Rapports de Recherche

N°1889

Programme 4

Robotique, Image et Vision

CLASSICAL MECHANICS AND ROADS DETECTION IN SPOT IMAGES

Josiane ZERUBIA
Nicolas MERLET

Avril 1993

UNITÉ DE RECHERCHE
INRIA-SOPHIA ANTIPOLIS

Institut National
de Recherche
en Informatique
et en Automatique

2004 route des Lucioles
B.P. 93
06902 Sophia-Antipolis
France

CLASSICAL MECHANICS AND ROADS DETECTION IN SPOT IMAGES

Josiane Zerubia*

and

Nicolas Merlet**

*INRIA
BP 93
2004 Route des Lucioles
06902 Sophia Antipolis Cedex, France
E-mail : zerubia@sophia.inria.fr

**Institute of Computer Science
The Hebrew University of Jerusalem
91904 Jerusalem, Israel
E-mail : nicolas@cs.huji.ac.il

Avril 1993

Abstract

The detection of roads in satellite images has drawn a lot of attention in the last ten years. Problems of resolution, noise, and image understanding are involved, and one of the best method developed so far is the F* algorithm of Fischler, which is based on dynamic programming.

We present herein a mathematical formalization of the F*, an extension to cliques of higher order to deal with the contrast, an extension to neighborhoods of higher order to take into account the curvature, and a physical dynamic model to use the curvature information in a more natural and global way.

Keywords

edge detection, energy minimization, dynamic programming, curvature, satellite images

Résumé

La reconnaissance de routes sur des images satellite a soulevé beaucoup d'interêt au cours des dix dernières années. Des problèmes de résolution, de bruit, et de compréhension de l'image sont impliqués et l'une des meilleures méthodes développées jusqu'à présent est l'algorithme F* de Fischler, qui utilise la programmation dynamique.

Nous présentons ici une formalisation mathématique du F*, une extension à des cliques d'ordre supérieur pour tenir compte du contraste, une extension à des voisinages d'ordre supérieur pour introduire la courbure, et un modèle physique dynamique pour utiliser l'information de courbure d'une manière plus globale et naturelle.

Mots-clés

détection de contours, minimisation d'énergie, programmation dynamique, courbure, images satellite.

Acknowledgements

The authors would like to thank M. BERTHOD, G. GIRAUDON, S. HOUZELLE and S. MATHIEU for many interesting discussions.

This work was partially supported by an AFIRST grant.

Contents

1 Introduction **2**

2 Distance based energy function **2**

2.1 Reminder 2

2.2 Definitions 3

2.3 Segmentation algorithms 5

2.3.1 Computation of the distance function : Algorithm 1 5

2.3.2 Algorithm of pattern recognition : Algorithm 2 7

3 Curvature information and higher order neighborhoods **9**

3.1 Necessity of adding curvature information 9

3.2 Interest of using higher order neighborhoods 9

3.3 Modifications of the algorithm 11

3.3.1 Computation of the distance function : Algorithm 3 11

3.3.2 Algorithm of pattern recognition : Algorithm 4 12

3.4 Expected performances and drawbacks 13

4 Curvature information and dynamic energy model **14**

4.1 Some fantasy about dynamic programming 14

4.2 Theory of the dynamic energy model 14

4.2.1 Altitude 14

4.2.2 Speed 14

4.2.3 Forces 15

4.2.4 Laws of propagation 15

4.2.5 Choices 16

4.3 Different implementations - results 16

4.3.1 In a discrete frame 16

4.3.2 In a continuous frame 16

4.3.3 Non-stability of the parameters 17

4.3.4 Lost of symmetry 17

4.4 Expected performances 17

4.4.1 Importance of the speed 17
4.4.2 Initial speed 17
4.4.3 Why a braking force 17
4.4.4 Choice of the parameters 18
4.4.5 Influence of this new method on the definition of the potential . . . 18
4.4.6 Complexity 18

5 Applications 18

5.1 Original images 18
5.1.1 Geological images 18
5.1.2 Images with roads 19
5.2 Using the basic model with cliques of order two - g1 19
5.3 Using the basic model with cliques of order three 20
5.3.1 g2 20
5.3.2 r1 20
5.4 Using higher order neighborhoods 20
5.4.1 g2 20
5.4.2 g3 21
5.4.3 r1 21
5.4.4 r2 21
5.5 Results to be improved 22
5.5.1 r3 22
5.5.2 r4 22

6 Discussion 22

6.1 Determination of the values of the parameters 22
6.1.1 What has been done 22
6.1.2 Directions of research 23
6.2 Comparison with other methods 23
6.2.1 Edge detection 23
6.2.2 Comparison with other distances 24
6.2.3 Graph theory and dynamic programming 24

6.3 General framework, drawbacks 26
6.3.1 General framework 26
6.3.2 Drawbacks 27
6.4 Future research 27

7 Conclusion 28

8 Appendix : proofs of the theorems and algorithms 28
8.1 Theorem 1 : 28
8.1.1 Symmetry 28
8.1.2 Separability 28
8.1.3 Triangular Inequality 29
8.2 Theorem 2 : 29
8.2.1 $H(2)$ 29
8.2.2 $H(n+1)$ 30
8.2.3 Set of minima of R 30
8.3 Algorithm 1 30
8.3.1 Convergence 30
8.3.2 Rightness 31
8.4 Algorithm 2 31
8.4.1 Convergence 31
8.4.2 Rightness 31
8.5 Definition 3 32
8.6 Algorithm 3 32
8.6.1 Convergence 32
8.6.2 Rightness 33
8.7 Algorithm 4 34
8.7.1 Convergence 34
8.7.2 Rightness 35

9 References 35

1 Introduction

The detection of roads in satellite images is a difficult topic for which several methods have been tested and sometimes specially developed. Problems of resolution, noise, blurring, and image understanding are involved essentially.

[Des 86] uses Mathematical Morphology and top-hat transforms in particular for detecting roads. [Graf 89] performs Deriche edge detection followed by probabilistic edge prolongation. [Ser 89] detects agglomerations with Mathematical Morphology and roads with dynamic programming, but her approach faces problems of convergence and rightness, and deals with them with thresholds of failure. [Dao 89] also uses dynamic programming ([Mar 76] in particular) and deals also with convergence problems by defining several thresholds. [Gar 89] solves ambiguities with an expert system, and detects roads by prolongating segments. [Jed 91]'s approach is based on dynamic programming in relation with width, connectivity and maximal curvature.

We extend in this work, done partially at the INRIA-Sophia Antipolis, the algorithm of F* [Fis 81], based on dynamic programming and commonly considered as giving the best results in road detection.

In this report, we will first present a mathematical formalization of the algorithm and an extension to cliques of order higher than two, to take into account the contrast between roads and background.

Then, we will consider two ways of taking curvature information into account : by using bigger neighborhoods and aligned points first, and then by developing a dynamic physical model, based on the use of forces, acceleration and energy.

Presentation of the results, discussion and conclusion will follow.

2 Distance based energy function

2.1 Reminder

Let us first recall some basic notions of the theory of Markov Random Fields which will be useful throughout the paper [Bes 74][Cha 88][Aze 87].
We denote by S the lattice of the original image.

<u>Definition 1 :</u> A *neighborhood system* on S is a family $\mathcal{N} = \{\mathcal{N}_x \subset S | x \in S\}$ such that :

- $x \notin \mathcal{N}_x$,
- $\forall x_1 \in S, \forall x_2 \in S, x_1 \in \mathcal{N}_{x_2} \Longleftrightarrow x_2 \in \mathcal{N}_{x_1}$.

The elements of $\mathcal{N}_x$ are the *neighbors* of x.

Definition 2 : $c \subset S$ is a *clique* relatively to $\mathcal{N}$ iff :

- $|c| = 1$,
- or $|c| \geq 2$ and any two different elements of c are neighbors.

$|c|$ (number of elements of c) is called the *order* of the clique.

In the hexagonal lattice for instance,

- there are six cliques of order two corresponding to the six directions of the lattice,
- there are six cliques of order three corresponding to the six elementary triangles of the lattice.

In the square lattice, the number of cliques of order two, three and four depends on the type of connectivity chosen.

In the following, we will consider both lattices, depending on the application.

We will denote by :

- $\mathcal{C}_n$ the set of the cliques of order n and of the cliques of order one,
- $P(x, y)$ the set of paths from x to y,
- $\mathcal{F}(S, \mathcal{R}^+)$ the set of functions from S into $\mathcal{R}^+$,
- f an element of $\mathcal{F}(S, \mathcal{R}^+)$, typically a grey-level image, and f_c its restriction to c.

We now have the tools for defining a potential and an energy on these cliques.

2.2 Definitions

Our final aim is to define a distance such that an edge is the shortest path between its extremities for this distance. We proceed in two steps : we first define a cost associated to a path between two points, and then the distance as the minimum cost for all the paths joining them. We naturally think to define the cost as the sum of the potentials along the path.

<u>Definition 1 :</u>
Given a potential $\phi : \mathcal{C}_n \text{x} \mathcal{R}^{+^n} \rightarrow \mathcal{R}^+$ (in practice $n = 2$ or $n = 3$), and given a path h : $x_0 = x$,...,x_i,..., $x_p = y$ between two points x and y of S, the *attraction cost along h* is defined as the function μ :
$\mu(h) = \min_{c_i \in \mathcal{C}_n | x_i, x_{i+1} \in c_i} \sum_{i=0}^{p-1} \phi(c_i, f_{c_i})$

We notice that, in the case of cliques of order two, the minimization is not necessary since a unique clique is defined by two successive points of the path, and the cost is simply the sum of the potentials along the path : $\mu(h) = \sum_{i=0}^{p-1} \phi(c_i, f_{c_i})$, where c_i is the clique composed of x_i and x_{i+1}.
However, for higher orders, two points of the path do not necessarily define a unique clique : for cliques of order three for example, two points define two cliques (or four) on both sides of the path, and the minimization takes into account only the one with the lower potential.

The definition of the energy is straightforward :

<u>Definition 2 :</u>
The *attraction energy U* between two points x and y of S is defined as the minimum value of the attraction cost along any path between x and y :
$U(x, y) = \min_{h \in P(x,y)} \mu(h)$.

Under a very simple constraint on the potential, this function is also a distance.

<u>Theorem 1 :</u>
We suppose that ϕ satisfies the constraint :

$\forall c \in \mathcal{C}_n, \forall f \in \mathcal{F}(S, \mathcal{R}^+),\ \phi(c, f_c) = 0 \Longleftrightarrow |c| = 1.$

Then $\Delta(x, y) = U(x, y)$ is a distance function on S, called *directional numerical distance*.

The separability condition follows from the constraint, and the triangular inequality may be shown by combining two paths. The complete proofs of the theorems and algorithms may be found in the Appendix.

Remarks :

- The constraint on ϕ defined in Theorem 1 is necessary.
- In general, Δ is not regular (there may be gaps in the values of Δ according to the values of ϕ).
- Note that elementary local properties lead us to a high-level global notion.

