Large deformable splines, crest lines and matching

André Gueziec

INRIA

UNITÉ DE RECHERCHE
INRIA-ROCQUENCOURT

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P 105
78153 Le Chesnay Cedex
France
Tél. (1) 39 63 55 11

Rapports de Recherche

1992

25ème anniversaire

N° 1782

Programme 4
Robotique, Image et Vision

LARGE DEFORMABLE SPLINES, CREST LINES AND MATCHING

André GUÉZIEC

Octobre 1992

Large Deformable Splines, Crest Lines and Matching *

André Guéziec

November 9, 1992

INRIA
BP 105 78153 Le Chesnay Cédex
e-mail: Andre.Gueziec@inria.fr

Programme 4
Robotique, Image et Vision

*This work was supported in part by a grant from Digital Equipment Corporation.

Abstract

We present new deformable spline surfaces for segmentation of 3–D medical images. We explore parametric surfaces of the form $\mathbf{x}(u, v)$ with two different topologies, planar and cylindrical, that permit to segment fine anatomical structures. With respect to earlier approaches by Cinquin's team [LMLC91] in Grenoble or Médioni's team in U.S.C. [MSMM90] and also by Cohen et al. [CCA92] or Kass et al. [KWT87] that minimize the "energy" of a deformable surface in a potential field, we perform this optimization with successive approximations of dense data, and propose the key following improvements:

- We show that the Euler equation has a closed form solution in a quadratic potential field. Each approximation requires only one iteration.
- We use tensor products of splines to solve independently the system along parameters u and v. This enables us to work with large meshes of control vertices, e.g. *10,000 vertices* and more.
- With a regularly sampled potential field, each point in the same image voxel in processed in the same way. We use a *continuous* potential field defined with 3–D volumetric splines to avoid this problem.

When the deformation process stops, we end up with a smooth differentiable surface where we measure principle curvature and directions. We describe next an original algorithm that extracts lines of extremal curvature on the surface. We present experimental evidence with real medical images that illustrate all the previous points.

Finally we give two marginal contributions. We outline the *spherical* topology for spline surfaces. We use Ostrogradsky's formula to compute the exact volume bounded by such a surface. We also give some hints about the tricky topic of avoiding self-intersections.

Modèles Déformables à Partir du Produit Tensoriel de Fonctions Splines

Résumé : *Nous présentons un nouveau modèle de surface déformable pour segmenter des images médicales tridimensionnelles. Nous utilisons des surfaces paramétriques* $\mathbf{x}(u, v)$ *avec des topologies simples : topologie plane, cylindrique, torique et sphérique. Par rapport aux approches précédentes de l' équipe de Cinquin à Grenoble [LMLC91], de l' équipe de Médioni à U.S.C. [MSMM90], de Cohen et al. à Paris [CCA92] ou de Kass et al. [KWT87] où l'on minimise l' énergie d'une surface déformable dans un champ de potentiel, nous résolvons le problème d'optimisation qui en découle à l'aide d' approximations successives de données denses, et nous proposons les améliorations suivantes :*

- *Nous montrons que l' équation d'Euler associée au problème d'optimisation peut être à chaque fois résolue de manière non itérative, pourvu que le potentiel puisse être exprimé localement comme une forme quadratique des paramètres de la surface (points de contrôle).*
- *Nous définissons la surface comme produit tensoriel de fonctions spline, et résolvons indépendemment notre système le long des paramètres u et v. Nous pouvons ainsi utiliser de larges maillages de points de contrôles, e.g.* 10 000 points *et davantage.*
- *Avec un champ de potentiel échantillonné régulièrement, tous les points appartenant au même voxel sont traités de manière identique. Nous utilisons un champ de potentiel* continu *défini à l'aide de fonctions splines volumiques (voir figure 5).*

La surface obtenue après déformation est différentiable, et nous mesurons continûment courbures et directions principales. Nous décrivons un procédé original pour tracer des lignes de crêtes sur la surface, et présentons des résultats expérimentaux sur des images médicales.

Enfin, nous apportons deux contributions marginales : Nous décrivons la topologie sphérique *pour des surfaces splines et utilisons le théorème d' Ostrogradsky pour calculer le volume de surfaces splines fermées. Nous examinons aussi le problème des auto-intersections de surfaces.*

1 Introduction

The problem of segmenting and registering 3−D medical images is very important to physicians. For instance, information contained in X-ray Scanner or MRI images must be extracted and fused to help physicians for diagnosis or surgical planning. Research in 3−D image processing has achieved considerable progress in this attempt. In particular, it is now possible to register fully automatically two 3−D images of the same patient in the same modality, with the help of stable feature curves (see [TGM+92]). These characteristic curves could even permit a standard approach to observe the same individual in time, to compare different individuals [Cut89], or to compute statistics.

Our work is the continuation of the work accomplished by Monga [MBF92], Thirion [TG92] or Cohen et al. [CCA92] that used differential geometry of surfaces to extract robust and useful features from 3−D volumes of data, and also features that are invariant to some classes of deformations.

In our formulation of the problem, the segmentation with deformable surfaces is decomposed into two parts: Part (1) consists in characterizing contour points and computing a $3-D$ distance potential. When done with a sequential machine, this operation may take prohibitive computing time. Part (2) consists in piloting a parametric surface in the $3-D$ potential towards zeros of the potential. We concentrate on part (2).

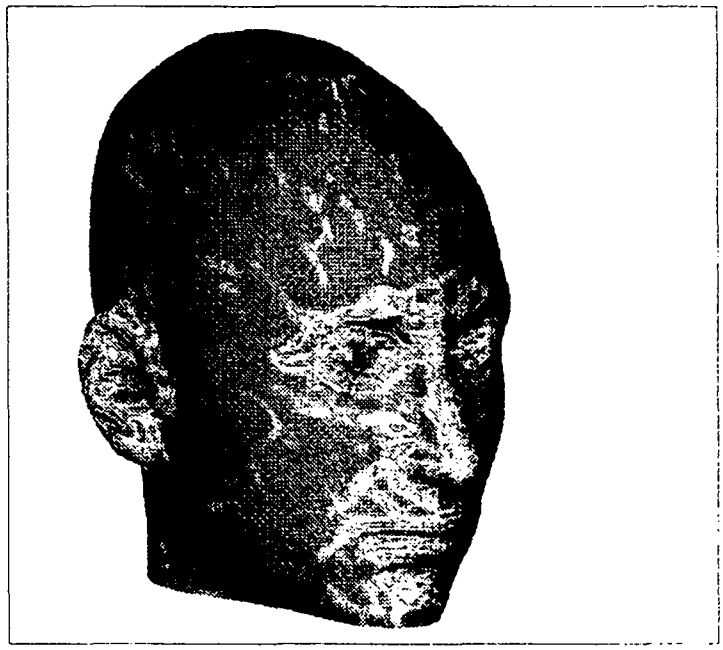


Figure 1: After 14 iterations with a surface that counts finally as many as 256 · 128 control vertices and 280 · 160 sample points (see section2.6.2), the segmentation is exhaustive, and we represent the "global curvature" $k_1^2 + k_2^2$ that measures the density of bending energy accumulated. The color ranges from blue to red in increasing magnitude of the curvature. For this really huge model, the computing time is still tolerable: with a Dec 5900, one iteration takes in average 42 sec. CPU.

The advantages of deformable surfaces for segmentation are the following: (i)Coutour

points belong necessarily to a unique surface, and the topology of the resulting surface is simple. For instance, These are key features when dealing with the segmentation and the representation of the human face; (ii)The lack of information (contour points) in small regions can be overcome by the regularity of the surface; (iii)Differential structures, as well as the connectivity between these structures, can be easily inferred from the surface representation.

In Section 2, we present our original contribution in detail with copious experimental evidence. We see the surface deformation process as a sequence of least squares *approximations of dense data*. To our knowledge, this approach is original, even though it is very related to earlier works, especially from Cinquin's team [LMLC91] or Médioni's team [MSMM90], who pioneered in the development of spline deformable curves and surfaces. We thus design a very efficient algorithm which is able to deal with really *large* meshes with a moderate computational burden. The number of iterations is reduced thanks to a local quadratic fit of the potential field in which the surface evolves. After few iterations, the shape converged towards an acceptable solution, except from isolated points or regions. We exhibit as an example the segmentation of the face of G. Malandain (see section 2.6.1) which is nearly performed *real time*: one iteration takes *7.3 seconds* on a Dec 5900, including preprocessing (sections 2.4 and 2.5 provide a detailed analysis of the complexity). The speed of the method was not our primary concern here, and we concentrated on differentiability and ways to obtain a high resolution with a B-spline surface. However, we believe that real time deformable models will soon operate. Since differentiability is costly, we might use simple polygonal models for the convergence process and fit a spline when the polygonal surface is stabilized. Lastly, we introduce in section 2.6.2 an iterative refinement of the mesh that permits to obtain a higher degree of precision with less computations.

We take advantage of the *differentiability* of our model and compute curvature and crest lines that are invariants of the euclidian group and are thus used for matching. For instance, we represent in Fig. 1 the sum of squared principle curvatures or "global curvature" after only 15 iterations of our method with a very fine grid. We can also measure the evolution of the curvature as the surface deforms. Lines of extremal curvature of a surface were previously computed inside the volume of 3−D data [BMGA92] when the surface had an implicit equation. As an alternative, we can also draw the crest lines on a smooth parametric representation of the surface. We have developed an original algorithm for this purpose that we depict in section 3. This substitute algorithm is very promising for even surfaces as in Fig. 1. For particularly smooth surfaces, previous crest lines extraction software [TG92] is likely to produce unstable lines.

Next, we concentrate in section 4.1 on topologies that permit to describe closed surfaces i.e. toric topology and spherical topology. For these type of surfaces, we will see how to use Ostrogradsky's theorem to compute the exact volume that is bounded by the spline surface. This could be particularly useful for volume measurements of organs before the transplantation.

Finally, we see in section 4.2 how the precomputation of the potential image could be skipped. Note that this off-line computation and the storage of the resulting distance image make sense for other purposes, such as rigid and non-rigid matching of $3-D$ curves, surfaces and images. Instead, we will simulate the potential field on-line, i.e. determine on-line the best displacement field along the deformable surface normal. We will see that the dis-

placement field can constitute a *tubular neighborhood* of the surface under some assumptions related to the curvature of the surface. This will help us to avoid *self-intersections* of the parametric surface.

In summary, we significantly extend previous 3–D segmentation methods using deformable surfaces and give a complementary approach for the extraction and the matching of crest lines.

2 $3-D$ Segmentation with a Deformable Surface

We suppose that contour points are characterized with a contour detector such as Canny and Deriche's (see Monga, Malandain et al. [MDMC90]) and represented as points in a $3-D$ image. Further, a distance transformation (see Danielsson [Dan80]) maps the contour image to a distance image d that takes zero values on contour points. This distance image can be seen as representing a 3–D potential field d^2. This potential integrates elastic forces that would pull the surface towards the contour points as if springs were attached from each surface point to some contour point. In order to minimize the integral of this potential over the surface, several approaches exist, using finite differences or finite elements as well as physically based methods including modal analysis [PS91, NA92]. Note that analogies exist between different formulations, e.g. the stiffness matrix in the mechanical formulation is equivalent to the spline matrix A of section 2.3, thus a modal analysis would be possible using splines.

2.1 Energy minimization as an approximation problem

Let us write an "energy" term $\mathcal{E}$ for a surface plunged in a square-distance potential field. This energy includes a surface (or internal) energy $\mathcal{E}_s$ expressed with the surface $\mathbf{x}(u,v)$ derivatives. The first derivatives $\mathbf{x}_u$ and $\mathbf{x}_v$ measure the tension. The second derivatives $\mathbf{x}_{uu}$,$\mathbf{x}_{uv}$ and $\mathbf{x}_{vv}$ measure the bending. We believe that energy terms can be extended to higher derivatives, which measure less intuitive but equally interesting energies.

