Soft Computing manuscript No.
(will be inserted by the editor)

A new edge-detection method based on global evaluation using fuzzy clustering

Pablo A. Flores-Vidal · Pablo Olaso · Daniel Gómez · Carely Guada

Received: date / Accepted: date

Abstract Traditionally, the edge detection process requires a final step -known as scaling- for deciding pixel by pixel if they are selected as final edge or not. This can be considered as a local evaluation method that presents practical problems, since the edge candidate pixels should not be considered as independent. We propose a strategy to solve these problems through connecting pixels that form arcs -that we have called *segments*-. To accomplish this, our edge detection algorithm is based on a more global evaluation inspired by the human vision. Our paper further develops ideas first proposed in 1995 by Venkatesh-Rosin [1]. These *segments* contain visual features similar to those used by humans, which lead to better comparative results against humans. In order to select the relevant segments to be retained we use fuzzy clustering techniques. Finally, this paper shows that this fuzzy clustering of segments presents a higher performance compared to other standard edge detection algorithms.

Pablo A. Flores Vidal
Department of Statistics an Operation Research, Faculty of Mathematics of Complutense University, Madrid, Spain
Tel.: +34-913944535
E-mail: pflores@ucm.es

Pablo Olaso
Department of Quantitative Methods, University College of Financial Studies, Madrid, Spain

Daniel G´omez
Department of Statistics an Operation Research II, Faculty of Mathematics of Complutense University, Madrid, Spain

Carely Guada
Department of Statistics an Operation Research, Faculty of Mathematics of Complutense University, Madrid, Spain

Keywords edge detection; Global Evaluation; non-supervised classification; Fuzzy Clustering; edge segments

1 Introduction

In the last decades, edge detection [2] has been considered one of the main techniques for image processing. This technique aims to localize significant differences in intensity values of the image. This occurs in line with the way human vision works, being an important part of a visual process that has been named *primal sketch* [3].

Edge detection is quite useful in many fields. For instance, the recognition of different pathologies for medical diagnosis [4], a field that has grown in recent years. As well it is being used in images taken by satellites or drones -*remote sensing* - in agriculture. Other relevant application fields are the military industry, law enforcement, among others [5–11].

Due to the rapid development of computers, computer vision, which is the computational approach of human vision, emerged as a new possibility of understanding and explaining how human vision works. Computer vision is based on the underlying principle that processes involved in human vision work like a computer, or at least that computers can imitate the way human vision works [12]. The theory of edge detection was proposed in an article by D. Marr, and E. Hildreth [2]. This new computational technique allowed that different algorithms were developed. Example of this is Canny's [13] and similar others. These algorithms were based in different operators -*functions*- that worked over the picture elements -*pixels*- of an image.

Edge detectors are image processing algorithms which analyze the spectral information of an image, which means commonly analyze each pixel's brightness intensity (see [4]). When a variation between two neighboring pixels in the image is located, an edge of the region (or boundary) containing one of these pixels is detected, [14]. These boundaries can be depicted on the digital image drawing a white line onto these selected pixels and setting the other ones as the black background.

Many edge detectors have been developed in the literature. Some of them tried to detect if a certain pixel could be an edge using only information provided by adjacent pixels -*neighbors*-, others used a different strategy. The decision of what should become a definitive edge depends on the strength of luminosity gradient of each *edge candidate pixel* (see 2.1). This decision is taken in the second step of the *scaling* known as *thresholding process*, and it is traditionally made pixel by pixel. Then, this strategy of decision could be considered as a *local edge evaluation* [15]. The approach presented in [13] is situated between local and global evaluation. Canny's approach works with two thresholds, one for the lower bound of intensity and another for the upper bound -this process is known as *Hysteresis*-. This technique can be viewed as going one step beyond the standard of its time as that edge detection process traditionally consisted of a simple evaluation pixel by pixel. Nevertheless, it seems that Canny's technique could be improved ([1], pg.149). For example, the behavior inside the noisy areas of the image -mostly negligible for humans- may change the value of the thresholds applied, and then affect to the selection of definitive edges.

Due to the limitations of Local Edge Evaluation, the Global Evaluation approach emerged as a more natural strategy. In [1], the idea of *edge segment* is developed (see Subsection 2.1). In [13] and [15] it is employed a thresholding process based on more global criteria. Canny's method tended to connect adjacent edges as far as they were over a lower bound. This continuity allowed the edges to be configured in a more organic way. In practice, this usually leads to a successful discrimination of fragments of contours that belong to the objects of the image. Meanwhile, in [16] it was set out a different strategy which consisted in using the mean intensities of the pixels that made up the *edge list*. In [1] the concept of *edge list* was expanded with the use of another key feature of it: the length.

This work is based on the use of *edge segments* (see Subsection 2.1), and the methodology for creating them was introduced in Flores-Vidal et al. [17] although in this paper it is expanded. From these edge segments, different features were extracted (see Subsection 3.1) that proved to be useful to discriminate the good edges from the bad ones. The set of *good segments* is expected to be similar to the set of those detected by the human in the sketches of the ground-truth images [30, 18]. Thus, the quality of the comparatives between our algorithm's output and the ground-truth's is expected to be good.

The remaining of this paper is organized as follows: The next section is dedicated to the preliminaries, which consists in two subsections, a basic introduction to edge detection methodology and, more specifically for our proposal, the Venkatesh-Rosin strategy based on segments ([1]). Section 3 focuses on our proposal, an algorithm based on two main steps, the first one about building edge segments with different features, and the second one based on selecting the good segments through fuzzy clustering techniques. The last two sections are dedicated to the comparatives and results and final comments respectively.

2 Preliminaries

In this section are introduced some classical concepts of image processing and edge extraction problem. Let us denote by I a digital image, and by (i,j) the pixel coordinates of the *spatial domain*. For notational simplification the coordinates are integers, where each point (i,j) represents a pixel with $i = 0,\dots,n$ and $j = 0,\dots,m$. Therefore, the size of an image, $n \times m$, is the number of its horizontal pixels multiplied by its number of verticals. Let us denote by $I_{i,j}$ the spectral information associated with each pixel (i,j) (see [19]). The values range of this information depends on the type of image considered as we see in Figure 1.

- *Binary map*: $I_{i,j} \in \{0, 255\}$.
- *Grayscale*: $I_{i,j} \in \{0, 1, \dots, 255\}$.
- *RGB*: $I_{i,j} \in \{0, 1, \dots, 255\}^3$. (R=*Red*; G=*Green* and B=*Blue*).

