International Journal of Computational Intelligence Systems, Vol. 9, Supplement 1 (2016), 43-68

Classifying image analysis techniques from their output

C. Guada [1] *, D. Gómez [2], J.T. Rodríguez [1], J. Yáñez [1], J. Montero [1,3]

[1] *Facultad de Ciencias Matemáticas, Complutense University,*
Plaza de las Ciencias 3,
Madrid, 28040, Spain†

E-mail: cguada@ucm.es, jtrodriguez@mat.ucm.es, jayage@ucm.es, monty@mat.ucm.es

[2] *Facultad de Estudios Estadísticos, Complutense University,*
Av. Puerta de Hierro s/n,
Madrid, 28040, Spain ‡

E-mail: dagomez@estad.ucm.es

[3] *Instituto de Geociencias IGEO (CSIC, UCM),*
Plaza de las Ciencias 3,
Madrid, 28040, Spain§

Received 5 November 2015

Accepted 12 March 2016

Abstract

In this paper we discuss some main image processing techniques in order to propose a classification based upon the output these methods provide. Because despite a particular image analysis technique can be supervised or unsupervised, and can allow or not the existence of fuzzy information at some stage, each technique has been usually designed to focus on a specific objective, and their outputs are in fact different according to each objective. Thus, they are in fact different methods. But due to the essential relationship between them they are quite often confused. In particular, this paper pursues a clarification of the differences between image segmentation and edge detection, among other image processing techniques.

Keywords: Image segmentation, image classification, edge detection, fuzzy sets, machine learning, graphs.

1. Introduction

Image analysis or image processing has become a scientific hot topic during the last decades, particularly because of the increasing amount of relevant information being stored in this format. Image analysis has a wide range of applications in different areas as remote sensing, image and data storage for transmission in business applications, medical imaging, acoustic imaging and security, among many other fields. Digital image processing techniques are being increasingly demanded through all areas of science and industry.

Many different techniques are considered "image

* Facultad de Ciencias Matemáticas, Complutense University, Plaza de las Ciencias 3, Madrid, 28040, Spain.
† Facultad de Ciencias Matemáticas, Complutense University, Plaza de las Ciencias 3, Madrid, 28040, Spain.
‡ Facultad de Estudios Estadísticos, Complutense University, Av. Puerta de Hierro s/n, Madrid, 28040, Spain.
§ Instituto de Geociencias IGEO (CSIC-UCM), Plaza de las Ciencias 3, Madrid, 28040, Spain.

Co-published by Atlantis Press and Taylor & Francis

processing" or "image analysis" techniques. Usually, each technique is appropriate for some a small range of tasks or for a specific problem.

For example, machine learning algorithms used in image analysis help in finding solutions to many problems in speech recognition [1], robotics [2] and vision [3]. Computer vision uses image processing algorithms to extract significant information automatically from images. A number of algorithms are available to analyze images and recognize objects in them, and depending on the delivered output and the manner in which the images were encoded, specific learning methodologies are needed [4].

Another well-known example is image segmentation. Image segmentation methods try to simplify an image into something of easier analysis. Usually the image is transformed into segments or regions. These regions are supposed to be connected and represent a set of homogeneous pixels. Nevertheless, it is not clear when certain techniques should be considered as image segmentation methods, and such a situation also applies to other image processing techniques. It is quite common to classify two methods in the same group of techniques even if the output and the information they provide are very different. This is the case, for example, of edge detection methods [5], clustering image [6] and image thresholding algorithms [7], all of them classified as "image segmentation procedures" despite the obvious differences in their outputs.

There is a significant difference between image classification methods, edge detection methods, image segmentation methods and hierarchical image segmentation methods. Particularly, besides the strong relationship between them, all these methods address different problems, and produce different outputs. However, those different methods are not always clearly differentiated in the literature, and sometimes even confused. This paper poses a critical view on some of these techniques in order to show the conceptual differences between them. Depending on how a particular algorithm detects objects in an image and shows the output, the method might be understood as either performing classification, detection, segmentation or hierarchical segmentation.

Hence, the main objective of this paper is to present a classification of a set of widely used image processing algorithms, attending to the problems they face, the learning scheme they are based on, and the representational framework they utilize. Some new image analysis concepts and methods are also provided.

In order to understand the remainder paper, the following notation is adopted. The image we consider is being modeled as a two-dimensional function, where x and y are the coordinates in a plane, and $f(x,y)$ represents each pixel by a fixed number of measurable attributes [8]. Formally, an image can be defined as $I = \{f(x,y); x = 1,\ldots,n \text{ and } y = 1,\ldots m\}$ which can be represented computationally in:

- Binary $\equiv f(x,y) \in \{0,1\}$.
- Gray $\equiv f(x,y) \in \{0,\ldots,255\}$.
- RGB $\equiv f(x,y) \in \{0,\ldots,255\}^3$.

A binary image is represented as a matrix allowing two possible values for each cell, two-tone colors (usually 0 refers to black and 1 to white). Similarly, a grey scale image may be defined as a two dimensional function $f(x,y)$ where the amplitude of f at any pair of coordinates (x,y) refers to the intensity (gray level) of the image at that point. Instead, a color image is defined by a combination of individual 2D images. For instance, the RGB color system represents a color image in three components (red, green and blue). That is, it has an array of three matrices of equal size where the intensity of color of a pixel compound is represented by the three colors [8].

Figures 1, 2 and 3 show a binary image, a grayscale image and RGB image, respectively.

The remainder of this paper is organized as follows: a review on image classification is presented in Section 2. Section 3 is devoted to fuzzy image classification. Some edge detection techniques are shown in Section 4, which is extended to fuzzy edge detection in Section 5. Image segmentation and some methods addressing this problem are analyzed in Section 6. Section 7 pays attention to hierarchical image segmentation. In Section 8 fuzzy image segmentation is addressed. Finally, some conclusions are shed.

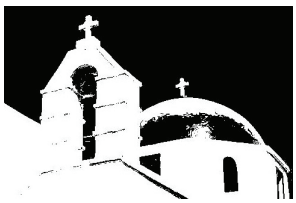

Fig. 1. Binary image.

Fig. 2. Gray level image.

Fig. 3. RGB image [9].

2. Image classification

Classification techniques have been widely used in image processing to extract information from images by assigning each pixel to a class. Therefore, two types of outputs can be obtained at the end of an image classification procedure. The first kind of output is a thematic map where pixels are accompanied by a label for identification with a class. The second kind of output is a table summarizing the number of image pixels belonging to each class. Furthermore, both supervised and unsupervised learning schemes are applied for this image classification task [10]. Next we provide an overview of these methods.

2.1. Supervised classification

Supervised classification is one of the most used techniques for image analysis, in which the supervisor provides information to the system about the categories present in each pattern in the training set.