We have thus defined a distance function from a local potential. One could think, however, that the computation may be expensive, if we have to compute the attraction cost for all the paths joining two points. A second theorem will show us that this is not the case : it will give us both a fast algorithm for computing Δ and a new method of segmentation. It claims that if a function R satisfies some simple property, than it is equal to the numerical distance. Algorithm 1 will build this function R.

Theorem 2 :
Let us consider a function $R : S \longrightarrow \mathcal{R}^+$ such that :

- $\exists x \in S, R(x) = 0$. We denote by S_0 the kernel of R : $S_0 = \{x \in S, R(x) = 0\}$.
- $\forall x \in S, R(x) \neq 0 \Rightarrow R(x) = \min_{c \in \mathcal{C}_n, x \in c, x_i \in c}\{R(x_i) + \phi(c, f_c)\}$

Then we have two important results : 1- $\forall x \in S, R(x) = \Delta(x, S_0)$,
2- S_0 is the set of the minima of R.

The first point may be proved by induction. The second point follows from the equation satisfied by R (see Appendix).

2.3 Segmentation algorithms

We now face two problems :

1. how to compute Δ
2. how to find the shortest paths for Δ ; there may be several shortest paths if for example we consider at the same time the extremities of different roads in order to perform all the computation in parallel

Theorem 2 will give the solution to both of these problems with two new algorithms. It may be shown first that if we iterate the minimization equation of Theorem 2, we obtain at convergence the distance Δ (Algorithm 1).

Second, starting from one point we can reconstruct backwards the succession of points through which the distance to the kernel was computed, by comparing the difference between the distance values and the values of the potential (Algorithm 2).

2.3.1 Computation of the distance function : Algorithm 1

We consider two images :

- Im1 will be the output image. At the beginning, it is equal to zero for points belonging to S_0, and to $+\infty$ for the other points.

- Im2 is a series of images which correspond to the potentials for the different possible cliques. In the case of cliques of order two, an image of Im2 is the potential of one clique ; for cliques of higher order, an image is the minimum of the potentials of the cliques containing the point and one of its (8) neighbors (in 8-connectivity and with cliques of order three, Im2 is composed of 8 such images). Accordingly, Im2 is computed before the iterations in one or two scannings for each clique.

The general idea is to perform until convergence (alternatively forward and back) :

$$Im1(x) \longleftarrow \min_{x_i \in \mathcal{N}_x} \{Im1(x), Im1(x_i) + \phi(c_i, f_{c_i})\}$$

More precisely, the algorithm may be written in pseudo-C as :

```
change=1;
while(change>0)
  {
   change=0;
   for(row=0;row<size_y;row++)
     {
      for(col=0;col<size_x;col++)
        {
         tmp=min(Im1(col-1,row)+Im2(col,row,-1,0),
                 Im1(col-1,row-1)+Im2(col,row,-1,-1));
         update_Im1(col,row);
        }
      for(col=size_x-1;col>=0;col--)
        {
         tmp=min(Im1(col,row-1)+Im2(col,row,0,-1),
                 Im1(col+1,row-1)+Im2(col,row,1,-1));
         update_Im1(col,row);
        }
     }
   for(row=size_y-1;row>=0;row--)
     ....
  }
```

with :

```
update_Im1(col,row)
int col,row;
{
if(tmp<Im1(col,row))
```

```
  {
   Im1(col,row)=tmp;
   change=1;
  }
```

At convergence, Im1 is equal to R, equivalently to the distance to the kernel. Proofs of convergence and correctness follow directly from Theorem 2 (see Appendix). We will study the complexity in Section 6, but one may already notice the systematic use of the four different scanning directions.

Thus, at each point, R has been computed by a succession of minimizations along a path joining it to S_0. The following algorithm reconstructs this path : this is the shortest path to S_0.

2.3.2 Algorithm of pattern recognition : Algorithm 2

We now want to use Δ for recognizing features in grey-level images. We suppose that a potential ϕ has been chosen that characterizes locally (see Section 5 : Applications) a given type of feature. More precisely, a feature passing through two given points (or sets of points) corresponds exactly to the shortest path between these sets *for the distance Δ associated to ϕ.*

We already know that $R(x) = \Delta(x, S_0)$, so S_0 must be one of these two sets of points. We call S_1 the second set. One should notice that S_0, S_1, and ϕ are all defined relatively to a particular application.

The following algorithm reconstructs the shortest paths from S_1 to S_0.

We consider the following images :

- Im1 : grey-level image R (obtained from Algorithm 1).
- Im2 : binary image defined as :

$$Im2(x) \begin{cases} > 0 & \text{iff } x \in S_1 \\ = 0 & \text{else.} \end{cases}$$

We propagate the labels until convergence :

```
change=1;
while(change>0)
  {
   change=0;
```

```
  for(row=0;row<size_y;row++)
    {
     for(col=0;col<size_x;col++)
       {
        if((Im2(col,row)!=0)
         &&(Im2(col+1,row)==0)
         &&(Im1(col,row)=Im1(col+1,row)+Im2(col,row,1,0)))
          {
            Im2(col+1,row)=Im2(col,row);
            change=1;
          }
        if((Im2(col,row)!=0)
         &&(Im2(col+1,row+1)==0)
         &&(Im1(col,row)=Im1(col+1,row+1)+Im2(col,row,1,1)))
          {
            Im2(col+1,row+1)=Im2(col,row);
            change=1;
          }
       }
     for(col=size_x-1;col>=0;col--)
       ...
    }
  for(row=size_y-1;row>=0;row--)
    ....
}
```

Due to the property satisfied by R, $Im2$ contains, at convergence, the shortest paths between S_0 and S_1 (see Appendix). We also use the four possible directions of scanning to get a fast propagation. For speeding the algorithm, one could think to use two methods :

- to define arrows during the first algorithm to remember which pixel caused another pixel to take its final value, and to simply follow the arrows in the second algorithm
- to propagate from S_1 systematically towards the smallest neighbor and to store the propagation in a tree

However, this would limit us since we would obtain (except with complex constructions) only one shortest path for each pixel of S_1, which is not satisfactory, neither theoretically nor practically.

Remark : One should notice the possibility to obtain different features at the same time, if for example S_1 has several connected components. In this case, Im2 is defined as a labeling of S_1. The result is a labeling of the corresponding features, according to the connected components which they originate from.

In summary, this method of segmentation consists of :

- defining the potential ϕ, the sets S_0 and S_1,
- computing the distance Δ to S_0 with Algorithm 1,
- building the shortest paths from S_1 to S_0 with Algorithm 2.

We have described the last two steps, but before presenting the applications and the definition of the potential, we want to present theoretically several extensions to deal with contrast and curvature.

3 Curvature information and higher order neighborhoods

3.1 Necessity of adding curvature information

We will see later that local information is not sufficient to characterize roads, in two common configurations in particular :

- when two parallel roads get close to each other. A short interruption in one of the roads may force the propagation to jump from one road to the other, especially if the differences with the background are not much pronounced,
- when the road crosses an agglomeration or a highly textured region. The human eye then uses the entry and the exit of the road into the agglomeration for recognizing the road. However, with local information the correspondence may be missed.

3.2 Interest of using higher order neighborhoods

We want first to keep the general structure of the method and to simply extend it such that it takes into account very local curvature. We may do it in the following way :

- We consider neighborhoods of size *two* or more (and not one as until now).
- We consider *three* or more successive points of the path (and not two).
- We consider now not only one clique, but two cliques : one containing good candidates for road's points, the second good candidates for background points. This reminds us the hit-and-miss transform of Mathematical Morphology (Golay transform [Gol 69][Mat 75][Ser 82]) which looks for configurations where one part of the

structuring element belongs to the object, and the other part to the background. Contrary to Mathematical Morphology, we do not consider binary logic where the relevant part of the configuration is/is not inside the object, but we associate to the corresponding points of the image a local cost which characterizes the probability for them to fit well the configuration.

- The potential depends on the angle between the points of the clique characterizing the road. This last point is what allows us to take local curvature into account.

We have presented in Fig. 1 below such a configuration, where the chosen neighborhood is of size four and where four successive points of the path are taken into account. The clique characterizing the road (of order four) is represented by round points, and the clique corresponding to the background (of order sixteen) by square points. In this example one couple of cliques fits a straight line, while the other corresponds to a curve.

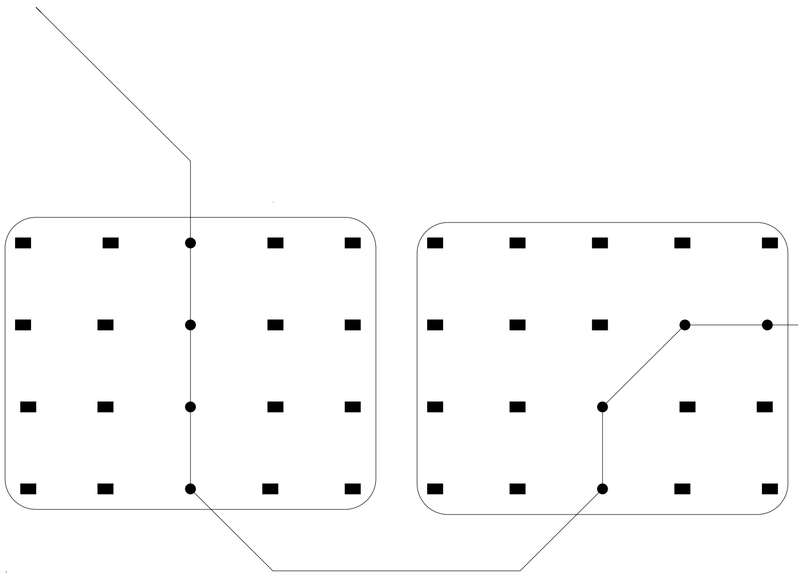

Figure 1: Two couples of cliques for a neighborhood of size four

This yields to substitute the following definition to *Definition 1* :

Definition 3 :
Given a potential $\phi : \mathcal{C}_n \mathrm{x} \mathcal{C}_m \mathrm{x} {\mathcal{R}^+}^{n+m} \rightarrow \mathcal{R}^+$, and given a path h : $x_0 = x, \ldots, x_i, \ldots, x_p = y$ between two points x and y of S, the attraction cost along h is defined as the function μ :
$\mu(h) = \min_{c_i \in \mathcal{C}_n, c_j \in \mathcal{C}_m | c_i \cup c_j \in \mathcal{C}_{n+m}, x_i \in c_i, \ldots, x_{i+n-1} \in c_i} \sum_{i=0}^{p-n+1} \phi(c_i, c_j, f_{c_i}, f_{c_j})$

For example we may consider nine points forming a square : three adjacent points out of the nine correspond to the road, the six others to the background. The three points define an angle, and the potential of the clique may depend on a partial cost of this angle ; a penalty may be added to the old potential depending on the angle : if the points are

aligned, the penalty is null ; if the angle of alignment is small, the penalty is small, and the penalty can be infinite for important deviations.
We illustrate in Fig. 2 below this choice. c_i, of order three, corresponds to the road while c_j, of order six, characterizes the background. α is the angle of deviation to the straight line, and the potential should increase with it.

