Thin nets and Crest lines : Application to Satellite Data and Medical Images

Olivier Monga, Nasser Armande, Philippe Montesinos

INRIA
INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET AUTOMATIQUE

Thin nets and Crest lines : Application to Satellite Data and Medical Images

Olivier MONGA, Nasser ARMANDE, Philippe MONTESINOS

N° 2480

February 1995

PROGRAMME 4

Robotique,

image

et vision

ISSN 0249-6399

ROCQUENCOURT

Thin nets and Crest lines : Application to Satellite Data and Medical Images

Olivier MONGA, Nasser ARMANDE, Philippe MONTESINOS

Programme 4 — Robotique, image et vision

Projet SYNTIM

Rapport de recherche n ° 2480 — February 1995 — 27 pages

Abstract: In this paper, we describe a new approach for extracting *thin nets* in grey level images. The key point of our approach is to model thin nets as the crest lines of the image surface. Crest lines are the lines where the magnitude of the maximum curvature is locally maximum in the corresponding principal direction. We define these lines using first, second and third derivatives of the image. We compute the image derivatives using recursive filters approximating the Gaussian filter and its derivatives. Using an adapted scale factor, we apply this approach to the extraction of roads in satellite data and blood vessels in medical images. We also apply this method to the extraction of the crest lines in depth maps of human faces.

Key-words: Crest lines, Satellite images, Blood vessels, Medical Images, Roads, Thin nets, Differential Geometry, Curvatures

(Résumé : tsvp)

Unité de recherche INRIA Rocquencourt

Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)

Extraction de réseaux fins et lignes de crête : Application aux images satellites et aux images médicales

Résumé : Dans cet article, nous décrivons une nouvelle approche pour extraire des réseaux fins dans une image de niveaux de gris. Le point clé de notre approche est de modéliser les réseaux fins comme les lignes de crête de la surface image. Les lignes de crête sont les lignes où la valeur absolue de la courbure maximum est localement maximum dans la direction principale correspondante. On définit ces lignes en utilisant les dérivées d'ordre 1,2,3 de l'image. Nous calculons les dérivées partielles de l'image avec des filtres récursifs approximant le filtre gaussien et ses dérivées. En utilisant un facteur d'échelle, nous appliquons cette méthode à l'extraction des routes dans des images satellites et des vaisseaux sanguins dans des images médicales. Nous présentons aussi des résultats sur l'extraction de lignes de crête dans des cartes de profondeur de visages.

Mots-clé : Lignes de crêtes, images satellites, géométrie differentielle, reseaux fins, routes, vaisseaux sanguins, images médicales, courbures

1 Introduction

In many images, Thin Nets (TN) correspond to important features [GJ 94] [GJ 91]. For instance, in aerial and medical images, TN are attached respectively to roads and blood vessels. TN are formed by the points where the grey-level is locally extremum in a given direction. This direction is the normal to the curve traced by the TN at this point. Classic edge detection algorithms [Can 83] [Har 84] [TP 86] [Der 87] are not able to detect TN. In this paper, we propose a new method to detect TN using differential geometry. An important point of our approach is its ability to identify TN by crest lines on the surface defined by the image [PB 85].

We use the definition of crest lines proposed in [MBF 92] for 3D volumic images, i.e. the points where the magnitude of the maximum curvature is maximum along the maximum curvature direction. In the present case, we come up with different expressions of the surface differential properties using partial derivatives of the 2D images. The principal curvatures and principal curvature directions of the surface defined by the image are expressed using first and second order partial derivatives of this image. These are the same as those given in reference [PB 85]. We propose a new criterion which uses in addition to the first and second order partial derivatives a third partial derivative for characterizing the crest lines. We stress that this criterion is different from the one proposed in [MBF 92] for 3D volumic images, although it characterizes the same differential property. To compute the partial derivatives, we use an extension to the third order of the recursive filters approximating the Gaussian and its derivatives [Der 93][MLD 93]. We come up with a three-stage algorithm which extracts the TN :

1. Computation of $1, 2, 3$ order partial derivatives of the image using recursive Gaussian filters.

2. Computation of the principal curvatures, the principal curvature directions and the directional Derivative of the Maximum Curvature along the corresponding principal direction (DMC).

3. Extraction of the zero-crossings of the DMC which form crest lines or TN.

We present the partial derivative computations in the first section. In the second section, we give the definition of TN and we compute the DMC and the zero-crossings of the DMC. We finish this paper by the algorithm of the extraction of TN, the experimental results and an appendix on computation of our image derivatives. We have tested this method on satellite data and medical images in which TN correspond to roads and blood vessels respectively. We have also applied this method on depth maps of human faces provided with a stereo vision process [FL 94] to extract the crest lines. The quality of the results obtained clearly shows the flexibility and the pertinence of our approach.

2 Computing partial derivatives of a 2D image using the Gaussian filter and its derivatives

Let $I(x, y)$ be a 2D image. We look for the partial derivatives of $I(x, y)$ using the following formula :

$$I_{x^m y^p} = \frac{\partial^n (I(x,y))}{\partial x^m \partial y^p}, \quad \mathrm{n = m + p} \tag{1}$$

Here, we use the subscript notation $I_{x^m y^p}$ to describe of the derivative orders. If $g(x, y) = g(x)g(y)$ is the impulse response of a smoothing filter, the restored image I_r is equal to $I * g$, where $*$ is the convolution product. Typically, when the properties of the convolution product are used, we obtain :

$$\frac{\partial^n I_r}{\partial x^m \partial y^p} = \frac{\partial^n (I * g)}{\partial x^m \partial y^p} = I * (\frac{\partial^n g}{\partial x^m \partial y^p}) \tag{2}$$

where, the impulse response of the filter which computes $I_{x^m y^p}$ can be defined by : $\frac{\partial^n g}{\partial x^m \partial y^p}$. Using the separability property, we use the Gaussian smoothing filter and its derivatives up to the third order described in the Appendix to compute the derivatives of a 2D image. We use these filters because all the results of the forthcoming algorithms strongly depend on the way the partial derivatives are computed. We come up with the following algorithm for computing $\frac{\partial^n g}{\partial x^m \partial y^p}, m + p \leq 3$ [MLD 93] :

$$\text{for } (m, p) \text{ such that } (m + p) \leq 3 \text{ do}$$
$$\left[\begin{array}{l} R = I * g_m(x) \\ I_{x^m y^p} = R * g_p(y) \end{array} \right.$$

where the convolution products are implemented using the recursive implementation of the filters. In the next section, we use $I_x, I_{xx}, I_{xxx}, I_y, I_{yy}, I_{yyy}, I_{xy}, I_{xyy}, I_{xxy}$ which we estimate using this algorithm.

3 Using differential properties of the image surface to extract the TN

3.1 TN and crest lines of an image surface

In the grey level images, the Thin Nets (TN) are formed by the points where the luminance is locally extremum in a given direction. This direction is the normal to the curve traced by the TN at this point. Classic edge models [Can 83] [Har 84] [TP 86] [Der 87] are not able to characterize TN. One way to tackle this problem is to define TN, using the differential properties of the image surface (an image $I(x,y)$ defines the surface $(x,y,I(x,y))$). Some work on corner and vertex detection [GD 91] [Nob 88] use the same paradigm. We define TN as the crest lines of the image surface for an adapted scale factor. We use the definition of crest lines proposed in [MBF 92] for 3D volumic images, i.e. the points where the maximal curvature is locally maximal in the corresponding principal direction. These lines can be characterized as the zero-crossings of a coefficient, noted e_m, which they called *extremality*. e_m is the directional Derivative of the Maximum principal Curvature (DMC) in the corresponding principal direction defined by :

$$e_m = \nabla \vec{k_{max}} \cdot \vec{t_{max}} \tag{3}$$

We use expression e_m=0 in order to extract TN or crest lines according to scale factor, but our explicit equation of e_m is completely different because it must be obtained from a 3D surface defined by a 2D image.

We can also extract the TN by the zero-crossings of the following expression :

$$g_m = \vec{\nabla} I(x,y) \cdot \vec{t_{max}} \tag{4}$$

where $I(x,y)$ is the 2D grey level image function. The explicit expression of g_m shows that g_m can be computed directly from the first and second derivatives of the image , an interesting

aspect of this method. We have tested this expression on a number of images provided with satellite data, medical images and depth maps but unfortunately, the quality of the results obtained is not acceptable for various reasons including noise. In particular, for the depth maps it is evident that this expression is not a solution of our problem. Moreover conceptually speaking, this method does not stem from the differential geometry properties of the surfaces. This is why we have chosen $e_m = 0$ using the third derivative of the image.

3.2 From partial derivatives to crest lines of an image surface

3.2.1 Computation of the directional Derivative of the Maximum principal Curvature (DMC)

Let us consider the suface $\vec{S}(x, y)$ associated to the grey-level intensity of a 2D image $I(x, y)$ described by the equation :

$$\vec{S}(x, y) = (x, y, I(x, y))^t \tag{5}$$

At each point P of this surface, there is an infinite number of curvatures attached to each direction $\vec{t}$ in the tangent plane at P [Car 76]. There are two privileged directions, called the principal directions ($\vec{t_1}$ and $\vec{t_2}$), which correspond to the two extremal values of the curvatures k_1 and k_2, except for the umbilic and flat points, where we can not define two principal directions. One of these two principal curvatures is maximal in absolute value and is called the maximal curvature. The other is the minimal curvature.

We define the tangent plane $S_T(x, y)$ of surface $\vec{S}(x, y)$ described in Equation 5, at each point P by :

$$\begin{aligned} \vec{S_x} &= (\frac{\partial \vec{S}}{\partial x}) = (1, 0, I_x)^t \\ \vec{S_y} &= (\frac{\partial \vec{S}}{\partial y}) = (0, 1, I_y)^t \\ S_T(x, y) &= \{\vec{S_x}, \vec{S_y}\} \end{aligned} \tag{6}$$

where I_x and I_y represent respectively the partial derivatives of the image function along x and y.