Neural networks [11], genetic algorithms [12], support vector machines [13], bayesian networks [14], maximum likelihood classification [15] or minimum distance classification [16] are some techniques for supervised classification. Depending on certain factors such as data source, spatial resolution, available classification software, desired output type and others, it is more appropriate to use one of them to process a given image.

Supervised classification procedures are essential analytical tools for extracting quantitative information from images. According to Richards and Jia [4], the basic steps to implement these techniques are the following:

- Decide the classes to be identified in the image.
- Choose representative pixels of each class which will form the training data.
- Estimate the parameters of the algorithm classifier using the training data.
- Categorize all the remaining pixels in the image with the classifier in each of regions desired.
- Summarize the classification results in tables or display the segmented image.
- Evaluate the accuracy of the final model using a test data set.

A variety of classification techniques such as those mentioned above have been used in image analysis. In addition, some of those methods have been jointly used (e.g., neural networks [17,18], genetic algorithm with multilevel thresholds [19] or some variant of support vector machines [20]).

In general, supervised classification provides good results if representative pixels of each class are chosen for the training data [4]. Next, we pay attention to methods for unsupervised classification.

2.2. Unsupervised classification, clustering

Unsupervised classification techniques are intended to identify groups of individuals having common characteristics from the observation of several variables for each individual. In this sense, the main goal is to find regularities in the input data, because unlike supervised classification techniques, there is no supervisor to provide existing classes. The procedure tries to find groups of patterns, focussing in those that occur more frequently in the input data [21].

According to Nilsson [22], unsupervised classification consists of two stages to find patterns:

- Form a partition R of the set Ξ of unlabeled training patterns. The partition separates Ξ into mutually exclusive and exhaustive subsets R known as clusters.
- Design a classifier based on the labels assigned to the training patterns by the partition.

Clustering [23] is perhaps the most extended method of unsupervised classification for image processing, which obtains groups or clusters in the input image.

Particularly, cluster analysis looks for patterns in images, gathering pixels within natural groups that make sense in the context studied [21]. The aim is that the proposed clustering has to be somehow optimal in the sense that observations in a cluster have to be similar, and in turn be dissimilar to those of other clusters. In this way, we try to maximize the homogeneity of objects within the clusters while the heterogeneity between aggregates is maximized. In principle, the number of groups to form and the groups themselves is unknown [24].

A standard cluster analysis creates groups that are as homogeneous as possible, and in turn the difference between the various groups is as large as possible. The analysis consist on the main following steps:

- First, we need to measure the similarity and dissimilarity between two separate objects (similarity measures the closeness of objects, so closer values suggest more homogeneity).
- Next, similarity and dissimilarity between clusters is measured so that the difference between groups become larger and thus group observations become as close as possible.

In this manner, the analysis can be divided into these two basic steps, and in both steps we can be use correlation measures or any distance [25] as similarity measure. Some similarity and dissimilarity measures for quantitative features are shown in [26].

Correlation measurements refer to the measurement of similarity between two objects in a symmetric matrix. If the objects tend to be more similar, the correlations are high. Conversely, if the objects tend to be more dissimilar, then the correlations are low. However, because these measures indicate the similarity by pattern matching between the features and do not observe the magnitudes of the observations, they are rarely used in cluster analysis.

Distance measurements represent the similarity and proximity between observations. Unlike correlation measurements, these similarity measures of distance or proximity are widely used in cluster analysis. Among the distance measures available, the most common are the Euclidean distance or distance metric, the standardized distance metric or the Malahanobis distance [26].

Let $X(n \times p)$ be a symmetric matrix with n observations of p variables. The similarity between the observations can be described by the matrix $D(n \times n)$:

$$D = \begin{pmatrix} d_{11} & d_{12} & \dots & \dots & d_{1n} \\ \vdots & d_{22} & & & \vdots \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ d_{n1} & d_{n2} & \dots & \dots & d_{nn} \end{pmatrix}$$

where d_{ij} represent the distance between observations i and j. If d_{ij} is a distance, then $d'_{ij} = \max_{i,j}\{d_{ij}\} - d_{ij}$ represents a measure of proximity [25].

Once a measure of similarity is calculated, we are able to proceed with the formation of clusters. A popular algorithm is k-means clustering [27]. The algorithm was first proposed in 1957 by Lloyd [28] and published much later [29]. K-means clustering algorithm classifies the pixels of the input image into multiple classes according to the distance from each other. The points are clustered around centroids $\mu_i, i = 1, \dots k$ obtained by minimizing the Euclidean distance. Let x_1 and x_2 be two pixels whose Euclidean distance between them is:

$$d(x_1, x_2) = \sqrt{\sum_{i=1}^{N} (x_{1i} - x_{2i})^2} \,. \qquad (1)$$

Thus, the sum of distances to k-center is minimized, and the objective is to minimize the maxi-

mum distance from every point to its nearest center [30].

Then, having selected the centroids, each observation is assigned to the most similar cluster based on the distance of the observation and the mean of the cluster. The average of the cluster is then recalculated, beginning an iterative process of centroid location depending on how observations are assigned to clusters; until the criterion function converges [31].

Clustering techniques have been used to perform unsupervised classification for image processing since 1969, when an algorithm finding boundaries in remote sensing data was proposed [32]. Clustering implies grouping the pixels of an image in the multispectral space [4]. Therefore, clustering seeks to identify a finite set of categories and clusters to classify the pixels, having previously defined a criteria of similarity between them. There are a number of available algorithms [33].

The resulting image of the k-means clustering algorithm with *k=3* over Figure 3 is shown in Figure 4.

Fig. 4. Cluster's result.

Thus, to achieve its main goal, any clustering algorithm must address three basic issues: *i)* how to measure the similarity or dissimilarity?; *ii)* how the clusters are formed?; and *iii)* how many groups are formed?. These issues have to be addressed so the principle of maximizing the similarity between individuals in each cluster and maximizing the dissimilarity between clusters can be met [33]. A detailed review on clustering analysis is presented in [6].

Thus, summarizing, in the framework of supervised classification pixels are assigned to a known number of predefined groups. However, in cluster analysis the number of cluster groups and the groups themselves are not known in advance [24]. Anyway, image classification is a complex process for pattern recognition based on the contextual information of the analyzed images [34]. A more detailed discussion of image classification methods is presented in [35].

3. Fuzzy image classification

In this section, we present an overview of fuzzy image classification. As in Section 2, these methods are divided in supervised techniques and unsupervised techniques.

3.1. Supervised fuzzy classification

Fuzzy classification techniques can be considered extensions of classical classification techniques. We simply need to take advantage of the fuzzy logic proposed to Zadeh in such a way that restricted applications within a crisp framework we produce all those techniques presented in Section 2.1. And similarly to traditional classification techniques, fuzzy classification includes both supervised and unsupervised methods.

