



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

Pablo A. Flores-Vidal¹ · Pablo Olaso² · Daniel Gómez³ · Carely Guada¹

Published online: 26 September 2018
© Springer-Verlag GmbH Germany, part of Springer Nature 2018

Abstract

Traditionally, the edge detection process requires one final step that is known as scaling. This is done to decide, pixel by pixel, if these will be 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. In this article, 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 actual human vision. Our paper further develops ideas first proposed in Venkatesh and Rosin (Graph Models Image Process 57(2):146–160, 1995). 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.

Keywords Edge detection · Global evaluation · Supervised classification · Fuzzy clustering · Edge segments

1 Introduction

In the last decades, edge detection (Marr and Hildreth 1980) has been considered one of the main techniques for image processing. This technique aims to identify significant differences in intensity values of an image. Human vision works similarly, and this important part of the visual process has been called *primal sketch* (Marr 1982).

Edge detection is quite useful in many fields. For instance, for the recognition of different pathologies in medical diagnoses (Bogunovic et al. 2014), a field has grown in recent years. It is also used in images taken by satellites or drones—*remote sensing*—for agricultural purposes. Some other relevant fields of application are the military industry, law enforcement, among others (Monga et al. 1991; Fathy and

Siyal 1995; Zielke et al. 1993; Pal and King 1983; Bustince et al. 2009; Perfilieva et al. 2016; Daňková et al. 2011).

Due to the rapid development of computers, computer vision, which is the computational approach to human vision, emerged as a new way 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 does, or that at least that computers can imitate the way that human vision works (Goldstein 2009). The theory of edge detection was proposed in an article by Marr and Hildreth (1980). This new computational technique allowed different algorithms to be developed. An example of this is Canny's (1986). These algorithms were based in different operators—*functions*—that worked over the picture elements—*pixels*—of an image.

Edge detectors are image processing algorithms that analyze the spectral information of an image. This commonly means analyzing each pixel's brightness intensity (Bogunovic et al. 2014). 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 (Guada et al. 2016). These boundaries can be depicted on the digital image drawing a white line onto these selected pixels and setting the remaining ones as a black background.

Communicated by I. Perfilieva.

✉ Pablo A. Flores-Vidal
pflores@ucm.es

- ¹ Department of Statistics and Operation Research, Faculty of Mathematics of Complutense University, Madrid, Spain
- ² Department of Quantitative Methods, University College of Financial Studies, Madrid, Spain
- ³ Department of Statistics and Data Science, Faculty of Statistical Studies of Complutense University, Madrid, Spain

In the literature, many edge detectors have been developed. Some of them tried to detect if a certain pixel could be an edge by using only information provided by adjacent pixels—*neighbors*, while 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* (Sect. 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 decision strategy could be considered as a *local edge evaluation* (Kitchen and Rosenfeld 1981). The approach presented in Canny (1986) can be placed between a local and a 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 the edge detection process traditionally consisted of a simple pixel-by-pixel evaluation. Nevertheless, it seems that Canny's technique could be improved (Venkatesh and Rosin 1995, p. 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 Venkatesh and Rosin (1995), the idea of *edge segment* is developed (Sect. 2.1). In Canny (1986) and Kitchen and Rosenfeld (1981), it is employed a thresholding process that is 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, Lowe (1987) set out a different strategy which consisted in using the mean intensities of the pixels that made up the *edge list*. In Venkatesh and Rosin (1995), the concept of *edge list* was expanded with the use of another key feature: length.

This work is drawn on the use of *edge segments* (Sect. 2.1), and the methodology for creating them was introduced in Flores-Vidal et al. (2017) although it is expanded in this paper. From these edge segments, different features were extracted (Sect. 3.1). These were useful to discriminate the good from the bad edges. 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 (University 2017; Martin et al. 2001). Thus, the quality of the comparatives between our algorithm's output and the ground truth's is expected to be good.

The next section of this paper 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

(Venkatesh and Rosin 1995). 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 the final comments.

2 Preliminaries

In this section, some classical concepts of image processing and the edge extraction problem are introduced. 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, $n \times m$, equals the number of its horizontal pixels multiplied by its number of vertical ones. Let us denote by $I_{i,j}$ the spectral information associated with each pixel (i, j) (González and Woods 2008). The value range of this information will depend on the type of image considered, as shown in Fig. 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 (López-Molina et al. 2013):

1. *Conditioning* During this step, the image is well prepared for the next phases of edge detection. Traditionally, this consists in smoothing, de-noising, or other similar procedures (Basu 2002; Morillas et al. 2009). In practice, this phase basically helps making the edges easier to detect. After the conditioning phase, the resulting image is a grayscale image that we will denote as I^s .
2. *Feature extraction* Once the image is well prepared, the spectral differences between adjacent pixels are obtained (see for example Bezdek et al. 1998; Bustince et al. 2009; Kim et al. 2004). Then, the output of these differences is computed for each pixel (i, j) based on an operator function. For instance, if the operator is the one proposed by Sobel, for each pixel two values are obtained. 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