$\mathcal{E}$ also includes an external energy term $\mathcal{E}_e$, corresponding for instance to elastic forces that are attached between the surface and the contours. Although these forces are not known directly from the $3-D$ original image, they can be inferred from an euclidian-distance potential that gives everywhere the distance d between the surface and the contour. We have for instance, with the choice of a 4th order energy term which is classical for bicubic surfaces, by analogy with the bending energy for a rod in $1-D$ (the reader may consult for instance [Cin87]):

$$2\mathcal{E} = 2(\mathcal{E}_s + \mathcal{E}_e) = \tau \int_s \mathbf{x}_{uuvv}^2 + \int_s d^2$$

The objective is now to minimize $\mathcal{E}$, that is to solve an optimization problem. In order to perform this optimization, we distinguish between the following techniques: (i)Finite Elements: We derive first an euler differential equation. We project $\mathbf{x}$ on a limited number of finite support basis function and compute integrals of products involved in the euler equation for this functions. We end up with a linear system. (ii)Finite Differences: We also derive first an euler differential equation. We discretize the surface in samples $\mathbf{x_{ij}}$, we discretize

also the differential euler equation and obtain again a linear system. (iii)Approximation problem: We discretize the surface in samples $\mathbf{x_{ij}}$, and write a discrete energy term. We choose smooth approximating functions that will minimize the discrete energy. A new Euler equation corresponds to this discrete energy, provided we differentiate with respect to the coefficients of the functions. Schematically, we approximate integrals in (i) by Riemann sums.

Solution (iii) was not explored to our knowledge, yet it permits to combine advantages of finite differences (no integrals to compute over basis functions) and of finite elements (smoothness of the solution, which is also known between sample points, flexibility in the size of basis functions).

We thus take on solution (iii), and write the discrete energy term E as:

$$2E = \tau \sum_{ij} \mathbf{x}^2_{uuvv} + \sum_{ij} d^2 \tag{1}$$

If instead of d^2, we have a quadratic function of the surface point $\mathbf{x}$, i.e. $(\mathbf{x} - \mathbf{c}_{\text{extr}})^2$, $\mathbf{c}_{\text{extr}}$ extrapolated contour point from the knowledge of the potential, we can solve a penalized least squares problem, i.e. miminize:

$$\tau \sum_{ij} \mathbf{x}^2_{uuvv} + \sum_{ij} (\mathbf{x} - \mathbf{c})^2$$

Although the two notations d^2 and $(\mathbf{x} - \mathbf{c})^2$ lead to the same solution, they induce a great difference in the speed of the convergence. This very simple observation is a key point for the success and for the efficiency of our method. Fig 2 visualizes this difference.

With the first notation, we will need many euler steps to reach a minimum (Fig. 2a). In Appendix A we derive the formulation of the problem in this case which turns out to be the classical formulation. Linear systems could then be solved both implicitly or explicitly.

The second notation represents a *quadratic approximation of the potential*, and we will find a minimum in a reduced number of iterations (Fig. 2b). (By quadratic, we mean of course quadratic in terms of the surface parameters). We concentrate on this approach.

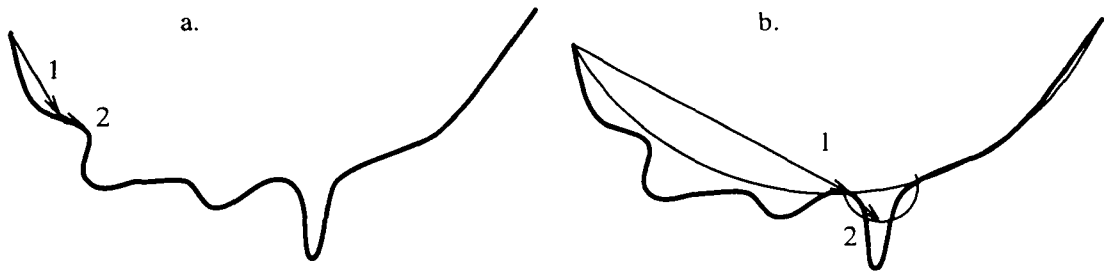


Figure 2: **a. left:** Classical euler steps to minimize a function.
b. Right: Accelerated convergence with successive quadratic approximations.

We believe that B-splines are especially well suited for solving such a problem, in virtue of their best and smoothest approximation properties (see De Boor [dB78]) and also because of their algorithmic advantages. In addition, B-splines functions have a finite support and thus

serve frequently as a basis for finite elements. For these reasons, we choose B-spline basis functions for the representation of the surface and we derive the Euler equation. Assuming a quadratic potential field, this equation has a closed form solution.

2.2 Spline Formulation is Most Desirable

People familiar with spline surfaces might skip this section. It is not in the scope of this article to give a survey on B-spline functions. Splines have been extensively studied and we recommend to consult the following bibliography: Ahlberg et al. [JAJ67], De Boor [dB78], Barsky et al. [BBB87], Farin [Far88], and Bohm et al. [BFK84] for a good survey. However, we give a simple and minimal introduction that will be enough to understand fully our implementation of the problem.

B-spline basis functions are piecewise polynomial with a finite support and a recursive definition. The basis splines of degree 1 are simply the characteristic functions of the intervals between real $\{u_j\}$ values called *knots*:

$$B_{j,1}(u) = \begin{cases} 1 & u_j \leq u < u_{j+1} \\ 0 & \text{otherwise} \end{cases}$$

Successively higher-order splines are formed by blending lower-order splines:

$$B_{j,K+1}(u) = \frac{u - u_j}{u_{j+K} - u_j} B_{j,K}(u) + \frac{u_{j+K+1} - u}{u_{j+K+1} - u_{j+1}} B_{j+1,K}(u). \tag{2}$$

It can be seen from this construction that $B_{j,K+1}$ functions are globally C^{K-1}. Evaluation(2) is especially efficient (i.e. can be computed with divided differences) and permits to implement splines of all orders. For each application we implement the most adapted order. We plot in Fig. 3a quadratic B-spline functions. In order to model a curve, we associate a control point $\mathbf{c_x}$ (3 coefficients for a $3-D$ curve, 1 coefficient in the nonparametric case) to each function. Note that with the shape of the functions in Fig. 3a, endpoints will be interpolated. We call them "*straight*" functions. It is also possible to obtain a closed curve, with the help of functions as in Fig. 3b. We call them "*circular*" functions.

We now recall briefly the best and smoothest approximation properties for interpolating splines of odd order ($K = 2m$, cubic $K = 4$, quintic $K = 6$, etc. This must be the reason for noting order K = degree +1.) [dB78].

1. smoothest approximation: the spline $\mathbf{Sx}$ interpolating data $\mathbf{x_i}$ minimizes the norm $\int ||\mathbf{Sx}^{(m)}||^2$
2. best approximation: the m-th derivative of the interpolant $\mathbf{Sx}^{(m)}$ approximates the m-th derivative of the underlying function $\mathbf{x}^{(m)}$ in the least squares sense.

Recall that interpolation is equivalent to the least squares approximation of selected sample points. Taking these considerations into account, we prefer to use *cubic* splines in the sequel.

In order to model a surface, we consider a tensor product of spline functions, i.e. a current surface point $\mathbf{x}(u,v)$ will be written as $\mathbf{x}(u,v) = \sum_{kl} B_k(u) B_l(v) \mathbf{c}_{\mathbf{x}kl}$. We are already able to write the discrete energy $E($ 1) in terms of B-splines. We write in matrix $x(i,j)$ of size

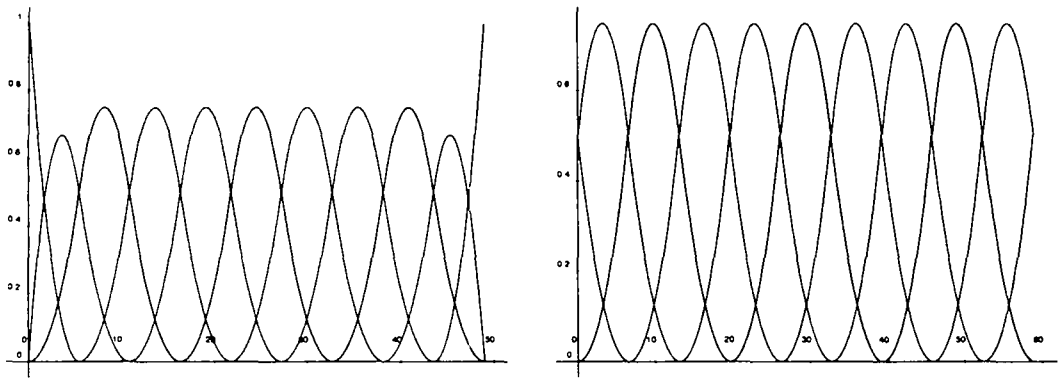


Figure 3: **a. left:** Uniform quadratic **straight** B-spline functions. If a control point is attached to each function, you obtain a $\mathcal{C}^1$ curve that interpolates endpoints.
b. Right: Uniform quadratic **circular** splines. Set the two last control points equal to the two first in order to obtain a closed $\mathcal{C}^1$ curve.
We plot here quadratic functions for simplicity, but we implement splines of all order. Particularly, we make ample use of cubic splines, because of their property to minimize the bending energy in $1 - D$.

$\dim_u \times \dim_v \times 3$ all surface sample points $\mathbf{x}(\tilde{u}_i, \tilde{v}_j)$. We also define a control point matrix $c_x(k,l)$, $\text{dimc}_u \times \text{dimc}_v \times 3$. Notice that because we approximate, there are necessarily more sample points than control points. We gather all spline evaluations $B_k(\tilde{u}_i)$ in a Matrix B_u, $\dim_u \times \text{dimc}_u$.

We next write matrix B_u for the common cubic splines: for each sample point only 4 functions are non zero. Analogously, we define B_v, $\dim_v \times \text{dimc}_v$.

$$B_u = \begin{pmatrix} B_0(\tilde{u}_0) & 0 & 0 \\ B_1(\tilde{u}_0) & B_1(\tilde{u}_1) & 0 \\ B_2(\tilde{u}_0) & B_2(\tilde{u}_1) & \backslash \\ B_3(\tilde{u}_0) & B_2(\tilde{u}_1) & \backslash \\ 0 & B_3(\tilde{u}_1) & \backslash \\ 0 & 0 & \backslash \\ \backslash & \backslash & \backslash \end{pmatrix}$$

Under some regularity conditions on the sampling, i.e. $\{\tilde{u}_i\}$ real values, and on the parametrization, $\{u_k\}$ values, we observe that the matrix B_u has rank dimc_u. Since questions regarding this rank value were repeatedly asked to the author, and also since proofs are difficult to find (De Boor [dB78] gives a reference in p.200), we discuss this point in Appendix B.

Next, we write the sample points matrix x as $B_v^t c_x B_u$. In doing so, we can only achieve three *simple topologies*, planar topology, cylindrical topology and toric topology, summarized in the next table.

$v$functions\$u$functions	straight	circular
straight	planar	cylindrical
circular	cylindrical	toric

If basis functions are *straight*, isoparametric curves will interpolate end control points. If basis functions are *circular*, isoparametric curves will be closed curves.

2.3 Euler Equation has a Closed Form Solution

With the above introduced notations, the discrete energy $2E(\ 1)$ can be written:

$$2E = ||B_v'^t c_x B_u'||^2 + ||d||^2$$

with the norm $||x(i,j)||^2 = \sum x_{ij}^2$. Assuming *cubic* splines, we prefer to write the energy in terms of the *third* derivative: $||B_v'''^t c_x B_u'''||^2$. This derivative is piecewise constant and thus discontinuous. By constraining this derivative norm to be minimum, we implicitly constrain lower order derivatives and we reduce the roughness component of the surface. Note that since this derivative is constant on each interval between knots, one evaluation suffices to compute exact quadratures, and also an exact expression for penalized least squares, where the customary regularization term involves an integral $\int$ rather than a discrete sum $\sum$.

We take the gradient of E, ∇E, with respect to spline coefficients c_x. Since we can only measure the gradient of the distance field d on the surface $\mathbf{x}$, we must apply the chain rule: $x = B_v^t c_x B_u$, $\nabla||d(x(c_x))||^2 = B_v \nabla||d(x)||^2 B_u^t$. We obtain:

$$\nabla E \ = \ B_v''' B_v'''^t c_x B_u''' B_u'''^t + B_v(d\nabla d)B_u^t = 0 \qquad (3)$$

Refer to appendix A for the standard numerical solution of equation 3. Next, we write the gradient of the potential as a function of the surface point $\mathbf{x}$ and obtain a closed form solution:

$$\nabla E \ = \ B_v''' B_v'''^t c_x B_u''' B_u'''^t + B_v(x-c)B_u^t,$$

with c, *extrapolated "contour" matrix* equal to matrix $x_0 - d_0\nabla d_0$ (index $_0$ stands for initial value or position). For simplicity, we note A matrix BB^t ($A_u = B_u B_u^t$, $A_v = B_v B_v^t$), and with a slight abuse of notation: $A_u''' = B_u''' B_u'''^t$.

$$\boxed{0 = A_v''' c_x A_u''' + A_v c_x A_u - B_v c B_u^t.} \qquad (4)$$

(4) is a linear system, i.e. the promised closed form. However, we would have here the global minimum if the potential was truly quadratic. Instead, $(x-c)$ is only a local fit of $d\nabla d$ and the exact solution will issue from the *iteration of the process*. Of course, as visualized in Fig. 2, few iterations will be needed in general. In addition, no step size needs to be specified. Note that (i) there is no theoretical proof that we can find the global minimum of the potential, as with classical "snakes". (ii) there is however an exact measure of the residual external energy E_e, provided directly by the potential image. This information is very useful for piloting the convergence process.

