Edge Detection in Contaminated Images, Using Cluster Analysis

Héctor Allende[1] and Jorge Galbiati[2]

[1] Universidad Técnica Federico Santa María, Departamento de Informática, Casilla 110-V, Valparaíso
hallende@inf.utfsm.cl
[2] Pontificia Universidad Católica de Valparaíso, Instituto de Estadística, Casilla 4059, Valparaíso, Chile
jorge.galbiati@pucv.cl

Abstract. In this paper we present a method to detect edges in images. The method consists of using a 3x3 pixel mask to scan the image, moving it from left to right and from top to bottom, one pixel at a time. Each time it is placed on the image, an agglomerative hierarchical cluster analysis is applied to the eight outer pixels. When there is more than one cluster, it means that window is on an edge, and the central pixel is marked as an edge point. After scanning all the image, we obtain a new image showing the marked pixels around the existing edges of the image. Then a thinning algorithm is applied so that the edges are well defined. The method results to be particularly efficient when the image is contaminated. In those cases, a previous restoration method is applied.

1 Introduction

Edge detection is based on the assumption that discontinuities in the intensity of an image correspond to edges in the image, without disregarding the fact that often changes of intensity are not due only to edges, but can be produced by light effects, like shades or brightness, effects which demand additional treatment.

Among the operators used most frequently for edge detection are gradient operators and compass operators. The first one computes the gradient in two perpendicular directions, which are used to find the module and phase, and the second measures the gradient module in a set of different directions, selecting the one with largest value at each point. Unfortunately the derivative amplifies the noise, for that reason, filters must be used to smooth the images. When there are steep changes of intensity in the image, the gradient and compass operators work well, but don't do so when there are gradual changes in intensity. The Laplace operator is used in these cases, but it is more sensitive to noise so it requires a better smoothing of the image.

The amplification of the noise produced by most of the edge detectors usually result in reporting non existing edges. In this paper we introduce a detector based on cluster analysis for contaminated images, which also filters the image without altering it too much. Is is based on the cluster analysis filter proposed

M. Lazo and A. Sanfeliu (Eds.): CIARP 2005, LNCS 3773, pp. 945–953, 2005.

by [1], to which was added the ability to detect edges, using the same structure of grouping pixels into clusters. The results obtained by this edge detector are compared with known detectors to investigate its effectiveness.

2 Edge Detection Algorithm

The image is analyzed through sliding windows, which move along the image from left to right and from top to bottom, one pixel at a time. These windows consist of a 3x3 pixel square, numbered according to Figure 1. To analyze the central pixel in each window, a cluster analysis algorithm is applied to the eight surrounding pixels to detect groups with similar light intensity. Because of its simplicity, the best algorithm to use is the agglomerative hierarchical algorithm. In each iteration, the two nearest clusters are combined to form one, according to some distance measure previously determined.

The result is a nested or hierarchical series of groups of clusters formed with these eight pixels, starting with eight clusters with one pixel each, followed by seven, etc., ending with one single cluster containing the eight pixels. At each iteration, the distance at which the two closest clusters are grouped is recorded, forming an ascending sequence. A significantly big increase at iteration k + 1, as compared to some threshold value $T_{cluster}$, indicates that the optimum pattern of clusters is the one defined in the k-th step. A value of 25 for the threshold $T_{cluster}$ of in a scale of 1 to 255 has shown empirically to be a good choice [1]. The usual number of clusters is one, which corresponds to a smooth area.

Once the cluster pattern for the surrounding pixels is defined, the central pixel is examined and compared to the average intensity value of each cluster, with the purpose of determining if it belongs to one of them. If it differs too much from all the clusters, then it is considered a contaminated pixel (an outlier). To decide whether it belongs to some cluster, a second threshold value is introduced, T_{member}, to which the distances to the average of each group are compared. Empirical evidence shows that a suitable value for T_{member} is 36 [1].

