Edge-Based Image Segmentation with MOVels

Giancarlo Iannizzotto
Department of Mathematics,
University of Messina - Italy.
E-mail: ianni@ingegneria.unime.it

Lorenzo Vita
Istituto di Informatica e Telecomunicazioni,
University of Catania, - Italy.
E-mail: vita@iit.unict.it

Abstract

We present a new tecnique for edge-based segmentation, which is fast and accurate even in presence of nested contours. The shape characteristics of the contours are perfectly recovered, with no smoothing, and only closed contours are returned; this leads to a well-defined segmentation of the images. The algorithm is easily parallelizable, and is well-suitable for specialized, real-time hardware. Experimental results are presented, in comparison with the Canny algorithm.

1. Introduction

Segmentation (that is, partitioning of an image into separate areas whose points can be characterized according to some specific properties) is often the first step for image analysis processes. The edge-based techniques of segmentation are based on the identification of the *border points*, at which abrupt changes in some specific quantities (for example, either in the level of brightness, or in texture, or in colour) occur. The process consists of three stages:

- Preprocessing: filtering of noise, equalization of brightness levels, edge enhancement...
- Edge Detection: application of one or more operators tending to the identification of the points with high *edgeness* (that is, those more likely to be contour points of the objects);
- Edge Linking: to link edges into edge chains that better match with the borders in the image.

Active contours are models representing open or closed curves that, having a very high ability of morphological adaptation, can almost perfectly adapt to contours of any shape. These models are called *active* because they automatically respond to specific characteristics of the points of the image to which are applied, by changing their shape consequently. For example, an active contour can respond to the edgeness values of the points of the image, by changing its shape till it passes only for all the points characterized by values of edgeness locally higher (*edgeness local maxima*). A particular type of active contour is the *snake*, that does not only respond to the characteristics of the points of the image (through the minimization of a quantity called external energy), but also responds to specific internal laws ruling shape and way of deformation tending to minimize a quantity called internal energy [1]. Some other variations of the snake are represented by its parametric formulations, whose purpose is to change the problem of the minimization of internal/external energy into a problem of estimate of parameters (the control points of a spline, for example), much quicker and computationally less expensive (snake spline [2]). Snakes are generally used as semi-automatic techniques of segmentation, because, on conditions of high noise, they can *lose contact* with their primary target, if they are not properly helped by interactive user operations. On the other hand, in some recent works with active contours very good results were obtained in the field of automatic segmentation, and even the problem of the *hierarchy of contours,* due to nested contours, was dealt with [3]. An interesting unifying approach to the described techniques is in [4], where a class of constrained clustering algorithms for boundary extraction (as a generalization of known algorithms) is introduced. However, those algorithms generally seem to be suffering from an intrinsic high computational complexity, and from an effect of smoothing of contours which can be undesired. In this paper we describe a technique of segmentation based on a specific type of active contour, that can reveal both closed and open contours very accurately, and with a good rejection to noise. The algorithm used, which in a preliminary phase was presented in ICIP '96 [5] and in [6] is intrinsically fast and easily parallelizable, and is susceptible of integration oriented to the introduction of specific properties in the resulting contour, such as smoothing, shape factors, etc. Finally, it does not only reveal the outer contours of the objects, but also the inner ones, due for example to the presence of nested objects.

2. The Amoeba Edge Tracer

Amoeba [5] is essentially an active contour, whose behavior is based on the principle of modeling the chain as a sequence of points, each having its own capacity of movement, the same way as a living tissue consists of a set of independent cells, tightly connected at the same time. Each of the points, that we will call *MOVels* (*MOVing ELements*), reproduce, move and die following a very simple set of rules that are all based on strictly local information about the two MOVels adjacent to it (the antecedent MOVel in the chain, and the subsequent). Due to the locality of the information used, the algorithm is therefore highly parallelizable, and at the same time maintains continuity and coherence in the movement of the chain as a whole. The chain needs (as the only starting assumption) that all the objects that must be revealed are at its inside: it is therefore obvious to initialize it always as a frame surrounding the whole portion of interest of the image, so that all the objects of interest are originally internal to it. With a quick sequence of steps, the chain adapts its shape and dimensions until it follows exactly those of the objects in its inside. The chain can also reveal the presence of several objects, both at the same level and nested each other even at more levels. The characteristics and behavior of this active contour, better described in [6], closely recalls that of an amoeba, and this name was given to the whole algorithm. Amoeba operated on a binary image where the points whose value is different than zero are edges, and the points whose value is zero are points of background. As we have already said in the introduction, the most effective approaches tend to combine the phase of edge tracing with thresholding. Therefore, the subsequent step in the evolution of the algorithm has been quite *the integration with edge detection.*

