Integration of Global Information for Roads Detection in Satellite Images

Nicolas Merlet, Josiane Zerubia

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Integration of Global Information for Roads Detection In Satellite Images

Nicolas Merlet , Josiane Zerubia

N^o 3239

Août 1997

THEME 3

ISSN 0249-6399

Integration of Global Information for Roads Detection In Satellite Images

Nicolas Merlet [*] , Josiane Zerubia [**]

Thème 3 — Interaction homme-machine,
images, données, connaissances
Projet PASTIS

Rapport de recherche n°3239 — Août 1997 — 104 pages

Abstract: Several difficulties are met when detecting roads in satellite images. They are due to the poor quality of existing information and may be solved by using global information. Dynamic programming is popular in line detection, but deals awkwardly with global information due to its constraints on the decision process and to the local nature of the cost.

There are various ways to integrate global information in this framework. The most widely used consists in attributing to each state the pre-processed characteristics of a more global feature which contains the state. A second solution is to use a hierarchic approach.

Thirdly, the states may be defined as a set of successive pixels. We have developed a new algorithm of roads detection, which takes into account local curvature on three points.

We present in our work a new solution : auxiliary functions. They store temporary information about the current shortest path, to take into account global information (direction, curvature) in the potential. They are updated in a recursive way each time a new shortest path is found. Optimality is lost, but the show the effect of the constraint in several examples. The constraint may be modulated, and the complexity of the problem is not changed.

Besides shape information, we present a new method to define automatically the cost from the grey-levels in windows around the extremities of the roads.

We finally compare the speeds of computation with various scannings of the states : by eliminating part of the states and ordering them, the gains are often above 50 %.

Key-words: roads detection, dynamic programming, satellite images, global information

(Résumé : tsvp)

[*] Institute of Computer Science,
The Hebrew University of Jerusalem,
91904 Jerusalem, Israel.
nicolas@cs.huji.ac.il

[**] INRIA,
BP 93,
2004 Route des Lucioles,
06902 Sophia Antipolis Cedex, France.
zerubia@sophia.inria.fr

Integration d'Information Globale en Detection de Routes sur Images Satellitaires

Résumé : La détection de routes sur images satellitaires comporte de nombreuses difficultés liées à la mauvaise qualité de l'information existante. L'utilisation d'une information globale est alors d'un grand recours. La programmation dynamique est un outil courant en détection de lignes, mais se prête mal à l'utilisation d'une information globale compte tenu des contraintes sur le processus de decision et de la nature locale du coût.

Plusieurs méthodes sont envisageables pour intégrer de l'information globale dans ce formalisme. La plus fréquente consiste à calculer tout d'abord les caractéristiques globales d'une forme de l'image, et à les attribuer aux états qui composent la forme.

Une deuxième possibilité utilise une approche hierarchique, du grossier vers le fin.

Troisièmement, les états peuvent être définis de manière structurée. Nous avons développé un nouveau programme de détection de routes, qui prend en compte la courbure locale en définissant le coût sur trois pixels successifs.

Nous présentons une nouvelle solution : les fonctions auxiliaires. Elles représentent au cours des itérations une information temporaire sur la meilleure solution trouvée, et sont prises en compte dans le potentiel. Elles sont mises à jour récursivement à chaque nouvelle solution trouvée. La solution n'est plus optimale, mais elle incorpore correctement la contrainte, qui peut même être modulée. La complexité du problème est inchangée.

Outre l'information de forme, nous présentons une nouvelle méthode pour définir automatiquement le potentiel en fonction des teintes de gris présentes au voisinage des extrémités des routes. Contraste et teinte de gris sont pondérés à l'aide de probabilités conditionnelles.

Finalement, nous comparons les temps de calcul avec différents balayages des états : en ignorant une partie des états et en les ordonnant, ou en arrêtant le processus avant convergence. Les gains sont souvent de l'ordre de 50 %.

Mots-clé : détection de routes, programmation dynamique, images satellitaires, information globale.

Contents

1 INTRODUCTION

1.1 Problem

An important domain of image processing is image interpretation, which consists in extracting from an image the meaningful parts. These parts may be one-dimensional (edges) or two-dimensional (regions). The edges are generally related to discontinuities of properties while the regions correspond to similarities. The applications of image segmentation concern most of human activities : industry, medicine, agronomy, robotics, remote sensing to quote a few of them. When segmenting edges in grey-level images, a further distinction may be done between methods involving a thresholding and those not involving a threshold. The use of a threshold is often preceded by a pre-processing such as a convolution or a morphological operation and followed by a post-processing such as linking, tracking or applying a Hough transform. Methods not using a thresholding are generally based on the minimization of an energy.

Roads detection in satellite or aerial images has been the subject of intensive research in the last twenty years, for civil as well as military purposes, using the different methods of edge detection above mentioned. Roads are generally classified as thin networks of width one pixel, and thick networks of width more than one pixel. We will focus our study on thin networks. The methods developed for thin networks may then be applied to detect one of the two sides of thick networks. Of course, many of the algorithms developed in roads detection may be applied as well to other fields, such as medical imaging, document understanding, industrial control. However, several problems are particularly acute in remote sensing :

- lack of information :
  - interruptions appear in the road because trees or clouds hide it
  - the road is confused with neighboring regions of similar grey-level (fields) or of similar contrast (high-textured urban area).
- vague information : noise blurs the limit of a road.
- repetitive information : two parallel roads have similar characteristics.
- disturbing information : local artifacts of high-contrast are caused by noise.
- multiple information : sometimes, the road may be well-defined only by the conjunction of information of different kind (grey-level, contrast, shape...)
- global information : it may play a decisive role to solve ambiguities (shape information in particular).

A widely used method to deal with these difficulties is to define an energy model, and to minimise the energy with dynamic programming. Dynamic programming solves an N-stage decision process by decomposing it into a sequence of N single-stage processes. The mathematical formulation leads to a recurrence relation which can be evaluated sequentially. For a path of length N and with states having k neighbors, the complexity is k^N with a N-stage decision process, but only $k.N$ with N single-stage processes. One of the most popular formulations consists in three steps : first, a local cost or potential function is defined, then it is integrated on all or part of the space relatively to some reference space, and finally the shortest path is retrieved by back-tracking. Dynamic programming presents several advantages :

- the principle is simple and no additional pre- or post-processing is necessary. Dynamic programming binds directly global features with local characteristics through integration. Global features correspond to the human observation and request, while local characteristics correspond to the computer's most basic possibilities.
- the complexity remains low through recursivity, in particular if some minimal information is known about the road, such as its direction. In this case, a constant number of scannings is enough when the direction of scanning is chosen accordingly. The computations are local.
- no threshold is used. All the information is used in a smooth way, none is lost at an early stage. Methods relying on thresholding often have to retrieve later the information lost earlier, through hysteresis thresholding, linking, and even dynamic programming.
- Several problems of roads detection are directly solved :
  - short interruptions are jumped over as neglectable terms in the integration
  - isolated artifacts are entirely ignored when looking for a global optimum
  - multiple information may be taken into account in the cost function as a function of several variables

However, there are obvious limits of dynamic programming, which are related to the local definition of the cost :

- Long interruptions (fields or urban areas) may produce wrong results.
- The solution does not necessarily correspond exactly to the feature in case of blurring.
- Repetitive patterns may be mixed.

The integration of global information may be therefore of great benefit. However, it is not easy for several reasons :

- Global information may be difficult to take into account in the local potential, in particular shape information. For instance, how to take into account curvature when the computation is done at the pixel level ?
- The use of dynamic programming assumes two conditions on the nature of the problem :
  1. Decision stages must be ordered so that all the stages whose results are needed at a given stage have been processed in advance.
  2. The decision process should verify : at any stage the behavior of the process depends solely on the current state and does not depend on the previous history.

In this report, we will consider various ways to integrate global information within this framework :

1. The most widely used technique consists in computing as pre-processing some global characteristics of a higher-order feature, such as the orientation of a contour, and attributing these characteristics to each state belonging or related to the feature.
2. An other solution is to use a multi-resolution or hierarchical approach, from coarse to fine.
3. The states and decisions may be defined in a structured way. A state may be composed of several basic elements (for instance several successive pixels in roads detection).
4. We present in our work a new solution : auxiliary functions in dynamic programming. They store temporary information about the current shortest path to take into account global information in the cost function. They are updated in a recursive way each time a new shortest path is found. Optimality is lost and the second condition to apply dynamic programming is not verified, but we will show several examples where the results fit the expectations.
5. Sometimes, the decision stages can not be ordered and even the first condition is not verified. Relaxation may be then a good alternative to dynamic programming.

The first method above mentioned (1) does not impose directly a constraint on the solution, but it imposes the solution to contain points of a feature verifying this constraint. On the contrary, the other methods intervene directly on the solution (2,3,4,5).

1.2 Method

1.2.1 Theoretical definitions

The principle of optimality [2, 3] is the basis for dynamic programming [18]. It claims : "An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision". Intuitively, it means that a subpath of a shortest path is also a shortest path.

We adopt similar formalism and notations as Bellman [3] : We consider a state s and a decision q which uniquely defines its successor $\tau(s,q)$. In many applications of image analysis, s is a pixel of the image and q is one of the 8 (or 4) directions of the grid.

We suppose the existence of a potential $\phi(s,q)$ (or cost function) and of two distinct sets of reference states S_0 and S_1. We define the function to minimize U (energy) as :

$$U = \sum_{s=S_0}^{S_1} \phi(s,q) \tag{1}$$

We suppose the following basic hypotheses :

- there is a finite number of states s
- there is a finite number of decisions q
- $\forall s, \forall q, \phi(s,q) \in \mathcal{R}^{+*}$ (to avoid cycles)
- between two states s_1 and s_n there is always at least one path from s_1 to s_n $[s_1, \tau(s_1,q_1), \tau(\tau(s_1,q_1),q_2), ..., s_n]$ and one path from s_n to s_1 $[s_n, \tau(s_n,q_n^*), \tau(\tau(s_n,q_n^*),q_{n-1}^*), ..., s_1]$

Under these assumptions, there is a minimum for U and it is reached for some path (Appendix A). Minimizing U also defines a path between S_0 and S_1.

It can be shown that the minimization is obtained by iterating the relation :

$$\forall s \neq s_0, U(s) \leftarrow \min_{q \in Q}\{\phi(s,q) + U(\tau(s,q))\} \tag{2}$$

with, at initialization :

$$U(s_0) = \min_{q \in Q}\{\phi(s_0,q)\}$$
$$\forall s \neq s_0, U(s) \leftarrow +\infty.$$

A more algorithmic approach uses a temporary variable tmp :

$$\forall s, \forall q, \quad tmp \leftarrow \phi(s,q) + U(\tau(s,q)) \tag{3}$$

$$tmp \leq U(s) \Rightarrow U(s) \leftarrow tmp \tag{4}$$

This formulation is equivalent to the first one, since $\min_{q \in Q}\{\phi(s,q) + U(\tau(s,q))\}$ decreases between two scannings of the states (Appendix A). In the following, we will adopt the second formulation for three reasons :

- it allows to optimize and compare the number of elementary operations for various possible scannings of the states
- it allows to decompose the set Q in a partition (Q_i), where each subset Q_i is scanned in a different scanning to speed the integration.
- for more sophisticated expressions, the condition of decreasing of $\phi + U$ is not verified anymore, but the second formulation still enforces the convergence of the algorithm.

If we suppose that ϕ is a function of the state s (independent of the decision q) only, we define a function d_ϕ of two states in the following way :

$$\forall s_1, \forall s_2 \neq s_1, d_\phi(s_1, s_2) = \min \sum_{s=s_1}^{s_2} \phi(s) \tag{5}$$

$$d_\phi(s_1, s_1) = 0$$

and d_ϕ is then a distance function on the set of the states (Appendix A).

The framework of dynamic programming is very general and allows a lot of applications according to the choice of the space, states, decisions, potentials, extremity states, and scannings.

1.2.2 Practical choices

The state is the basic unity of the problem. To the state is associated an energy. It may consists of one, two or more points of the grey-level image or of some other search space, and verify eventually some constraint.

The decision defines the way states succeed one another along the solution path and along the partial solutions found. It is related to the neighborhood of interaction between states, to constraints on the shape of the path, and to speed considerations.

The potential may depend on many concepts which define intuitively the basis of the problem : grey-level, local direction, disparity, contrast, local curvature... Once these concepts are qualitatively defined, the potential must be defined quantitatively.

Once these initial data are defined, integration of the potential may be performed in a variety of ways. We have to scan the states and decisions of the space, but we may skip part of them or order them in an efficient way to impose constraints on the result or to increase the speed of computation. Besides, the iterations may be stopped before convergence when the result found is "good enough", and the potential may be updated through the iterations when more information has been obtained.

The third step is retrieving the shortest path through backtracking. Some choices must still be done : the energy has been computed for all the states of the space or for the majority of them, and we have to choose the state from which backtracking starts. This influences greatly the result in general, and one or more states may be selected.

1.3 Outline of the report

We will first overview, in Section 2, several works done using dynamic programming in road and line detection.

We will then see how to define the state s and the decision q according to the shape wished for the solution (Section 3), in particular by choosing the state as a state of pixels to take into account local curvature.

In Section 4, we will consider how to define numericall the potential, interactively with heuristics, or with a new automatic method based on conditional probabilities.

Section 5 will be devoted to the integration of the potential through equations (3) and (4). There is a bias with dynamic programming, because it favors paths containing less states since the energy is then computed from a smaller number of terms, and we will deal with this bias by computing the average instead of the sum of the potentials. We will also compare different methods to scan efficiently the states by ordering and eliminating part of the states and decisions at initialization or during integration, and by stopping the integration before convergence. Finally we will consider the updating of the potential during integration when more information has been obtained.

An important innovation in this report is the use of auxiliary functions incorporated to equations (3) and (4). They store temporary information about the current shortest path to take into account global information in the potential. They are updated in a recursive way each time a new shortest path is found. Optimality is lost and the second condition to apply dynamic programming is not verified, but the results fit the expectations. These functions appear through the study with different kind of information. Their general framework is presented in Section 3.

2 OVERVIEW OF RELATED WORK

We stress that many methods have been developed and used in roads detection, but we focus our study only on the ones which are based on dynamic programming. Dynamic programming is an organized framework which is widely used to integrate information in various ways. This restriction allows to restrain both the review and the research to a reasonable scale and to dispose of simple elements of comparison.

We have classified papers dealing with line detection by dynamic programming according to the way they integrate global information (see Section 1) :

- with characteristics of higher-order features
- with a hierarchical approach
- with complex states
- with auxiliary functions
- with several methods together

2.1 With characteristics of higher-order features : Fischler et al.'s method

Fischler et al. [6] consider two types of detectors : those without false alarms (type I) and those without misdetections (type II), and combine them in the framework of dynamic programming. They obtain edges with the Duda road operator (type I) and automatic thresholding. Global information is therefore integrated at this step, as characteristics of a higher-order feature.

The cost c is then defined as the grey-level intensity (type II), except for edges obtained with type I, where c is equal to zero. Then the F* algorithm finds the lines by performing the minimization $P(i_0, j_0) = min\{P(i, j) + c(i_0, j_0)\}$ where P is the value of the current minimum path to (i_0, j_0) (initialized to $+\infty$), and (i,j) is one of the eight neighbors of (i_0, j_0). A curvature constraint is obtained by adding a constant to c, and filling of gaps is limited by taking c^n instead of c. Therefore :

- The state s is one pixel.
- The decision q is one of the 8 directions of the grid.
- The potential is independent from the decision and is a function both of the Duda operator and of the grey-level.
- The search is recursive. The maximum number of scannings is proportional to the length of the longest minimal path, but the average number of scannings is much lower, due to the computing of the energy for several successive pixels belonging to a given path in the same scanning.

2.2 With a hierarchical approach

2.2.1 Geman and Jedynak's algorithm

First, edges of high contrast are filtered at high resolution, and dynamic programming is then applied on this binary image at two different resolutions [7]. At low resolution, configurations where segments are perpendicular are prohibited. Then, at high resolution, constraints are imposed on the angle between two successive segments in order to localize the edges. At low resolution,

- a state s is a couple of neighboring windows 32*32. There are 16*16 windows in an image of size 512*512, and each window has 8 neighbors.
- a decision q is a direction in the low resolution grid. For a given state, a direction producing a perpendicular new state is prohibited.
- the potential of a state is the number of pixels belonging to the straight line crossing the border of the two windows and maximizing this number of pixels. This means that we apply the Hough transform on a binary image.
- the function to maximize is the average of the potentials and not the sum ; therefore, the authors exclude the use of dynamic programming for this function. Instead, successive approximations are done, and dynamic programming is used at each step. This is practically possible because the number of states is small (16*16).

Thus, a rough solution for the road is found. Its localization is then precisely determined at high resolution. The road is assumed to be piecewise linear, and :

- a state is a linear segment.
- a decision is a direction of the grid, where constraints on the angle between successive segments are taken into account.
- the potential of a segment is the number of pixels on a straight line.
- then, either dynamic programming or search in a graph may be used.

2.2.2 Figueiredo and Leitao's method

Line detection is a common problem to many domains. In medical imaging for instance, Figueiredo and Leitao [5] detect vessel borders and skeleton in angiography. The user first defines a start cross-section and an end cross-section of the vessel. Both sections are approximately parallel, and intermediary sections are defined automatically between them. In each section, a 1D morphological filter is applied : top-hat transform to detect the center of the vessel (difference of grey-level and morphological opening) and dilation gradient to detect the two borders (difference of morphological dilation and grey-level). Then, dynamic programming is applied on these sections :

- a state is a local maximum in a section
- a decision is a vector between local maxima in two different and successive sections
- the potential is a function (linear combination) of the euclidean distance between two successive states (increasing) and of the value of the morphological filter on these states (decreasing).
- virtual states are added before the start section and after the end section to allow the selection of a state also in these sections.

The shortest path found is composed of a series of local maxima of the filter considered, it minimizes the distance between successive states and maximizes the values of the filter in these states.

2.3 With complex states : Martelli, Daoud et al.'s techniques

Martelli has shown in [9] that "any optimization problem which can be posed as a dynamic programming problem can also be stated as the problem of finding the shortest path on a graph". He uses this equivalence for detecting edges with shortest path search algorithms and corresponding heuristics. The initial conditions may be summarized as follows :

- The state s is a set of n couples of neighboring pixels. In each couple, one pixel is on one side of the edge, and the other pixel on the other side of the edge. The couples succeed one another in a supposed edge. n is a constant depending upon the application.
- The decision q is one of the 4 directions of the grid.
- The potential ϕ is defined as a function of the grey-levels of the $2 * n$ points and of their curvature (defined on three successive pairs). It depends upon the difference of grey-levels of both pixels of each pair.
- The search is performed with the A* algorithm, which imposes to store and compare a great number of paths. More precisely, the complexity is exponential w.r.t. the length of the path and w.r.t. n. Martelli [9] speeds the search with a heuristic based on a thresholding, and optimality may be therefore lost.

Daoud et al. [4] applied Martelli's algorithm to roads detection. Several heuristics and criteria of failure are used to speed the computation and avoid dead ends and artifacts (convergence is not guaranteed). For instance, the initial direction at the extremity is taken into account into the potential in the neighborhood of the extremity, and search is done starting from both extremities of the road.

2.4 With auxiliary functions : Shashua and Ullman's algorithm

Shashua and Ullman [19] have developed a method to detect salient structures in binary images with a multistage optimization approach. The information used is curvature, curvature variation and length. Gaps are filled in and contours may be smoothed.

The formalism used is slightly different from [3]. A state is a couple (P, θ) where P is a pixel and θ a direction to a neighbor. P may be null or not. The energy is defined for a sequence of N states $(p_i, .., p_{i+N})$ as $E_i = \sum_{j=i}^{i+N} C_{i,j}.\rho_{i,j}.\sigma_j$ where :

- $C_{i,j} = \Pi_{k=i}^{j-1} f_{k,k+1}$ is a curvature factor. $f_{k,k+1}$ is a positive decreasing function of the angle between the states of rank k and k+1 ; it is maximum and equal to one when the states are aligned. The underlying idea is curvature.
- $\rho_{i,j} = \Pi_{k=i+1}^{j} \rho_k$ is an attenuation factor. ρ_k is equal to one for non-null pixels, otherwise it is equal to a constant between 0 and 1. The attenuation factor allows to penalize gaps.
- σ_j is a saliency measure equal to one for non-null pixels, to zero otherwise.

The energy is then maximized for the set of curves passing through every point. Due to the form of the curvature and attenuation factors, it is possible to optimize with dynamic programming. At initialization, the energy is defined as $E_i^{(0)} = \sigma_i$ and then updated at iteration (n+1) with : $E_i^{(n+1)} = \sigma_i + \rho_i . \max_{p_j \in \delta(p_i)} E_j^{(n)} . f_{i,j}$ where p_j is a neighbor of p_i. At iteration N, the curves considered are those of length smaller than N. The authors claim that the algorithm converges and that it converges for a simple curve (one successor only to each point) towards the value defined above. In the result, curves of great saliency are obtained with a greater brightness and width than the other curves. We notice that both $C_{i,j}$ and $\rho_{i,j}$ are components of the potential which contain global information on the current shortest path. They correspond to the concept of auxiliary functions which we define in Section 3.

2.5 With several methods together

2.5.1 Gruen and Li's technique

In a recent paper [8], Gruen and Li present a method of semi-automatic extraction of roads. After a Wallis filter and a wavelet transform to enhance roads, dynamic programming is applied in the following way :

- The authors assume a positive contrast between the road and the background.
- According to the authors, a state is a pixel.
- Along the curve, the authors consider the lines of length 5 pixels, perpendicular to the curve and separated by a fixed distance. A decision is a vector joining pixels of two successive lines. The curve is approximated as a polygon. During the iterations,

the length of the decision vector decreases. Thus, global information is integrated by using different resolutions.

- Dynamic programming is applied between several seed pixels given by the user.
- The potential takes into account the grey-level of a pixel, the square of its difference with the average in a short segment of the road, the geometric distance to the linear approximation of the curve, the angle between 3 successive points and the distance between two successive pixels. The linear approximation of a curve is a second way of integrating global information. Finally, the potential is a function of several successive points. Therefore, the three first ways of integrating global information (see Section 1) are combined in this work.

2.5.2 Barzohar and Cooper's method

Barzohar and Cooper [1] detect thick networks automatically, without any manual input of road extremities. They define a model of road and background grey-levels and apply dynamic programming in two steps :

- partitioning the whole image in small windows, to find good road candidates, starting from each of the four sides of the window. The second-best road is also selected.
- then starting from these good candidates and tracking the road by applying dynamic programming in new windows along the road. Thus, the approach is hierarchical.

A particular original point is the synthesis of completely artificial satellite images, which look quite natural.

In each window, dynamic programming is applied in the following way :

- a state is a 4-uple of pixels corresponding to two edge points on both sides of the road and to the two following edge pixels. The global information is therefore also integrated through complex states.
- for a horizontal road, a decision is a direction from a pixel to a pixel in the coming column. In the following a road is supposed horizontal, without loss of generality.
- the potential is a function of :
  - the curvature on three successive pixels, more precisely an approximation of the second derivative (of the ordinates for a horizontal road). The pixels considered are the road centers.
  - the difference between the road width and the width at the two previous columns
  - the difference between the road mean grey-level and the grey-level at the two previous columns

- the detection of an edge (by a thresholding of the gradient) on both sides of the road at the current pixel
- the variance of grey-level on the current cross-section of the road.

3 DEFINING THE STATES AND DECISIONS

When we determine the initial conditions, we define which types of features we want to detect, and these conditions are therefore closely related to the problem to solve. They involve the following questions :

- in which space do we work (a 2D space or a 3D space) ?
- on real images or on artificial ones created in an ad hoc way ?
- what are the possible states s ?
- what are the possible decisions q ?
- does the potential ϕ depend upon the decision q ?
- on which characteristics of the state does the potential depend upon ?
- what are the numerical values of the potential ϕ for the various values of s and q ?

The first questions are qualitative and we will consider them together in this section. Then, we will try to answer the last one, which is quantitative, in the Section 4.

In the chosen formalism, the energy is a function of the state s solely, and the choice of s should be done in relation to the question : which energy do we want to store and how far in the past do we want to memorize information ? In particular, we consider the four following cases :

1. The state s is a single point. Very often, global information is then taken into account by attributing to each state the characteristics of a higher-order feature which it belongs to.
2. The state s is a pair of two neighboring points (more precisely an oriented pair). This allows to take into account local curvature.
3. The state is composed of more than 2 points to deal with more global curvature, or of non neighboring points to change the level of resolution. These are two different ways of integrating global information (complex states and multi-resolution approaches).
4. For dealing with more global information, we will use auxiliary functions, giving up optimality. This is the most novel way of integrating global information.

The choice of the possible decisions q corresponds to the way states succeed to each other on the resulting feature and this involves the following concepts :

- dimension of the space (2D or 3D)
- connectivity (4, 6, 8 ...)

- resolution on the path
- restriction of the local directions of propagation to reduce the computing time
- restriction of the local directions of propagation to impose a constraint on the result ; in particular, we will see how to limit the allowed curvature along the path. The allowed decisions may vary for different states or may be the same.
- avoiding loops on the path (see Section 5), when minimizing the average of the potential instead of the sum.

In all these configurations of the state and the decision, the potential may depend upon a set of states $S(s,q)$ uniquely determined by s and q. This set is implicitly contained in the definition of the potential without loss of generality in the definitions, proofs or algorithms. For instance, when the potential depends on a filter related to a convolution on a given neighborhood, then $S(s,q)$ is this neighborhood. This corresponds to the integration of global information by using higher-order features.

In all the applications, the principal rule for defining the potential is that the potential should be lower for the values and attributes of the state s and decision q that are characteristic of the features to detect. For instance, if the feature is clear relatively to the background, then the potential should be small for high values of the grey-level ; then the feature is expected to be the shortest path for this potential.

3.1 State as a single point

3.1.1 Potential dependent on the decision

When the potential is independent from the decision, $\phi(s)$ depends only upon the point of the state, and eventually upon a neighborhood S(s) of it. This is the configuration most often used in image processing (see [6] in particular), where the decision is one direction of the grid (out of 4 or 8).

When the potential ϕ is a function of the decision q, $\phi(s,q)$ may take into account more information than $\phi(s)$:

- non-homogeneity of the directions : in a film of the flow of vesicules in a nervous cell for instance, one direction of the decision corresponds to time and the others to space. In this case, the space is 3D.
- anisotropy : in radiologic images of the brain, the vertical direction corresponding to the fissure of the brain is favored. In an other application, if we want to recognize staves in a music score, the horizontal direction is a characteristic of the features.

3.1.2 Detecting roads in a satellite image

Herein, we want to recognize the main roads in the SPOT satellite image [11] shown in Fig. 1, left. These roads have the following characteristics :

- they have a relatively low density, but with great variations ; besides, the density of the road in a clear part of the image may be higher than the density of the background in a dark part ; thus, a density criterion (only) appears insufficient ;
- due to the topology, they may make great zig-zags ; in the whole image, they have no dominant direction and thus we can not use this information ;
- they have a contrast between ten and twenty with the background ; this information does not appear particularly robust however, since this is also the case of hills and clouds ;
- they are junctions between the different borders of the image ; they present coherent characteristics on a wide length, contrary to the hills and clouds in this image.

For this application, the problem is defined as follows ::

- A state is one pixel of the image,
- There are 8 decisions corresponding to the 8 directions of the grid.
- The potential takes into account both grey-level and contrast, which separately are not sufficient. We define the potential as the sum of a density factor h and a contrast factor g. It depends upon a set $S(s,q)$ which contains the states s and $\tau(s,q)$, and a third neighboring point used to compute the contrast. This set intervenes only during the computation of the potential, and not during the several iterations of the integration.
- S_0 is the left vertical border of the image.
- S_1 is the set of the local minima of the energy on the three other borders. It is computed after the potential has been integrated.

Let us consider a set of three points A, B, C ordered in increasing order of grey-level value f. The ideal configuration is the one where $f(A)$ and $f(B)$ are minima (road) and $f(C)$ is a maximum (background) : for this configuration both h and g should be minima so that the potential is minimum.

This allows us to precise the potential :

- $\phi(s) = h(s) + g(s)$
- we define $S(s,q)$ as the pixel (out of 2 or 4) which is neighbor both of s and of $\tau(s,q)$, and where the grey-level is the higher.

- $h(s)$ is a function of the median of the three grey-level values of s,$\tau(s,q)$, and $S(s,q)$: $h(s) = h(med(f(s), f(\tau(s,q)), f(S(s,q))))$. Effectively, the median of these three values is the maximum of the two lower values, which in the case of a road, crossing the ideal configuration, is an upper bound of the grey-levels on the road. h should be low for usual values in roads.

- $g(s)$ characterizes the contrast between the road and the background ; we define it as a function of the difference between the maximum value (background) and the median value (local maximum of the grey-levels on the road) : $g(s) = g(d)$ with $d = max(f(s), f(\tau(s,q)), f(S(s,q))) - med(f(s), f(\tau(s,q)), f(S(s,q)))$. g should decrease in order to favor high contrasts.

We present in Fig. 1 (right) the result superimposed on the energy. One should notice how the values of the energy increase when we go away from S_0 and how the roads are underlined. We effectively obtained the main roads crossing the City (Fig. 2, left), although the initial image was of poor quality (low contrast in particular). The small streets inside the City and between the roads were not detected : since we recognize only the smallest paths for ϕ, we always take shortcuts.
Due to the many zig-zags of the roads, the computation was a bit heavier than in the examples of the music staves and of the brain fissure [11], forty scannings all together (ten for each of the four directions of scanning, top-down and bottom-up, rightwards and leftwards).

The potential is shown in Fig. 2, right. The h and g functions are displayed in Fig. 3. They were defined heuristicly.

