling must be coherent to the surface representation.

We will realize the convolution operations efficiently, by defining a real work-space corresponding to the discrete structure including the initial data.

Our goal is to determine a list of discrete images, denoted $G_0, \ldots, G_{max}$, and which represents the pyramid. G_0 is the initial image (given for segmentation) of sizes M, N and P. It is the image that possesses the greatest amount of informations. G_{max} will be the image that includes only the lowest frequencies. Let M_h, N_h and P_h be the sizes of the discrete image G_h. Their values are still unknown. Let $I_h = M_h \times N_h \times P_h$. With these definitions a discrete image G_h is a function from I_h toward $[0, 1]$.

We denote I_R the space defined by the real image of size $[0, \mu[\times[0, \nu[\times[0, \pi[$. Because all images G_h represent at different scales the same real image, all of them have a real size of μ, ν, π. The immersion of a voxel (i, j, k) of G_h into the real image space I_R is given by the transformation $\mathcal{T}_h$ as follows:

$$\begin{aligned} \mathcal{T}_h \; : \quad I_h \quad &\longrightarrow I_R \\ (i,j,k) &\longrightarrow \left(i\tfrac{\mu}{M_h}, j\tfrac{\nu}{N_h}, k\tfrac{\pi}{P_h}\right) \end{aligned} \tag{3}$$

We call *unit* of the real space the value $U_h = \min(\mu/M_h, \nu/N_h, \pi/P_h)$. It's the smallest distance between the immersions of two voxels in the real image. In the case of an isotropic image, we got $U_h = \mu/M_h = \nu/N_h = \pi/P_h$.

A reduction factor is needed in order to build the successive pyramid levels. Being for the moment unpredictable (see section 5.1), our pyramid construction must authorize any reduction factor. Unlike purely discrete formulations, the transformation into a continuous image (see section 4.1) associated with our immersion process allows us to build pyramids of any factor. The chosen kernel is of side 5, therefore the reduction factor T must be less than two.

Let V_0 be the base of our pyramid of real images. V_0 is given by the immersion then by the interpolation of the discrete data. As a matter of fact $V_0 = \Pi_{G_0}$. Let V_h be the level h of the real image pyramid. V_{h+1} is calculated from V_h. The number and the localization (in I_R) of the points to be calculated are determined by the reduction factor T, and the values are obtained after convolution of some points of V_h. Their storing after computation on the real image space is of course done in an array of voxels, assimilated to the discrete pyramid (G_i) at level $h+1$.

The discrete sizes M_h, N_h, P_h and the measure unit U_h correspond to a real image V_h. Its characteristics are defined recursively with:

$$\begin{array}{llll} M_0 = M & N_0 = N & P_0 = P & U_0 = \min(\mu/M, \nu/N, \pi/P) \\ M_{h+1} = \lfloor \frac{M_h}{T} \rfloor & N_{h+1} = \lfloor \frac{N_h}{T} \rfloor & P_{h+1} = \lfloor \frac{P_h}{T} \rfloor & \quad U_{h+1} = U_h \cdot T \end{array} \tag{4}$$

Let $R(i, j, k)$ be a voxel of the discrete data of G_{h+1}. Its immersion R_V in the real image V_{h+1} has coordinates of $\mathcal{T}_{h+1}(i, j, k)$ (see figure 1a). In order to establish the value of R, the convolution operation is defined over points of V_h. The central point has the same position in V_h and in V_{h+1}. The localization of the other points involved in the convolution is determined via the use of the unit U_h to discretize V_h around the point R_V (see figure 1b).

G_h is known, V_h is defined by Π_{G_h}. We obtain:

$$G_{h+1}(i,j,k) = \sum_{m=-2}^{2} \sum_{n=-2}^{2} \sum_{p=-2}^{2} \omega(m,n,p)\, V_h[\mathcal{T}_{h+1}(i,j,k) + (mU_h, nU_h, pU_h)] \tag{5}$$

G_{h+1} defined then V_{h+1} implicitly ($V_{h+1} = \Pi_{G_{h+1}}$).