Most edge detection algorithms are built as a combination of four sequential sub-tasks [20]:

1. *Conditioning*: During this step, the image is well prepared for the next phases of edge detection. Traditionally it consists in smoothing, de-noising or some other similar procedures ([21,22]). In practice, this phase basically helps making the edges easier to detect. After the conditioning phase our resulting image is a grayscale image that we will denote as I^s.
2. *Feature extraction*: Once the image is well prepared, in this step are obtained the spectral differences between adjacent pixels (see for example [23–25]). Then, the output of these differences is computed

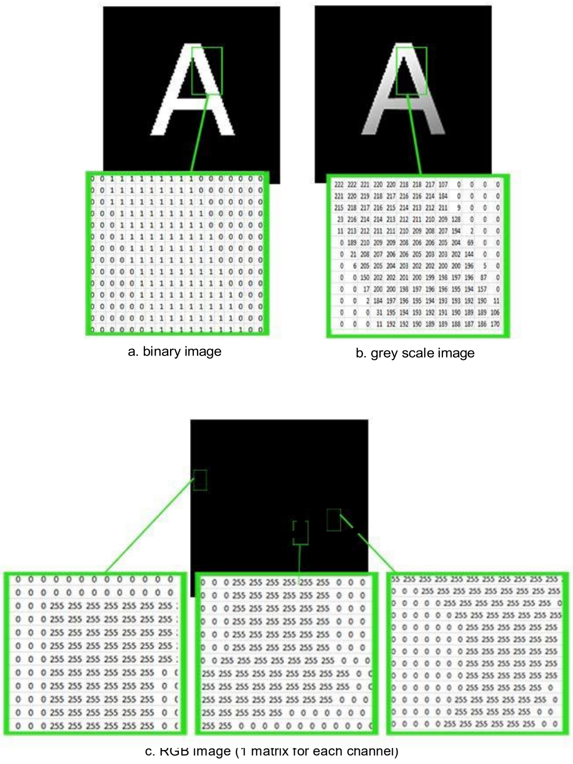


Fig. 1 Three types of images.

for each pixel (i,j) based on an operator function. For instance, if the operator is the one proposed by Sobel, for each pixel are obtained two values. Each one of these values represents the spectral variation (luminosity variation in grayscale images) in vertical and horizontal directions as shown below: Given a pixel with coordinates (i,j) and $i \geq 2, j \geq 2$, let

$$S_x(I^s_{ij}) = \sum_{a=i-1}^{i+1}\sum_{b=j-1}^{j+1} S^x I^s_{a,b}$$

be the horizontal variation, and let

$$S_y(I^s_{ij}) = \sum_{a=i-1}^{i+1}\sum_{s=j-1}^{j+1} S^y I^s_{a,b}$$

be the vertical variation, where

$$S^x = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$$

$$S^y = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$$

Taking into account the previous consideration, for a given pixel (i,j) we will denote by $X^1_{ij}, \ldots\ldots, X^k_{ij}$ the k extracted characteristics in this step.

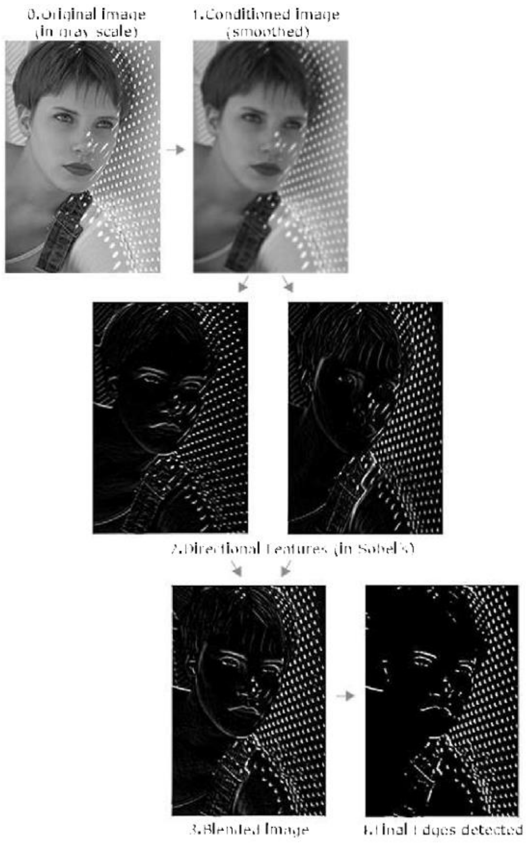

0.Original image (in gray scale)
1.Conditioned image (smoothed)
2.Directional features (in Sobel's)
3.Blended image
4.Final Edges detected

Fig. 2 The four sequential phases of edge detection using Sobel operator.

3. *Blending-aggregation*: During this phase, it is usually aggregated the information of the different features extracted into a single value denoted as *edginess*. From now on, let us denote by

$$I^{bf} = \varphi(X^1, \ldots, X^k)$$

the aggregated image resulting from this step and where φ denotes an aggregation function. For a given pixel (i,j), the value $I^{bf}_{i,j}$ represents the total variation of this pixel. It is common to represent this matrix as a grayscale image, where for each pixel we have a degree of *edginess* (see Figure 2.3). After this phase our resulting image is I^{bf}.
4. *Scaling*: In this last step, it is necessary to create the final output with the definitive edges (see Figure 2.4). Traditionally, after the constraints of Canny [13], there are only two possibilities, every pixel has to be declared as an edge or as a non-edge pixel. This decision is usually made by means of a thresholding process. As a result of this, the final output consists in a binary image. All the edges have to be as thin as possible as we see in Figure 4d. See Figure 2 for the whole sequence of edge detection.

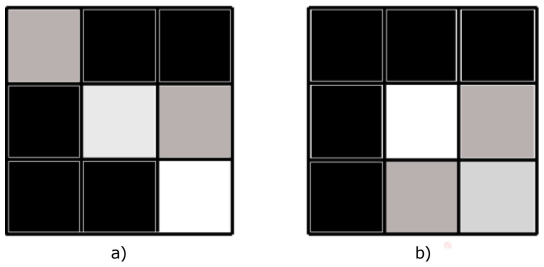


Fig. 3 Two kinds of 3x3 pixel windows from an image after going through the process of thinning: a) 4 candidates to edge pixels; b) 4 non-candidates to edge pixels as square structures of 2x2 non-zero intensity pixels are not allowed (they are not thinned).

2.1 Edge detection based on Segments

It is important to define with what kind of pixels our algorithm works. Let c be a candidate to edge pixel. Then c has to meet two conditions:

1. If (i_c,j_c) is the position of c in the spatial domain then $I^{bf}_{i_c,j_c} > 0$. In other words, it has to be a non-zero intensity pixel.
2. If there are three adjacent pixels to c that meet (1), then it is not possible that they set up a square shape. Therefore this is called a *thinned image*, which idea is shown in Figure 3 a).

From previous definitions we are able to define the set that contains these pixels. Let $C = \{c_1, ..., c_m\}$ be the set of all the *edge candidate pixels* in an image. This idea of connected pixels is strongly related to the concept that we explain below and that is a key point of our proposal.