Fuzzy logic [36] was introduced in the mid-sixties of last century as an extension of classical binary logic. Some objects have a state of ambiguity regarding their membership to a particular class. In this way, a person may be both "tall" and "not tall" to some extent. The difference lies in the degree of membership assigned to the fuzzy set "tall" and its complement "not tall". Therefore, the intersection of a fuzzy set and its complement is not always an empty set. Also, conflictive views can simultaneously appear within a fuzzy set [37]. Such a vagueness is in the roots of fuzzy logic, and brings specific problems difficult to be treated by classical logic. Still, these problems are real and should be addressed [38].

A fuzzy set C refers to a class of objects with membership degrees. This set is usually characterized by a continuous membership function $\mu_C(x)$ that assigns to each object a grade of membership in $[0,1]$ indicating the degree of membership of x in C. The closer μ_C is to 1, the greater the degree of membership of x in C [36]. Classical overviews of fuzzy sets can be found in [36,39,40,41,42,43].

Hence, classes are defined according to certain attributes, and objects that possess these attributes belong to the respective class. Thus, in many appli-

cations of fuzzy classification, we consider a set of fuzzy classes $\mathscr{C}$. The degree of membership $\mu_C(x)$ of each object $x \in X$ to each class $C \in \mathscr{C}$ has to be then determined [44].

The membership function is given by $\mu_c : X \to [0,1]$ for each class $c \in \mathscr{C}$ [44], where a quite complete framework is proposed, subject to learning).

Fuzzy rule-based systems (FRBS) are methods of reasoning, where knowledge is expressed by means of linguistic rules. FRBS are widely used in various contexts, as for example fuzzy control [45] or fuzzy classification [46]. FRBS allow the simultaneous use of certain parts of this knowledge to perform inference.

The main stages of a FRBS are listed below:

- Fuzzification, understood as the transformation of the crisp input data into fuzzy data [47].
- Construction of the fuzzy rule base, expressed through linguistic variables, i.e., *IF antecedent THEN result* [48,49,50].
- Inference over fuzzy rules, where we find the consequence of the rule and then combine these consequences to get an output distribution [51].
- Defuzzification, to produce a crisp value from the fuzzy or linguistic output obtained from the previous inference process [37,52].

FRBS are commonly applied to classification problems. These classifiers are called fuzzy rules based classification systems (FRBCS) or fuzzy classifiers [53].

Fuzzy classification is the process of grouping objects in a family of fuzzy sets by assigning degrees of membership to each of them, defined by the truth value of a fuzzy propositional function [44].

One of the advantages of fuzzy classifiers is that they do not assigned two similar observations to different classes in case the observations are near the boundaries of the classes. Also, FRBCS facilitate smooth degrees of membership in the transitions between different classes.

3.2. Unsupervised classification, fuzzy clustering

Regarding unsupervised techniques, perhaps the most widely used technique is fuzzy c-means, that extends clustering to a fuzzy framework.

The fuzzy c-means algorithm [54] constitutes an alternative approach to the methods defined in subsection 2.2, based on fuzzy logic. Fuzzy c-means provide a powerful method for cluster analysis.

The objective function of the fuzzy c-means algorithm, given by Bezdek [55], is as follows:

$$J_m(U,v) = \sum_{k=1}^{N} \sum_{i=1}^{c} u_{ik}^m \|y_k - v_i\|_A^2. \qquad (2)$$

where $Y = \{y_1, y_2, \ldots, y_N\} \subset \Re^n$ are the observations, c are the numbers of clusters in $Y; 2 \leqslant c < n$, $m; 1 \leqslant m < \infty$ is the weighting exponent which represents the degree of fuzziness, $U; U \in M_{fc}$, $v = (v_1, v_2, \ldots, v_c)$ are the vectors of centers, $v_i = (v_{i1}, v_{i2}, \ldots, v_{in})$ is the centroid of cluster i, $\| \quad \|_A$ introduce A-norm above $\Re^n$, and A is positive weight matrix $(n \times n)$.[56]

According to Bezdek *et al.* [56], the fuzzy c-means algorithm basically consists of the following four steps:

- Set $c, m, A, \|k\|_A$ and choose a matrix $U^{(0)} \in M_{fc}$.
- Then at step k, $k = 0, 1, \ldots, LMAX$, calculate the mean $v^{(i)}, i = 1, 2, \ldots, c$ with $\widehat{v}_i = \frac{\sum_{k=1}^{N} (\widehat{u}_{ik})^m y_k}{\sum_{k=1}^{N} (\widehat{u}_{ik})^m}$ where $1 \leqslant i \leqslant c$.
- Calculate the updated membership matrix $\widehat{U}^{(k+1)} = [\widehat{u}_{ij}^{(k+1)}]$ with $\widehat{u}_{ik} = (\sum_{j=1}^{c} (\frac{\widehat{d}_{ik}}{\widehat{d}_{jk}})^{\frac{2}{(m-1)}})^{-1}$; $1 \leqslant k \leqslant N; \quad 1 \leqslant i \leqslant c$.
- Compare $\widehat{U}^{(k+1)}$ and $\widehat{U}^{(k)}$ in any convenient matrix norm. If $\|\widehat{U}^{(k+1)} - \widehat{U}^{(k)}\| < \varepsilon$ stop, otherwise set $\widehat{U}^{(k)} = \widehat{U}^{(k+1)}$ and return to the second step.

Fuzzy techniques have found a wide application into image processing (some studies can be found in [57,58,59]). They often complement the existing techniques and can contribute to the development of more robust methods [60]. Applying the technique of fuzzy c-means on Figure 2, the resulting image is shown in Figure 5.

Fig. 5. Fuzzy c-means's result.

As it has been noted, the fuzzy c-means produces a fuzzy partition [61] of the input image characterizing the membership of each pixel to all groups by a degree of membership [56].

In conclusion, fuzzy classification assigns different degrees of membership of an object to the defined categories.

4. Edge detection

Edge detection has been an important topic in image processing for its ability to provide relevant information of the image and provide the boundaries of objects of interest.

Edge detection is a useful and basic operation to acquire information about an image, such as its size, shape and even texture. This technique is considered the most common approach to detect significant discontinuities in the values of intensities of an image, and this is achieved by taking spatial derivatives of first and second order (typically with the gradient and Laplacian, respectively). That is, non smooth changes in the function $f(x,y)$ of the image can be determined with the derivatives. Thus, the operators that describe edges are tipically expressed by partial derivatives [8,62].

The result obtained with this technique consists of a binary image such that those pixels where sharp changes have occurred appear bright, while the other pixels remain dark. This output in turn allows a significant reduction on the amount of information while preserving the important structural properties of the image.