The first and second Fundamental Forms can be used to compute the principal curvatures and principal directions at each point P of the surface $\vec{S}(x, y)$ [Car 76]. The coefficients of

the first Fundamental Form, in the basis $\{\vec{S_x}, \vec{S_y}\}$, form a matrix F_1 defined by:

$$F_1 = \begin{pmatrix} e & f \\ f & g \end{pmatrix}$$

where

$$e = \|\vec{S_x}\|^2 \quad f = \vec{S_x}.\vec{S_y} \quad g = \|\vec{S_y}\|^2 \tag{7}$$

Using the partial derivatives of the image $I(x, y)$, for derivative of $\vec{S}(x, y)$, we have:

$$e = 1 + I_x^2 \quad f = I_x.I_y \quad g = 1 + I_y^2 \tag{8}$$

Similarly, we obtain for the second Fundamental Form, in the same basis, a matrix noted F_2:

$$F_2 = \begin{pmatrix} l & m \\ m & n \end{pmatrix}$$

The elements l, m, and n can be written as:

$$l = \vec{N}.\vec{S_{xx}} \quad m = \vec{N}.\vec{S_{xy}} \quad n = \vec{N}.\vec{S_{yy}} \tag{9}$$

where $\vec{S_{xx}}, \vec{S_{xy}}, \vec{S_{yy}}$ are respectively the second derivatives of $\vec{S}(x, y)$ along the corresponding axes and $\vec{N}$ is the surface unit normal given by:

$$\vec{N} = \frac{\vec{S_x} \wedge \vec{S_y}}{\|\vec{S_x} \wedge \vec{S_y}\|} \tag{10}$$

Similarly, using partial derivatives, we obtain:

$$\begin{aligned} l &= \frac{I_{xx}}{\sqrt{(1+I_x^2+I_y^2)}} \\ m &= \frac{I_{xy}}{\sqrt{(1+I_x^2+I_y^2)}} \\ n &= \frac{I_{yy}}{\sqrt{(1+I_x^2+I_y^2)}} \end{aligned} \tag{11}$$

The principal curvatures and principal directions correspond respectively to the eigenvalues (k_1 and k_2) and the eigenvectors ($\vec{t_1}$ and $\vec{t_2}$) of the Weingarten endomorphism, the matrix of which is:

$$W = F_1^{-1}.\ F_2 = \begin{pmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{pmatrix}$$

The coefficients of W are computed as follows :

$$\begin{aligned} w_{11} &= \tfrac{1}{H}(gl - fm) \\ w_{12} &= \tfrac{1}{H}(gm - fn) \\ w_{21} &= \tfrac{1}{H}(em - fl) \\ w_{22} &= \tfrac{1}{H}(en - fm) \qquad (\text{with} \quad H = eg - f^2) \end{aligned} \tag{12}$$

By definition, K=$k_1 k_2$ is the Gaussian curvature and S=$\frac{1}{2}(k_1 + k_2)$ is the average curvature at each point of the surface $\vec{S}(x, y)$. Moreover, K and S are also respectively the determinant and half the trace of the matrix W. Therefore, the expressions of K and S are given by :

$$\begin{aligned} K &= w_{11}w_{22} - w_{12}w_{21} = \tfrac{1}{H}(ln - m^2) \\ S &= \tfrac{1}{2}(w_{11}w_{22}) = \tfrac{1}{2H}(en - 2fm + gl) \end{aligned} \tag{13}$$

Using Equations 8 and 11, K and S are expressed by :

$$\begin{aligned} K &= \tfrac{1}{d}(I_{xx}I_{yy} - {I_{xy}}^2) \\ S &= \tfrac{1}{2d^{3/2}}(I_{xx} - 2I_x I_{xy} I_y + I_{xx}{I_y}^2 + I_{yy} + I_y y {I_x}^2) \quad (\text{with} \quad d = 1 + {I_x}^2 + {I_y}^2) \end{aligned} \tag{14}$$

Finally, the principal curvatures k_1 and k_2 are the solutions of a second-order equation given as follows :

$$\begin{aligned} k_1 &= S + \sqrt{S^2 - K} \\ k_2 &= S - \sqrt{S^2 - K} \end{aligned} \tag{15}$$

where the explicit expressions can be written as :

$$\begin{aligned} k_{1,2} = \tfrac{1}{2d^{3/2}} \quad & [I_{xx} - 2I_x I_{xy} I_y + I_{xx} I_y^2 + I_{yy} + I_x^2 I_{yy} \\ & \pm[\ I_{xx}^2 + 4I_{xy}^2 - 2I_{xx}I_{yy} + 4I_x^2 I_{xy}^2 + I_x^4 I_{yy}^2 - \\ & \quad 2I_{xx}I_y^2 I_{yy} - 4I_x I_{xx} I_{xy} I_y^3 - 2I_x^2 I_{xx} I_{yy} - \\ & \quad 4I_x I_{xy} I_y I_{yy} + 2I_{xx}^2 I_y^2 + 4I_{xy}^2 I_y^2 + I_{xx}^2 I_y^4 + \\ & \quad I_{yy}^2 + 2I_x^2 I_{yy}^2 - 4I_x^3 I_{xy} I_y I_{yy} - 4I_x I_{xx} I_{xy} I_y + \\ & \quad 2I_x^2 I_{xx} I_y^2 I_{yy} + 4I_x^2 I_{xy}^2 I_y^2 \\ & \ ]^{1/2} \\ &]. \quad (\text{with} \quad d = 1 + I_x^2 + I_y^2) \end{aligned} \tag{16}$$

The expressions of k_1 or k_2 show that the principal curvatures of a surface can be computed directly from the first and second derivatives of the image. Once k_1 and k_2 are computed,

the principal directions $\vec{t_1}$ and $\vec{t_2}$, which are the two eigenvectors of the matrix W, may be represented in the basis $\{\vec{S_x}, \vec{S_y}\}$ with $\vec{t_i} = (u_i\ v_i)^t$ and thus satisfy the system of:

$$\begin{array}{rcrcl} (w_{11} - k_i)u_i & + & w_{12}v_i & = 0 & \\ w_{21}u_i & + & (w_{22} - k_i)v_i & = 0 & i \in 1, 2. \end{array} \tag{17}$$

For each $i \in 1, 2$, the two equations are dependent, and thus the solution set for $(u_i\ v_i)^t$ will lie along a line colinear with $\vec{t_i}$. We can compute $\vec{t_i}$ with either of the two equations presented in 17, which gives us two vectors:

$$\vec{t_1} = \begin{pmatrix} w_{12} \\ k_1 - w_{11} \end{pmatrix} \qquad \vec{t_2} = \begin{pmatrix} w_{12} \\ k_2 - w_{11} \end{pmatrix} \tag{18}$$

Let $k_{max} = k_1$ be the maximum curvature in absolute value ($|k_1| \geq |k_2|$) and also $\vec{t_{max}} = \vec{t_1}$. The gradient of k_{max} in the associated direction $\vec{t_{max}}$, called the directional Derivative of the Maximum Curvature (DMC), is given by:

$$DMC = \nabla\vec{k_{max}} \ . \ \vec{t_{max}} \tag{19}$$

where

$$\nabla\vec{k_{max}} = \begin{pmatrix} \frac{\partial k_{max}}{\partial x} \\ \frac{\partial k_{max}}{\partial y} \end{pmatrix} \qquad \vec{t_{max}} = \begin{pmatrix} w_{12} \\ k_{max} - w_{11} \end{pmatrix} \tag{20}$$

The expression of the DMC can also be written as:

$$DMC = (\frac{\partial k_{max}}{\partial x})(w_{12}) + (\frac{\partial k_{max}}{\partial y})(k_{max} - w_{11}) \tag{21}$$

We will now present in detail, the explicit expression of the DMC, where $k_{max} = k_1$ and $\vec{t_{max}} = \vec{t_1}$. We have computed the derivatives of k_{max} along x and y. These results can be written as follows:

$$
\begin{aligned}
(\tfrac{\partial k_{max}}{\partial x}) = \quad & \tfrac{3}{2\,d^{5/2}}\ [(-I_{xx}+2I_xI_{xy}I_y-I_{xx}I_y^2-I_{yy}-I_x^2I_{yy})(I_xI_{xx}+I_yI_{yy})]+\\
& \tfrac{1}{2\,d^{3/2}}\ [I_{xxx}-2I_xI_{xxy}I_y-2I_{xx}I_{xy}I_y+I_{xxx}I_y^2+2I_xI_{xx}I_{yy}-\\
& 2I_xI_{xy}I_{yy}+2I_{xx}I_yI_{yy}+I_{yyx}+I_x^2I_{yyx}]+\\
& [\ \tfrac{1}{d^4}(\ (6I_xI_{xx}+6I_yI_{yy})(-I_{xx}^2-4I_{xy}^2-4I_x^2I_{xy}^2+4I_xI_{xx}I_{xy}I_y-I_x^4I_{yy}^2\ -\\
& 2I_{xx}^2I_y^2-4I_{xy}^2I_y^2-4I_x^2I_{xy}^2I_y^2+4I_xI_{xx}I_{xy}I_y^3-I_{xx}^2I_y^4+2I_{xx}I_{yy}-2I_x^2I_{yy}^2+\\
& 2I_x^2I_{xx}I_{yy}+4I_xI_{xy}I_yI_{yy}+4I_x^3I_{xy}I_yI_{yy}+2I_{xx}I_y^2I_{yy}-2I_x^2I_{xx}I_y^2I_{yy}-I_{yy}^2)\\
&)+\tfrac{1}{d^3}(\quad 2I_{xx}I_{xxx}+8I_{xxy}I_{xy}+8I_x^2I_{xxy}I_{xy}+8I_xI_{xx}I_{xy}^2-\\
& 4I_xI_{xx}I_{xxy}I_y-4I_{xx}^2I_{xy}I_y-4I_xI_{xxx}I_{xy}I_y+\\
& 4I_{xx}I_{xxx}I_y^2+8I_{xxy}I_{xy}I_y^2+8I_x^2I_{xxy}I_{xy}I_y^2+\\
& 8I_xI_{xx}I_{xy}^2I_y^2-4I_xI_{xx}I_{xxy}I_y^3-4I_{xx}^2I_{xy}I_y^3-\\
& 4I_xI_{xxx}I_{xy}I_y^3+2I_{xx}I_{xxx}I_y^4-4I_xI_{xx}^2I_{yy}-\\
& 2I_{xxx}I_{yy}-2I_x^2I_{xxx}I_{yy}-4I_xI_{xx}I_{xy}I_{yy}+\\
& 4I_{xx}^2I_yI_{yy}-4I_xI_{xxy}I_yI_{yy}-4I_x^3I_{xxy}I_yI_{yy}-\\
& 4I_{xx}I_{xy}I_yI_{yy}-12I_x^2I_{xx}I_{xy}I_yI_{yy}+8I_{xy}^2I_yI_{yy}+\\
& 8I_x^2I_{xy}^2I_yI_{yy}+4I_xI_{xx}^2I_y^2I_{yy}-2I_{xxx}I_y^2I_{yy}+\\
& 2I_x^2I_{xxx}I_y^2I_{yy}-12I_xI_{xx}I_{xy}I_y^2I_{yy}+\\
& 4I_{xx}^2I_y^3I_{yy}+4I_xI_{xx}I_{yy}^2+4I_x^3I_{xx}I_{yy}^2-\\
& 4I_xI_{xy}I_{yy}^2-4I_x^3I_{xy}I_{yy}^2-4I_{xx}I_yI_{yy}^2+\\
& 4I_x^2I_{xx}I_yI_{yy}^2-2I_{xx}I_{yyx}-2I_x^2I_{xx}I_{yyx}-\\
& 4I_xI_{xy}I_yI_{yyx}-4I_x^3I_{xy}I_yI_{yyx}-2I_{xx}I_y^2I_{yyx}+\\
& 2I_x^2I_{xx}I_y^2I_{yyx}+2I_{yy}I_{yyx}+4I_x^2I_{yy}I_{yyx}+\\
& 2I_x^4I_{yy}I_{yyx}\\
& \quad)\\
&]\ /\ \tfrac{4}{d^{3/2}}\ [I_{xx}^2+4I_{xy}^2+4I_x^2I_{xy}^2-4I_xI_{xx}I_{xy}I_y+\\
& 2I_{xx}^2I_y^2+4I_{xy}^2I_y^2+4I_x^2I_{xy}^2I_y^2+I_x^4I_{yy}^2-\\
& 4I_xI_{xx}I_{xy}I_y^3+I_{xx}^2I_y^4-2I_{xx}I_{yy}+2I_x^2I_{yy}^2-\\
& 2I_x^2I_{xx}I_{yy}-4I_xI_{xy}I_yI_{yy}-4I_x^3I_{xy}I_yI_{yy}-\\
& 2I_{xx}I_y^2I_{yy}+2I_x^2I_{xx}I_y^2I_{yy}+I_{yy}^2\\
&]^{1/2}\\
& \qquad (\text{ with }\ \ d=1+I_x^2+I_y^2).
\end{aligned}
\tag{22}
$$