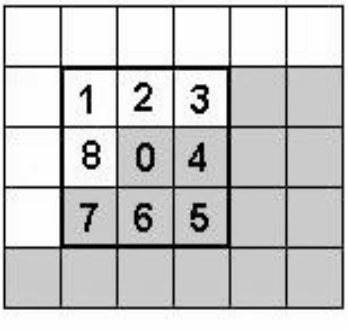

Fig. 1. 3 by 3 pixel window. The central pixel $Px(0)$ is being analyzed, based on the eight pixels $Px(1)$ to $Px(8)$.

If the central pixel is considered to be contaminated, then it is assigned to one of the surrounding clusters. This is done through a probabilistic procedure, favoring the clusters with highest number of pixels, and those that have greater adjacency with the central pixel and those that group more adjacent pixels together [1]. This way it manages to eliminate noise from the image without blurring it, as other well-known filter do. Once that the image has been treated to eliminate the existing noise, a second smoothing is carried out, using a median filter, to improve the definition of each cluster. Then the edge detection procedure is applied. It consists sliding 3x3 pixel windows, in a similar way as was described before. The same cluster analysis algorithm is applied, but this time it is used to keep record of the number of clusters present in each window. If there are more than one cluster in the window, then it means that it contains an edge, so the central pixel is marked as an edge point. Suppose that $Px(0)$ denotes the central pixel which is being analyzed, and $Px(y)$ one of its neighborhood pixels, y=1,2,..,8, numbered as in Figure 1. Let $C_{Px(y)}$ be the cluster containing $Px(y)$. Then $Px(0)$ is marked as a border pixel if in satisfies one of the following conditions:

$$C_{Px(2)} \neq C_{Px(0)}$$
$$C_{Px(4)} \neq C_{Px(0)}$$
$$C_{Px(6)} \neq C_{Px(0)}$$
$$C_{Px(8)} \neq C_{Px(0)}$$

When scanning the entire image using the previous method, the resulting edges are not very precise. In order to enhance the border lines, a thinning algorithm is applied. An appropriate algorithm is the one proposed by Nagendraprasad ,Wang and Gupta (1993) based on a previous one due to Wang and Zhang (1989) and improved by Carrasco and Forcecada [2], and it consists of the following:

Let $b(p)$ be the number of neighbors of $Px(0)$ that are marked as borders. We will call these pixels "black", while the ones that are not marked as borders will be referred to as "white". let $a(p)$ be the number of transitions from white to black, of the neighboring pixels, visited in the same order established in Figure 1. Let $c(p)$, $e(p)$ and $f(p)$ be functions defined in the following way:

$$c(p) = \begin{cases} 1, \text{ if } & Px(2) = Px(3) = Px(4) = Px(7) \quad \text{and} \quad Px(6) = Px(8) \\ 1, \text{ if } & Px(4) = Px(5) = Px(6) = Px(1) \quad \text{and} \quad Px(8) = Px(2) \\ 0, & \text{in other cases} \end{cases}$$

$$e(p) = (Px(4) + Px(6)) * Px(2) * Px(8)$$

$$f(p) = (Px(8) + Px(2)) * Px(6) * Px(4)$$

We proceed to scan the image iteratively. At each step, if $b(p)$ has a value between 1 and 7 and $a(p)$ or $(1-g) * c(p) + g(p) * d(p) = 1$, with $g = 0$ for odd iterations, $g = 1$ for even iterations. $d(p)$ is defined by