The first derivative is determined by the gradient, which is able to determine a change in the intensity function through a one-component vector (direction) pointing in the direction of maximum growth of the image's intensity [62]. Thus, the gradient of a two-dimensional intensity function $f(x,y)$ is defined as:

$$\nabla f \equiv \text{grad(f)} = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}. \tag{3}$$

The magnitude of this vector is:

$$\nabla f = mag(\nabla f) = \sqrt{g_x^2 + g_y^2} = \sqrt{(\partial f/\partial x)^2 + (\partial f/\partial y)^2}. \tag{4}$$

And the direction is given by the angle:

$$\alpha(x,y) = \tan^{-1}\left[\frac{g_y}{g_x}\right]. \tag{5}$$

However, the errors associated with the approximation may cause errors which require adequate consideration. For example, an error may be that the edges are not detected equally well in all directions, thus leading to erroneous direction estimation (anisotropic detecting edges) [10].

Therefore, the value of the gradient is related to the change of intensity in areas where it is variable, and is zero where intensity is constant [8].

The second derivative is generally computed using the Laplace operator or Laplacian, which for a two-dimensional function $f(x,y)$ is formed from the second order derivatives[8]:

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2}. \tag{6}$$

Remarkably, the Laplacian is rarely used directly because its sensitivity to noise and its magnitude can generate double edges. Moreover, it is unable to detect the direction of the edges, for which it is used in combination with other techniques [8].

Thus, the basic purpose of edge detection is to find the pixels in the image where the intensity or brightness function changes abruptly, in such a way that if the first derivative of the intensity is greater in magnitude than a predetermined threshold, or if the second derivative crosses zero, then a pixel is declared to be an edge [8,62]. In practice, this can be determined through the convolution of the image masks, which involves taking a 3×3 matrix of numbers and multiply pixel by pixel with a 3×3 section of the image. Then, the products are added and the result is placed in the center pixel of the image [63].

There are several algorithms which implement this method, using various masks [8,64]. For example, the Sobel operator [65], a Prewitt operator [66], Roberts cross operator [67], Gaussian Laplacian filter[5], Moment-based operator [68], etc. Next, we look more closely to two of the most used operators.

4.1. Sobel operator

Sobel operator was first introduced in 1968 by Sobel [65] and then formally accredited and described in [69]. This algorithm uses a discrete differential operator, and finds the edges calculating an approximation to the gradient of the intensity function in each pixel of an image through a 3×3 mask. That is, the gradient at the center pixel of a neighborhood is calculated by the Sobel detector [8]:

$$G_x \Rightarrow \begin{vmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{vmatrix} \qquad G_y \Rightarrow \begin{vmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{vmatrix}$$

Where the neighborhood would be represented as:

$$\begin{vmatrix} z_1 & z_2 & z_3 \\ z_4 & z_5 & z_6 \\ z_7 & z_8 & z_9 \end{vmatrix}$$

Consequently:

$$\begin{aligned} g &= \sqrt{g_x^2 + g_y^2} \\ &= \sqrt{[(z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)]^2 \ldots} \\ &\quad \ldots \overline{+[(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)]^2}. \end{aligned} \quad (7)$$

Thus, a pixel (x, y) is identified as an edge pixel if $g > T$ in this point, where T refers to a predefined threshold.

The resulting image of edge detection using Sobel operator on Figure 2 is shown in Figure 6.

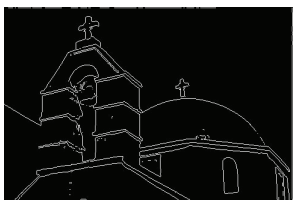

Fig. 6. Edge detection's result - Sobel.

In summary, this technique estimates the gradient of an image in a pixel by the vector sum of the four possible estimates of the single central gradients (each single central gradient estimated is a vector sum of vectors orthogonal pairs) in a 3×3 neighborhood. The vector sum operation provides an average over the directions of gradient measurement. The four gradients will have the same value if the density function is planar throughout the neighborhood, but any difference refers to deviations in the flatness of the function in the neighborhood [65].

4.2. Canny operator

This edge detection algorithm is very popular and attempts to find edges looking for a local maximum gradient in the gradient direction [70]. The gradient is calculated by the derivative of the Gaussian filter with a specified standard deviation σ to reduce noise. At each pixel the local gradient $g = \sqrt{g_x^2 + g_y^2}$ and its direction (Eq. (5)) is determined. Then, a pixel will be an edge when his strength is a local maximum in the gradient direction, and thus the algorithm classifies border pixels with 1, and those that are not in the peak gradient with 0 (Ref. 11).

This method detects both strong and weak edges (very marked), and it uses two thresholds (T_1 and T_2 such that $T_1 < T_2$). The hard edges will be those who are superior to T_2 and will be weak edges those who are between T_1 and T_2. Then, the algorithm links both types of edges but just considering as weak edges those connected to strong edges. This would ensure that those edges are truly weak edges [8].

The resulting image of edge detection using Canny operator on Figure 2 is shown in Figure 7.

Fig. 7. Edge detection's result - Canny.

Summarizing, edge detection techniques use a gradient operator, and subsequently evaluate through a predetermined threshold if an edge has been found or not [71]. Therefore, edge detection is a process by which the analysis of an image is simpli-

fied by reducing the amount of processed information, and in turn retaining valuable structural information on object edges [70]. A more detailed analysis on edge detection techniques can be obtained in [5].

5. Fuzzy edge detection

Frequently, the edges detected through an edge detection process (as those previously presented) are false edges because these classics methods are sensitive to various issues such as noise or thick edges, among others, and require a great amount of calculation, so that discretization may present problems [64,72].

Fuzzy logic has proved to be well suited to address the uncertainty characterizing the process of extracting information from an image [73]. Hence, many algorithms have included fuzzy logic in the entire process or at any particular stage of the image processing [72]. An overview of models and methods based on fuzzy sets for image processing and image understanding can be found in [74].

A first idea for applying fuzzy logic in edge detection was proposed by Pal and King [75]. Subsequently, Russo [76] designed fuzzy rules for edge detection. Different algorithms were created to address fuzzy edge detection since then.

Fuzzy edge detection detects classes of pixels corresponding to the variation in level of gray intensity in different directions. This is achieved by using a specified membership function for each class, in such a way that the class assigned to each pixel corresponds to the highest membership value [64]. Russo [76] proposed that an efficient way to process each pixel of the image must consider the set of neighboring pixels belonging to a rectangular window, called fuzzy mask.

Let K be the number of gray levels in an image, the appropriate fuzzy set to process that image are made up of membership functions defined in the universe $[0,\ldots,K-1]$ [76]. Fuzzy sets are created to represent the intensities of each variable and they are associated to the linguistic variables "black" and "white".

The resulting image of applied fuzzy edge detection on Figure 2 is shown in Figure 8.

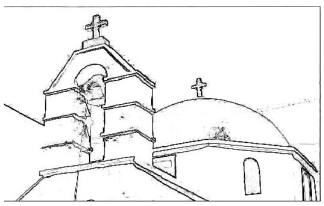

Fig. 8. Fuzzy edge detection's result.