3. Using MOVels to Segment Images

Although the technique of segmentation presented in this paper is suitable for any operator selected for the edge detection, the algorithm that we have selected as a reference is the one of Canny [7], *without the edge tracing section.* At the end of the non-maximum suppression phase, the output image brings information about the magnitude of the "gradient-of-gaussian" filtering results, for all the points survived to non-maximum suppression. On this image, a chain of MOVels, an *Amoeba*, is laid, so that it surrounds the whole portion of interest of the image. Therefore, MOVels start their life cycle, moving towards the barycentre of the Amoeba and making it contract progressively, until it models itself on the contours of the objects that it contains. While Amoeba contracts, the number of MOVels must decrease; the mechanism causing this is based on the *death* of the MOVels that, during their path, meet an adjacent MOVel (the antecedent or the subsequent in the chain). A MOVel that dies is simply deleted from the chain. Then its antecedent is linked to its subsequent, so that the continuity of the chain is not interrupted. At the moment of its creation, an Amoeba is a *dense* chain of MOVels. This means that the distance between two subsequent MOVels is never higher than 1; the algorithm maintains this property, which is therefore a property of the contours obtained in output. For keeping this continuity, in case two consecutive MOVels depart, a mechanism of interpolation adding new MOVels to the Amoeba, is activated. MOVels continue to move towards the barycentre of the Amoeba, until they die or a condition of *deactivation* occurs: in case of deactivation, the MOVel in question remains *alive* (i.e. it is still part of the chain), but can no longer move. Besides, this means that, if the MOVels adjacent to it are not deactivated, the mechanism of interpolation will necessarily have to work. The most important cause of deactivation for a MOVel is its *mutation.* MOVels move on a matrix whose values represent the intensity of the response of the image to the application of the "gradient-of-gaussian" filter, after the thinning made by the non-maximum suppression. We could simply select as contour points those associated with a value of the response to the gradient-of-gaussian filter higher than a certain threshold. But Canny has proved that the use of *two* thresholds (with a *hysteresis* [8] mechanism) can be a much more advantageous choice. That's why MOVels bring with themselves, in their *genetic set*, a pair of thresholds, one lower than the other and both positive. They are determined either statically (according to the overall characteristics of the image) or dynamically, through appropriate heuristics and according to some local information propagated along the chain through appropriate mechanisms of "diffusion". When a MOVel reaches a point of the image whose magnitude is higher than the first threshold (the lowest one), it checks if one of its two adjacent MOVels (the antecedent or the subsequent) has already gone under mutation: in this case, it switches to the state of *mutation.* This means that it has found an edge and will stop there. If none of the adjacent MOVels has changed to mutation, it remains in a "metastable" stage, in which it is subject to mutation in case one of its two adjacents changes to mutation before its next move. This implies that a MOVel in the metastable state, before its next move, will have to check the state of its adjacent again. If a MOVel finds a point whose magnitude is also higher than the second threshold, it will immediately change to the mutation state. The mutation of a MOVel immediately causes a process of *propagation* of the alteration itself: when a MOVel changes its state, its antecedent is tested for checking whether it is in a metastable state; if it is in this state, the MOVel changes, and then *its* antecedent is checked. This process of *back-propagation* ends as soon as a MOVel is found in a non-metastable state, or when the first point of the chain is reached. It is easy to prove that, if the metastable points always check the status of their adjacents before to move, also the *forward propagation* of the mutation is automatically granted. The mechanism that we have just seen works like the edge tracing of Canny, but with some differences:

- In the case of Canny's algorithm, some subsequent stages are needed for doing the segmentation (the contours only are identified, but then they are not classified and associated with objects), while each Amoeba (since it is a closed contour) identifies only one object of the image;

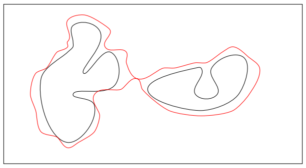


Figure 1: Splitting of Amoeba

- Since the MOVels of the same Amoeba are all linked each other, some appropriate mechanisms of diffusion can allow the propagation of thresholds determined heuristically according to local information, and progressively adapted and changed in progress. In the case of Canny (since there is no real segmentation) we cannot determine different thresholds for each object, but thresholds are global for the image;
- While the edge tracing of Canny links *all* the possible contour points, apart from the fact that they are part of closed contours or not, Amoeba can select the only closed contours, that for obvious reasons are the more significant for the segmentation. This factor is outlined in the test images reproduced in next paragraph.

Amoeba can reveal more than one object in the same image: a *splitting* mechanism starts its action as soon as a MOVel finds another MOVel of the same Amoeba, not adjacent to it. This case is shown in figure 1, where we can easily see how the described situation corresponds to the presence of several objects surrounded by the same Amoeba at the same time. The mechanism of *splitting* divides the Amoeba into two independent Amoebas, which are bordering at the point that has caused the process of division. Then the two Amoebas go on individually, and each of them surrounds an object. Since splitting can take place a virtually unlimited number of times, the algorithm can locate a very high number of objects in an image. This property makes Amoeba strong against salt and pepper noise. In fact, if a point with noise is mistaken for a contour point and this one is isolated, the Amoeba will completely surround it and, due to the effect of splitting, will create a new Amoeba. This will be so small (generally 3 points), that will be automatically deleted; the effect of the noise will therefore be cancelled. Amoeba can also reveal the "inner contours" of objects (nested contours). The mechanism adopted in this case is the *reproduction*. Once all the MOVels of an Amoeba have been changed and have therefore been deactivated, the Amoeba has completely revealed a contour. At this point, it *lays* a new Amoeba inside itself. This new Amoeba is active and ready to move; it starts a new life cycle in the area delimited by the "mother" Amoeba, revealing any "nested" contours in this area. This process can also take place for the "child" Amoeba, which could deposit another Amoeba inside it, and so on. If an Amoeba does not find any contour on which it could lean, it will completely contract, until it disappears. The output of this algorithm is a hierarchy of closed contours, each of them representing an object of the image, and nested contours corresponding to different layers of this hierarchy.

4. Tackling With Discontinuities

In real-world images, such as photographic images, the continuity of the contours of objects is only a theoretical hypothesis: bad conditions of brightness, obstructions due to partial superimposition, lack of contrast, excessive similarity among the characteristics of the surfaces belonging to different objects, shades, are only some of the factors that can make the contour of an object incomplete, discontinuous, and uneven. In case of small interruptions of the contour[1] the different techniques of edge linking behave in a different way:

- The algorithm proposed by Canny, in its original version, does not manage the event; it only interrupts the tracing. Some changed version use various heuristics for going over the found contours again, and doing a process of merging on them.
- The algorithms based on the fitting of parametric curves do not have this problem, since they are mainly based on preventive hypotheses about the shape of the object, and can therefore make assumptions and interpolations in the areas where the contour is not thick.
- The active contours with mechanisms of rigidity (such as snakes) manage the event, making the penetration through the hole "difficult" for the snake (from the energy point of view): the "effort" is as high as the interruption is narrow, and as the curvature of the contour of the hole is high. This way, the snake is *discouraged* from interrupting the contour for penetrating inside it through the hole. Yet, this technique has a drawback, since the rigidity that we have just described makes difficult (for the snake) to follow contours with abrupt changes of curvature, and to accurately recover contours characterized by specific texture, such as the one of a leaf [6], [4].