Similarly, the derivative of k_{max} along y is given as:

$$
\begin{aligned}
(\tfrac{\partial k_{max}}{\partial y}) = \quad & \tfrac{3}{2d^{5/2}}\ [(-I_{xx}+2I_xI_{xy}I_y-I_{xx}I_y^2-I_{yy}-I_x^2I_{yy})(I_xI_{xy}+I_yI_{yy})]+ \\
& \tfrac{1}{2d^{3/2}}\ [(I_{xxy}-2I_{xy}^2I_y+I_{xxy}I_y^2+2I_{xx}I_yI_{yy}-2I_xI_yI_{yyx}+I_{yyy}+I_x^2I_{yyy})]+ \\
& [\ \tfrac{1}{d^4}(\ (6I_xI_{xy}+6I_yI_{yy})(-I_{xx}^2-4I_{xy}^2-4I_x^2I_{xy}^2+4I_xI_{xx}I_{xy}I_y- \\
& \quad 2I_{xx}^2I_y^2-4I_{xy}^2I_y^2-4I_x^2I_{xy}^2I_y^2+ \\
& \quad 4I_xI_{xx}I_{xy}I_y^3-I_{xx}^2I_y^4+2I_{xx}I_{yy}+ \\
& \quad 2I_x^2I_{xx}I_{yy}+4I_xI_{xy}I_yI_{yy}+4I_x^3I_{xy}I_yI_{yy}+ \\
& \quad 2I_{xx}I_y^2I_{yy}-2I_x^2I_{xx}I_y^2I_{yy}-I_{yy}^2- \\
& \quad 2I_x^2I_{yy}^2-I_x^4I_{yy}^2) \\
&)+\tfrac{1}{d^3}(\quad 2I_{xx}I_{xxy}+8I_xI_{xy}^3-4I_xI_{xxy}I_{xy}I_y-4I_{xx}I_{xy}^2I_y+ \\
& \quad 4I_{xx}I_{xxy}I_y^2+8I_xI_{xy}^3I_y^2-4I_xI_{xxy}I_{xy}I_y^3- \\
& \quad 4I_{xx}I_{xy}^2I_y^3+2I_{xx}I_{xxy}I_y^4-2I_{xxy}I_{yy}- \\
& \quad 2I_x^2I_{xxy}I_{yy}-8I_xI_{xx}I_{xy}I_{yy}+4I_{xx}^2I_yI_{yy}+ \\
& \quad 4I_{xy}^2I_yI_{yy}-4I_x^2I_{xy}^2I_yI_{yy}-2I_{xxy}I_y^2I_{yy}+ \\
& \quad 2I_x^2I_{xxy}I_y^2I_{yy}-8I_xI_{xx}I_{xy}I_y^2I_{yy}+ \\
& \quad 4I_{xx}^2I_y^3I_{yy}-4I_{xx}I_yI_{yy}^2+4I_x^2I_{xx}I_yI_{yy}^2+ \\
& \quad 8I_{xy}I_{yyx}+8I_x^2I_{xy}I_{yyx}-4I_xI_{xx}I_yI_{yyx}+ \\
& \quad 8I_{xy}I_y^2I_{yyx}+8I_x^2I_{xy}I_y^2I_{yyx}-4I_xI_{xx}I_y^3I_{yyx}- \\
& \quad 4I_xI_yI_{yy}I_{yyx}-4I_x^3I_yI_{yy}I_{yyx}-2I_{xx}I_{yyy}- \\
& \quad 2I_x^2I_{xx}I_{yyy}-4I_xI_{xy}I_yI_{yyy}-4I_x^3I_{xy}I_yI_{yyy}- \\
& \quad 2I_{xx}I_y^2I_{yyy}+2I_x^2I_{xx}I_y^2I_{yyy}+2I_{yy}I_{yyy}+ \\
& \quad 4I_x^2I_{yy}I_{yyy}+2I_x^4I_{yy}I_{yyy} \\
& \quad)\ \\
&]\ /\ \tfrac{4}{d^{3/2}}\ [I_{xx}^2+4I_{xy}^2+4I_x^2I_{xy}^2-4I_xI_{xx}I_{xy}I_y+ \\
& \quad 2I_{xx}^2I_y^2+4I_{xy}^2I_y^2+4I_x^2I_{xy}^2I_y^2- \\
& \quad 4I_xI_{xx}I_{xy}I_y^3+I_{xx}^2I_y^4-2I_{xx}I_{yy}-2I_x^2I_{xx}I_{yy}- \\
& \quad 4I_xI_{xy}I_yI_{yy}-4I_x^3I_{xy}I_yI_{yy}-2I_{xx}I_y^2I_{yy}+ \\
& \quad 2I_x^2I_{xx}I_y^2I_{yy}+I_{yy}^2+2I_x^2I_{yy}^2+I_x^4I_{yy}^2 \\
& \quad]^{1/2}.
\end{aligned}
\tag{23}
$$

The expression of the DMC shows that the directional derivative of the principal maximum curvature at each point of a surface can be directly computed from the first, second and third derivative of the image. Once the DMC is computed, we extract the points where DMC=0 (the zero-crossings of the DMC).

3.2.2 Extraction of the zero-crossings of the DMC

Here, we present how to extract the zero-crossings of the DMC, computed in previous section. Theoretically, the absolute value of the DMC is invariant, but its sign depends on the orientation of the vectors $\vec{t_1}$ or $\vec{t_2}$ (See Equation 18) [Thi 94]. This is one reason why Thirion introduced the Gaussian extremality, an Euclidian invariant but having a different geometric meaning than crest lines. However, in our case, by assuming that $(\vec{n}, \vec{t_1}, \vec{t_2})$ is always a direct orthogonal frame, where $\vec{n}$ is the surface unit normal, we can ensure the coherence of the orientation of $\vec{t_{max}}$ (See Equation 20). Therefore, the zero-crossings of the DMC can be characterized, in most cases, as the points where the sign of the DMC changes.

Using Equation 21 and separating the flat and umbilic points [BK 76], we compute the DMC image. The sign image (S_{dmc}) can be built from the DMC image as follows :

$$\begin{cases} S_{dmc} = 1 & \text{if } DMC > 0 \\ S_{dmc} = 0 & \text{if } DMC \leq 0 \end{cases} \tag{24}$$

The zero-crossings of the DMC are the transitions $0 \nearrow 1$ or $1 \searrow 0$. We select the zero-crossings of the DMC such that the maximum curvature k_{max} is greater than a given positive threshold. This thresholding image allows us to remove the valley points having one principal curvature which is negative with a high absolute value and the other having a small value near zero. Moreover, this thresholding stage yeilds to hold only the points where the maximum curvature is locally maximum (the zero-crossings of the DMC correspond to both the maximum and the minimum of the maximum curvature).

4 Algorithm

Our TN extraction algorithm can be split as follows :

1. Computation of $1, 2, 3$ order partial derivatives of the image which can be denoted by : I_x, I_{xx}, I_{xxx}, I_y, I_{yy}, I_{yyy}, I_{xy}, I_{xyy}, I_{xxy}. We estimate these derivatives using recursive Gaussian filter and its derivatives (See Appendix).

2. Computation of the two principal curvatures k_1, k_2, as well as the two principal curvature directions $\vec{t_1}$, $\vec{t_2}$.

3. Using the direct derivatives up to the third order of the image, computation of the directional Derivative of the Maximum Curvature along the corresponding principal direction (DMC) as follows :

   - Separating the flat and umbilic points.
   - Computing the DMC on the other points, i.e. elliptic, hyperbolic and parabolic points.

$$\begin{cases} \text{if } |k_1| > |k_2| \text{ do} & k_{max} = k_1 \quad \vec{t_{max}} = \vec{t_1}, \\ \text{else} & k_{max} = k_2 \quad \vec{t_{max}} = \vec{t_2}, \\ \text{end if} & \\ \text{do} \quad DMC = \nabla \vec{k_{max}} \ . \ \vec{t_{max}} & \end{cases}$$

4. Extraction of the zero-crossings of the DMC : Using the sign image (S_{dmc}) of the DMC, we extract the transitions of $0 \nearrow 1$ or $1 \searrow 0$.

5. Extraction of the transitions $0 \nearrow 1$ or $1 \searrow 0$, where the principal maximum curvature, k_{max}, is greater than a given positive threshold.

5 Experimental results

In this section, we discuss some experimental results obtained on synthetic and real data. For the real data, we use satellite data, medical images, and depth maps of human faces. All the low-level processes (smoothing and deriving) required, are performed using an extension of the development method for implementing the Gaussian filter and its derivatives (See Appendix). The detection and extraction of k_{max}, DMC and TN are performed at pixel precision.

We illustrate the results on synthetic data in Figures 1 to 5. In Figures 1 and 4, we have added a Gaussian noise with a standard deviation σ_n, to pictures representing a circle and a diamond which have a slope of 45 degrees in the grey-level. This is done in order to make complicated noisy data. The Signal to Noise Ratio (SNR) on each image can be computed as follows :

$$SNR_{dB} = 10 \log(\frac{\sum_i \sum_j I(i,j)^2}{\sum_i \sum_j N(i,j)^2}) \tag{25}$$

where $I(i,j)$ denotes the pure image and $N(i,j)$ the noise image. We have taken σ_n=10 and SNR=14dB for these images. Figure 2 presents the results of the edge detection with a Gaussian filter ($\sigma = 1$) [Der 87] of the original images. These images show that a classic edge detection algorithm is not able to detect these pictures. Figure 3 is the results of our extraction algorithm of TN in which we illustrate the extraction of these thin nets. We have also compared in Figure 5 the response fidelity of our method using two pictures in which step edge and TN are presented. This result shows the pertinence and fidelity of " TN extraction algorithm " for thin nets and not for step edge.

Figures 6 to 11 present the results of our algorithm on satellite data in order to extract TN in these images.

In Figures 12 and 13, we have extracted blood vessels in medical images.

Finally, we have also applied this method on depth maps of human faces as illustrated in Figures 14 to 16. Figure 14(left) is a human face grey-level image. From this image, we can obtain the depth maps (Figure 14 right), using a stereo vision process. We have applied our method on the depth maps obtained and for two values of σ, we illustrate the results in Figures 15 and 16.