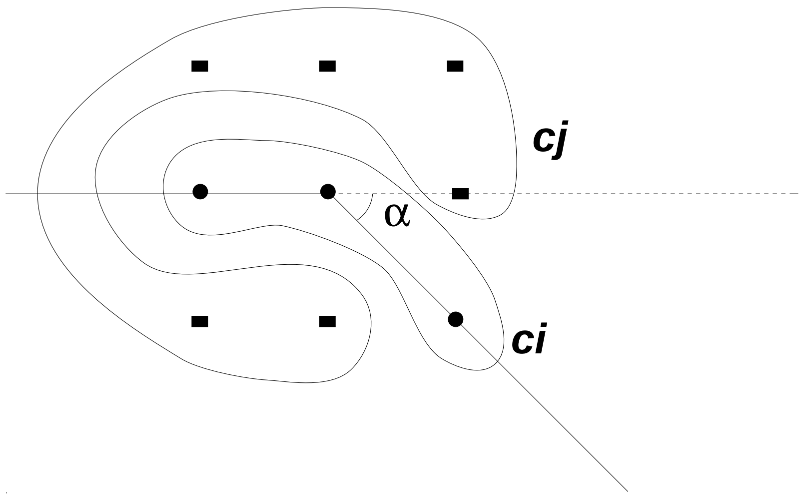

Figure 2: Couple of cliques for a neighborhood of size two

It can be noticed that with this new definition U is not a distance since the Triangular Inequality is not satisfied any longer. Since there is no real meaning to define a clique of order higher than p with a path of length p, we restrict μ to paths of length bigger than the order of the clique, n.

3.3 Modifications of the algorithm

Both algorithms are deeply modified due to this new constraint, since we must take into account neigborhoods of order two. However, we keep the central idea of dynamic programming to perform iteratively local minimization.

In all our applications, we considered only three successive pixels, and maximal deviations of 45 degrees. Thus there were only 24 cliques to be considered : each point has eight direct neighbors, and only three second order neigbors could come afterwards without forming together an angle of less than +/- 135 degrees.

3.3.1 Computation of the distance function : Algorithm 3

Now, in order to take into account the dependencies between successive cliques on the path, we must remember for each point the minimum energy which was assigned to it by each of its (8) neighbors. We must therefore define two types of images :

- Im1 is a series of 8 output images corresponding to the 8 neighbors of a point. At the beginning, they are equal to zero for starting points (assumed known) and for their direct neighbors, and to $+\infty$ for other points. They store for each point the minimum energy which was assigned to it by the corresponding neighbor. (For each neighbor, three cliques may intervene corresponding to the three second order neighbors).
- Im2 is a series of 24 images, representing the potentials associated to the 24 types of cliques. For practical reasons, we associate to each point M_0 the clique of order three formed of M_0, of one of its 8 neighbors N_0, and of one of the three neighbors of N_0 prolongating the two points.

We then perform the following operations, for each point of the image M_0 and for each of its 8 direct neighbors N_0 :

- We consider each of the three cliques c_0, c_1, and c_2 which contain M_0 and N_0, each corresponding to a direct neighbor P_0, P_1 and P_2 of N_0 (second order neighbor of M_0).
- We consider the three values of Im1 in N_0 for the three images corresponding to P_0, P_1 and P_2. We call them $Im1_{P_i}(N_0)$.
- We compute the three values $U_{P_i}(M_0) = Im1_{P_i}(N_0) + Im2_{c_i}(M_0)$.
- We compute their minimum $U_1(M_0) = \min_{i=1,2,3} U_{P_i}(M_0)$.
- If $U_1(M_0)$ is smaller than $Im1_{N_0}(M_0)$, then we update $Im1_{N_0}(M_0)$ with this new smaller value.
- We stop the algorithm when nothing is updated in a complete cycle.

In fact, we also perform here scannings in the four possible directions, and for each scanning we consider only the neighbors which have been updated in the same scanning, in order to optimize the propagation. We obtain the result in Im1 in a coded form. The convergence of the algorithm is assured since Im1 decreases until stationarity. Convergence and rightness are proved in details in the Appendix. We will see now how to exploit in a very easy way the information in $Im1$ for reconstructing the paths.

3.3.2 Algorithm of pattern recognition : Algorithm 4

The new algorithm of back-propagation is entirely different from the previous one for neighborhoods of size one, since it is based on the directional information of Im1 (See the algorithm described in Section 3.3.1) : we follow the direction shown by the successive cliques. We use two types of images :

- Im1 is the series of 8 output images (Im1) obtained by the algorithm described in Section 3.3.1, where the value at a point is the minimum value of the energy from paths coming from the corresponding 8 neighbors.
- Im2 is the binary image of the goal points (assumed to be known), eventually labeled.

We then perform the following operations : for each point M_0 outside of S_0 such that $Im2(M_0)$ is not null (M_0 is labeled),

- We look for the minimum value of the energy $Im1_{N_i}(M_0)$ in the eight images $Im1$.
- We mark in Im2 the point N_i corresponding to this minimum with the same label as M_0, if N_i is not already labeled.

We stop the process when no labeling is performed in a complete cycle.

The algorithm converges since we deal only with points which are not labeled and since we label them in an irreversible way. At convergence, Im2 contains the shortest paths (see Appendix) ; they are labeled if Im2 has been labeled at initialization. We may implement the algorithm by performing scannings in the four directions, or by using a tree structure. In both cases, we avoid bottlenecks with the diagonals and optimize the propagation.

3.4 Expected performances and drawbacks

We expect to reduce the number of sinuosities of the segmentation, eventually to fix an upper bound, and to use the direction followed so far to guess the next privileged direction.

However, this information remains very local :

- the only information we may dispose of is the direction between the last pixel and the current one.
- only local curvature is considered with this method. Thus a path oscillating around a fixed direction, with smooth and regular zig-zags, would produce a higher cost than a path having one sudden change of direction between two straight segments.

One may envision to generalize this method to n successive pixels, but the number of cliques to be considered becomes huge, programming becomes very difficult, and the algorithms become much less competitive. We will see now how to introduce global curvature by placing the principle of the method in the more general context of classical mechanic.

4 Curvature information and dynamic energy model

4.1 Some fantasy about dynamic programming

So far, we have defined an energy by integrating a local potential. This energy and its use remind us the classical potential energy for several reasons :

- it is defined as a function of the localization only
- the local minima of this energy (S_0) play an important role
- when we propagate in the second algorithm, we tend to move down towards these minima.

In some sense, propagation is performed "with zero speed" since we move always towards the lowest neighbor, without taking into account the direction followed so far.

However, this past direction is important for crossing agglomerations and keeping the right direction in highly textured regions. Besides, we hope to reduce the curvature in this way. We want now to use this information by introducing a kinetic energy related to a speed vector.

4.2 Theory of the dynamic energy model

4.2.1 Altitude

We have defined a potential energy relatively to S_0. The classical potential energy related to the weight may be written under some simplifying assumptions : $E_p = m * g * z$ were m is the mass of the moving body, g the gravitation, and z the local altitude. m and g will be two parameters defined by the user ; we will see later how they influence the model. Thus, we obtain a new image of the altitude which is proportional to the potential energy.

4.2.2 Speed

The propagation is performed with zero speed with the model developed so far. Now, we take into account the speed during the propagation. At each step of the propagation we associate a speed vector, which - together with the known potential energy - determines the next position. The direction of this vector corresponds to the direction followed so far, and its norm characterizes the validity of this information. For computing the next position, we need to know the evolution of the speed, thus the acceleration. We must therefore list all the forces applied to the object, in order to apply the fundamental law of dynamic. These forces are the contribution of the current local configuration to the speed vector.

4.2.3 Forces

By prolongating the parallel with the physical model of an object rolling down a surface, we may define three forces :

- the weight : it is defined as $\vec{P} = m * \vec{g}$ and allows the virtual object to propagate towards regions of lower potential energy.
- the reaction of the surface, $\vec{R}$: to simplify the computation, we consider that it is orthogonal to the surface and distinct from the braking force. It allows to take into account the local slope in the propagation.
- the braking force $\vec{B}$: for reasons which we will develop later, we define it as not null and inversely proportional to the speed, $\vec{B} = -k * \vec{v}$. This coefficient k is the third (after m and g) and last parameter of the dynamic model. This force aims at regulating the propagation.

4.2.4 Laws of propagation

We consider moreover some simplifying hypotheses :

- the surface is continuous
- the moving object is a point
- the object moves without collision (this is a severe and important restriction)

The method is performed according to the following steps :

1. we build the potential energy as before (Algorithm 1), relatively to S_0
2. only the propagation is changed. We start from S_1, possibly with zero speed, and propagate until we reach S_0. At each point, we use both the fundamental equation of dynamic : $\sum \vec{f} = m * \vec{\gamma}$, where $\vec{f}$ and $\vec{\gamma}$ are respectively the forces applied to the object and the acceleration, and the conservation of the energy $E_c = \frac{1}{2} * m * v^2$, $E_p = m * g * z$ and $E_p + E_c = constant$ (four equations) to obtain the value of the four unknown variables : the norm of $\vec{R}$, and the three coordinates of the speed vector (which we are looking for). More precisely, at each point :
   (a) we call $\vec{v_0}$ the speed vector with which the object arrived to this point (known)
   (b) we apply the fundamental equation of dynamic, $\sum \vec{f} = m * \vec{\gamma}$:
   $m * \vec{g} + \vec{R} - k * \vec{v} = m * \vec{\gamma}$.
   (c) since we do not know the norm or the reaction, we project the equation to the tangent plane to the surface : $m * \vec{g_p} - k * \vec{v_p} = m * \vec{\gamma_p}$. The user defines m, $\vec{g}$ and k at initialization, $\vec{v}$ is the speed at the last step, the tangent plane is known from the altitude, so we obtain $\vec{\gamma_p}$ and the speed in the tangent plane.

(d) to obtain the speed in the normal direction to the surface, we apply the conservation of the energy between the current point and the last one

(e) from the new speed vector, we may compute the location of the next point, and go back to (a).

4.2.5 Choices

There are several important choices to make to implement this model :

- the model is continuous, while the image is discrete. Thus some discretization method must be developed/chosen for determining in particular :
  - if the next point belongs to the frame (integer coordinates) or may be located at any place (floating coordinates)
  - what is the shape of the surface between the discrete points of the frame. If some regular elementary shape is defined between neighbors, how should these elementary shapes be combined (derivability ?).
  - if we work with the differential equation in $\vec{v}$ or assume $\vec{\gamma}$ constant on some distance.
- in order to simplify the model, we have assumed that there were no collisions. What should be done when the speed vector does not belong to the current tangent plane ?

We want yet to present the first tests done with different implementations. We will then discuss what we may expect from the method.