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

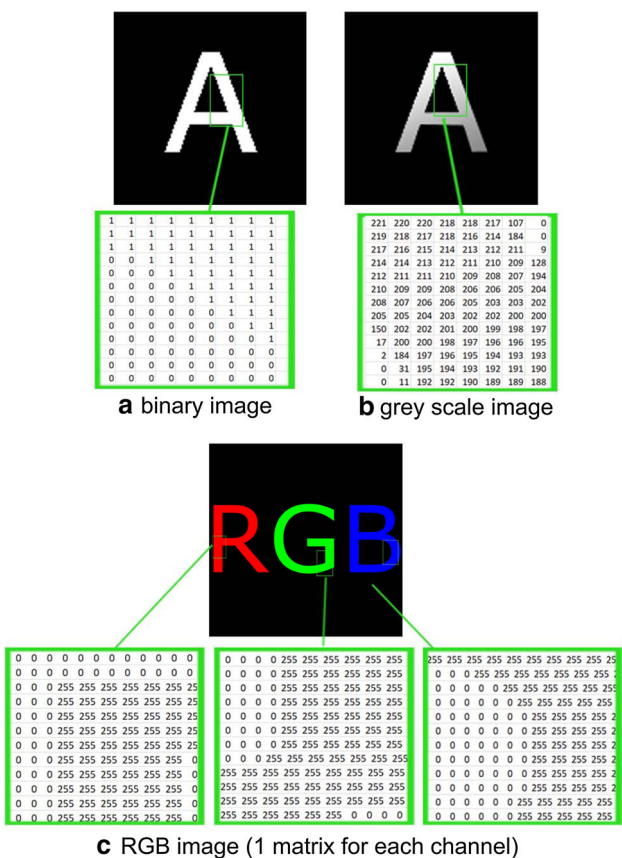


Fig. 1 Three types of images

be the horizontal variation, and let

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

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

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

$$I^{bf} = \phi(X^1, \dots, X^k)$$



Fig. 2 Four sequential phases of edge detection using Sobel operator

the aggregated image resulting from this step, where ϕ denotes an aggregation function. For a given pixel (i, j) , the value $I_{i,j}^{bf}$ 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* (Fig. 2.3). After this phase, our resulting image is I^{bf} .

- 4. **Scaling** In this last step, the final output with the definitive edges is necessarily created (Fig. 2.4). Traditionally, after Canny's constraints (Canny 1986), there are only two possibilities: any given pixel has to be declared as an edge or as a non-edge pixel. This decision is usually made through 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 shown in Fig. 4d. See Fig. 2 for the whole sequence of edge detection.

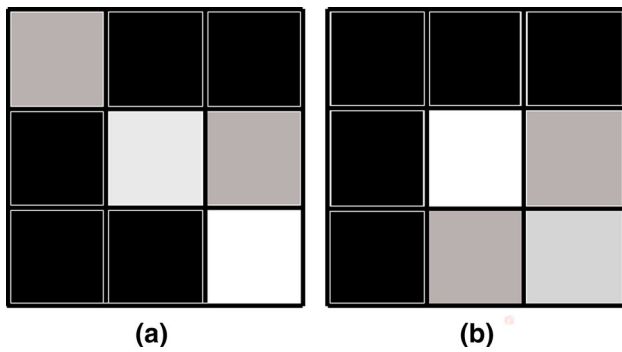


Fig. 3 Two kinds of 3×3 pixel windows from an image after going through the thinning process: **a** 4 candidates to be edge pixels; **b** 4 noncandidates to be edge pixels as square structures of 2×2 nonzero intensity pixels are not allowed (they are not thinned)

2.1 Edge detection based on segments

It is important to define the type of pixels our algorithm works with. Let c be a candidate to be an 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 $I_{i_c, j_c}^{bf} > 0$. In other words, it has to be a nonzero 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. This is then called a *thinned image*, which idea is shown in Fig. 3a).

From previous definitions, we are then able to define the set that contains these pixels. Let $C = \{c_1, \dots, c_m\}$ be the set of all the *edge candidate pixels* in an image. The connected pixels' notion is strongly related to the concept that will be explained 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 Fig. 4a. After the three aforementioned 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 and 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, $I_{i, j}^{bf}$, 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 evaluation pixel by pixel,