Fig. 4 (left) is a zoom of a region where dynamic programming shows its power. Due to the poor contrast, segmentation of the roads appears very difficult using other methods. The energy is shown in Fig. 4 (middle), with the result in white ; the roads appear much better. Fig. 4 (right) represents the result of the propagation superimposed on the original image. One should notice in this last image the perfect fitting of the segmentation to the road, and how the result is, by construction, continuous and thin.

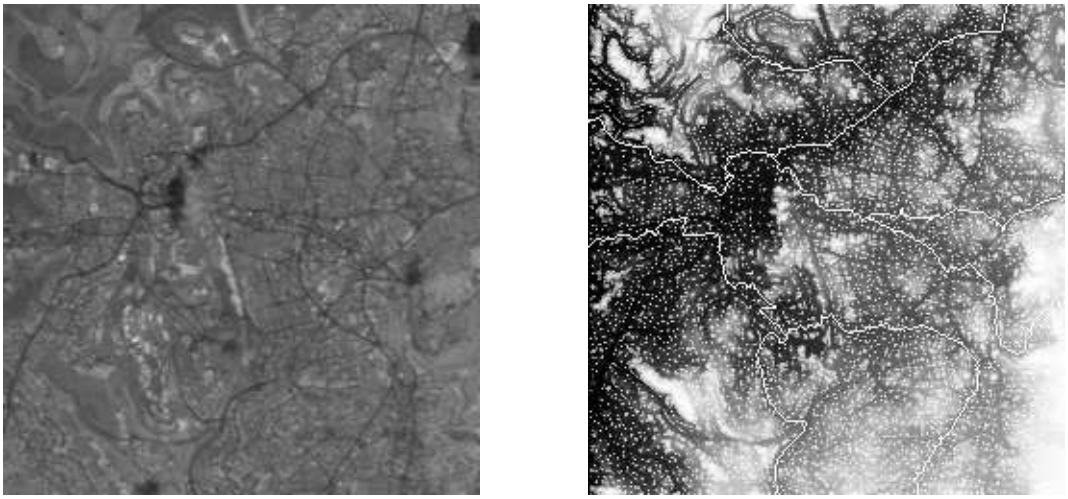

Figure 1: Original SPOT image of Jerusalem (left). Result superimposed on the energy (right).

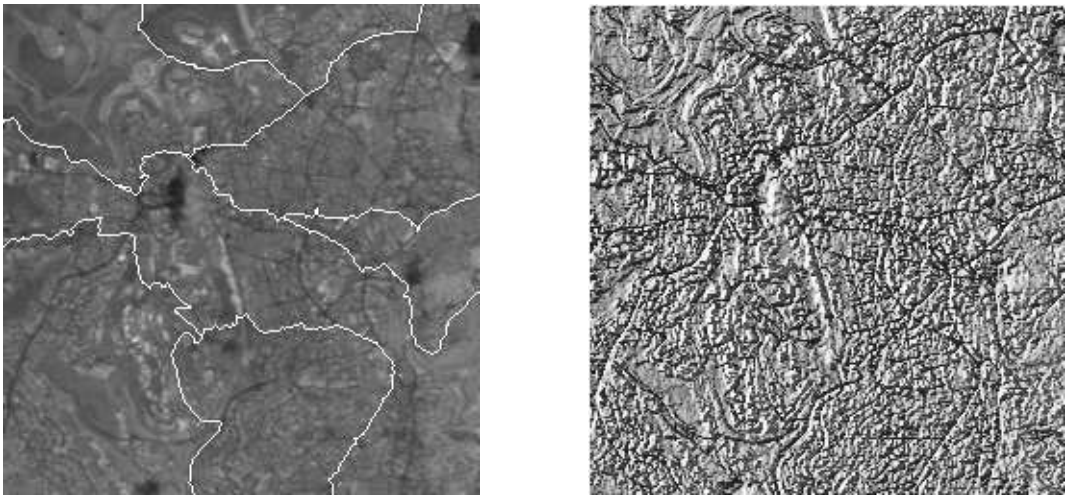

Figure 2: Result superimposed on the SPOT image (left). Potential (right).

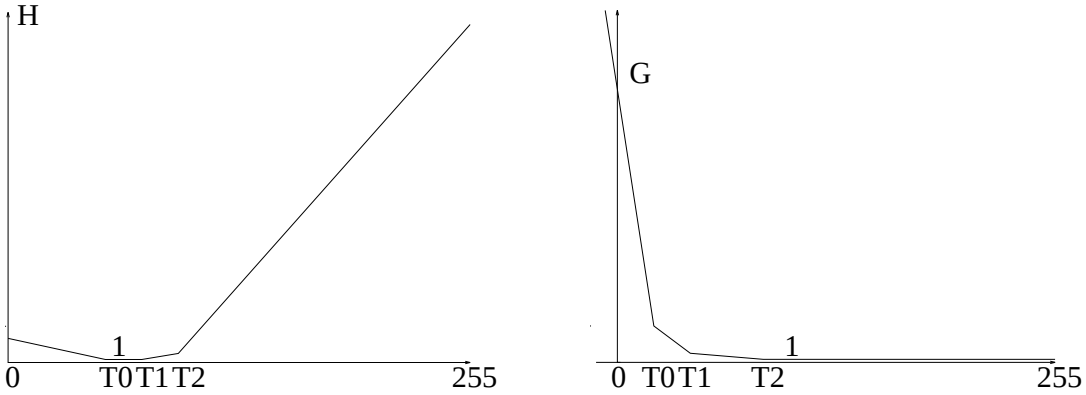

Figure 3: Potential of grey-level for the SPOT image (left). Potential of contrast (right).

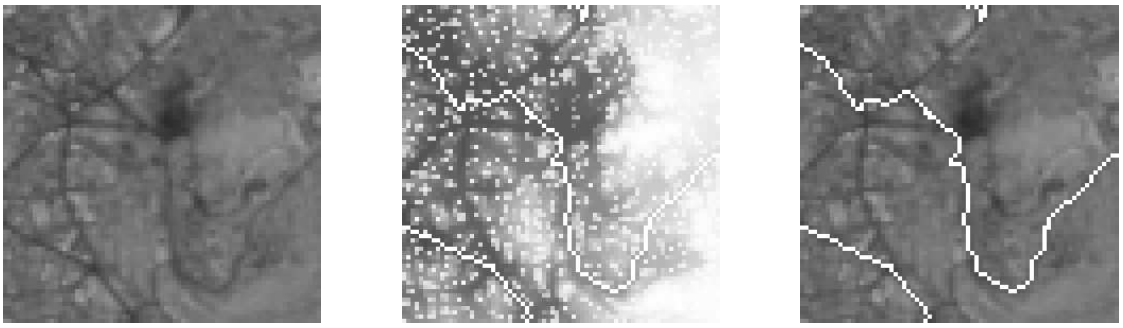

Figure 4: Zoom on the original SPOT image (left). Zoom on the result, superimposed on the energy (middle). Zoom on the result superimposed on the SPOT image (right).

3.2 State defined as 2 neighboring points

For some configurations of roads in satellite images, taking a state as a single pixel is not sufficient.

We present in Fig. 5 (right) a SPOT image [1] containing geological structures. The valleys are the transition lines from clear regions (up left) to dark ones (down right) - the light of the sun comes from the right bottom corner (south east). Since two structures are close to each other, they are mixed during the detection, when using a single pixel as a state (see Fig. 6, left). This may be solved by taking into account the local curvature. When the state is defined as 2 neighboring (oriented) points, these points form an angle with the decision, which may be used to take into account direction or curvature. An eventual use of two-point states is also to allow two paths to cross one another.

If we know the local direction at the extremity, we can restrict S_1 to a single pixel : the local direction defines the second pixel of the state. Otherwise, and this is the case in our applications, S_1 is the set of the states containing a given pixel (extremity) of the feature, since the second pixel as well as the local direction are undetermined. The same applies for S_0.

3.2.1 Local curvature

For instance, local curvature is defined between 3 successive points ([12, 13, 17]) : 2 from the state, one from the decision. Grey-level is taken into account in the state s and contrast in s and S(s,q). Therefore :

- a state is an oriented pair of two neighboring pixels ; there are 8 states for each pixel.
- a decision is one of the 3 directions prolongating a given state without making a deviation of more than 45 degrees. The decisions vary according to the orientation of the pair.
- S_0 is the set of states containing one extremity of the feature as first pixel (in this example, an overset : the left column).
- S_1 is the set of states containing the other extremities of the features as second pixel.
- the potential depends also upon the decision q (which defines the curvature). It is a function of grey-level, contrast, and curvature.

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 the three points of s and $\tau(s,q)$ on one hand, and on the six points of $S(s,q)$ on the other hand (see Fig. 5,

[1] Except for the previous image of Jerusalem, the satellite images in this work were supplied by the French Space Agency (CNES). The collaboration between the Hebrew University and the INRIA was financed by the Israeli Ministery of Science and Technology (MOST) and by the "Association Franco-Israelienne pour la Recherche Scientifique et Technologique" (AFIRST).

left). We call M_1, M_2 and M_3 the three points of s and $\tau(s,q)$, ordered according to the increasing values of the grey-level f : $f(M_1) \leq f(M_2) \leq f(M_3)$. ϕ is defined as :

$$\phi(s,q) = h(f(M_1)) + g(f(M_1) - median(S(s,q))) + k(s,q). \quad (6)$$

where $h(f(M_1))$ characterizes the worst of the three candidates for the valley, $f(M_1) - median(S(s,q))$ is the contrast between the valley and the background (more precisely the median value of the background), and $k(s,q)$ corresponds to the angle between s and $\tau(s,q)$. h and g are piecewise linear and decrease, while k increases with the angle (practically, k may take three values for reducing the computation).

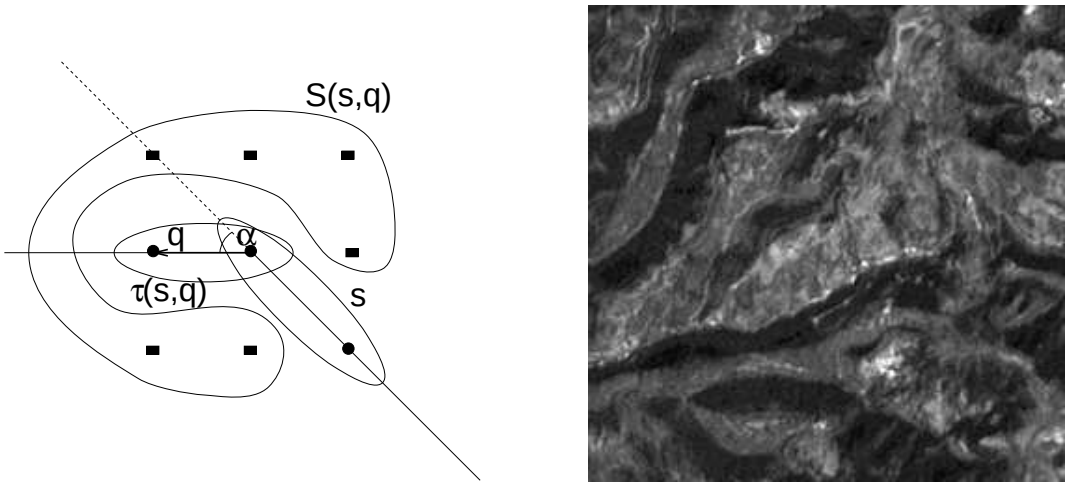

Figure 5: State s, decision q, and set $S(s,q)$ (left). Original image (g) (right).

The final result is presented in Fig. 6, right. The notion of curvature does not intervene here at the specific point where the propagation turns towards another feature, but on all the small curves of the deviation between both features. We notice this time the good fitting between segmentation and features.

When applying the same type of potential to a road image (Fig. 7, left), we obtain Fig. 7, right, as a final result. S_0 is the right bottom corner, S_1 is composed of the extreme points of the three features, and they are also defined manually. The result fits perfectly the features, although the human eye had difficulties to precisely localize the roads. The corresponding energy function U is presented in Fig. 8.

We have made other tests by varying the curvature coefficient k. In Fig. 9 (left), it was taken four times smaller than in Fig. 7 (right), and the path got lost in zigzags in highly textured regions. When using on the contrary a four times bigger coefficient, in Fig. 9

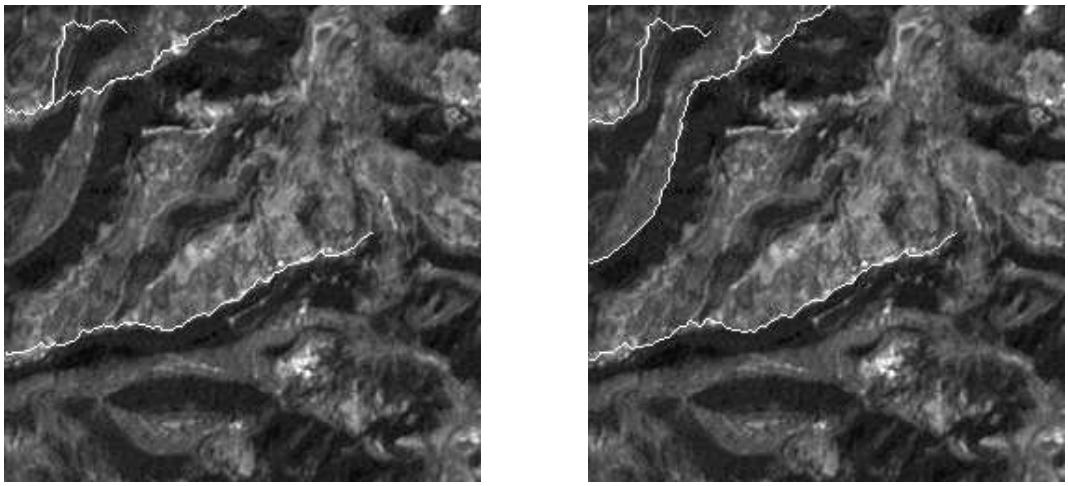

Figure 6: Segmentation of the valleys in (g) with one-point states (left). Segmentation of the valleys in (g) with two-points states (right).

Figure 7: Original image (r1) (left). Segmentation of the roads in (r1) with two-points states (right).

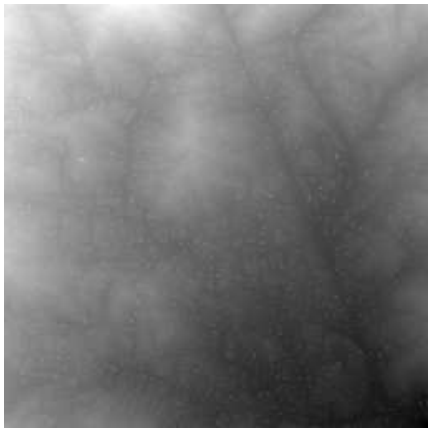

Figure 8: Energy function U for (r1) with two-points states.

(right), the real feature had too many zigzags, and the propagation jumped directly over the fields in straight lines.

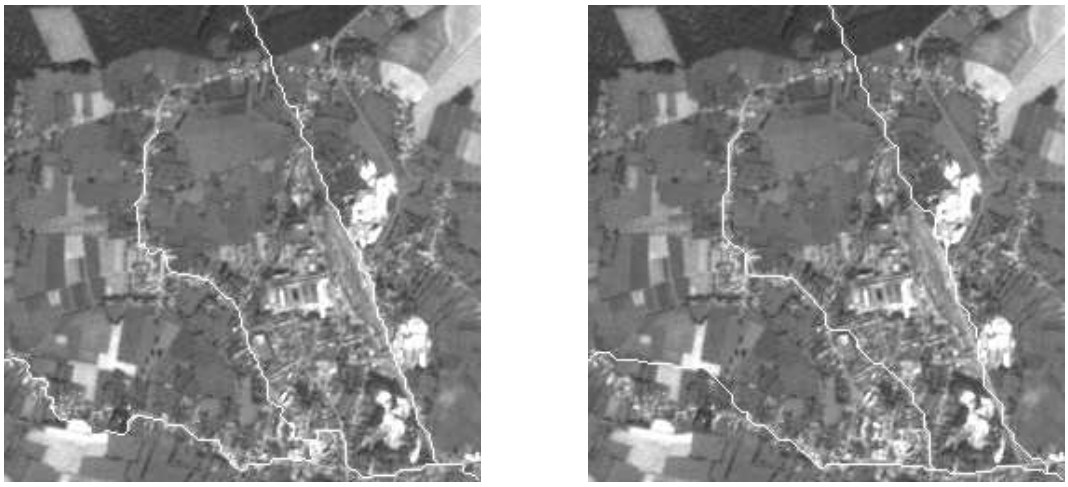

Figure 9: (r1) with a weaker constraint (left). (r1) with a stronger constraint (right).

3.2.2 Crossing of paths

A pixel located at the crossing of two curves may belong to two different states (two-point states) of the curves : the curves may have no state in common, and could be detected simultaneously without interference between them.

We have tried to detect simultaneously such paths crossing each other, and we have found that this is practically very difficult : most of the time, the paths merge between the junction and S_0. More precisely, only the shortest path between the junction and S_0 is detected. Actually, several factors could allow the detection of crossing paths, among them :

- the cost of the two different paths between the junction and two different points of S_0 should be similar : if a path is much shorter, then it may be the only one detected,
- the curvature should be very global and its weight in the potential high, in order to avoid sudden changes of direction at the junction,
- the attributes (grey-level, contrast, ...) of the features should be characteristic at the junction. A "blurred" junction favors merging between two different paths.

Practically, these conditions are rarely verified, and it is more convenient and robust to define S_0 at the junction of the paths and S_1 as the set of their extremities. The use of two pixels in the state in order to detect crossing roads remains essentially theoretical.

3.3 Other choices for the state

3.3.1 State defined as n neighboring points

For considering less local curvature, one may define the state on n points and use the (n+1) points obtained by adding the decision, for computing the curvature on them. The problem is that the number of states increases exponentially with n.

We have seen how Martelli [9] defines the curvature on n couples of points and looks for the shortest path with the A* algorithm associated with several heuristics ; the solution is not necessarily optimal.

3.3.2 State defined as n non-neighboring points

One may change the resolution by considering n points at a distance d ($d > 1$) from each other. The decision q then also corresponds to a vector of norm greater than one. There are then two problems :

1. at the low resolution, information from the high resolution should still be used, for detecting the presence of the road.
2. the path is found at a low resolution, with "holes" in it, and its localization is very rough.

Geman et al. [7] solve the first problem by applying a convolution and a Hough transform in windows of size 32*32. They then solve the second problem by applying again dynamic programming, at a higher resolution.

3.4 Use of auxiliary functions

In the formalism adopted herein, ϕ is a function of the state s and decision q. Works relying on the use of an edge-detector or a convolution remain in this framework, since the local neighborhood considered may be seen as a set of states S(s,q), and ϕ may be defined as a function of it (as seen previously). However, this additional information does not take into account the current value of the energy and the current corresponding shortest path since $S(s,q)$ is defined at initialization.

In order to take into account information in a farther past and to impose global constraints on the solution itself without obtaining high complexities, we must define some memory process. We propose to do this by defining an auxiliary function V(s) or V(s,q) [16]. It is updated before the energy U according to a function v depending upon the application, and U depends on it. More formally (the modifications in the equations are underlined below) :

$$\forall s, \forall q, \qquad tmp \leftarrow \quad \phi(s,q,\underline{V(\tau(s,q))}) + U(\tau(s,q)) \tag{7}$$

$$tmp < U(s) \Rightarrow \quad U(s) \leftarrow tmp \tag{8}$$

$$\underline{V(s) \leftarrow v(s,q,V(\tau(s,q)))} \tag{9}$$

and the corresponding new definition of the energy to minimize is

$$U = \sum_{i=1}^{n} \phi(s_i, q_i, V((q_j)_{j \in <i+1,n>})) \tag{10}$$

for any n. Thus, the value of each term depends upon the decisions already made. We note that this formula is related to the definition of the energy used by Shashua and Ullman.

In the general case, the principle of optimality is not verified anymore (see Appendix A), and the solution given by the algorithm is not necessary optimal. The sum $\phi + U$ (Equation 7) does not necessarily decrease with time, but the formulation of the algorithm imposes the decreasing of U, and the algorithm still converges in a finite time.

With the auxiliary functions, there is actually a learning process which is taken into account in the search. We have mentioned in Section 1 that applying dynamic programming requires from the decision process two conditions. With auxiliary functions, only the first one is verified since the potential depends upon several past states and eventually all of them.

3.4.1 Global direction

For the roads, we first take as auxiliary function V(s) the 2D unit vector corresponding to the main direction of the shortest path until the current state. We define the state s as a pixel and the decision q as the direction (among the 4 or 8 directions of the grid) to the new pixel.

We define as before a strictly positive potential ϕ_1 which depends on the contrast and grey-level of the pixel (eventually also on S(s,q)). It may also depend on q, in which case it depends actually on 2 pixels.

Then the iterations proceed according to the following rules :

$$\forall s, \forall q, \qquad tmp \leftarrow \quad \phi_1(s,q) + \phi_2(\alpha(V(\tau(s,q)),q)) + U(\tau(s,q)) \tag{11}$$

$$tmp < U(s) \Rightarrow \quad U(s) \leftarrow tmp \tag{12}$$

$$V(s) \leftarrow k_2 * V(\tau(s,q)) + (1-k_2) * vect(\tau(s,q), s) \tag{13}$$

The shape potential ϕ_2 is a (linear) function of the angle α between the past global direction $V(\tau(s,q))$ and the new local direction q (more precisely, the unit vector from $\tau(s,q)$ to s). $\phi_2 = k_1 * \alpha(V(\tau(s,q)),q)$ increases with the deviation of s from the line defined by $\tau(s,q)$ and $V(\tau(s,q))$, and relatively to a smoothness parameter k_1 chosen a priori. Thus, pixels prolongating the current shortest path are favored. The potential is the sum of the grey-level and contrast potential ϕ_1 and of the shape potential ϕ_2.

When the energy tmp obtained on s with respect to the decision q is smaller than the previous value of U(s), the global direction V(s) is updated as a linear combination of $V(\tau(s,q))$ and of the unit vector corresponding to the decision q. Thus, the location of s updates the global direction of the path. The coefficient k_2 of the linear combination is a memory parameter chosen a priori between 0 and 1, for instance $\frac{n}{n+1}$ where n is a constant related to the number of pixels of great influence on V, and to the weight of the new pixel. For large values of k_2, far apart pixels have more importance, that is the direction is more global. We should notice only that when $k_2 < 1$ there is also a forgetting process : after some pixels along the path, the old information becomes irrelevant, and has less influence on V(s). If k_2 is chosen equal to 1, then V is a constant on each path, which is equal to the value at the beginning of the path. If, on the other hand, k_2 is chosen equal to 0, we then obtain a similar algorithm to the one used with local curvature and two-pixels states (see Section 3.2), except that now we need less memory but lose optimality.
We impose V(s) to be a unit vector. If we did not impose it, its norm would become huge, and the weight of a new pixel at the end of the path would become neglectable.

We note that the auxiliary function remembers the global direction followed until now, so how should we define it and the potential in the neighborhood of S_0 at initialization ? We define three regions in the image : region A contains the pixels at distance $d < T_d$ from S_0, region B the pixels at distance $d = T_d$ and region C the pixels at distance $d > T_d$. T_d is

a threshold value greater or equal to one. Then, we initialize V and define ϕ in the following way (see Fig. 10) :

- in A, $V = 0$. ϕ will be equal to ϕ_1 through all the iterations.
- in B, V is the unit vector of the direction from S_0 to the point considered in B. ϕ will be equal to ϕ_1 through all the iterations.
- in C, $V = 0$. As long as V is null, ϕ will be infinite. When V becomes a unit vector, that is when a path leads from S_0 to the current state, then $\phi = \phi_1 + \phi_2$.

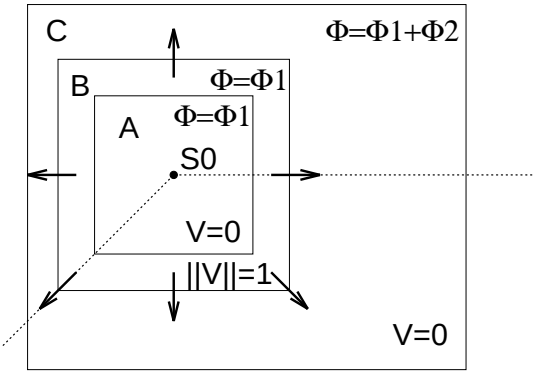

Figure 10: Initialization values for the potential and the global direction

We present in Fig. 11 the results obtained with a memory parameter of 0.1 and smoothness parameters of 10, 100, 200 and 400. In Fig. 12, 13, 14 and 15, the corresponding results are displayed with memory parameters of 0.3, 0.6, 0.9, and 0.95. From one figure to the next one, as the memory coefficient gets bigger, small zig-zags of the roads disappear, and the problem of the deviation caused by a high-contrast region on the left bottom corner of the image is solved as well (see Fig. 14 and 15, bottom right for example).

The smoothness parameter weights the relative importance of direction on one hand, and of grey-level and contrast on the other hand. The dependency upon the memory parameter is therefore stronger when the smoothness parameter is larger (bottom right in the figures). When it is small, the result fits more to the local information (images at the top left part of the figures), and the result may jump over fields for too high values (see Fig. 16, bottom right, with a smoothness parameter of 800).

In Fig. 16, we present the final auxiliary image of global direction (top, left) with a memory parameter of 0.9 and a smoothness parameter of 400. We display actually the angle between 0 and 180 degrees. The image displayed top right, is the corresponding final energy. Bottom left, the memory parameter is very large (0.99) and the initialization values produce a bias on the result.

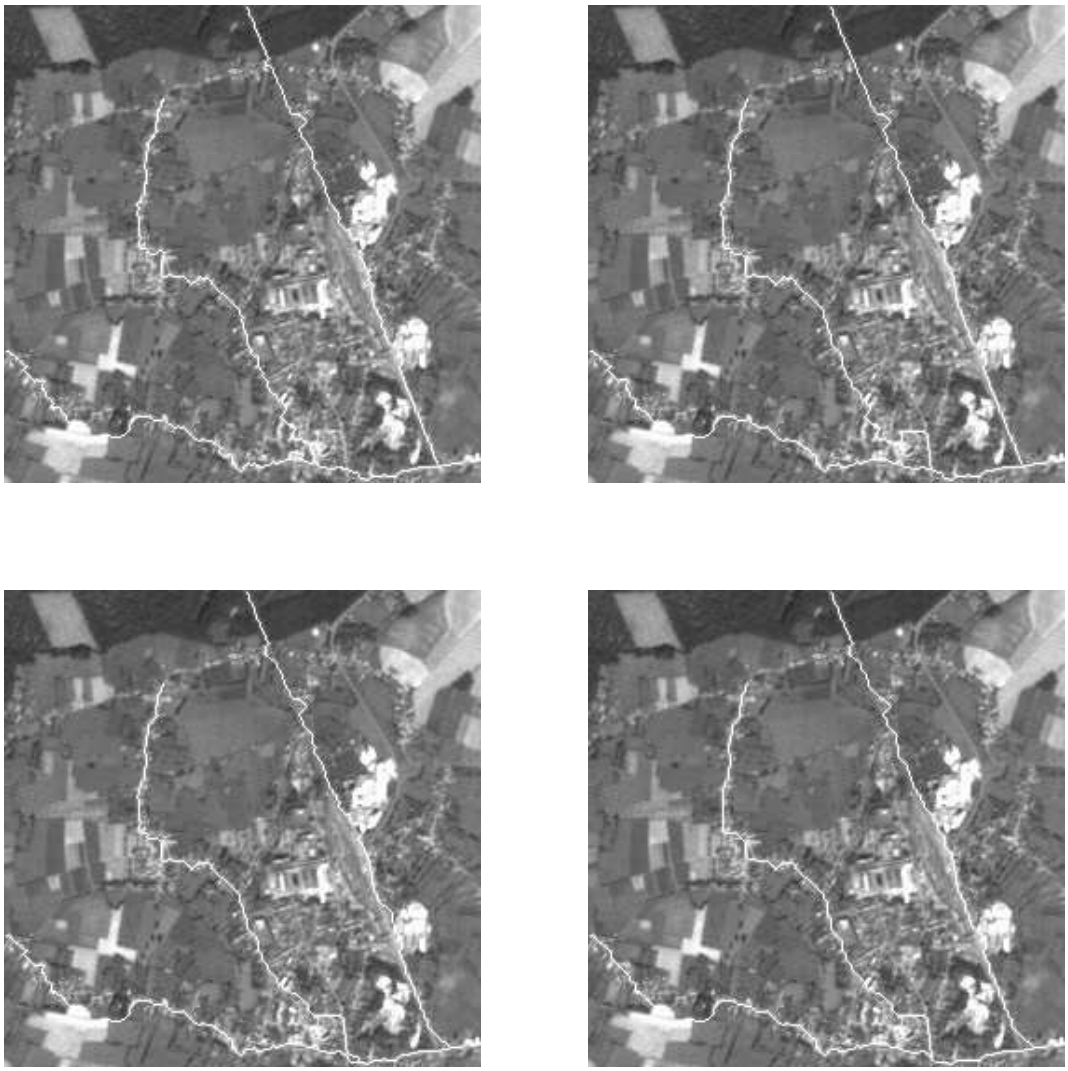

Figure 11: Global direction with coefficient of memory 0.1 and coefficient of smoothness 10 (top, left), 100 (top, right), 200 (bottom, left), 400 (bottom, right).