4.3 Different implementations - results

4.3.1 In a discrete frame

We have tried several implementations by imposing that every point of the trajectory belongs to the discrete frame. A major problem was met with infinite oscillations between neighboring points (with a period of 2, 3, or 4 points...) ; at some step the speed had to be approximated to zero and a new cycle started. This is due to the fact that we lost important information in this approximation, and this information would have allowed to escape from the oscillations.

4.3.2 In a continuous frame

We are currently working on the idea, to allow a point of the trajectory not to belong to the frame. We must deal with the important issue of defining the surface consistently between points of the frame such that the propagation converges always towards S_0.

4.3.3 Non-stability of the parameters

In the implementations we have performed, the parameters m,g, and k tend to produce instable behaviors. This may be related to the non-continuity in the implementation.

4.3.4 Lost of symmetry

One difficulty is that we loose the homogeneity of the trajectory. Now, a constant direction at the beginning of the trajectory may influence heavily the following, while more reliable information may be available at the end. This is the price to pay for the idea of using acquired information.

4.4 Expected performances

4.4.1 Importance of the speed

The speed synthesizes the directional information of the past trajectory. If the trajectory has kept the same direction for a certain length, the speed in the corresponding direction increases, and the propagation has a greater tendency to follow this direction : only a higher hill may modify again the direction.

When this information is not important (no clear privileged direction during the last step), the potential energy has more influence for determining the next point.

Besides, we tend not to follow the sinuousities of the depth of the valley, but to follow its global direction.

4.4.2 Initial speed

If no information is available when starting the propagation, the initial speed should be set to zero. However, when some information is known, the propagation may be oriented at the beginning by defining accordingly the initial speed.

4.4.3 Why a braking force

If k (braking coefficient) is null, there is not braking force, and the speed is not limited. Thus, the role of the potential decreases greatly, and huge jumps may be expected at the end of the trajectory, which do not take into account local information.

We have chosen for the braking force an expression common in fluid environments, and for which the speed tends asymptoticly towards a limit (and as the opposite of an exponential). We expect also to solve problems of oscillation in the continuous domain with this expression. A third reason is to limit the memory to a reasonable distance in the past trajectory. The contribution of past accelerations decreases constantly with time, and this corresponds also to the fact that the validity of directional information in roads has a spatial limit.

4.4.4 Choice of the parameters

There are three parameters, m, g and k which all have a clear intuitive meaning :

- k limits the jumps at the end of the trajectory, the oscillation frequency, and the memory of past trajectory. This is a stabilization factor.
- g determines the relative influence of the potential on the kinetic energy. If g has a greater value, the trajectory tends to adapt more closely to the sinuousities of the potential. A small g limits the curvature.
- the product $m * g$ determines the geometric relation between x and y coordinates on one hand, the altitude on the other hand. The greater $m * g$, the smaller the altitude.

4.4.5 Influence of this new method on the definition of the potential

This new method suggests new approaches for the definition of the potential. We now want the surface to be smoother, in particular in the neighborhood of the roads, and this may lead to new constraints for defining automatically the potential.

4.4.6 Complexity

The complexity of the algorithm is proportional to the number of points of the trajectory (a hundred basic operations at each point), so the segmentation remains very fast.

5 Applications

5.1 Original images

We have tested our methods on seven different satellite images, three with geologic structures (valleys) and four with roads and agglomerations.

5.1.1 Geological images

Fig. 3 (g1), Fig. 4 (g2) and Fig. 5 (g3) represent valleys.

- In Fig. 3, the clear vertical line on the left part of the image is a valley, while a road has a similar aspect on the right.

- Due to shadowing, the valleys in Fig. 4 are the transition lines from clear regions (up left) to dark ones (down right) ; the light comes from the right bottom corner (south-east).
- In Fig. 5, the river in the valley also appears as a clear vertical line on the left handside of the image.

5.1.2 Images with roads

The roads in these four images have various aspects :

- In Fig. 6 (r1), the sea may be easily recognized up and left. The roads are clear lines, which sometimes may hardly be distinguished from the highly textured agglomerations or from channels.
- The roads have a similar aspect in Fig. 7 (r2), but in this case they are the most textured regions of the image.
- We have a grid of roads in Fig. 8 (r3) ; the image is poorly contrasted and even the human eye has difficulty to define what is a road in the grid.
- Several problems are combined in Fig. 9 (r4) : roads can hardly be distinguished from borders of the fields, from the agglomerations, and their density varies a lot, both absolutely and relatively to the background, with a contrast which may be positive or negative.

5.2 Using the basic model with cliques of order two - g1

In g1, recognizing the road and the river is easy since they are the only clear lines of the image. No information of contrast or curvature is necessary, and we used cliques of order two. The potential is defined as a piecewise-linear decreasing function of the minimum of the two pixels of the clique :

$$\phi(M_1, M_2) = h(min(f(M_1), f(M_2))).$$

Fig. 10 represents the distance/energy function R obtained for g1, and Fig. 11 the corresponding result, superimposed on the original image. One may notice the good correspondence between the segmentation and the features, this is our final result for g1. Since the two roads (right part of the image) meet at their extremities, only one road could be recognized.

5.3 Using the basic model with cliques of order three

5.3.1 g2

When we consider g2, we notice that the contrast plays an important role for determining the valleys, and we use therefore cliques of order three. The potential is the sum of a grey-level term h and a contrast term g. If we index the three points of the clique according to their increasing values ($f(M_1) \leq f(M_2) \leq f(M_3)$) - M_3 is useful only for this ordering :

$$\phi(M_1, M_2, M_3) = h(f(M_2)) + g(f(M_2) - f(M_1)).$$

$f(M_2)$ is the median of the three values, it is also the minimum value on the valley if the valley crosses the clique. $f(M_1)$ is the value in the near neighborhood, and $f(M_2) - f(M_1)$ characterizes the local contrast between the valley and the background. h and g decrease and are piecewise-linear.

The result of the segmentation is presented in Fig. 12. Due to the proximity of two valleys, the propagation jumped from one to another.

5.3.2 r1

By applying the same type of potential, we obtain for r1 and with two different sets of parameters Fig. 13 and Fig. 14. One may notice on both images that the results are wrong for parts of the roads, in two cases in particular :

- when two parallel roads are close to each other and separated by a highly textured region or by highly contrasted fields
- when a highly contrasted line (e.g. a river) comes close to a road.

In those cases, the propagation jumps from one path to another. We will see now how to deal with the segmentation problems of g2 and r1 by introducing curvature information.

5.4 Using higher order neighborhoods

5.4.1 g2

To take into account the three notions of grey-level, contrast and curvature, we define the potential as the sum of three terms h, g and k, which depend on two cliques c_i and c_j of order three and six as mentioned in 3.2. We call M_1, M_2 and M_3 the three points of c_i, ordered according to the increasing values of f : $f(M_1) \leq f(M_2) \leq f(M_3)$. ϕ is defined as :

$$\phi(c_i, c_j, f_{c_i}, f_{c_j}) = h(f(M_1)) + g(f(M_1) - median(c_j)) + k(c_i).$$

$h(f(M_1))$ characterizes the worst of the three candidates for the valley, $f(M_1) - median(c_j)$ is the contrast between the valley and the background (more precisely the median value of the background), and $k(c_i)$ corresponds to the angle between the points of c_i. h and g are piecewise-linear and decrease, while k increases with the angle (practically, k may take three values to reduce the computation).

The final result is presented in Fig. 15. The notion of curvature does not intervene at the specific point where the propagation turns towards another feature, but on all the sinuosities of the jonction between both features. We notice this time the good fitting between segmentation and features.

5.4.2 g3

With the same method and the same form of potential, we obtain for g3 the final result in Fig. 16, where the segmentation fits well the features.

5.4.3 r1

For r1, we also used the same type of potential.

- Fig. 17 is obtained by using the average in the background instead of the median. Noise influences this average value more than the median value, and the result shows a road from which the propagation went wrong, from the upper right.
- Fig. 18, 19, 20, 21 were obtained with a median value for the points of the background, and all with the same set of parameters but with different choices for S_0. In Fig. 18, S_0 is the left vertical column. In Fig. 19 it is the sea and in Fig. 20 the center of the agglomeration. Finally, S_0 is constituted both from the sea and from the center of the agglomeration in Fig. 21. Obviously, the result may change greatly according to the choice of S_0.
- The best result is presented in Fig. 22. S_0 is the center of the agglomeration, and the angle coefficient has been lowered relatively to the preceding results. It is important to notice that there is no way to distinguish between roads and channels even for the human eye, so it is not surprising to get such a result.

5.4.4 r2

When applying the same type of potential to r2, we obtain Fig. 24 as a final result. The result fits perfectly the features, although the human eye had difficulties to precisely localize the roads. Fig. 23 is the corresponding energy function.

5.5 Results to be improved

5.5.1 r3

Due to the original confusion in r3, the results for r3 are not very interesting. Fig. 25 was obtained with cliques of order two, and Fig. 26 was obtained with second order neighborhoods and cliques of order three.

5.5.2 r4

The same remark applies for Fig. 27, obtained with second order neighborhoods (see for instance the right handsize of the image). We hope to deal with these two images (r3 and r4) with the dynamic model, whose study is currently underway.

We want now to discuss the results : determination of the parameters, comparison of this method with other methods, and future objectives.

6 Discussion

6.1 Determination of the values of the parameters

An important topic is how to define the potential corresponding to a given image.

6.1.1 What has been done

So far, the potential is computed from :

- a priori constraints : functions (h,g) which are piecewise-linear for limiting the search, separability of these terms, definition of the potential as a sum of terms, stationarity, continuity of the functions,
- general characteristics of the image : features presenting a positive or negative contrast, dark or clear grey-levels (which determine if h and g are increasing or decreasing),
- case study : taking small windows of the image and computing values which are characteristic of the image (such as relation between grey-level values and probability to belong to a feature, usual length of interruptions)

6.1.2 Directions of research

Our future aims are to get rid of the case study, in one of the following ways :

- either to find a potential valid for a class of images
- or to find an automatic way of computing the potential from the analysis of small windows.

6.2 Comparison with other methods

We want now to recall the main advantages of the F* method in comparison with other methods.

6.2.1 Edge detection

We want to stress that most methods of edge detection are based on a thresholding ; some are performed on a binary image obtained after a thresholding [Hou 62], others perform a final thresholding after a filter [Can 86][Der 87][Cox 90]. Thus, for the general case :

- there may be problems for fixing this threshold value, especially when there is much noise. Hysteresis thresholding [Can 86] deals partially with this question, but the same problem may appear again (although less roughly) for the lower threshold : how to get all the edge, without getting any noise ?
- the fine grey-level information may be completely neglected.
- these methods are aimed more to recognize all the edges than to detect a particular pattern, so the purpose is slightly different.

Some methods have dealt with the problem of exploiting grey-level information (following the ridges), using local neighborhood-based propagation, but local interruptions may be fatal.