In order to explain what an edge segment is and to show its importance, let us introduce this concept with an example. Let us suppose that we want to determine the final edges of Figure 4a. After three steps (conditioning, feature extraction and blending) and the thinning process, we have to decide over the *edge candidate pixels* in order to create the final output. In this last step, we can appreciate different gradient intensities -level of grays- of the pixels. The color differences -white and black- mean that these pixels are just candidates to become an edge, they are not yet definitive edges. We have defined them as c_i. In order to obtain the final solution we have to evaluate each *edge candidate pixel* to decide if it has to be declared as an edge pixel in the final output or not. Commonly, it is used a threshold value of the luminosity gradient of the pixel, $I^{bf}_{i,j}$, in order to make this decision. The more luminosity the gradient has -the whiter it is-, the more likely that it will be declared as an edge pixel. If we perform this

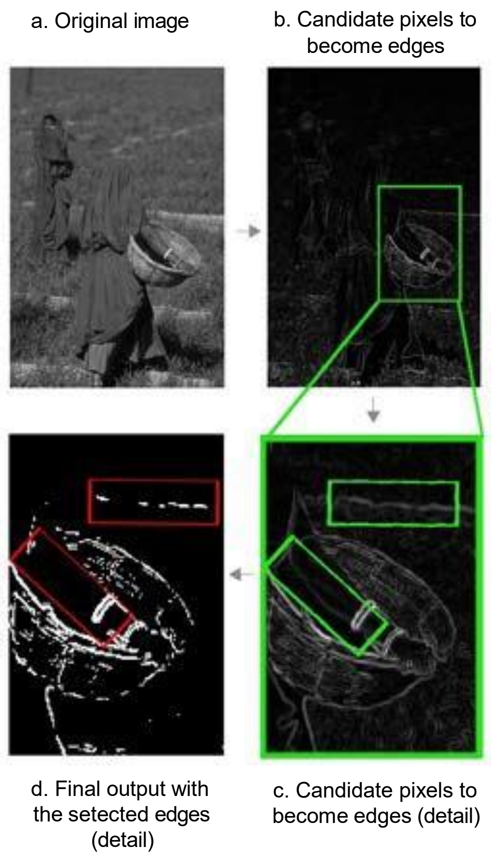

a. Original image
b. Candidate pixels to become edges
d. Final output with the setected edges (detail)
c. Candidate pixels to become edges (detail)

Fig. 4 Some limitations in local evaluated edges.

evaluation pixel by pixel, which is the traditional way, we can consider it as a *local evaluation process* as it is argued in [1]. Following this local evaluation approach, we could easily end up, for instance, in a situation like the one shown in Figure 4.

In Figure 4d, we see that some contours are extracted in a too fragmented way, loosing continuity. Instead, part of the contours of the objects have been extracted, while some other have not. We can easily appreciate this thanks to the details placed inside the red rectangles. If we go back one step behind in the process (Figure 4c) we will agree that the contours of these background lines should be continuous lines. It seems that something went wrong at the decision of those pixels being declared as *non-edge*. Furthermore, this mistake is not an exception as we can easily find some other similar discontinuous contours. Moreover, this seems to not just happen in this image, as López-Molina *et. al* pointed out [20]: *One of the most common errors committed by edge detection methods is not being able to complete the silhouette of an object*. In order to avoid this kind of mistakes when performing edge detection, we propose the use of a Global Evaluation method over the pixels. More precisely, this will be possible thanks to the evaluation over a list of connected pixels -linked edges- that will be referred below as *edge segments*. For instance, in Figure 4, some lines ended up being discontinuous because the decision did not take into account that the pixels belonged to a bigger common structure. This idea of connection between *edge candidate pixels* in a common structure lead us towards a fuller definition of an important concept that will be defined in the next paragraph.

Let $S = (c_1, \dots, c_n) \subset C$ be a subset of *edge candidate pixels set*, then we will call it an *edge segment* if and only if:

1. S is connected, i.e., $\forall c_a, c_b \in S$ there is a path through adjacent pixels $(c_i)_{i\in\{1,\dots,n\}} \subset S$ from c_a to c_b.
2. S is maximal, i.e., if $S^1 \subset C$ is another connected set of *edge candidate pixels*, then $S \subset S^1 \Rightarrow S = S^1$.

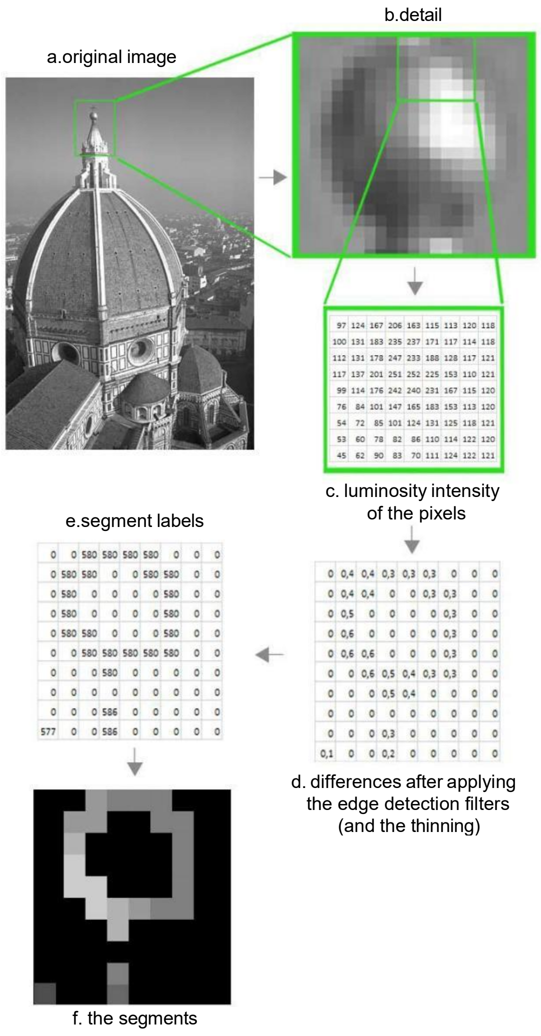
Notice that, given this definition, every edge candidate pixel belongs to one and only one edge segment since it is easy to see that the set of such defined edge segments, $\mathbf{S} = \{S_l : l = 1, \dots, s\}$, establishes a partition of C, i.e., $\cup_{l=1,\dots,s} S_l = C$ and $\cap_{l=1,\dots,s} S_l = \emptyset$.

Another important consideration about the edge segments is that any candidate to become a final edge will not be just a single pixel, but the whole segment containing that pixel. In the next section we will see how this way of linking pixels will affect to the binarization process. We can appreciate in Figure 5 the whole process of building an edge segment.

Setting a threshold for the gradient luminosity in order to decide which edges to retain can be considered the traditional method. Instead, the edge segment allows the use of features to make this decision. Following this approach, the thresholding value would change depending of the features values of the segments, and all this happen for each image. Therefore, this strategy for applying the thresholding process was called by [1] *Dynamic threshold determination*. However, we consider better not to think anymore about the classic idea of thresholding. We would rather address it in a more statistical way, as it is done in [17], as a problem of clustering between two possible groups: the "true" edge segments against the "false" edge segments.