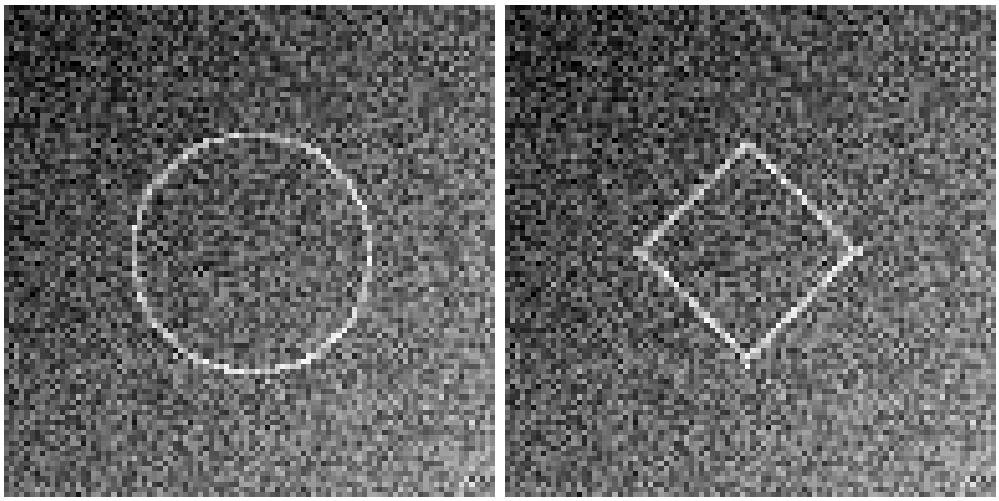

Figure 1: Original images: Circle and diamond with white Gaussian noise $\sigma_n = 10$, SNR=14dB.

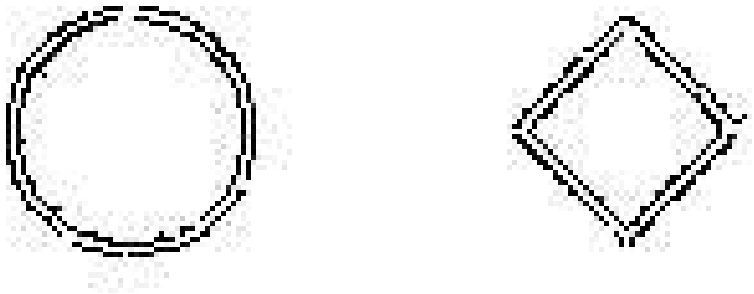

Figure 2: Edge detection by Gaussian filter ($\sigma = 1$).

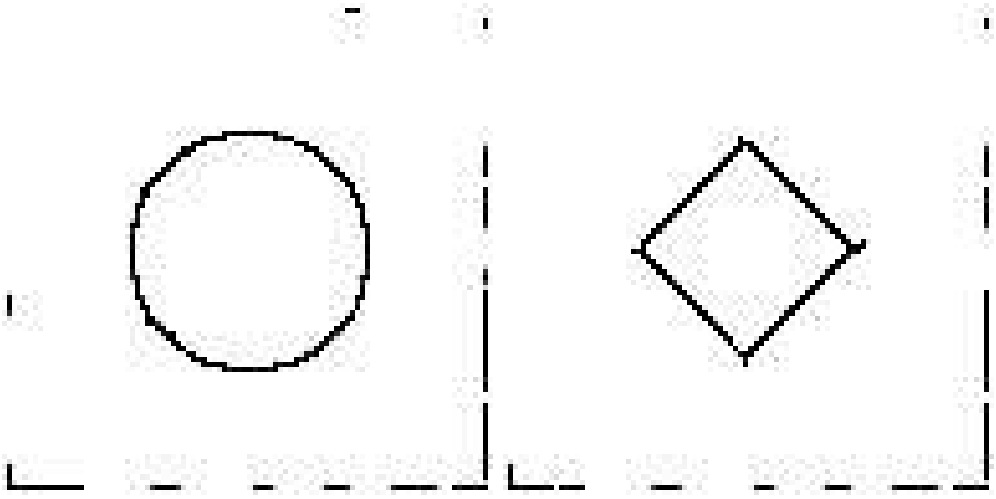

Figure 3: Extraction of TN, using the zero-crossings of the DMC threshold by k_{max} ($\sigma = 1$)

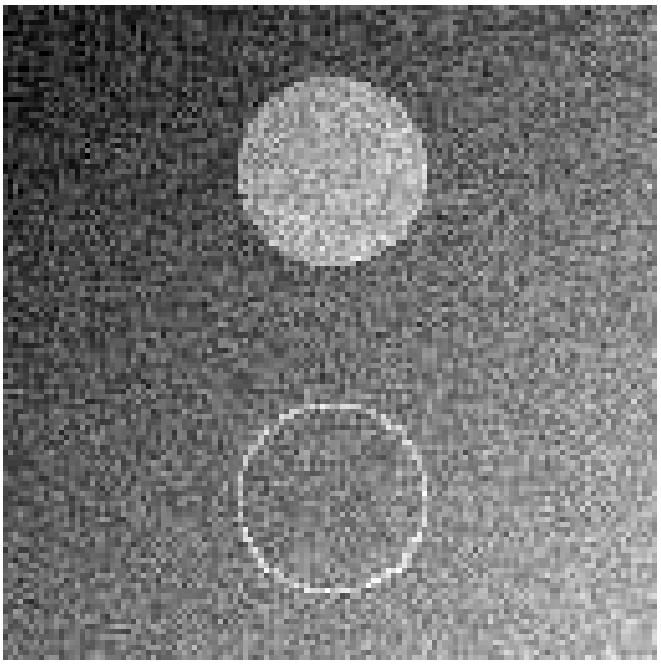

Figure 4: Original image with white Gaussian noise $\sigma_n = 10$, SNR=14dB.

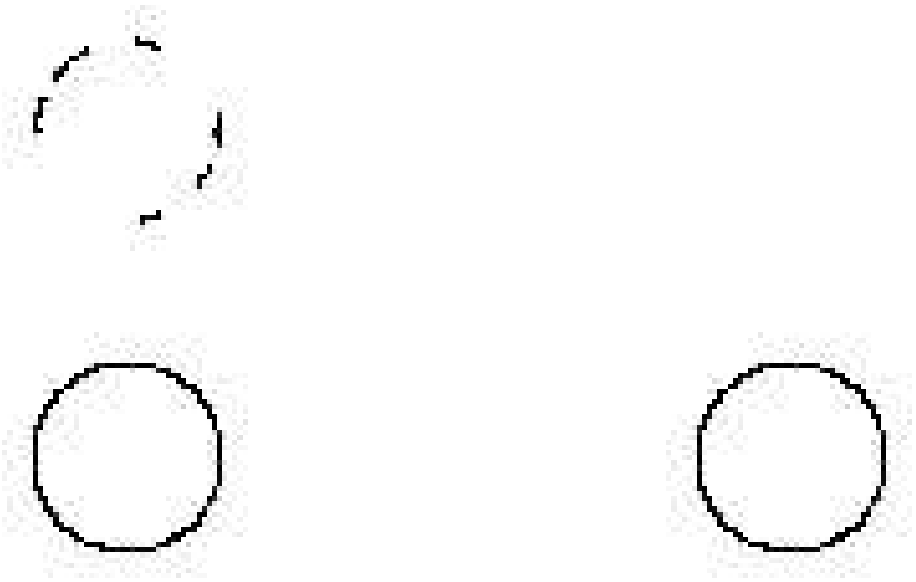

Figure 5: Extraction of TN ($\sigma = 1$). Left : threshold=0.5. Right : threshold=0.8.

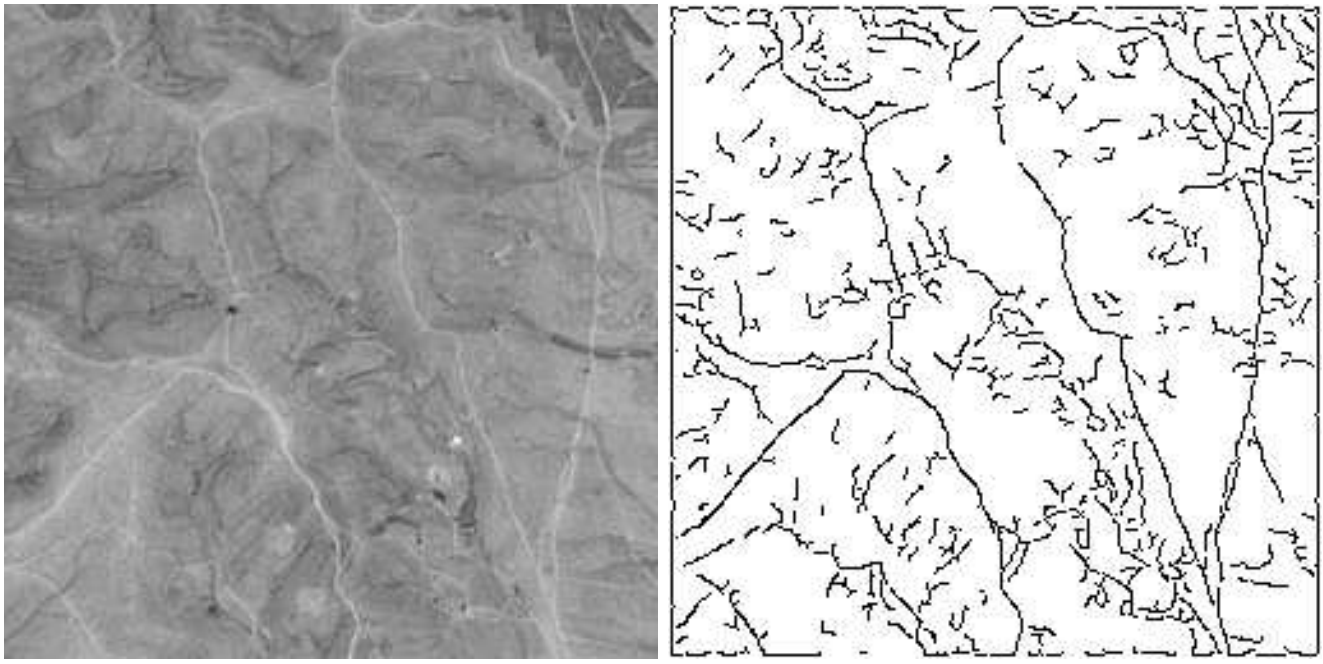

Figure 6: Left : Original image. Right : Extraction of TN ($\sigma = 1$).

Figure 7: Left : Original image. Right : Extraction of TN ($\sigma = 1$).

