A VITERBI ALGORITHM FOR IMAGE EDGE DETECTION

I.Pitas
Department of Electrical Engineering
University of Thessaloniki
Thessaloniki 54006
GREECE

ABSTRACT

Two dimensional random fields have been proposed as stochastic models of real world images. Random fields can also be used for the modelling of image edges. Such models are useful in edge detection especially when the image is noisy, because they incorporate part of our knowledge about the edge distribution in the image. In particular, the use of Markovian random fields introduces the dependence of the existence of an edge element on the existence of an edge element in the neighbouring image pixels. An algorithm is proposed which makes an optimal decision on the existence of an edge element in an image pixel by taking into account the image intensities and the contributions of the adjacent image pixels. The algorithm is very similar to the Viterbi algorithm, which is very well known in the communications.

1. INTRODUCTION

One of the most important tasks in image analysis is the recognition of objects in the image. Many objects are very well described by the shape of their boundaries. These boundaries tend to show up as intensity discontinuities in an image. The local image grey level discontinuities are called edge elements. A continuous series of edge elements forms an edge which is usually part of the boundary of an object.

There exist several algorithms for edge detection [1]. They can be grouped in three main classes:

1) Local edge operators

2) Global edge detection algorithms

3) Relaxation techniques.

The first class includes the edge operators that detect local intensity discontinuities. The second class includes algorithms that use global information to decide on the existence of an edge in a particular pixel. The edge relaxation techniques improve edge operator measurements by adjusting them, based on measurements of the neighbouring edges. The relaxation techniques are usually iterative and they are computationally intensive. Their results are usually very satisfactory.

Our approach is to use stochastic edge models for edge detection. The detection of an edge element at a particular edge pixel is based on two information sources:

a) the apriori information from the stochastic edge model which incorporates the influence of the neighbouring pixels

b) the information of the image intensities.

The stochastic edge model used is the two-dimensional discrete Markov process. Such a model can easily incorporate the influence of the pixels of the immediate past on the pixel of the present (the notions of present and past will be explained in the next section). This model leads to an algorithm for edge detection line-by-line. The edge detection algorithm for each line is similar to the Viterbi algorithm used in the communications.

The outline of the paper is as follows. Section 2 describes the random edge model. The use of this model in edge detection is described in section 3. The edge detection algorithm is described in section 4. Examples and conclusions are given in section 5.

2. RANDOM EDGE MODEL

An edge element is described by its direction which is aligned with the direction of the maximal local gray-level change and its magnitude which describes the severity of this change. We assume that an edge can have only K discrete directions (eg. along the axes of 0, 45, 90, 135, 180, 225, 270 and 315 degrees). We denote by x_{ij} a dicrete random variable which describes the existence of an edge element at pixel (i,j). This random variable may take the values 0,1,..,K. The values 1,..,K denote the possible edge orientation. The value 0 denotes the absence of an edge element in the pixel (i,j). Edges are randomly distributed in an image. Therefore x_{ij} form a two-dimensional random field denoted by a vector x of dimension NM if the image has dimensions NxM. A reasonable model already used in the description of natural images is the discrete two- dimensional Markov process [3,4,5]. This model can also be used as an edge model, because in real cases the existence of an edge in an image pixel depends only on the existence of an edge in its immediate neighbourhood. This property is extensively used in the relaxation techniques [2]. We shall give a definition of the discrete 2-d Markov process.

We define an image line $I = \{(I,j), j=1,M-1\}$ as

'present'. We define as 'past' all the image lines i, 1<i<I. The line I-1 is called 'immediate past'. The lines i, 1<i<N are defined as 'future'. We denote by $\underline{x}_I$, $\underline{x}_{I-1}$, the vectors of the random variables x_{ij} in the lines I, I-1 and by $p(\underline{x}_I)$, $p(\underline{x}_{I-1})$ their probability distributions. The random field x_{ij} is called Markovian if the probability distribution of the 'present' vector depends only on the probability distribution of the vector of the 'immediate past':

$$(1) \quad p(\underline{x}_I) = p(\underline{x}_I|\underline{x}_{I-1})\,p(\underline{x}_{I-1})$$

The definition given here is a special case of the definition given in I3I. The only difference is that we use lines as boundaries instead of the rectangular boundaries used in I3I. The Markov property is stated as probability propagation from boundary to boundary both here and in I3I. The vector $\underline{x}_I$ of a Markov process may take (K+1)**M different values, which are called states of the process. The transition in the Markov model is associated to the so called state diagram shown in Figure 1. The arrows of the diagram denote transitions and the numers near the arrows denote the corresponding transition probabilities. Another redundant description of the Markov process is the so called trellis I6I, shown in Figure 2. Each node corresponds to a distinct state of an image line and each branch corresponds to a transition of a state of an image line to the state of its next line. The trellis begins and ends with the known states $\underline{x}_1, \underline{x}_N$. These states are usually assume to be the zero states in the edge detection because no edge exists at the lines 1 and N of the image.