Amoeba has not been equipped with any mechanism of rigidity, although this was not too difficult. This choice is suggested by the need for not limit the flexibility of the algorithm and its adaptability to any type of contour. Conversely, if we want to use Amoeba as an effective technique of segmentation for real images, we need to equip it with an adequate mechanism for the treatment of the discontinuities in the contours.

The algorithm studied to detect and deal with holes and contour discontinuities is based (as well as the

---

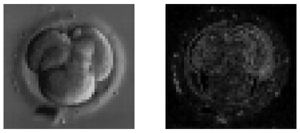
[1] The size of the contour interruptions is supposed to be of a few pixels; bigger interruptions need information *a priori* on the structure of the object, for dealing with them. They are therefore beyond this paper.

whole Amoeba) on look-up tables and on a minimum number of tests and accesses to memory, in order to make the system as fast as possible. A simple operation of test on indexes enables us to determine the direction of the movement of the MOVel. The algorithm searches (in appropriate positions in normal direction to the one of the movement) for the presence of points of high edgeness included between the two mentioned thresholds and, if it finds one, it activates the hysteresis mechanism, the same way it would do if the point were in the path of the MOVel. The radius within which the search is done can be selected among the values belonging to the set $\{1, 3, 5\}$. Of course, when this radius increases, the performances slightly decrease, but since the test process is not exhaustive but selective on some positions, this does not affect the performances of the system. The operation of the algorithm consists of the following steps[2]:

- Let us consider a square neighborhood in the image, whose sides are parallel to the axes of reference of the image; the centre is the MOVel and the side has the value selected as radius for the search;
- in case the motion is horizontal, we apply the test about the edgeness to the central column of the neighborhood and to the following one in the direction of the motion (that is, the column with a subsequent index, if the motion is rightwards, or with a previous index, if the motion is leftwards);
- In case the motion is vertical, the test is applied to the central row of the contour and to the following one in the direction of the motion (that is, the row of the subsequent index if the motion is downwards, or of the previous index if the motion is upwards);
- In case the motion is diagonal, the test is applied to the main diagonal and to the two following ones in the direction of the motion.
- The algorithm of hysteresis for the tracing is applied to all the points detected, characterized by edgeness included between the two thresholds.

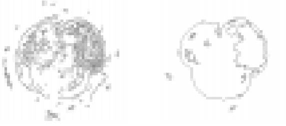
Several comparative tests have been made between the technique we used and the complete algorithm of Canny, in the implementation made available by J.R. Parker [8]. This implementation, in accordance with Canny's work, has no tools for the detection and the management of discontinuities. A summary of the tests performed is presented in next section.

5. Experimental Results



Original Image

Gradient Magnitude Image



Canny results Image

Amoeba results Image

Figure 2: The Embrio1 image.

In order to evaluate the characteristics of accuracy of the proposed algorithm of segmentation, we have used about 200 images, which have been downloaded from freely accessible sources in Internet[3]. The images reproduced below show the characteristics of Amoeba in comparison with the algorithm of Canny. The parameters used are absolutely the same for the two algorithms. Conversely (except for the image of the coffee maker) the mechanism for the detection and handling of the discontinuities of the borders of Amoeba has always been used, with a detection radius $\rho = 1$. No pretreatment has been done on images, except for the format conversion into 256 grey levels bitmaps. If we carefully observe Fig.2, we can notice how Amoeba has recognized the overall contour consisting of the three embryo-cells as a single object, unlike the algorithm of Canny. Besides, Amoeba has detected the presence of one of the inner areas, in the left region of the image. Conversely, Amoeba has not correctly revealed the presence of a circular contour outside the three cells, for which the algorithm of Canny has found two distinct portions (bottom on the left and on the right). In the case of image 3, we have deliberately switched off the function of discontinuity detecting. We can notice that Amoeba, due to its nature, has anyway detected the outer contour of the coffee maker as whole, therefore recognizing it as a single object. On the other hand, the fragmentation of the contours representing the inner lines (in particular, the vertical ones in the lower region of the coffee maker) has prevented there detection. We have had the same problem with the horizontal line representing the central junction between the two components of the coffee maker. The Figure 4 clearly shows the very good behavioral characteristics of Amoeba for the segmentation. The outer contour of the object in the figure (a puppet representing the famous cat Felix) has been correctly identified. Inside it we can notice (perfectly identified) the eyes, the mouth,