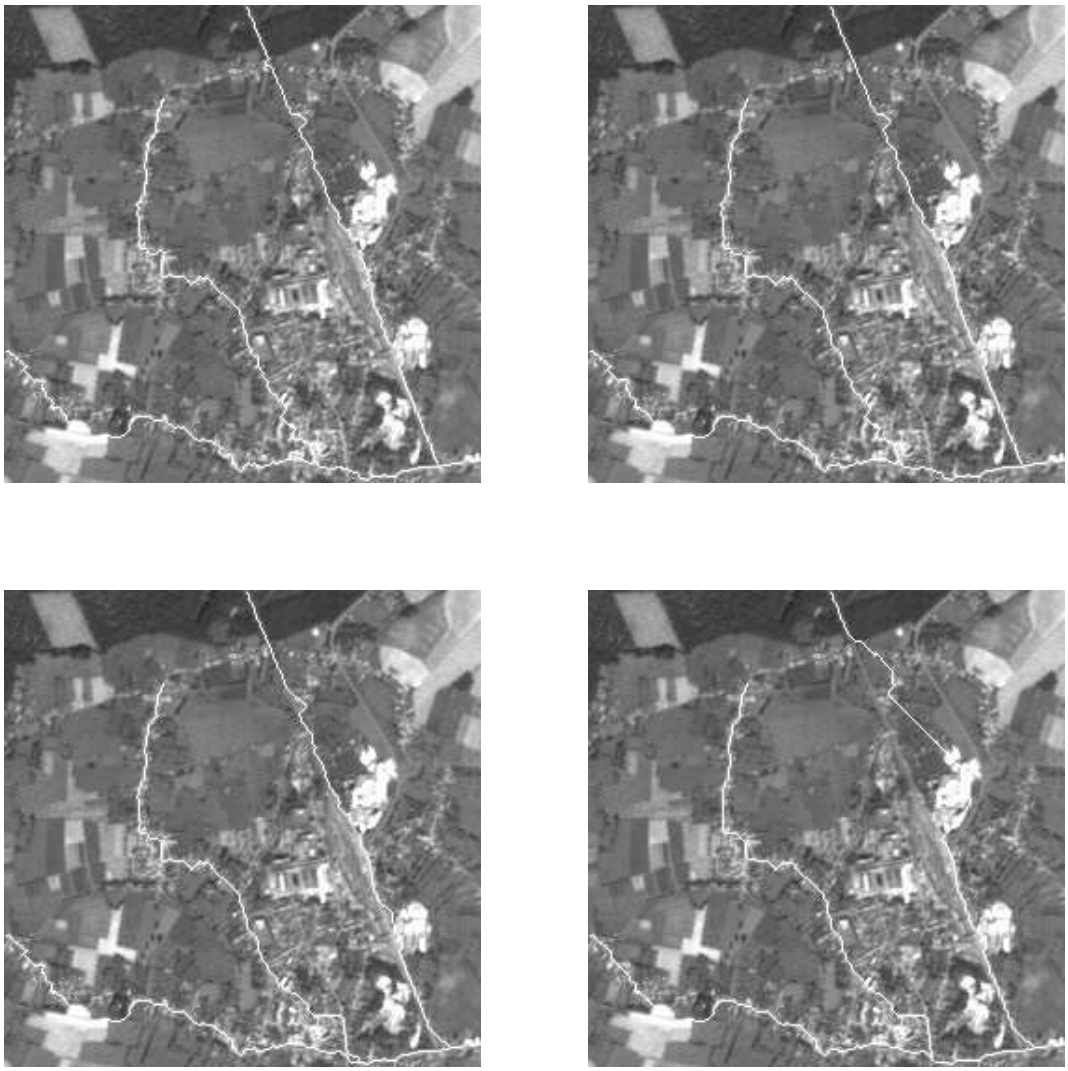

Figure 12: Global direction with coefficient of memory 0.3 and coefficient of smoothness 10 (top, left), 100 (top, right), 200 (bottom, left), 400 (bottom, right).

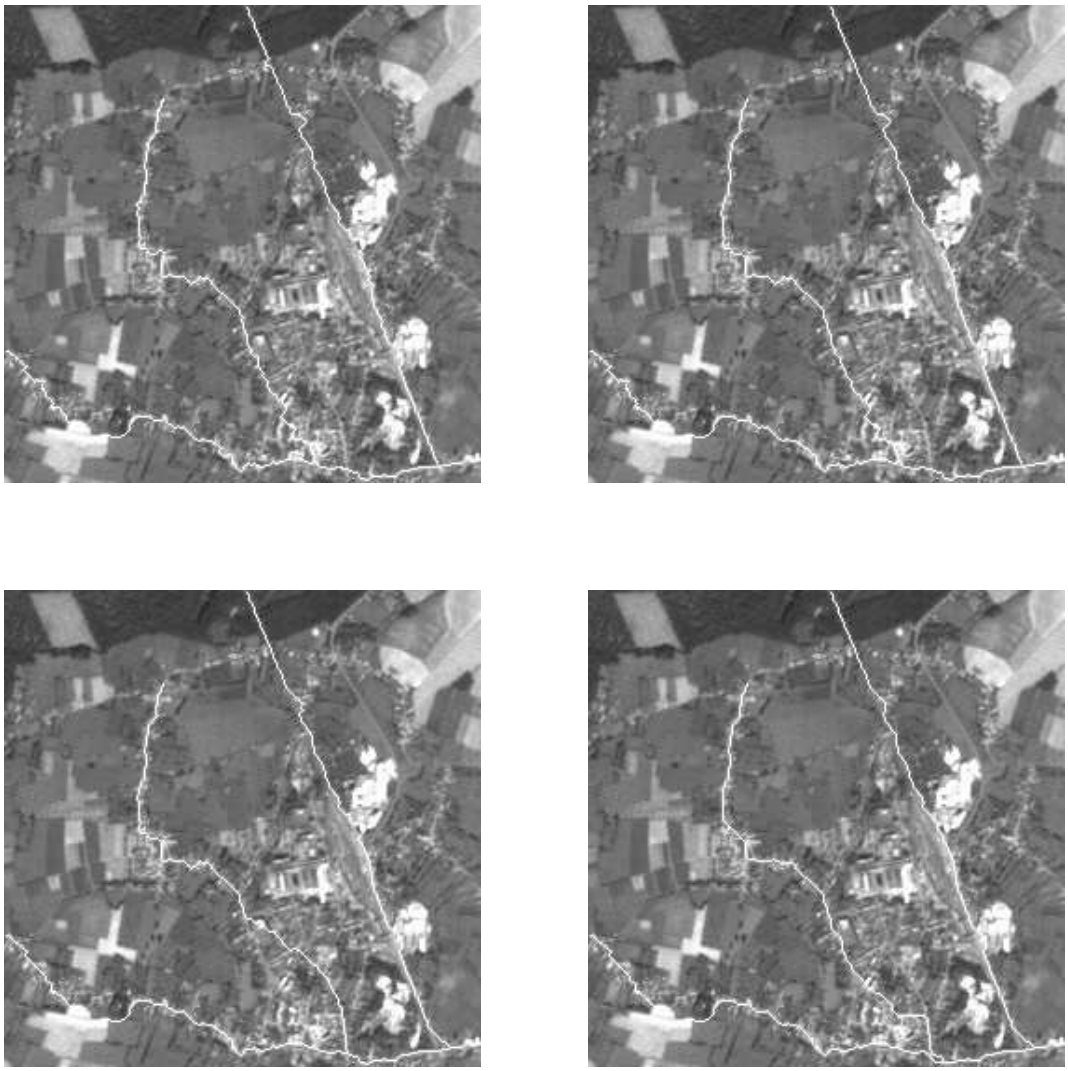

Figure 13: Global direction with coefficient of memory 0.6 and coefficient of smoothness 10 (top, left), 100 (top, right), 200 (bottom, left), 400 (bottom, right).

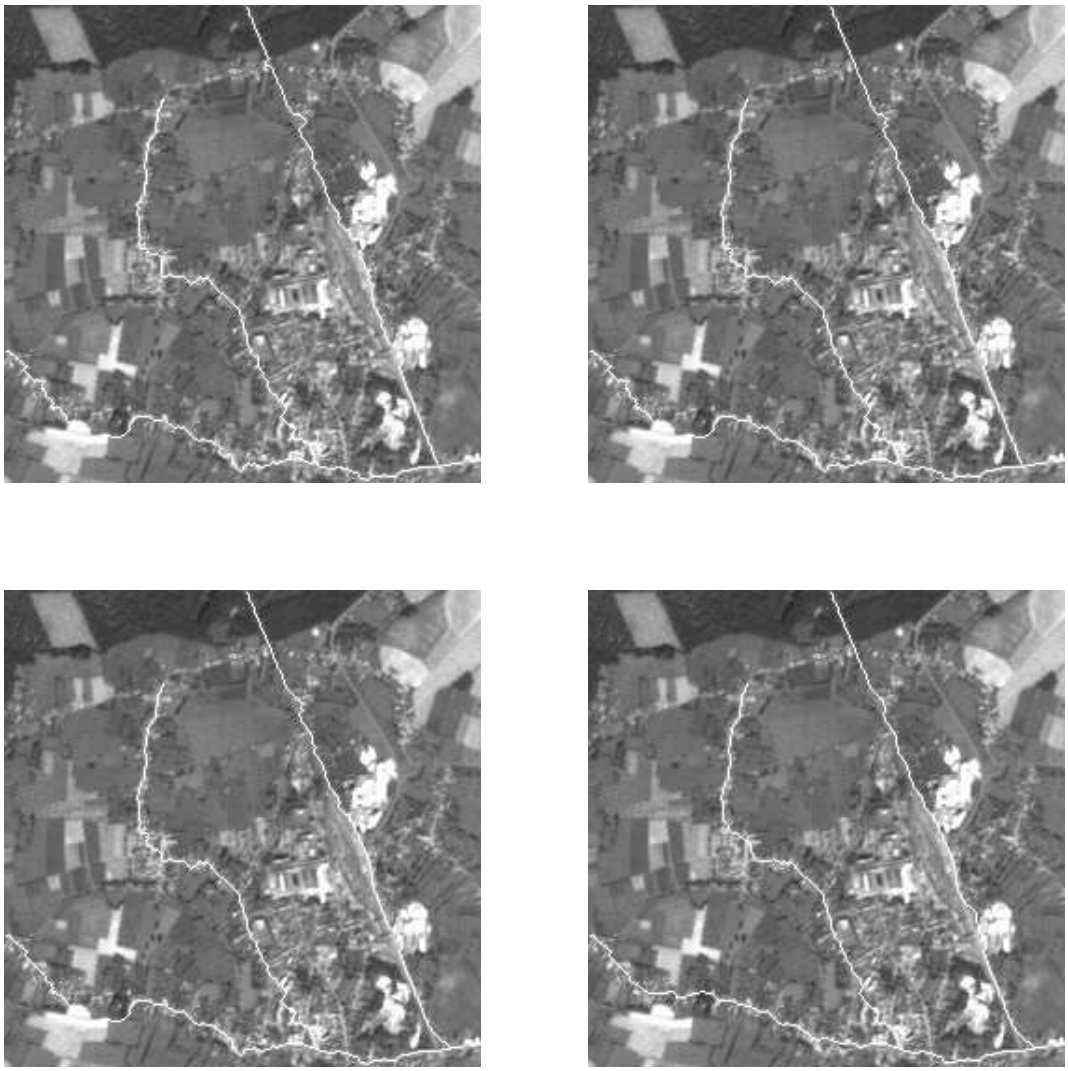

Figure 14: Global direction with coefficient of memory 0.9 and coefficient of smoothness 10 (top, left), 100 (top, right), 200 (bottom, left), 400 (bottom, right).

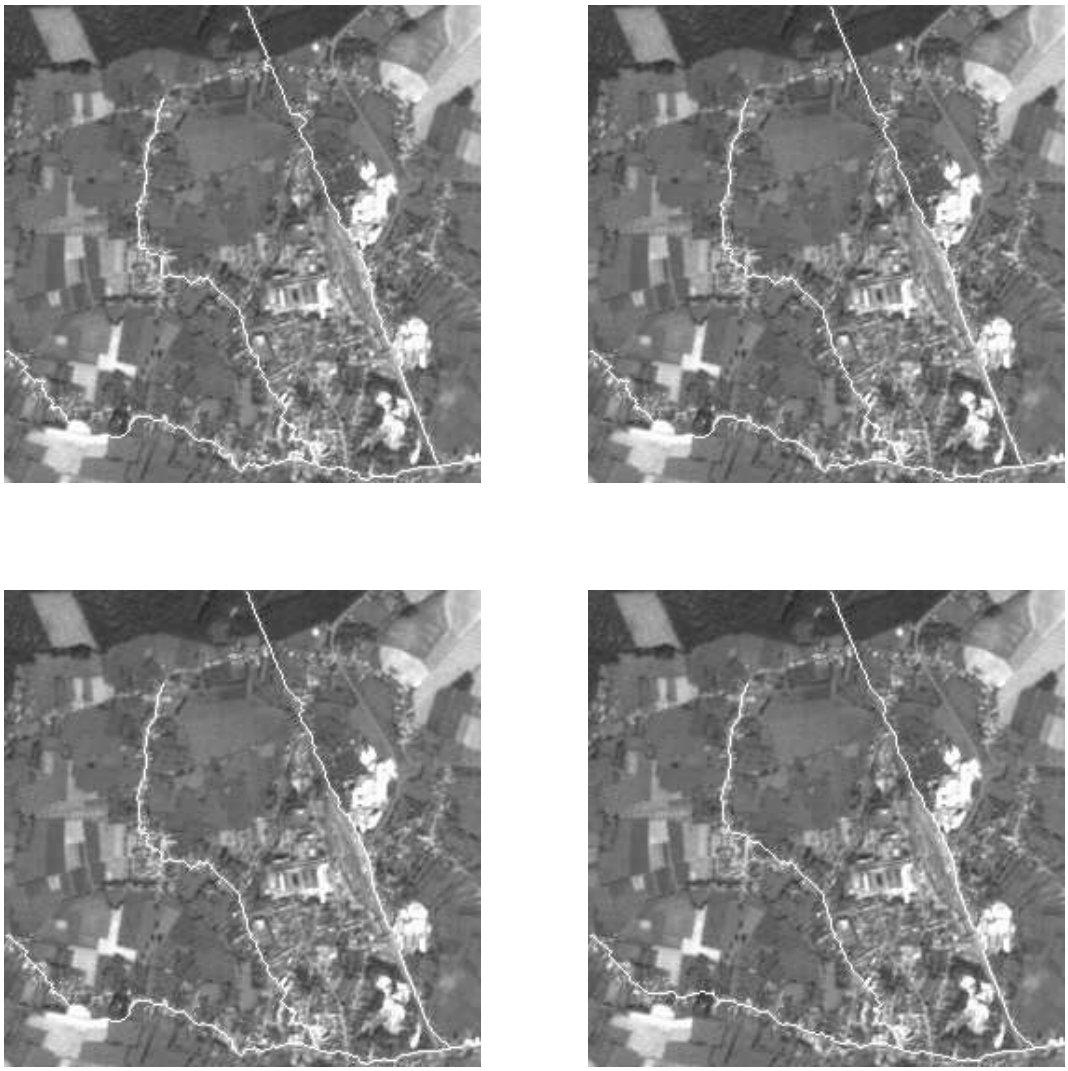

Figure 15: Global direction with coefficient of memory 0.95 and coefficient of smoothness 10 (top, left), 100 (top, right), 200 (bottom, left), 400 (bottom, right).

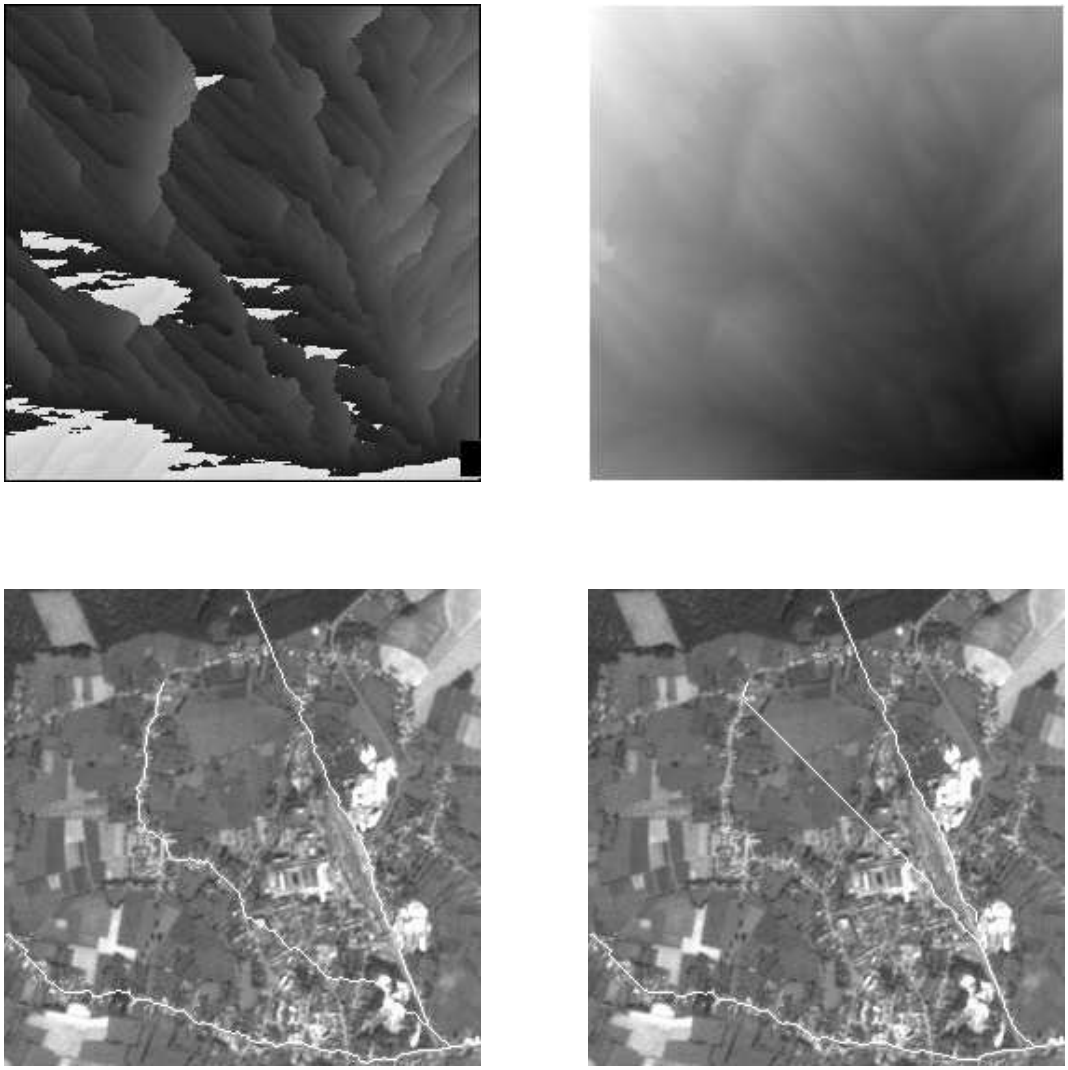

Figure 16: Final auxiliary image of global direction (top, left) and final energy (top, right) with memory coefficient of 0.9 and smoothness coefficient of 400. Result with coefficients (memory, smoothness) of [0.99,400] (bottom, left) and of [0.9,800] (bottom, right).

3.4.2 Global curvature

Instead of imposing a constraint on the global direction, we can impose a constraint on the global curvature. We then define the auxiliary function V(s) as the 2D vector $(x_r - x_s, y_r - y_s)$ where (x_s, y_s) are the coordinates of the state s and (x_r, y_r) are the coordinates of the center of curvature of the current shortest path. As previously, the state is one pixel of the image, the decision q corresponds to one of the 4 or 8 directions of the grid, and the partial potential ϕ_1 is a strictly positive function of contrast and grey-level.

The iterations proceed according to the following rules (see previous section) :

$$\forall s, \forall q, \quad tmp \leftarrow \quad \phi_1(s,q) + \phi_2(s, \tau(s,q)) + U(\tau(s,q)) \tag{14}$$

$$tmp < U(s) \Rightarrow \quad U(s) \leftarrow tmp \tag{15}$$

$$V(s) \leftarrow \psi(s,q) \tag{16}$$

The shape potential ϕ_2 is a linear function of the distance from the state s to the circle tangent to the path in $\tau(s,q)$ and is defined by $\phi_2(s, \tau(s,q)) = k_1 * (d(s, C(\tau(s,q))) - d(\tau(s,q), C(\tau(s,q))))$ where d is the euclidean distance, C(s) the center of curvature at the state s, and k_1 is a scalar smoothness parameter (see Fig. 17).

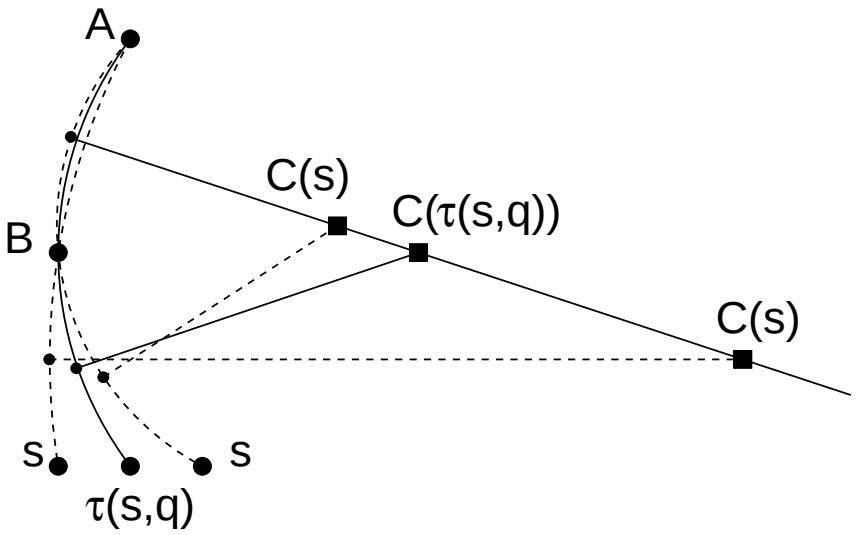

Figure 17: Updating of the center of curvature

The auxiliary function is updated according to the function ψ (see Fig. 17). We define :

- a strictly positive scalar constant k_2. It characterizes the number of pixels to consider in the current path for further interpolation.

- a point A which is obtained by following the decisions k_2 times along the current path, and a point B obtained by following the decisions $\frac{k_2}{2}$ times. They do not necessarily belong to the tangent circle in $\tau(s,q)$, if the circle has a small radius for instance.
- we define the center of curvature at s as the center of the circle passing through A, B and s. The definition of $\psi(s,q)$ follows.

We note that k_2 is actually a memory parameter whose value characterizes the level of resolution at which curvature is considered. Both k_1 and k_2 are defined heuristicly.

Since V(s) stores the global curvature, we face again the problem of initialization of the potential and of the global curvature in a neighborhood of S_0. We define in the same way as Section 3.4.1 three regions A, B, and C, and initialize V and ϕ in the following way (see Fig. 18) :

- in A, $V = 0$. ϕ will be equal to ϕ_1 through all the iterations.
- in B, $\phi = \phi_1$ through all the iterations and we consider as tangent circle to s one of the two circles tangent to the line (S_0, s) of radius equal to a high constant value (infinite in theory). The choice of the circle (out of the two orientations) is arbitrary and has no influence on the results since the radius is taken high. V(s) is then the vector from s to the center of this circle.
- in C, $V = 0$. As long as V is null, ϕ will be infinite. When V becomes non null, that is when a path leads from S_0 to the current state, then $\phi = \phi_1 + \phi_2$.

We present in Fig. 19 the results obtained with a memory parameter of 10 pixels and smoothness parameters of 10, 100, 200 and 400. In Fig. 20, and 21, the corresponding results are displayed with memory parameters 20 and 30 pixels. We notice the same phenomena as with global direction : zig-zags and sudden deviations disappear as the memory parameter gets larger. The dependency upon the memory parameter is also stronger when the smoothness parameter is larger (bottom right in the corresponding figures).

In Fig. 22, we present the final auxiliary images with a memory parameter of 30 pixels and a smoothness parameter of 400 : norma of V, that is the radius (top, left) and angle of V (top, right). The corresponding final energy is displayed (bottom, left). The image shown at the bottom, right, is the result with a memory coefficient of 30 pixels and a too large smoothness coefficient of 800 : the segmentation jumps over fields.

When comparing the results for global direction and global curvature, we can notice that in the first case the roads tend to be modeled as a succession of segments, while in the second case they look more like a succession of arcs of circles.

We would like to mention still some possible extensions :

- the memory function V(s) may be more elaborated and may contain a covariance matrix of past history. The updating rule may involve a Kalman filter in particular.

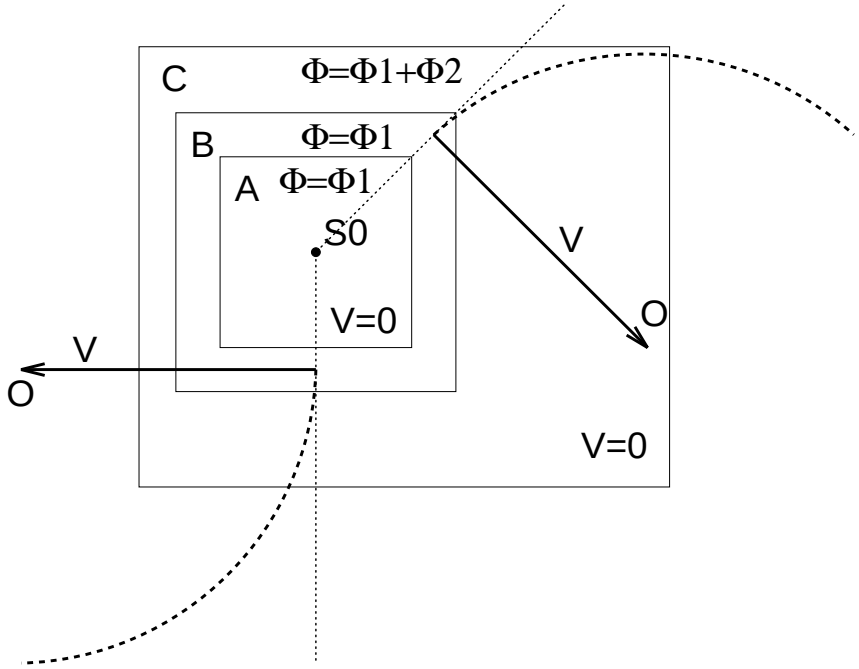

Figure 18: Initialization values for the potential and the global curvature

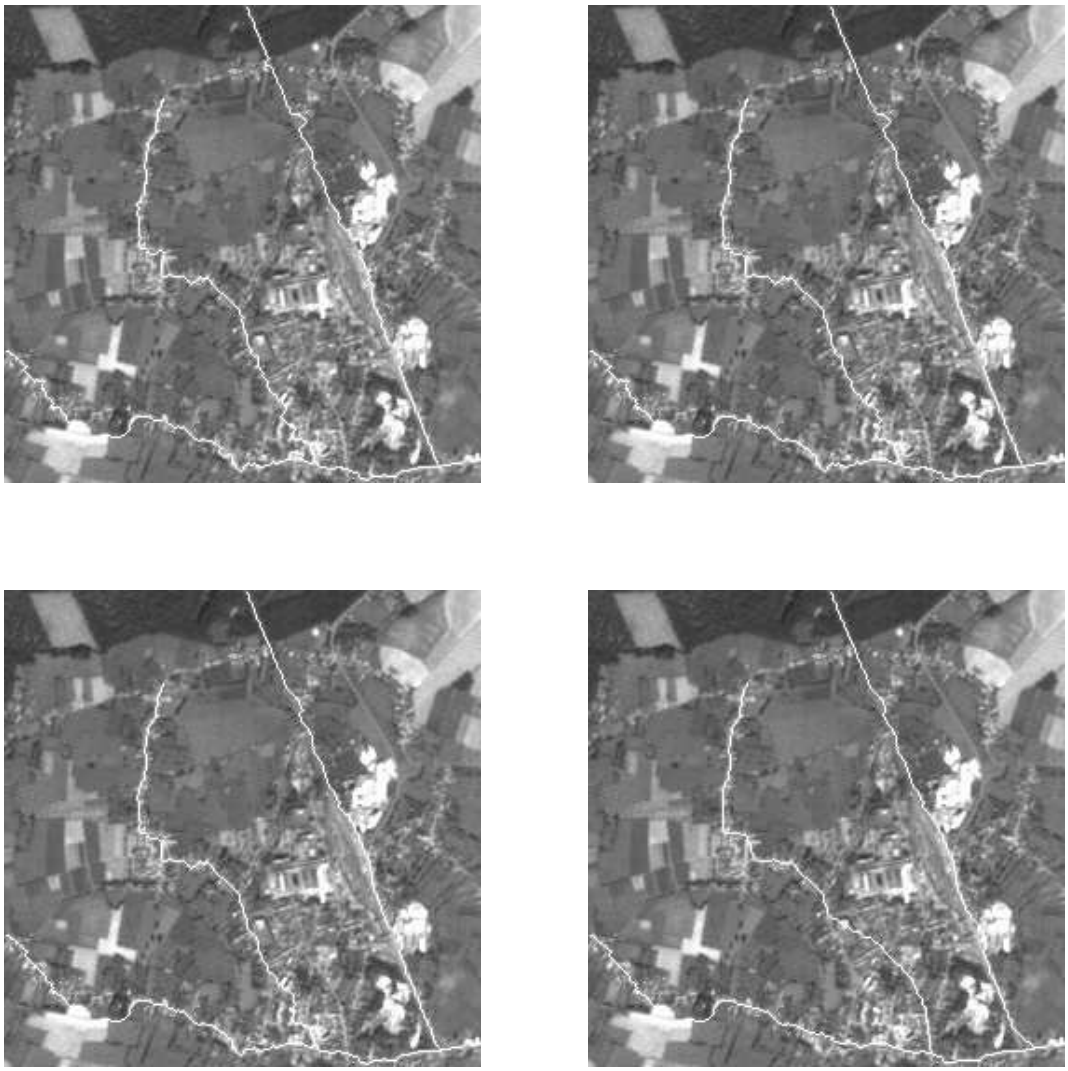

Figure 19: Global curvature with coefficient of memory 10 pixels and coefficient of smoothness 10 (top, left), 100 (top, right), 200 (bottom, left), 400 (bottom, right).