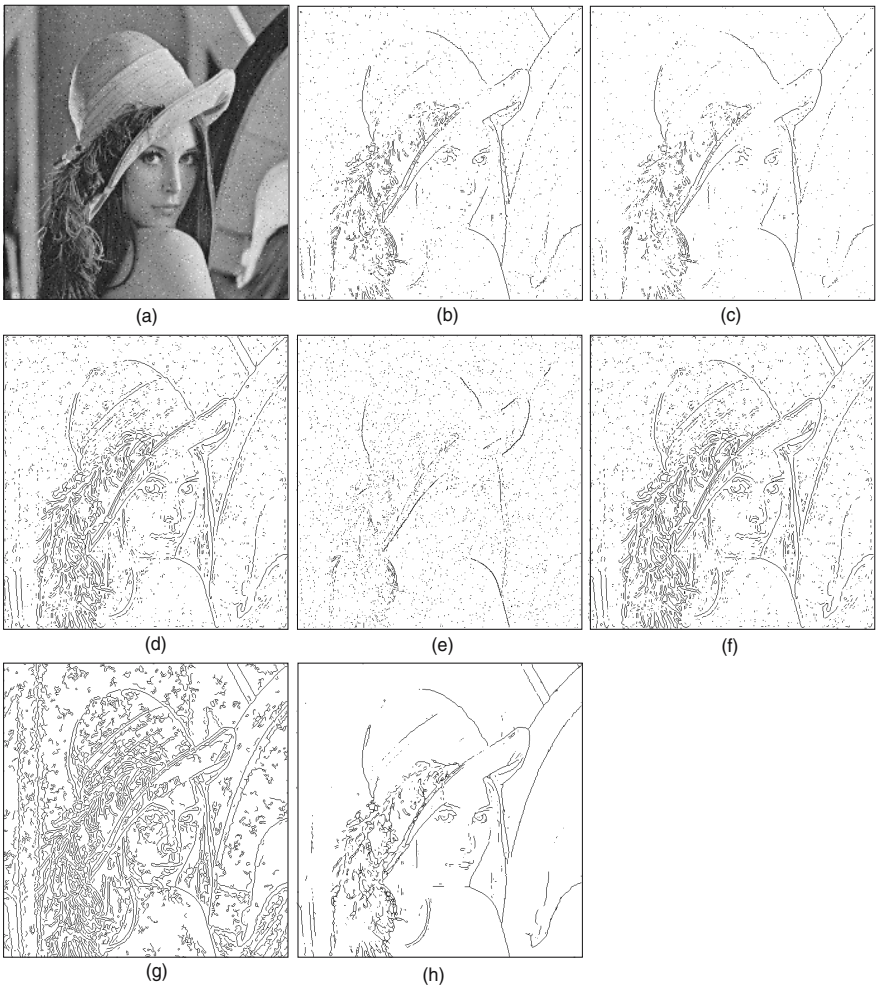

Fig. 2. (a) Original image 50 percent contaminated, standard deviation 20. Edge detection methods:(b) Prewitt (c) Sobel (d) Log (Laplacian of normal) (e) Roberts (f) Zero-cross (g) Canny (h) Proposed.

$$d(p) = \begin{cases} 1, \text{if} \quad Px(3) = Px(6) = Px(7) = Px(8) \quad \text{and} \quad Px(2) = Px(4) \\ 1, \text{if} \quad Px(2) = Px(5) = Px(8) = Px(1) \quad \text{and} \quad Px(4) = Px(6) \\ 0, \qquad\qquad\qquad\qquad\qquad\qquad \text{in other cases} \end{cases}$$

If we are in an even number iteration, then if $e(p) = 0$ the $p - th$ pixel is changed to white. If the iteration is odd-numbered, then if $f(p) = 0$, the p-th pixel is turned to white. In other cases, the p-th pixel is not changed. This process is carried out along the entire image. With this procedure we obtain the edges of the image, as connected lines, one pixel wide.

3 Experimental Results

We present several study cases with different levels of contamination, as a percentage P of contaminated pixels, and a standard deviation. A percentage P of pixels is randomly chosen and are contaminated in the following way: Let (i,j) be a chosen pixel and let x_{ij} be its light intensity. A random number Y is generated from a normal random variable with mean 0 and some fixed standard deviation. The intensity is then substituted by x_{ij} +Y, approximated to the nearest integer between 0 and 255. Tests are carried out using the threshold values mentioned

Fig. 3. (a) Original image 25 percent contaminated, standard deviation 40. Edge detection methods:(b) Prewitt (c) Sobel (d) Log (Laplacian of normal) (e) Roberts (f) Zero-cross (g) Canny (h) Proposed.