2.2 The Venkatesh-Rosin algorithm

After explaining how to build the segments and before focusing on the rest of our proposal, it seems important to explain the idea behind the algorithm proposed by [1], in which is based ours. After using the concept of *edge list* -equivalent to the concept of edge segment that we have formalized in this section- these authors


b.detail
a.original image
c. luminosity intensity of the pixels
e.segment labels
d. differences after applying the edge detection filters (and the thinning)
f. the segments

97	124	167	206	163	115	113	120	118
100	131	183	235	237	171	117	114	118
112	131	178	247	233	188	128	117	121
117	137	201	251	252	225	153	110	121
99	114	176	242	240	231	167	115	120
76	84	101	147	165	183	153	113	120
54	72	85	101	124	131	125	118	121
53	60	78	82	86	110	114	122	120
45	62	90	83	70	111	124	122	121

0	0,4	0,4	0,3	0,3	0,3	0	0	0
0	0,4	0,4	0	0	0,3	0,3	0	0
0	0,5	0	0	0	0	0,3	0	0
0	0,6	0	0	0	0	0,3	0	0
0	0,6	0,6	0	0	0	0,3	0	0
0	0	0,6	0,5	0,4	0,3	0,3	0	0
0	0	0	0,5	0,4	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0,3	0	0	0	0	0
0,1	0	0	0,2	0	0	0	0	0

0	0	580	580	580	580	0	0	0
0	580	580	0	0	580	580	0	0
0	580	0	0	0	0	580	0	0
0	580	0	0	0	0	580	0	0
0	580	580	0	0	0	580	0	0
0	0	580	580	580	580	580	0	0
0	0	0	580	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	586	0	0	0	0	0
577	0	0	586	0	0	0	0	0

Fig. 5 From the original image to the segments.

settled up two features that defined the edge segment. The first one was the length of the edge segment (inspired by [16]) and the second one was the average intensity -edginess- of the pixels that compounded the edge segment.

Then, they built a two-dimensional feature space to represent the segment values in these two variables. In Figure 6 (taken from [1]) we see the scheme for generating the decision over the edge segments. The logic behind this geometric approach was to create a curve that separates the diagram in two parts, one containing the noise and the other one the true edge segments. Therefore, on the one hand the *true* segments are the most easily distinguishable by the human, as they are the longest, whether they have medium high or low intensity, and even short ones with medium or high intensity were distinguishable too. On the other hand, there are the disposable segments -the *false* segments- that will

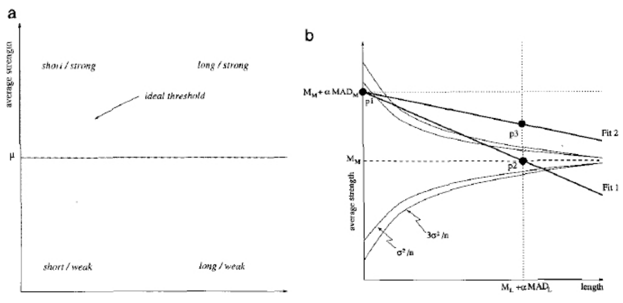


Fig. 6 The geometric approach of Venkatesh-Rosin ([1]): The feature space of the edges (a) and the scheme for deciding over the segments (b).

be those not so easily perceivable, that is, those that in addition to shortness present little average intensity.

We found slightly questionable -or at least that it could be improved- the heuristic method proposed by [1] to make this decision. From this point is where we will propose on of the main differences compared with the work of these authors. In the next section we will explain how the fuzzy clustering techniques suit perfectly for choosing to what cluster each edge segment belongs.

In the next section, we will see the difference between the Venkatesh-Rosin way of making the decision over the segments compared to the one that we propose.

3 Our proposal: Fuzzy clustering based on edge segments

Our proposal can be expressed as an algorithm that has two different parts, each one of them having a few steps. The first part is made of two steps related to the segments (1-2), while the other is focused on the performing of a fuzzy clustering approach studied in [27], specially designed for this work.

1. Given an already blended and thinned grayscale image I^{bf}, we have to obtain the set C (see 2.1 and Figure 5) and the segments set $\mathbf{S} = \{S_l : l = 1, \ldots, s\}$ of the image I^{bf}.
2. For each segment S_l, we obtain the segment's features (see 3.1 for more detail). Such features can be normalized and thus be measured as values in $[0, 1]$. Let us denote by x_r^l the *r-th* associated characteristic of segment S_l, for $l = 1, ..., s$; $r = 1, ..., f$ where f is the number of features extracted for each segment.

   Thus the space of segment features can be defined as $F = [0, 1]^f$ and $\mathbf{x}^l$ the vector of characteristics of segment S_l.
3. On the space F we apply a fuzzy clustering algorithm over the segments set based on relevance, redundancy and covering concepts [27] (see Subsection 3.2 for more details) fixing the number of clusters to 2 (bad segments and good segments). From the defuzzification of this fuzzy clustering solution we obtain the classification between *bad* and *good* segments that will give the final solution.

In the following two subsections we explain in detail the main two steps of this algorithm: features segment extraction and the 2-fuzzy clustering process.

3.1 Segment features

In this subsection the segment features used in our proposal are presented. All such features are eventually measured by a value in the $[0, 1]$ interval -even length, which was normalized-, hence the notation previously presented, where the vector of characteristics for segment S_l can be regarded as a point in the space of segment features: $\mathbf{x}^l = (x_1^l, x_2^l, ..., x_f^l) \in F = [0, 1]^f$ with f the number of characteristics considered. In this work 8 characteristics ($f = 8$) are taken into account, namely:

- Length. For each segment S_l, $x_1^l = Length_l = |S_l|$. Therefore, it can be seen as the number of pixels in the segment.
- Intensity Mean. For each segment S_l,

$$x_2^l = IM_l = \frac{_{p \in S_l} I_p^{bf}}{x_1^l},$$

  where I_p^{bf} represents the intensity of pixel p, which was obtained as the intensity gradient between p and its adjacent.
- Maximum and Minimum edginess. For each segment S_l, we obtained $x_3^l = Max\{I_p^{bf} : p \in S_l\}$ and $x_4^l = Min\{I_p^{bf} : p \in S_l\}$.
- Standard deviation of the intensity. For each segment S_l,

$$x_5^l = \sigma_l = \frac{_{p \in S_l}(I_p^{bf} - x_2^l)^2}{x_1^l}.$$

- Median of the edginess. For each segment S_l,

$$x_6^l = Median\{I_p^{bf} : p \in S_l\}.$$

- Average position. For each segment S_l, we obtained the coordinates of the pixel that occupies the central position in the segment: $(x_7^l, x_8^l) = Central_l$; where x_7^l is the average vertical position and x_8 is the average horizontal position of the pixels in S_l, i.e.,

$$x_7^l = \frac{_{p=(p_1,p_2) \in S_l} p_2}{x_1^l} \text{ and } x_8^l = \frac{_{p=(p_1,p_2) \in S_l} p_1}{x_1^l}.$$

  Once the average position is computed we get its Euclidean distance to the intersection points following the rule of thirds, which is an standard

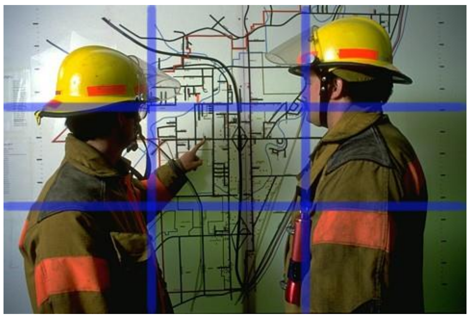


Fig. 7 The rule of thirds. In this example men faces tend to be close to the intersection points.

in photography composition (see [12]). This rule establishes that the most important objects in an image are usually placed close to the intersection of the lines that part the images in three equal parts. Following this principle, it seemed interesting to compute the minimum of its four distances, as there are four intersection points created by the four lines, as we can see in Figure 7.