Because of the unknown reduction factor, the 5^3 points involved in the convolution do not coincide with given points of G_h in the general case (see figure 1c). Moreover there usually won't be any cover between points involved in two neighbouring convolutions. Besides, each point of V_h compulsory for the convolution computation is interpolated from 8 data points of V_h (so stored in G_h) which form the parallelepiped containing this point (see section 4.1 and figure 1c too).

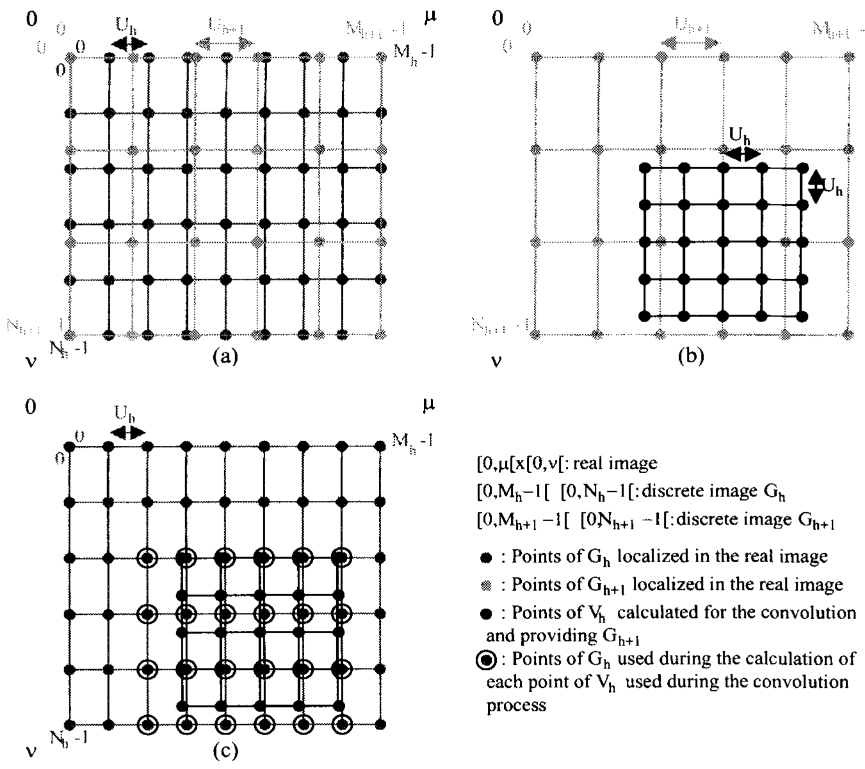
(a)
(b)
(c)
0
μ
v
U_h
U_{h+1}
M_h-1
M_{h+1}-1
N_h-1
N_{h+1}-1
[0,μ[x[0,v[: real image
[0,M_h−1[[0,N_h−1[:discrete image G_h
[0,M_{h+1}−1[[0,N_{h+1}−1[:discrete image G_{h+1}
● : Points of G_h localized in the real image
● : Points of G_{h+1} localized in the real image
● : Points of V_h calculated for the convolution and providing G_{h+1}
◉ : Points of G_h used during the calculation of each point of V_h used during the convolution process

Fig. 1. View of a convolution computation in 2-D: (a) The two superposed levels G_h and G_{h+1}, (b) computation of level G_{h+1} together with the localization of the convolution mask applied on one point, (c) application of the convolution mask over level G_h and visualization of the discrete points of G_h involved in the computation.

The Gaussian convolution kernel (of size 5^3) is applied successively along the three dimensions because of its separability. We can estimate the optimization savings by using the following notations and equations: Let T be the reduction factor, t_1 the execution time of a classical algorithm, t_2 the execution time of the optimized algorithm. A short calculation gives:

$$\frac{t_2}{t_1} = \frac{1}{25}(T^2 + T + 1) \quad \text{if the Gaussian kernel size is } 5^3 \tag{6}$$

The optimized algorithm is thus faster for $0 \leq T \leq \frac{-1+\sqrt{97}}{2}$ (≈ 4.42).