Note that the matrices A_u and A_v are symmetric positive because the associated quadratic form is positive ($c_x^t A c_x = c_x^t B B^t c_x = ||B^t c_x||^2$, $||B^t c_x||^2$ is a semi-norm) and because A and B have the same rank from the same equality $c_x^t B B^t c_x = ||B^t c_x||^2$ (c_x is in the nullspace of B^t if and only if it is in the nullspace of BB^t). Moreover, Appendix B tells us that this rank is maximum so A_u and A_v are symmetric positive definite.

2.4 Resolution can be Separated for u and v

The separability, which is the second key advantage of our method is explained in this section. We could here consider c_x as a *vector* and write our system (4) as (which we classify as "NOT separated") :

$$\text{Huge}A c_x = B_v c B_u^t. \tag{5}$$

In the next figure is the doubly banded matrix Huge$A$. Each element has size dimc$_u$ and the number of lines and columns is dimc$_v$. This matrix shape is frequent in 2-D problems.

$$\overbrace{\begin{pmatrix} \overbrace{A_uA_v(0,0)}^{\text{dimc}_u} & A_uA_v(0,1) & A_uA_v(0,2) & A_uA_v(0,3) & 0 & 0 & \backslash \\ A_uA_v(0,1) & A_uA_v(1,1) & A_uA_v(1,2) & A_uA_v(1,3) & A_uA_v(1,4) & 0 & \backslash \\ A_uA_v(0,2) & A_uA_v(1,2) & A_uA_v(2,2) & A_uA_v(2,3) & A_uA_v(2,4) & A_uA_v(2,5) & \backslash \\ A_uA_v(0,3) & A_uA_v(1,3) & A_uA_v(2,3) & A_uA_v(3,3) & A_uA_v(3,4) & A_uA_v(3,5) & A_uA_v(3,6) \\ 0 & A_uA_v(1,4) & A_uA_v(2,4) & A_uA_v(3,4) & A_uA_v(4,4) & A_uA_v(4,5) & A_uA_v(4,6) \\ 0 & 0 & A_uA_v(2,5) & A_uA_v(3,5) & A_uA_v(4,5) & A_uA_v(5,5) & A_uA_v(5,6) \\ \backslash & \backslash & \backslash & \backslash & \backslash & \backslash & \backslash \end{pmatrix}}^{\text{dimc}_v}$$

Much more elegant is to solve (which we classify as "separated"):

$$A_{\tau_v} c_y = B_v c B_u^t;$$

followed by:

$$A_{\tau_u} c_x = c_y^t; \tag{6}$$

(with $A_{\tau_u} = A_u + \tau_u A_u'''$, and $A_{\tau_v} = A_v + \tau_v A_v'''$) thus taking advantage of the symmetry of the problem. The price to pay is to consider a different internal energy term E_s:

$$E_s = \tau_u ||B_v^t c_x B_u'''||^2 + \tau_v ||B_v''''^t c_x B_u||^2 + \tau_u \tau_v ||B_v''''^t c_x B_u'''||^2, \tag{7}$$

instead of $\tau ||B_v''''^t c_x B_u'''||^2$. The addition of matrix $\tau A'''$ to matrix A intensifies the positive definite character of A and decreases its condition number. This is an important fact in of the theory of *regularization*. As a surrogate for a proof, we show in Fig. 4 an experimental plot of the condition number of matrix A_τ versus τ.

We adopt the compromise for the energy term(7), use equation(6) and analyze in the next paragraph the advantage obtained in terms of computational complexity. In terms of *storage*, the advantage is, of course, *decisive*. Take for instance a mesh of 100 × 100 = 10000 control vertices and compare the space needed to store matrices A_{τ_u} and A_{τ_v} versus matrix

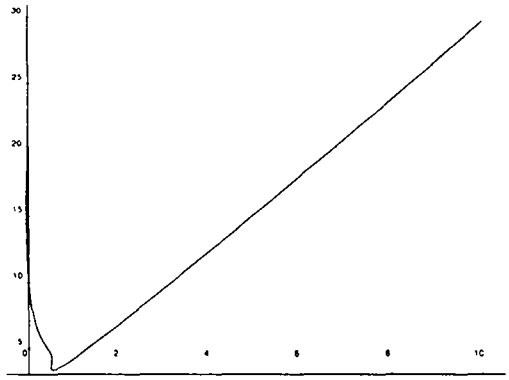


Figure 4: Experimental measurement of condition number for matrix A_τ versus τ. The obtainment of the optimum τ is a non-trivial problem. However, there is a large amount of τ values that ameliorate the condition number.

HugeA_τ, for a floating point computation, i.e. 4 bytes per matrix element. Assume even that these matrices are band matrices (i.e. exclude cylindrical and toric topologies!), which permits a dense storage. We obtain $2 \times K \times 100 \times 4$ as compared with $K \times 100^3 \times 4$, i.e. for cubic functions, 3.2 Kbytes versus 1.6 Mbytes. With a cylindrical topology, accounting for the symmetry of the matrix, 200Mbytes would be theoretically needed, and unfortunately, also in practice: The factorized matrix is rather dense and we know no obvious method to reduce the storage space. Even if discussions about memory space are not fashionable, the reader will agree that the above mentioned sizes are rather critical!

Because our matrices are positive definite (see above), we adopt Cholesky's factorization for linear systems and we compare in the next table the time complexities with and without variable separation (See for instance [WR71] p.50-56). We assume the *planar topology* which is the best case for the NOT separated method(5). Recall that in the separated case(6), right hand sides consists of matrices instead of vectors.

complexity / 6	operation	NOT separated (5)	separated (6)
decomposition	$\cdot$	$\text{dimc}_u^3\text{dimc}_v\text{deg}^2$	$(\text{dimc}_u + \text{dimc}_v)\text{deg}^2$
decomposition	$\surd$	$\text{dimc}_u\text{dimc}_v$	$\text{dimc}_u + \text{dimc}_v$
solution	$\cdot$	$\text{dimc}_u^2\text{dimc}_v\text{deg}$	$2(\text{dimc}_u\text{dimc}_v)\text{deg}$
leading term	$\cdot$	$O(\text{dimc}_u^3\text{dimc}_v)$	$O(\text{dimc}_u\text{dimc}_v)$

In sum, the advantage of the separated method(6) is immense, both in terms of space and time complexities, and the shortcomings due to energy term(7) are not obvious.

2.5 Spline Interpolation of Potential d

We describe in this section the use of a continuous potential field, which is the third principal characteristic of our method. Two main problems arise with potential field d, which is essentially discrete: (i) We need to compute a useful gradient vector ∇d. (ii) If two neighboring

surface point belong to the same voxel, d and ∇d should be different for those points. For instance, we initialize a cylinder in a distance image originated by an ellipsoid. We compare the elastic forces computed with a discrete gradient operator (see Fig. 5a) with the elastic forces computed we the $3-D$ interpolating splines we describe in the sequel (see Fig. 5b).

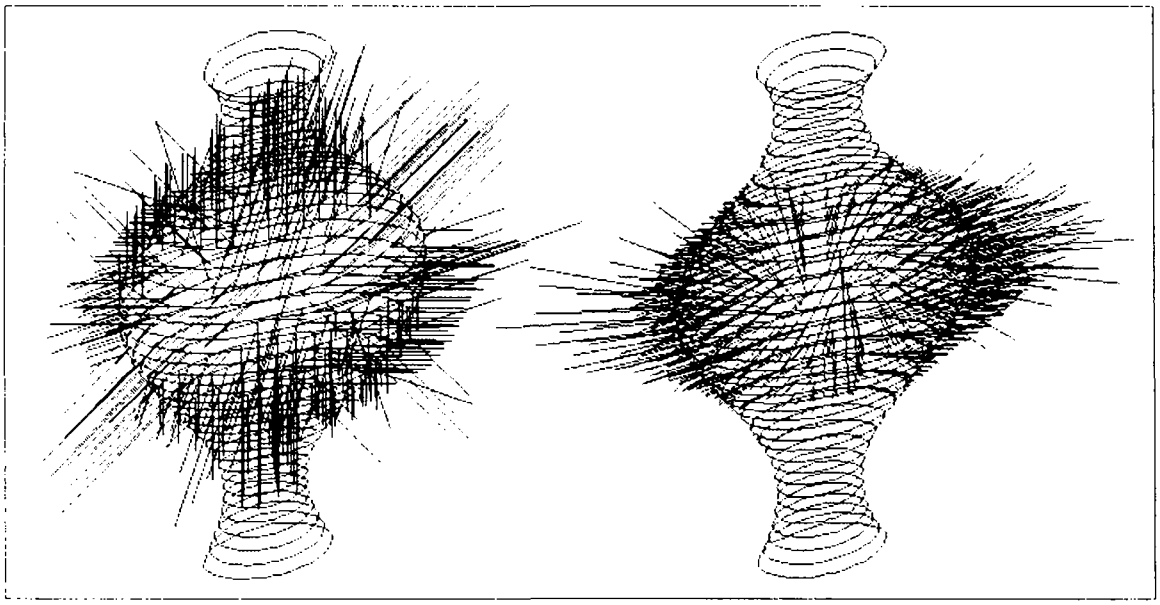


Figure 5: **a. left:** Synthetic data: A cylinder evolves in the potential originated by an ellipsoid. We represent the cylinder after a few iterations. We plot elastic forces obtained with a standard difference operator. The forces vary abruptly and many neighboring points are subject to the same forces.
b. Right: We use $3-D$ interpolating splines to compute a smooth and regular field of forces.

The solution we adopt is to model *continuously* potential d, in order to compute continuous gradient vectors and to extrapolate d in subvoxel precision. Techniques for modeling 3-D images are well known, and incorporate many different function types. Most popular are nearest neighbor, linear and cubic B-spline interpolation as well as different convolution methods involving polynomials, exponentials and cardinal sines (see Parker et al. [PKT88], Maeland [Mae88], Hummel [Hum83] or Cinquin [Cin87]). Cardinal sines are often considered too costly. Nearest neighbor and linear interpolation suffer from well known shortcomings, viz. they respectively shift or smooth the data (loss of sharpness). In addition, the gradient would be respectively zero and constant, thus we eliminate these methods.

Cubic spline interpolation was considered one of the best methods in [Mae88], after comparison of the spectra of the kernels involved for different functions. This particular spectrum has a higher response in the pass band and a lower response in the stop band than any other spectrum tested. We implement this method. In order to obtain the cubic spline coefficients, we must solve a system for which the separation of variables is compulsory. This operation is delicate and can be done off-line. However, it is interesting to keep these spline coefficients for other purposes, e.g. the resampling of images or the computation of high degree derivatives.

The on-line sub-voxel gradient computation ∇d represents one of the bottlenecks of the method with a high computational load: With cubic splines, 3×64 multiplications are needed for one gradient evaluation. Thus we totalize $3 \times 64 \times \dim_u \times \dim_v$ multiplications for the whole sampled surface $\mathbf{x}$. This is of the same order of magnitude as the number of multiplications in Cholesky's factorization,"separated" case (see section 2.4). Yet these volumetric splines contribute to the quality of the segmentation, especially when we take a limited number of sample points (see example of Gregoire, section 2.6.1), or in alternatively, when many surface sample points belong to the same voxel (see example of Carl, section 2.6.2).