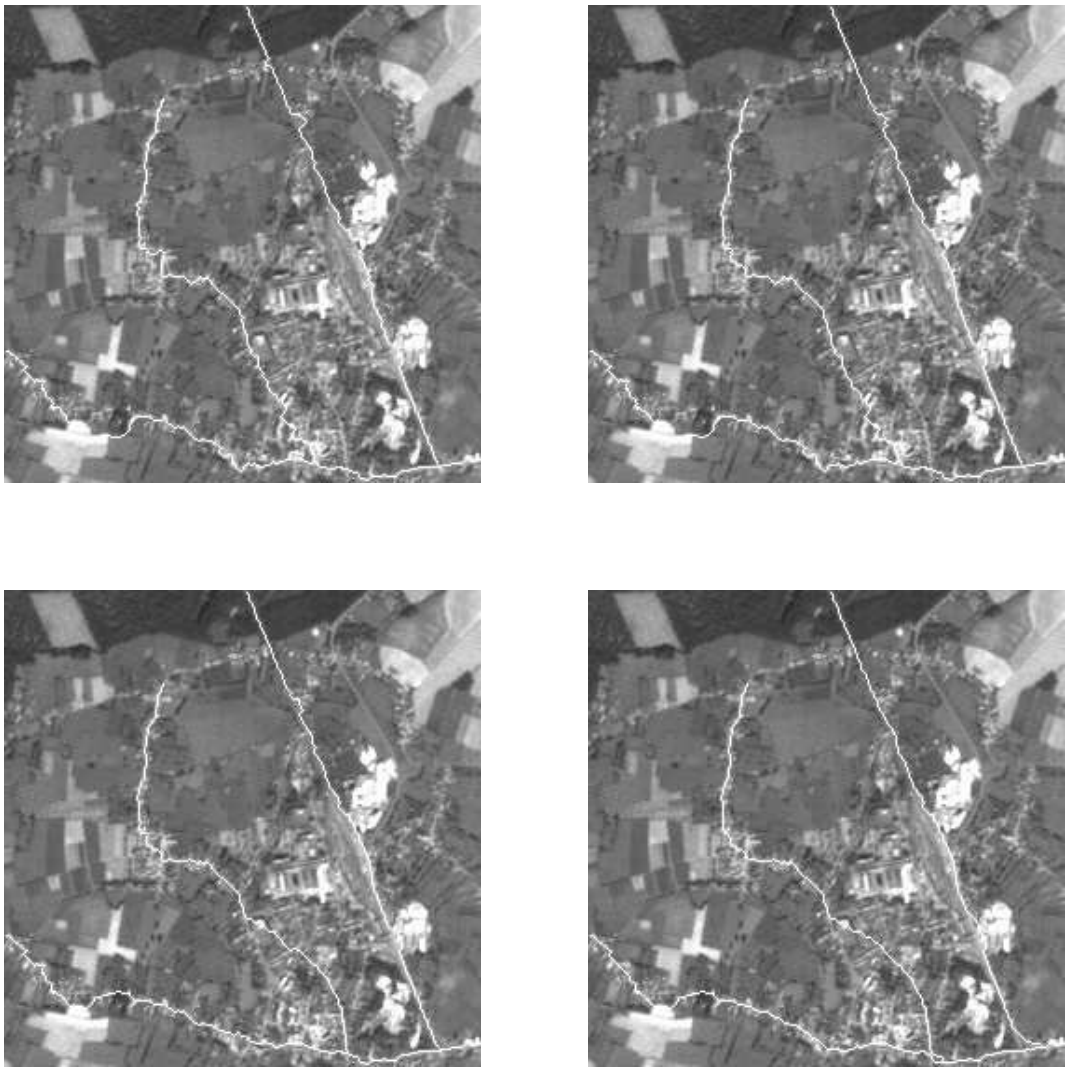

Figure 20: Global curvature with coefficient of memory 20 pixels and coefficient of smoothness 10 (top, left), 100 (top, right), 200 (bottom, left), 400 (bottom, right).

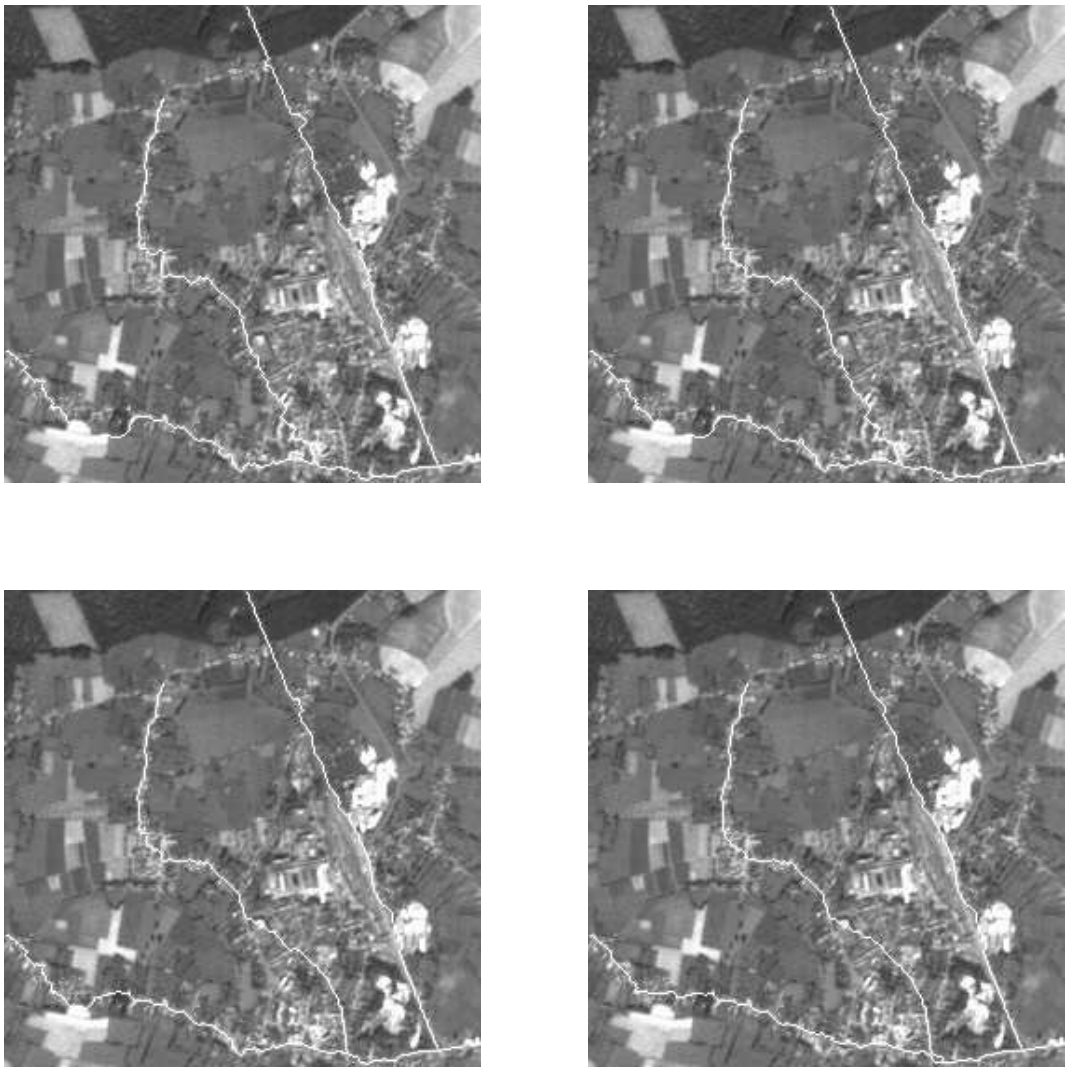

Figure 21: Global curvature with coefficient of memory 30 pixels and coefficient of smoothness 10 (top, left), 100 (top, right), 200 (bottom, left), 400 (bottom, right).

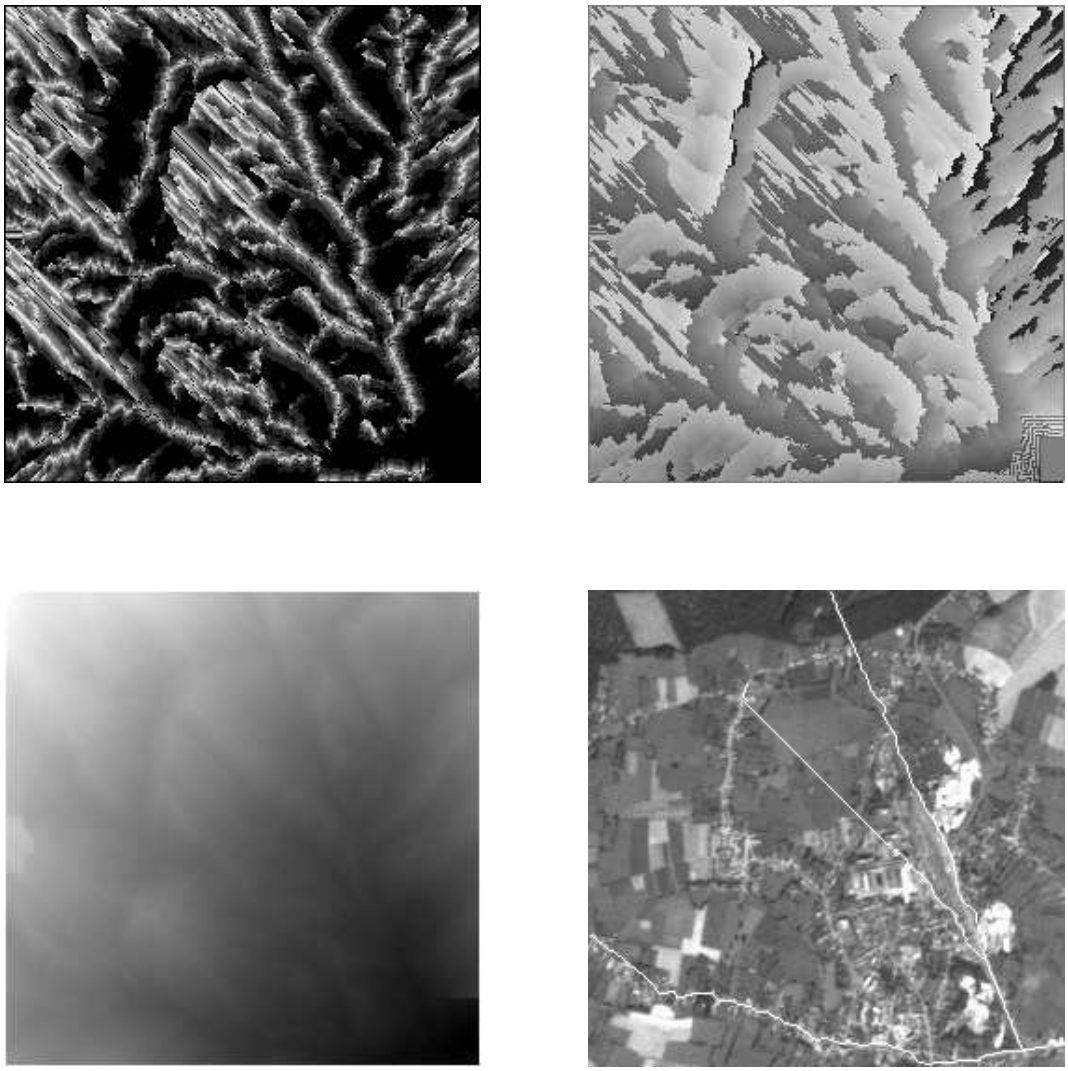

Figure 22: Final auxiliary images of norm of radius (top, left) and of direction of radius (top, right) with memory coefficient of 30 pixels and smoothness coefficient of 400, and final energy (bottom, left). Result with memory coefficient of 30 pixels and smoothness coefficient of 800 (bottom, right).

In this case, old states would keep the same influence (there would be no forgetting process), and every past state would have the same influence on the current state.

- when the characteristics of the feature are standard and known, this information may be used in defining $\phi(s, q, V(\tau(s, q)))$.

4 DEFINING THE POTENTIAL

4.1 Interactive potential

When the images are complicated and when the road looks very confuse in the background, the potential or cost function must sometimes be determined interactively. This is the case in particular for dense urban areas.

A supervised method of finding a potential is to compute first the automatic potential (see Section 4.2), to find in the result which parts of the roads were badly detected, and to correct the potential in order to favor the characteristics of these parts of the roads, for instance by increasing the contrast potential for low contrasts. Several tests may be performed.

Another interactive method is to proceed directly :

- What are the characteristics of the roads, for instance the grey-level and contrast ?
- What are their common values on the roads, on the background, and with which frequency ?
- In order to limit the complexity of the search, it is possible to assume that the contrast and grey-level are independent variables and to define the potential as an arbitrary expression such as the product of a grey-level function and a contrast function for instance.
- Further constraints may be imposed. The functions may be assumed continuous if there is no gap between the values on the road and in the background. Considerations of storage and implementation may also limit the scale of the functions. The ratio between the weight of different characteristics, such as contrast and grey-level, is a delicate topic, although a possible solution is presented in Section 4.2 with the automatic potential.
- Extreme configurations may be selected on the road to be detected, and it is possible to analyze for instance the length of an interruption of a road, the maximum deviation in direction or curvature and the frequency of such deviations.

In Fig. 23, the sea in white is disturbing the detection of the roads in its neighborhood, since it corresponds to low values of the potential (high contrasts and grey-level values). Therefore, we have used a binary mask corresponding to the sea, and given to the potential an infinite value there. The potential is interactively chosen, but the channel on the right top side of the image disturbs the detection of the neighboring road, due to its high contrast.

In Fig. 2 (see Section 3.2.1.), the potential has been manually selected, and the roads were correctly detected.

Figure 23: Original SPOT image (left). Result superimposed on the original image (right), with an interactive potential.

4.2 Automatic potential

Determining a different potential heuristicly for each image is not very efficient, therefore we propose an automatic method for computing it. The underlying idea is that the potential should favor values of the grey-level and contrast which are likely to appear on the feature. We notice that :

- the potential should be low for such values.
- the conditional probability for a pixel to belong to the feature, given its grey-level (or contrast) value, is high w.r.t. values which are likely to appear on the feature, and characterizes them.
- the conditional probability may be easily evaluated from histograms.

Therefore, we define the potential as a decreasing function of the conditional probabilities of contrast and grey-level [14].

We assume that the only information available on the images is the set S of starting points (S_0) and of goal points (S_1). This is why we have to assume in the following that the values of the conditional probabilities in a neighborhood of the extremities are a good estimation of the values in the whole image, which is obviously not true in a lot of cases.

We propose to exploit the information around these points in three steps :

1. *From the windows to the partial segments* : we find segments of lines around the points of S_0 and S_1, where the information is the most relevant, since a line is there for sure. We use for this a general and approximative potential and apply the F* algorithm in small windows.

2. *From the partial segments to the histograms* : we then compute the histograms of grey-level values and contrasts on the partial segments.

3. *From the histograms to the potential* : we smooth the histograms and define the potential as a function of the histograms.

In this way, we integrate global information according to the first method : relatively to the characteristics of a global feature, the partial segments.

Until now, we have only shown why it is worth to define the potential as a decreasing function $\phi(p)$ of the conditional probability. We present yet a justification for the function $\phi(p)$ that we have chosen.

4.2.1 Necessary form of the potential

Without loss of generality, we only consider the potential on one point and as a function of the grey-level. We consider the case where the image contains two extremities S0 and S1, and only two paths joining them. We suppose that the rest of the image contains non relevant values (corresponding to an infinite potential for instance). Furthermore, we also assume that the first path only contains two points of conditional probabilities p_1 and p_2, that these probabilities are independent, and that the second path only contains one point of conditional probability $p_3 = p_1 * p_2$ (in addition to the two extremities S0 and S1 common to both paths, see Fig. 24).

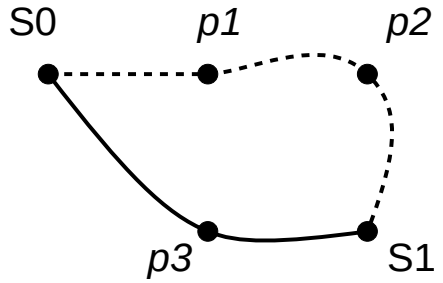

Figure 24: Two paths of same probability

The probability that the road passes through one or the other path is the same, equal to $p_1 * p_2$. Therefore, the sum of the potentials should also be the same on both paths.

We get : $\phi(p_1 * p_2) = \phi(p_1) + \phi(p_2)$. We write ϕ as a function g of the logarithm of the probability : $\phi(p) = g(log(p))$ and obtain : $g(log(p_1 * p_2)) = g(log(p_1)) + g(log(p_2))$, therefore : $g(x+y) = g(x) + g(y)$ for any couple (x,y) of real numbers. We deduce that $g(x)$

is linear in x and $\phi(p) = a * log(p)$. Since ϕ decreases w.r.t. p, a should be negative.
This means that the energy is actually a multiplicative function of the properties of the pixels, as proposed by [1] and [19].

If only the grey-level is considered (or only the contrast) we may take a of norm 1, and we obtain $\phi(p) = -log(p) = log(1/p)$.

If we consider both grey-level and contrast, we obtain both $\phi(p) = a_g * log(p_g)$ and $\phi(p) = a_c * log(p_c)$. We deduce : $\phi(p) = a * log(p_g) * log(p_c)$ and this time we may take a equal to one (due to the sign of the logarithm), and we obtain : $\phi(p) = log(p_g) * log(p_c)$.

This argument applies with other criteria related to conditional probabilities, one should only be careful to take $a = (-1)^n$ where n is the number of criteria used. When working on multi-spectral images, one may consider for instance the grey-levels and contrasts in each of these bands.

There is a fundamental problem when the potential depends upon characteristics of different nature such as grey-level and contrast : how to weight them ? The use of conditional probabilities is a solution : the characteristics may be compared and weighted properly since their probabilities may be compared.

Let us look now in more details at the three steps of the method. In the following sections, we indifferently consider points of S_0 or of S_1 and call their union S.

4.2.2 From the windows to the partial segments

We define the windows as rectangles of points which are at a distance not greater than 10% of the size of the image from S (25 pixels in our images) :

- there is a line in each of these windows, since they contain a point of S, and we get enough information to find the initial segment
- the chance for getting a second line in the same window is low if we assume that the points of S were not chosen close to each other
- the computing time for finding the partial segments is neglectable (less than 1 % of the total computing time).

Moreover, we reduce the scale of grey-level values to 20% of the original scale, in order to get more data in each level of the histograms and therefore to be more robust (this step is not necessary when we approximate the probability by a Gaussian function).

We then define partial potentials P_{pg} and P_{pc} depending respectively upon the grey-level and the contrast. We have seen a justification for defining the potential of grey-level and contrast as $\phi(s) = \log(p_c) * \log(p_g)$ where p_c and p_g are the conditional probabilities of contrast and grey-level. If we approximate these probabilities by Gaussian functions, we obtain $p_g = \exp(\frac{(g-\overline{g})^2}{\sigma_g^2})$ and $p_c = \exp(\frac{(c-\overline{c})^2}{\sigma_c^2})$. Therefore $\phi(s) = (g-\overline{g})^2 * (c-\overline{c})^2$ since the linear coefficient is irrelevant in the computation.

However, the results obtained experimentally are better when taking $\phi(s) = |g - \overline{g}| * (c - \overline{c})^2$ instead of the above mentioned function This is probably due to the fact that in these windows (with points of S selected manually), the contrast information is more reliable than the grey-level one. The experiments were performed on 27 windows in 5 images with different contrasts and grey-levels, and the results with this last form of the potential were exact.

We face the problem of evaluating $\overline{g}$ and $\overline{c}$ from very little information. We define g_{max}, g_{min} and g_{moy} as the maximum, minimum, and average grey values over all the windows, we call g_s the grey level of any point of S, and consider three configurations :

- if the grey levels g_s of the points of S are always higher than g_{moy} (clear features) : $\overline{g} = g_{max}$. That is, we assume that the maximum grey-level is reached for a pixel of the feature.
- if the grey-levels g_s are smaller than g_{moy} (dark features) : $\overline{g} = g_{min}$.
- if the values g_s may be higher or smaller than g_{moy} : $\overline{g} = g_s$, where g_s is the grey-level of the element of S and changes in each window.

We define the contrast c as the morphological gradient, which is the difference between maximum and minimum in a neighborhood of size one, and $\overline{c}$ as its maximum value in the considered window.

We then integrate the potential $\phi(s)$ with the F* algorithm. We look for the other extremity of the segment in each window and define it as the point on the opposite side of the window where the energy is minimum. In order not to favor trivial solutions (paths containing the minimum number of pixels), we store in a separate image the number of pixels of each path, and compare on the opposite side of the window the average potential, instead of the sum (see Section 5.1). We then propagate from these extremities and get the complete partial segments.

4.2.3 From the partial segments to the histograms

The conditional probability $P(M \in segment | g_m = g)$ characterizes the probability for a pixel to belong to the segment given its grey-level. By definition :

$$P(M \in segment | g_m = g) = \frac{P(M \in segment \wedge g_m = g)}{P(g_m = g)} \tag{17}$$

Thus, we may easily compute an approximation of this probability :

$$P(M \in segment | g_m = g) \sim \frac{|M \in segment \wedge g_m = g|}{|g_m = g|} \tag{18}$$

where $||$ denotes the number of elements in a set.

As a consequence, the conditional probability is obtained by computing the histogram of the segment H_l and the histogram of the image H_i, and by dividing H_l by H_i. We apply the same steps for getting the histogram H_c of the contrast.

4.2.4 From the histograms to the potential

We obtain an histogram which is not smooth, due to the small number of points considered. We approximate it by a Gaussian function, this provides the following advantages :

- the curve is smooth.
- there are no zero values.
- the curve has an asymptote at the extremities, where variations are less significative.
- the curve is characterized by only two parameters (m,σ)
- the result is independent from the sampling in grey-level.
- when only one criterion is used (grey-level for instance), the expression of the potential simplifies to least-mean squares optimization.

The grey-level histogram is approximated by a single Gaussian function. The contrast histogram is approximated by two Gaussian functions separately for positive and negative contrasts, and the values of the potential at the extremities, where the contrast is high, are defined as the minimum values on each side.

Approximation is done in a simple way. We suppose that we have a function $p(x)$ and we want to approximate it as : $t(x) = K.\exp(-\frac{(x-m)^2}{2.\sigma^2})$. We look for the values of K, m, and σ.

- We define m as $m = \int x.p(x)$ on the considered interval.
- We define σ by $\sigma^2 = \int (x-m)^2.p(x)$ on the considered interval.
- We compute the coefficient K by least-mean-squares optimization ; we minimize $f(K,m,\sigma) = \sum(p(x) - t(x))^2$. The equation $\frac{\partial f}{\partial K} = 0$ implies $\sum(p(x) - t(x)).\exp(-\frac{(x-m)^2}{2.\sigma^2}) = 0$ and therefore : $K = \frac{\sum p(x).\exp(-\frac{(x-m)^2}{2.\sigma^2})}{\sum \exp(-\frac{(x-m)^2}{2.\sigma^2})^2}$.

We have experimentally found that the potential $\phi(s) = \log(p_c) + \log(p_g)$ gives better results than $\phi(s) = \log(p_c) * \log(p_g)$.

As far as curvature is concerned, the smoothness of the segmented features is a subjective criterion which depends on the considered application, therefore it should be chosen interactively.

4.2.5 Results

We present herein, for each experiment, the original grey-level image, the result superimposed on the original image, and the potentials of grey-level and contrast as they are represented at logarithmic scale by the ELIESER software [15]. The results are good for Fig. 25 and Fig. 26.

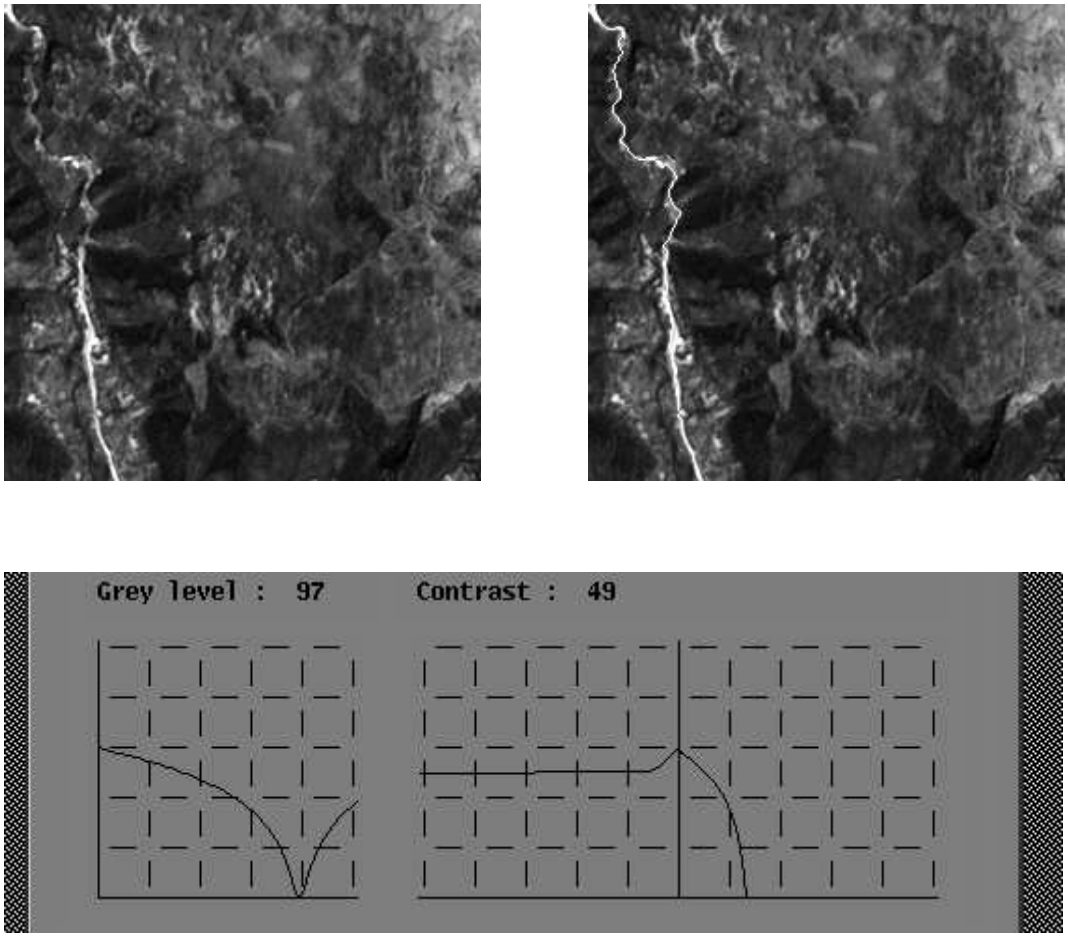

Figure 25: Original image (top, left). Result (top, right). Potentials (bottom).

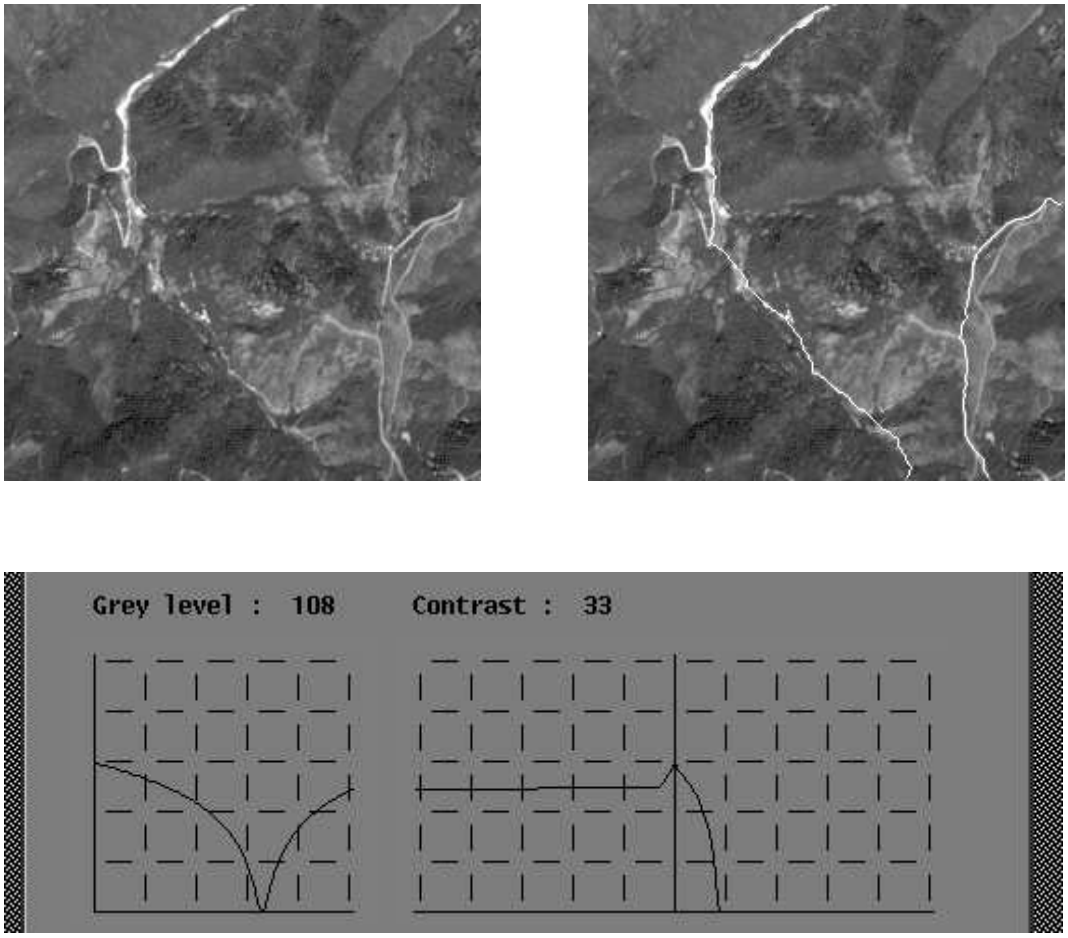

Figure 26: Original image (top, left). Result (top, right). Potentials (bottom).

