AUXILIARY FUNCTIONS AND OPTIMAL SCANNING FOR ROAD DETECTION BY DYNAMIC PROGRAMMING

N. Merlet

Institute of Comp. Science,
The Hebrew University,
91904 Jerusalem, Israel

J. Zerubia

INRIA, BP 93
2004 Route des Lucioles
06902 Sophia Antipolis Cedex, France

ABSTRACT

Shape information is useful for road detection to improve the correctness and smoothness of the results. Within the frame of dynamic programming, the proposed method stores in an auxiliary image the global direction V(M) followed in the current shortest path. The potential is a function of this image, so that pixels prolongating the current shortest path are favored. The auxiliary image is updated recursively at the same time as the energy, during the optimization.

A variant of this method stores in the auxiliary image the center of the circle tangent to the current shortest path. Another application presented herein computes the average of the potential instead of its sum.

The optimality principle is not verified anymore with the auxiliary functions but they give smoother results without increasing the complexity. Furthermore, several improvements w.r.t. the scanning allow gains of up to 50 % for the computational time.

1. INTRODUCTION

Shape information is useful for road detection when characteristics such as grey-levels and contrasts do not discriminate enough the roads from the background, or to obtain smoothed results.

In [4], we defined a potential on several pixels in order to be able to both integrate curvature and use dynamic programming (DP). This curvature information remains very local, and the complexity becomes too high if we generalize this potential to more than three successive pixels.

In order to take into account information in a farther past and to impose global constraints on the solution with still a low complexity, we must define some memory process. We propose to use a new "auxiliary function" V to do so.

We use the same notations as in Belman's book [1] for the state s (herein a pixel), the decision q which uniquely defines its neighboring successor $\tau(s,q)$, the energy U, and the potential ϕ. The potential is a function of s, q but now also of the auxiliary function V at $\tau(s,q)$. tmp is a temporary test value for U. If tmp is smaller than U, U takes its value, and V is updated relatively to the decision. More formally, we perform until convergence of U :

$$\forall s, \forall q,$$

$$tmp \leftarrow \phi(s,q,\underline{V(\tau(s,q))}) + U(\tau(s,q)) \quad (1)$$

$$tmp < U(s) \Rightarrow U(s) \leftarrow tmp \quad (2)$$

$$\underline{V(s) \leftarrow v(s,q,V(\tau(s,q)))} \quad (3)$$

where the innovations have been underlined, and U is initialized at $+\infty$ except at one extremity of each road, where it is equal to zero. The choice of V and v allows a great variety of applications. We present in the following three possibilities : smoothness of the direction, of the curvature, and minimizing the average of the potential.

2. AUXILIARY FUNCTIONS FOR SMOOTHNESS

In this section, our aim is to obtain smoother results than with ordinary dynamic programming. We see in Fig. 2 the results (in white) obtained without any smoothness constraint : the path is very irregular and is deviated by a white spot at the bottom left.

First we define the auxiliary function as the global direction $\vec{V}$ followed in the current shortest path (see Fig. 1). The potential at a point M is the sum of a grey-level and contrast potential $\phi_1(M,N)$, and of a shape potential $\phi_2(\vec{V}(N), N\vec{M})$, where N is a neighbor of M. ϕ_2 increases with the deviation of M from the line defined by N and $\vec{V}(N)$, and relatively to a smoothness parameter a priori chosen. Thus, pixels prolongating the current shortest path are favored.

$V(\vec{M})$ is recursively defined in Equation (3). It is updated as a linear combination of $\vec{V}(N)$ and of $N\vec{M}$ relatively to a memory parameter. Thus, the location of M updates the global direction of the path. The solution roughly appears as a succession of lines (for more details, see [6]) and the white spot does not deviate the path anymore (see Fig. 3).

In a variant, we define the auxiliary function as the center of the circle tangent to the current shortest path in M. The memory parameter determines the number of points considered to compute the circle. The solution roughly appears as a succession of arcs in Fig. 4 (for more details, see [6]).

3. AUXILIARY FUNCTIONS AND AVERAGE POTENTIAL

A well known problem of dynamic programming is the bias of the shortest path. Until now, we have defined the energy as the sum of the potentials along the path, therefore it depends not only on the values of the potential but also on the number of terms in the sum : paths containing a smaller number of pixels tend to be favored, although they do not necessarily correspond to the feature we are looking for.

Thus, we define the auxiliary function as the number of points in the shortest path, and the energy as the average of the potential. We have to tackle this problem both theoretically and practically : if there is a loop in a path and if the average potential is smaller on the loop than on the path, then the minimum of the energy may never be reached, and the algorithm may never stop. We solve this problem by scanning only part of the couples (state, decision), practically half of the decisions. Of course, such a simple restriction was possible in these examples because the features had a constant global direction, there was no "return" of the roads. With these strong restrictions, loops were impossible.