Our method which extends the F* algorithm deals with these two problems of grey-level information and of global information by the use of a global grey-level energy function. In comparison with the classical methods, we can stress the following points (which characterize also the F* algorithm) :

- we get *by construction* a continuous line,
- we get *by construction* a thin line (except if we want the potential to be constant for a part of the grey-level scale),

- the result is localized exactly where the feature is,
- the result is robust ; we definitely also use parameters but contrary to the thresholding methods, their choice is less critical and discrete noisy pixels may have any value without influencing the result (see Fig. 24).

The complexity is in $O(k * n)$ where n is the number of pixels in the image and k the maximum number of zigzags in a feature, and the number of operations is thus comparable to the best performances of edge detection.

Another classical method of edge detection is the snake method [Kas 87]. The applications are quite different. The snakes in particular deal with the problem of movement of a contour through time, and with the detection of virtual contours, which is out of the field of applications of our attraction energy. On the other side, for a fixed pattern, we synthesize the grey-level information through the attraction energy before any propagation, while snakes perform much computation during the propagation. Algorithm 1 has most of its computation performed in the interest zone automatically (zone where the energy values change the most). Besides, only one point is propagated, not a complete line, thus we spare computer time.
Furthermore, we do not need any initial guess, we obtain a continuous line, and we are sure of reaching a global minimum (not a local one).

6.2.2 Comparison with other distances

We may notice that the numerical distance generalizes usual distances. Effectively, the pseudo-Euclidean distance corresponds to a numerical distance where ϕ is uniformly equal to the constant 1, and the grey-weighted distance defined by [Rut 68] to a numerical distance where ϕ is equal to the grey-level value of a pixel. Our distance develops this notion by taking into account not only a *function* of the grey-level but also the local direction of the path or the contrast.

6.2.3 Graph theory and dynamic programming

Other methods have extracted the information of minimum-cost paths in image processing [Mar 72][Har 68]. They used a cost function of the contrast and then performed a heuristic search in a graph for detecting minimum cost paths. Our method is also an extension of these works :

- we define the potential as a more general function ; not only the contrast intervenes, but also a function of the grey-level, the direction, and even neighboring pixels for cliques of order higher than two,
- we do not use a special structure for the graph, but work directly on the image, and thus avoid the computational problems,

- we do not take the risk of defining and using a global heuristic (it is worth noting that [Mar 76] used a threshold as a heuristic : grey-level information had to be added in this way to the contrast information used in the cost - but the information added was once again based on a thresholding...),
- we are sure to find a minimum cost path.

More generally, let us define a non-directed graph induced by the lattice of the image : $G = (V, E)$, where a vertex is a pixel of the image, and an edge is a couple of neighboring pixels for the chosen connectivity. For every edge, we define a cost λ as the minimum of the potentials of the cliques containing the two points : $\lambda(x_i, x_{i+1}) = \min_{c_i \in \mathcal{C}_n | x_i, x_{i+1} \in c_i} \phi(c_i, f_{c_i})$.

Then, $\mu(h) = \sum_{i=0}^{p-1} \lambda(x_i, x_{i+1})$. Thus, our first algorithm [Mer 91] is an adaptation of Ford's algorithm [For 56][For 62] to image processing. There are still important differences :

- our potential is general and adapted to image processing
- our graph is planar and each vertex has the same degree (except on the border of the image)
- we exploit this special structure of the graph (planar and with vertices of constant degrees), the coherency of the lines and eventually their direction to reduce the complexity : while the complexity of Ford's algorithm is in $o(card(V)^2)$, the complexity of our algorithm for all usual cases is in $o(card(V))$ - a constant number of scannings ; we do not expect in usual images to see random zigzags at the level of the pixel...

These advantages are also to be found if we compare the numerical distance to different algorithms of dynamic programming [Bel 65] ; each point has a constant number of neighbors, and we get an important part of the line in the same scanning, due to the structure of the pixels/states, and to the fact that we use our last computation immediately for our current computation : a neighbor of a point which is part of a line is a good candidate for being also on the line. Thus, instead of propagating one pixel in the line at each scanning, we propagate many pixels at the same time, depending on the sinuosity of the line. By scanning alternatively in all directions/in the privileged directions (depending on the application), we optimize the propagation.

If we compare our algorithm with the F* algorithm, we find that they are quite similar. However, we should stress :

- We have presented a mathematical formalization of the notions involved. In particular we have shown that the image obtained by integrating the cost is a distance function, and defined the cost as a potential on a clique.

- While Fischler [Fis 81] defines the cost relatively to one *point*, we define it relatively to a couple of points (and theoretically relatively to more points) - it corresponds to the *segment* between the two points. This allows to deal with more complex configurations and in particular the relative values of the two points : for computing the value of the energy in one point we do not take into account one cost but as many costs as there are neighbors. A consequence of second order is that the initial value for the starting points in our algorithm is equal to zero while the F* initializes them to their cost.
- We perform the proposed algorithm not only on isolated starting and goal points, but on sets of points. The distance formalization allows this generalization.
- The organization of the scannings in [Fis 81] is not optimal : there is a redundancy of computation for some directions (for each of the two horizontal directions left/right and right/left, the minimization is performed twice in the same cycle of scannings). In our algorithm, we solve this problem :
  1. Every direction of neighborhood appears only once.
  2. Every direction of neighborhood appears in the scanning for which there is recursivity : in this direction, the change of one pixel is taken into account in the computation of the next pixel during the same scanning.
  3. The algorithm is anisotropic through the use of four equivalent scannings in the four directions of propagation.
- This mathematical formalization leads also to practical extensions ; we consider cliques of order higher than two, and also neighborhoods of order higher than one. This allows to take directly into account notions such as contrast and curvature.

6.3 General framework, drawbacks

6.3.1 General framework

There are several ways of considering this distance based energy function. The general field of applications is the following : we have a feature (or a set of features) joining two extreme points (sets of points), and it may be characterized by some grey-level criterion, by a dominant direction, by a particular contrast, or by any local property depending eventually on the location. It may be interrupted and its direction may vary, it may even be sinuous. Then, we look for a distance function such that the shortest path between the extreme points for this distance is the feature. For this, we simply have to define the corresponding local potential.

We may notice that these developments stress the connection between important notions, such as distance and energy, global information and local information, graphs and energy (see [Gau 92] for another interesting connection between graph, energy and distance, in a completely different context). Besides, we may see the importance of non-stationary energies and potentials, for instance for features whose characteristics vary according to the location in the image.

The algorithms are performed generally within thirty or forty scannings, and the complexity is proportional to the number of pixels in the image. This rapidity is due to the fact that these algorithms privilege the zone of the feature (where the energy values change) and that they are both recursive : the change of one pixel value is taken into account *during the same scanning*. Thus, the number of scannings is linear with respect to the number of sinuosities of the feature.
Finally, one should notice that the lines obtained by using the proposed method are by construction continuous and thin.

6.3.2 Drawbacks

Two major drawbacks could be mentioned. First, we need to know the extreme points of the feature to be detected. However, this constraint may be partly relieved since S_0 may be an overset of one extremity, located far away from the feature (left border for some SPOT images for example), and since S_1 may be determined automatically as the set of the local minima of R in an overset of the other extremity. This last point shows the importance of the distance function for representing global information by synthesizing local information, independently from pattern recognition.

Second, there is a search process for determining ϕ, as usual with energy methods, which is naturally not limited. We proposed to impose constraints for guiding the search and to define the parameters relatively to the worst admitted configurations. Since ϕ may take into account wished tolerances (such as allowed deviations and maximal length admitted for the interruptions), and since it may be computed automatically from the histogram, we managed in previous applications not presented in this work [Mer 91] to obtain a robust potential for a whole class of applications (such as recognizing the staves in a music score, recognizing the brain fissure in MRI medical images). However, we did not reach this point so far for the SPOT images.

6.4 Future research

As a summary, our objectives are the following :

- Implementing the dynamic model, in particular to deal with the problematical images r3 and r4.
- Automatizing the determination of the potential, by finding a general shape of the potential for each class of images, or by computing automatically the potential from a series of small windows.

- Automatizing the search for S_1 and S_0. The search for S_1 should be easier since we may recognize it as the extremity of a valley. On the other hand, S_0 should be defined before any computation, and no information from the numerical distance function can therefore be used for this purpose.
- Speeding the computation by using pile strategies, propagating at the beginning the new values of the distance only from a restricted number of pixels having low distance values, or even using branch and bound when S_1 is known early.

7 Conclusion

This distance function is thus at a meeting point between graph theory, energy methods, and edge detection. Its efficiency, simplicity, rapidity, and adaptability make it powerful for detecting imperfect edges in digital images.
We have actually extended the F* algorithm, and optimized its recursivity. In particular, we have developed a mathematical formalism of the F* algorithm which allows to extend it to cliques of order higher than two and to neighborhoods of order higher than one, taking into account notions of contrast and curvature. Besides, we have shown how the F* could be considered as the static understanding of a physical mechanical model.
Further developments may be envisioned. We intend in particular to automatize the search for the potential, for S_0 and S_1, to speed the computation, and to implement the dynamic mechanical model, which may nicely introduce the global notion of curvature in the F*.

8 Appendix : proofs of the theorems and algorithms

8.1 Theorem 1 :

8.1.1 Symmetry

The symmetry of Δ follows from the symmetry of a clique (there is no privileged point in the clique).

8.1.2 Separability

Δ takes positive values since the potential takes positive values.

The separability is satisfied because of the constraint imposed on the potential :

$$\begin{aligned}\Delta(x_1, x_2) = 0 &\Longleftrightarrow \forall i, \phi(c_i, f/c_i) = 0\\ &\Longleftrightarrow \forall i, |c_i| = 1\\ &\Longleftrightarrow x_1 = x_2\end{aligned}$$

8.1.3 Triangular Inequality

Let us assume that there exist three points x_1, x_2 and x_3 such that :

$\Delta(x_1, x_2) + \Delta(x_2, x_3) < \Delta(x_1, x_3)$.

Let h_1 be the shortest path for μ between x_1 and x_2, and h_2 the shortest path for μ between x_2 and x_3 :

$\Delta(x_1, x_2) = \mu(h_1)$ and $\Delta(x_2, x_3) = \mu(h_2)$.

Let us define h'_3 the union of h_1 and h_2 : $h'_3 = h_1 \cup h_2$, path joining x_1 to x_3.

Then : $\mu(h'_3) = \mu(h_1) + \mu(h_2) = \Delta(x_1, x_2) + \Delta(x_2, x_3) < \Delta(x_1, x_3)$. In contradiction with the definition of $\Delta(x_1, x_3)$.

Thus, the triangular inequality is satisfied.

We conclude from 8.1.1, 8.1.2, and 8.1.3 : Δ is a distance.$\Box$

8.2 Theorem 2 :

Let us consider a point x, a set S_0 and the shortest path between them, h_0 : $\Delta(x, S_0) = \mu(h_0)$.