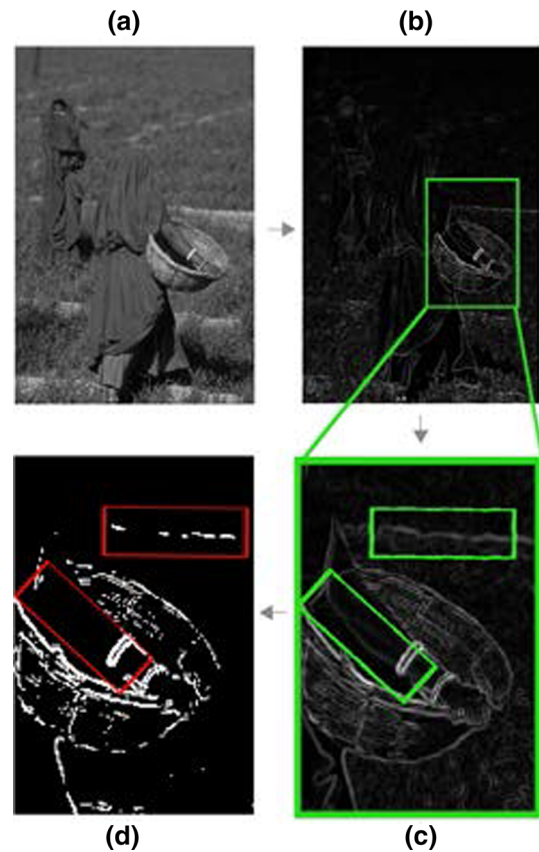


Fig. 4 Some limitations in local evaluated edges

which is the traditional way, we can consider it as a *local evaluation process* as it is argued in Venkatesh and Rosin (1995). Following this local evaluation approach, we could easily end up, for instance, in a situation like the one shown in Fig. 4.

In Fig. 4d, we see that some contours are extracted in a way that is too fragmented, losing 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 (Fig. 4c), we can 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 only happen in this image, as López-Molina et al. (2013) pointed out: *One of the most common mistakes in edge detection methods is not being able to complete the silhouette of an object*. In order to avoid this while 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 of a list of connected pixels—linked edges—that will be referred below as *edge segments*. For instance, in

Fig. 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 leads us toward a fuller definition of an important concept, defined below.

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 $S' \subset C$ is another connected set of *edge candidate pixels*, then $S \subset S' \Rightarrow S = S'$.

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. In Fig. 5, the whole process of building an edge segment can be seen.

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 feature values of the segments. All of this happens for each image. The strategy of applying the thresholding process was called as *dynamic threshold determination* (Venkatesh and Rosin 1995). However, we will address the idea of thresholding in a more statistical way, instead of using the classical approach. This new methodology is also used in Flores-Vidal et al. (2017), 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 the segment building, and before focusing on the rest of our proposal, we will explain the idea behind the algorithm proposed by Venkatesh and Rosin (1995), in which ours is based. After using the concept of *edge list*—equivalent to the concept of edge segment that we have formalized in this section—these authors established two features that defined the edge segment. The first one was the length of the edge segment (inspired by Lowe 1987), and the second one was the average intensity—edginess—of the pixels that made up the edge segment. Then, they built a two-dimensional feature space to represent the segment values

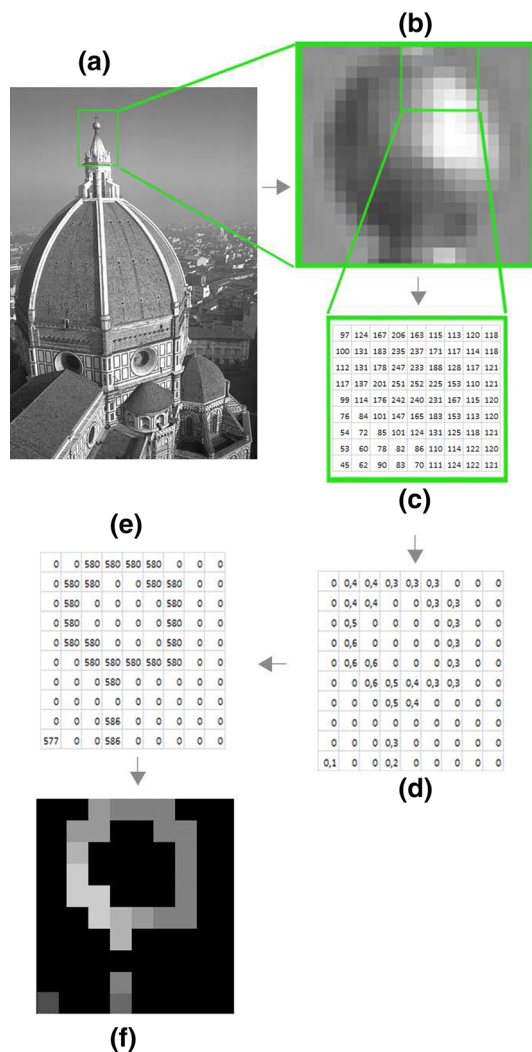


Fig. 5 From the original image to the segments

in these two variables. In Fig. 6 (taken from Venkatesh and Rosin 1995), we can see the scheme for generating the decision over the edge segments. The logic behind this geometric approach was to create a curve that separated the diagram in two parts, one containing the noise and the other containing 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. Even short ones with medium or high intensity were distinguishable too. On the other hand, there are disposable segments—the *false* segments—that will be those not so easily perceivable, this is, those that in addition to shortness present little average intensity.