The curves are longer and less smooth than with simple DP (see Fig. 5 and 6). This shows that the classical definition of the energy as the sum of potentials actually imposes an implicit smoothness constraint. Furthermore, the adequacy of the grey-level and contrast potentials to a given image may now be evaluated without bias of smoothness.

4. OPTIMAL SCANNING

We have compared the number of operations performed on four images with 14 different methods of scannings. All the images have been rotated (modulo $\pi/2$) such that the roads are approximately horizontal. Each of the four images is represented by one of the four black dots in each of the three graphs of Fig. 7. Each graph corresponds to one of the three fastest methods.

The aim is to avoid unnecessary or redundant computations. In particular, the three fastest and independent methods described below allow a gain of up to 50 % relatively to the reference scanning used by [3] :

- eliminating part of the states during integration, well known as prunning, and shown in the middle graph of Fig. 7. When the energy of a state is higher than the energy of the aim, this state is not evaluated. Local prunning is slightly better than global prunning, but it imposes to associate any pixel of the image to the closest path on the basis of its extremities, and optimality may be lost. Local prunning is different from global prunning only when there are several roads in the image.

- ordering the states at initialization, shown in the right graph of Fig. 7. This is a surprising point : scanning with the external loop along the rows and the internal loop along the columns does not have the same performance as scanning first the columns and then the rows, and this depends upon the orientation of the roads (horizontal or vertical). A general algorithm is used in the case where the direction of the road (horizontal or vertical) is not a priori known. It alternates external loops along the rows and along the columns. The reference scanning [3] is shown in thick, the external loop is along the rows. In the four images considered herein, the direction of the roads is roughly horizontal, so that an external loop along the columns leads to the fastest convergence.

- stopping the iterations before convergence, shown in the left graph of Fig. 7. The final iterations are used in these examples only to check optimality, not to modify the solution anymore. They may be skipped after using a heuristic criterion, for instance when the path has not changed position within two successive iterations.

These three methods are independent, so they may be combined and their gains cumulated.

Other methods did not improve the results, such as :

- eliminating part of the states at initialisation, scanning only regions of interest. Then, a compromise has to be found between speed and rightness of the result, which heavily depends on the chosen windows.

- ordering the states during iteration, according to energy value or time of last change. The computations were worse than with the reference scanning [3].

5. DISCUSSION AND CONCLUSION

Two pionnering works should be quoted. When Sha'ashua and Ullman [7] compute the saliency map in binary images, the result of one iteration is taken into account in the potential at the next iteration. Auxiliary functions are therefore used implicitly, although the concept is not formally defined.

The other work is from Danielson [2], by computing an octagonal distance, a mixing of 4-connectivity and 8-connectivity. Here too, the value of the potential depends on the oddness of the energy, which is actually the auxiliary function in this application.

Another application may be mentionned for the use of auxiliary functions : computing maps of the Mahalanobis distance [5] for stereovision. The algorithm is a variant of Danielson's algorithm, where the auxiliary function contains the label of the nearest point of the object.

Finally, we should notice that the optimality principle is not verified anymore and the path found is not necessarily optimal. In spite of this problem, the roads found are indeed globally smoothed with constraints of direction or curvature. The complexity is similar to dynamic programming without taking shape into account.

6. REFERENCES

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

[2] P.E. Danielson, Euclidean Distance Mapping, *CGIP* 14, 227-248, 1980.

[3] 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.

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

[5] N. Merlet, Integration of Global Information for Features Matching in Stereo Vision and for Roads Detection in Satellite Images. Ph. D. Thesis, Hebrew University of Jerusalem, 1996.

[6] N.Merlet, J.Zerubia, Integration of Global Information for Roads Detection in Satellite Images, joint research report 3239, Hebrew University/INRIA, August 1997. (ftp.inria.fr/INRIA/publication/RR/RR-3239.ps.gz)

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

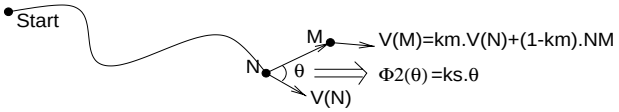

Figure 1: Computation of the global direction

Figure 2: Global direction with a memory parameter equal to 0.9 and a smoothness parameter equal to 10 (almost no direction constraint).

Figure 3: Global direction with a memory parameter equal to 0.9 and a smoothness parameter equal to 400.

Figure 4: Global curvature with a memory parameter equal to 30 pixels and a smoothness parameter equal to 400.