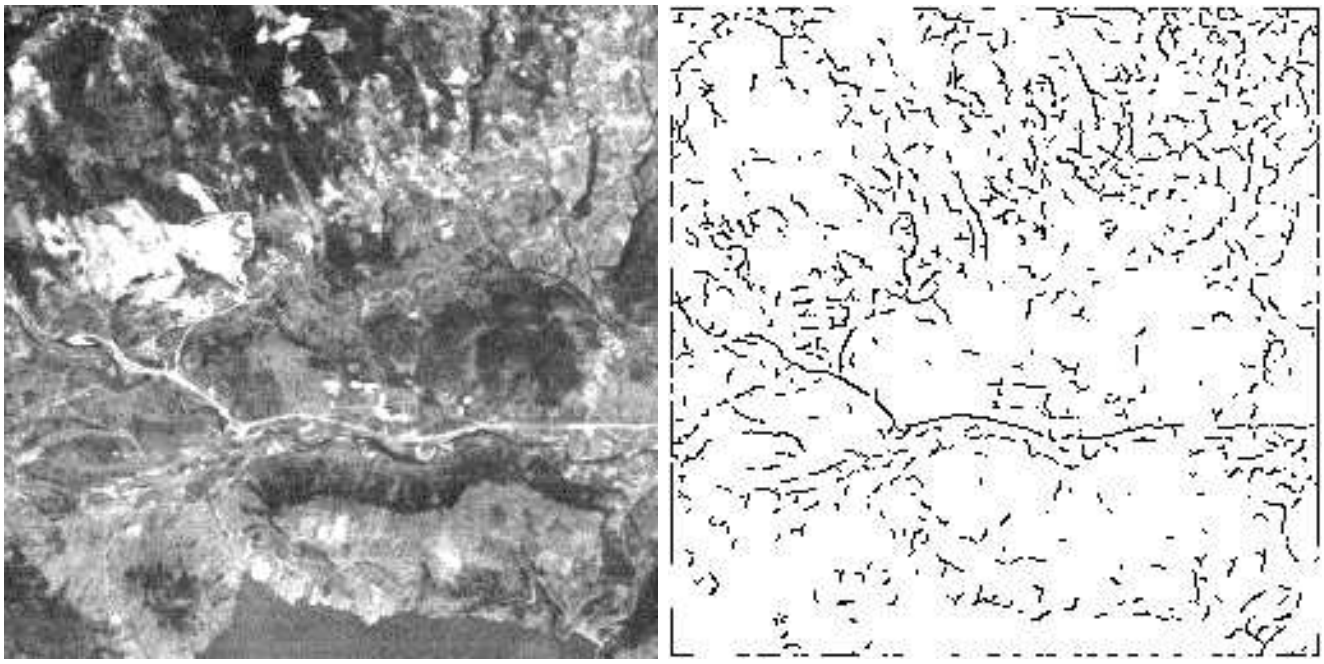

Figure 8: Left : Original image. Right : Extraction of TN ($\sigma = 1$).

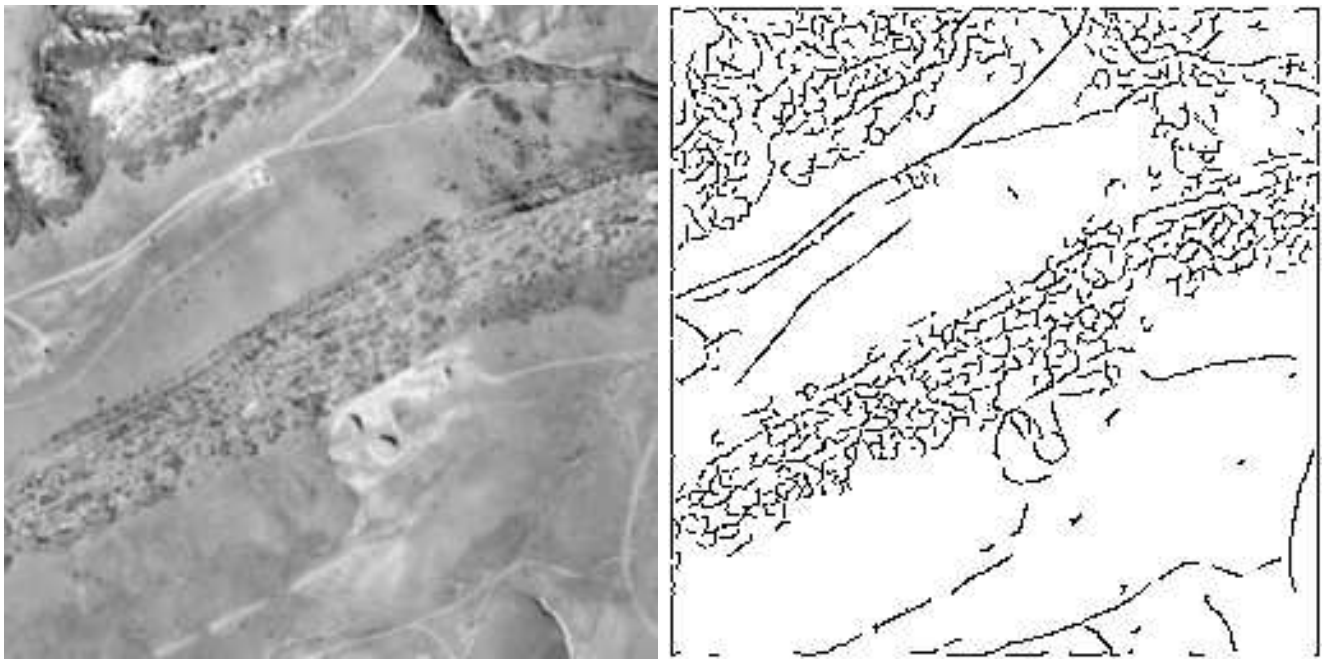

Figure 9: Left : Original image. Right : Extraction of TN ($\sigma = 1$).

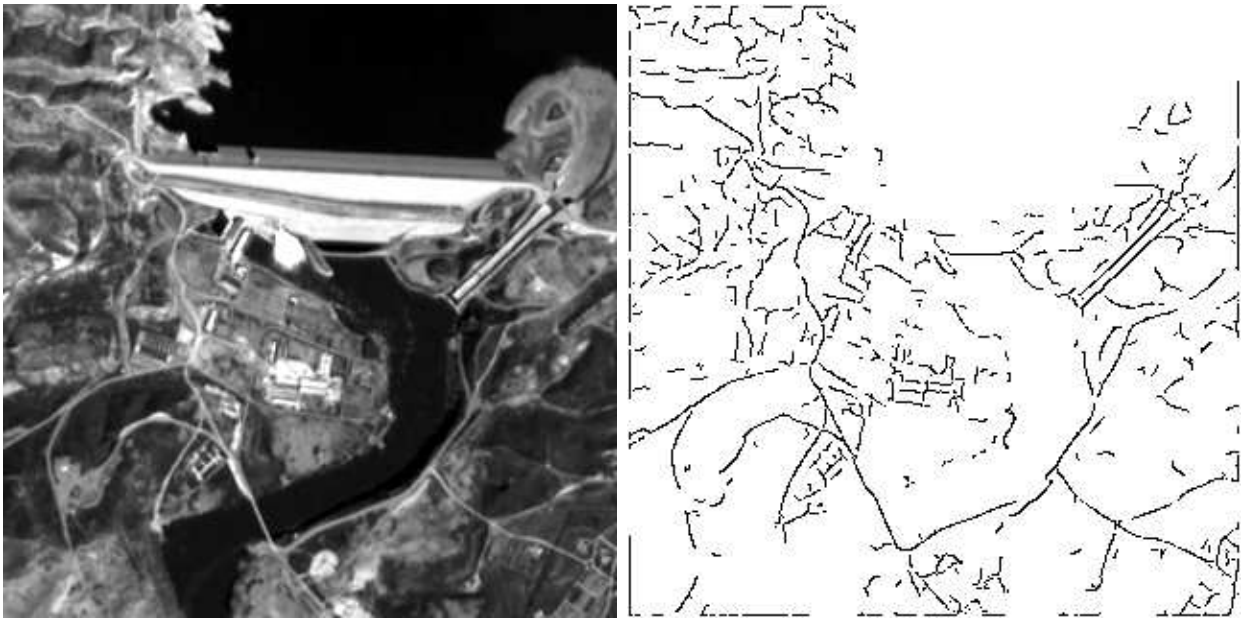

Figure 10: Left : Original image. Right : Extraction of TN ($\sigma = 1$).

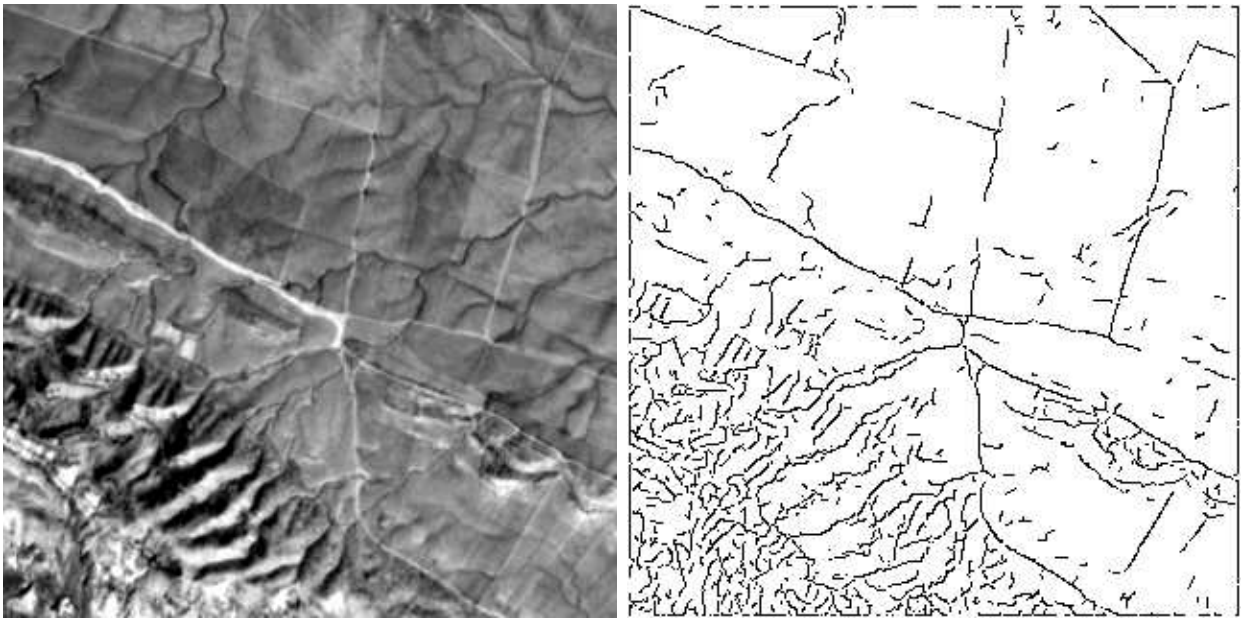

Figure 11: Left : Original image. Right : Extraction of TN ($\sigma = 1$).