The heuristic method proposed by Venkatesh and Rosin (1995) to make this decision is questionable, or at least improvable. From this point onward, we will propose one of the main differences with the work of these authors. We

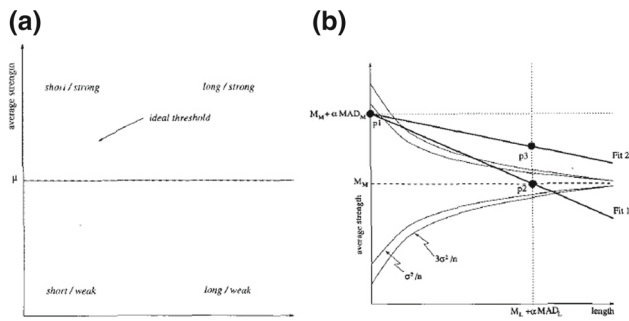


Fig. 6 Geometric approach of Venkatesh and Rosin (1995): the feature space of the edges (a) and the scheme for deciding over the segments (b)

will explain how the fuzzy clustering techniques are perfectly suited to choose which cluster each edge segment belongs to.

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 focused on performing a fuzzy clustering approach studied in Amo et al. (2001), specially designed for this work.

1. Given an already blended and thinned grayscale image I^{bf} , we have to obtain the set C (see Sect. 2.1 and Fig. 5) and the segments set $S = \{S_l : l = 1, \dots, s\}$ of the image I^{bf} .
2. For each segment S_l , we obtain the segment’s features (see Sect. 3.1 for further details). 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 $l = 1, \dots, s; r = 1, \dots, f$, 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 can apply a fuzzy clustering algorithm over the segments set, based on relevance, redundancy, and covering concepts (Amo et al. 2001) (see Sect. 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: $\mathbf{x}^l = (x_1^l, x_2^l, \dots, x_f^l) \in F = [0, 1]^f$ 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, this can be seen as the number of pixels in the segment.
- *Intensity mean*. For each segment S_l ,

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

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 ,

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

- *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^l is the average horizontal position of the pixels in S_l , i.e.,

$$x_7^l = \frac{\sum_{p=(p1, p2) \in S_l} p2}{x_1^l} \text{ and } x_8^l = \frac{\sum_{p=(p1, p2) \in S_l} p1}{x_1^l}.$$

Once the average position is computed, we get its Euclidean distance to the intersection points following the rule of thirds, which is a standard in photography composition (Goldstein 2009). 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 shown in Fig. 7.

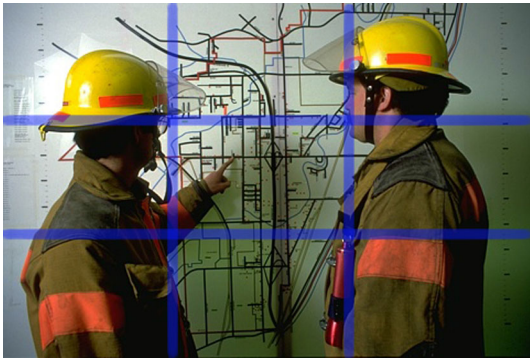


Fig. 7 Rule of thirds. In this example, faces tend to be closer to the intersection points

Most of these features were created specifically for our proposal using principles that come from the theory of human perception (Goldstein 2009). This entails an important improvement from our previous work (Flores-Vidal et al. 2017), as we have employed four new features in this paper. The last feature, the average position, is especially important from a theoretical point of view. The objects that a human recognizes tend to occupy certain positions in the image. This is why the information related to the position of the segment S_i inside the image I^{bf} is relevant—this relevance will later be confirmed in the experiment results—(see Sect. 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 edges 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 Venkatesh and Rosin (1995), the authors used heuristic techniques to solve this problem: They established two regions separated by a curve obtained by a heuristic method (Figs. 6 and 8). This method can be considered a *linear discriminant* type. Employing a thresholding value is common in the 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 purpose of this work, they do not perform well enough when the clusters are unbalanced, as is the case when real edges are few when compared to the non-edge segments (especially when there is too much noise in the image). In these situations, 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 Flores-Vidal et al. (2017) and Rojas et al. (2014), 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 on identifying the cluster centroids by taking into account three quality measures (Amo et al. 2001):