Fig. 4. (a) Original image 10 percent contaminated, standard deviation 80. Edge detection methods:(b) Prewitt (c) Sobel (d) Log (Laplacian of normal) (e) Roberts (f) Zero-cross (g) Canny (h) Proposed.

earlier for $T_{cluster}$ and T_{member}. The percentage of contamination and the standard deviation were given the values (50,20), (25, 40) and (10, 80). To observe the quality of the resulting edges they were compared to other commonly used edge detectors, like Prewitt [3], Sobel [4], LOG (Laplacian of Gaussian), Roberts, Zero-Cross and Canny.

The Roberts, Sobel and Prewitt edge detectors are based on the gradient of the image, formed by a vector field associated to each pixel. The vector's module is associated to the light intensity, and the direction of the vector to the direction of the major change in intensity. The Zero-Cross, Canny and LOG are based on the Laplacian, which is associated to the second derivative of the light

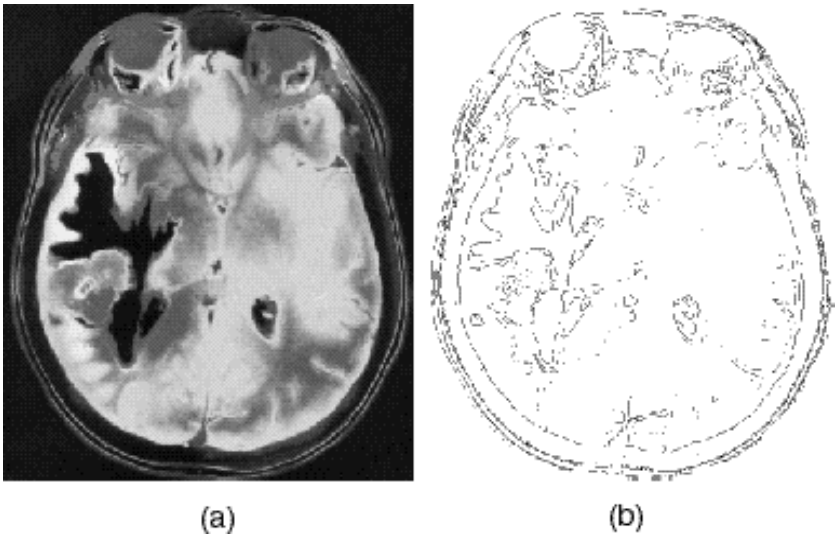

Fig. 5. (a) Original contaminated image. (b) Edge detection using proposed method.

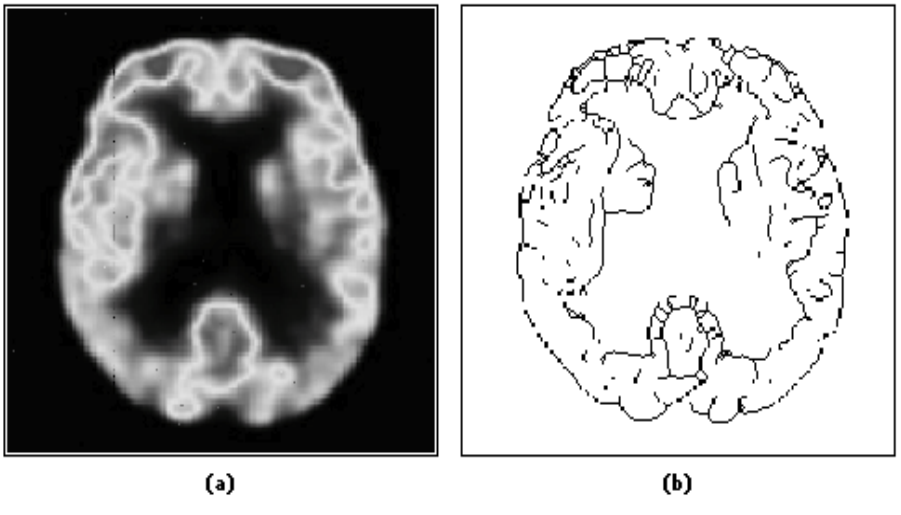

Fig. 6. (a) Original contaminated image. (b) Edge detection using proposed method.