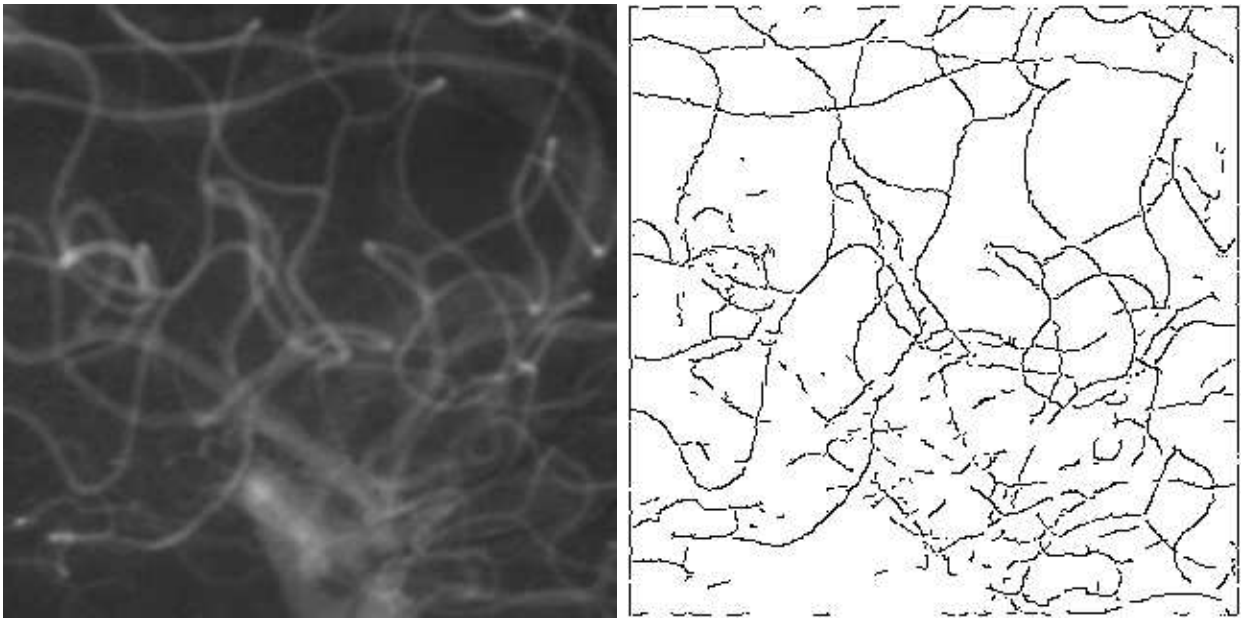

Figure 12: Left : Original image of blood vessels. Right : Extraction of TN ($\sigma = 1$)

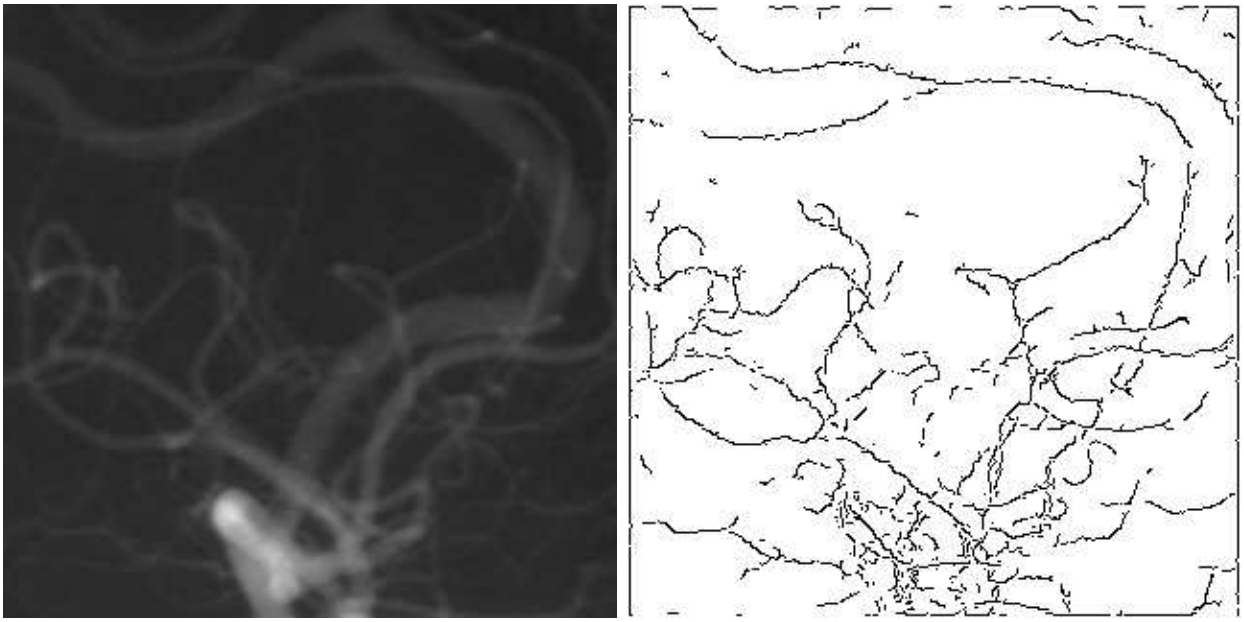

Figure 13: Left : Original image of blood vessels. Right : Extraction of TN ($\sigma = 1$)

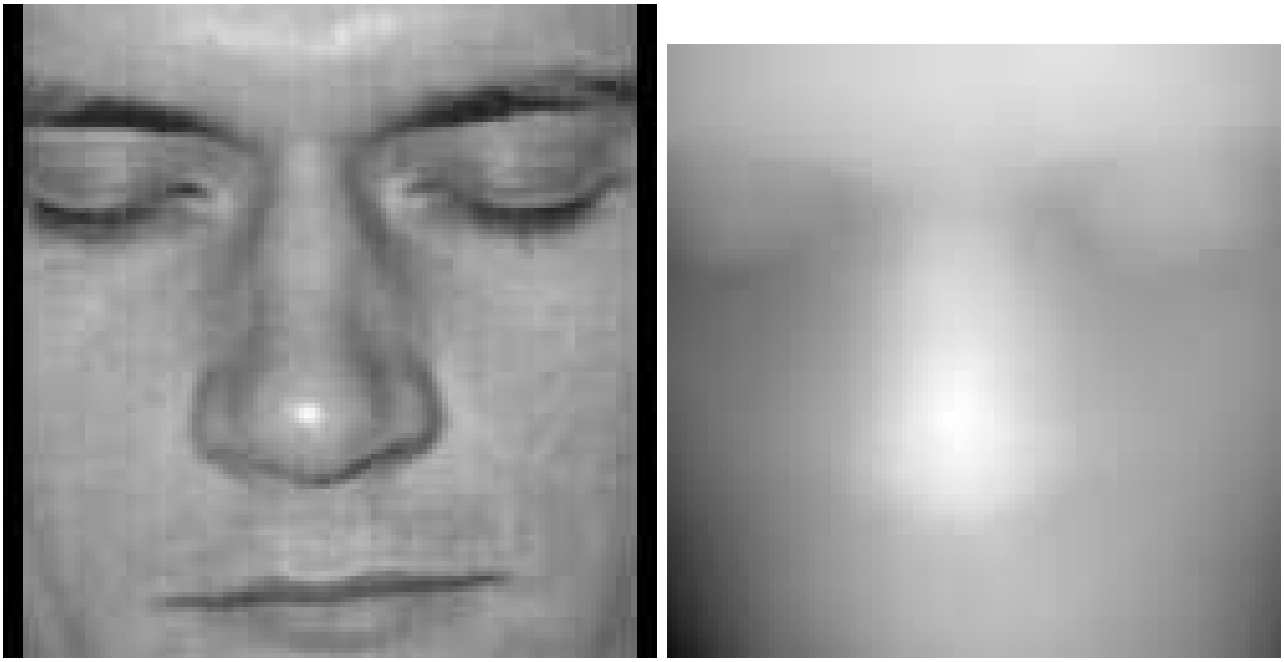

Figure 14: Left : Original image. Right : Depth maps of the original image built by a stereo vision process

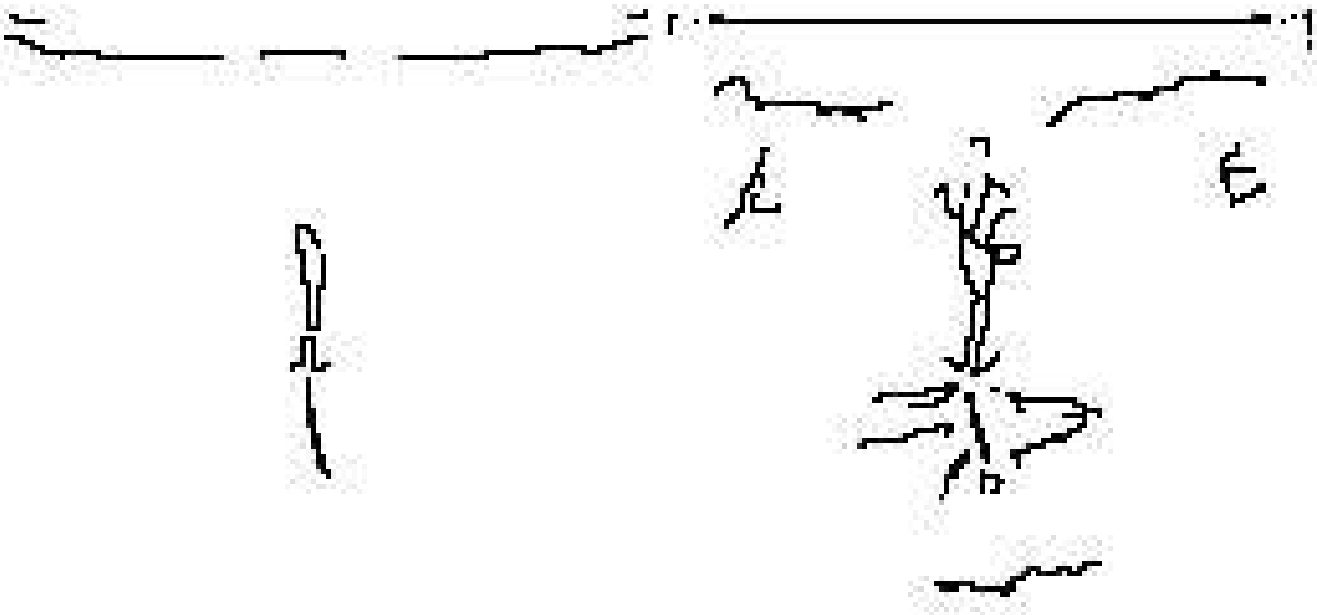

Figure 15: Crest lines detection by extraction of the zero-crossings of DMC. Left : $\sigma = 4$. Right : $\sigma = 2$