The fuzzy rules used to fuzzy inference system for edge detection consist on "coloring a pixel *white* if it belongs to a uniform region. Otherwise, coloring the pixel *black*":

(i) IF I_x is zero and I_y is zero THEN I_{out} is white
(ii) IF I_x is not zero or I_y is not zero THEN I_{out} is black.

where I_x and I_y are the image gradient along the x-axis and y-axis and they have a zero-mean Gaussian membership function. Also, it is specified for I_{out} the triangular membership functions, white and black.

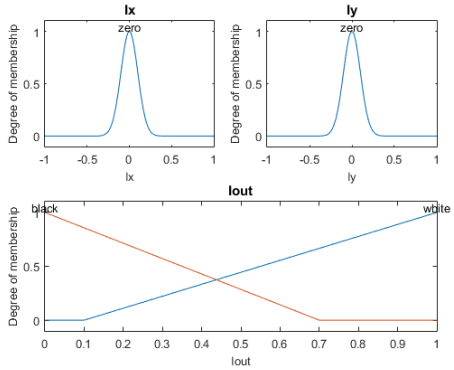

Fig. 9. Membership functions of the inputs/outputs.

As we mentioned previously, there are several versions of fuzzy edge detection algorithms in the literature. For instance, there is an algorithm that uses Epanechnikov function extended as a membership function for each class to be assigned to each pixel [64], or another algorithm which uses operators that detect specific patterns of neighboring pixels for fuzzy rules [77]. A study about representing the dataset as a fuzzy relation, associating a membership degree with each element of the relation is presented in [78]. Other issue is that fuzzy logic can be used only on a certain part of the edge detection process [72].

Fuzzy edge detection problems in which each pixel has a degree of membership to the border can

be seen in some studies [79,80,81,82,83]. Moreover, in these studies the concept of fuzzy boundary have been introduced.

6. Image segmentation

Around 1970, image segmentation boomed as an advanced research topic. In computer vision, image segmentation is a process that divides an image into multiple segments, in order to simplify or modify the representation of the image to be more meaningful and easy to analyze [84]. Actually, there are many applications of image segmentation [85], as it is useful in a robot vision [86], detection of cancerous cells [87], identification of knee cartilage and gray matter/white matter segmentation in MR images [88], among others.

The goal of segmentation is subdividing an image into regions or non-overlapping objects with similar properties. For each pixel, it is determined whether it belongs to an object or not, producing a binary image. The subdivision level is dependent on the problem to be solved. Segmentation stops when the objects of interest have been identified and isolated. However, in most cases, processed images are not trivial and hence the segmentation process becomes more complicated.

The most basic attribute for segmentation is the intensity of a monochrome image and the color components for a color image, as well as objects boundaries and texture, which are very useful. In summary, the regions of a segmented image should be uniform and homogeneous with regard to some property such as color, texture, intensity, etc [84,89]. Also, the quality of the segmentation depends largely on the output. However, although image segmentation is an essential image processing technique and has applications in many fields, there is no single method applicable to all kinds of images. Current methods have been classified according to certain characteristics of their algorithms [84,90].

Formally, image segmentation can be defined as a process of partitioning the image into a set of non-intersecting regions such that each group of connected pixels is homogeneous [85,91]. It can be defined as follows:

Image segmentation partitions an image R in C subregions $R_1, R_2, \ldots, R_C$ for a uniformity predicate P, such that the following items are satisfied:

(i) $\bigcup_{i=1}^{n} R_i = R$.
(ii) $R_i \cap R_j = \emptyset$ for all i and $j, i \neq j$.
(iii) $P(R_i) = TRUE$ for all $i = 1, 2, \ldots, C$.
(iv) $P(R_i \cup R_j) = FALSE$ for any pair of adjacent regions R_i and R_j.
(v) R_i is connected, $i = 1, 2, \ldots, C$.

The first condition states that the segmentation must be complete, i.e., the union of the segmented regions R_i must contain all pixels in the image. The second condition indicates that regions have to be disjoint, that is, that there is no overlap between them. The third condition states that pixels belonging to the same region should have similar properties. The fourth condition states that pixels from adjacent regions R_i and R_j differ in some properties. Finally, the fifth condition expresses that each region must be connected [8].

Next sections describe main methods of image segmentation. They are classified into four categories: thresholding methods (6.1), watershed methods (6.2), region segmentation methods (6.3) and graph partitioning methods (6.4).

6.1. Thresholding methods

Thresholding [92] is considered one of the simplest and most common techniques that has been widely used in image segmentation. In addition, it has served in a variety of applications such as object recognition, image analysis and interpretation of scenes. Its main features are that it reduces the complexity of the data and simplifies the process of recognition and classification of objects in the image [93].

Thresholding assumes that all pixels whose value, as its gray level, is within a certain range belong to a certain class [94]. In other words, thresholding methods separate objects from the background [7]. So, according to a threshold, either preset by the researcher or by a technique that determines its initial

value, a gray image is converted to a binary image [84]. An advantage of this procedure is that it is useful to segment images with very bright objects and with dark and homogeneous backgrounds [33,100].

Let $f(x,y)$ be a pixel of an image I with $x = 1,\ldots,n$ and $y = 1,\ldots,m$ and $B = \{a,b\}$ be a pair of binary gray levels. Then, the pixel belongs to the object if and only if its intensity value is greater than or equal to a threshold value T, and otherwise it belongs to the background [8,33]. The resulting binary image of thresholding an image function at gray level T is given by:

$$f_T(x,y) = \begin{cases} a & \text{si} \quad f(x,y) > T \\ b & \text{si} \quad f(x,y) \leqslant T \end{cases} \tag{8}$$

In this way, the main step of this technique is to select the threshold value T, but this may not be a trivial task at all [84]. However, many approaches to obtain a threshold value have been developed, as well as a number of indicators of performance evaluation [93]. For example, threshold T can remain constant for the entire image (global threshold), or this may change while it is classifying the pixels in the image (variable threshold) [33]. Hence, some thresholding algorithms lie in choosing the threshold T either automatically, or through methods that vary the threshold for the process according to the properties of local residents of the image (local or regional threshold) [8,84]. Some algorithms are: basic global thresholding [95], minimum error thresholding [96], iterative thresholding [97], entropy based thresholding [98] and Otsu thresholding [99].

According to Sezgin and Sankur [7], thresholding techniques can be divided into six groups based on the information the algorithm manipulates:

(i) Histogram shape-based techniques [100], where an analysis of the peaks, valleys and curvatures of the histogram is performed.
(ii) Clustering-based methods [101], where the gray levels are grouped into two classes, background and object, or alternatively they are modeled as a mixture of two Gaussians.
(iii) Entropy-based techniques [102], in which the algorithms use the entropy of the object and the background, the cross entropy between the original binary image, etc.
(iv) Object attribute-based techniques [103,104], which seek measures of similarity between the gray level and binary images.
(v) Spatial techniques [105], using probability distributions and/or correlations between pixels.
(vi) Local techniques [106], which adapts the threshold value at each pixel according to the local characteristics of the image.