2.6 Results of Segmentation

We tested our method on various data. We focus in this section on 2 examples, both with MRI data. We thus analyze in detail the convergence process. Each time contours points have been labelled and a $3-D$ euclidian distance image d has been precomputed. In addition, $3-D$ spline coefficients have been evaluated, which permit to compute a continuous cubic C^2 version of the potential field d^2. Note that contours are provided by the distance image and that we concentrate only on an efficient way to pilot a spline surface towards the contours. We do not claim to detect simultaneously the position of contours.

Each time we initialize the process with a simple surface, respectively a plane and a circular cylinder (see Fig. 9a), inside the $3-D$ distance image d. After 15 iterations, i.e. resolutions of system(6), we consider that the solution is stationary.

2.6.1 Example of Gregoire (Figs.6a to8b)

This example illustrates the **planar** topology. The original data is a $3-D$ $256 \times 256 \times 124$ axial MRI image of the head of Gregoire Malandain, researcher.

We take 96×64 samples on the plane and 64×48 spline control vertices for the doubly cubic C^2 spline surface. For the evolution of the spline surface, we only take into account the component of the gradient perpendicular to the original plane.

We observe the evolution of the sample points over the successive iterations. Since no resampling is done, we could trace the "trajectory" of each sample. The surface stops evolving after about 15 iterations and 111 seconds CPU on a Dec 5900 (i.e. one iteration takes 7.3 seconds). For an analysis of the computational complexity, see sections 2.4 and 2.5.

The successive shapes we observe are rather **not intuitive** (we think especially about Fig. 6a and Fig. 6b). They illustrate that the potential d^2 is not quadratic (only our approximation of d^2 is), but instead extremely elaborate. They give a feeling of what results the method would yield with **contours of poor quality**, with dilated contours, or with simple thresholding in MRI images.

After iteration six (see Fig. 7a and Fig. 7b), we have a good overall idea of the shape concealed in the $3-D$ image. The subsequent 9 iterations permit to segment the face with full precision. Notice in Fig. 8b the lack of data at the tip of the nose in the original image that the spline surface would not surmount.

When observing these pictures, we realize the need for an **iterative refinement** of the mesh. This refinement would increase the speed of the method since the data would be smaller at the beginning, and would maybe decrease the number of iterations, because the

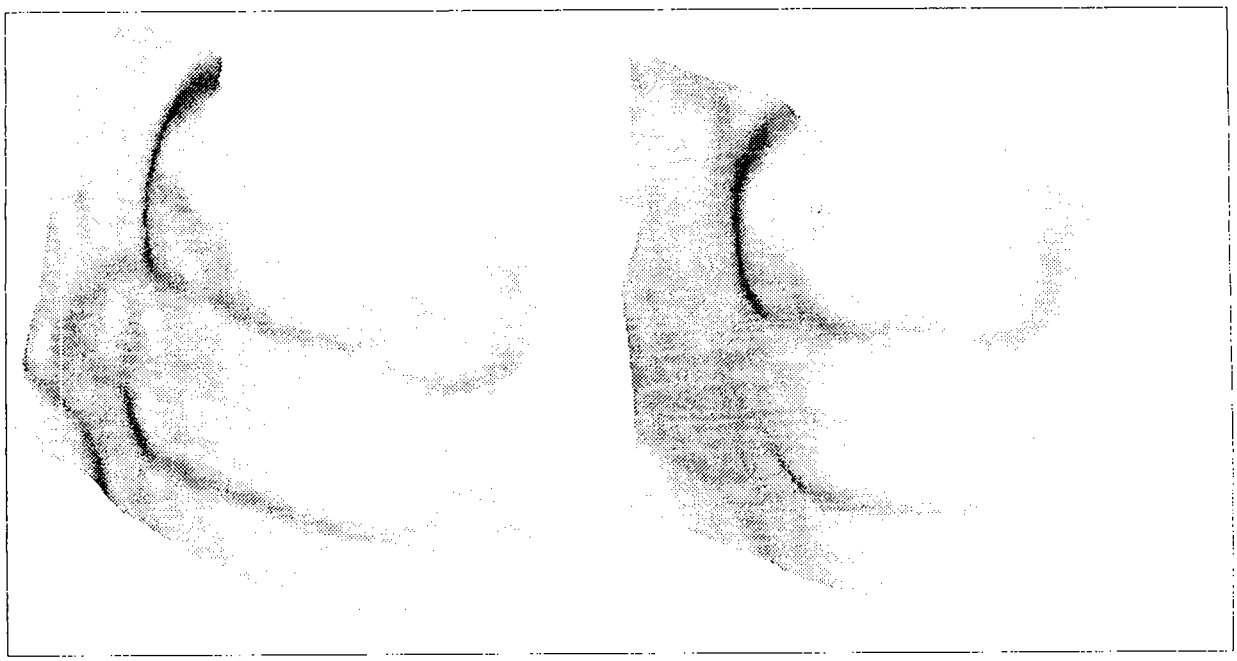


Figure 6: **a. Left:** Example of Gregoire, first iteration. This picture, together with the following one, gave Gregoire Malandain a stroke.
b. Right: Second iteration. Together with the previous picture, it demonstrates that the euclidian potential originated by the contours of Gregoire's head is not quadratic !

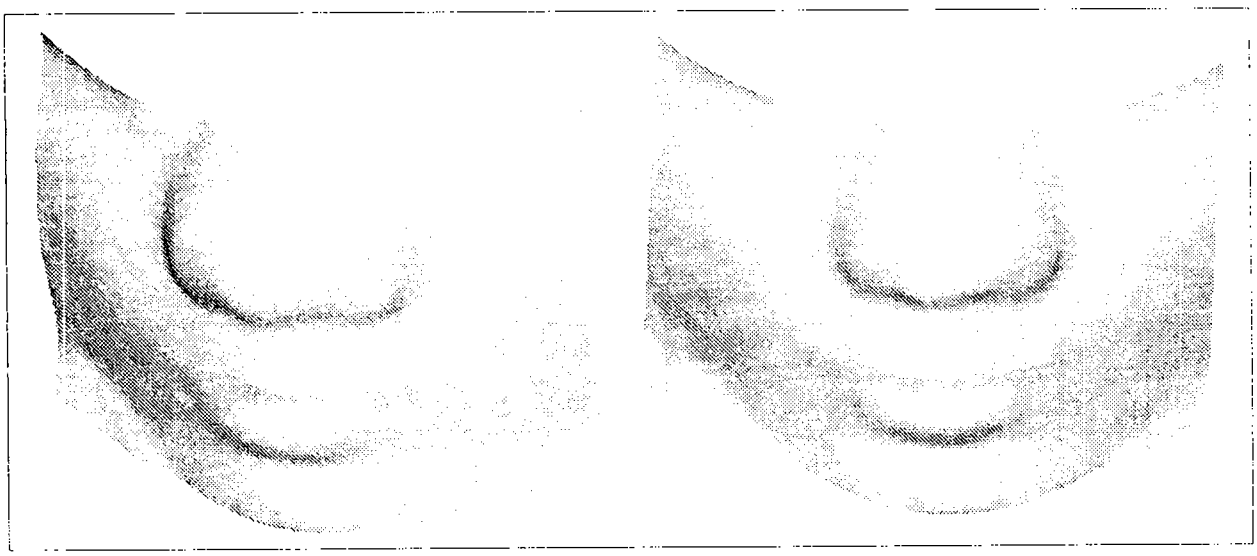


Figure 7: **a. Left:** Fourth iteration. The surface is stabilized in the region of the mouth.
b. Right: Sixth iteration. The nose appears with more details.

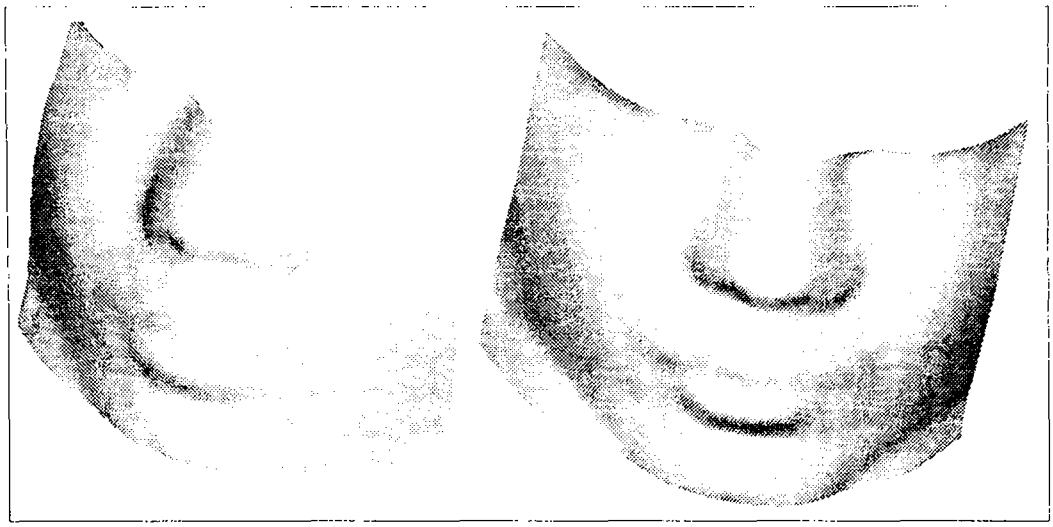


Figure 8: **a. Left:** Eighth iteration. The segmentation is almost complete.
b. Right: Fifteenth iteration. The spline surface is stationary. Note that the spline could not fill up the lack of data at the tip of the nose.

too high precision in the first iterations pilots the surface toward marginal minima of the potential that should be disregarded. Our method if fully compatible with such an iterative refinement. We implement the refinement procedure for the next example.

2.6.2 Example of Carl (Figs.9a to11b)

This example illustrates the **cylindrical** topology. The original data is a $3-D$ $256\times256\times127$ complete sagittal MRI image of the head of a patient.

1. Brute Force Strategy.

   We take 192 × 192 samples on an initial circular cylinder and 96 × 96 spline control vertices for the doubly cubic C^2 spline surface. For the evolution of the spline surface, we only take into account the component of the gradient perpendicular to the axis of the original cylinder. The surface stops evolving after 15 iterations and 915 seconds CPU on a Dec 5900 (i.e. one iteration takes 61 seconds).

   A new phenomenon can be observed here, due to the presence of **isolated contour points** that perturb the convergence in Figs. 9b, 10a, where we observe sharp swellings on the surface. The internal energy of the model will triumph over parasite contours after fifteen iterations (see Fig. 11a).

   Recall that the deformable spline surface is globally C^2 from the first iteration on. Thus, we can observe the evolution of the curvature as the surface deforms. We can also apply specific transformations to discover the rules that govern the evolution of the curvature and observe the **stability** of curvature measures. If necessary, we can observe higher order derivatives with higher order splines (remember scheme(2) for the construction of B-spline functions).

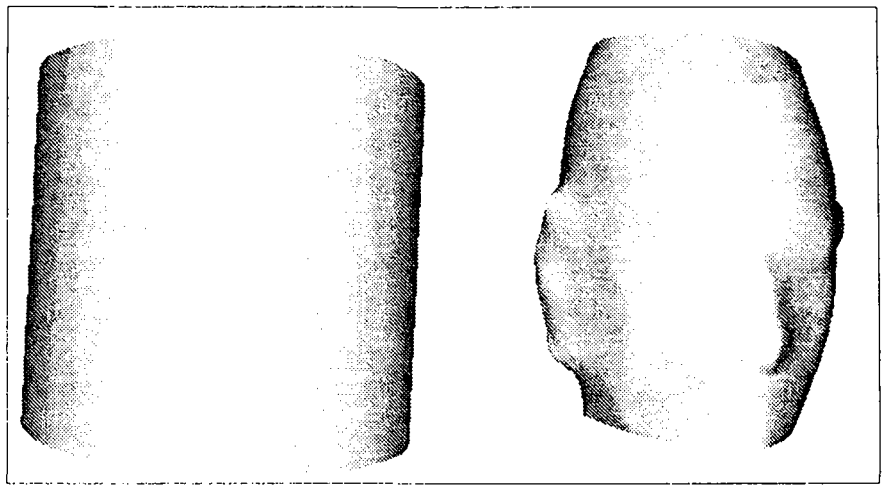


Figure 9: **a. Left:** Example of Carl: The initial surface is a circular cylinder.
b. Right: Third iteration: We localize already the mouth, the nose an one ear. Near the eye and the mouth, isolated contours refrain the surface from evolving.