- *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 the three different criteria to be optimized. 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 , $K = \{\mathbf{y}^i : i = 1, \dots, k\} \subset F$; 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 $i = 1, \dots, k$.
- Select the most relevant cluster, i.e.,

$$i' = \arg \max_{i=1, \dots, k} \{r_i\}.$$

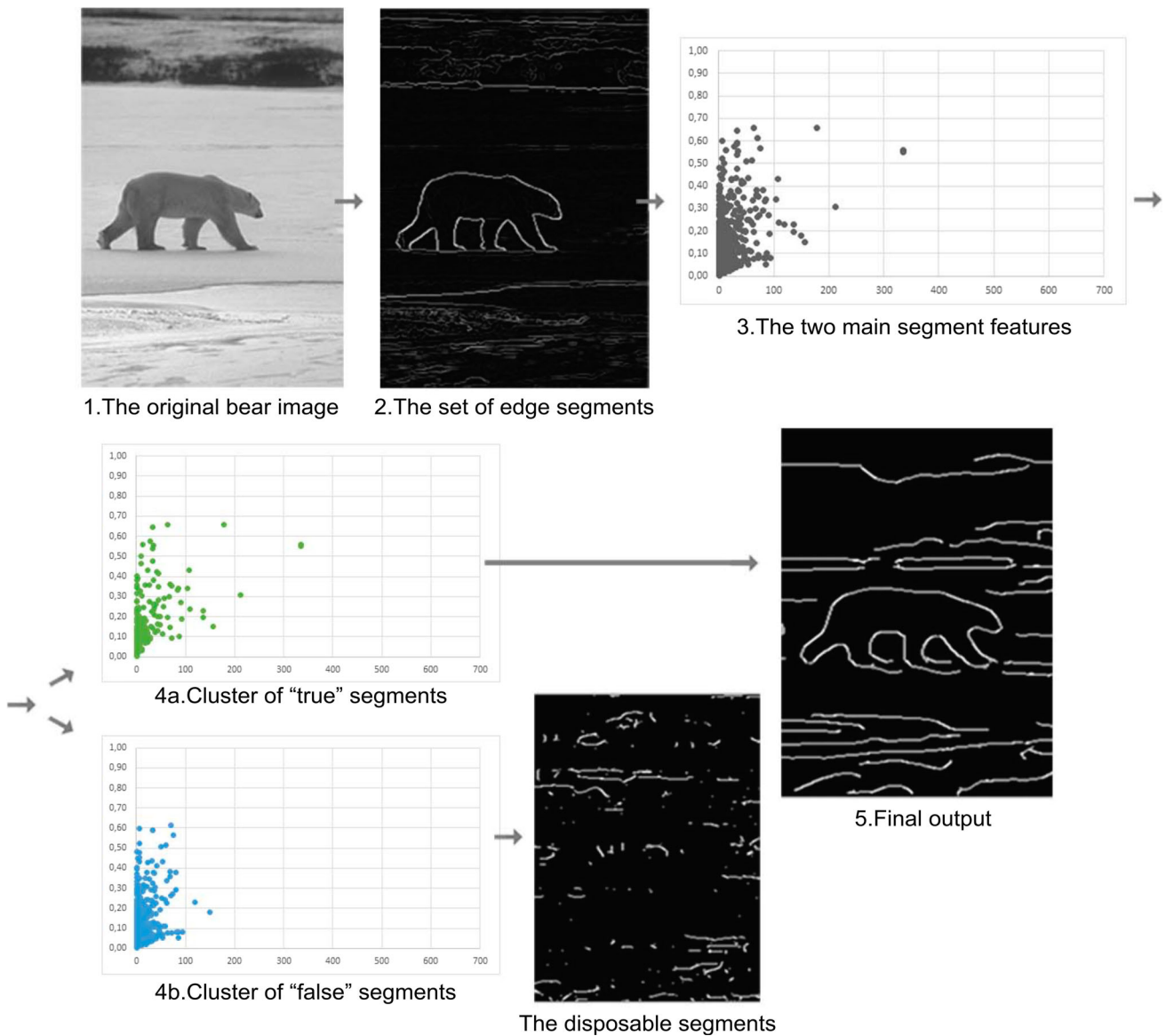


Fig. 8 An abbreviated scheme of our proposal

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

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

where $\mu_i(l)$ represents the degree of membership of segment S_l to the i th cluster, which is calculated as $\frac{1}{\|x^l - y^i\|}$ 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' and i are considered as redundant if $|D_i| \geq pr$. Now, calculate the set of clusters which are not redundant with i' , $D = \{l \in \{1, \dots, k\} : |D_l| < pr\}$. Finally, select the most relevant cluster, i'' , among those in D , i.e., $i'' = \arg \max\{r_l : l = 1, \dots, k\}$.

- If $\|y^{i'}\| < \|y^{i''}\|$ then interchange them, i.e., , since the edges should be those with greater norm in F .

We can see the whole approach in Fig. 8.

Let us now 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 $O(k \cdot s)$.