Most of these features were created specifically for our proposal using principles that come from theory of human perception [12]. This means an important improvement with respect to our previous work [17], as we have employed four new features in this paper. The last feature, the average position, is specially important from a theoretical point of view, as the objects that a human recognizes tend to occupy certain positions in the image. That is the reason why the information related to the position of the segment S_l inside the image I^{bf} is relevant -this importance will be confirmed later in the experiment results- (see Section 5).

3.2 A Fuzzy Clustering

Now, we can use the characteristics that we have defined above to classify the segments in two sets: the *true* edges and the *false* edges. On the one hand we consider as the true ones those segments that a human eye can easily perceive. On the other hand we can consider the false segments as non-relevant noise. Being these *true* edges the segments perceived by the human, the comparatives that we will show later should lead to better results. This was, partly, the key issue that motivated our research: selecting those segments whose characteristics made them similar to what humans easily recognize in an image. Therefore, we seek to find two clusters of segments, one was expected to include the real edges and the other the false ones. Let us call them C_{true} and C_{false}. Both sets are a partition of the set of all segments: $C_{true} \cup C_{false} = \mathbf{S}$ and $C_{true} \cap C_{false} = \emptyset$. In [1], the authors used heuristic techniques to solve this problem: They established two regions separated by a curve obtained by a heuristic method (see Figure 6 and Figure 8). This method can be considered a *linear discriminant* type. Employing a thresholding value is common in edge detection literature, however our approach bases its decision on fuzzy clustering techniques as we will see below.

Algorithms like *Fuzzy C-means* or *K-means* could be used for this purpose, but they do not consider the nature of the data, and what is more important for the purposes of this work, they do not perform good enough when the clusters are unbalanced, as it is the case when the real edges are few when compared with the non-edge segments (specially when there is too much noise in the image). In such situation the mentioned algorithms would consider the real edges as outliers. Moreover, these clustering techniques try to optimize only one quality measure at a time.

We propose here an algorithm based on the approach presented in [17] and [26], which instead of just minimizing the sum of distances of the segments to their centroids as in *fuzzy c-means* algorithm, it is based on a multi-criteria problem that focuses in identifying the cluster centroids by taking into account three quality measures (see [27]):

- *Covering* : Rate of elements which are covered in a certain degree by any cluster.
- *Relevance*: A cluster will be relevant if it offers much information, in other words, if it has many elements with a certain degree of membership (higher than a given minimum).
- *Redundancy*: Represents the overlap degree between the clusters.

Hence, these three quality measures represent three different criteria to optimize. As in any multi-criteria problem, many approaches can be followed to solve this clustering problem, our proposal is as follows:

Let us consider the following parameters: $m \in [0, 1]$ as the minimum degree of membership to calculate *relevance*, *re* as the minimum degree of membership to calculate *redundancy*, and *pr* as the percentage of *allowed redundancy*. Thus the *relevance* of a potential cluster can be calculated as the number of segments belonging to it with a membership degree of at least m, and two given clusters will be redundant (and hence incompatible) if a proportion greater than *pr* of the segments belong to both clusters with at least degree *re*. The steps of the algorithm are as follows:

- Define the set of potential clusters by building a grid on the space of segment features F, $K = \{\mathbf{y}^i : i = 1, ..., k\} \subset F$, then each of the vertices of such grid will be the centroid of a potential cluster.
- Calculate each centroid's relevance, r_i, as the number of elements that belong to the *i-th cluster* with a membership degree of at least m, for $i = 1, ..., k$.
- Select the most relevant cluster, ie:

$$i^I = \arg \max_{i=1,...,k} \{r_i\}$$

- Calculate the sets of common segments between clusters i^I and i, $\forall i \in \{1, ..., k\} \setminus \{i^I\}$:

$$D_i = \{l : \min\{\mu_{i^I}(l), \mu_i(l)\} \geq re\},$$

  where $\mu_i(l)$ represents the degree of membership of segment S_l to the *i-th* cluster, which is calculated as $\frac{1}{\|\mathbf{x}^l - \mathbf{y}^i\|}$ or, in other words, the inverse of the (Euclidean) distance between the vector of characteristics of segment S_l and the centroid of the *i-th* cluster. Then i^I and i are considered as redundant if $|D_i| \geq pr$. Now calculate the set of clusters which are not redundant with i^I, $D = \{l \in \{1, ..., k\} : |D_l| < pr\}$. Finally, select the most relevant cluster, i^{II}, among those in D, ie $i^{II} = \arg \max\{r_l : l = 1, ..., k\}$.
- If $\|y^{i^I}\| < \|y^{i^{II}}\|$ then interchange them, ie , since the edges should be those with greater norm in F.

We can see the whole approach in Figure 8.

Let's study the computational complexity of each step of the algorithm: The first step is defining the set of potential clusters, and its time and space requirements are in the order of $O(k)$ where k represents the initial number of potential clusters and is an input parameter. The second step is to calculate the relevance of each cluster, and its time and space requirements are in the order of $O(k \cdot s)$ and $O(k)$ respectively, where s is the number of segments. The third step can be performed parallel to the previous one, and in any case requires an $O(k)$ order time. The fourth step, which consists of calculating the sets of common segments between the i^1 cluster and the other ones, requires $O(k \cdot s)$ in time. In short, the complexity of the entire algorithm is in the order of $O(k \cdot s)$.