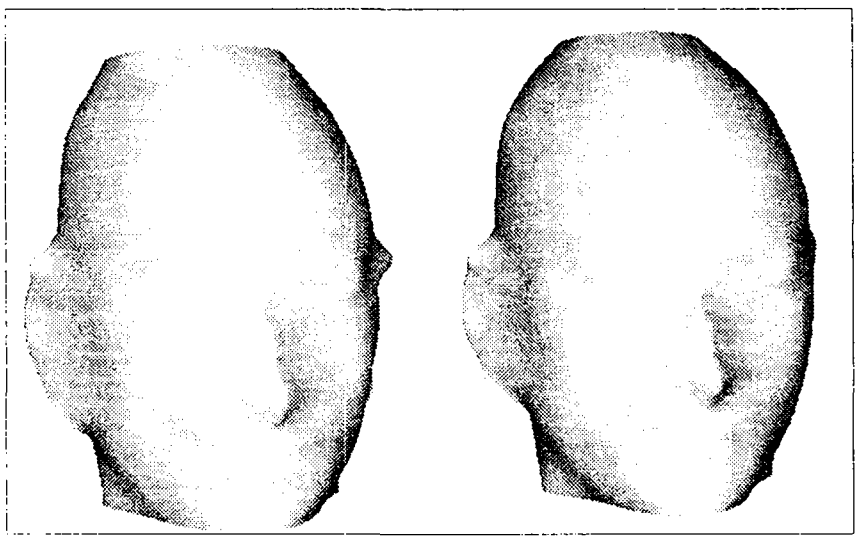


Figure 10: **a. Left:** Sixth iteration: Swellings still appear near the mouth, the right ear and the left eye.
b. Right: Tenth iteration: The protuberances have almost disappeared, but ears do not have a normal shape.

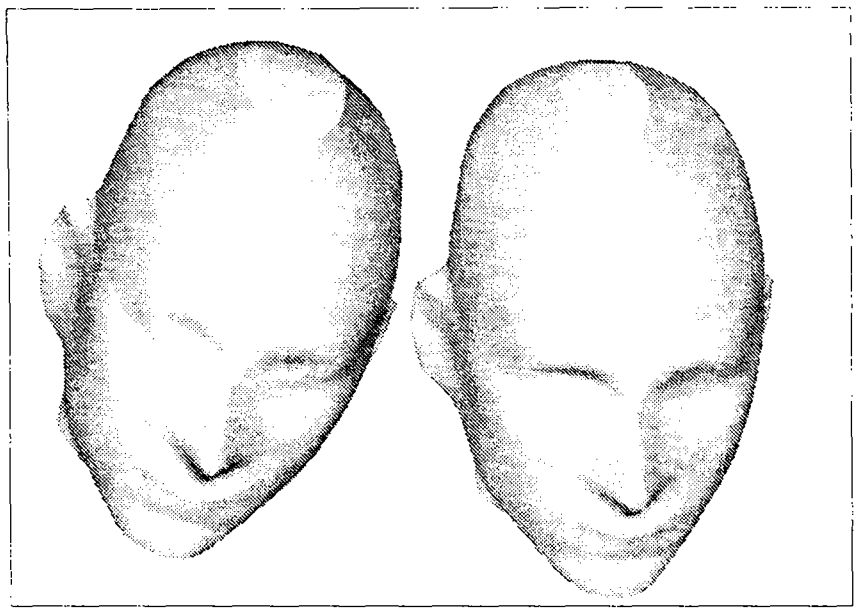


Figure 11: **a. Left:** Fifteenth iteration. Five more iterations were necessary to complete the segmentation and overcome the isolated contour points.
b. Right: Twenty ninth iteration: The surface evolved a little around the eyes.

2. Iterative Refinement.

   We depicted below a brute-force method to obtain the convergence of the spline surface. Now, for comparison, we show the result of an iterative refinement method. We linearly increase during 10 iterations the number and samples from 30×15 to 160×100 and in parallel we increase the number of control vertices up to 140×80. The whole process operates in 160 sec. CPU (in average, 16 sec. per iteration). The resulting surface is shown in Fig 12a. We pursue the refinement up to 280×160 samples and 256×128 control vertices. This takes 510 sec. CPU. We show the result in Fig. 12b. For completeness, we show the profile of Carl in Fig. 13a. The strategy we chose to refine the grid was to linearly increase its size along each coordinate, that is to increase quadratically the number of sample points. Other strategies might perform better and will be studied in the future.

We provide additionally a limited number of color prints to visualize the total curvature $k_1^2 + k_2^2$ of the face of Carl. In the first three prints, we visualize six steps of the convergence with our own software that traces isoparametric curves. The algorithm of the moving horizon is useful to discover some of the hidden parts. Im these pictures, the color determines the magnitude of the curvature as follows: (From lowest to highest curvature) white, blue, green, yellow, red. In Figs. 14, 15 and 16, we visualize iterations 0, 1, 3, 6, 9 and 21. All these pictures are at the same scale, and the same color represents the same curvature value.

In order to draw crest lines, we define a mask of interest of surface regions having a total curvature higher than some threshold. We then walk discretely along surface points and mark crest points inside the interest mask (see Fig. 17). Fig. 18 visualizes the fineness of the mesh, that could be achieved by using splines. Lastly, we show in Figs. 19 and 20 renderings

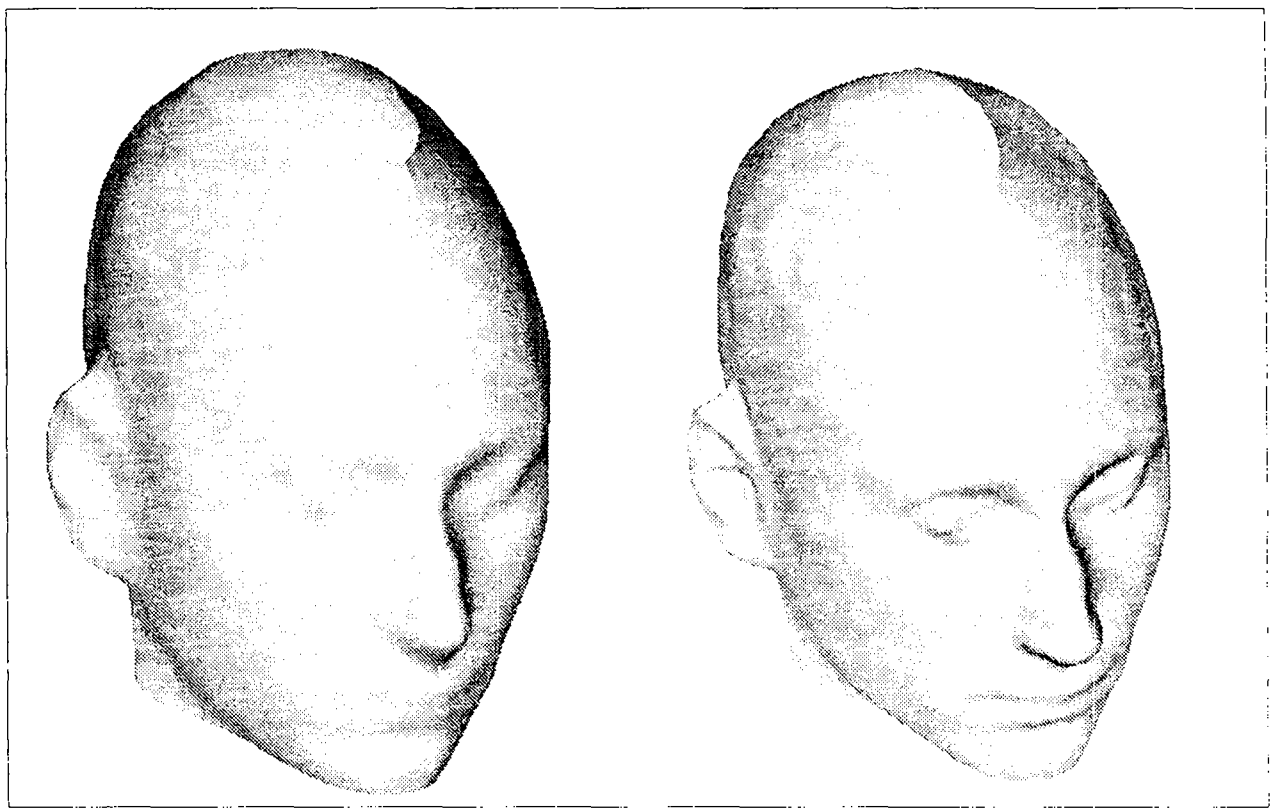


Figure 12: **a. Left:** Surface obtained with only ten iterations and a linear increase of samples points up to 160 × 100 The whole process operates in 160 sec. CPU (in average, 16 sec. per iteration).
b. Right: Result with four more iterations on a 280 × 160 grid. this refinement takes 510 sec. CPU. There are extraneous contour points near the eye that were not overcome, due to the fineness of the spline surface.

Figure 13: **a. Left:** Right profile of Carl with the 280 × 160 grid. Note that the total amount of computing time needed is less than for the brute force method.
b. Right: Crest lines are drawn on Carl's face after iteration fifteen, with our original method that measures local extrema of $k_1^2 + k_2^2$ on the surface (section3).

of Carl with the AVS software, respectively front part and right profile.

3 Crest Lines and Matching

In this section, we exploit the differentiability of the model by drawing lines of extremal curvature on the surface with a new algorithm. We will not recall here the expressions for the surface curvatures. The reader may refer to Do Carmo [dC76] (Note that we adopt the notations of Do Carmo for our surface $\mathbf{x}(\mathbf{u}, \mathbf{v})$). The quantity $k_1^2+k_2^2$, or "global curvature" is of special importance to us. It is a local measure of the bending energy density for a thin plate (see Courant and Hilbert [CH57]). For instance, in Fig. 1, the color value indicates the density of potential bending energy gained by the surface during the convergence. In addition, it can be seen from the equality $k_1^2 + k_2^2 = 4H^2 - 2K$ that this measure is differentiable, as opposed to principle curvatures k_1 and k_2 that are not differentiable at umbilics. Note that with our differentiable spline representation of the surface, we can use the criterion defined for crest lines in [MBF92], i.e. lines as zeros of the quantity $\nabla k_1 \cdot \mathbf{e_1}$ ($\mathbf{e_1}$ is the principle direction associated with curvature k_1). The experimentation of this alternative is in progress.

In general, local extrema of curvature will appear as lines. This has been studied for instance in [TG92]. An intuitive way to discover that we will obtain lines is to represent level curves of $k_1^2 + k_2^2$ and to remark that extremality is also a measure of medialness for the families of level curves.

We next extract locii of extremal global curvature $k_1^2 + k_2^2$. We sample $k_1^2 + k_2^2$ values

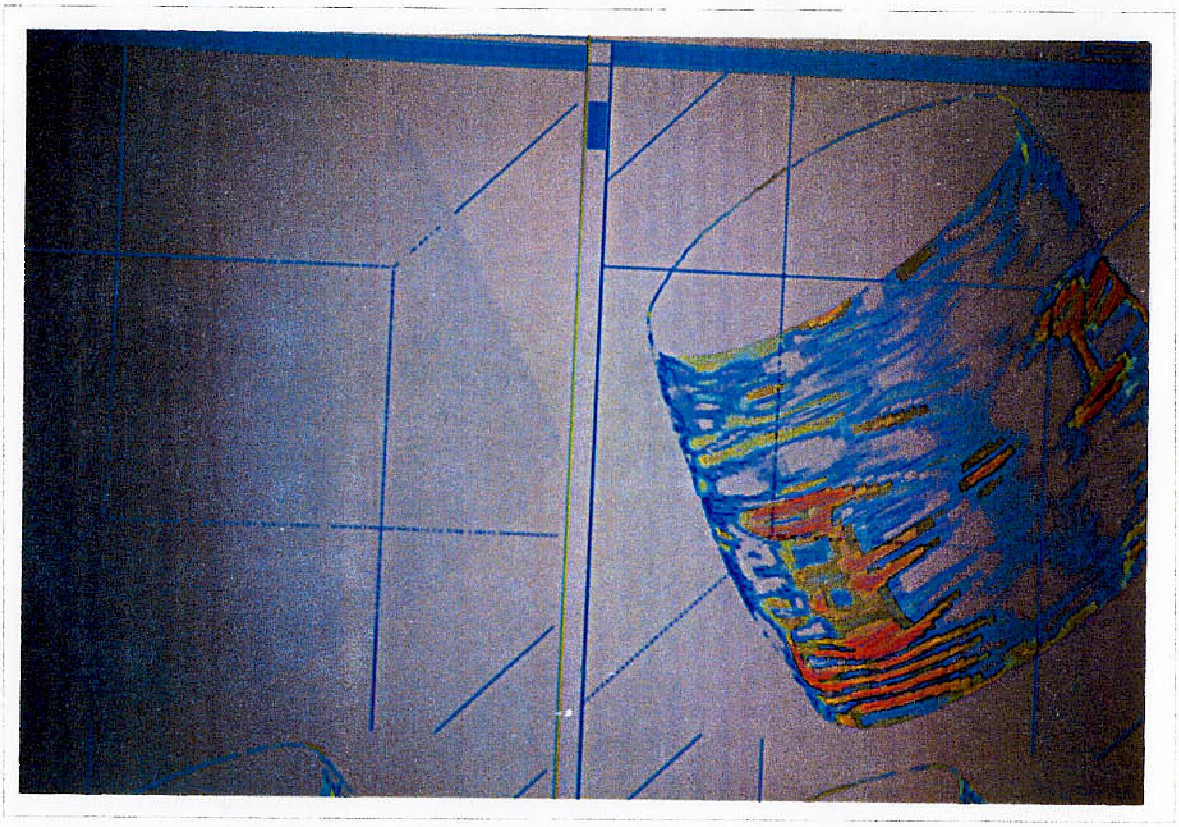


Figure 14: Iterations zero and one for Carl. Representation of the total curvature $k_1^2 + k_2^2$ of the surface. We use, from lowest to highest curvature: white, blue, green, yellow, and red color. The spline has 96 by 96 control vertices, and we take 192 by 192 samples on the surface in order to perform one iteration.