In Fig. 27 and Fig. 28, we face the problem of jumping between close features and of disturbance by a region of high contrast. The curvature does not help in this case, maybe because the automatic potential does not favor enough high contrasts. The corresponding results with an interactive potential were shown in Fig. 7 and Fig. 6.

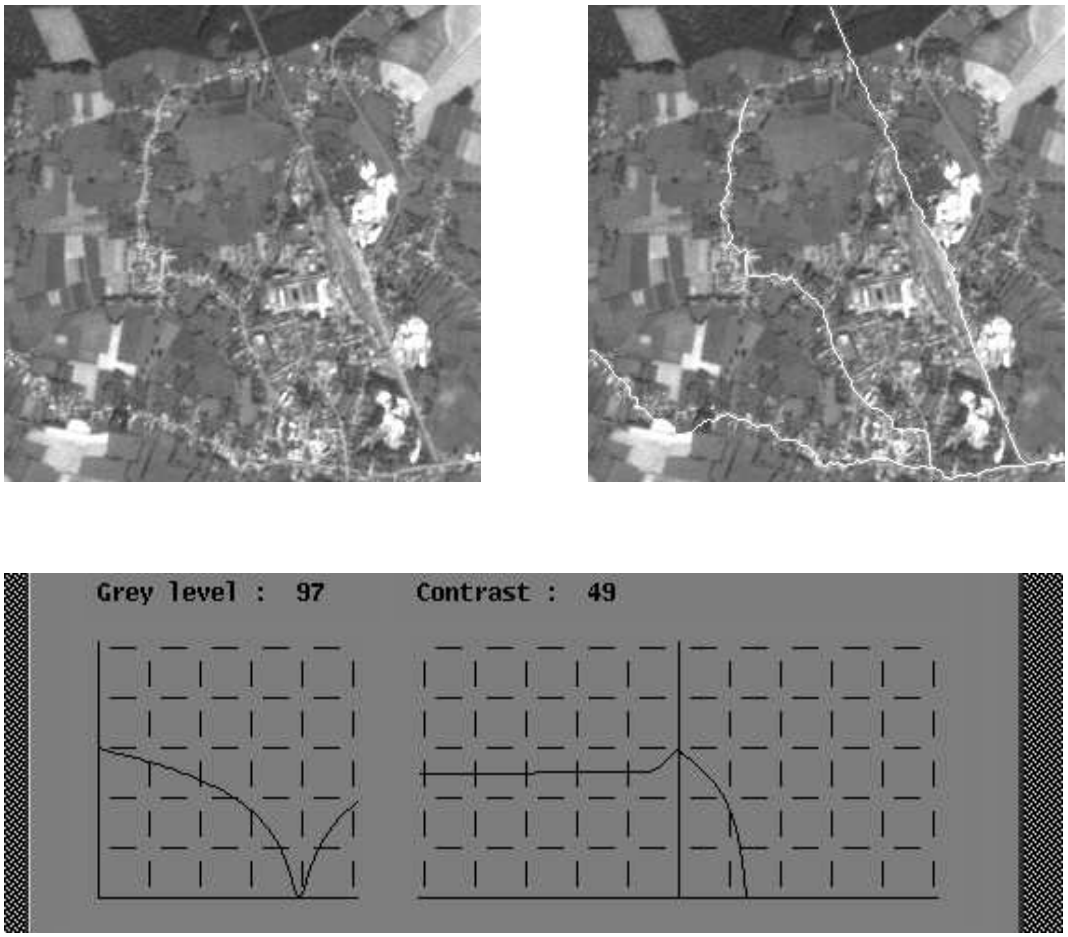

Figure 27: Original image (top, left). Result (top, right). Potentials (bottom).

Finally, we present in Fig. 29 and Fig. 30 the results for urban areas. Part of the roads are correctly obtained, but the detection is wrong when there is a high textured region or when the road is too much blurred. One may notice the negative contrast in Fig. 29 and in

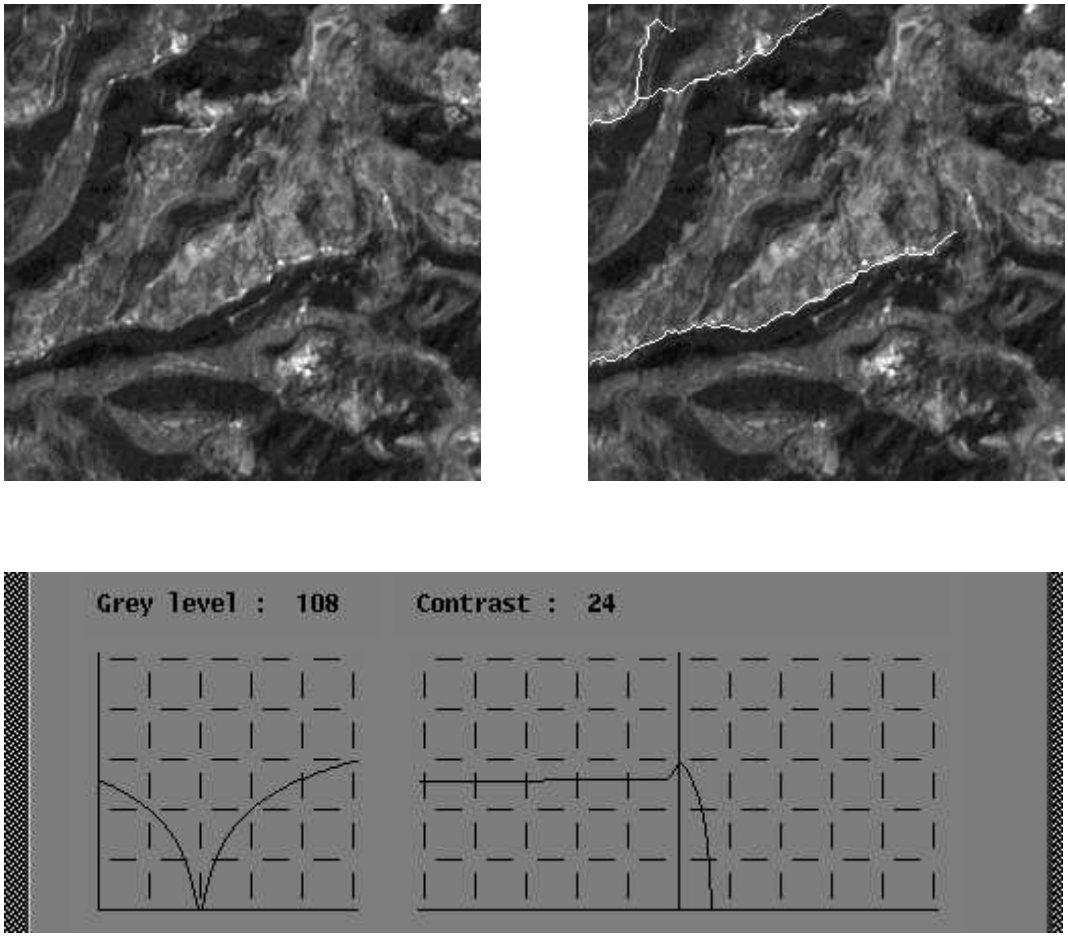

Figure 28: Original image (top, left). Result (top, right). Potentials (bottom).

the potentials. The corresponding results with an interactive potential were shown in Fig. 2 and Fig. 23.

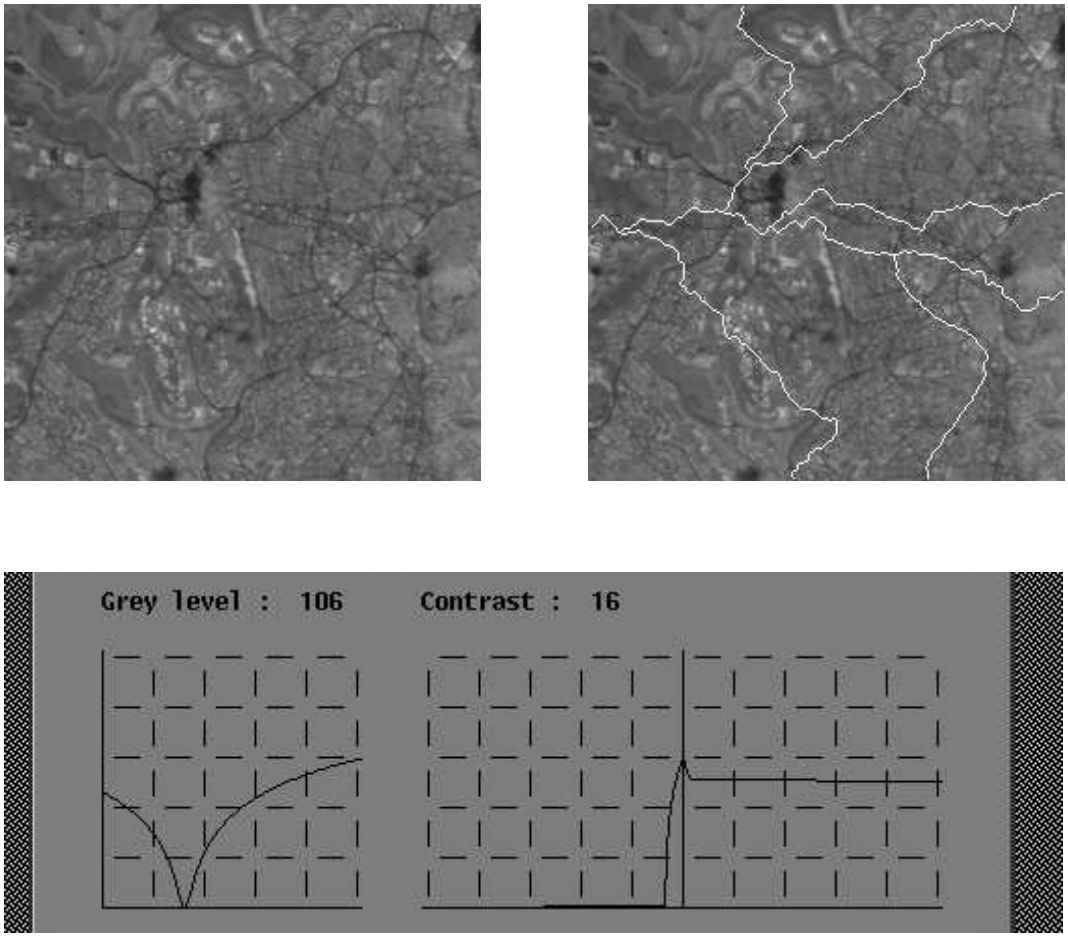

Figure 29: Original image (top, left). Result (top, right). Potentials (bottom).

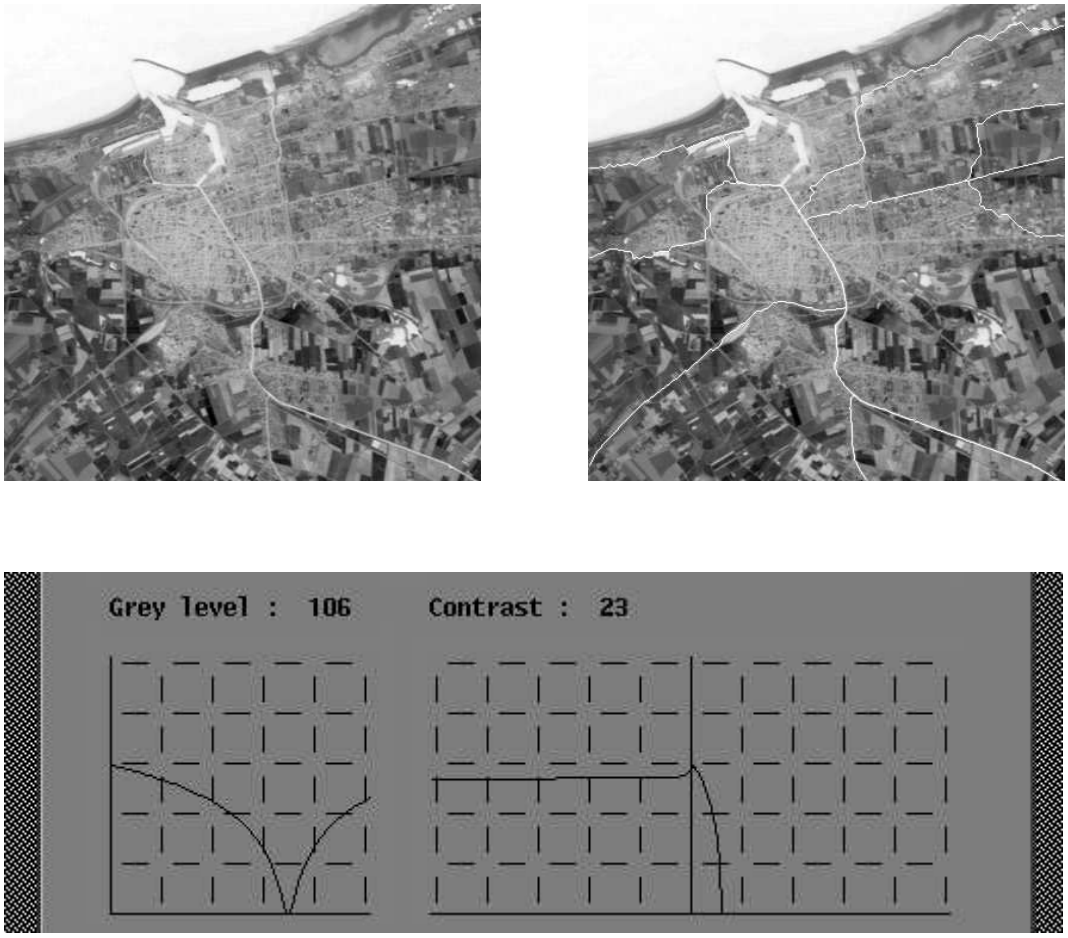

Figure 30: Original image (top, left). Result (top, right). Potentials (bottom).

5 INTEGRATING THE POTENTIAL

After focussing in Section 3 on global information, we want herein to study various ways of integrating the potential.

5.1 Bias of the shortest path

A well known problem of dynamic programming is the bias of the shortest path [7]. Until now, we have defined the energy as the sum of the potentials along the path, therefore it depends not only upon the values of the potential but also upon the number of terms in the sum : shortest paths in the meaning of the euclidean distance tend to be favored, although they do not necessarily correspond to the feature which we look for.

This problem is particularly annoying during the computation of the automatic potential (see Section 4.2.2). There, we look for segments with "good" contrast and grey-level in windows of size 25-50 pixels. S_0 is one extremity of the segment, but the other extremity has to be defined as the minimum of the energy on the opposite border. Unfortunately, with so few pixels, the path found is often the one containing the minimum number of pixels. Therefore, we actually do not look for the minimum of the energy but for the minimum of the ratio between energy and length of the corresponding path. Equation (3.2) then becomes :

$$\forall s, \forall q, \qquad tmp \leftarrow \qquad \phi(s,q) + U(\tau(s,q)) \tag{19}$$

$$tmp < U(s) \Rightarrow \quad U(s) \leftarrow tmp \tag{20}$$

$$V(s) \leftarrow V(\tau(s,q)) + 1 \tag{21}$$

The image V is initialized at 1 for points of S_0 and at $+\infty$ everywhere else. Actually, nothing is changed in the computation of the energy, and the only difference with Equation (1.4) is that we store the length of the current shortest path. We do not use this information during the iterations but only once the energy has converged, for finding the other extremity of the segment.

However, this information may also be used for solving the bias of the shortest path. We may define the energy as the average of the potential along a path instead of the sum, and shortest paths are not favored anymore :

$$U = \frac{\sum_{s=S_0}^{S_1} \phi(s,q)}{\psi(ns)} \tag{22}$$

where ns is the number of states of the path considered. ψ is a real strictly positive function of the number of states ; it may be the identity for instance. Equation (3.2) becomes using

an auxiliary function :

$$\forall s, \forall q, \qquad tmp \leftarrow \qquad \frac{1}{\psi(V(\tau(s,q))+1)} * \phi(s,q)$$
$$+\frac{\psi(V(\tau(s,q)))}{\psi(V(\tau(s,q))+1)} * U(\tau(s,q)) \tag{23}$$
$$tmp < U(s) \Rightarrow \quad U(s) \leftarrow tmp \tag{24}$$
$$V(s) \leftarrow V(\tau(s,q)) + 1 \tag{25}$$

and we initialize the auxiliary function V i.e. the number of states of the current shortest path with :

$$V(s_0) = 1, \forall s \neq s_0, V(s) \leftarrow +\infty \tag{26}$$

There are several theoretical problems :

- the optimality principle is not verified, as in Sections 3.4.1 and 3.4.2.
- it is necessary to impose the decreasing of U with this formulation, since tmp may increase with time for some configurations where $V(\tau(s,q))$ decreases (see Appendix A).
- there may be loops (see Appendix A), and the minimum of the energy may be never reached. We solve this problem by scanning only part of the couples (state, decision). In the considered examples, it was enough to take only half of the decisions, the ones belonging to a half plane : this half plane is defined by the line perpendicular to $S_0\vec{S}_1$ and by the orientation $S_0\vec{S}_1$. Of course, such a simple restriction was possible in these examples because the roads have a constant global direction, there is no "return". With these strong restrictions, loops were impossible.

In Fig. 31, we display the results obtained with different functions ψ : the identity, the square root, the cubic root and the logarithm (in increasing order of influence of the bias of the shortest path). We notice that :

- there is a better fitting to the pixels of "good" characteristics of contrast and grey-level.
- the paths found are longer than without compensating the bias. On one hand, the effect is inverse to using global direction or global curvature. On the other hand, one may consider the definition of the energy as the sum (and not as the average) as a way of imposing a smoothness constraint. Let us recall that [6] suggested to use a power of the potential for enforcing more smoothness.
- these two last points are more visible when the averaging function ψ has a higher complexity with respect to the length of the path. There is no meaning to use a function of complexity higher than the identity since then paths which are longer w.r.t. the euclidean distance would be favored.

- pixels near S_0 have much more weight than pixels farther, since the weight of the potential in Eq. 5.5 is the inverse of the length of the path. That is, after a certain number of pixels along a path, the choice of a new pixel tend to be indifferent, the average of the potentials of previous pixels becomes overweighting, and pixels with great potentials may be jumped over. The non-optimality of the method is obvious, and impairs the quality of the results. This may be seen in particular in the final energy obtained, on Fig. 32, top-left : the energy has a stronger contrast near S_0 than farther.
- the auxiliary function in Fig. 32, top-right, is the length of the shortest path from any pixel. It gives simple information on the propagation of the energy, which may be used for instance in a second step for redefining S_1 in order to select new pathes. Both the energy U and the auxiliary function V displayed here were obtained at convergence, and with the identity as choice for ψ.

When we look at the two bottom images of Fig. 32, obtained on a second example with respectively the identity and the logarithm, we notice that there is little change (shown partly in two white circles on the image) in compensating the bias when the feature has well-defined characteristics relatively to the background. The previous example had more confused features, and the compensation of the bias had more significant effects.

5.2 Speed issues

We have seen in Equations 1.3, 1.4 and their different variations that integration is performed by executing a number of scannings of the couples (state, decision). The choice of the scannings has a direct effect on the computing time. We can reduce the number of operations by :

- reducing the number of operations in each scanning : part of the operations may be useless. We may reduce this number of operations statically before the integration or dynamically during it, by skipping part of the states and decisions.
- reducing the number of scannings :
  - we may order the couples (state, decision) so that the result of an operation is used in the same scanning in a following operation. This ordering may be done also statically before the integration or dynamically during it.
  - we may stop the iterations before convergence, giving up optimality. We should then evaluate the imprecision.

To ease the programming, we have considered four images in the following order : 4.6, 4.4, 4.5, and 4.3, and rotated them so that S_0 is on the left side of the image and S_1 on the right side. All scannings were applied on them, and the results are presented in the same order in tables. The equations used are 1.3 and 1.4 and the states are composed of two points (see Section 3.2), but we note the scannings as if states were single points to

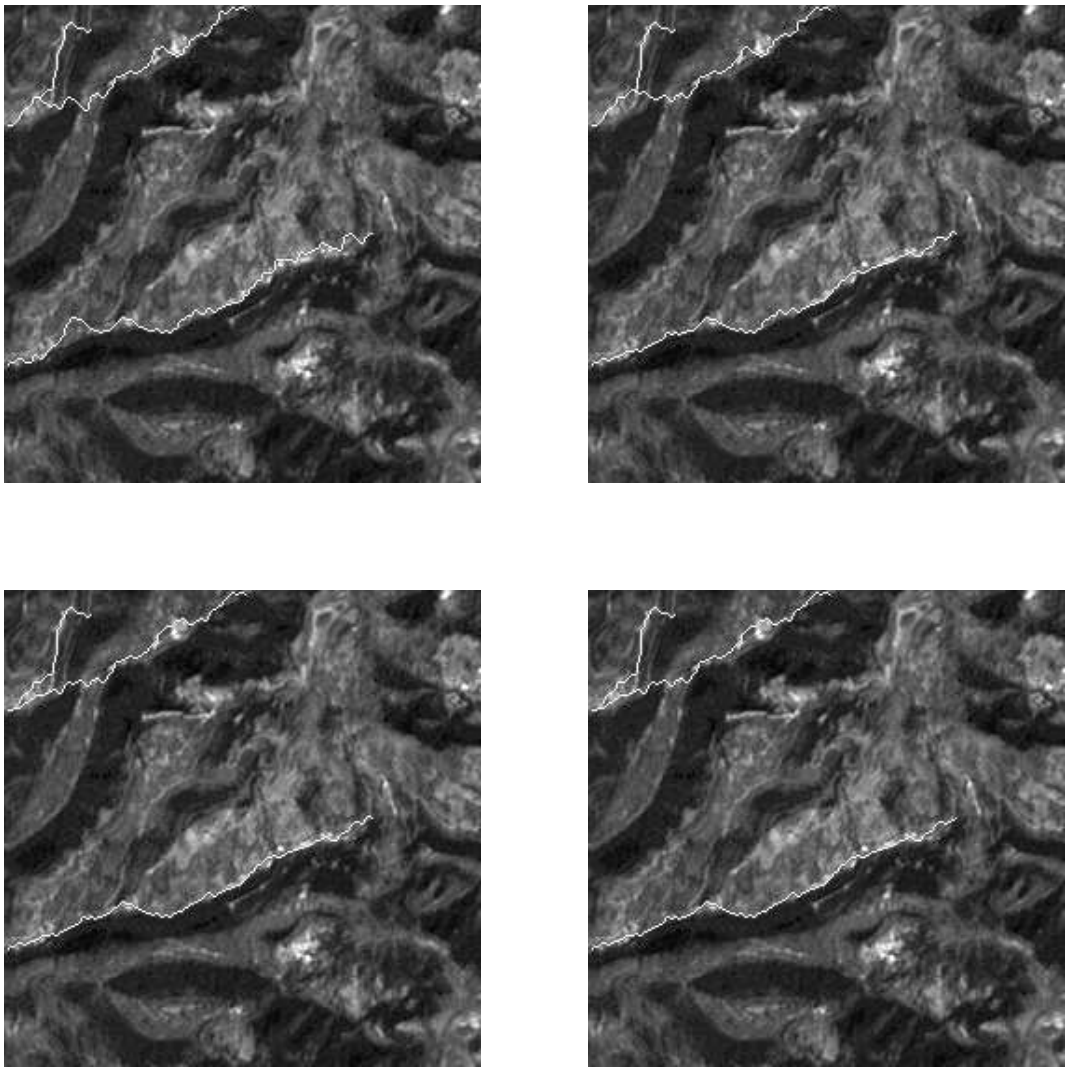

Figure 31: Results when compensating bias with the identity (top, left), the square root (top, right), the cubic root (bottom, left) and the logarithm (bottom, right).

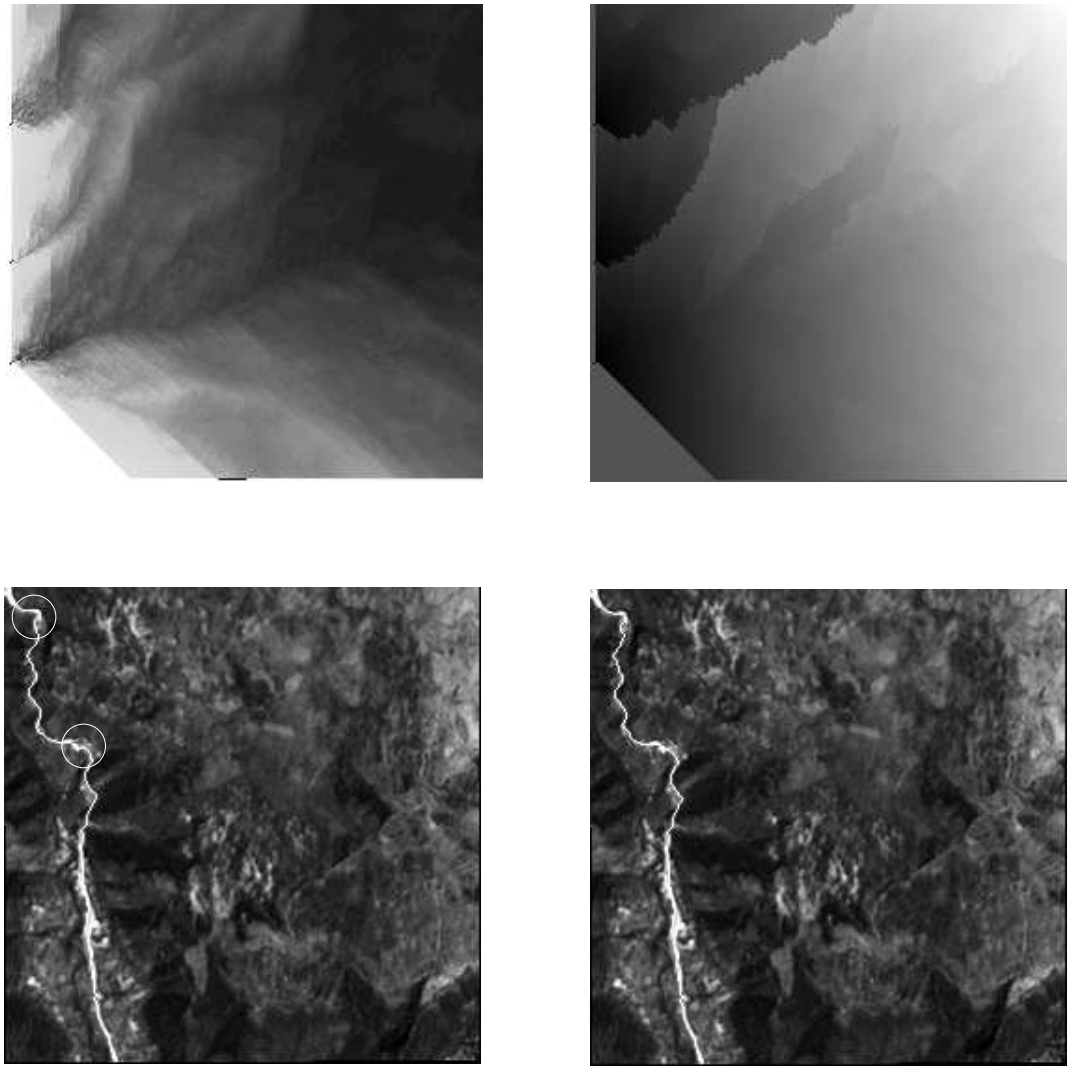

Figure 32: Final energy (top, left) and final auxiliary function (top, right) with the identity. Results when compensating bias with the identity (bottom, left) and the logarithm (bottom, right).

simplify the notation. We first consider static ordering of states and decisions. Then we present static elimination, dynamic ordering, dynamic elimination of states and decisions, and finally early stopping of the iterations.

5.2.1 Static ordering of states and decisions

If the scanning of the image is done randomly, many useless operations are performed. Let us consider for instance a path of n states from s_1 to s_n, with corresponding decisions q_1 to q_n : $s_i = \tau(s_{i+1}, q_{i+1})$.
We consider first an ordering O_1 of the couples (s_i, q_i) where i goes from n to 1. Then after the first scanning, only $U(s_2)$ has been updated, $U(s_3)$ after the second scanning, and we need (n-1) scannings to find the value of $U(s_n)$, after having performed $(n-1)^2$ elementary operations, most of them useless.
Let us consider now an ordering O_2 where i goes from 1 to n. During the first scanning, $U(s_2)$ is first updated, then immediately afterwards $U(s_3)$, and at the (n-1) th step of the same scanning, $U(s_n)$ gets its final value. We have performed only one scanning and (n-1) elementary operations.
O_1 and O_2 are respectively the worst and best orderings of these n couples (s_i, q_i), and all the other orderings have a performance between these two extreme cases, that is they perform between (n-1) and $(n-1)^2$ operations.
This example illustrates the main principle of ordering the couples : considering a couple (s_i, q_i) and a couple (s_j, q_j) belonging to the solution, and such that $s_i = \tau^n(s_j, q_j)$, we should deal with (s_i, q_i) before (s_j, q_j) in order to reduce the number of scannings and of useless operations.

A consequence of this principle is that states close to S_0 should be dealt with before states close to S_1. For applying this principle, we should consider two cases :

- we may have information a priori on the decisions and states belonging to the solution
- there may be no such information if the feature has a various and complicated shape for instance

a) With a priori knowledge of the shape of the solution

Let us consider a satellite image where S_0 (respectively S_1) contains only one point. We define new axes x_1 in the direction and orientation of $\vec{S_0S_1}$ and y_1 in the orthogonal direction. We define a scanning O_3 making two loops relatively first to x_1 (direct), second to y_1 (direct and reverse). If x_1 never decreases along the road, that is if the road does not "come back towards S_0", then the road is found in one unique iteration of O_3, although there is no absolute proof of optimality.