It will be shown in the next section that the model described by (1) is not very convenient because of its computational complexity. Therefore simpler edge models must be defined. Such a convenient model is the one described by:

$$(2) \quad p(x_{ij}) = p(x_{ij}|x_{i,j-1}, \underline{x}_{ijp})\,p(x_{i,j-1}, \underline{x}_{ijp})$$

where $\underline{x}_{ijp}$ is the vector:

$$\underline{x}_{ijp} = \left[x_{i-1,j-1}, x_{i-1,j}, x_{i-1,j+1}\right]^T$$

This model states that the probability distribution of the random variable x_{ij} depends on the probability distributions of the random variables $x_{i,j-1}$, $x_{i-1,j-1}$, $x_{i-1,j}$, $x_{i-1,j+1}$ of its immediate neighbourhood, corresponding to image pixels from the 'past' of the pixel (i,j) (if the image is scanned line by line). This model is only a special case of (1). It will be efficiently used in edge detection.

3. EDGE DETECTION

The problem of edge detection is stated as follows. Given an observed image $\underline{y}$, which is the original image plus additive white noise, estimate the edge distribution $\underline{x}$ of the original image. It is known that the edges of the original image are a Markovian random field described by (1).

We shall use the Bayesian approach and we shall try to maximize $p(\underline{x},\underline{y})$:

$$(3) \quad p(\underline{x},\underline{y}) = p(\underline{x})\,p(\underline{y}|\underline{x})$$

where $p(\underline{y}/\underline{x})$ is the conditional probability density function of having image intensity $\underline{y}$ given the edge element vector $\underline{x}$ of the original image. Since additive white noise is assumed, the following assumption is reasonable:

$$(4) \quad p(\underline{y}|\underline{x}) = \prod_{i=1}^{N} p(\underline{y}_i|\underline{x}_i) = \prod_{i=1}^{N}\prod_{j=1}^{M} p(y_{ij}|x_{ij})$$

According to (1),(3-4) the probability p(x,y) is given by:

$$(5) \quad p(\underline{x},\underline{y}) = p(\underline{x}_1)\prod_{i=2}^{N} p(\underline{x}_i|\underline{x}_{i-1})\prod_{i=1}^{N} p(\underline{y}_i|\underline{x}_i)$$

By taking the logarithm, (5) becomes:

$$(6) \quad \ln p(\underline{x},\underline{y}) = \ln p(\underline{x}_1) + \ln p(\underline{y}_1|\underline{x}_1) + \sum_{i=2}^{N}\left[\ln p(\underline{y}_i|\underline{x}_i) + \ln p(\underline{x}_i|\underline{x}_{i-1})\right]$$

where the length:

$$(7) \quad \lambda_i = \ln p(\underline{y}_i|\underline{x}_i) + \ln p(\underline{x}_i|\underline{x}_{i-1})$$

is assigned to each transition of the trellis. The problem of finding the optimal edge distribution x is equivalent to the problem of finding the shortest route in the trellis. This problem can be solved by the Viterbi algorithm (to be discussed in the next section). However, the number of states of the Markov process is $(K+1)^M$, which is an extraordinary number for practical applications. A trementous computational effort is required for the application of the Viterbi algorithm in such cases. Furthermore, values must be given to an extremely high number of transition probabilities $p(\underline{x}_i/\underline{x}_{i-1})$. These reasons lead us to the use of the Markovian model (2) and to the maximization of the joint probability $p(\underline{x},\underline{y})$ line by line.