The fourth step, which consists of calculating the sets of common segments between the i^1 cluster and the others, requires $O(k{\cdot}s)$ in time. In short, the complexity

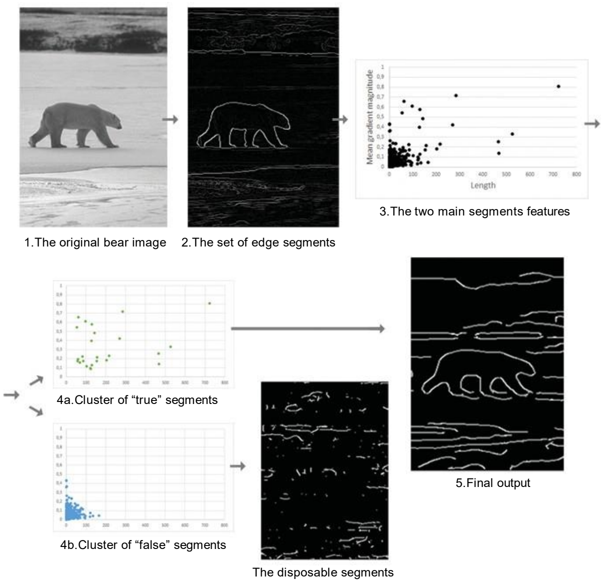

1.The original bear image
2.The set of edge segments
Mean gradient magnitude
Length
3.The two main segments features
4a.Cluster of "true" segments
4b.Cluster of "false" segments
The disposable segments
5.Final output

Fig. 8 An abbreviated scheme of our proposal.

of the entire algorithm is in the order $O(k \cdot s)$. Let's study what this means: k depends on the number of characteristics to be taken into account, and the size we assign to the grid, if for example 5 characteristics are being studied and for each of them a grid of 10 steps is considered, there will be a total of $k = 10^5$ potential clusters, in general if n is the number of steps and f the number of characteristics, it will be $k = n^f$. Finally, s can vary greatly, depending on both the image being studied and the algorithm previously selected to obtain the segments.

4 Comparison and results

For evaluating the performance of our *Fuzzy Cluster of Segments (FCS)* algorithm, we have used the image set provided by the Computer Vision and Pattern Recognition group of the University of South Florida (USF) that is presented in [29] (and can be downloaded from [30]). This set consists in 60 images between objects and aerial images, and it is been specially created for comparison in edge detection. Due to the nature of the USF dataset -having three different pixel categories- and in order to compute precision and recall measures, the "non-relevant" pixels were ignored in the matching process. Then it did not matter whether the edge detector detected an edge or not in a non-relevant area. Doing it this way, these non-relevant areas would not affect precision and recall measures. Then we compared our FCS algorithm with other five high standard edge detection algorithms by means of the matching technique proposed by [28]. This works by means of a circular window ξ that is centered in the pixel that is being compared. In this case, the parameter employed for circular distance was $\xi = 5$, following these authors advise. *Precision*, *Recall* and F-measure were employed by these authors to evaluate the quality of the comparatives. These measures have been commonly used for edge detection comparisons (see for example [10]). *Precision* measures the rate of edges selected by the algorithm that match with the edges in the human sketches belonging to the ground truth. *Recall* computes the rate of edges detected by the human -ground truth- that are detected as well by the algorithms output. Let C_{human} be the set of edges detected by the human, then:

$$Precision = \frac{Matched(C_{true}, C_{human})}{|C_{true}|}$$

$$Recall = \frac{Matched(C_{human}, C_{true})}{|C_{human}|}$$

$$F = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

We have employed the philosophy of *Benchmarking* [18] for the comparison over the six edge algorithms. Therefore, we made the comparatives by using a range of different parameter values for each algorithm. In Table 1 we can see the comparative results for the best fixed parameter values found. Therefore, the computational experiments were executed over the 60 images belonging to the USF dataset [30]. 35 images -from "131" to "cone" sortened by number and after alphabetically- were used as training set and the other 25 images for the test set -from "101" to "130" and from "egg" to "woods". For each edge detector were considered different parameters and procedures. All of them were applied in two different versions, with a Gaussian smooth filter (σ_s=1) and without it (σ_s=0):

1. *Canny algorithm:* It was applied the 'sigma of Canny' parameter (σ_{Canny}), which is the Gaussian filter that works in the convolution of Canny's and produces even smoother edges. In the case of our algoritm the higher this parameter is the less reduces amount of edge segments are selected. Different values were explored for this parameter 0.5, 1, 2 and 4. After that was applied the well-known non-maximum suppresion for the "thinning" process. For the scaling step, the double threshold called "Hysteresis" was applied [13].
2. *F^1-Transform algorithm*: This is the F^1-transform method used for preprocessing in Canny's [31, 10]. This algorithm is used for both, smoothing the image first and then doing the convolution. It requires the use of a h parameter for these two steps. The higher h is, the smoother it results the smoothed image I^s. Values $h = 3$ in the first step, and $h = 2$ were respectively applied. The next steps were the usual of Canny's (Non-maximum suppression and Hysteresis).
3. *Gravitation algorithm*: This is an edge detection algorithm based on the Law of Universal Gravity of Isaac Newton. This algorithm computes the gradient at each of the pixels using the gravitational approach based on a t-norm (that is why it is named GED-T). See details in [32]. This method does not perform any of the other processes of the image needed to obtain binary edges, e.g. smoothing, binarization etc. Unluckily, we could not use this algorithm to generate any edges when we applied previous smoothing, then it only worked in the non-smoothed version. Different triangular t-norms were employed following the approach of [32]. In our case we have used the Lukasiewicz and the Nilpotent minimum t-norms. As with the F-transform algorithm, the last steps are the usual of Canny's (Non-maximum suppression and Hysteresis).
4. *Sobel algorithm:* This is the classic algorithm that was proposed by Sobel in a talk [33]. A single threshold was applied with values ranging from 0.10 to 0.99.
5. *Venkatesh and Rosin algorithm*: This is an slightly improved version (the code can be found in [36]) of the proposal presented in [1] on which our work is inspired (see Subsection 2.2). Like with our algorithm, first steps were the same as Canny's (till the non-maximum suppresion). The alpha parameter is the only one specifically required ($\alpha = 6$).
6. *Fuzzy Cluster of Segments algorithm*: First steps were the same as Canny's (till the non-maximum suppresion). Then, at the scaling step FCS was implemented for three different quality measures that range from 0 to 1 (see 3.2 for further information). The first of the parameters related to FCS is $m \in [0, 1]$ which is the minimum degree of membership to calculate relevance. It has to be theoretically high enough. The best value reached was 0.90. *redundancy* or *re* has to be smaller than relevance, as it represents the maximum membership function value allowed for a certain segment in both clusters. In this case the best fixed value for *re* was 0.60. Finally, we configured a third theoretical parameter, *the percentage of allowed redundancy*, *pr* =0.15, since we noticed in previous experimental proofs that the output of the algorithm didn't change much when this parameter took values inside the percentage range that seemed to us reasonable for the redundancy (between 5% and 20%).