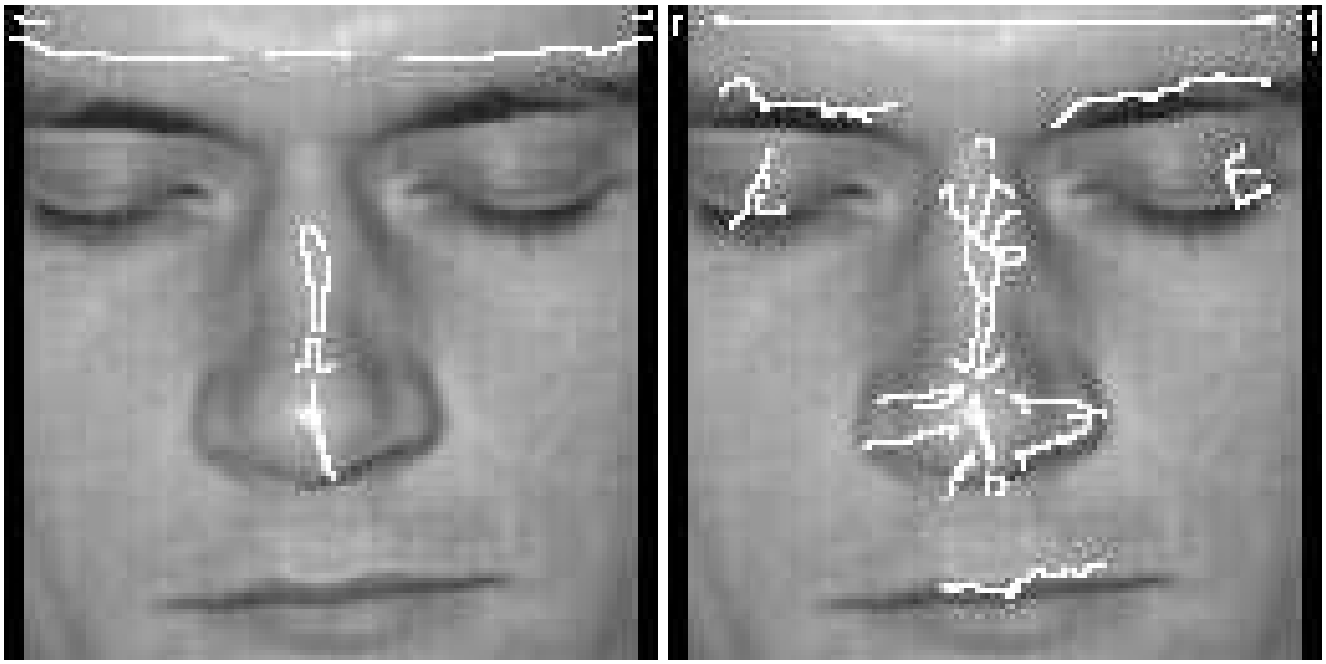

Figure 16: Superposition of the original image and crest lines detection for the same value of σ

6 Conclusion

We have presented a new approach for extracting crest lines and thin nets in grey level images, using differential properties of the image surface. This work clearly shows that even third order differential features can be robustly extracted from grey level images. A key element in achieving this is, of course, the filters used to compute the partial derivatives of the image. The quality of the experimental results shows the pertinence of our approach but also illustrates that great efforts need to be made in order to extract more sophisticated modelsthe than step edges or ramp edges.

7 Acknowledgments

We wish to thank J.P. Fabre for stimulating discussions about the differential geometry of surfaces. Thanks also to G. Szelisky and P. Fua who provided us with the medical images and depth maps. We wish also to thank H. Chabbi for her advice on this paper.

References

[Bog 93] J.E. Boggess. Identification of roads in satellite imagery using artificial neural networks : a contextual approach. *Technical Report*, Mississippi St. Univ. August. 1993.

[BK 76] J.M. Braemer and Y. Kerbrat. *Géomértrie des courbes et des surfaces*. Hermann Paris, 1976.

[Car 76] Manfredo P. Do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice Hall, 1976.

[Can 83] J.F. Canny. Finding Edges and Lines in Images. *Technical Report 720. MIT, Artificial Intelligence Laboratory*, Cambridge, Massachusetts, June 1983.

[Der 87] R. Deriche. Optimal edge detection using recursive filtering. In *Proc. First International Conference on Computer Vision*, London, June 8-12 1987.

[Der 93] R. Deriche. Recursively implementing the Gaussian and its derivatives. *INRIA Research Report*, 1993.

[FL 94] P. Fua and Y.G. Leclerc. Representation without Correspondences. *IEEE Conference on Pattern Recognition*, June 1994, Seattle, USA.

[FM 81] M. Fischler, J. Tenenbaum and H. Wolf. Detection of roads and linear structures in low-resolution aerial imagery using a multisource knowledge integration technique. *Computer Graphics and Image Processing*, 15, 201-223, 1981.

[GJ 94] D. Geman and B. Jedynak. Tracking Roads in Satellite Images by Playing Twenty Questions. *INRIA Research Report*, August, 1994.

[GJ 91] D. Geman and B. Jedynak. Detection of roads in SPOT satellite images. *Proc. IGRASS 91*, Helsinski 1991.

[GD 91] G. Giraudon and R. Deriche. On Corner and Vertex Detection. In *Conference on Computer Vision and Pattern Recognition*, Hawaii (USA), June 1991.

[Har 84] R.M. Haralick. Digital step edge from zero-crossing of second directional derivatives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(1):58-68, January 1984.

[MB 94] P. Montesinos, H.V. Blanc. Perceptual Grouping of Continuation: Application to Satellite Images. *CARI 94, 2^nd African Conference on Reasearch in Conputer Science*, Ouagadougou, Burkina Faso, October 1994.

[MB 93] O. Monga and S. Benayoun. Using differential geometry in r^4 to extract typical surface features. In *IEEE Conference on Computer Vision and Pattern Recognition*, New York, June 1993.

[MBF 92] O. Monga, S. Benayoun and D. Faugeras. Using partial derivatives of 3D images to extract typical surface features. In *IEEE Conference on Vision and Pattern Recognition*, Urbana Champaign, Illinois, July 1992. Also INRIA Research Report 1599.

[MLD 93] O. Monga, R. Lengagne, R. Deriche. Extraction of the zero-crossings of the curvature derivative in volumic 3D medical images : a multi-scale approach. In *IEEE Conference on Computer Vision and Pattern Recognition, Seattle, USA*, June 1994.

[MLD 93b] O. Monga, R. Lengagne, R. Deriche. Crest lines extraction in volumic 3D medical images : a multiscale approach. In *International Conference on Pattern Recognition (ICPR), Jerusalem, Israel*, October 1994.

[Nob 88] J.A. Noble. Finding Corners. In *Image and Vision Computing,* Vol 6, May 1988, pp 121-128.

[PB 85] J. Ponce and M. Brady. Towards a surface primal sketch. In *Proceedings, IJCAI, 1985.*

[TG 92] J.P. Thirion and A. Gourdon. The 3D marching lines algorithm and its application to crest lines extraction. *INRIA Technical Report 1672*, 1992.

[TP 86] V. Torre and T.A. Poggio. On edge detection. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 8(2):187-163, March 1986.

[TG 93] J.P. Thirion and A. Gourdon. The 3D marching lines algorithm : new results and proofs. *INRIA Technical Report 1881*, 1993.

[Thi 94] J.P. Thirion. The Extremal Mesh and Understanding of 3D Surfaces. *INRIA Research Report 2149*, December 1993.

A Appendix

A.1 Recursive filters to approximate the Gaussian filter and its derivatives

A.1.1 Approximation of the Gaussian filter and its derivatives

Here, we remind the reader of the main results of [Der 93] and we show how to approximate the Gaussian filter and its derivatives up to the third order, using Prony's method.

The 1D Gaussian smoothing filter is given by :

$$g_0(x) = \beta_0 e^{-\frac{x^2}{2\sigma^2}} \tag{26}$$

Its first, second and third derivatives are respectively as follows :

$$g_1(x) = \beta_1 x e^{-\frac{x^2}{2\sigma^2}}, \quad g_2(x) = \beta_2(1-\beta_3 x^2)e^{-\frac{x^2}{2\sigma^2}}, \quad g_3(x) = \frac{1}{\sigma^2}(\beta_4 x + \beta_5 x^3)e^{-\frac{x^2}{2\sigma^2}} \tag{27}$$

where β_i, i=0, ..., 5 are the normalization coefficients, the explicit expressions of which are given in section (A.1.2). The filters $g_0(x)$, $g_1(x)$, $g_2(x)$ and $g_3(x)$ can be approximated in a

mean square error sense by a function $h(x)$ such that [Der 93]:

$$h(x) = \sum_{i=1}^{m} \alpha_i e^{-\frac{\lambda_i x}{\sigma}} \tag{28}$$

where α_i and λ_i are the coefficients that we find in the following (in the present case, m is set equal to 4).

This approximation problem can be solved using Prony's approximation method [Der 93]. This method uses an equidistant sampling of a function $y = f(x)$ (in our case, $f(x) = e^{-\frac{x^2}{2\sigma^2}}$) at coordinates $(x_0, y_0), (x_1, y_1), ..., (x_n, y_n)$. We can set $x_r = x_0 + rh$, for $r = 0, ...n$, $n \geq m$ and define a function as follows:

$$f(x_r) = \sum_{k=1}^{m} a_k e^{\lambda'_k x_r} = \sum_{k=1}^{m} a_k e^{\lambda'_k x_0} e^{\lambda'_k rh} \quad (\text{with } \lambda' = -\frac{\lambda}{\sigma}) \tag{29}$$

Let $c_k = \alpha_k e^{\lambda'_k x_0}$ and $v_k = e^{h\lambda'_k}$. Consequently, $f(x_r)$ forms a system of n equations $(r = 0, ...n)$ as follows:

$$\begin{cases} c_1 + c_2 + ... + c_m = y_0 \\ c_1 v_1 + c_2 v_2 + ... + c_m v_m = y_1 \\ \vdots \\ c_1 v_1^n + c_2 v_2^n + ... + c_m v_m^n = y_n \end{cases} \tag{30}$$

To solve this equation system, we use a polynomial defined by:

$$\phi(v) = (v - v_1)(v - v_2)...(v - v_m), \tag{31}$$

which can also be written as:

$$\phi(v) = v^n + s_1 v^{n-1} + ... + s_m, \tag{32}$$

where $s_0 = 1, s_1 = v_1 + v_2 + ... + v_m, ...$. We now multiply, in turn, each equation belonging to system 30 by $s_m, s_{m-1}, ..., s_1, s_0$ (with $s_0 = 1$). By adding together all the terms of the equation system obtained, we get the following equation:

$$\phi(v_1)c_1 + \phi(v_2)c_2 + ... + \phi(v_m)c_m = s_m y_0 + s_{m-1} y_1 + ... + s_1 y_{m-1} + s_0 y_m = 0 \tag{33}$$

Using Equation 33 and since $\phi(v_k) = 0$ for $k = 1, ..., m$, if we shift the origin by a distance $h = 1$ to the right and repeat this operation, we obtain a new equation system as follows :

$$\begin{cases} y_{m-1}s_1 + y_{m-2}s_2 + ... + y_0 s_m = -y_m \\ y_m s_1 + y_{m-1}s_2 + ... + y_1 s_m = -y_{m+1} \\ \vdots \\ y_{n-1}s_1 + y_{n-2}s_2 + ... + y_{n-m}s_m = -y_n \end{cases} \tag{34}$$

These $n-m+1$ equations with m unknowns s_k can then be solved by a least-square method. Afterwards, we obtain the unknowns v_k by solving Equation 33. Finally, we obtain the coefficients λ'_k with the formula $\lambda'_k = \frac{\ln v_k}{h}$, the c_k from Equation 30, and the a_k from $a_k = c_k v_k^{-\frac{x_0}{h}}$.

The coefficients $\alpha_i = a_i$ and $\lambda_i = -\sigma\lambda'_i$, $i = 1, .., 4$, may be complex but conjugate (i.e. $\alpha_{m-i} = \alpha_i^*$ and $\lambda_{m-i} = \lambda_i^*$) in order to deal with real coefficients.

The advantage of this method is that we need not set σ to a particular value to compute the coefficients λ_i and α_i.