Let us study the meaning of this: 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 Heath et al. (1997) (and can be downloaded from University 2017). 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 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 to other five high standard edge detection algorithms by means of the matching technique proposed by Estrada and Jepson (2009). 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 advice. *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 Perfilieva et al. 2016). *Precision* measures the rate of edges selected by the algorithm that match to 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 by the algorithms output as well. 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* (Martin et al. 2001) 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 (University 2017). Firstly, the dataset was shortened by name, and then the 35 images placed in the middle—from “131” to “cone”—were used as the training set; meanwhile, the other 25 images were used as 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 (all but Gravitation’s and F^1 -transform’s), with a Gaussian smooth filter ($\sigma_s = 1$) and without it ($\sigma_s = 0$):

1. *Canny algorithm* The “sigma of Canny” parameter (σ_{Canny}) was applied. This is the Gaussian filter that works in the convolution of Canny’s and produces even smoother edges. In our algorithm, the higher the parameter is, the less amount of edge segments are selected. Different values were explored for that parameter: 0.5, 1, 2, and 4. After that, the well-known non-maximum suppression for the “thinning” process was applied. For the scaling step, the double threshold called “Hysteresis” was applied (Canny 1986).
2. *F^1 -Transform algorithm* This is the F^1 -transform method used for preprocessing in Canny’s (Perfilieva et al. 2012, 2016). 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 the h is, the smoother the smoothed image I^s results. Values $h = 3$, in the first step, and $h = 2$ were, respectively, applied. The next steps were Canny’s usual: non-maximum suppression and hysteresis.
3. *Gravitation algorithm* This is an edge detection algorithm based on the Law of Universal Gravity by Isaac Newton. This algorithm computes the gradient at each of the pixels using the gravitational approach based on a t-norm (hence it is named GED-T). See details in López-Molina et al. (2010). This method does not perform any of the other processes of the image needed to obtain binary edges, e.g., smoothing, binarization. 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 López-Molina et al. (2010). In our case, we have used the Lukasiewicz and the Nilpotent minimum t-norms. As with the F-transform algorithm, the last steps are

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

	F		Precision		Recall	
	$\sigma_s = 0$	$\sigma_s = 1$	$\sigma_s = 0$	$\sigma_s = 1$	$\sigma_s = 0$	$\sigma_s = 1$
Canny ($\sigma_{\text{Canny}} = 2, T.H = 0.26, T.L = 0.10$)	0.73	0.72	0.75	0.76	0.74	0.72
GED-T Lukastewicz t-norm ($\sigma_{\text{Canny}} = 2, T.H = 0.28, T.L = 0.11$)	0.62	-	0.72	-	0.59	-
GED-T Nilpotent t-norm ($\sigma_{\text{Canny}} = 2, T.H = 0.19, T.L = 0.08$)	0.64	-	0.59	-	0.74	-
F^1 transform ($h = 3, T.H = 0.18, T.L = 0.10$)	0.66	-	0.63	-	0.74	-
Sobel ($T = 0.19$)	0.72	0.73	0.73	0.72	0.77	0.77
Venkatesh and Rosin ($\sigma_{\text{Canny}} = 2, \alpha = 6$)	0.70	0.70	0.68	0.71	0.74	0.70
FCS ($\sigma_{\text{Canny}} = 1.75, \text{relev} = 0.90, \text{overlap} = 0.65$ and $\text{overlap} (\%) = 15\%$)	0.74	0.73	0.75	0.77	0.75	0.72

Bold represents highest value in each column

- the usual of Canny’s (non-maximum suppression and hysteresis).
4. *Sobel algorithm* This is the classical algorithm that was proposed by Sobel and Feldman (1968) in a talk . 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 The Kermit 2017) of the proposal presented in Venkatesh and Rosin 1995 by which our work is inspired (Sect. 2.2). Like with our algorithm, first steps were the same as Canny’s (until the non-maximum suppression). The alpha parameter is the only one specifically required ($\alpha = 6$).
 6. *Fuzzy cluster of segments algorithm* The first few steps were the same as Canny’s (until the non-maximum suppression). Then, at the scaling step FCS was implemented for three different quality measures that range from 0 to 1 (see Sect. 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 and, thus, 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 did not change much when this parameter took values inside the percentage range that seemed to us reasonable for the redundancy (between 5 and 20%).

Notice that our FCS algorithm had the best performance—even slightly better than Canny’s and Sobel’s—compared to the other five edge detection algorithms (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) while comparing the algorithms output with the humans’ ground truth.

We can see in Fig. 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 while 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.