---

[2] Below we will assume that the image originates from the coordinates in the above edge on the left, and that the coordinates are always positive integers

[3] The authors wish to acknowledge and to thank the Administrators of the sites:

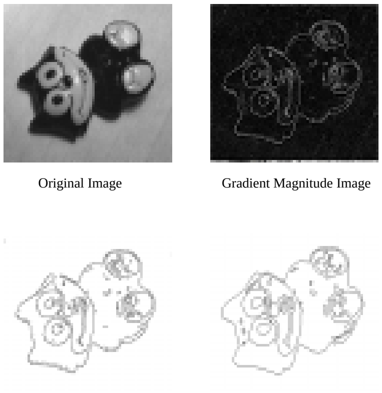
- http://www-syntim.inria.fr/syntim/analyse/images-eng.html
- ftp://vision.caltech.edu/pub/Images
- http://www.advancedfertility.com

for some of the images which have been used in this paper.

Original Image Gradient Magnitude Image

Canny results Image Amoeba results Image

Figure 3: The Cafet image.



Original Image Gradient Magnitude Image

Canny results Image Amoeba results Image

Figure 4: The Felix image.

the nose, and the legs. Each of these "inner" objects has been revealed as a separate unit, inside the main contour. Note that the algorithm of Canny has not worked in the case of the small discontinuity on the left side (top of the image) of the cat's face.

6. Conclusions

In this paper, a technique of segmentation for real 2-D images, based on a specific type of active contour, is presented. The main characteristic of the described active contour, which is called Amoeba, is its algorithmic (and not analytical) nature: the points that form Amoeba, which are called MOVels (Moving Elements), move and interact only according to few and simple laws, coded in the form of Look-Up tables. Amoeba can segment complex images and nested objects, and can manage nondensek and discontinuous contours. Besides, it can easily recover shapes that are characterized by very high curvature and are not smooth. Comparative tests have also been performed with well-known algorithms such as the one of Canny, with very good results: in the previous paragraph some significant examples have been presented. We are currently developing a mechanism for making the function more effective for the treatment of the discontinuities of contours in case of small and very close objects, such as the ones present in biological images (cells, biological tissues, etc). Besides, while optimizing the creation of the code, we are studying the best ways for its parallelization. Further developments concern the study of an effective approach to the introduction of structural and geometric properties in Amoeba, such as factors of shape, symmetries, etc. Such properties should be maintained during the execution of the algorithm, in order to introduce some "a priori" information in the process of segmentation.

References

[1] K. F. Lai and R. T. Chin, "Deformable contours: Modeling and extraction," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 1994.

[2] Myron Flickner, James Hafner, Eduardo J. Rodríguez, and Jorge L. C. Sanz, "Periodic quasi-orthogonal spline bases and applications to least-squares curve fitting in digital images," *IEEE Transactions on Image Processing*, vol. 5, no. 1, pp. 71–88, Jan 1996.

[3] R. Malladi, J. A. Sethian, and B.C. Vemuri, "Shape modelling with front propagation," *IEEE Trans. on PAMI*, vol. 17(2), Feb. 1995.

[4] R. Malladi and J. A. Sethian, "A unified approach to noise removal, image enhancement,and shape recovery," *IEEE Trans. on Image Processing*, vol. 5 (11), November 1996.

[5] G. Iannizzotto and L. Vita, "A fast, accurate method to segment and retrieve object contours in real images," in *International Conf. on Image Processing*, Lausanne, Switzerland, 1996.

[6] G. Iannizzotto and L. Vita, "Fast and accurate edge-based segmentation with no contour smoothing in 2-d real images," *to be published in IEEE Transactions on Image Processing.*

[7] J. F. Canny, "A computational approach to edge detection," *IEEE Trans. on PAMI*, vol. 8(6):679-698, June 1986.

[8] J. R. Parker, *Algorithms for Image Processing and Computer Vision*, Wiley Computer Publishing, 1997.