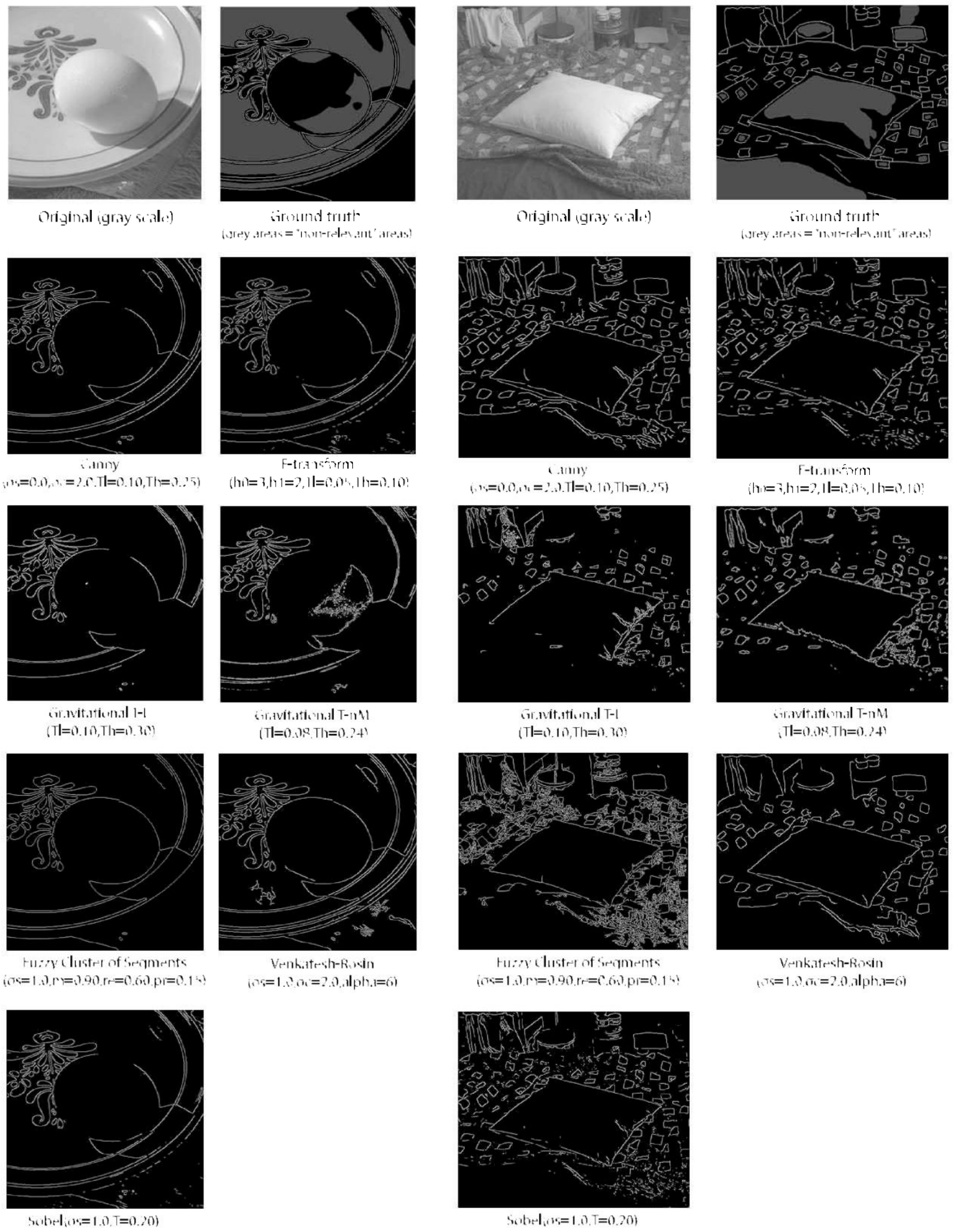


Fig. 9 Binarized outputs of USF dataset images for different algorithms with best fixed parameters. σ_s=Gaussian smoothness, Tl=lower threshold, Th=higher threshold, n=operator dimension, m=relevance, rev=redundancy, pr=redundancy % allowed (see Section 4 for more details)

Notice that our FCS algorithm had the best performance -even slightly better than Canny's- compared against the other five edge detection algorithms (see Table 1). The F-measure values correspond to the F-measure average results for the 25 images belonging to the test set (see second paragraph of this section for more details about the training and test sets) when comparing the algorithms output with the humans ground truth.

We can see in Figure 9 the output of all the edge detection algorithms employed. The visual comparative shows that the edges provided by FCS are cleaner -less noisy- than the rest of the algorithms, specially in the *egg* image. It can be appreciated as well an improvement in F-measure and Precision when comparing FCS with Venkatesh and Rosin's. The *pillow* image shows that FCS is capable as well to retain most of the relevant edges that were extracted by the ground truth -the human-.

5 Conclusions

The algorithm *FCS*, which we propose in this work performs significantly better than the other five algorithms on the USF image dataset. Only Canny's performance can be considered close to ours. Even if FCS performance seems good enough, we believe that there is enough room for improvement. One reason for supporting this idea is that for the construction of the *edge segments* it is possible to collect other characteristics specially designed to compute a certain visual task. For instance, building other features related to the shape of the segment or even its position could be useful for edge detection. Another interesting aspect for improving this research would be to contemplate more than two possible *clusters* to perform the *fuzzy clustering* what seems as a complex line that could point out to future research. Following this idea, the output of the comparatives would not be unique, allowing to establish diffuse hierarchies or partitions, similar to the ones that arise in [14], [34] and [35].

Finally, we would like to point out that building other comparatives more suitable for the edge segments would be a good recommendation for the future evolution of this research line about *edge segments*. However, in order to construct this new kind of comparative based on segments, it seems that it would be necessary to adapt the current human ground truth into a modified version of it. We believe that an interesting future research line about edge segments could follow this last idea, and maybe this could be done as well in a supervised approach.

It has been strongly helpful for the conducting of this research the code created by Kermit Research Unit [36].

Acknowledgements This research has been partially supported by the Government of Spain, grant TIN2015-66471-P.

6 Compliance with ethical standards

Conflict of Interest: Pablo Flores-Vidal declares that he has no conflict of interest, Daniel G ómez declares that he has no conflict of interest, Carely Guada declares that she has no conflict of interest and Pablo Olaso declares that he has no conflict of interest.

Ethical approval: This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Venkatesh S., Rosin, P. L.: Dynamic threshold determination by local and global edge evaluation Graphical Models and image processing. 57(2), 146–160 (1995)
2. Marr, D. Hildreth, E.: Theory of edge detection Proceedings of the Royal Society of London - Biological Sciences. 207(1167), 187–217 (1980)
3. Marr, D.: Vision: a Computational Investigation into the Human Representation and Processing of Visual Information The MIT Press, Cambridge, Massachusetts (1982)
4. Sonka, M. IEEE transactions on medical imaging statement of editorial policy. IEEE Transactions on Medical Imaging, 33(4), (2014)
5. Monga, O., Deriche, R., Malandain, G., Cocquerez, J. P.: Recursive filtering and edge tracking: Two primary tools for 3D edge detection. Image and Vision Computing, 9(4), 203–214 (1991)
6. Fathy, M., Siyal, M. Y.: An image detection technique based on morphological edge detection and background differencing for real-time traffic analysis. Pattern Recognition Letters, 16(12), 1321–1330 (1995)
7. Zielke, T., Brauckmann, M., Vonseelen, W.: Intensity and edge-based symmetry detection with an application to car-following. CVGIP: Image Understanding, 58(2), 177–190, (1993)
8. Pal, S., King, R.: On edge detection of X-Ray Images Using Fuzzy Sets, IEEE Transactions on Pattern. Analysis and Machine Intelligence, 5(1), 69-77, (1983)
9. Bustince, H., Barrenechea, E., Pagola, M. and Fern´andez, J.: Interval-valued fuzzy sets constructed from matrices: Application to edge detection, Fuzzy Sets and Systems 160(13), 1819–1840 (2009).
10. Perfilieva, I., Hod´akov´a P., and Hurt´ık, P.: Differentiation by the F-transform and application to edge detection, Fuzzy Sets and Systems 288, 96–114 (2016)
11. Daňková, M., Hod´akov´a P., Perfilieva, I. and Vajgl, M.: Edge detection using F-transform, 11th International Conference on Intelligent Systems Design and Applications, IEEE, 672–677 (2011)