intensity of the image, with which the zero crossings are detected, determining thus, the location of edges. Figures 2 to 4 show the results obtained for each of these detectors and for the one introduced in this article.

The times taken to complete the edge detection process, including smoothing and line enhancing, ranged between 515.5 and 523.4 seconds. For uncontaminated images, times for edge detection and line enhancement ranged between 255.8 and 258.4 seconds.

Fig. 7. (a) Original contaminated image. (b) Edge detection using proposed method.

Also, tests were carried out with other type of images, which by their particular way of obtaining the image, are naturally contaminated with noise, like satellite images and medical images. Figures 5 to 7 show the results.

4 Conclusions

The method introduced in this article approaches the problem of detecting edges in contaminated images. As it can be seen from the experimental results shown here, most of the edge detectors behave relatively well when there is a low level of contamination or when the standard deviation of the contamination is small (figure 2), due to the fact that the contaminated pixels are easy to smooth. But when there is a high contamination standard deviation is large (figures 3 and 4) then nonexisting edges appear, because most of the contaminated pixels cannot be smoothed out. In both cases, the proposed edge detector is able to find the proper borders, avoiding to point out contaminated pixels as edges. We can see with the results obtained in figures 5, 6 and 7, the power of the proposed detector to find borders in contaminated images, therefore it is a good alternative for processing medical images and satellite images. Observing the figures we can notice that for different contamination levels, we get similar results, obtaining the proper borders of the image, and not of the contamination.

References

1. H. Allende, J. Galbiati : A non-parametric to filter for digital image restoration, using cluster analysis. Pattern Recognition Letters, Vol 25, (June 2004), pp. 841-847.
2. Rafael C. Carrasco and Mikel L. Forcada : A note on the Nagendraprasad-Wang-Gupta thinning algorithm. Pattern Recognition Letters 16(5), (1995) pp. 539-541.
3. J. M. S. Prewitt: Object enhancement and extraction. In A. Rosenfeld and B. S. Lipkin, editors, Picture Processing and Psychophysics, pp. 75-149. Academic Press, New York (1970).
4. K. K. Pingle: Line of vision perception by to computer. In A. Grasselli, publisher, Automatic Interpretation and Classification of Images, pp. 277-284. Academic Press, New York,(1969).

5. H. Allende, J. Galbiati, R. Vallejos : Digital Image Restoration Using Autoregressive Time Series Models. Proceedings of the Second Latino-American Seminar on Radar Remote Sensing, ESA-SP-434,(1998) pp. 53-59.
6. H. Allende, J. Galbiati, R. Vallejos: Robust Image Modeling on Image Processing. Pattern Recognition Letters, Vol. 22, No. 11,(2001) pp. 1219-1231.
7. H.O. Bustos: Robust statistics in SAR image processing. ESA-SP No. 407, (February 1997), pp. 81-89.
8. C. da Costa Freitas, A. C. Frery, A. H. Correia: Generalized Distributions for Multilook Polarimetric SAR Data under the Multiplicative Model. Technical Report, (June 2002).
9. R.L. Kashyap, K.B. Eom: Robust Image Techniques with an Image Restoration Application. IEEE Transactions on Acoustics, Speech and Signal Processing, Vol. 36, No. 8, pp. 1313-1325 (1988).
10. L. Kaufman, P.J. Rousseeuw : Finding Groups in Data: An Introduction to Cluster Analysis. J. Wiley, N. York (1990).