5 Segmentation

5.1 Image/Model Appropriateness

In section 4.2 we have chosen to express surface-image interaction with constraints locally computed around each vertex. In order to obtain a correct appropriateness between the surface and the pyramid images, we must first examine the relations linking the model preciseness to the resolution of an image, and secondly, we must establish an algorithm of surface refinement, which ensures the appropriateness surface-image during the whole coarse-to-fine process.

Surface-Image Relationships. According to (1) we have $d_{min} = \delta$ and $d_{max} = 2.5\,\delta$. The refinement can so be defined entirely with the invariant δ. Let δ_h be the invariant δ at level h of the pyramid. d^h_{min} and d^h_{max} are defined as same.

The image resolution is linked to the unit U_h (see section 4.3). We can deduce the relationships between δ_h and U_h with the help of the following considerations:

1. an edge may represent a contour formed by two 6-connected voxels (distant of U_h), so $d^h_{min} \leq U_h$,
2. an edge may represent a contour formed by two stricly 26-connected voxels (distant of $\sqrt{3}\,U_h$), so $d^h_{max} \geq \sqrt{3}\,U_h$.

$$\text{Hence, } \frac{\sqrt{3}}{2.5} \leq \frac{U_h}{\delta_h} \leq 1 \quad \text{with } U_h = \min\left(\frac{\mu}{M_h}, \frac{\nu}{N_h}, \frac{\pi}{P_h}\right) \tag{7}$$

Surface Refinement. The surface works in an image pyramid and, consequently, must refine its mesh every time it goes down a level of the pyramid. We propose a process refining triangulated surface with a factor K. This factor will determine the reduction factor T of the pyramid.

Refinement process (or global surface division) (see figure 2):

1. in a first scan, a new vertex is created in the center of each facet of the model; the vertex is connected to the three vertices that delimit its facet,
2. in a second scan, the edges, which link together the old vertices (those which were not created during the first pass), are reversed in order to regularize edge lengths in a systematic way.

Such an algorithm reduces the average edge length to $1/\sqrt{3}$ of the old one. We may so apply to the invariant a reduction factor whose value is also $\sqrt{3}$, so $K = \sqrt{3}$. In order that inegality (7) be respected at the initialization moment and during all successive pyramid levels, an identical reduction factor is chosen for the pyramid construction:

$$T = K = \sqrt{3} \text{ and } \forall(h = 0 \ldots h_{max} - 1), \begin{cases} \delta_{h_{max}} = \delta_{init},\ \delta_h = \delta_{h+1}/K \\ U_0 = U,\ U_{h+1} = U_h \cdot T \\ \text{thus we got } \frac{\sqrt{3}}{2.5} \leq \frac{U_h}{\delta_h} \leq 1 \end{cases} \tag{8}$$

At the initialization moment, a bubble or a set of bubbles, whose invariant δ_{init} is consistent with (7) at level h_{max}, are created. After that, the recursive process described by (8) guarantees a correct surface-image appropriateness, whichever are the iteration or the current level in the pyramid.

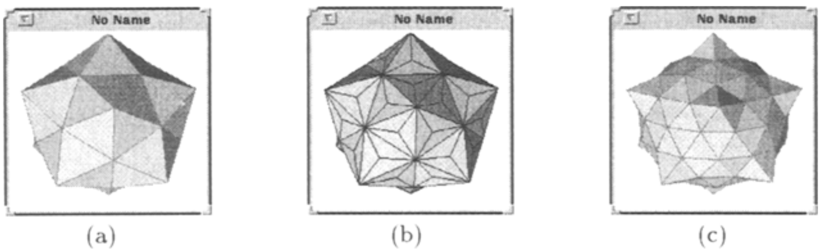

Fig. 2. Example of a global division process over a polyhedron with sixty facets: (a) before global division, (b) after first pass, (c) after second pass.