	F	*F*	*Precision*	*Precision*	*Recall*	*Recall*
	$\sigma_s = 0$	$\sigma_s = 1$	$\sigma_s = 0$	$\sigma_s = 1$	$\sigma_s = 0$	$\sigma_s = 1$
Canny ($\sigma_{Canny} = 2$, T.H=0.26, T.L=0.10)	0.73	0.73	0.75	0.76	0.74	0.72
GED-T Lukasiewicz t-norm ($\sigma_{Canny} = 2$, T.H=0.29, T.L=0.10)	0.	-	0.	-	0.	-
GED-T Nilpotent t-norm ($\sigma_{Canny} = 2$, T.H=0.24, T.L=0.08)	0.	-	0.	-	0.	-
F^1 transform (h=3, T.H=0.18, T.L=0.06)	0.	-	**0.**	-	0.	-
Sobel (T=0.20)	0.	0.	0.	0.	0.	0.
Venkatesh and Rosin ($\sigma_{Canny} = 2$, $\alpha = 6$)	0.70	0.70	0.68	0.71	**0.74**	**0.70**
FCS ($\sigma_{Canny} = 2$, relev=0.95, overlap=0.60 and overlap (%)=15%)	**0.71**	**0.71**	**0.77**	**0.78**	0.70	0.68

Table 1 F, Precision and Recall averages for USF image test set (25 images). Best fixed parameters.

12. Goldstein, E.B.: Sensaci´on y percepci´on Sexta ed. Thomson Editores Spain (2009)
13. Canny, J.: A Computational Approach to edge detection IEEE Trans. on Pattern Analysis and Machine Intelligence, PAMI-8(6), 679–698 (1986)
14. Guada, C., G´omez, D., Rodr´ıguez, J.T., Y´aez, J., Montero, J.: Classifying images analysis techniques from their output International journal of Computational Intelligence Systems, 9(Supplement 1), 43–68 (2016)
15. Kitchen, L., Rosenfeld, A.: Edge Evaluation Using Local Edge Coherence IEEE Trans. on Systems, Man and Cybernetics. 11(9), 597–605 (1981)
16. Lowe, D. G.: Three-dimensional object recognition from single two-dimensional images Artificial Intelligence, 31(3), 355–395 (1987)
17. Flores-Vidal, P.A., G´omez, D., Olaso, P., Guada, C.: A New edge detection Approach Based on Fuzzy Segments Clustering, Advances in Fuzzy Logic and Technology 2017, 58–67, (2017)
18. Martin D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics Proceedings of the IEEE International Conference on Computer Vision, 2, 416–423 (2001)
19. Gonz´alez, R. C. Woods, R. E.: Digital image processing Third Edition Pearson Prentice Hall (2008)
20. L´opez-Molina, C., De Baets, B., Bustince, H. Quantitative error measures for edge detection Pattern Recognition. 46(4), 1125–1139 (2013)
21. Basu, M.: Gaussian-based edge-detection methods- A survey. IEEE Trans. on Systems, Man, and Cybernetics, Part C: Applications and Reviews 32(3), 252-260 (2002)
22. Morillas, S., Gregori, V., Hervas, A.: Fuzzy peer groups for reducing mixed Gaussianimpulse noise from color images. IEEE Trans. on image processing 18(7), 1452-1466 (2009)
23. Bezdek, J., Chandrasekhar, R., Attikouzel, Y.: A geometric approach to edge detection. IEEE Trans. on Fuzzy Systems 6(1), 52-75 (1998)
24. Bustince, H., Barrenechea, E., Pagola, M., Fernandez, J.: Interval-valued fuzzy sets constructed from matrices: Application to edge detection. Fuzzy Sets and Systems 160(13), 1819-1840 (2009)
25. Kim, D.S., Lee, W.H., Kweon, I.S.: Automatic edge detection using 3x3 ideal binary pixel patterns and fuzzy-based edge thresholding. Pattern Recognition Letters 25(1), 101-106 (2004)
26. Rojas, K., G´omez, D., Montero, J., Rodr´ıguez, J.T., Valdivia, A., Paiva, F.: Development of child's home environment indexes based on consistent families of aggregation operators with prioritized hierarchical information. Fuzzy sets and Systems, 241, 41–60, Elsevier (2014).
27. del Amo, A., G´omez, D., Montero, J., Biging, G.: Relevance and redundancy in Fuzzy Classification Systems Mathware and Soft Computing. 8, 203–216 (2001)
28. Estrada, F. J., Jepson, A. D.: Benchmarking Image Segmentation Algorithms International journal of computer vision. 85(2), 167–181 (2009)
29. Heath, M., Sarkar, S., Sanocki, T. and Bowyer, K.W.: A Robust Visual Method for Assessing the Relative Performance of Edge-Detection Algorithms. IEEE Transactions on Pattern Analysis and Machine Intelligence. 19(12), 1338–1359 (1997)
30. University of South Florida Dataset: ftp://figment.csee.usf.edu/pub/ROC/edge_comparison_dataset.tar.gz, 09-27-2017
31. Perfilieva, I., Hod´akov´a P., and Hurt´ık, P.: Differentiation by the F-transform and application to edge detection, IPMU 2012, CCIS 297, 230–239 (2012)
32. L´opez-Molina, C., Bustince, H., Fern´andez, J., Couto, P., De Baets, B.,: A gravitational approach to edge detection based on triangular norms Pattern Recognition. 43, 3730–3741 (2010)
33. Sobel, I., Feldman, G.: A 3x3 isotropic gradient operator for image processing, Presented at a talk at the Stanford Artificial Project, (1968).
34. G´omez, D., Zarrazola, E., Y´an˜ez, J., Montero J.: A Divide-and-Link Algorithm for Hierarchical Clustering in Networks Information Sciences. 316, 308–328 (2015)
35. G´omez, D., Y´an˜ez, J., Guada, C., Rodr´ıguez, J., Montero, J., Zarrazola, E. Fuzzy image segmentation based upon hierarchical clustering Knowledge-Based Systems. 87, 26–37, (2015).
36. The Kermit Image Toolkit (KITT), Ghent University, De Baets, B. and Lopez-Molina, C.,: Available on-line at www.kermitimagetoolkit.net, 09-27-2017