Practically, one of the two axes x or y may often be selected as a main axis. We will consider in the following three types of scannings :

- with the main loop along the y axis

- with the main loop along the x axis
- according to the distance to the segment $[S_0, S_1]$

If the satellite image contains several roads, the image may be partitioned in several regions [i], each corresponding to a road and to an associated such scanning O_3^i.

Scanning with the main loop along the y axis

As said above, we assume that S_0 is on the left-hand side of the image and S_1 on the right-hand side, the x-axis is horizontal rightwards and the y-axis vertical downwards. We work in 8-connectivity and 8 directions must be considered. The scanning and notations are as follows :

- main loop in y, increasing from the first row to the last : $\forall y \uparrow$,
  - secondary loop in x, increasing from the first column to the last : $\forall x \uparrow$, apply equations (1.3)-(1.4), for the decisions left (-1,0) and up-left (-1,-1)
  - $\forall x \downarrow$, apply (1.3)-(1.4) for the decisions up (0,-1) and up-right (1,-1)
- $\forall y \downarrow$,
  - $\forall x \uparrow$, apply (1.3)-(1.4) for the decisions down (0,1) and down-left (-1,1)
  - $\forall x \downarrow$, apply (1.3)-(1.4) for the decisions right (1,0) and down-right (1,1)

Thus, we scan all the couples (state, decision) only once in each complete scanning, but the decisions appear in the partial scannings where the principle of ordering may be applied. This scanning is similar to the one used by [6]. It is taken as basis for comparison and for further developments since this is the most common scanning.

The resulting images are respectively shown in Figures 4.6, 4.4, 4.5, and 4.3 and the number of operations is presented in the following table. Each column shows the results for images 4.6, 4.4, 4.5, and 4.3 from left to right. "Cycles" is the number of iterations. "States" is the number of couples (state, decision) scanned. "Energies" is the number of times that the sum $\phi + U$ is actually computed. "Energies" is different from "States" because of border effects in the general case, and in another section because of dynamic elimination of states and decisions. "Changes" is the number of times that the energy is updated. We present in Table 1 first the total number of operations "States", "Energies" and "Changes" during the whole computation, and then the average at each iteration. We finally present at each iteration the actual number of "Changes". Through all the experiments, the actual computing time is roughly proportional to the "Energies" number.

We note in Table 1 that the number of "Changes" rapidly decreases with the number of the iteration. The number of scannings is approximately constant ; it depends essentially upon the number of zig-zags in the shortest paths.

Cycles	20	16	13	18
States (tot)	34078720	27262976	22151168	30670848
Energies (tot)	33884400	27107520	22024860	30495960
Changes (tot)	5630769	3213335	2742197	3967957
States (ave)	1703936	1703936	1703936	1703936
Energies (ave)	1694220	1694220	1694220	1694220
Changes (ave)	281538	200833	210938	220442
Changes at 1	949966	770458	680126	912504
Changes at 2	848149	783053	690632	858859
Changes at 3	661980	516354	388670	460094
Changes at 4	569038	390592	269384	377037
Changes at 5	501987	265463	220698	287598
Changes at 6	437928	187343	180792	252613
Changes at 7	370160	126180	134815	228803
Changes at 8	322174	71081	101037	210293
Changes at 9	277033	42422	49448	149432
Changes at 10	240028	28479	19741	86453
Changes at 11	169604	17253	6706	61868
Changes at 12	127468	10213	148	38517
Changes at 13	80968	3728	0	23012
Changes at 14	45246	699	0	11283
Changes at 15	19471	17	0	6511
Changes at 16	6882	0	0	2803
Changes at 17	2502	0	0	277
Changes at 18	184	0	0	0
Changes at 19	1	0	0	0
Changes at 20	0	0	0	0

Table 1

Scanning with the main loop along the x axis

The roles of x and y are permuted and the scanning is :

$$\left\{ \begin{array}{l} \forall x \uparrow, \left\{ \begin{array}{l} \forall y \uparrow, \forall q \in \{(0,-1),(-1,-1)\} \\ \forall y \downarrow, \forall q \in \{(-1,0),(-1,1)\} \end{array} \right. \\ \forall x \downarrow, \left\{ \begin{array}{l} \forall y \uparrow, \forall q \in \{(1,-1),(1,0)\} \\ \forall y \downarrow, \forall q \in \{(1,1),(0,1)\} \end{array} \right. \end{array} \right\}, (1.3) - (1.4)$$

The images obtained are the same as the reference ones. We note in Table 2 (below) that the number of iterations is modified by -55 %, -37.5 %, +15.4 % and -33.3 %. The number of "Energies" is modified by the same ratio. The number of "Changes" is modified by -73 %, -51.4 %, +54.7 % and -40.1 %.

The most important conclusion from both experiments is that the efficiency of a scanning is related more to expectancy than to complexity. Both scannings obviously have the same complexity, but their performances are very different. In the second experiment, there is a better use of recursivity since the main loop is in the direction of the road. There are much more useless "Changes" in the first experiment. The scanning of the second experiment is therefore much better for examples one, two and four.

In the third example, the results are opposite. This is due to the many zig-zags of the road and to the fact that two roads are roughly perpendicular (see Fig. 3.31).

Cycles	9	10	15	12
States (tot)	15335424	17039360	25559040	20447232
Energies (tot)	15247980	16942200	25413300	20330640
Changes (tot)	1532376	1565504	4245940	2374623
States (ave)	1703936	1703936	1703936	1703936
Energies (ave)	1694220	1694220	1694220	1694220
Changes (ave)	170264	156550	283063	197885
Changes at 1	957942	991917	973397	1024977
Changes at 2	353657	309399	915589	524017
Changes at 3	129888	139131	690114	333406
Changes at 4	57893	70552	381744	198908
Changes at 5	24328	33806	308585	127055
Changes at 6	6725	14817	263139	80178
Changes at 7	1905	4925	230070	51975
Changes at 8	38	944	200807	24222
Changes at 9	0	13	133896	8092
Changes at 10	0	0	85404	1745
Changes at 11	0	0	43304	48
Changes at 12	0	0	15227	0
Changes at 13	0	0	4343	0
Changes at 14	0	0	321	0
Changes at 15	0	0	0	0

Table 2

Scanning according to the distance to the segment $[S_0, S_1]$

Assuming that a pixel close to the straight line joining S_0 to S_1 has more chance to belong to the road, we tried another scanning : we divided the image in ten regions according to the distance to this straight line, scanned the regions according to the growing distance to this straight line, and scanned the states in each region independently. The scanning is :

$$\forall region \uparrow, \left\{ \begin{array}{l} \forall y \uparrow, \left\{ \begin{array}{l} \forall x \uparrow, \forall q \in \{(-1,0),(-1,-1)\} \\ \forall x \downarrow, \forall q \in \{(0,-1),(1,-1)\} \end{array} \right. \\ \forall y \downarrow, \left\{ \begin{array}{l} \forall x \uparrow, \forall q \in \{(-1,1),(0,1)\} \\ \forall x \downarrow, \forall q \in \{(1,1),(1,0)\} \end{array} \right. \end{array} \right\}, (1.3) - (1.4)$$

We show in Fig. 33 and Fig. 34 the regions computed for the four examples, and the performance is given below. The shape of the regions is sometimes very different from the shape of the road since we wanted to rely at initialization only on the information given by the extremities. In comparison with the first experiment, the number of "Cycles" is modified by +35 %, + 43.7 %, +115.4 "Energies" by the same ratio, and the number of "Changes" by + 27.7 %, + 73.2 %, +150.4 % and + 7.6 %.

The scanning of this experiment is therefore worse than for the first experiment. Through the division in regions, we have actually "broken" the recursivity of the computation.

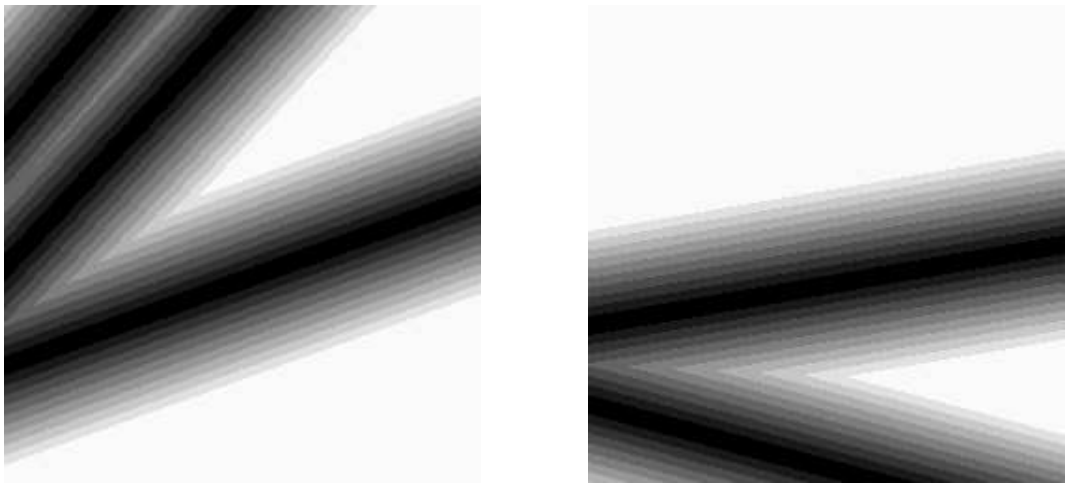

Figure 33: Static ordering according to the distance to the line joining the extremities, for images 4.6 (left) and 4.4 (right)

Cycles	27	23	28	26
States (tot)	46006272	39190528	47710208	44302336
Energies (tot)	45743940	38967060	47438160	44049720
Changes (tot)	7198080	5565179	6861093	4263417
States (ave)	1703936	1703936	1703936	1703936
Energies (ave)	1694220	1694220	1694220	1694220
Changes (ave)	266596	241964	245039	163978
Changes at 1	899685	850713	932185	893162
Changes at 2	841662	780956	913866	850508
Changes at 3	716594	582809	852326	413647
Changes at 4	619933	505693	814824	352844
Changes at 5	552788	439740	708440	281425
Changes at 6	503717	399422	587776	250774
Changes at 7	451829	369252	448817	233296
Changes at 8	402354	324966	335466	218717
Changes at 9	361987	278002	242964	161884
Changes at 10	325524	235185	198820	151464
Changes at 11	289817	188284	147489	91793
Changes at 12	267251	150319	104283	85715
Changes at 13	220816	121318	78817	68064
Changes at 14	179489	103890	71990	53048
Changes at 15	151523	86082	68879	43155
Changes at 16	123736	64746	61493	35685
Changes at 17	94023	46536	57285	27870
Changes at 18	75281	23259	55658	21407
Changes at 19	46773	9951	52912	13027
Changes at 20	34713	3544	49287	6976
Changes at 21	20970	508	37309	4414
Changes at 22	8805	4	19077	2475
Changes at 23	5615	0	9803	1451
Changes at 24	2559	0	5726	593
Changes at 25	587	0	3144	23
Changes at 26	49	0	1681	0
Changes at 27	0	0	776	0
Changes at 28	0	0	0	0

Table 3

b) Without a priori knowledge of the shape of the solution

In the most general case, we do not have a priori information on the shape of the solution. In order to find the optimal solution, many paths must be tested which can have very various shapes.

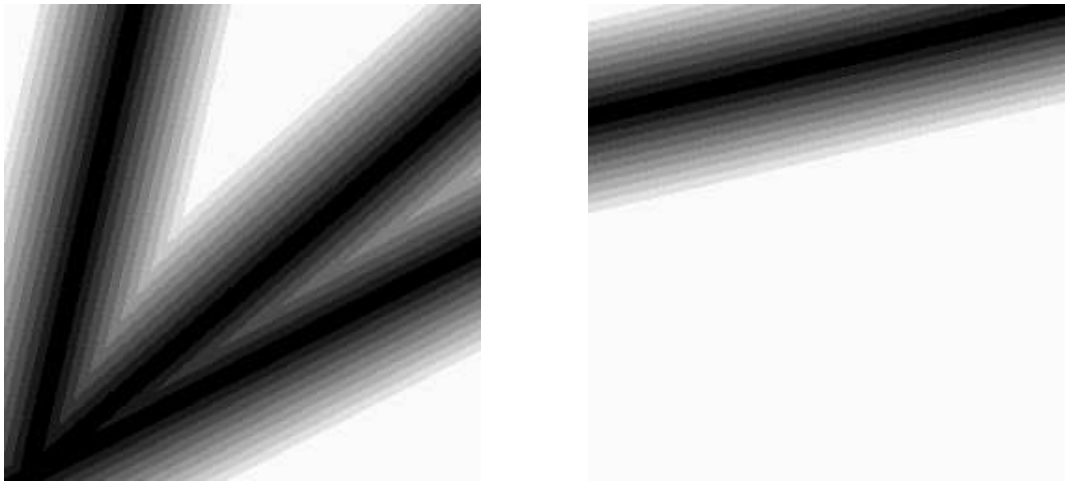

Figure 34: Static ordering according to the distance to the line joining the extremities, for images 4.5 (left) and 4.3 (right)

Moreover, S_1 is not necessary known a priori and may depend upon the final energy if it is defined for instance as the local minima of the energy in some geometric set ; then the direction from S_0 to S_1 is even not known.

Even in such cases, some scannings are better than others. We can exploit several facts related to the structure of the image. First, every pixel has a constant number of neighbors (except on the border of the image). Second, the image is a planar graph : this allows a simple and organized propagation of the information.

We consider the scanning defined as the combination of the first experiment and the second ; the main loop is alternatively along the y axis and along the x axis.

$$\left\{ \begin{array}{l} \forall y \uparrow, \left\{ \begin{array}{l} \forall x \uparrow, \forall q \in \{(-1,0),(-1,-1)\} \\ \forall x \downarrow, \forall q \in \{(0,-1),(1,-1)\} \end{array} \right. \\ \forall y \downarrow, \left\{ \begin{array}{l} \forall x \uparrow, \forall q \in \{(-1,1),(0,1)\} \\ \forall x \downarrow, \forall q \in \{(1,1),(1,0)\} \end{array} \right. \\ \forall x \uparrow, \left\{ \begin{array}{l} \forall y \uparrow, \forall q \in \{(0,-1),(-1,-1)\} \\ \forall y \downarrow, \forall q \in \{(-1,0),(-1,1)\} \end{array} \right. \\ \forall x \downarrow, \left\{ \begin{array}{l} \forall y \uparrow, \forall q \in \{(1,-1),(1,0)\} \\ \forall y \downarrow, \forall q \in \{(1,1),(0,1)\} \end{array} \right. \end{array} \right\}, (1.3) - (1.4)$$

The resulting images are the same as before. In Table 4, we should compare the number of "Cycles" of experiment one with the double of the "Cycles" in the current experiment, since every iteration now contains two complete scannings of the states and decisions. The differences of "Cycles" and of "Energies" with the first experiment are then : -50 %, -50 %, -23 %, -44 %. The differences of "Changes" are -63.7 %, -47.6 %, -44.5 %, -46 %.

This series of scannings is therefore the best which we have obtained by static ordering of the states and decisions. It is anisotropic, and no matter how many zig-zags the road does, the maximum of recursivity is exploited. If we consider for instance the letter "W" and turn it in any direction, this scanning recovers it in only one (double) iteration. With previous experiments, between one and four iterations were needed according to the orientation.

We note that any curve not containing two decisions of opposite direction is also obtained in only one complete scanning.
Therefore, any path composed of n such curves is found in n complete scannings at most.
However, due to the optimality request, we should remember that the sinuousities of the solution are not enough for evaluating the total number of scannings, and that other paths may be taken into account. The smoother the image, the smaller the number of scannings needed.

Cycles	5	4	5	5
States (tot)	17039360	13631488	17039360	17039360
Energies (tot)	16942200	13553760	16942200	16942200
Changes (tot)	2043898	1685273	1521217	2143520
States (ave)	3407872	3407872	3407872	3407872
Energies (ave)	3388440	3388440	3388440	3388440
Changes (ave)	408780	421318	304243	428704
Changes at 1	1783318	1520403	1318681	1662985
Changes at 2	250146	164743	177579	478694
Changes at 3	10349	127	24810	1839
Changes at 4	85	0	147	2
Changes at 5	0	0	0	0

Table 4

5.2.2 Static elimination of states and decisions

We may eliminate specific couples (state, decision) from the scannings when we know that for a given problem, they can not be part of the solution. However, the restriction in our applications was to eliminate separately specific states and/or specific decisions, which can not be part of the solution.

Elimination of states

When some states can not be part of the solution, we can skip them in the scannings, that is we can perform a windowing. For instance, pixels located far away from the line joining S_0 and S_1 can be assumed not belonging to the feature. We present in Fig. 35 and Fig. 36 the windows used. They were defined "manually".

In Table 5, the numbers of "Cycles" are different from the reference experiment by -25 %, -31 %, -15.4 %, -5.6 %. The numbers of "Energies" are different by -76.8 %, -61 %, -51.8 %, -85.1 %. Clearly, windowing allows to reduce considerably the computations, but the window must be chosen carefully...

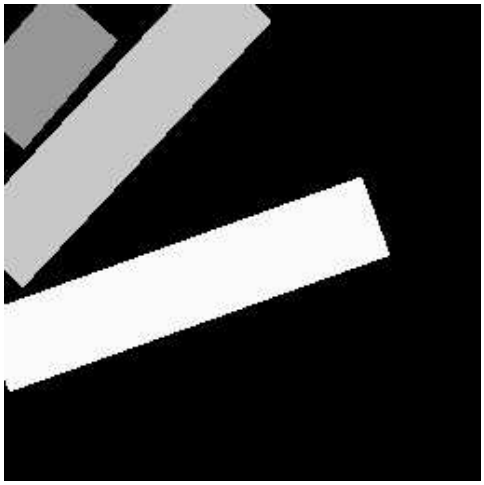
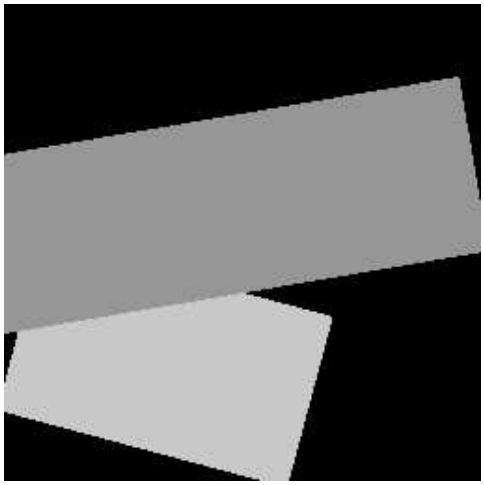

Figure 35: Static elimination of states, for images 4.6 (left) and 4.4 (right)

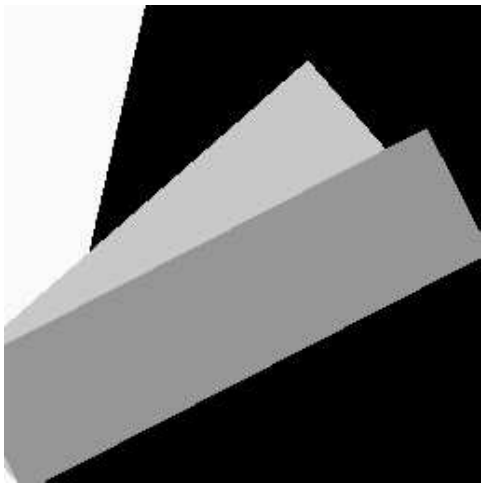
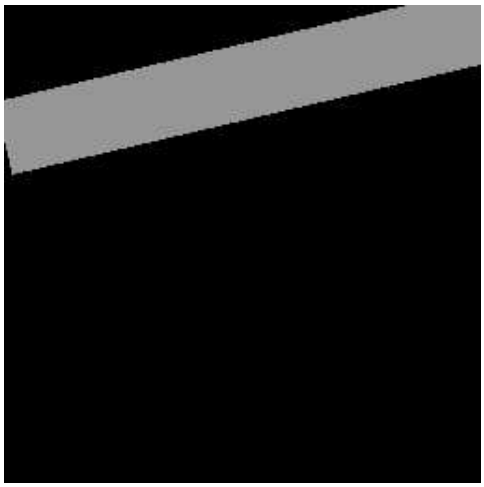

Figure 36: Static elimination of states, for images 4.5 (left) and 4.3 (right)

Cycles	15	11	11	17
States (tot)	7875270	10484188	10660650	4550390
Energies (tot)	7846110	10467204	10621204	4534818
Changes (tot)	938290	1596443	1082381	823334
States (ave)	525018	953108	969150	267670
Energies (ave)	523074	951564	965564	266754
Changes (ave)	62553	145131	98398	48431
Changes at 1	193456	434980	367689	120767
Changes at 2	220010	453150	320316	139022
Changes at 3	124801	282249	146466	106687
Changes at 4	92557	190992	85239	91311
Changes at 5	76045	114399	65153	76354
Changes at 6	62950	65797	48407	62620
Changes at 7	47814	44166	27999	52977
Changes at 8	41755	9256	16910	45608
Changes at 9	35509	1205	4059	37999
Changes at 10	23865	249	143	32430
Changes at 11	12846	0	0	24885
Changes at 12	5341	0	0	16725
Changes at 13	1335	0	0	9218
Changes at 14	6	0	0	4599
Changes at 15	0	0	0	2076
Changes at 16	0	0	0	56
Changes at 17	0	0	0	0

Table 5

We consider states composed of two pixels, and may also impose restrictions on the direction of the vector joining them. In the next experiment, we eliminate four directions out of the eight possible ones, i.e. those roughly perpendicular to the straight line joining S_0 to S_1. We consider the same windowing as previously for the location.

The gain is now -85.7 %, -74.1 %, -70.4 %, -90.8 % for the "Energies". It is much higher, but a constraint has been imposed on the solution, which is clearly visible in Fig. 37. The images were rotated as explained in the beginning of Section 5.2.

Figure 37: Result when windowing the orientation of the state.

Cycles	15	12	11	17
States (tot)	4846320	7038336	6560400	2800240
Energies (tot)	4823100	7022160	6527125	2792607
Changes (tot)	574739	994933	658659	533081
States (ave)	323088	586528	596400	164720
Energies (ave)	321540	585180	593375	164271
Changes (ave)	38316	82911	59878	31358
Changes at 1	141981	303276	265112	85216
Changes at 2	123420	280645	170465	93919
Changes at 3	73530	172879	84131	69115
Changes at 4	55211	106981	50278	57706
Changes at 5	45586	64251	38740	49015
Changes at 6	36926	34560	26341	39843
Changes at 7	28527	19240	13899	33459
Changes at 8	25228	8237	7990	27638
Changes at 9	20692	3513	1652	23691
Changes at 10	13182	1140	51	19827
Changes at 11	7193	211	0	14654
Changes at 12	2789	0	0	10023
Changes at 13	472	0	0	5431
Changes at 14	2	0	0	2622
Changes at 15	0	0	0	900
Changes at 16	0	0	0	22
Changes at 17	0	0	0	0

Table 6

Elimination of decisions

Similarly, some decisions/directions may be eliminated from the scannings when the probability that they belong to the solution is null or small. In brain images, the horizontal direction can be eliminated, while the vertical direction can be eliminated for the images of the music staves. Besides, only one orientation is to be considered in those both applications.

Sometimes, we must impose these restrictions as a definition of the feature which we are looking for. For instance, in a film of moving objects, we can scan the axis of time only in one direction.

Moreover, eliminating some directions is a necessity for avoiding possible loops when we deal with the bias of the shortest path (see Section 5.1).

Table 7 was obtained by eliminating four decisions out of the eight possible, i.e. those roughly perpendicular to the straight line joining S_0 to S_1. The gain in "Energies" is -41.5 %, -38.4 %, -38.5 %, -38.5 %.

Cycles	19	16	13	18
States (tot)	19922944	16777216	13631488	18874368
Energies (tot)	19811205	16683120	13555035	18768510
Changes (tot)	4341765	2476344	2211406	2583576
States (ave)	1048576	1048576	1048576	1048576
Energies (ave)	1042695	1042695	1042695	1042695
Changes (ave)	228514	154772	170108	143532
Changes at 1	683400	605362	559918	527551
Changes at 2	638616	560975	518208	520429
Changes at 3	513790	391631	319018	359128
Changes at 4	434617	303689	224319	277607
Changes at 5	389443	213822	185133	211051
Changes at 6	338193	156372	146275	165890
Changes at 7	285766	97477	111766	134888
Changes at 8	248762	59715	83725	106856
Changes at 9	217327	36297	39993	88568
Changes at 10	186712	25024	17843	73113
Changes at 11	148412	14347	5075	51304
Changes at 12	114948	8215	133	32083
Changes at 13	73461	2891	0	18710
Changes at 14	40436	514	0	9105
Changes at 15	18999	13	0	5082
Changes at 16	7338	0	0	2045
Changes at 17	1509	0	0	166
Changes at 18	36	0	0	0
Changes at 19	0	0	0	0

Table 7

Now, we eliminate locations, orientation of states, and decisions. The gain in "Energies" relatively to the reference scanning is -87.5 %, -77.3 %, -74.1 %, -92 %.

Cycles	15	12	11	17
States (tot)	4240530	6158544	5740350	2450210
Energies (tot)	4220325	6144240	5711123	2444209
Changes (tot)	574737	994932	658660	533076
States (ave)	282702	513212	521850	144130
Energies (ave)	281355	512020	519193	143777
Changes (ave)	38316	82911	59878	31357
Changes at 1	141978	303275	265113	85211
Changes at 2	123421	280645	170465	93919
Changes at 3	73530	172879	84131	69115
Changes at 4	55211	106981	50278	57706
Changes at 5	45586	64251	38740	49015
Changes at 6	36926	34560	26341	39843
Changes at 7	28527	19240	13899	33459
Changes at 8	25228	8237	7990	27638
Changes at 9	20692	3513	1652	23691
Changes at 10	13182	1140	51	19827
Changes at 11	7193	211	0	14654
Changes at 12	2789	0	0	10023
Changes at 13	472	0	0	5431
Changes at 14	2	0	0	2622
Changes at 15	0	0	0	900
Changes at 16	0	0	0	22
Changes at 17	0	0	0	0

Table 8

Finally, we scan with a main loop along the x axis, in only one direction (rightwards) and for half of the decisions.

Cycles	2	2	2	2
States (tot)	1703936	1703936	1703936	1703936
Energies (tot)	1694220	1694220	1694220	1694220
Changes (tot)	555310	491590	451071	679155
States (ave)	851968	851968	851968	851968
Energies (ave)	847110	847110	847110	847110
Changes (ave)	277655	245795	225536	339578
Changes at 1	555310	491590	451071	679155
Changes at 2	0	0	0	0

Table 9

As a conclusion, huge reductions of computation may be obtained by eliminating part of the states (location and orientation), decisions, and directions of scanning. Unfortunately,

the results deteriorate very much with these reductions, in particular when the decision or direction of state is involved.

5.2.3 Dynamic ordering of states and decisions

During the iterations, we may favor regions where "things happen", leaving aside whole regions where there is no change. To simplify, we consider only the ordering of the states, and take all possible decisions for each state.

We have tried to compute first states close to the states where the energy has changed most recently. Intuitively, this new value of the energy is smaller, and the neighbors should now be updated (this ordering is similar to the A* algorithm). But it leads to scan the feature in a direction which avoids recursivity. Several hundred iterations are obtained.

When scanning the states inversely, starting from states which have not changed for the longest time, the results are slightly better, but still about sixty iterations are needed in the chosen examples.

Thus, ordering states dynamically according to the last date of change gives poor results. The basic reason is that this re-ordering breaks the recursive propagation of the information.

We have tried another dynamic ordering of states, relatively to the current value of the energy. We can define a partition of the image, according to the value of the energy U. We define a series of thresholds from t_1 to t_n such that $t_i < t_{i+1}$ and corresponding classes $C_i = \{s | t_i \leq U(s) < t_{i+1}\}$. We then deal first with the class C_1, then with C_2, ..., and finally with C_n. Thus, we prolongate the shortest paths preferably (as with the A* algorithm). This is actually equivalent to a series of pruning of decreasing roughness. The gain in "Cycles" and "Energies" is -5 %, 0 %, -7.7 %, 0 %. It is neglectable, and dynamic ordering of states does not seem useful.

Cycles	19	16	12	18
States (tot)	32374784	27262976	20447232	30670848
Energies (tot)	32190180	27107520	20330640	30495960
Changes (tot)	4686821	2917141	2445418	3797690
States (ave)	1703936	1703936	1703936	1703936
Energies (ave)	1694220	1694220	1694220	1694220
Changes (ave)	246675	182321	203785	210983
Changes at 1	949966	770458	680126	912504
Changes at 2	810362	740896	662002	829975
Changes at 3	547759	414530	369442	431732
Changes at 4	496306	354379	240911	340398
Changes at 5	409860	229926	200242	271717
Changes at 6	292209	146115	127866	231513
Changes at 7	296374	102798	93690	217399
Changes at 8	263652	56086	46353	205073
Changes at 9	206186	40580	17641	147928
Changes at 10	151087	28348	6911	85890
Changes at 11	118096	16215	234	59531
Changes at 12	78121	11809	0	34062
Changes at 13	43637	4321	0	17071
Changes at 14	16144	658	0	8801
Changes at 15	4560	22	0	3818
Changes at 16	2318	0	0	275
Changes at 17	183	0	0	3
Changes at 18	1	0	0	0
Changes at 19	0	0	0	0

Table 10

We show in Fig. 38 the energy classes after 1 iteration (left) and at convergence (right) for the image presented in Fig. 4.3.

We show in Fig. 39 the energy classes at convergence (left) for the image presented in Fig. 4.5, and the date of last change of U (right).