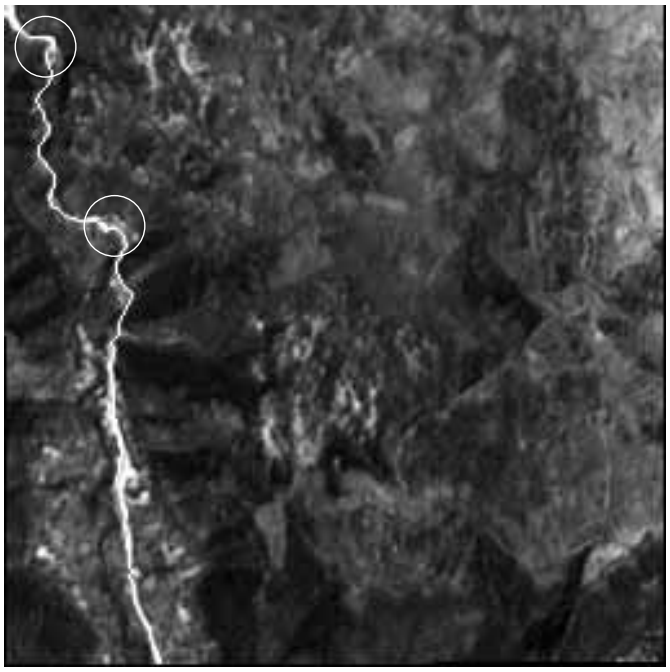

Figure 6: Minimizing the average of the potential.

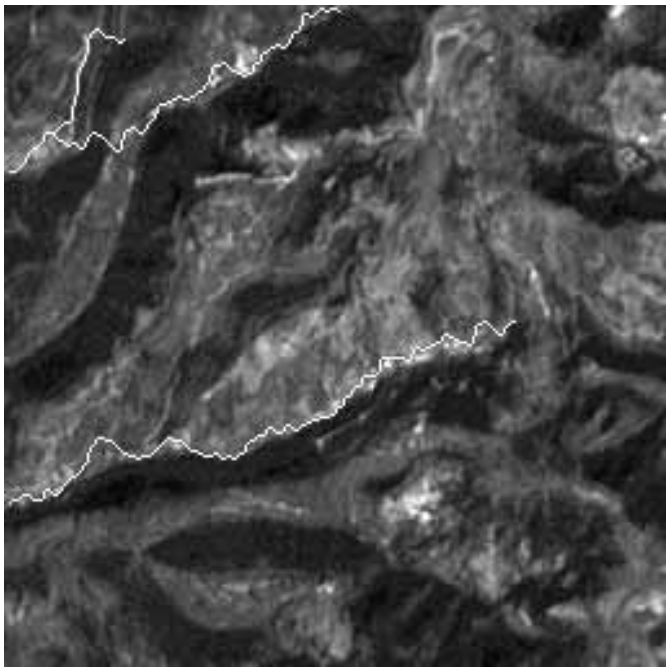

Figure 5: Minimizing the average of the potential.

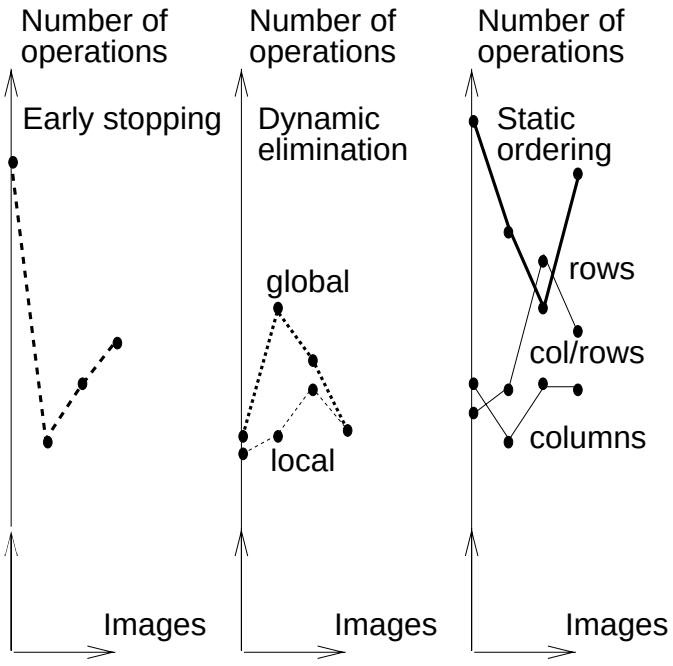

Figure 7: Comparison of different scanning organisations. Left : stopping the iterations before convergence. Middle : prunning. Right : with different orders of scanning.