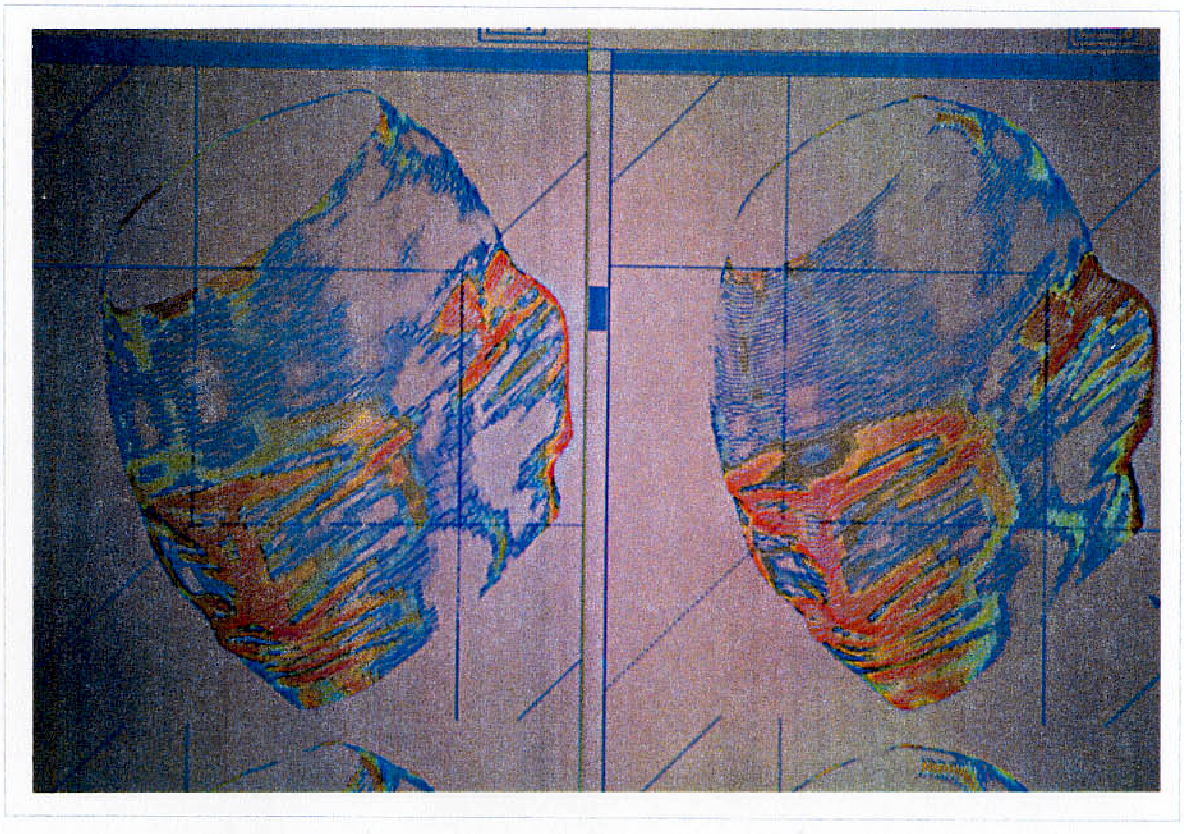


Figure 15: Iterations three and six. The curvature is high on the nose and the ear and also on sharp protuberances originated by parasite isolated contour points.

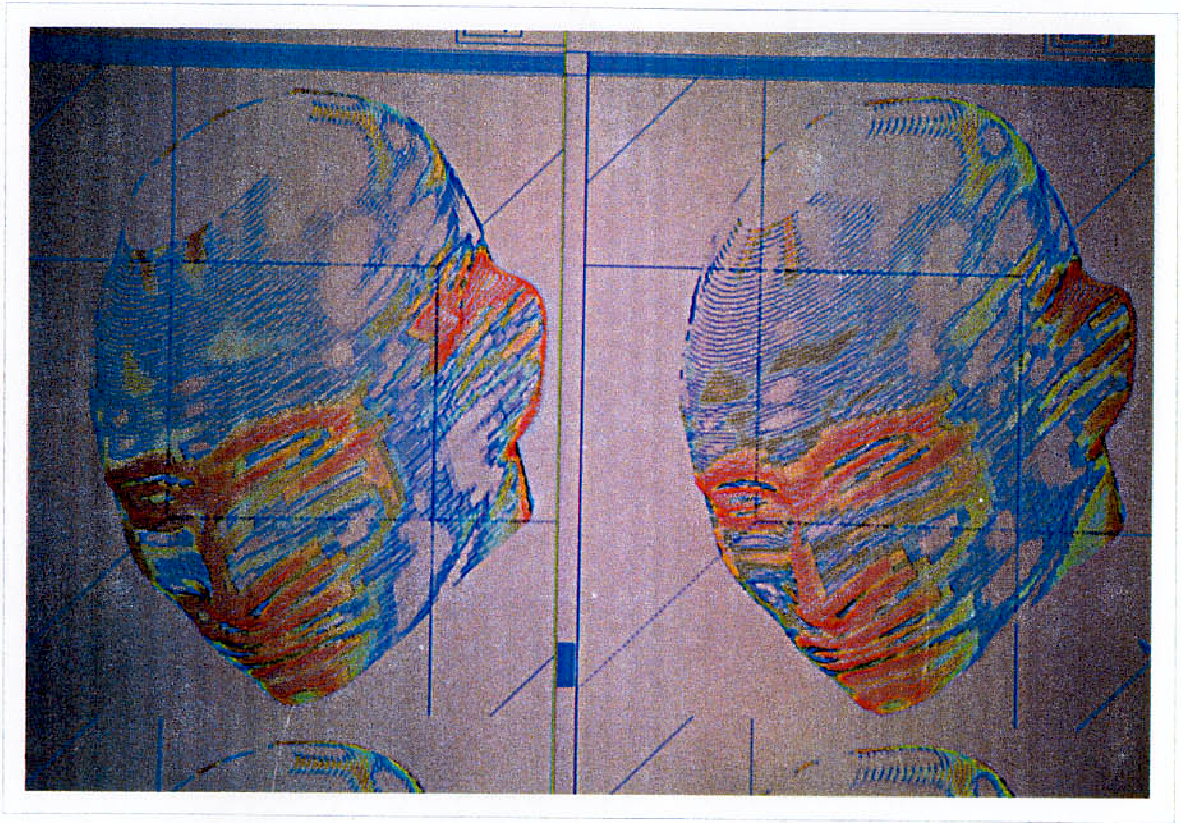


Figure 16: Iterations nine and twenty one. Final curvature map.

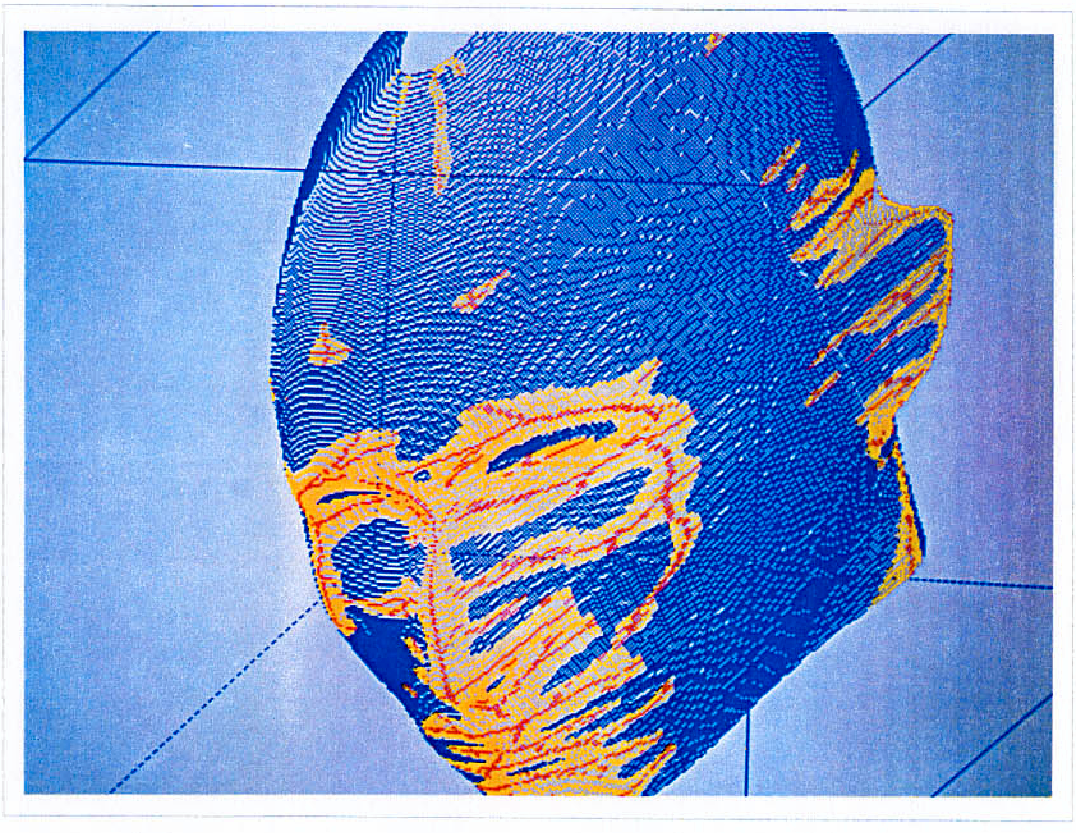


Figure 17: Inside the interest mask marked in yellow (high curvature points), we walk discretely along surface points and mark crest points, or extrema of curvature in red.