5.2 Anisotropy during segmentation

During the segmentation process, the edges have to keep their meaning with regards to a voxel space deformed by a possible anisotropy. So we make the surface evolve in a real space derived from the space (μ, ν, π) by affine transformation such that it has the same proportions than the discrete image it interpolates (its sizes are thus $(\lambda\ M_h, \lambda\ N_h, \lambda\ P_h)$. The internal forces have a slightly different behaviour than the one they would have in the real physical space. An alternative was to equip the real space with an anisotropic metrics in order to obtain a real physical behaviour, but such a work would not have been relevant in the context of segmentation.

6 Results

We first test our model on a volumic image of a human skull [1] of discrete sizes $256 \times 256 \times 68$ and of real sizes $1.0 \times 1.0 \times 1.0625$. The model segments the image by searching an isopotential surface. We give the surface some inner constraints to smooth the result. We provide our process with a full reliable heuristic that quickens the treatment of motionless or quasi-motionless vertices. The segmentation process is run two times for comparison purposes:

- First a direct processing on the image without any pyramid is shown on figure 3: the surface slowly sticks on the outer part of the skull and then goes inside to segment its inner part (orbits of the eyes, brain cavity, ...). The surface needs more than 400 iterations to meet equilibrium.
- We run after the process by making use of the pyramid built up from this volumic image with a reduction factor of $\sqrt{3}$. The process waits for its complete convergence at one pyramid level before going down one level. Figure 4 represents the coarse-to-fine evolution of the surface in the image pyramid.

Figure 5 analyzes the behaviour of both algorithms. The one of the classical segmentation algorithm is quite simple: the kinetic energy curve shows the slow segmentation convergence (see figure 5b), the number of vertex (see figure 5d) and the average edge length (see figure 5c) are subject to few changes, the time cost (see figure 5a) slowly decreases but only because of the use of the heuristic.

[1] Thanks to Yves Usson (C.H.U. Grenoble) for the volumic database.

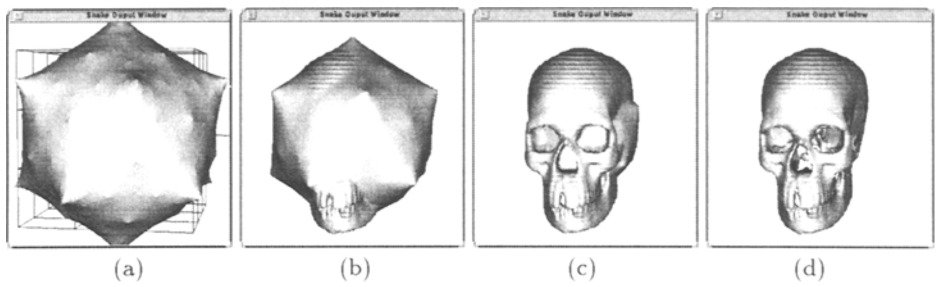

Fig. 3. Surface evolution during a segmentation without pyramid: (a) iteration 0 on image G_1, (b) iter. 100 on image G_1, (c) iter. 250 on image G_1, (d) iter. 700 on image G_1.

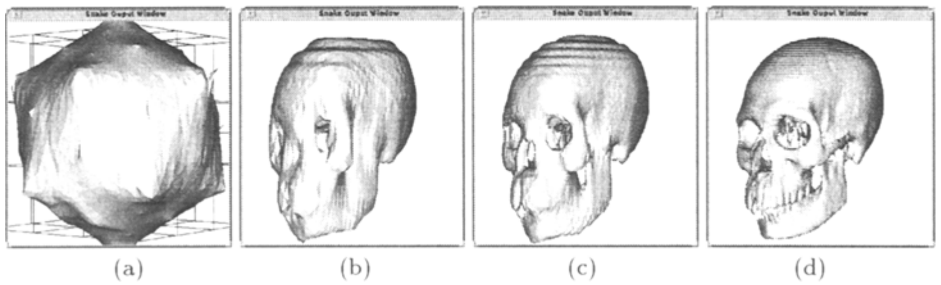

Fig. 4. Surface evolution during a pyramidal segmentation: (a) iteration 0 on image G_3, (b) iter. 225 on image G_3, (c) iter. 333 on image G_2, (d) iter. 461 on image G_1.