Possibly, the most known and used thresholding techniques are those that focus on the shape of the histogram. A threshold value can be selected by inspection of the histogram of the image, where if two different modes are distinguished then a value which separates them is chosen. Then, if necessary this value is adjusted by trial and error to generate better results. However, updating this value by an automatic approach can be carried out through an iterative algorithm [8]:

(i) An initial estimate of a global threshold T is performed.
(ii) The image is segmented by the T value in two groups G_1 and G_2 of pixels, resulting from Eq. (8).
(iii) The average intensities t_1 and t_2 for the two groups of pixels are calculated.
(iv) A new threshold value is calculated: $T = \frac{1}{2}(t_1 + t_2)$.
(v) Repeat steps 2 to 4 until subsequent iterations produce a difference in thresholds lower than a given sensibility or precision (ΔT).
(vi) The image is segmented according to the last value of T.

Therefore, the basic idea is to analyze the histogram of the image, and ideally, if two dominant modes exist, there is a remarkable valley where threshold T will be in the middle [84]. However, depending on the image, histograms can be very complex (e.g., having more than two peaks or not obvious valleys), and this could lead this method to select a wrong value. Furthermore, another disadvantage is that only two classes can be generated, therefore, the method is not able to be applied in multispectral images. It also does not take into account spatial

information in the image, so it makes the method sensitive to noise [33].

The histogram of Figure 2 is shown in Figure 10. The threshold value is 170,8525.

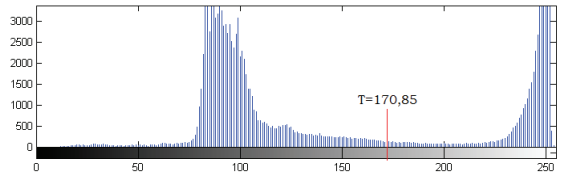

Fig. 10. Histogram.

The outcome of thresholding to the previous image is shown on Figure 11.

Fig. 11. Thresholding's result.

In general, thresholding is considered to be a simple and effective tool for supervised image segmentation. It however must achieve an ideal value T in order to segment images in objects and background. More detailed studies can be found in [7,107,108,109].

6.2. *Watershed methods*

Watershed [110] was introduced in the field of digital topography. This method considered a grayscale picture as topographic reliefs [111,112,113,114]. Later, this notion was studied in the field of image processing [115]. Watershed aims to segmenting regions in catchments, i.e., in an area where a stream of conceptual water bounded by a line trickles through the top of the mountains, which decomposes the image into rivers and valleys [116]. Light pixels can be considered the top of the mountain, and dark pixels to be in the valleys [117]. The gradient is treated as the magnitude of an image as a topographic surface [84]. Thus, it is considered that a monochrome image is a surface altitude where the pixels of greater amplitude or greater magnitude of intensity gradient correspond to the points of the river, i.e. the lines of the watershed that form the boundaries of the object. On the other hand, the pixels of lower amplitude are referred to as valley points [84,89].

By leaving a drop of conceptual water on any pixel from an altitude, this will flow to a lower elevation point until reaching a local minimum. And in turn, the pixels that drain to a common minimum, form a retention gap characterizing a region of the image. Therefore the accumulations of water in local minima neighborhood form such catchments. So, the points belonging to these basins belong to the same watershed [89]. Thus, an advantage of this method is that it first detects the leading edge and then calculates the basins of the detected gradients [84].

Overall, the watershed algorithm is as follows:

- Calculate curvature of each pixel.
- Find the local minima and assign a unique label to each of them.
- Find each flat area and classify them as minimum or valley.
- Browse the valleys and allow each drop to find a marked region.
- Allow remaining unlabeled pixels fall similarly and attach them to the labeled regions.
- Join those regions having a depth of basin below a predetermined threshold.

There are different algorithms to calculate the watershed, depending on how to extract the mountain rims from the topographic landscape. Two basic approaches can be distinguished in the following subsections.

6.2.1. *Rainfall approximation*

The rainfall algorithm [118] is based on finding the local minima of the entire image, which are assigned a unique label. Those that are adjacent are also combined with a unique tag. Then a drop of water is placed in each pixel without label. These drops will flow through neighboring lower amplitudes to a local minimum assuming the value of the label. Then local minimum pixels are shown in black and the roads of every drop to the local minimum are indicated by white [89].

6.2.2. *Flooding approximation*

The flooding algorithm [110] extracts the mountain rims by gradually flooding the landscape. The flood comes from below, through a hole in each minimum of the relief and immerse the surface in a lake entering through the holes while it fill up the various catchment basins [119]. In other words, the method involves placing a water fountain in local minima, then the valleys are submerged with water and each catchment filled until it reaches almost to overflow, which will create a reservoir in which neighbors are connected by the ridge [89,117]. This approach comes from the bottom up to find the watershed, i.e., it starts from a local minimum, and each region is flooded incrementally until it connects with its neighbors [116].

By performing the procedure of watershed, image transform complement of Figure 1, is as shown in Figure 12.

Fig. 12. Transform complement of binary image.

In Figure 13 is shown the outcome of watershed.

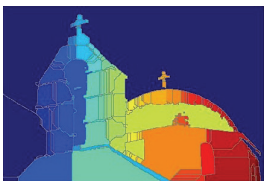

Fig. 13. Watershed's result.

In summary, watershed is a very simple and advantageous technique for real-life applications, mainly for segmenting grayscale images troubleshooting, and which uses image morphology [84,117]. Notably, because of this method finds the basins and borders at the time of use to segment images, the key lies in changing the original image in an image where the basins correspond to objects you want to identify. Also, when it is combined with other morphological tools, watershed transformation is at the basis of extremely powerful segmentation procedures [115,120].

A critical review of several definitions of the watershed transform and their algorithms, can be found in [121,122].

6.3. ***Region segmentation methods***

In this section, we present two region segmentation methods. These techniques try that neighboring pixels with similar properties are in the same region, i.e., they focus on groups of pixels that have similar intensity [71,94]. This leads to the class of algorithms known as region growning and split and merge, where the general procedure is to compare a pixel with its neighbors, and if they are homogeneous then they are assigned to the same class [94]. Region growning and split and merge are presented in section 6.3.1 and section 6.3.2 respectively.

6.3.1. *Region growning*

Region growning [123] is a method for grouping neighboring pixels in segmented regions according to predefined criteria of growth. Firstly, it specifies small sets of initial pixel called seed points. From them, the regions grow by adding neighboring pixels with similar predefined properties, until no more pixels that meet the criteria for inclusion in the regions are found and all pixels in the image have been scanned [8,33]. Similarly, if two or more regions are homogeneous, they are grouped into one. Also, once a region has no more similar pixels, then we proceed to establish a new region from another seed [124].