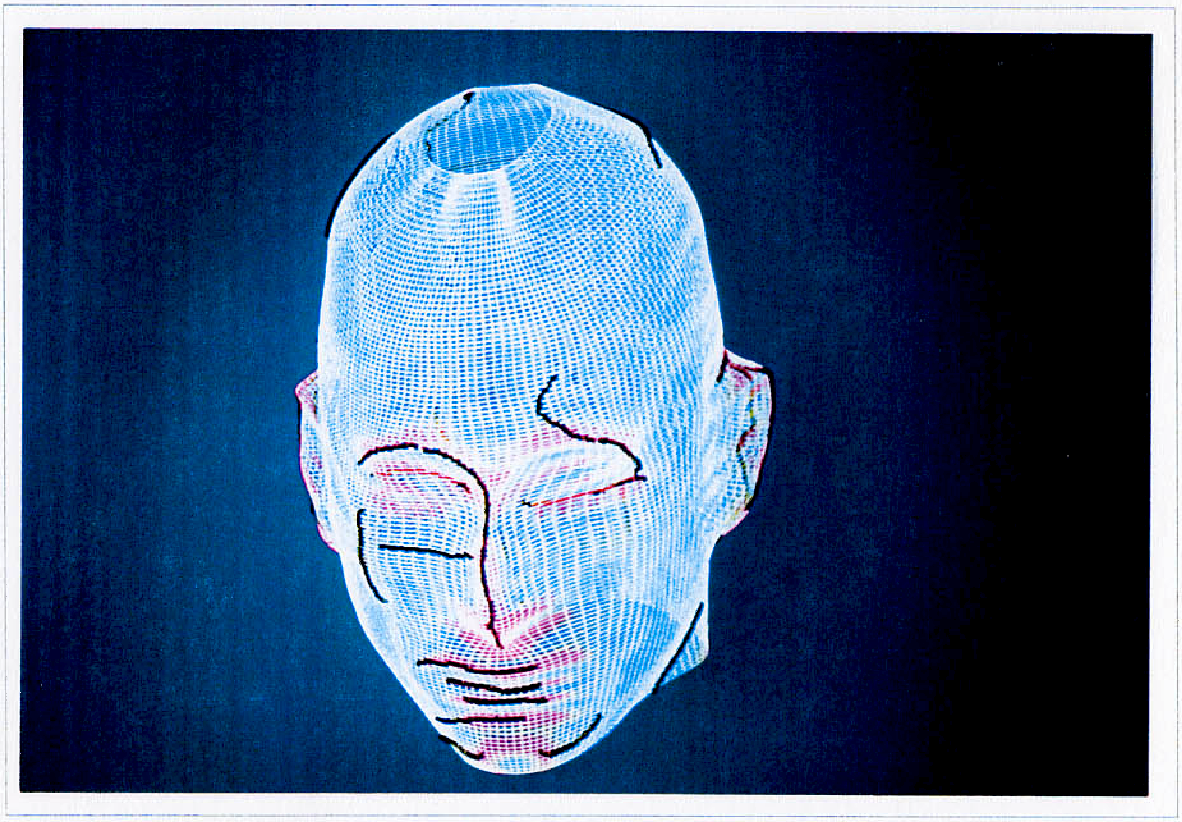
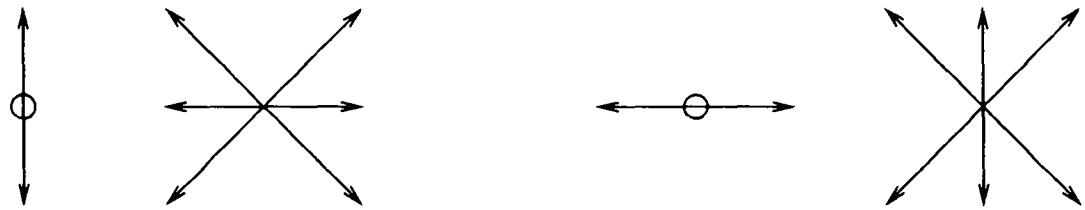


Figure 18: Superimposition of crest lines in black on the spline mesh of 192 by 192 points. The color of the mesh is representative of the total curvature measure $k_1^2 + k_2^2$: (From lowest to highest curvature) blue, green, yellow and red color. This figure visualizes the fineness of the mesh.

Figure 19: Rendered version of Carl with the AVS software. Crest lines are superimposed in black. (From lowest to highest curvature) blue, green, yellow, red color.

Figure 20: Rendering of the right profile of Carl with the AVS software.

on a regular rectangular grid. We test if the current point corresponds to a local discrete extremum of curvature along the axis South-North (see Fig. 21a) or along the axis East-West (see Fig. 21b) and explore recursively 6 directions for finding a next crest point. Thus we assume the 4-connectivity for the surface and the 8-connectivity for the crest curves. We avoid any paradox of discrete topology and a closed curve will split the surface in exactly to connected components, thus respecting Jordan's theorem (see for instance [Mal92]). The result of the procedure is shown on the Example of Carl, after iteration fifteen (Fig. 13b).



a. b.

Figure 21: **a. Left:** If we have a local maximum of curvature $k_1^2 + k_2^2$ along the axis South-North, we recursively explore eight directions except South and North. Thus we assume the 4-connectivity for the surface and the 8-connectivity for the crest curves. In doing so, we avoid a paradox of discrete topology (see for instance [Mal92]).
b. Right: If we have a local maximum of curvature along along the axis East-West, we explore eight directions except East and West.

We next test experimentally the stability of the crest lines as defined and extracted below. Of course, the measure $k_1^2 + k_2^2$ is invariant to rigid motion. We consider the example of Arthur, a "phantom" available in two different positions in $3 - D$ CT-scan images. For the purpose of the experiment, we resampled dramatically both views, A and B, from $256 \times 256 \times 135$ respectively down to $128 \times 128 \times 128$ and $96 \times 96 \times 96$. Thus, contours from B are considerably corrupted. In addition, we use splines with fewer coefficients for view B. The result of algorithm of section 2 is shown in Fig. 22.

In Fig. 23 we show the crest lines obtained with our software on views A and B of Fig. 22 and the rigid superposition obtained with algorithm [GA92]. Results are encouraging and attest the stability of the crest lines as defined in this section.

4 Current Research

4.1 Toric and Spherical Topology, Exact Volume

We briefly describe the characteristics of the spherical topology for spline surfaces. The toric topology is simple and has already been introduced in section 2.2.

As in the toric case, *circular* B-splines functions are used for parametrization variables u and v. We must distinguish between odd order and even order splines. With odd order splines, two control vertices, the north and the south poles, play a special role. Numerically, the problem of solving equation(4) is delicate because the system is sparse and badly conditioned (We use the Lanczos algorithm).

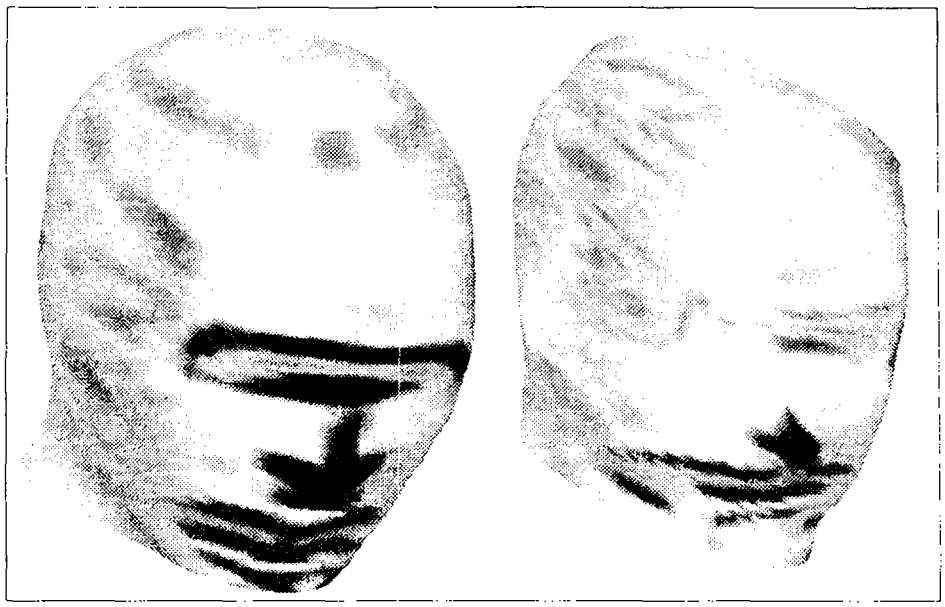


Figure 22: **a. Left:** Result of the algorithm of section2 on the example of phantom Arthur, view A. In addition, we represent the global curvature as follows: Black regions correspond to high curvatures, white regions to average curvature and light grey regions to low curvatures.
b. Right: Result on the example of phantom Arthur, view B. The quality of the segmentation is much better in view **a.** than in view **b.**

There are no poles with even order splines. Figure 24 visualizes the mesh for a biquadratic spline in spherical topology. On each meridian, the last control vertices are the same as the first control vertices on the opposite meridian.

We cannot separate the problem along discretization variables u and v as in equation(6) simply because we cannot write the sample points matrix x as $B_v^t c_x B_u$. Thus we must solve system(5). This can be done with Cholesky's method and the computational complexity is given in section 2.4.

Splines with spherical topology have been fully implemented and tested on synthetic data (note that Fig. 24 exhibits a real example where a sphere converged in the potential field generated by an ellipsoid).

Now, suppose we have a closed spline surface S bounding a volume V. The Ostrogradsky theorem states that:

$$3V = \int_V \mathrm{div}(\mathbf{x})dV = \int_S \mathbf{x}.\mathbf{N}dS = \int_{uv} \mathbf{x}\mathbf{N}\sqrt{EG - F^2}dudv, \tag{8}$$

div is the divergence operator, $\mathbf{N}$ is the surface normal pointing outwards, and E, F and G are coefficients of the first fundamental form of the surface. $\mathbf{N}$, E, F and G are well know and easy to compute from the spline representation of the surface. We can compute the last integral involved in equation(8) either with a Riemann sum or with a Romberg quadrature. This volume computation can be extended to surface triangulations, when $\mathbf{N}$, E, F and G can be approximated on each triangle. It will be very useful for computing the volume of

Figure 23: **a. Top:** Crest lines on the two views of phantom Arthur, obtained with the algorithm described in this section and smoothed with splines.
b. Bottom: Superposition of the lines of **a.** with the software of[GA92]. The algorithm still performs, despite the bad quality of the surfaces.

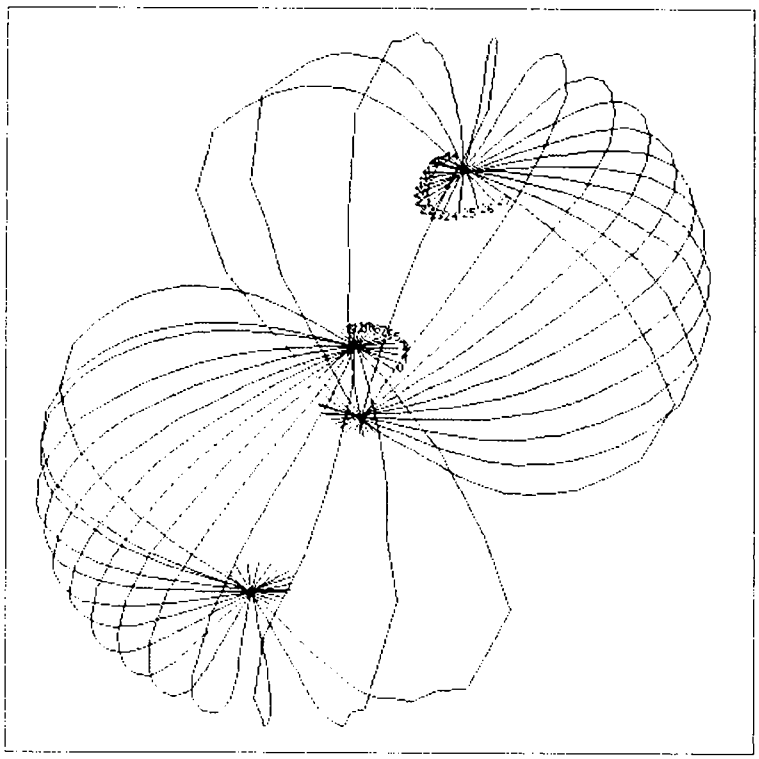


Figure 24: Mesh of control vertices for a bi-quadratic spline in spherical topology. For simplicity, we split the sphere in two. The number of meridians is even and each terminal control vertex on a meridian is also the first control vertex on the opposite meridian.

organs before the transplantation. This formula has not been implemented yet. We will present results as soon as adequate data is processed.

4.2 Avoidance of Self-Intersections, Tubular Neighborhood, On-Line Contour Extraction

In this section, we use the following property of differentiable surfaces: there is a number ϵ such that the segments along the surface normals of length ϵ do not intersect. ϵ defines a *tubular neighborhood* of the surface (consult [dC76]) Clearly, if we deform the surface along the surface normal, with displacements smaller than ϵ, we will avoid "local" self-intersections. "global" self-intersections are tricky, when dealing with a parametric surface, and not likely to occur in the examples we usually encounter (face, heart and other smooth organs). We exclude them. With the example of Carl, we indeed observed "local" self-intersections in the region of the ear, that would not occur if the motion of the spline surface would be properly piloted in these regions.