We define a new function λ, for the purpose of the proof, as the shortest length (in the sense of the usual distance) of the shortest paths (in the sense of the numerical distance) from x to S_0 :

$\lambda(x) = \min_{\{h \in P(x,S_0), \Delta(x,S_0)=\mu(h)\}} l(h)$, where $l(h)$ is the length (number of points) of h.

We define the induction hypotheses $H(n) : \forall x, \lambda(x) = n \Longrightarrow R(x) = \Delta(x, S_0)$.

Let us prove first $H(2)$.

8.2.1 $H(2)$

We consider x such that $\lambda(x) = 2$. Then, $\Delta(x, S_0) = \phi(c, f/c)$ where $x \in c$ and $x_1 \in c \cap S_0$.

Since $R(x_1) = 0$, we have : $\Delta(x, S_0) = R(x_1) + \phi(c, f/c)$.

Thus, by definition of R, we get : $R(x) \leq \Delta(x, S_0)$.

Let us show the other side of the inequality. We assume that there is some x for which :

$R(x) < \Delta(x, S_0)$ and $\lambda(x) = 2$.

Since $x \notin S_0$ and by definition of R, we have : $R(x) = R(u_1) + \phi(c_0, f/c_0) < \Delta(x, S_0)$ and $\phi(c_0, f/c_0) > 0$.

In the same way, we define a series (u_i) : $u_0 = x$, and by induction $R(u_i) = R(u_{i+1}) + \phi(c_i, f/c_i)$.

The potential is positive and bounded from below, thus $R(u_i)$ decreases. Since R is positive, $R(u_i)$ converges. The potential has a strictly positive lower bound outside of S_0, thus $R(u_i)$ converges to 0 in a finite time : $R(u_p) = 0$ and $u_p \in S_0$.

From these p equalities which define (u_i), we deduce : $R(x) = \sum_{i=0}^{p-1} \phi(c_i, f/c_i) < \Delta(x, S_0)$, in contradiction with the definition of $\Delta(x, S_0)$.

Thus, $H(2)$ is true : $\forall x, \lambda(x) = 2 \Longrightarrow R(x) = \Delta(x, S_0)$.

8.2.2 $H(n+1)$

Let us assume that $H(n)$ is true. We consider x such that $\lambda(x) = n+1$: $\Delta(x, S_0) = \mu(h_0)$ with $l(h_0) = n+1$.

We have : $\Delta(x, S_0) = \phi(c_0, f/c_0) + \mu(h_1)$ with $h_0 = [x, x_1] \cup h_1$ and $l(h_1) = n$.

Thus, by $H(n)$, we get : $\Delta(x, S_0) = \phi(c_0, f/c_0) + \Delta(x_1, S_0) = \phi(c_0, f/c_0) + R(x_1)$.

Hence : $R(x) \leq \Delta(x, S_0)$.

We want now to show the other side of the inequality.

If we assume that there is some x for which $R(x) < \Delta(x, S_0)$, we define a series (u_i) in the same way as in $H(2)$ and by the same argument obtain a contradiction.

Thus, $H(n+1)$ is true and, by induction, $H(n)$ is true for $n \geq 2$.

In S_0, $R(x) = 0 = \Delta(x, S_0)$ too, and thus we conclude : $\forall x \in S, R(x) = \Delta(x, S_0)$. □

8.2.3 Set of minima of R

If x is outside S_0, then $R(x) = R(x_1) + \phi(c_0, f/c_0)$ by definition of R. Since ϕ is strictly positive, R is not minimum in x.

If x belongs to S_0, $R(x) = 0 \leq R(x_1)$ for any point x_1 of S. Thus, R is minimum, both locally and globally.□

8.3 Algorithm 1

8.3.1 Convergence

Im is decreasing with time and bounded from below, so it converges. It is defined on a finite grid and takes a finite number of values, so it converges after a finite time.

8.3.2 Rightness

We assume that we are at convergence and consider a pixel x.

If x belongs to S_0, $Im(x) = 0$ (unchanged). Thus, $Im(x) = \Delta(x, S_0)$.

If x does not belong to S_0, $Im(x) = \min_{x_i \in \mathcal{N}_x}\{Im(x_i) + \phi(c, f/c))\} > 0$ (ϕ is strictly positive).

Thus, Im satisfies the conditions of Theorem 2.

Hence : $Im(x) = \Delta(x, S_0)$.

The image we get is R, or equivalently $\Delta(., S_0)$. □

8.4 Algorithm 2

8.4.1 Convergence

We notice :

- we mark only points such that $Im2(x) = 0$,
- after x is marked, $Im2(x) > 0$,
- there is a finite number of points.

Thus, Im2 converges after a finite time.

8.4.2 Rightness

We assume that we are at convergence. We call $\mathcal{M}$ the set of marked points and $\mathcal{S}$ the set of points belonging to a shortest path between S_0 and S_1.

We want to show that $\mathcal{M}$ and $\mathcal{S}$ are equal.

Let h_0 be a shortest path between S_0 and S_1 : $x_0 \in S_0, x_1, ..., x_n \in S_1$. At the initialization step, x_n has been marked.

Along the path, $Im2(x_{i+1}) = Im2(x_i) + \phi(c_i, f/c_i)$ since :

- $Im2(x_{i+1}) \leq Im2(x_i) + \phi(c_i, f/c_i)$ by computing Im2.
- if $Im2(x_{i+1}) < Im2(x_i) + \phi(c_i, f/c_i)$, $Im2(x_{i+1}) = Im2(y) + \phi(c, f/c)$. y produces a shorter path for μ, in contradiction with the hypothesis that h_0 is a shortest path between S_0 and S_1.

By induction from x_n to x_0, all the points of h_0 are marked. Thus, $\mathcal{S} \subset \mathcal{M}$.

Let us consider now a point x of $\mathcal{M}$: x has been marked, and we want to find a shortest path between S_0 and S_1 passing through x.

We consider the marking process (Algorithm 2) starting from S_1 and define the series (v_i) of propagation of the marking from S_1 to x : $v_0 \in S_1, v_1, ..., v_n = x$.

We consider the building process of Im (Algorithm 1) starting from S_0 and define the series (u_i) as the successive points of the shortest path from S_0 to x : $u_0 \in S_0, u_1, ..., u_p = x$.

We define the series (w_i) by combining (u_i) and (v_i) :

$w_0 = u_0 \in S_0, ..., w_p = u_p = v_n = x, w_{p+1} = v_{n-1}, ..., w_{p+n} = v_0 \in S_1$.

Along this path, $Im(w_{i+1}) = Im(w_i) + \phi(c_i, f/c_i)$.

Thus, $Im(w_{p+n}) = \mu(h_w) = \Delta(w_{p+n})$ where h_w is the path defined by (w_i), and h_w is a shortest path between S_0 and S_1.

Since x belongs to h_w, x belongs to $\mathcal{S}$.

Thus, $\mathcal{M} \subset \mathcal{S}$.

We deduce : $\mathcal{M} = \mathcal{S}$; the result of the algorithm is exact. □

8.5 Definition 3

The lattice is finite, n and m are finite constants, so there is no problem to define the attraction cost as a minimum.
There is a finite number of paths between x and y, so the definition of the attraction energy creates no difficulty here neither.
The fonction U is separable if the potential satisfies the condition defined on the potential in Theorem 1 (same demonstration as in Theorem 1), and is symmetric because of the symmetry of the cliques, but it does not satisfy anymore the Triangular Inequality. An example for this last point may be easily found with two paths such that the potential of the clique at their junction is not null and such that the background has high potential values. Then the cost of the combination of these paths takes into account the potential at the junction and is thus bigger than the sum of the costs of its both subpaths which do not include this potential in their computation. □

8.6 Algorithm 3

8.6.1 Convergence

Algorithm 3 converges in a finite time since there is a finite number of images Im1, and since each image Im1 decreases with time and is bounded from below (see Appendix, Algorithm 1).

8.6.2 Rightness

The demonstration is very similar to the one of Theorem 2.

We have at convergence :

$$Im1_{N_0}(M_0) == \min_{i=1,2,3}\{Im1_{P_i}(N_0) + Im2_{c_i}(M_0)\}$$

with the same notations as in Algorithm 3.

We want to prove that $Im1_{N_0}(M_0)$ is the cost of the shortest paths from S_0 to M_0 whose point before the last one is N_0. Let h_0 be such a shortest path.

We define a new function λ_{N_0}, for the purpose of the proof, as the shortest length (in the sense of the usual distance) of the shortest paths (in the sense of the numerical distance) from M_0 to S_0 passing through N_0 :
$\lambda_{N_0}(M_0) = \min_{\{h=x_0,\dots x_p \in P(S_0,M_0), \Delta_{N_0}(S_0,M_0)=\mu(h) \wedge x_{p-1}=N_0\}} l(h)$, where $l(h)$ is the length (number of points) of h, and $\Delta_{N_0}(x,y)$ is the minimum cost of the paths between x and y whose point before the last one is N_0.

We define the induction hypotheses $H(n)$:
$\forall M_0, \lambda_{N_0}(M_0) = n \Longrightarrow Im1_{N_0}(M_0) = \Delta_{N_0}(S_0, M_0)$

Let us notice here that for the sake of brevety N_0 is a neighborhood direction, not a point, in the expressions Δ_{N_0}, λ_{N_0} and $Im1_{N_0}$.

Let us prove first $H(3)$.

$H(3)$

We consider M_0 such that $\lambda_{N_0}(M_0) = 3$. Then, $\Delta_{N_0}(M_0, S_0) = \phi(c, f/c)$ where $M_0 \in c$, $N_0 \in c$, and $P_0 \in c \cap S_0$.

Since $Im1_{P_0}(N_0) = 0$, we have : $\Delta_{N_0}(M_0, S_0) = Im1_{P_0}(N_0) + \phi(c, f/c)$.

Thus, due to the property satisfied by $Im1_{N_0}$, we get : $Im1_{N_0}(M_0) \leq \Delta_{N_0}(M_0, S_0)$.

Let us show the other side of the inequality. We assume that there is some M_0 for which :

$Im1_{N_0}(M_0) < \Delta_{N_0}(M_0, S_0)$ and $\lambda(M_0) = 3$.

Since $M_0 \notin S_0$ and by definition of $Im1_{N_0}$, we have :
$Im1_{N_0}(M_0) = Im1_{u_2}(N_0) + \phi(c_0, f/c_0) < \Delta(M_0, S_0)$ and $\phi(c_0, f/c_0) > 0$.

In the same way, we define a series (u_i) : $u_0 = M_0$, $u_1 = N_0$ and by induction $Im1_{u_{i+1}}(u_i) = Im1_{u_{i+2}}(u_{i+1}) + \phi(c_i, f/c_i)$, where c_i contains u_i, u_{i+1} and u_{i+2}.

The potential is positive and bounded from below, thus $Im1_{u_{i+1}}(u_i)$ decreases. Since Im1 is positive, $Im1_{u_{i+1}}(u_i)$ converges. The potential has a strictly positive lower bound outside of S_0, thus $Im1_{u_{i+1}}(u_i)$ converges to 0 in a finite time : $Im1_{u_{p+1}}(u_p) = 0$ and $u_p \in S_0$.