The probability to be maximized is given by:

$$(8) \quad p(\underline{x}_i,\underline{y}_i) = p(\underline{x}_i)\,p(\underline{y}_i|\underline{x}_i)$$

According to (2), relation (8) becomes:

(9) $p(x_{ij},y_{ij})=p(y_{ij}|x_{ij})p(x_{ij}|x_{i,j-1},\underline{x}_{ijp})p(x_{i,j-1},\underline{x}_{ijp})$

Since the edge detection is done line by line, the optimal vector $\underline{x}_{i-1}$ is known to be $\underline{a}_{i-1}$. In this case vector $\underline{x}_{ijp}$ of (9) is equal to vector $\underline{a}_{ijp}$ and (9) becomes:

(10) $$p(\underline{x}_i,\underline{y}_i)=p(x_{i1})\prod_{j=2}^{M} p(x_{ij}|x_{i,j-1},\underline{a}_{ijp})\prod_{j=1}^{M}p(y_{ij}|x_{ij})$$

In this method we use an 1-d Markov process, whose transition probabilities $p(x_{ij}/x_{i,j-1},\underline{a}_{ijp})$ depend on the already made decisions in the previous line, instead of using a 2-d Markov process. The number of states in such a Markov process is (K+1). This random process can be described by a state diagram whose transition probabilities vary. Such a two-state diagramm is shown in Figure 1. It can also be described by a trellis, similar to the one shown in Figure 2. Each node corresponds to a state x of a pixel (i,j). The extreme pixels (i,1), (i,M) of each line possess no edge and their corresponding state is 0. By taking the logarithm of (10), we have:

(11) $$\ln p(\underline{x}_i,\underline{y}_i) = \ln p(x_{i1}) + \ln p(y_{i1}|x_{i1}) + \sum_{j=2}^{M}\left[\ln p(y_{ij}|x_{ij}) + \ln (x_{ij}|x_{i,j-1},\underline{a}_{ijp})\right]$$

The following length is attached to each transition of the trellis:

(12) $$\lambda(x_{ij},x_{i,j-1})=\ln p(y_{ij}|x_{ij})+\ln p(x_{ij}|x_{i,j-1},\underline{a}_{ijp})$$

Again, the problem of finding an optimal vector $\underline{x}_i$ is equivalent to finding the shortest route in the trellis, which is solved by the Viterbi algorithm.

3. THE VITERBI ALGORITHM

The Viterbi algorithm has extensively been used in communications theory, e.g. in convolutional coding and intersymbole interference [6]. In this paper we shall taylor this algorithm to the needs of edge detection, and especially to the formulation described by (10). A very similar algorithm can be developed for the problem formulation described by (5).

We denote by $\underline{x}_{Ik}$ the vector $(x_{I1}, x_{I2},.., x_{Ik})^T$ which corresponds to the pixel (I,k). Such a vector denotes a specific path in the trellis which starts at node 1 and stops at one of the K+1 states of the pixel (I,k). A length $\Lambda(\underline{x}_{Ik})$ is associated to this vector:

(13) $$\Lambda(\underline{x}_{Ik}) = \sum_{j=1}^{k}\lambda(x_{Ij},x_{I,j-1})$$

where $\lambda(x_{I,j},x_{I,j-1})$ are the lengths of the transitions which correspond to this path. The vector $\underline{x}_{IM}$ corresponds to a path in the trellis from node 1 to node M. One of the paths starting form node 1 and stoping at a state of pixel k has the minimal length and it is called 'survivor'. The corresponding vector $\underline{x}_{Ik}$ is denoted by $\hat{\underline{x}}_{Ik}$. The shortest complete path $\hat{\underline{x}}_{IM}$ must include one survivor which corresponds to one of the K+1 states of the pixel k. Therefore if we want to find the complete path $\hat{\underline{x}}_{IM}$, we can find the survivor of each state of pixel k. Based on these suvivors, we can find the survivors of each state of pixel k+1 by adding to the lengths of the survivors $\hat{\underline{x}}_{Ik}$ the lengths of the transitions from the states of pixel k to the states of pixel k+1 and by choosing the path having the minimal length as a survivor. This procedure can be repeated recursively, until we find the shortest complete path $\hat{\underline{x}}_{IM}$. The algorithm is a simple version of forward dynamic programming. Its formal statement is as follows.

Initialization: $k = 0$

$$\hat{\underline{x}}_{I1} = \underline{0}$$

$$\Lambda(\hat{\underline{x}}_{I1}) = 0$$

Recursion:

Compute

$$\Lambda(x_{I,k+1}, x_{Ik}) = \Lambda(\hat{\underline{x}}_{Ik}) + \lambda(x_{I,k+1},x_{Ik})$$

for all transitions $(x_{I,k+1}, x_{Ik})$.