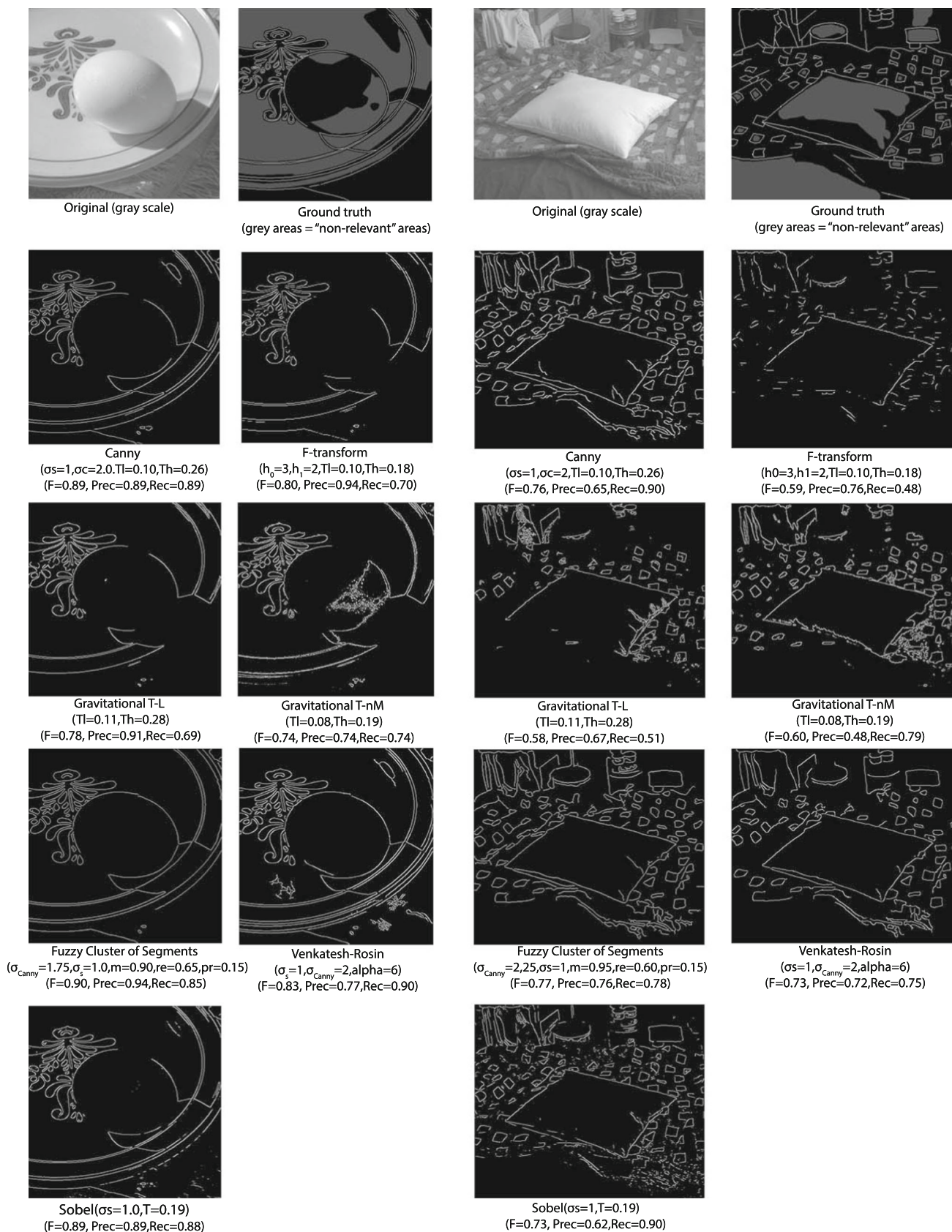


Fig. 9 Binarized outputs of USF dataset images for different algorithms with best fixed parameters. σ_s Gaussian smoothness, Tl lower threshold, Th higher threshold, n operator dimension, m relevance, re redundancy, pr redundancy % allowed (see Sect. 4 for more details)

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*. This seems like a complex line that could lead 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 Guada et al. (2016) and Gómez et al. (2015a, b).

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 adapting the current human ground truth into a modified version of it would be necessary to. We believe that this could lead to an interesting future research line about edge segments, and maybe this could be done as well in a supervised approach.

For the conducting of this research, the code created by Kermit Research Unit has been strongly helpful (The Kermit 2017).

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

Compliance with ethical standards

Conflict of interest All authors declare that they have 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