From these p equalities which define u_i, we deduce :
$Im1_{N_0}(M_0) = \sum_{i=0}^{p-1} \phi(c_i, f/c_i) < \Delta_{N_0}(M_0, S_0)$, in contradiction with the definition of $\Delta(M_0, S_0)$.

Thus, $H(3)$ is true : $\forall M_0, \lambda(M_0) = 3 \Longrightarrow Im1_{N_0}(M_0) = \Delta_{N_0}(M_0, S_0)$.

$H(n+1)$

Let us assume that $H(n)$ is true. We consider M_0 such that $\lambda(M_0) = n+1$: $\Delta_{N_0}(M_0, S_0) = \mu(h_0)$ with $l(h_0) = n+1$.

We have : $\Delta_{N_0}(M_0, S_0) = \phi(c_0, f/c_0) + \mu(h_1)$ with $h_0 = [M_0, M_1] \cup h_1$ and $l(h_1) = n$.

Thus, by $H(n)$, we get : $\Delta_{M_1}(M_0, S_0) = \phi(c_0, f/c_0) + \Delta_{M_2}(M_1, S_0) = \phi(c_0, f/c_0) + Im1_{M_2}(M_1)$.

Hence : $Im1_{M_1}(M_0) \leq \Delta_{M_1}(M_0, S_0)$.

We want now to show the other side of the inequality.

If we assume that there is some M_0 for which $Im1_{N_0}(M_0) < \Delta_{N_0}(M_0, S_0)$, we define a series (u_i) in the same way as in $H(3)$ and by the same argument obtain a contradiction.

Thus, $H(n+1)$ is true and, by induction, $H(n)$ is true for $n \geq 3$.

Thus we conclude : $\forall M_0 \in S, \lambda_{N_0}(M_0) \geq 3 \Longrightarrow Im1_{N_0}(M_0) = \Delta_{N_0}(M_0, S_0)$. □

8.7 Algorithm 4

8.7.1 Convergence

We notice :

- we mark only points such that $Im2(N_i) = 0$,
- after N_i is marked, $Im2(N_i) > 0$,
- there is a finite number of points.

Thus, Im2 converges after a finite time.

8.7.2 Rightness

We assume that we are at convergence. We call $\mathcal{M}$ the set of marked points and $\mathcal{S}$ the set of points belonging to a shortest path between S_0 and S_1.

We want to show that $\mathcal{M}$ and $\mathcal{S}$ are equal.

Let h_0 be a shortest path between S_0 and S_1 : $x_0 \in S_1, x_1, ..., x_n \in S_0$. At the initialization step, x_0 has been marked.

Along the path, let x_i be the last point marked. If x_i does not belong to S_0, then x_{i+1} is the neighbor for which $Im1_{x_{i+1}}(x_i)$ is minimum. So it must have been labeled, which contradicts our hypothesis that x_i is the last point marked. Thus, all the points of the path S are labeled. Thus, $\mathcal{S} \subset \mathcal{M}$.

Let us consider now a point x of $\mathcal{M}$: x has been marked, and we want to find a shortest path between S_0 and S_1 passing through x.

We consider the marking process (Algorithm 4) starting from S_1 and define the series (v_i) of propagation of the marking from S_1 to x : $v_0 \in S_1, v_1, ..., v_i = x$. This propagation carries on until $v_n \in S_0$.

Along the path, we have : $Im1_{v_{i+1}}(v_i) = Im1_{v_{i+2}}(v_{i+1}) + Im2_{v_i,v_{i+1},v_{i+2}}(v_i)$. Thus, the shortest path from S_0 to v_i passes through v_{i+1}. By induction starting from S_0, this is a shortest path from S_0 to S_1.

Thus, $\mathcal{M} \subset \mathcal{S}$.

We deduce : $\mathcal{M} = \mathcal{S}$; the result of the algorithm is exact. $\square$

9 References

[Aze 87] R. Azencott. "Image Analysis and Markov Fields". *Proc. of the Int. Conf. on Ind. and Appl. Math*, SIAM, Paris, 1987.

[Bel 65] R. Bellman, R. Kalaba, Shortest Paths Through Networks, in *Dynamic Programming and Modern Control Theory*, Academic Press, New York and London, 1965, pp. 50-54.

[Bes 74] J. Besag, Spatial Interaction and the Statistical Analysis of Lattice Systems, *J.R. Statist. Soc.*, 1974, vol. JRSS B-36, pp. 192-236.

[Can 86] J. Canny, A computational approach to edge detection. *IEEE PAMI*, vol. 8(6), pp. 679-698, 1986.

[Cha 88] B. Chalmond, Image Restoration Using an Estimated Markov Model, *Signal Processing* 15, 1988, pp. 115-129.

[Cox 90] I.J. Cox, R.A. Boie, D.A. Wallach, Line Recognition, *IEEE* 1990, pp. 639-645.

[Dao 89] M. Daoud, C. Roux, A. Hillion, Une application de la theorie des graphes a l'extraction automatique des reseaux de communication dans les images du satellite SPOT. *12th Colloquium GRETSI*, Juan-Les-Pins, 12th-16th June 1989.

[Der 87] R. Deriche, Using Canny's criteria to derive a recursively implemented optimal edge detector, *International Journal of Computer Vision*,1(2), May 1987.

[Des 86] I. Destival, Morphologie Mathematique appliquee aux images des satellites de teledetection. *International Electronic Image Week*, Second Image Symposium, Nice, April 1986.

[Fis 81] M.A. Fischler, J.M. Tenenbaum, and H.C. Wolf, Detection of roads and linear structures in low-resolution aerial imagery using a multisource knowledge integration technique. *CVGIP* 15, 201-223, 1981.

[For 56] L.R., Jr., Ford, Network Flow Theory, *The Rand Corp.*, 1956, August, P-923.

[For 62] L.R., Jr., Ford, D.R. Fulkerson, Flows in Networks, *Princetown Univ. Press*, 1962, Chap. III, Sec. 5.

[Gar 89a] P. Garnesson, G. Giraudon, P. Montesinos, Messie : un systeme multi specialistes en vision, application a l'interpretation en imagerie aerienne. *Research report 1012*, INRIA, April 1989.

[Gar 89b] P. Garnesson, G. Giraudon, P. Montesinos, Detecting buildings and roads in aerial images by a multi expert system. *Conference on Image Processing*, Avignon, 29th May-2 June 1989.

[Gau 92] I. Gaudron, 2D Objects Recognition By Graph Matching. *Proc. ICPR*, The Hague, September 1992.

[Gol 69] M. Golay, Hexagonal Parallel Pattern Transformation. *IEEE Trans. Comput.*, C18, 1969, pp. 733-740.

[Gra 89] C. Graffigne, I. Herlin, Modelisation de reseaux pour l'imagerie satellite SPOT. *RFIA Conference*, Paris, 29th Nov 89-1st Dec 89.

[Har 68] P.E. Hart, N.J. Nilsson, B. Raphael, A Formal Basis for the Heuristic Determination of Minimum-Cost Paths, *IEEE Trans. Sys.Man.Cyb.*, 1968, vol. SMC-4, pp. 100-107.

[Hou 62] P.V.C. Hough, Methods and Means for Recognizing Complex Patterns, *US Patent 3,069,654*. 1962.

[Hou 91a] S. Houzelle, G. Giraudon, Automatic feature extraction using data fusion in remote sensing. *Proc. conf SPIE : sensor fusion*, vol. 1611, pp. 12-15, Nov. 91.

[Hou 91b] S. Houzelle, G. Giraudon, Data fusion using SPOT and SAR images for bridge and urban area extraction. *Proc. IGARSS*, Helsinki, June 1991.

[Jed 91] B. Jedynak, D. Geman, A. Gagalowicz, Detection de reseaux routiers a partir des images du satellite SPOT. *RFIA Conference*, Lyon-Villeurbanne, 25th-29th Nov 1991.

[Kas 87] M. Kass, A. Witkin, D. Terzopoulos, Snakes : active contour models. *Proc. of the First Int. Conf. on Comp. Vis.*, pp. 259-268, London, June 1987.

[Mar 72] A. Martelli, Edge Detection Using Heuristic Search Methods, *Comp. Graphics Image Proc.*, 1972, vol. 1, pp. 169-182.

[Mar 76] A. Martelli, An application of heuristic search methods to edge and contour detection. *Commun. Ass. Comput. Mach.*, vol. 19, pp. 73-83, Feb. 1976.

[Mat 75] G. Matheron, Random sets and integral geometry, *Wiley and S.*, New York, 1975.

[Mer 91] N. Merlet, From music staves to brain fissure through a distance based energy function, *4th International Conference of Computer Analysis of Images and Patterns (CAIP)*, Dresden, September 17-19th, 1991.

[Rut 68] D. Rutowitz, Data structures for operations on digital images, *Pictorial pattern recognition (G.C. Cheng et al., eds.)*, Thompson, Washington DC, 1968, pp. 105-133.

[Ser 82] J. Serra, Image Analysis and Mathematical Morphology, *Academic Press*, London 1982.

[Ser 89] M.A. Serendero, Extraction d'informations symboliques en imagerie SPOT : reseaux de communication et agglomerations. *These de Doctorat*, Universite de Nice, 9th December 1989