For each state $x_{I,k+1}$ find x_{Ik} such that:

$$\Lambda(\hat{\underline{x}}_{I,k+1}) = \min_{x_{Ik}} \Lambda(_{I,k+1},x_{Ik})$$

Find the vector $\hat{\underline{x}}_{I,k+1}$:

$$\hat{\underline{x}}_{I,k+1} = (\hat{\underline{x}}_{I,k},x_{I,k+1})$$

The vector $\hat{\underline{x}}_{IM}$ found gives the optimal choice of edge elements on the image line I. The same algorithm with slight modifications can be used for the problem formulation (5).

4. EXAMPLES

We have tested the algorithm in the image shown in Figure 3a. In this example, we check only the existence or not of an edge element at an image pixel. We are not interested in the directions of the edge elements. The result of the Sobel edge detector, after appropriate thresholding, is shown in Figure 3b. The Viterbi algorithm described in the prevous section has been used for edge detection. The detection has been done line by line and it has been based on (10). The transition probabilities are chosen apriori. The probabilities $p(y_{ij}/x_{ij})$ are chosen in the following way. A known edge detector is applied to the image

(eg. Sobel operator) and produces an image having intensity e_{ij} in pixel (i,j) and histogram $p(e_{ij})$. The following choice is made:

$$(14)\quad p(y_{ij}|x_{ij}=1) = \frac{e_{ij}}{\max_{(i,j)} e_{ij}}$$

$$p(y_{ij}|x_{ij}=0) = 1-p(y_{ij}|x_{ij}=1)$$

The edges detected by the Viterbi algorithm, are shown Figure 3c. The results of the Viterbi algorithm are clearly superior to that shown in Figure 3b, because this algorithm takes into account the information of the adjacent pixels.

REFERENCES

1. W.K.Pratt 'Digital Image Processing', Wiley Interscience, 1978.

2. D.H.Ballard, C.M.Brown 'Computer Vision', Prentice Hall, 1982.

3. E.Wong 'Recursive filtering of two-dimensional random fields', IEEE Transactions on Information Theory, pp.84-86, Jan. 1975

4. J.W.Modestino, R.W.Fries 'Construction and properties of a useful two-dimensional random field' IEEE Transactions on Information Theory, vol.IT-26, pp.44-50, Jan. 1980.

5. D.S.Lebedev 'Probabilistic characterization of images in filtration and restoration problems', Signal Processing: Theories and applications (M.Kunt, F. de Coulon editors), pp.55-64, North Holland, 1980.

6. G.F.Forney 'The Viterbi algorithm', Proc. IEEE, vol.61, pp.268-278, March 1973.

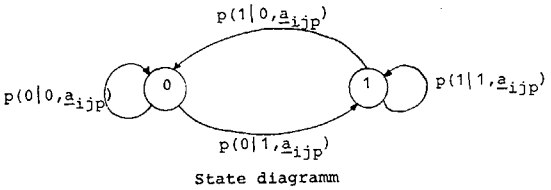


Figure 1: State diagram of a two-state Markov process describing the edge distribution on an image line, with the transition probabilities varying according to the information of the previous line

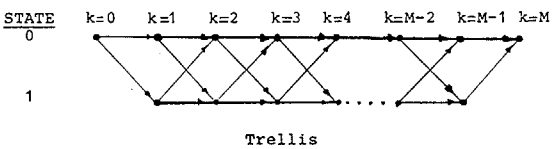
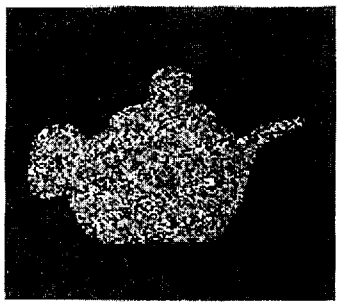
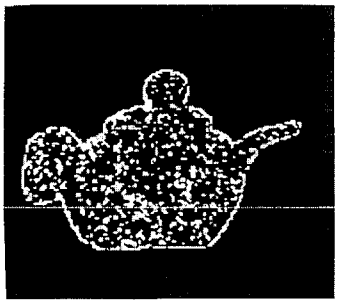


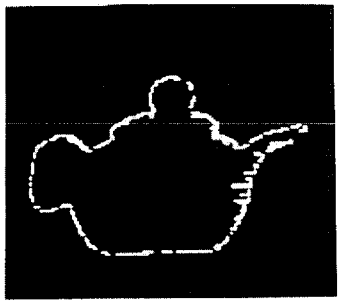
Figure 2: Trellis of a two-state Markov process



(a)



(b)



(c)

Figure 3: (a) Test image corrupted by white Gausian noise.
(b) Result of the Sobel edge detector.
(c) Result of the Viterbi edge detector.