The behaviour of the pyramidal segmentation algorithm displays the four used pyramid levels: the number of vertex (see figure 5d) and the average edge length (see figure 5c) show that the triangulated mesh has gone down a level at the iterations 226, 334 and 462. The kinetic energy evolution (see figure 5b) highlights the convergences at each step and the duration graph of each iteration (see figure 5a) explains the time savings provided by a coarse-to-fine process. The final surface has 1 connected component and 23 topological holes.

We then test our model in a more problematic case: a phase contrast MR angiographic image [2]. Its discrete sizes are $256 \times 256 \times 124$. We can observe that such images are mainly composed of vessels whose proportions are not suited for a pyramidal representation. Within this context, the top image represents nearly nothing and the initialization of the process is very bad. Nevertheless the model succeeds in following the vessels to recover the forgotten ones as shown by figure 6, but each level needs more time to converge than for the first example.

7 Conclusion

We have designed and developed an efficient volumic segmentation algorithm by means of a deformable triangulated surface evolving in a 3D image pyramid. The obtained results show that their quality is identical to other similar algorithms;

[2] Acknowledgements to the UMDS Image Processing Group, London, for the angiographic image.

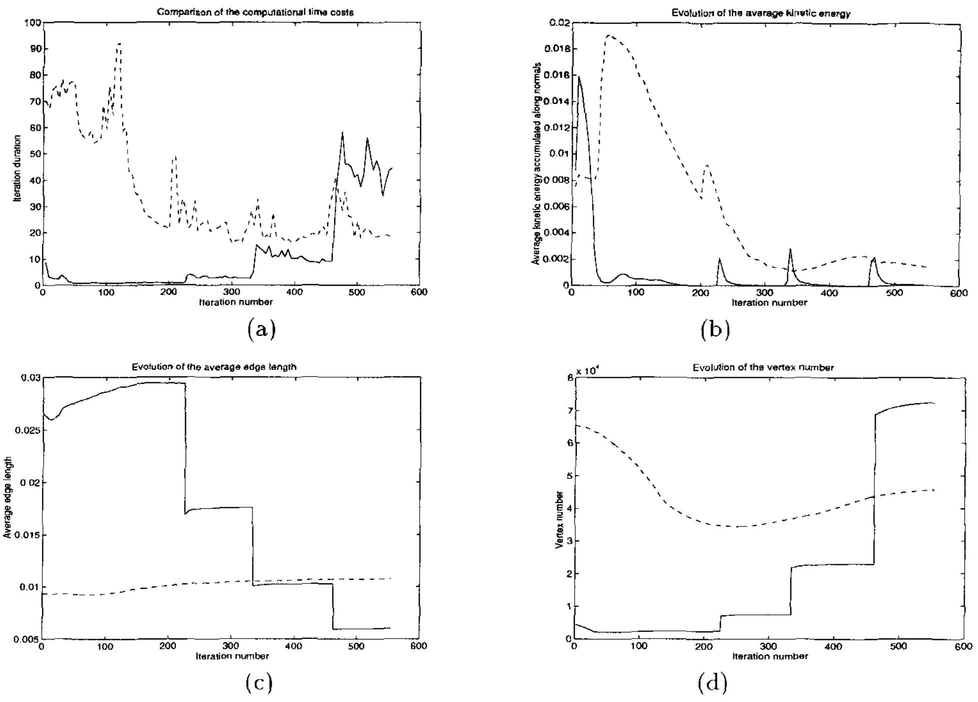

Fig. 5. Statistics over a skull segmentation process and comparison between the approach without a pyramid (in dotted line) and the pyramidal approach (in solid line): (a) duration of each iteration, (b) evolution of the average kinetic energy accumulated along the surface normals, (c) average edge length according to the iteration, (d) number of vertex of the surface according to the iteration.

the tests demonstrate the efficiency and the quickness of our algorithm. Several points may be nevertheless explored:

- The first convergence step can be greatly speeded up if the surface initialization contains more information. A marching-cube process [8] applied on the highest level of the pyramid would give a better first approximation. However surfaces obtained by this process are generally not closed.
- The convergence may be also guided by some new constraints, for instance by introducing attractive forces generated either by *sure* points of the object or by particular edges detected during a pre-treatment.
- The overall speed can be improved by accelerating the detection of topological breaks. For instance the use of some efficient algorithms [13] would reduce the cost of self-collision detection, which is for the moment within $\mathcal{O}(n \cdot \log(n))$ if n is the number of vertex.