- Basu M (2002) Gaussian-based edge-detection methods—a survey. *IEEE Trans Syst Man Cybern Part C Appl Rev* 32(3):252–260
- Bezdek J, Chandrasekhar R, Attikouzel Y (1998) A geometric approach to edge detection. *IEEE Trans Fuzzy Syst* 6(1):52–75
- Bogunovic H, Sonka M, Kwon YH, Kemp P, Abramoff MD, Wu X (2014) Multi-surface and multi-field co-segmentation of 3-D retinal optical coherence tomography. *IEEE Trans Med Imag* 33(12):2242–2253. <https://doi.org/10.1109/TMI.2014.2336246>
- Bustince H, Barrenechea E, Pagola M, Fernandez J (2009) Interval-valued fuzzy sets constructed from matrices: application to edge detection. *Fuzzy Sets Syst* 160(13):1819–1840
- Canny J (1986) A computational approach to edge detection. *IEEE Trans Pattern Anal Mach Intell PAMI* 8(6):679–698
- Daňková M, Hodáková P, Perfilieva I, Vajtl M (2011) Edge detection using F-transform. In: 11th international conference on intelligent systems design and applications. IEEE, pp 672–677
- del Amo A, Gómez D, Montero J, Biging G (2001) Relevance and redundancy in fuzzy classification systems. *Mathw Soft Comput* 8:203–216
- Estrada FJ, Jepson AD (2009) Benchmarking image segmentation algorithms. *Int J Comput Vis* 85(2):167–181
- Fathy M, Siyal MY (1995) An image detection technique based on morphological edge detection and background differencing for real-time traffic analysis. *Pattern Recognit Lett* 16(12):1321–1330
- Flores-Vidal PA, Gómez D, Olaso P, Guada C (2017) A new edge detection approach based on fuzzy segments clustering. *Adv Fuzzy Log Technol* 2017:58–67
- Goldstein EB (2009) *Sensación y percepción*, 6th edn. Thomson Editores, Madrid
- Gómez D, Zarrazola E, Yáñez J, Montero J (2015a) A Divide-and-Link algorithm for hierarchical clustering in networks. *Inf Sci* 316:308–328
- Gómez D, Yáñez J, Guada C, Rodríguez J, Montero J, Zarrazola E (2015b) Fuzzy image segmentation based upon hierarchical clustering. *Knowl Based Syst* 87:26–37
- González RC, Woods RE (2008) *Digital image processing*, 3rd edn. Pearson, Prentice Hall
- Guada C, Gómez D, Rodríguez JT, Yáñez J, Montero J (2016) Classifying images analysis techniques from their output. *Int J Comput Intell Syst* 9(Supplement 1):43–68
- Heath M, Sarkar S, Sanocki T, Bowyer KW (1997) A robust visual method for assessing the relative performance of edge-detection algorithms. *IEEE Trans Pattern Anal Mach Intell* 19(12):1338–1359
- Kim DS, Lee WH, Kweon IS (2004) Automatic edge detection using 3×3 ideal binary pixel patterns and fuzzy-based edge thresholding. *Pattern Recognit Lett* 25(1):101–106
- Kitchen L, Rosenfeld A (1981) Edge evaluation using local edge coherence. *IEEE Trans Syst Man Cybern* 11(9):597–605
- López-Molina C, Bustince H, Fernández J, Couto P, De Baets B (2010) A gravitational approach to edge detection based on triangular norms. *Pattern Recognit* 43:3730–3741
- López-Molina C, De Baets B, Bustince H (2013) Quantitative error measures for edge detection. *Pattern Recognit* 46(4):1125–1139
- Lowe DG (1987) Three-dimensional object recognition from single two-dimensional images. *Artif Intell* 31(3):355–395
- Marr D (1982) *Vision: a computational investigation into the human representation and processing of visual information*. The MIT Press, Cambridge
- Marr D, Hildreth E (1980) Theory of edge detection. *Proc R Soc Lond Biol Sci* 207(1167):187–217
- Martin D, Fowlkes C, Tal D, Malik J (2001) A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. *Proc IEEE Int Conf Comput Vis* 2:416–423
- Monga O, Deriche R, Malandain G, Cocquerez JP (1991) Recursive filtering and edge tracking: two primary tools for 3D edge detection. *Image Vis Comput* 9(4):203–214
- Morillas S, Gregori V, Hervas A (2009) Fuzzy peer groups for reducing mixed Gaussian-impulse noise from color images. *IEEE Trans Image Process* 18(7):1452–1466

- Pal S, King R (1983) On edge detection of X-ray images using fuzzy sets. *IEEE Trans Pattern Anal Mach Intell* 5(1):69–77
- Perfilieva I, Hodáková P, Hurtík P (2012) Differentiation by the F-transform and application to edge detection. In: *IPMU 2012, CCIS*, vol 297, pp 230–239
- Perfilieva I, Hodáková P, Hurtík P (2016) Differentiation by the F-transform and application to edge detection. *Fuzzy Sets Syst* 288:96–114
- Rojas K, Gómez D, Montero J, Rodríguez JT, Valdivia A, Paiva F (2014) Development of child's home environment indexes based on consistent families of aggregation operators with prioritized hierarchical information. *Fuzzy Sets Syst* 241:41–60
- Sobel I, Feldman G (1968) A 3×3 isotropic gradient operator for image processing, Presented at a talk at the Stanford Artificial Project
- The Kermit Image Toolkit (KITT), Ghent University, De Baets B, Lopez-Molina C (2017) www.kermitimagetoolkit.net, 27 Sept 2017
- University of South Florida Dataset: [ftp://figment.csee.usf.edu/pub/ROC/edge_comparison_dataset.tar.gz](http://figment.csee.usf.edu/pub/ROC/edge_comparison_dataset.tar.gz), 27 Sept 2017
- Venkatesh S, Rosin PL (1995) Dynamic threshold determination by local and global edge evaluation. *Graph Models Image Process* 57(2):146–160
- Zielke T, Brauckmann M, Vonseelen W (1993) Intensity and edge-based symmetry detection with an application to car-following. *CVGIP Image Underst* 58(2):177–190

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.