Stochastic Edge Detection Based on Discrete Segments

P. Bertolino *
Laboratoire TIMC-IMAG
Institut Albert Bonniot, Domaine de la Merci
38706 La Tronche cedex FRANCE

A. Montanvert †
LIP - ENS
46, allée d’Italie
69364 Lyon cedex 07 FRANCE

Abstract

We present an original edge detector which provides a binary contour element map by looking for significant discontinuities along discrete segments which are computed in the image. The size and the direction of those segments are chosen randomly and thus, discontinuities may be evaluated with many different resolutions. In order to extract a potential discontinuity within each segment, we propose an adaptive criterion which only takes into account the pixels on this segment. This process is iterated many times, allowing to find weak edges in short segments and strong edges in longer segments, according to the criterion. Some results on different kinds of images are given.

Key Words

Image Segmentation, Edge Detector, Stochastic Process, Random Segments.

*E.Mail: Pascal.Bertolino@imag.fr - Tel 76 34 95 89 - FAX 76 34 95 49
†E.Mail: Annick.Montanvert@lip.ens-lyon.fr
1 Introduction

In order to study and understand the content of an image, a process called segmentation has to extract from the image its different entities. This process can be done by looking for either homogeneous regions (each one corresponding to an entity) or discontinuities (borders of the entities).

In this paper, we deal with the second approach called edge detection. Many of the existing methods [PM82] consist of several steps which are mainly:

1. A convolution between the image and an appropriate mask in order to remove the noise and enhance the discontinuities. This process provides a grey-level image where high frequencies (i.e. contrasts) appear lighter whereas homogeneous regions remain dark. The convolution mask may be simple [Rob65], [Pre70] or more complex [Der87], [SC86]. The size of the mask, i.e. the size of the subwindow, corresponds to a first implicit threshold.

2. A thresholding applied on the previous image with a thinning process gives a binary contour map. In the case of hysteresis thresholding [Der87], edges are kept or removed according to their length and their strength. Anyway, if the threshold is too high, the contour map will contain a lot of discontinuities even if they are not significant. Conversely, if the threshold is too low, only the strong edges will remain. Most of the time, static thresholds are applied to the whole image and there is no adaptivity to its local content. The methods which give the best results use several parameters that have to be tuned precisely for each kind of image.

3. A close contour process enables to close objects so that the final result can be used by a particular application [bio-medicine, robotics, satellites, etc]. This step may take into account a priori knowledge about the number of objects, their size, shape and possible occlusions.

Because of the complexity of getting appropriate tools and criteria to study 2D environments [Pav92] such as windows in an image, we present an algorithm which detects features using 1D supports: The segments (or flat windows).

In order to get rid of thresholds and to adapt the detection to the image, we propose a stochastic approach which can compute low-level information without the help of the user. In this approach, we replace the first and second step described above.

A few years ago, Quiguer and Al [QMD91] proposed an adaptive thresholding based on the statistic analysis of the lines of an image. Our main idea is to choose randomly a set of contiguous pixels in the image (segments are well suited) and, only using the knowledge of those pixels, look within this set to detect if it contains a significant discontinuity (figure 1).

![Figure 1: Example of 3 segments on a pixel grid](image)

In order to achieve our goal, we need to choose a set of segments to be processed in the image grid and to find which pixels of each segment form a potential discontinuity element.

In section 2, we explain how we choose the segments. Then, in section 3, we discuss about the discontinuity criterion to use. In section 4 we bring some improvements to the basic algorithm. At last, in section 5, we give experiment results on several test images.
2 A Stochastic segment algorithm

2.1 Principle

A digitized image $I$ is a matrix of pixels on a grid. Within this grid, we can define a finite set $S$ of all the possible discrete segments (figure 1).

For a $N \times N$ image, the number of segments (of length $1 < l \leq N$) is $|S| = (N^2 \times (N^2 - 1))/2$. Each segment $s \in S$ is a connected set of arranged pixels (figure 1), numbered from 0 to $l - 1$: $s = \{s_0, \ldots, s_{l-1}\}$. The sense of the segments and thus the sense the pixels are numbered is arbitrary chosen. We define a mapping between $S$ and $F$, the set of discrete 1D functions (called profiles and noted $f(x), x = 0, \ldots, l - 1$) of gray levels of the pixels crossed by the corresponding segment of length $l$. (figure 2). Along each segment $s \in S$, and thus inside each associated function $f \in F$, we would verify if there exists a significant discontinuity. Each pixel of the image may be crossed by $O(N^2)$ segments, from length 2 to length $N$. By this way, the contribution of each pixel (to an homogeneous or heterogeneous area) and its value are seen through many resolutions and many spatial distributions. Going on with this approach, one main problem appears: $|S|$ whose complexity is $O(N^4)$ prevents any implantation on a computer because of the execution time.