Figure 3: Original image g1

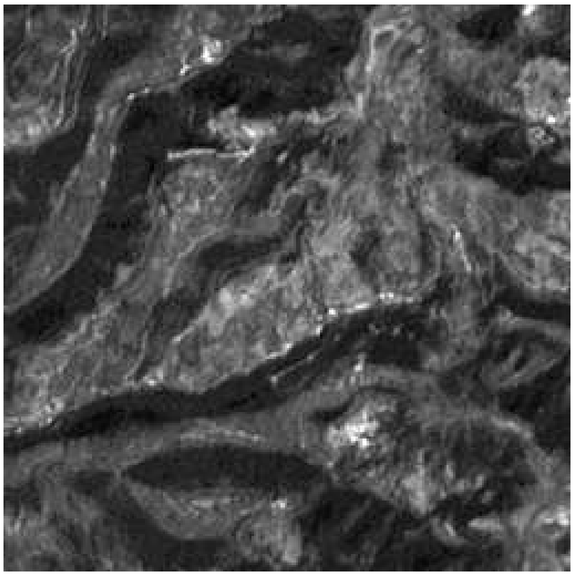

Figure 4: Original image g2

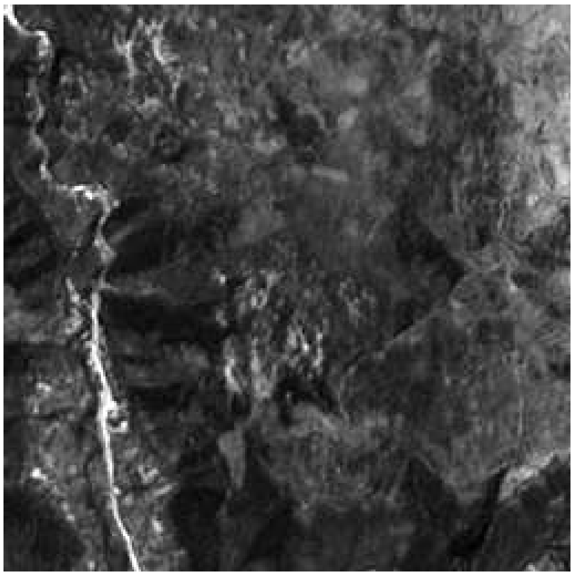

Figure 5: Original image g3

Figure 6: Original image r1

Figure 7: Original image r2

Figure 8: Original image r3

Figure 9: Original image r4

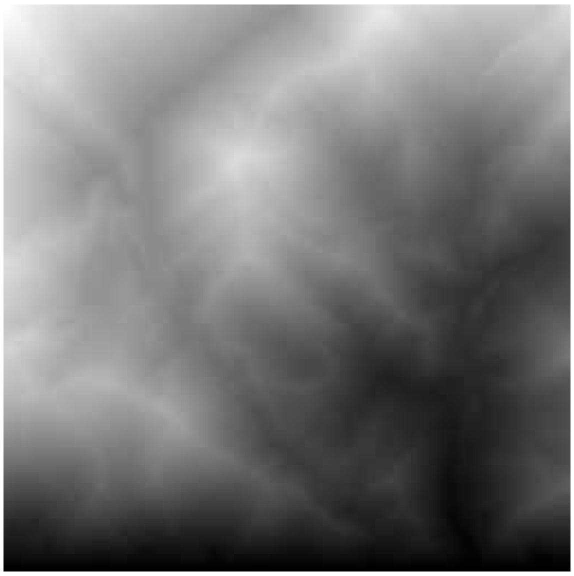

Figure 10: Distance-energy for g1 with cliques of order two and a static model

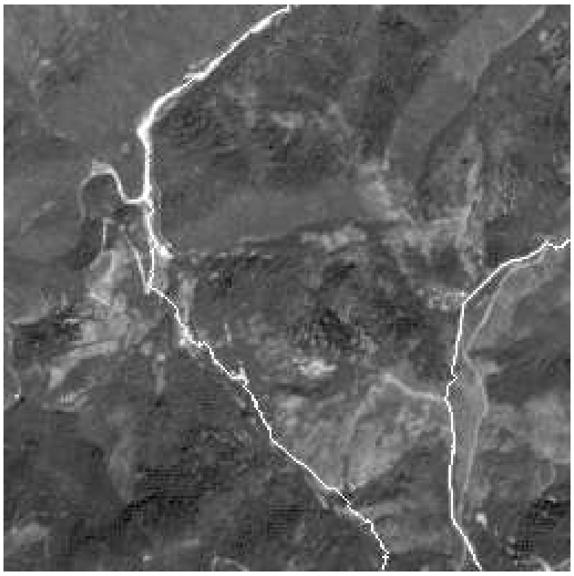

Figure 11: Segmentation of the valley and roads in g1 with cliques of order two and a static model

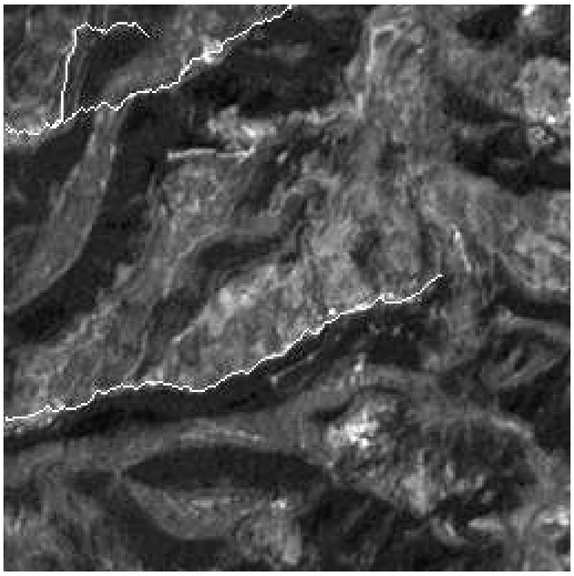

Figure 12: Segmentation of the valleys in g2 with cliques of order three and a static model

Figure 13: Segmentation of the roads and channels in r1 with cliques of order three and a static model - first test

Figure 14: Segmentation of the roads and channels in r1 with cliques of order three and a static model - second test

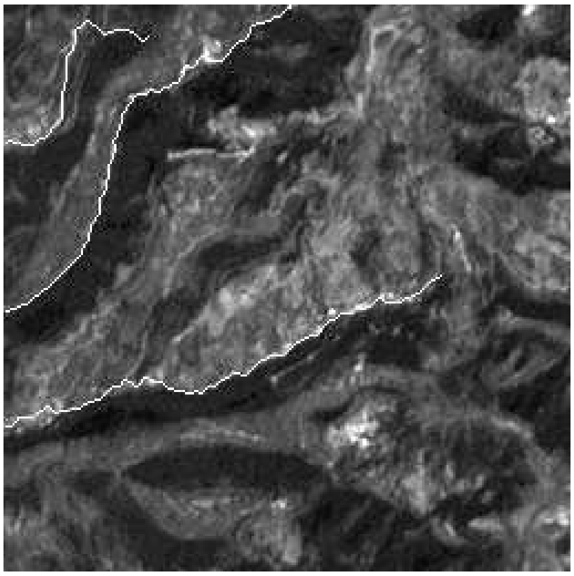

Figure 15: Segmentation of the valleys in g2 with neighbors of second order

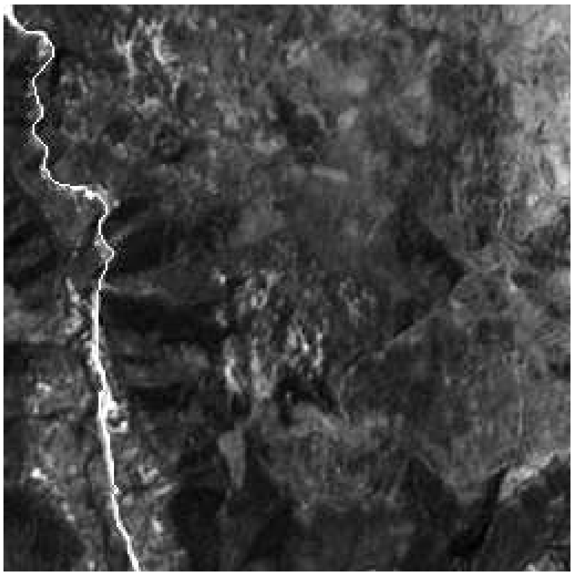

Figure 16: Segmentation of the valleys in g3 with neighbors of second order

Figure 17: Segmentation of the roads and channels in r1 with neighbors of second order - first test

Figure 18: Segmentation of the roads and channels in r1 with neighbors of second order - second test

Figure 19: Segmentation of the roads and channels in r1 with neighbors of second order - third test

Figure 20: Segmentation of the roads and channels in r1 with neighbors of second order - fourth test

Figure 21: Segmentation of the roads and channels in r1 with neighbors of second order - fifth test

Figure 22: Segmentation of the roads and channels in r1 with neighbors of second order - sixth test

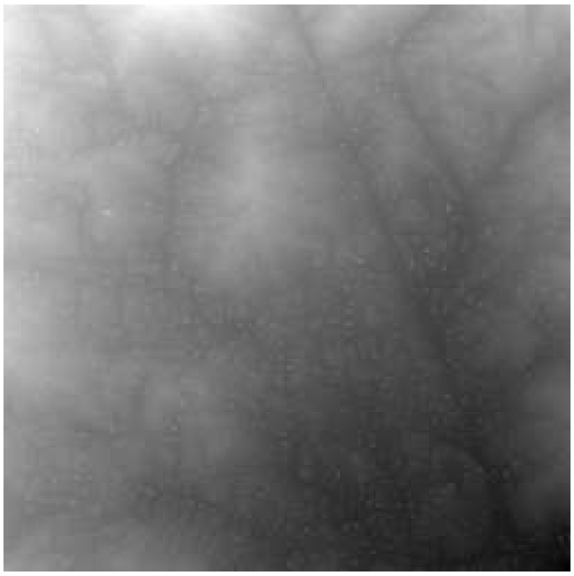

Figure 23: Energy for r2 with neighbors of second order

Figure 24: Segmentation of the roads in r2 with neighbors of second order

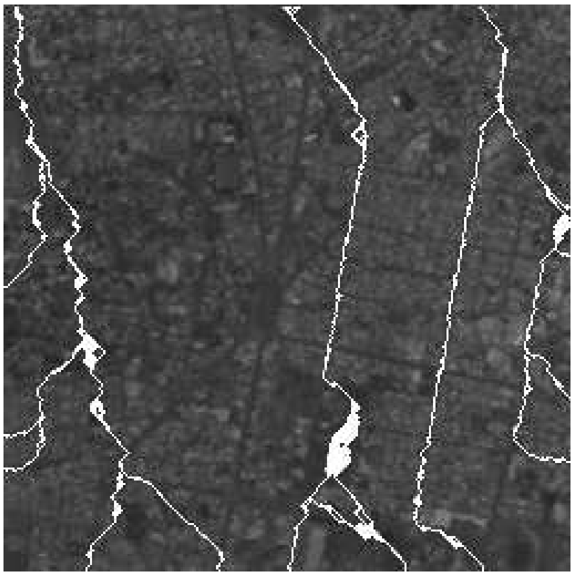

Figure 25: Segmentation of the roads in r3 with cliques of order two and a static model

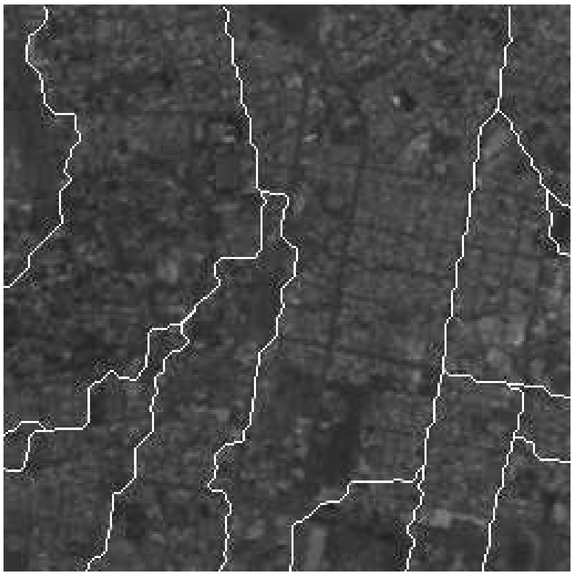

Figure 26: Segmentation of the roads in r3 with neighbors of second order

Figure 27: Segmentation of the roads in r4 with neighbors of second order