5.2.4 Dynamic elimination of states and decisions

During the integration, it is possible to reach conclusions which could not be found initially. When the energy of a state $\tau(s,q)$ is above the current energy of S_1, then computing a new $U(s)$ with respect to q would not contribute to the solution, and we skip the couple (s,q) during the current scanning : we perform a pruning of the search.

When S_1 is composed of several points s_1 corresponding to the extremities of different features, the pruning may be performed in two ways :

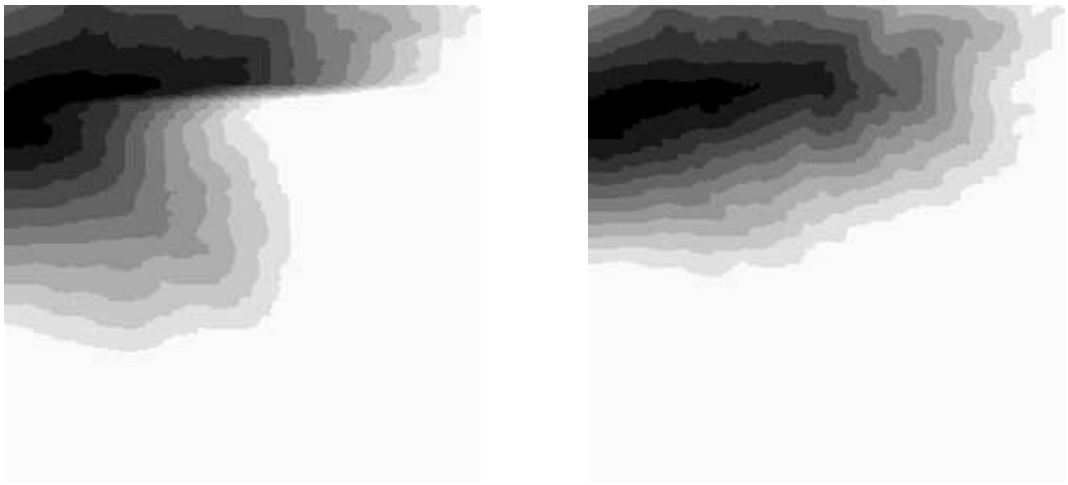

Figure 38: Dynamic ordering of the states according to U for image 4.3. Left : after 1 iteration. Right : at convergence.

- if we know for each state s that it may belong to only one known feature, we may compare the current value of the energy in s with its value in the corresponding (known) s_1 : we perform a local pruning.
- otherwise and if we want to keep optimality, we must compare U(s) with $\max_{S_1} U(s_1)$: this is a global pruning.

Of course, more states are skipped with local pruning than with global pruning : the conditions for applying local pruning are more restrictive, but it is more efficient.

Global pruning

There is a paradox that sometimes more computation of the energies are performed in the second iteration than in the first. This is due to the presence of regions of "infinite" energy during the first iteration. In these regions, the energy is not computed and updated.

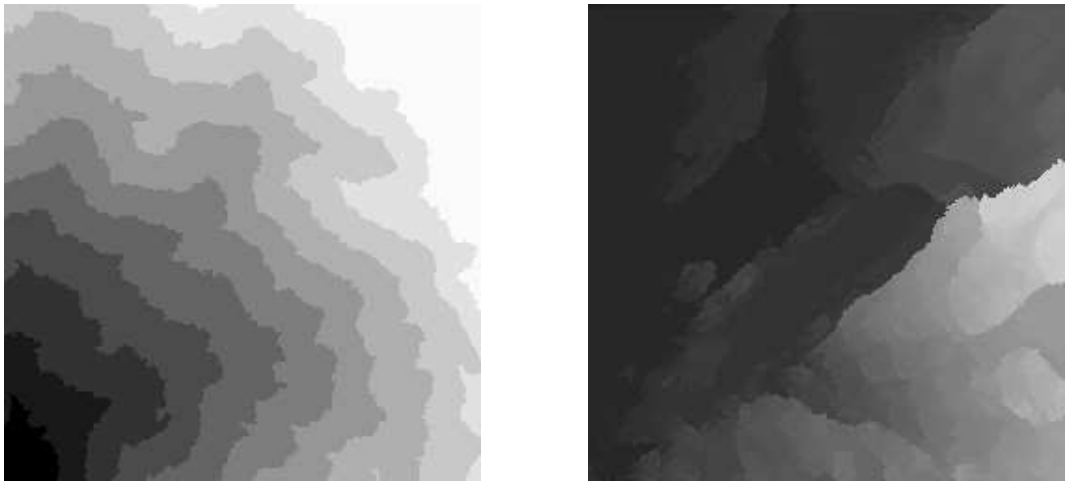

Figure 39: Dynamic ordering of the states according to U (left) and to the last change of U (right), for image 4.5.

Cycles	14	16	13	17
States (tot)	23855104	27262976	22151168	28966912
Energies (tot)	13768623	21800860	18749299	14226072
Changes (tot)	2751957	2883642	2338694	2787742
States (ave)	1703936	1703936	1703936	1703936
Energies (ave)	983473	1362554	1442254	836828
Changes (ave)	196568	180228	179900	163985
Changes at 1	949966	770458	680126	912504
Changes at 2	596402	783053	611554	528994
Changes at 3	328265	415010	322040	321070
Changes at 4	255934	314270	213709	257178
Changes at 5	205014	209627	171694	189238
Changes at 6	157409	146749	135751	150020
Changes at 7	102220	94705	94441	120480
Changes at 8	67983	57372	65147	103248
Changes at 9	46219	36967	28991	70527
Changes at 10	30353	25199	11710	55924
Changes at 11	10106	15907	3408	38680
Changes at 12	1971	9922	123	22182
Changes at 13	115	3690	0	12092
Changes at 14	0	696	0	4407
Changes at 15	0	17	0	1045
Changes at 16	0	0	0	153
Changes at 17	0	0	0	0

Table 11

The gains in "Cycles" are -30 %, 0 %, 0 % and -5.6 %. The gains in "Energies" are -59 %, -19.6 %, -14.9 %, -53.4 %, with respect to the performance of the reference scanning.

Local pruning

If we know for each state s^i that it can belong only to the feature between S_0 and s_1^i, then we can compare $U(s^i)$ with $U(s_1^i)$ at each iteration without loss of optimality. If this condition is not verified, we may lose optimality.

In Fig. 40, we show the distribution chosen for local pruning according to the location of the points of S_0 and S_1. We draw the several lines joining extremities of S_0 and S_1. We compute their voronoi diagram and associate each pixel with the nearest line.

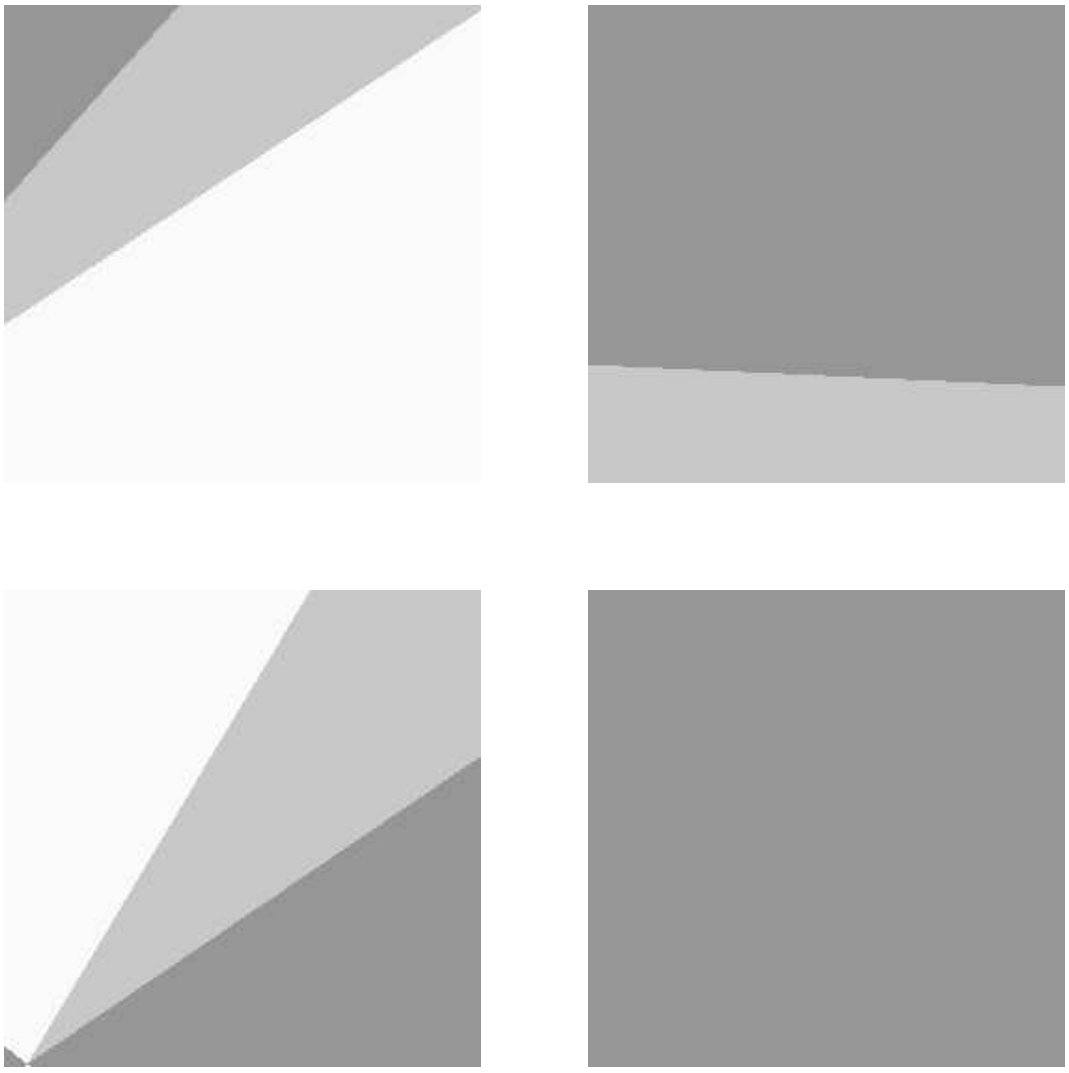

Figure 40: Labelling of the pixels in local pruning, for images 4.6 (top left), 4.4 (top right), 4.5 (bottom left) and 4.3 (bottom right).

Cycles	14	12	13	17
States (tot)	23855104	20447232	22151168	28966912
Energies (tot)	12621350	13867522	16895326	14226072
Changes (tot)	2572177	2177392	2168570	2787742
States (ave)	1703936	1703936	1703936	1703936
Energies (ave)	901525	1155627	1299640	836828
Changes (ave)	183727	181449	166813	163985
Changes at 1	949966	770458	680126	912504
Changes at 2	520722	709988	583416	528994
Changes at 3	299508	327488	283790	321070
Changes at 4	233544	193083	186913	257178
Changes at 5	186568	97378	149170	189238
Changes at 6	142004	45949	115584	150020
Changes at 7	92195	24341	77776	120480
Changes at 8	62522	7155	50991	103248
Changes at 9	43534	1454	25571	70527
Changes at 10	29436	97	11702	55924
Changes at 11	10092	1	3408	38680
Changes at 12	1971	0	123	22182
Changes at 13	115	0	0	12092
Changes at 14	0	0	0	4407
Changes at 15	0	0	0	1045
Changes at 16	0	0	0	153
Changes at 17	0	0	0	0

Table 12

The gains in "Cycles" are -30 %, -25 %, 0 % and -5.6 %. The gains in "Energies" are -62.7 %, -48.8 %, -23.3 %, -53.4 %. Local pruning gives better results than global pruning.

The values of the energy on the extremities are represented in Table 13 for the images shown in Fig. 4.6, 4.4 and 4.5 and for all the iterations. Images are separated by double bars and paths of the same image by single bars. Example : column number 5 corresponds to the the second element of S_1 (to the second path) for the image in Fig. 4.4. Each row corresponds to an iteration.

We see that the values of the energy vary a lot between extremities of the same image, and this is also an argument in favor of local pruning.

1000000	1000000	1000000	1000000	1000000	1000000	1000000	1000000
11121	27080	37196	125257	50771	101245	104221	85834
10586	26238	30132	83828	31233	100802	89780	85834
10586	25474	28693	82754	22643	100461	88772	85834
10586	24741	27534	81851	21536	100188	88301	85834
10586	24079	26829	81516	21501	100115	88140	85834
10586	23685	26340	0	0	100081	87900	85834
10586	23291	25895	0	0	100066	87739	85834
10586	23147	25542	0	0	100066	87737	85834
10586	23068	25301	0	0	100066	87737	85834
10586	23068	25010	0	0	100066	87737	85834
10586	23068	24768	0	0	100066	87737	85834
10586	23068	24616	0	0	100066	87737	85834
10586	23068	24616	0	0	0	0	0

Table 13

Both with local or global pruning, the equation of computation is :

$$\forall s, \forall q, U(\tau(s,q)) < T \Rightarrow \quad tmp \leftarrow \phi(s,q) + U(\tau(s,q)) \tag{27}$$

$$tmp \leq U(s) \Rightarrow U(s) \leftarrow tmp \tag{28}$$

and it may be rewritten by doing the loop on the states $\tau(s,q)$:

$$\forall s, U(s) < T \Rightarrow \forall q^*, \qquad tmp \leftarrow \phi(\tau(s,q^*),q) + U(s) \tag{29}$$

$$tmp \leq U(\tau(s,q^*)) \Rightarrow U(\tau(s,q^*)) \leftarrow tmp \tag{30}$$

In the second formulation, unnecessary tests $U(\tau(s,q)) < T$ are avoided. Since the gain is neglectable with respect to the overall number of operations, we kept using the first formulation.

5.2.5 Early stopping of the iterations

Let us consider the following four tables. They represent the Hausdorff distance between the current shortest path joining S_0 to S_1 and other paths. Each table corresponds to one example image. The first column is the number of the iteration, the second the number of the path and the third the number of points in the current solution. A "one" in the third column means that a solution has not been found yet.

Let us recall that the Hausdorff distance between two sets A and B is defined by $H_1(A,B) = H_2(B,A) = \max_{a \in A} \min_{b \in B} d(a,b)$ and $H(A,B) = \min\{H_1(A,B), H_2(A,B)\}$. We represent in the fourth and fifth columns the distances H_1 and H_2 between the current shortest path and the final solution. This is an indication of the imprecision of the current shortest path, once the solution has been found in a previous optimization, and it may be

used only for analysis. The sixth and seventh columns contain the Hausdorff distances between the current shortest path and the previous path at the last iteration.

Cycle	Path	Points	H1-fin	H2-fin	H1-prev	H2-prev
1	0	1	0.000	67.357	0.000	0.000
1	1	1	0.000	141.326	0.000	0.000
1	2	1	0.000	207.183	0.000	0.000
2	0	63	7.071	9.849	67.357	0.000
2	1	131	9.434	10.630	141.326	0.000
2	2	199	9.849	11.000	207.183	0.000
3	0	78	0.000	0.000	9.849	7.071
3	1	141	9.434	10.630	8.544	8.246
3	2	199	3.162	3.606	11.000	10.198
4	0	78	0.000	0.000	0.000	0.000
4	1	143	9.434	10.630	5.000	4.243
4	2	201	5.000	5.000	5.000	3.606
5	0	78	0.000	0.000	0.000	0.000
5	1	143	9.434	10.000	2.000	2.000
5	2	201	5.000	5.000	0.000	0.000
6	0	78	0.000	0.000	0.000	0.000
6	1	150	1.000	1.000	10.000	9.434
6	2	202	5.000	5.000	3.606	2.828
7	0	78	0.000	0.000	0.000	0.000
7	1	150	1.000	1.000	0.000	0.000
7	2	202	5.000	5.000	1.414	1.414
8	0	78	0.000	0.000	0.000	0.000
8	1	150	1.000	1.000	0.000	0.000
8	2	202	5.000	5.000	0.000	0.000
9	0	78	0.000	0.000	0.000	0.000
9	1	150	0.000	0.000	1.000	1.000
9	2	202	5.000	5.000	2.000	2.000
10	0	78	0.000	0.000	0.000	0.000
10	1	150	0.000	0.000	0.000	0.000
10	2	202	5.000	5.000	0.000	0.000
11	0	78	0.000	0.000	0.000	0.000
11	1	150	0.000	0.000	0.000	0.000
11	2	201	0.000	0.000	5.000	5.000
12	0	78	0.000	0.000	0.000	0.000
12	1	150	0.000	0.000	0.000	0.000
12	2	201	0.000	0.000	0.000	0.000
13	0	78	0.000	0.000	0.000	0.000
13	1	150	0.000	0.000	0.000	0.000
13	2	201	0.000	0.000	0.000	0.000
14	0	78	0.000	0.000	0.000	0.000
14	1	150	0.000	0.000	0.000	0.000
14	2	201	0.000	0.000	0.000	0.000

Table 14

Cycle	Path	Points	H1-fin	H2-fin	H1-prev	H2-prev
1	0	1	0.000	255.642	0.000	0.000
1	1	1	0.000	151.119	0.000	0.000
2	0	266	114.477	114.769	266.271	0.000
2	1	170	16.492	17.464	151.119	0.000
3	0	287	1.000	1.000	114.769	114.477
3	1	170	16.492	17.464	1.000	1.000
4	0	287	1.000	1.000	0.000	0.000
4	1	167	2.828	2.236	16.492	16.031
5	0	287	1.000	1.000	0.000	0.000
5	1	167	0.000	0.000	2.236	2.828
6	0	287	0.000	0.000	1.000	1.000
6	1	167	0.000	0.000	0.000	0.000
7	0	287	0.000	0.000	0.000	0.000
7	1	167	0.000	0.000	0.000	0.000
8	0	287	0.000	0.000	0.000	0.000
8	1	167	0.000	0.000	0.000	0.000
9	0	287	0.000	0.000	0.000	0.000
9	1	167	0.000	0.000	0.000	0.000
10	0	287	0.000	0.000	0.000	0.000
10	1	167	0.000	0.000	0.000	0.000
11	0	287	0.000	0.000	0.000	0.000
11	1	167	0.000	0.000	0.000	0.000
12	0	287	0.000	0.000	0.000	0.000
12	1	167	0.000	0.000	0.000	0.000
13	0	287	0.000	0.000	0.000	0.000
13	1	167	0.000	0.000	0.000	0.000
14	0	287	0.000	0.000	0.000	0.000
14	1	167	0.000	0.000	0.000	0.000
15	0	287	0.000	0.000	0.000	0.000
15	1	167	0.000	0.000	0.000	0.000
16	0	287	0.000	0.000	0.000	0.000
16	1	167	0.000	0.000	0.000	0.000

Table 15

We notice from columns 4 and 5 that the final path is reached much before convergence : at iterations 11 out of 14, 6 out of 16, 8 out of 13, and 9 out of 17 respectively. During the remaining iterations, all computations only check optimality, and the solution itself is not improved. Therefore, if we give up optimality, computations may be spared.

The problem is then to determine this decisive iteration : when do we reach the final shortest path ? We notice for these four examples that a possible criterion is the absence of

change in the shortest paths for two successive iterations. Then, computations are stopped at iterations 13, 8, 10, and 11. The gain in number of "Cycles" and "Energies" is -7 %, -50 %, -23 %, and -35 %. If this criterion is valid in a great variety of images, the early stopping of the iterations could be used to improve greatly the computations.

Cycle	Path	Points	H1-fin	H2-fin	H1-prev	H2-prev
1	0	1	0.000	272.356	0.000	0.000
1	1	1	0.000	256.632	0.000	0.000
1	2	1	0.000	259.563	0.000	0.000
2	0	266	2.236	2.828	272.356	0.000
2	1	239	59.203	60.959	256.632	0.000
2	2	269	0.000	0.000	259.563	0.000
3	0	266	2.236	2.828	1.000	1.000
3	1	317	1.000	1.000	60.959	59.203
3	2	269	0.000	0.000	0.000	0.000
4	0	266	2.236	2.828	1.000	1.000
4	1	317	1.000	1.000	1.000	1.000
4	2	269	0.000	0.000	0.000	0.000
5	0	266	2.236	2.828	1.000	1.000
5	1	317	1.000	1.000	1.000	1.000
5	2	269	0.000	0.000	0.000	0.000
6	0	266	0.000	0.000	2.828	2.236
6	1	317	0.000	0.000	1.000	1.000
6	2	269	0.000	0.000	0.000	0.000
7	0	266	0.000	0.000	0.000	0.000
7	1	317	1.000	1.000	1.000	1.000
7	2	269	0.000	0.000	0.000	0.000
8	0	266	0.000	0.000	0.000	0.000
8	1	317	0.000	0.000	1.000	1.000
8	2	269	0.000	0.000	0.000	0.000
9	0	266	0.000	0.000	0.000	0.000
9	1	317	0.000	0.000	0.000	0.000
9	2	269	0.000	0.000	0.000	0.000
10	0	266	0.000	0.000	0.000	0.000
10	1	317	0.000	0.000	0.000	0.000
10	2	269	0.000	0.000	0.000	0.000
11	0	266	0.000	0.000	0.000	0.000
11	1	317	0.000	0.000	0.000	0.000
11	2	269	0.000	0.000	0.000	0.000
12	0	266	0.000	0.000	0.000	0.000
12	1	317	0.000	0.000	0.000	0.000
12	2	269	0.000	0.000	0.000	0.000
13	0	266	0.000	0.000	0.000	0.000
13	1	317	0.000	0.000	0.000	0.000
13	2	269	0.000	0.000	0.000	0.000

Table 16

Cycle	Path	Points	H1-fin	H2-fin	H1-prev	H2-prev
1	0	1	0.000	258.815	0.000	0.000
2	0	257	9.434	12.530	258.815	0.000
3	0	257	6.000	7.071	7.000	5.657
4	0	274	5.385	5.657	7.071	6.403
5	0	274	3.606	3.162	3.606	3.606
6	0	274	3.606	3.162	2.000	1.414
7	0	274	3.606	3.162	1.414	1.414
8	0	274	3.606	3.162	2.000	2.000
9	0	274	0.000	0.000	3.162	3.606
10	0	274	0.000	0.000	0.000	0.000
11	0	274	0.000	0.000	0.000	0.000
12	0	274	0.000	0.000	0.000	0.000
13	0	274	0.000	0.000	0.000	0.000
14	0	274	0.000	0.000	0.000	0.000
15	0	274	0.000	0.000	0.000	0.000
16	0	274	0.000	0.000	0.000	0.000
17	0	274	0.000	0.000	0.000	0.000

Table 17

5.3 Updating the potential during the iterations

In Section 4.2, we have computed the automatic segments from neighborhoods of the extremities of the roads and from segments of roads found in these neighborhoods. The basic assumption was that the distribution of grey-levels for the road (respectively for the background) was the same for these neighborhoods and for the whole image.

In the next experiment, we update the potential once a solution has been found for every path of the image (at the second iteration for these four examples). We get a non optimal but complete solution for the roads, we expect it contains more information and more accurate information than the partial segments of Section 4.2.2, and we compute the potential now from the histograms of these solutions and from the whole image. Global information is therefore extracted here from the whole image. This also allows to check the validity of the assumption of similarity between the neighborhoods of the extremities and between the whole image.

In general we note that after the third iteration, two thirds of the energy changes have been done, and the solution is close to the optimal one. Therefore, the potential may be updated at this iteration instead.

We present in Fig. 41 the potentials of grey-level, in Fig. 42 the potentials of contrast, at initialization (dots) and once updated (continuous curve), and for the four example images presented in Fig. 4.6 (top left), 4.4 (top right), 4.5 (bottom left) and 4.3 (bottom right). The

updated potential is very close to the initial potential, the assumption of representativity of the neighborhood was therefore justified for these examples. The resulting images are similar to the ones obtained by the reference scanning.

Once the potential has been updated, we reinitialize the energy at $+\infty$ and restart computing it. Thus, one iteration has been added in the computations. The method has been implemented with a main loop of scanning along the x axis, in order to get a "good" solution as soon as possible on these examples. Relatively to this scanning, the differences of performance are by updating the potential + 7.1 %, -12.5 %, -7.7 % and + 11 % for the number of "Cycles" and +18.3 %, -12.1 %, -3 % and +24 % for the "Energies".

Cycles	15	14	12	19
States (tot)	25559040	23855104	20447232	32374784
Energies (tot)	16280239	19156883	18169959	17657476
Changes (tot)	3605800	3400045	3009056	3748061
States (ave)	1703936	1703936	1703936	1703936
Energies (ave)	1085349	1368349	1514163	929341
Changes (ave)	240387	242860	250755	197266
Changes at 1	949966	770458	680126	912504
Changes at 2	959596	774412	685638	913587
Changes at 3	561314	795984	659524	533127
Changes at 4	360816	375979	354022	328571
Changes at 5	257985	271919	229309	264803
Changes at 6	199182	164355	167664	198039
Changes at 7	139936	103566	119896	153267
Changes at 8	87720	77419	78428	121973
Changes at 9	46354	42412	28947	84581
Changes at 10	26777	17327	5329	75615
Changes at 11	8962	5761	173	61425
Changes at 12	4972	447	0	43319
Changes at 13	1999	6	0	27985
Changes at 14	221	0	0	16854
Changes at 15	0	0	0	7209
Changes at 16	0	0	0	3869
Changes at 17	0	0	0	1313
Changes at 18	0	0	0	20
Changes at 19	0	0	0	0

Table 18

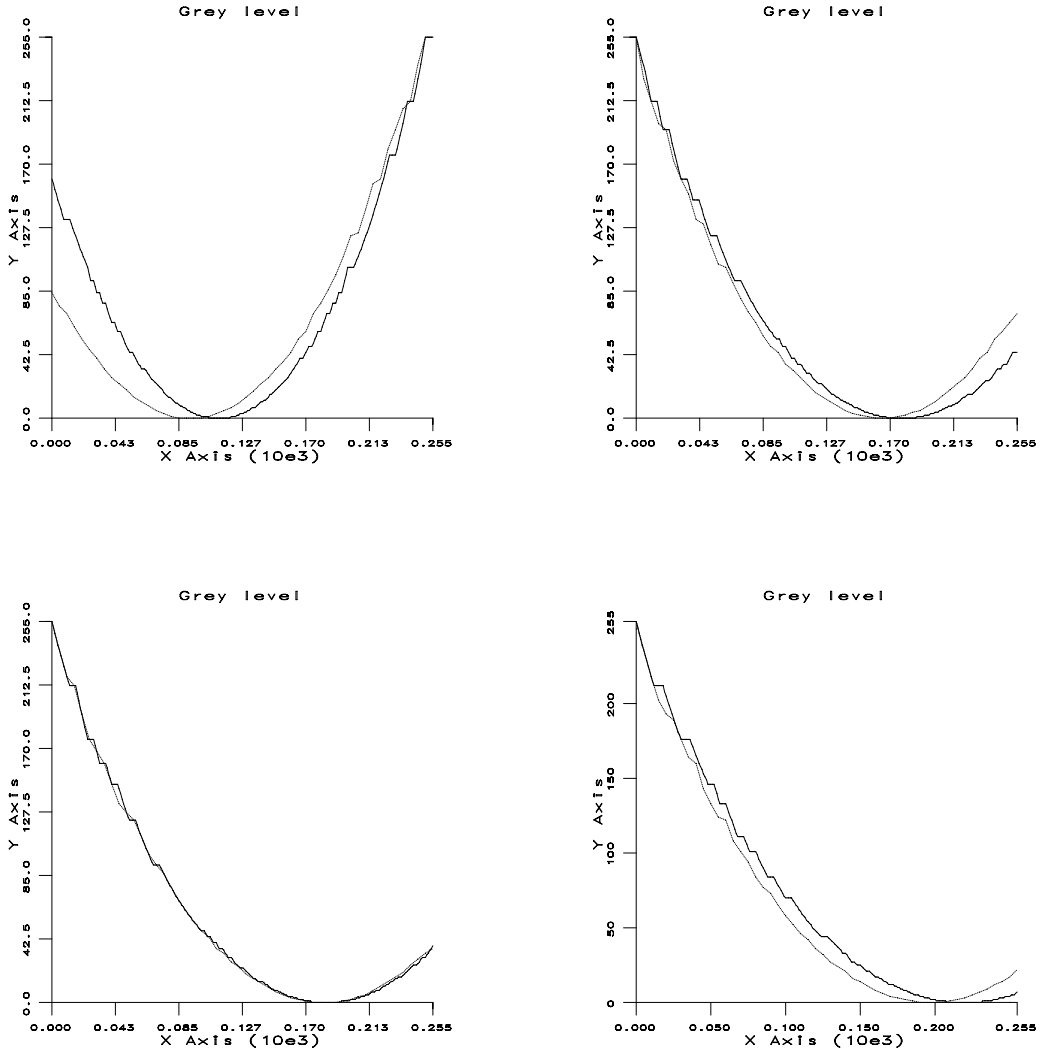

Figure 41: Updating the potential of grey-level during the iterations

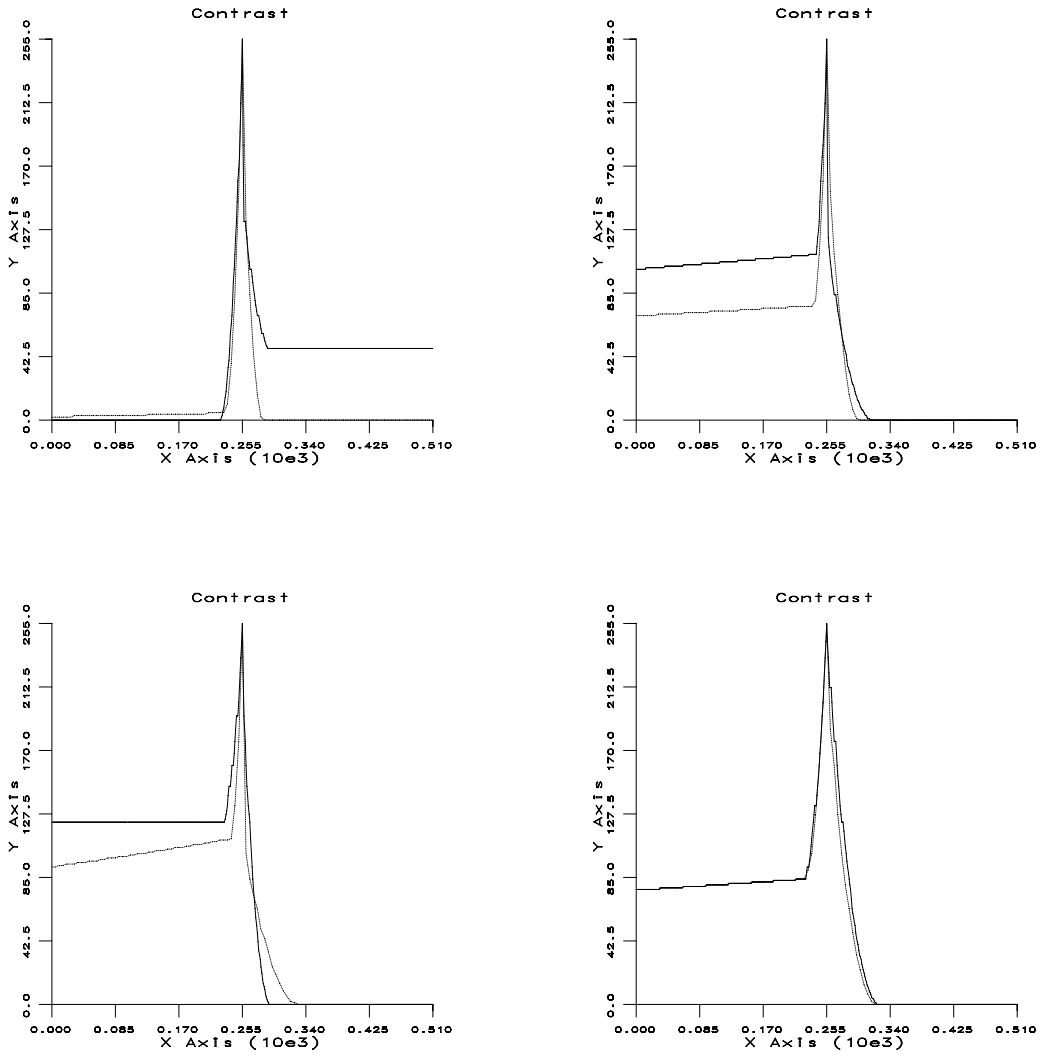

Figure 42: Updating the potential of contrast during the iterations

5.3.1 Extracting the shortest path

Finally, once the potential has been integrated, we generally wish to recover the shortest path. This is not always the case, for instance when computing the distance function we are interested by the value of the energy and not by the path itself.