A.1.2 Computation of the normalization coefficients

For the filters g_0, g_1, g_2, g_3 of the previous section, the normalization coefficients in the continuous case are chosen so that :

$$\int_{-\infty}^{+\infty} g_0(x)dx = 1;$$

$$\int_{-\infty}^{+\infty} x g_1(x)dx = 1;$$

$$\int_{-\infty}^{+\infty} g_2(x)dx = 0 \text{ and } \int_{-\infty}^{+\infty} \frac{x^2}{2} g_2(x)dx = 1;$$

$$\int_{-\infty}^{+\infty} x g_3(x)dx = 0 \text{ and } \int_{-\infty}^{+\infty} \frac{x^3}{6} g_3(x)dx = 1.$$

These conditions ensure that the product convolution by these filters yields the correct derivatives when applied to polynomials. When the filters above are to be applied in a discrete setting, the normalization constants are expressed as follows :

$$\begin{array}{ll} \gamma_1 = \sum_{k=0}^{\infty} e^{-\frac{k^2}{2\sigma^2}} & \gamma_2 = \sum_{k=0}^{\infty} k^2 e^{-\frac{k^2}{2\sigma^2}} \\ \gamma_3 = \sum_{k=0}^{\infty} k^4 e^{-\frac{k^2}{2\sigma^2}} & \gamma_4 = \sum_{k=0}^{\infty} k^6 e^{-\frac{k^2}{2\sigma^2}} \end{array} \tag{35}$$

We can show that a function $F(\sigma) = \sum_{k=1}^{\infty} k^{\alpha} e^{-\frac{k^2}{(2\sigma^2)}}, \alpha > 0$, can be approximated with a very low error by a polynomial function $F_a(\sigma) = I(\alpha)\sigma^{\alpha+1}$, with $I(\alpha) = \int_0^{\infty} t^{\alpha} e^{-\frac{t^2}{2}} dt$.

We know that $I(0) = \sqrt{\frac{\pi}{2}}$, $I(1) = 1$, and $I(\alpha + 2) = (\alpha + 1)I(\alpha)$.

If $\alpha = 0$, we simply approximate $F(\sigma)$ with $F_a(\sigma) = 1.25\sigma - 0.5$ for $\sigma \geq 0.5$.

Therefore, we obtain the following approximations :

$$\begin{aligned} \gamma_1 &\approx 0.5 + 1.25\sigma & \gamma_2 &\approx \sigma^3\sqrt{\tfrac{\pi}{2}} \\ \gamma_3 &\approx 3\sigma^5\sqrt{\tfrac{\pi}{2}} & \gamma_4 &\approx 15\sigma^7\sqrt{\tfrac{\pi}{2}} \end{aligned} \tag{36}$$

Finally, the normalization coefficients are equal to :

$$\begin{aligned} \beta_0 &= \frac{1}{2\gamma_1 - 1} \approx \frac{0.4}{\sigma} & \beta_1 &= -\frac{1}{2\gamma_2} \approx -\frac{0.4}{\sigma^3} \\ \beta_2 &= -\frac{2\gamma_2}{-2\gamma_2^2 + 2\gamma_3\gamma_1 - \gamma_3} \approx -\frac{0.4}{\sigma^3} & \beta_3 &= \frac{2\gamma_1 - 1}{2\gamma_2} \approx \frac{1}{\sigma^2} \\ \beta_4 &= \frac{3}{\frac{\gamma_4\gamma_2}{\gamma_3\sigma^2} - \frac{\gamma_3}{\sigma^2}} \approx \frac{1.2}{\sigma^3} & \beta_5 &= \frac{3\gamma_2}{\gamma_3(\frac{\gamma_4\gamma_2}{\gamma_3\sigma^2} - \frac{\gamma_3}{\sigma^2})} \approx \frac{0.4}{\sigma^5} \end{aligned} \tag{37}$$

In this way, g_0 and its derivatives g_1, g_2, g_3 can be approximated by a 4th recursive operator $h(x)$:

$$h(x) = (a_0 \cos(\frac{\omega_0}{\sigma}x) + a_1 \sin(\frac{\omega_0}{\sigma}x))e^{-\frac{b_0}{\sigma}x} + (c_0 \cos(\frac{\omega_1}{\sigma}x) + c_1 \sin(\frac{\omega_1}{\sigma}x))e^{-\frac{b_1}{\sigma}x} \tag{38}$$

where the coefficients a_0, a_1, b_0, b_1, c_0, c_1, w_0, w_1 are derived from β_i, α_i and λ_i. We have the following results for these coefficients :

	$g_0(x)$	$g_1(x)$	$g_2(x)$	$g_3(x)$
a_0	$0.657/\sigma$	$-0.173/\sigma^2$	$-0.724/\sigma^3$	$1.286/\sigma^4$
a_1	$1.979/\sigma$	$-2.004/\sigma^2$	$1.689/\sigma^3$	$-0.290/\sigma^4$
c_0	$-0.258/\sigma$	$0.173/\sigma^2$	$0.32¡5/\sigma^3$	$-1.286/\sigma^4$
c_1	$-0.239/\sigma$	$0.444/\sigma^2$	$-0.721/\sigma^3$	$0.262/\sigma^4$
b_0	1.906	1.561	1.295	1.012
b_1	1.881	1.594	1.427	1.273
ω_0	0.651	0.700	0.779	0.947
ω_1	2.053	2.145	2.234	2.338

A.2 Implementation of the Gaussian filter and its derivatives

Once the coefficients above are computed, we can easily implement the function $h(n)$ defined by :

$$h(n) = (a_0 \cos(\frac{\omega_0}{\sigma}n) + a_1 \sin(\frac{\omega_0}{\sigma}n))e^{-\frac{b_0}{\sigma}n} + (c_0 \cos(\frac{\omega_1}{\sigma}n) + c_1 \sin(\frac{\omega_1}{\sigma}n))e^{-\frac{b_1}{\sigma}n} \qquad (39)$$

The z-transform $H_+(z^{-1}) = \sum_{n=0}^{\infty} h(n)z^{-n}$ for the causal part of $h(n)$ is given by :

$$H_+(z^{-1}) = \frac{n_0^+ + n_1^+ z^{-1} + n_2^+ z^{-2} + n_3^+ z^{-3}}{1 + d_1^+ z^{-1} + d_2^+ z^{-2} + d_3^+ z^{-3} + d_4^+ z^{-4}}$$

where the coefficients of the numerator and the denominator n_i and d_i can be written as :

$n_0^+ = a_0 + c_0$

$n_1^+ = (a_1 \sin(\frac{\omega_0}{\sigma}) - (2c_0 + a_0)\cos(\frac{\omega_0}{\sigma}))e^{-\frac{b_0}{\sigma}} + (c_1 \sin(\frac{\omega_1}{\sigma}) - (c_0 + 2a_0)\cos(\frac{\omega_1}{\sigma}))e^{-\frac{b_1}{\sigma}}$

$n_2^+ = 2e^{-\frac{b_0+b_1}{\sigma}}((a_0 + c_0)\cos(\frac{\omega_0}{\sigma})\cos(\frac{\omega_1}{\sigma}) - a_1 \cos(\frac{\omega_1}{\sigma})\sin(\frac{\omega_0}{\sigma}) - c_1 \cos(\frac{\omega_0}{\sigma})\sin(\frac{\omega_1}{\sigma})) + c_0 e^{-\frac{2b_0}{\sigma}} + a_0 e^{-\frac{2b_1}{\sigma}}$

$n_3^+ = (a_1 \sin(\frac{\omega_0}{\sigma}) - a_0 \cos(\frac{\omega_0}{\sigma}))e^{-\frac{b_0+2b_1}{\sigma}} + (c_1 \sin(\frac{\omega_1}{\sigma}) - c_0 \cos(\frac{\omega_1}{\sigma}))e^{-\frac{b_1+2b_0}{\sigma}}$

$d_1^+ = -2e^{-\frac{b_1}{\sigma}}\cos(\frac{\omega_1}{\sigma}) - 2e^{-\frac{b_0}{\sigma}}\cos(\frac{\omega_0}{\sigma})$

$d_2^+ = 4\cos(\frac{\omega_1}{\sigma})\cos(\frac{\omega_0}{\sigma})e^{-\frac{b_0+b_1}{\sigma}} + e^{-\frac{2b_0}{\sigma}} + e^{-\frac{2b_1}{\sigma}}$

$d_3^+ = -2e^{-\frac{b_0+2b_1}{\sigma}}\cos(\frac{\omega_0}{\sigma}) - 2e^{-\frac{b_1+2b_0}{\sigma}}\cos(\frac{\omega_1}{\sigma})$

$d_4^+ = e^{-\frac{2b_0+2b_1}{\sigma}}$

For its anticausal part $H_-(z) = \sum_{n=-\infty}^{-1} h(n)z^{-n}$, the z-transform is :

$$H_-(z) = \frac{n_1^- z^1 + n_2^- z^2 + n_3^- z^3 + n_4^- z^4}{1 + d_1^- z^1 + d_2^- z^2 + d_3^- z^3 + d_4^- z^4}$$

We use the following relations for smoothing and second derivative filters (symmetrical filters) to compute the coefficients n_i^- and d_i^- :

$$\begin{aligned} d_i^- &= d_i^+ && i = 1,..,4 \\ n_i^- &= n_i^+ - d_i^+ n_0^+ && i = 1,..,3 \\ n_i^- &= -d_i^+ n_0^+ && i = 4 \end{aligned}$$

For the first and third derivative filters (asymmetrical filters), we have the following relations between the coefficients :

$$\begin{aligned} d_i^- &= d_i^+ && i = 1,..,4 \\ n_i^- &= -(n_i^+ - d_i^+ n_0^+) && i = 1,..,3 \\ n_i^- &= d_i^+ n_0^+ && i = 4 \end{aligned}$$

The z-transforms $H_+(z^{-1})$, $H_-(z)$ correspond to two rational system transfer functions of stable 4th order filters and is written in the time-domain by the following difference equations :

$$y_k^+ = n_0^+ x_k + n_1^+ x_{k-1} + n_2^+ x_{k-2} + n_3^+ x_{k-3} - d_1^+ y_{k-1}^+ - d_2^+ y_{k-2}^+ - d_3^+ y_{k-3}^+ - d_4^+ y_{k-4}^+ \qquad k = 1, ..., N$$

$$y_k^- = n_1^- x_{k+1} + n_2^- x_{k+2} + n_3^- x_{k+3} + n_4^- x_{k+4} - d_1^- y_{k+1}^- - d_2^- y_{k+2}^- - d_3^- y_{k+3}^- - d_4^- y_{k+4}^- \qquad k = N, ..., 1$$

$$y_k = y_k^+ + y_k^- \quad k = 1, ..., N$$

where the x_k and the y_k are respectively the input and output signal of the filter. The details of the implementation can be found in [Der 93].

Unité de recherche Inria Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 Villers Lès Nancy
Unité de recherche Inria Rennes, Irisa, Campus universitaire de Beaulieu, 35042 Rennes Cedex
Unité de recherche Inria Rhône-Alpes, 46 avenue Félix Viallet, 38031 Grenoble Cedex 1
Unité de recherche Inria Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay Cedex
Unité de recherche Inria Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 Sophia-Antipolis Cedex

Éditeur

Inria, Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay Cedex (France)

ISSN 0249-6399