In summary, the basic procedure is presented bellow [33]:

- Select a seed group of pixels in the original image.
- Define true value criterion for the stopping rule.
- Combine all pairs of spatially adjacent regions fulfilling the criterion value.
- Stop the growth of the regions when it is not possible to combine more adjacent pixels.

However, a disadvantage of this method is the fact that getting a good result depends on the selection of the initial seed points and the order in which they are scanned for grouping each pixel. This dependence makes the segmented result to be sensitive to the location and ordering of seeds [71,89].

The resulting image of region growing on Figure 2 by applying a seed in the pixel $x = 148$ $y = 359$, is shown in Figure 14.

Fig. 14. Region growing's result.

Consequently, region growing is a procedure which iteratively groups pixels of an image into subregions according to a predefined criterion of homogeneity from a seed [33,124]. Similarly, region growning has a good performance because it considers the spatial distribution of the pixels [124].

6.3.2. *Split and merge*

Split and merge [71] is based on a representation of data in a quadtree where if a region of the original image is non-uniform in attribute then a square image segment is divided into four quadrants. Hence, if four neighbors are uniform, they are united in a square formed by these adjacent squares [89].

Definition 1. Let I be an image and P be a predicate, the image is subdivided into small quadrants, such that for any region R_i it is $P(R_i) = TRUE$. If $P(R) = FALSE$, the image is divided into quadrants. If P is FALSE for any quadrant, the quadrant is subdivided into sub-quadrants, and so on.

This process is represented by a tree with four leaves descendants called quadtree, where the root refers to the initial image and each of its nodes represent its four subdivisions [8].

The procedure is as follow [8,33]:

(i) Start with the complete image, where R represent the entire image region, if $P(R) = FALSE$.

(ii) The entire image R is divided into four disjoint quadrants, and if P is false for any quadrant ($P(R_i) = FALSE$), then the quadrants are subdivided into sub-quadrants until no more splits are possible.

(iii) Once no more splits can be done, any adjacent regions R_i and R_j for which $P(R_i \cup R_j) = TRUE$ are joined together.

(iv) The process ends once further connections are not possible.

Thus, split and merge arbitrarily subdivides an image into disjoint regions, and then these regions are joined or divided according to the conditions of image segmentation in Section 6. This method performs well when the colors of objects in the image are not very opposite.

The resulting image of split and merge on Figure 15 is shown in Figure 16.

Fig. 15. RGB image [9].

Fig. 16. Split and merge's result.

Generally speaking, split and merge begins with nodes (squares) at some level of the quadtree, so if a quadrant is not uniform a split is done, being subdivided into four sub-quadrants. However, if four adjacent quadrants are uniform, a merge between them is performed [71].

Besides, it is important to mention that the main advantage of region segmentation based techniques is that they attempt to segment an image into regions quickly, because the image is analyzed by regions and no by pixels.

6.4. *Graph partitioning methods*

A digital image I is modeled as a valued graph or network composed by a set of nodes ($\upsilon \in V$) and a set of arcs E ($e \in E \subseteq V \times V$) that link such nodes [125,126,127]. Graphs have been used in many segmentation and classification techniques in order to represent complex visual structures of computer vision and machine learning. In this section algorithms

which use graph are presented, but first a formal proposition is shown.

Proposition 1. *If A digital image I is viewed as a graph where the nodes are the pixels, let* $V = \{P_1, P_2, \ldots, P_n\}$ *be a finite set of pixels in the image and let* $E\{\{P_a, P_b\}|P_a, P_b \in V\}$ *be unordered pairs of neighboring pixels. Two pixels are neighbors if there is an arc* $e_{ab} = \{P_a, P_b\} \in E$. *Let* $G = (V, E)$ *be the representation of neighboring relations between the pixels in an image. Let also* $d_{ab} \geqslant 0$ *be the degree of dissimilarity between pixels* P_a *y* P_b. *Let* $D = \{d_{ab}|e_{ab} \in E\}$ *be the set of all the dissimilarities. Let* $N(I) = \{G = (V, E); D\}$ *be the network which summarizes all the previous information about an image I.*

With the above proposition, it is possible to extract meaningful information about the processed image, checking the various properties of the graphs. As we shall see throughout this section, the graphs turn out to be extremely important and useful tools for image segmentation. Some of the most popular graph algorithms for image processing are presented next.

6.4.1. Random walker

A random walker [128] approach can be used for the task of image segmentation through graphs: a user interactively label a small number of pixels called seeds, and each of the not labeled pixels releases a random walker, so the probability is calculated when the random walker reaches a labeled seed [129]. This algorithm may be useful for segmenting an image in, for example, object and background, from predefined seeds indicating image regions belonging to the objects. These seeds are labeled, and unlabeled pixels will be marked by the algorithm using the following approach: let a random walker start on an untagged pixel, what is the probability that arrives first at each seed pixel?. A $K-tuple$ vector is assigned to each pixel, which specify the probability that a walker beginnings in an unlabeled pixel, first reaches each seed point K. Then, from those of $K-tuples$ it is selected the more likely seed for the random walker [127].

The random walker algorithm consist of four steps [127]:

- Represent the image structure in a network, defining a function that assigns a change in image intensities, so assign the weights of the edges in the graph.
- Define the set of K labeled pixels to act as seed points.
- Solve the Dirichlet problem for each label except the last, which is equivalent to calculate the probability that a random walker beginning from each unlabeled pixel reaches first each of the K seed pixels.
- Select from these $K-tuples$ the more likely seed for the random walker, that is, assign each node v_i to the maximum probability label.

The algorithm above consists in generating the graph weights, establishing the system of equations to solve the problem and implement the practical details. Several properties of this algorithm are quick calculation, rapid editing, inability to produce arbitrary segmentation with sufficient interaction and intuitive segmentation [127]. Moreover, an advantage is that the edges of faint objects are found when they are part of a consistent edge [130].

The resulting image of random walker on Figure 3 is shown in Figure 17. Notice that this figure shows the two selected seeds ($K = 2$), green and blue, as well as the probability that a random walker released at each pixel reaches the foreground seeds. That is, the near the green seed are the nearest objects so their color is similar to the pixel where seed is. Similarly, blue seed is surrounded of other color and its color is more similar. The intensity of the color represents the probability that a random walker leaving the first pixel reaches each seed.

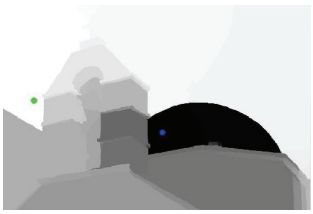

Fig. 17. Random walker's result.