5.3.2 Simple extraction

Usually, S_0 is the set of extremities on one side of the features, and S_1 is the set of the other extremities.

During integration, the energy was computed globally relatively to S_0, without distinguishing between elements of S_0. In particular, no correspondence was enforced between points of S_0 and S_1.

However, such a correspondence is enforced when recovering the shortest paths. We start the propagation at a point of S_1 and follow the best decisions (stored during integration) until we reach a point of S_0. The number of features obtained is the number of points of S_1, but several points of S_1 may lead to the same point of S_0 and the correspondence may be permuted. If the potential is sufficiently characteristic of the feature, such problems do not occur.

5.3.3 Extraction with oversets of the extremities

In the more general case, S_0 and S_1 may be defined less rigorously, as oversets of the extremities. This means that we have no precise notion of the extremities, only of the regions where they should be.

If S_0 is a strict overset of the extremities, the results should be the same if the additional points are "far" from the shortest paths. For instance, we often took for S_0 a complete border row or column in satellite images.

If S_1 is a strict overset of the extremities, non interesting paths may be found if we start propagation from all points of S_1. It is therefore more secure to eliminate non-relevant points of S_1. An efficient way is to keep only points of S_1 which are global or local minima of the energy in S_1 (see Section 3.1.2).

6 Conclusion

For roads detection in satellite images, global information may be integrated in several ways in the framework of dynamic programming :

1. by attributing to a state the characteristics of a more global feature to which it belongs to,
2. by defining the state as a structure composed of several points,
3. with a hierarchical approach from coarse to fine,
4. with the novel concept of auxiliary functions which store information on the current shortest path and which are taken into account in the potential. In this case, optimality is lost.

We have developed a new method to automatically define the potential from the grey-levels in windows around the extremities of the roads. The only information needed is the extremities of the roads. The respective influence of contrast and grey level is weighted by using conditional probabilities. The "more global features" used in this case are windows around the extremities and partial segments of roads detected inside these windows with a rough potential. The basic assumption is that the grey-level distribution is similar in the windows on one hand, and in the whole image on the other hand.

We have updated the potential during integration in order to use more global and accurate information. The global feature whose characteristics are used to compute the potential is then the current shortest path itself. The updated potential was similar to the initial one, and this justified the basic assumption used to define automatically the potential, for the examples presented in this report.

In order to take into account local curvature, we have defined the state as a set of two or three successive pixels and have presented a new algorithm of road detection under a constraint of local curvature. The constraint may be modulated by the choice of a curvature coefficient. This algorithm converges towards the optimum in a few dozens of scannings.

We have studied then three main improvements in the computation speed, relatively to scannings found in literature. We have presented an efficient and anisotropic ordering of states and decisions, which improved the number of computations by 50 %. When eliminating part of the states during the integration (pruning) according to the value of the energy, the gain regarding the reference scanning was between 20 % and 60 %. Local pruning was slightly better than global pruning. Finally, stopping the computations before convergence allowed to reduce the computations by up to 50 %. These improvements are independent, may be combined, and lead to the optimum solution (under certain precautions for the third method).

Since the curvature used in road detection was very local, we have defined a new concept : auxiliary functions in dynamic programming. Then we have integrated then global direction with an auxiliary function representing a 2D vector characterizing the direction of the path, and global curvature with a function representing the center of curvature of the path.

When looking at related work, we have noted a pioneering work with auxiliary functions, implicitly used : Shashua and Ullman extracted saliency networks with a potential function of the succession of angles along the shortest path. The framework presented in this report is actually more general.

Another application of auxiliary functions concerns for instance the average of the potential. Dynamic programming usually produces a bias towards solutions containing a smaller number of states. This is due to the fact that the energy is generally defined as the sum of the potential on a set of states, and therefore it depends upon the values of the potential as well as the number of states considered. We have computed the average of the potential instead of the sum, with an auxiliary function representing the length of the current shortest path. The resulting solutions were more sensible to local changes.

Among the various possibilities of integrating global information, auxiliary functions present several advantages : they do not modify the complexity of the original problem, they impose a constraint directly on the solution but in a way which may be modulated, and finally they can convey various information according to the application.

References

[1] M. Barzohar, D.B. Cooper, Automatic finding of main roads in aerial images by using geometric-stochastic models and estimation, *Proc. IEEE CVPR*, pp. 459-464, 1993.

[2] R. Bellman, *Dynamic Programming*, Princeton University Press, 1957.

[3] R. Bellman, R. Kalaba, *Dynamic Programming and Modern Control Theory*, Academic Press, New York and London, 1965.

[4] 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 Colloque Gretsi*, Juan-Les-Pins, pp. 699-702, June 1989.

[5] M.A.T. Figueiredo, J.M.N. Leitao, A Nonsmoothing Approach to the Estimation of Vessel Contours in Angiograms, IEEE Trans. on Medical Imaging, 1995.

[6] 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* vol. 15, pp. 201-223, 1981.

[7] D. Geman, B. Jedynak, Detection of roads in satellite images, *IGARSS Conference*, Espoo, 3d-6th June, 1991.

[8] A. Gruen, H. Li, Road extraction from aerial and satellite images by dynamic programming, *ISPRS Journal of Photogrammetry and Remote Sensing*, vol 50, n 4, pp. 11-20, 1995.

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

[10] 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)*, pp. 129-138, Dresden, September 1991.

[11] N. Merlet, A numerical distance for recognizing imperfect edges : roads, music staves, brain fissure, *9th Israeli Symposium on Artificial Intelligence and Computer Vision (IAICV)*, pp. 73-92, Tel Aviv, December 1992.

[12] N. Merlet, J. Zerubia, Classical Mechanics and Road Detection in SPOT Images. *Joint research report 1889*, INRIA/Hebrew University, April 1993.

[13] N. Merlet, J. Zerubia, A curvature-dependent energy function for detecting lines in satellite images, *8th SCIA Conference*, Tromso, May 1993.

[14] N. Merlet, J. Zerubia, New prospects in line detection for remote sensing images, *ICASSP Conference*, Adelaide, April 1994.

[15] N. Merlet, J. Zerubia, ELIESER : User Manual, *Joint technical report 180*, INRIA/Hebrew University, Sept. 1995.

[16] N. Merlet, J. Zerubia, Auxiliary functions in dynamic programming for smoothed road detection, *Conference on New Image Processing Techniques*, SPIE, June 1996.

[17] N. Merlet, J. Zerubia, New Prospects in Line Detection by Dynamic Programming, *IEEE-Trans PAMI*, vol. 18 n 4, pp. 426-431, April 1996.

[18] L.L. Scharf, H. Elliott, Aspects of Dynamic Programming in Signal and Image Processing, *IEEE-Trans on Automatic Control*, vol. AC-26, n 5, pp. 1018-1029, October 1981.

[19] A. Sha'ashua, S. Ullman, Structural Saliency : The Detection of Globally Salient Structures Using a Locally Connected Network, Proc. ICCV, pp. 321-327, 1988.

A Proofs

A.1 Equation 1.1

We assume that :

1. there is a finite number of states s
2. there is a finite number of decisions q
3. the potential ϕ is real and strictly positive
4. for two given states s_1 and s_n there is at least one path from s_1 to s_n and one path from s_n to s_1.

From assumption 4 above, we derive that $S = \{\sum_{s=s_1}^{s_n} \phi(s,q)\}$ is non empty, and from assumption 3 that it is a subset of $\mathcal{R}^{+*}$. Therefore it has a lower bound. Let us show that this bound is reached.

From assumptions 1 and 2, the set $\{\phi(s,q)\}$ is finite. From assumption 3, it is a subset of $\mathcal{R}^{+*}$. Therefore it has a minimum $\phi_m > 0$ which is reached at s_m and q_m.

Let U_0 be an element of S. We define $n_c = int(U_0/\phi_m) + 1$. Any path containing more than n_c elements has an energy U above U_0. We then partition S in the set S_a of energies obtained for paths containing less than n_c states and in the set S_b of energies obtained for paths containing more than n_c states. Any element of S_b is greater than U_0. S_a is a finite subset of $\mathcal{R}^{+*}$, therefore it has a minimum U_m which is also the minimum of S and it is reached for a path h_m.

A.2 Equation 1.2

We call U_m the minimum value obtained for $U = \sum_{s=s_0}^{s_1} \phi(s,q)$ and U_i the function obtained according to the process described below.

At initialization, $U_i(s_0) \leftarrow \min_{q \in Q}\{\phi(s_0,q)\}$, and $U_i(s) \leftarrow +\infty$ for $s \neq s_0$.

Then, we perform the iterations : $\forall s \neq s_0, U_i(s) \leftarrow \min_{q \in Q}\{\phi(s,q) + U_i(\tau(s,q))\}$ until no change appears in a complete scanning of all the states.

Let us show first that U_i is strictly positive. If there are values for which U_i is negative, we call $U_i^-(s)$ the first value (during the scannings of the state). It is impossible that $s = s_0$ since the potential is supposed to be strictly positive and Q finite. So, there is some q for which $U_i^-(s) = \phi(s,q) + U_i(\tau(s,q))$. Since $\phi(s,q)$ is strictly positive, we deduce that $U_i(\tau(s,q))$ is strictly negative, which is in contradiction with the hypothesis that $U_i^-(s)$ was the first negative value. Therefore U_i is strictly positive.

Let us show that U_i decreases at each iteration (it does not increase). If there are iterations at which $U_i(s)$ increases, let us call $U_i^2(s)$ the first occurrence of an increase, and $U_i^1(s)$ the preceding step : $U_i^1(s) < U_i^2(s)$. Then (s is necessarily different from s_0), $U_i^1(s) = \phi(s,q_1) + U_i^1(\tau(s,q_1))$ and $U_i^2(s) \leq \phi(s,q_1) + U_i^2(\tau(s,q_1))$. We get : $U_i^1(\tau(s,q_1)) <$

$U_i^2(\tau(s,q_1))$ which is in contradiction with the hypothesis that $U_i^2(s)$ was the first value to increase. Therefore U_i decreases.

Since U_i decreases and is bounded from below (by zero), it converges.

Let us show that U_i converges in a finite time. Let s be a state, and h the path of shortest length (in number of states) corresponding to a minimum of U. We call l the length of the path h and u_j the series of states of h, such that $u_1 = s$ and $u_l = s_0$. After l scannings of all the states, we have : $U_i(u_1) \leq \phi(u_1,q_1) + U_i(u_2) \leq ... \leq \sum_{j=1}^{l} \phi(u_j,q_j) = U_m(u_1)$. Therefore we have a first result that $U_i(s) \leq U_m(s)$ after l scannings.

We define now a series v_j of states such that $v_1 = s$ and $U_i(v_j) = \phi(v_j,q_j) + U(v_{j+1})$. ϕ is bounded from below by ϕ_m (see demonstration for equation (1.1)), ϕ_m is strictly positive, so v_j converges to s_0 in at most $U_1(s)/\phi_m$ steps. Therefore there is $p \leq U_1(s)/\phi_m$ such that $v_p = s_0$. Then, $U_i(s) = \sum_{j=1}^{p} \phi(v_j,q_j)$. Therefore, $U_m(s) \leq U_i(s)$, at any iteration.

We conclude that after l scannings $U_i(s) = U_m(s)$, that $U_i(s)$ converges in a finite time towards $U_m(s)$. There is a finite number of states, so U_i converges in a finite time $\max_s l(s)$ to U_m.

A.3 Equation 1.5

We want to prove that d_ϕ is a distance.

1. the potential is supposed strictly positive, and $\min \sum_{s=s_1}^{s_2} \phi(s)$ is reached (see demonstration of Eq. 1.2), therefore $d_\phi(s_1,s_2)$ is null if and only if s_1 is equal to s_2.

2. from the commutativity and associativity of the addition, we deduce that $d_\phi(s_1,s_2) = d_\phi(s_2,s_1)$.

3. Let us consider three states s_1, s_2 and s_3. We call h_1 the shortest path between s_1 and s_2, and h_2 the shortest path between s_2 and s_3. Then $h_1 \cup h_2$ is a path between s_1 and s_3. Therefore $d_\phi(s_1,s_3) \leq \sum_{h_1 \cup h_2} \phi(s) = \sum_{h_1} \phi(s) + \sum_{h_2} \phi(s) = d_\phi(s_1,s_2) + d_\phi(s_2,s_3)$. The triangular inequality is verified.

We conclude that d_ϕ is a distance.

A.4 Equation 3.4

Let us show an example where the optimality principle is not verified. We consider the case where the auxiliary function is the global direction followed until now, and take ϕ as a non-linear increasing function of the angle between the global direction and the new decision. We consider two paths h_1 and h_2 between s_0 and s_1, and call s_p the last state before s_1 (see Fig. 43). On h_1 there is no deviation from s_0 to s_p and on h_2 the deviations do not exceed 45 degrees, so we may choose values for ϕ on a neighborhood of 45 degrees such that $U_1(s_p) < U_2(s_p)$: the shortest path between s_0 and s_p is h_1, this is the path selected on

s_p. On the other hand, h_1 contains a right angle between s_p and s_1 while h_2 is straight on this part. Thus we may choose values for ϕ on a neighborhood of 90 degrees such that $U_1(s_1) > U_2(s_1)$. Thus, the optimality principle is not verified for these values of ϕ.

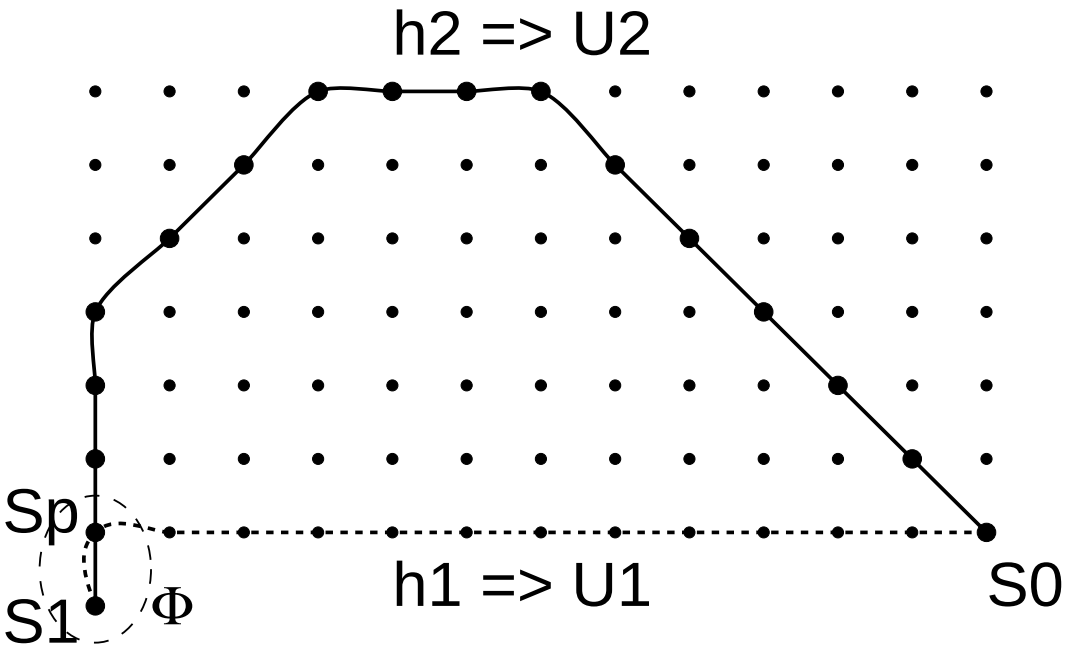

Figure 43: Violation of optimality with an auxiliary function

A.5 Equation 5.4

We want to find if there is a minimum for $U = \frac{\sum_{s=s_0}^{s_1} \phi(s,q)}{\psi(ns)}$ and if it is reached for some path. We consider the set $S = \{\frac{\sum_{s=s_0}^{s_1} \phi(s,q)}{\psi(ns)}\}$. It is non-empty since s_0 and s_1 are connected. It is bounded from below by zero, so it has a minimum.

However, this minimum is not necessary reached if there are loops. We consider for instance a path h_2 between s_0 and S_1, containing $\tilde{n_2}$ states and such that the sum of the potentials along it is $\tilde{U_2}$. Similarly, we consider a loop h_1 having at least one point in common with h_2, having $\tilde{n_1}$ states, and of sum of potentials $\tilde{U_1}$ (see Fig. 44).

We then define U_k as the energy obtained on h_2 by passing k times through h_1. We want to find if U_k can be strictly decreasing, then the minimum is not reached on a finite path.

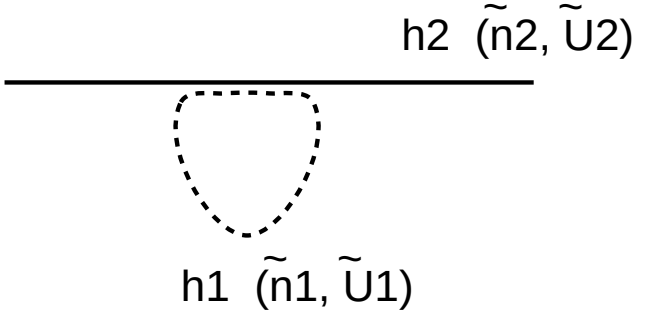

Figure 44: Infinite number of iterations in case of a loop

We note $n_k = \tilde{n_2} + k * \tilde{n_1}$. Then we have : $U_k = \frac{\tilde{U_2}+k*\tilde{U_1}}{\psi(n_k)}$, and $U_{k+1} = \frac{\tilde{U_2}+k*\tilde{U_1}+\tilde{U_1}}{\psi(n_{k+1})}$. Then :

$$\begin{aligned} U_k - U_{k+1} > 0 \Leftrightarrow & \quad (\tilde{U_2} + k * \tilde{U_1}) * \psi(n_{k+1}) - (\tilde{U_2} + k * \tilde{U_1} + \tilde{U_1}) * \psi(n_k) > 0 \\ \Leftrightarrow & \quad (\tilde{U_2} + k * \tilde{U_1}) * (\psi(n_{k+1}) - \psi(n_k)) > \tilde{U_1} * \psi(n_k) \\ \Leftrightarrow & \quad \frac{\psi(n_{k+1})-\psi(n_k)}{\psi(n_k)} > \frac{\tilde{U_1}}{\tilde{U_2}+k*\tilde{U_1}} \end{aligned}$$

If for instance ψ is the identity, then this condition becomes :

$$\begin{aligned} U_k - U_{k+1} > 0 \Leftrightarrow & \quad \frac{\tilde{n_1}}{\tilde{n_2}+k*\tilde{n_1}} > \frac{\tilde{U_1}}{\tilde{U_2}+k*\tilde{U_1}} \\ \Leftrightarrow & \quad \frac{1}{\frac{\tilde{n_2}}{\tilde{n_1}}+k} > \frac{1}{\frac{\tilde{U_2}}{\tilde{U_1}}+k} \\ \Leftrightarrow & \quad \frac{\tilde{n_2}}{\tilde{n_1}} < \frac{\tilde{U_2}}{\tilde{U_1}} \\ \Leftrightarrow & \quad \frac{\tilde{U_2}}{\tilde{n_2}} > \frac{\tilde{U_1}}{\tilde{n_1}} \end{aligned}$$

That is, if there is a loop on which the average of the potentials is lower than the average on the path, then $U_k = \frac{\frac{\tilde{U_2}}{k}+\tilde{U_1}}{\frac{\tilde{n_2}}{k}+\tilde{n_1}}$ decreases and converges when k tends to $+\infty$ towards $\frac{\tilde{U_1}}{\tilde{n_1}}$, which is never reached : the minimum is never reached since it corresponds to an infinite number of states.

Three solutions may be proposed :

1. to impose conditions on ψ such that adding a loop never decreases the energy. This means looking for loops in the image and defining ψ according to the values of the potential on these loops. Obviously, a general solution can not be easily obtained.

2. to detect loops during the integration of the potential. One could for instance compare the integral of the local curvature with the value $2 * \pi$ to detect closed curves, but it is

difficult to recognize small loops (due to the imprecision of the digitalization). Besides, we face the following dilemma : should we store a huge amount of information (past paths) or renounce to optimality in the detection.

3. to avoid loops by restricting the possible decisions during the scanning of the states and decisions. For instance, we consider only 4 directions (corresponding to the angles 0, 45, 90 and 135 degrees) instead of 8. This may be easily implemented, we only need to consider the directions which are compatible with the direction of the feature. An obvious drawback is that the detected feature can have only 4 different local directions.

We assume in the following that there are no loops in any shortest path during the integration. We call N_s the number of possible states and N_q the maximum number of decisions for a state. Then there are at most $N_q^{N_s}$ different paths between S_0 and S_1, not containing loops. Thus, the set $S = \{\frac{\sum_{s=s_0}^{s_1}\phi(s,q)}{\psi(ns)}\}$ is finite, non-empty and bounded from below by zero. Thus it has a minimum which is reached for some path.

However, the optimality principle is not verified. We consider a path h_1 containing n_1 states and of energy U_1, and a path h_2 containing n_2 states and of energy U_2. We suppose that these paths have their extremities in common and that $U_1 < U_2$. We prolongate them with one state, and call the resulting paths h_1^+ and h_2^+, of energies U_1^+ and U_2^+. We look for a condition on the potential ϕ^+ of the added state such that $U_1^+ > U_2^+$.

We have : $U_1^+ = \frac{\psi(n_1)*U_1+\phi^+}{\psi(n_1+1)}$ and $U_2^+ = \frac{\psi(n_2)*U_2+\phi^+}{\psi(n_2+1)}$. Therefore :

$$\begin{aligned} U_1^+ - U_2^+ > 0 \Leftrightarrow \quad & (\psi(n_1)*U_1+\phi^+)*\psi(n_2+1) - (\psi(n_2)*U_2+\phi^+)*\psi(n_1+1) > 0 \\ \Leftrightarrow \quad & \phi^+ * (\psi(n_2+1) - \psi(n_1+1)) \\ & > \psi(n_2)*\psi(n_1+1)*U_2 - \psi(n_1)*\psi(n_2+1)*U_1 \end{aligned}$$

If ψ is the identity and $n_2 > n_1$, this condition becomes :

$$U_1^+ - U_2^+ > 0 \Leftrightarrow \quad \phi^+ > \frac{n_2*(n_1+1)*U_2 - n_1*(n_2+1)*U_1}{n_2-n_1}$$

In the following example (See Fig. 45), we have : $n_1 = 3$, $n_2 = 4$, $U_1 = \frac{3}{3} = 1$, $U_2 = \frac{8}{4} = 2$, $\phi^+ = 20$ and $U_1^+ = \frac{23}{4} = 5.75$ and $U_2^+ = \frac{28}{5} = 5.6$. $U_1 < U_2$ and $U_1^+ > U_2^+$, therefore the optimality principle is not verified.

A.6 Equation 5.5

We want first to justify the expression assigned to tmp in equation 4.5. By definition, $U_n(s_1) = \min_{q_1,\ldots,q_n}\{\sum_{s=s_1}^{s_n}\frac{\phi(s,q)}{\psi(n)}\}$ therefore,

$$\begin{aligned} U_n(s_1) = & \quad \min_{q_1,\ldots,q_n}\{\tfrac{\phi(s_1,q_1)}{\psi(n)} + \tfrac{\psi(n-1)}{\psi(n)} * \textstyle\sum_{s=s_2}^{s_n}\frac{\phi(s,q)}{\psi(n-1)}\} \\ = & \quad \min_{q_1}\{\tfrac{\phi(s_1,q_1)}{\psi(n)} + \tfrac{\psi(n-1)}{\psi(n)} * \min_{q_2,\ldots,q_n}\textstyle\sum_{s=s_2}^{s_n}\frac{\phi(s,q)}{\psi(n-1)}\} \\ = & \quad \min_{q_1}\{\tfrac{\phi(s_1,q_1)}{\psi(n)} + \tfrac{\psi(n-1)}{\psi(n)} * U_{n-1}(s_2)\} \end{aligned}$$

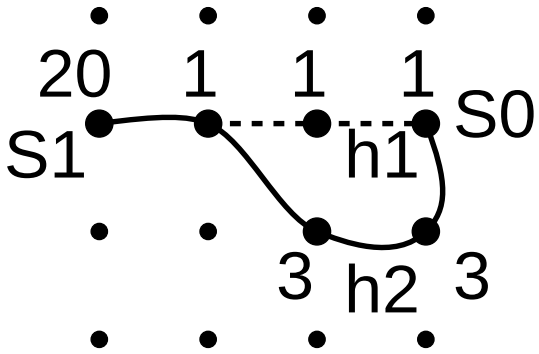

Figure 45: Violation of optimality when solving the bias of the shortest path

which corresponds to the expression assigned to tmp.

We want first to show that U and V are always positive, then that the algorithm converges, and finally that the temporary and final energies U are greater than the energy of the shortest path (due to the non-respect of the optimality principle, there is no equality).

If during the integration U(s) or V(s) are negative, we call $U^-(s)$ and $V^-(s)$ the first values of U(s) and V(s) for which U or V is negative. We have then : $U^-(s) = \frac{1}{\psi(V(\tau(s,q))+1)} * \phi(s,q) + \frac{\psi(V(\tau(s,q)))}{\psi(V(\tau(s,q))+1)} * U(\tau(s,q))$, and every term of the expression is positive, so $U^-(s)$ is positive. On the other hand, $V^-(s) = V(\tau(s,q)) + 1$, so $V^-(s)$ is also positive. We have got a contradiction. Therefore, U and V are always positive.

In equation 4.6, we impose that U decreases. Since it it bounded from below by zero, it converges. We avoid loops, so there is a finite number of possible paths and the algorithm converges in a finite time.

One can notice that the expression 4.5 does not allow any conclusion on the variations of tmp or V, so it is necessary to impose the decreasing of U with equation 4.6.

We want to show now that the energy $U_i(s)$ computed for any state at any step of the integration is always greater than $U_m(s)$, theoretical value of the energy. We consider a state s at a given step of the iteration process. We build then a series t_i by induction :

$$t_0 = s$$

and

$$\begin{aligned} U_i(t_i) = &\quad \frac{1}{\psi(V(t_{i+1})+1)} * \phi(t_i,q) + \frac{\psi(V(t_{i+1}))}{\psi(V(t_{i+1})+1)} * U_i(t_{i+1}) \\ V_i(t_i) = &\quad V_i(t_{i+1}) + 1 \end{aligned}$$

The series $V_i(t_i)$ decreases and is bounded from below, so it converges in a finite number of steps n. Therefore t_i converges towards $t_n = s_0$. We show now by induction on k, from 0 to n, the proposition H(k) :

- $U(t_{n-k}) = \frac{\sum_{i=n}^{n-k} \phi(t_i,q)}{k+1}$
- $V(t_{n-k}) = k + 1$

H(0) is true from the initialization of U and V (if we consider as arbitrary decision on t_n the one giving the smallest potential). If H(k) is true, H(k+1) follows from the definition of the series t_i and from the relation between U_n and U_{n-1} shown as a preliminary result. We conclude : $U_m(s) \leq U_i(s)$. The result is an upper bound of the score of the shortest path.

Unite de recherche INRIA Lorraine, Technopole de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LES NANCY
Unite de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unite de recherche INRIA Rhone-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unite de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unite de recherche INRIA Sophia Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA ANTIPOLIS Cedex

Editeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399