So, instead of processing the whole $S$, we take a subset $S'$ whose $s$ are randomly chosen in $S$.

In order to get a subset $S'$ we randomly choose couples of points whose junction forms segments. We use Bresenham algorithm [Bre65] to build them. This algorithm gives the best discrete segment which joins two points on a square grid. Moreover, it is very fast and easy to implement. On each segment, a criterion $C$ can be evaluated upon the values of $f(x)$ to know if the segment crosses an edge.

2.2 Algorithm

The process described above is repeated until a stabilized state is reached. In our experiments, on $256 \times 256$ images, we iterated the process as many times as the size of the image. The basic algorithm is the following:

Initialize the contour map with no contour

WHILE (the process is not stabilized) DO
BEGIN

Pick up randomly 2 points A and B (A ≠ B) in the image
Build the segment \( S = [A, B] \)
Derive \( f \) from \( S \)
Try to find \( f(i) \) and \( f(i + 1) \) which verify \( C \) (see section 3)
IF \( (f(i), f(i + 1)) \) exists
THEN Add the pixel \( S_i \) or \( S_{i+1} \) to the contour map
END.

This detector provides a binary image of the contours: Either a pixel is selected one or more times by the algorithm, or it is not. In the first case, the pixel belongs to the final contour map; in the second case, it does not. The use of the stochastic operator allows weak edges to be retained when they are crossed by short segments. Longer segments detect eventual stronger edges. No length of segment is preferred and then any part of the image is treated at different resolutions.

3 Adaptive edge detection

\( C \) can be either a single criterion or a union of criteria based on heuristical or statistical rules. \( C \) must not include static threshold or parameter in order to have an auto-adaptive process.

Let us give \( C_1 \), a basic criterion which finds two adjacent pixels which generate the maximum contrast value along each segment. More formally:

\( C_1 \) is verified for \( (f(k), f(k + 1)) \) if

\[
|f(k) - f(k + 1)| = \max_{i=0,\ldots,2^n-2}(|f(i) - f(i + 1)|) = MG > 0
\]

If \( C_1 \) is verified, then \( s_k \) or \( s_{k+1} \) is marked on the contour map. This criterion is computed very easily and very quickly. But it provides mediocre results: It assumes that each segment crosses an edge if it contains one or more contrast > 0. Moreover, it is very noise sensitive. In order to take into account the information given by the pixels of the segment, we choose \( C \) as following:

Compute the standard deviation \( \delta \) of elements of \( f \)
Detect \( k \) such as \( (f(k), f(k + 1)) \) verifies \( C_1 \)
IF \( MG > \delta \)
THEN \( C \) is verified for \( (f(k), f(k + 1)) \)

Only one of the two pixels is assigned to the contour map:

IF \( |f(k + 1) - f(k)| > |f(k + 1) - f(k + 2)| \)
THEN Add the pixel \( s_k \) to the contour map
ELSE Add the pixel \( s_{k+1} \) to the contour map.

Note that along one segment, several couples \( (f(k), f(k + 1)) \) may verify the criterion.

4 Improvement of the basic algorithm

The basic algorithm can be improved both with a better quality of the subset of segments to process \( S' \) and the quality of the criterion \( C \).
4.1 Distribution of the segments

In the basic algorithm, contours at the periphery of the image are underprivileged (figure 3.b) because since the chosen segments extremities are uniformly distributed in the image the segments distribution is normal: The more distant a pixel is from any border of the image, the higher is the number of segments crossing it (figure 3.a). To eliminate this problem, we could modify the distribution of the couples of pixels which defines each segment (from a normal distribution to a uniform one). We have chosen to put the image in the middle of an image large enough to avoid border effects. The whole constitutes the new image $I'$ to be treated ($I$ is included in $I'$). When a segment contains both pixels of the image $I$ and pixels of the border ($I' - I$), only the pixels of the image $I$ are submitted to the criteria $C$. Hence the bias due to the distribution is limited (figure 3.c).