The ϵ of the tubular neighborhood is related to the surface curvature: Consider the normal lines of a $2-D$ curve at neighboring points. As the points become closer, the normal lines will by definition intersect at the center of curvature (see [dC76] for instance). In other words, if we take an ϵ smaller than the ray of curvature, the normal segments will not intersect. Consider a surface and its two normal sections along principle directions. These sections are $2-D$ curves whose curvatures are principle curvatures. By applying the arguments for $2-D$ curves to the two normal sections, we discover that ϵ equals the smaller ray of curvature of the surface.

Thanks to the differentiability of the spline, we can measure this smaller ray of curvature and impose a motion along the surface normal. Again, this will NOT guarantee the absence of "global" self-intersections. We plan to walk along the surface normal directly in the $3-D$ original image and look for extrema of the intensity gradient in the direction of the gradient along the normal lines, only for a restricted number of grey values. This will avoid costly precomputations such as the global extraction of contours and the computation of a distance potential. The implementation of the ideas described above in in progress.

5 Conclusion

We presented an original and efficient algorithm to pilot a spline surface towards contours inside a $3-D$ image representing a square distance potential d^2. Our approach, including approximating spline formulation (section 2.3), separability (section 2.4) and iterative refinement (section 2.6.2), constitutes a step further towards real time convergence. Since the differentiability is costly, real time could be achieved with a combination of a model with a low degree of differentiability for convergence (such as, for instance [NA92]), and a C^2 fit of the model when it has converged. Remember that contour points are already provided, and that the potential field d^2 is precomputed. This operation may take a prohibitive amount of time. However, since both problems of characterizing contours (1) and deforming a surface (2) are essentially solved, we are exploring techniques that would permit to evolve in the original $3-D$ medical image without any precomputation.

In addition, the differentiable deforming surfaces provide and excellent tool for measuring the evolution and the stability of curvatures and crest lines.

6 Acknowledgements

We gratefully acknowledge Dr. Ericke from Siemens for providing the $3-D$ MRI image of Carl, as well as GE-CGR and Gregoire Malandain for providing the $3-D$ MRI image of Gregoire and GE-CGR for providing the $3-D$ CT-scan images of Arthur. We also thank Emmanuelle Clergue who participated actively in the experimental study.

A Classical Euler Approach

To solve numerically equation 3, one introduces a time variable t and one derives the *evolution equation*:

$$0 = \frac{\partial c_x}{\partial t} + \nabla E,$$

which becomes, in the discretized version:

$$0 = \frac{c_{xi} - c_{xi-1}}{\delta t} + \nabla E$$

Now, depending on the way we express ∇E, respectively in terms of c_{xi} or c_{xi-1}, we use respectively an *implicit*(9) scheme (for consistency, we use the spline formulation of sections 2.2 and 2.3, and expression(3) for ∇E):

$$0 = \frac{c_{xi} - c_{xi-1}}{\delta t} + A_v''' c_{xi} A_u''' + B_v (d \nabla d) B_u^t; \tag{9}$$

or an *explicit*(10) scheme:

$$0 = \frac{c_{xi} - c_{xi-1}}{\delta t} + A_v''' c_{xi-1} A_u''' + B_v (d \nabla d) B_u^t \tag{10}$$

The implicit scheme has been shown to give better results. The interested reader may consult Ciarlet [Cia88].

B Rank of the Spline Matrix

We show in this section that for uniform sampling $\{\tilde{u}_i\}$, uniform parametrization $\{u_k\}$, and *straight* quadratic spline functions (see Fig. 3a), the rank of the interpolation matrix B, dimc $\times$ dimc, is equal to dimc. otherwise, there would be a non zero coefficient vector c satisfying:

$$Bc = 0. \tag{11}$$

In other words, there would be a non zero quadratic spline having dimc zeroes evenly spaced. This is a contradiction and the proof follows.

dimc is also the number of different spline functions, but there are *fewer* intervals between knots. As an example, in Fig. 3a, you will enumerate 9 intervals and 11 functions for quadratic splines. The general rule is for degree $K-1$ splines, dimc $-K+1$ different intervals. Inside those intervals, the spline is $\mathcal{C}^{\infty}$. At knot values the spline is only $\mathcal{C}^{K-2}$. So we can apply Rolle's theorem anywhere at least $K-2$ times.

From equation(11), we know that the spline has globally at least dimc zeroes (remark: it is the case because we interpolate. If we approximate, we have more zeroes). Thus from Rolle's theorem, the derivative has at least dimc -1 zeros: in each open interval i between 2 sample points $\tilde{u}_i$ and $\tilde{u}_{i+1}$ there is one zero of the derivative.

Fortunately, for a quadratic spline, these zeroes are already too many zeroes. This is good because we could only apply Rolle once ! In fact, the derivative of the quadratic spline is piecewise linear $\mathcal{C}^0$ in dimc -2 intervals. One difficulty arises here, i.e. the zeros of the derivative are *not equally spaced* in general (see Fig. 25. This picture describes the situation with great fidelity).

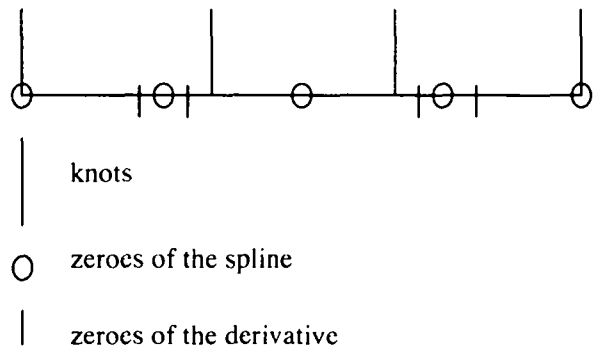


Figure 25: With uniform sampling for zeroes $\{\tilde{u}_i\}, i = 0..4$(circles), and uniform parametrization $\{u_k\}, k = 0..2$(large vertical bars), the spacing of the points where the derivative vanishes (small vertical bars) will not necessarily be uniform. For instance, in the second interval, there is no zero of the derivative. However, this could not be true for 2 consecutive intervals.

We know that the interval between zeroes is smaller than the interval between knots. There are necessarily 2 zeros in end intervals, thus there is exactly one zero in each internal interval. With this distribution, there is at least one interval k with two zeros of the derivative, certainly no interval with three zeroes of the derivative and no two consecutive intervals without zeroes.

Inside interval k, the linear derivative has two zeros and thus vanishes. The quadratic spline is constant, and by condition(11), zero. We apply the *continuity* conditions and discover zeroes u_k and u_{k+1} (knot values) for the spline and for its derivative. Then, we "propagate" the same argument on the left and on the right of interval k, since we "gained" by continuity two new zeros, u_k and u_{k+1}. If we encounter one interval where the derivative has no zero, there is necessarily one other interval l somewhere else with two zeroes. We would be blocked if we could exhibit two consecutive intervals without zeroes of the derivative, but

this is impossible. Finally, the quadratic spline vanishes completely.

We could try the same argument on higher order splines, but since the spacing between zeros of successive derivatives is likely to be more irregular, the proof will be tedious. We rather want to show what type of arguments can be used to prove that the least square matrix BB^t is positive definite, provided the sampling is regular enough compared with the knot spacing.

References

[BBB87] R. Bartels, J. Beatty, and B. Barsky. *An introduction to splines for use in computer graphics and geometric modelling.* Morgan Kaufmann publishers, 1987.

[BFK84] W. Bohm, G. Farin, and J. Kahmann. *A survey of curve and surface methods in CAGD*, pages 1–60. Elsevier Science Publishers (North Holland), 1984. in Computer Aided Geometric Design.

[BMGA92] S. Benayoun, O. Monga, A. Guéziec, and N. Ayache. Using surface curvatures for 3–D image registration. In *Proceedings of the Eleventh Conference on Computer Applications in Radiology (S'CAR92)*, Baltimore, Maryland U.S.A., June 1992.

[CCA92] Isaac Cohen, Laurent D. Cohen, and Nicholas Ayache. Using deformable surfaces to segment 3-D images and infer differential structures. *Computer Vision, Graphics, and Image Processing: Image Understanding*, 56(2), 1992.

[CH57] R. Courant and D. Hilbert. *Methods of Mathematical Physics.* Interscience, London, 1957.

[Cia88] P. G. Ciarlet. *Introduction à l'analyse numérique matricielle et à l'optimisation.* Masson, 1988.

[Cin87] Philippe Cinquin. Application des fonctions-spline au traitement d'images numériques. Thèse de Doctorat de mathématiques, Septembre 1987.

[Cut89] Court B. Cutting. Applications of computer graphics to the evaluation and treatment of major craniofacial malformation. In Jayaram K.Udupa and Gabor T. Herman, editors, *3–D Imaging in Medicine.* CRC Press, 1989.

[Dan80] P.E. Danielsson. Euclidean distance mapping. *Computer Graphics and Image Processing*, 14:227–248, 1980.

[dB78] C. de Boor. *A practical Guide to Splines.* Springer-Verlag, 1978.

[dC76] Manfredo P. do Carmo. *Differential Geometry of Curves and Surfaces.* Prentice-Hall, Englewood Cliffs, 1976.

[Far88] G. Farin. *Curves and surfaces for computer aided geometric design.* Academic Press, 1988.

[GA92] A. Guéziec and N. Ayache. Smoothing and matching of 3−D-space curves. In *Proceedings of the Second European Conference on Computer Vision 1992*, Santa Margherita Ligure, Italy, May 1992.

[Hum83] Robert Hummel. Sampling for spline reconstruction. *SIAM Journal of Applied Mathematics*, 43(2):278–288, April 1983.

[JAJ67] E.N. Nilson J.H. Ahlberg and J.L.Walsh. *The Theory of Splines and Their Applications.* Academic Press, New York, 1967.

[KWT87] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1:321–331, 1987.

[LMLC91] F. Leitner, I. Marque, S. Lavallée, and P. Cinquin. Dynamic segmentation: Finding the edges with snake splines. In P.J. Laurent, A. Le Méhauté, and L.L. Schumaker, editors, *Curves and Surfaces*, pages 279–284. Academic Press, Boston, 1991.

[Mae88] Einar Maeland. On the comparison of interpolation methods. *IEEE Transactions on Medical Imaging*, 7(3):213–217, September 1988.

[Mal92] Gregoire Malandain. Filtrage, topologie et mise en correspondance d'images médicales. Thèse de Doctorat, Septembre 1992.

[MBF92] Olivier Monga, Serge Benayoun, and Olivier D. Faugeras. Using third order derivatives to extract ridge lines in 3−D images. In *Proceedings of the IEEE Conference on Vision and Pattern Recognition*, Urbana Champaign, June 1992.

[MDMC90] Olivier Monga, Rachid Deriche, Gregoire Malandain, and Jean-Pierre Cocquerez. Recursive filtering and edge closing: two primary tools for 3−D edge detection. In *First European Conference on Computer Vision (ECCV), also submitted to Image and Vision Computing*, Antibes, April 1990.

[MSMM90] S. Menet, P. Saint-Marc, and G. Medioni. Active contour models: Overview, implementation and applications. In *System, Man and Cybernetic*, pages 194–199, 1990.

[NA92] Chahab Nastar and Nicholas Ayache. Fast segmentation, tracking and analysis of deformable objects. Technical Report 1783, INRIA, October 1992.

[PKT88] Anthony J. Parker, Robert V. Kenyon, and Donald E. Troxel. Comparison of interpolating methods for image resampling. *IEEE Transactions on Medical Imaging*, 7(3):31–39, September 1988.

[PS91] Alex Pentland and Stan Sclaroff. Closed-form solutions for physically based shape modelling and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-13(7):715–729, July 1991.

[TG92] J.P. Thirion and A. Gourdon. The 3–D marching lines algorithm and its application to crest lines extraction. Technical Report 1672, INRIA, May 1992.

[TGM+92] J.P. Thirion, A. Gourdon, O. Monga, A. Guéziec, and N. Ayache. Automatic registration of 3–D CAT–SCAN images using surface curvature. In *Proceedings of the Fourteenth Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS 92), Satellite Symposium on 3–D Advanced Image Processing in Medicine*, Rennes, France, November 1992.

[WR71] J. H. Wilkinson and C. Reinsch. *Handbook for Automatic Computation: Linear Algebra*, volume 2 of *Die Grundlehren der mathematischen Wissenschaften in Einzeldartsellungen*. Springer-Verlag, 1971.

ISSN 0249-6399