References

1. P. J. Burt. "Fast filter transforms for image processing". *Computer Graphics and Image Processing*, 16:20–51, January 1981.

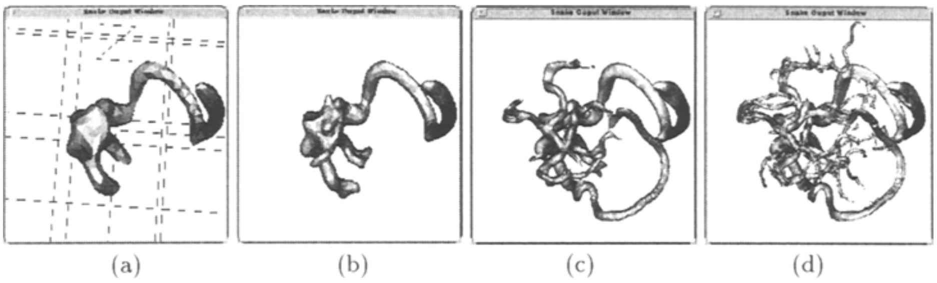

Fig. 6. Surface evolution during the segmentation of an angiographic image with pyramid: (a) After convergence on image G_3, (b) After convergence on image G_2,(c) After convergence on image G_1, (d) Final result

2. A. CHÉHIKIAN. "Algorithmes optimaux pour la génération de pyramides passes-bas et laplaciennes". *Traitement du Signal*, 9:297–308, January 1992.
3. H.B. GRIFFITHS. *"Surfaces"*. Cambridge University Press, January 1976.
4. M. KASS, A. WITKIN, and D. TERZOPOULOS. "Snakes: active contour models". In *1st Conference on Computer Vision*, Londres, June 1987.
5. J.O. LACHAUD and A. MONTANVERT. "Volumic Segmentation using Hierarchical Representation and Triangulated Surface". Research Report 95-37, LIP - ENS Lyon, France, November 1995.
6. J.O. LACHAUD and A. MONTANVERT. "Segmentation tridimensionnelle hiérarchique par triangulation de surface". In *10ème Congrès Reconnaissance des Formes et Intelligence Artificielle*, January 1996.
7. F. LEITNER and P. CINQUIN. "Complex topology 3D objects segmentation". In *Advances in Intelligent Robotics Systems*, volume 1609 of *SPIE*, Boston, November 1991.
8. W. E. LORENSEN and H. E. CLINE. "Marching Cubes: A High Resolution 3D Surface Construction Algorithm". *Computer Graphics*, 21:163–169, January 1987.
9. R. MALLADI, J. A. SETHIAN, and B. C. VEMURI. "Shape Modelling with Front Propagation: A Level Set Approach". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):158–174, February 1995.
10. J.V. MILLER, D.E. BREEN, W.E. LORENSEN, R.M. O'BARNES, and M.J. WOZNY. "Geometrically deformed models: A method for extracting closed geometric models from volume data". *Computer Graphics*, 25(4), July 1991.
11. R. SZELISKI and D. TONNESEN. "Surface Modeling with oriented Particle Systems". Technical Report CRL-91-14, DEC Cambridge Research Lab., December 1991.
12. D. TERZOPOULOS and A. WITKIN. "Deformable Models: Physically based models with rigid and deformable components". *IEEE Computer Graphics and Applications*, 8(6):41–51, November 1988.
13. P. VOLINO and Thalmann N. MAGNENAT. "Efficient self-collision detection on smoothly discretized surface animations using geometrical shape regularity". In *Eurographics'94*, volume 13(3), September 1994.
14. R.T. WHITAKER. "Volumetric deformable models: active blobs". In *VBC*, volume 2359 of *SPIE*, pages 122–134, March 1994.