But since the segments are discrete, another bias is introduced: all the directions do not have the same probability to be chosen [Kra91], [DD84]. This problem remains one of our future investigation.

![Figure 3: Problem of the distribution of the segments](image)

4.2 Noise and texture

According to $C$, our method is still noise sensitive. To strengthen a potential edge pixel, we must use information given by the parallel neighbor segments: When a couple $(f(k), f(k + 1))$ verifies criterion $C$, we check that the two parallel neighbor couples (one on each side, in the direction of $[s_k, s_{k+1}]$) generate the same gradient sign that $(f(k), f(k + 1))$. If so (figure 4.a), the potential edge pixel is marked on the contour map, otherwise (figure 4.b) the contour map is not updated.

4.3 Thin edges

The basic algorithm provides thick edges because several segments of similar direction or of different length may cross an edge extracting several close contour elements more or less in the gradient direction (figure 3.c). To avoid this shortcoming, when an edge point is marked in the contour map, its two neighbors belonging to the segment (one on each side) become forbidden as a future edge point. When a couple $(f(k), f(k + 1))$ is found as a potential edge component, $C$ criterion must choose $s_k$ or $s_{k+1}$ to represent the edge point in the contour map. In order to implement this
mechanism, we define forbidden edge pixels: when a pixel $s_k$ of the segment is selected, $s_{k-1}$ and $s_{k+1}$ take the flag "forbidden" in the contour map (figure 5.a). If latter within another segment, $s_{k-1}$ or $s_{k+1}$ is chosen as an edge pixel, it will not be marked as a contour point, because it has been forbidden during the process of a previous segment. On the other hand, this segment will generate in its turn two forbidden edge pixels if they are not already marked as edge pixels. Although this process may introduce a bias because all the segments do not have the direction of the gradient, the result is better but far from being perfect (figure 5.b).

5 Experimental results

We give some experimental results (figure 6) to show the advantages and the inconveniences of the stochastic operator and the used criterion. These results do not want to make a comparison between our new approach and existing operators, but rather to try to open a new investigation field.

Figure 6 shows the results obtained with the algorithm using the three improvements described above. We can see that significant weak edges are found as strength edges are (figure 6.a, the edge separating the two black cells looking like a butterfly). No region seems to be underprivileged. The noise process is not enough efficient for very noisy images (figure 6.c). Thin contours are obtained correctly, but their process may be a problem at the location where many contours are crossing. Some other problems still remain and are not processed by the algorithm: Edges are often dotted.
Figure 6: Original images and results
(figure 6.b) and thus difficult to interpret by further processes. Locally, the strong edges often disadvantage weak edges when both are close and rather parallel. Strong edges “attract” a lot of local potential edge points (figure 6.b).

6 Conclusion

In this paper, we have presented a stochastic edge detector based on discrete segments, which provides binary contours without any threshold. Its adaptivity allows to take into account various kinds of edges at multiple resolutions. We have given experimental results to show the potential and the weakness of the operator as it is currently used.

Some improvements are foreseen: The quality of the results can be improved by the closure of edges, by a better noise immunity and by using a union of complementary criteria.

More precisely, we orient our investigation towards four main directions:

- To have a better distribution of the segments through the image.
- To reduce the noise effect by smoothing the values of the pixels belonging to the segment to treat with 1D convolving masks (whose coefficients could vary with the length of segments).
- To have a better robustness of the discontinuity elements (by taking into account the number of times each one is extracted during the process and by trying to find the discontinuity in the direction of the local gradient).
- To analyse the edges of the image at different resolution using different ranges of segment length.

This detector can be used in cooperation with other treatments: The stochastic operator was used at the bottom of a pyramidal segmentation process [MMR91] in order to make a contour-region cooperation. Before the construction of the pyramid, it enriches the adjacency graph (built from the regular tessellation on the initial image) by weighting its edges with the value of lines process generated by the edge couples $(s_i, s_{i+1})$. Then, this information is updated bottom up (summed up on the new implicit edges generated by the regions, eliminated on the edge between two regions which have merged) and used at each level of the pyramid [MB92].

References