Random walker contributes as a neutral technique for image segmentation because it does not

consider any information of the image. Instead, the segmentation is derived from the $K-tuples$ of each pixel, selecting the most probable seed for each pixel or random walker [130].

6.4.2. Graph cuts

Graph cuts [131] is a multidimensional optimization tool that is capable of applying smoothing in pieces, while maintaining marked discontinuities [132]. It is supported in both graph theory and the problem of minimizing the energy function, uses the maximum flow problem within graph. Thus, through the max-flow min-cut theorem, it is defined the minimum cut of the graph, such that the cut size is not larger in any other cut [133].

A s-t cut is a subset of arcs $C \in E$ such that the S and T terminals are completely separated in the induced graph $G(C) = (V, E \setminus C)$ [134].

Definition 2. A *s-t cut* or simple cut of a graph $G = (V, E)$ is a binary partition of nodes V into two subsets with a primary vertex in each: source S and sink $T = V - S$ such that $s \in S$ and $t \in T$ [132,135].

Thus, each graph has a set of edges E with two types of terminal nodes P: $n-links$, referring to non-terminal neighboring arcs; and $t-links$, that are used to connect terminal pixels to non-terminal pixels. Each graph edge is assigned some nonnegative weight or cost $w(p,q)$, so a cost of a directed edge (p,q) may differ from the cost of the reverse edge (q,p). The set of all graph edges consist of $n-links$ in N and $t-links$ $\{(s,p),(p,t)\}$ for non-terminal nodes $p \in P$ [132,134].

The minimum cut problem is based on finding the cut with the minimum cost among all cuts. The cost of a cut $C = (S,T)$ is the sum of the $costs/weights$ of the arcs borders (p,q) such that $p \in S$ and $q \in T$. Thus, the min-cut and max-flow problems are equivalent, because the maximum flow value is equal to the cost of the minimum cut [132,136,137].

If f is a flow in a graph G with two terminal nodes called source s and sink t, respectively representing "object" and "background" labels, and a set of non-terminals P nodes, then the value of the maximum flow is equal to the capacity of a minimum cut C [132,134,135].

The basic algorithm for this procedure developed by Boykov and Kolmogorov [133] based on increasing paths through two search trees for the terminal nodes, consists of three steps that are iteratively repeated:

- Growth phase: search trees S and T grow until they touch giving an $s \to t$ path.
- Augmentation phase: the found path is augmented, search tree(s) break into forest(s).
- Adoption phase: trees S and T are restored.

Other max-flow min-cut algorithms are based on the push-relabel methods, which are based on two basic operations: push flow and relabel from an overflowing node [135].

In general, the cut may generate a binary segmentation with arbitrary topological properties [134]. For example, if you work with the observed intensity $I(p)$ of a pixel p, a cut would symbolize a binary labeling process f, which assigns labels $f_p \in \{0,1\}$ to the pixels in such a way that if $p \in S$ then $f_p = 0$ and if $p \in T$ then $f_p = 1$ [132].

One advantage of this method is that it allows a geometric interpretation, and also works with a powerful tool for the energy minimization (a binary optimization method). There are a lot of techniques based on graph cuts that produce good approximations. This technique is used in many applications as an optimization method for low vision problems, based on global energy formulations. In addition, graph cuts can be very efficient computationally, and this is why they provide a clean, flexible formulation of image segmentation [132].

Frequently, this method is combined with others, however, an improved graph cut method named normalized cuts [138] is based on modifying the cost function to eliminate outliers. The usage of this technique within image processing context is presented in the following section.

6.4.3. Normalized cuts

This procedure works with weighted graphs $G = (V,E)$ where the weight $w(i,j)$ of each arc is a function of the similarity between nodes i and j [139].

A graph $G = (V,E)$ is optimally divided into two disjoint subsets A and B with $A \cup B = V$ and $A \cap B = \emptyset$ when *Ncut* value is minimized. The degree of dissimilarity of A and B is measured as a fraction of the total weight of the connections arcs on all nodes in the network. This is called normalized cut (Ncut) [139]:

$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)}. \quad (9)$$

where $assoc(A,V) = \sum_{u \in A, t \in V} w(u,t)$ is the total cost of connecting the nodes in A with all nodes in the graph, and similarly with $assoc(B,V)$.

The total normalized association measure within a group for a given partition is expressed as [139]:

$$Ncut(A,B) = \frac{assoc(A,A)}{assoc(A,V)} + \frac{assoc(B,B)}{assoc(B,V)}. \quad (10)$$

where $assoc(A,A)$ y $assoc(B,B)$ refers to the total weight of the arcs connecting the nodes within A an B respectively.

Given a partition of nodes of graph V into two disjoint complementary sets A and B, let x be an $N = |V|$ dimensional indication vector, $x_i = 1$ if node i is in A and $x_i = -1$ otherwise. Also, let $d_i = \sum_j (W(i,j))$ be the total connection weight from node i to all other nodes [138]. Therefore, Eq. (6.4.3) can be rewritten as:

$$Ncut(A,B) = \frac{\sum_{(x_i>0, x_j<0)} -w_{ij} x_i x_j}{\sum_{x_i>0} d_i} \ldots + \frac{\sum_{(x_i<0, x_j>0)} -w_{ij} x_i x_j}{\sum_{x_i<0} d_i}. \quad (11)$$

The algorithm can be summarized in the following steps [138,139]:

- Given an image, represent it in a weighted graph $G = (V,E)$, and summarize the information into W and D (let $D = diag(d_1, d_2, \ldots, d_N$ be an $N \times N$ diagonal matrix and W be an $N \times N$ symmetric matrix with $W(i,j) = w_{ij}$).
- Solve $(D - W)_x = \lambda D_x$ for the eigenvectors with the smallest eigenvalues.
- Use the eigenvector with the second smallest eigenvalue to subdivide the graph by finding the splitting points to minimize *Ncut*.
- Check whether it is necessary to split the current partition recursively by checking the stability of the cut.
- Stop if for any segment *Ncut* exceeds a specified value.

The resulting image of gray-scale image segmentation using normalized graph cuts on Figure 2 is shown in Figure 18.

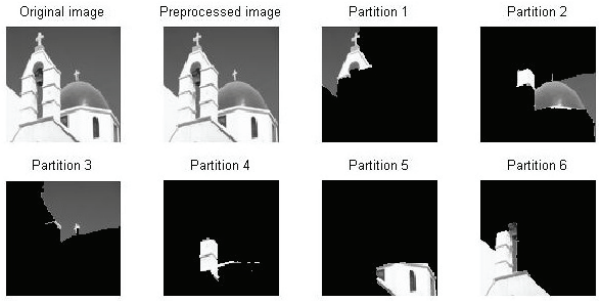

Fig. 18. Normalized cuts's result.

Thus, the normalized cuts method originated because if we only consider to minimizing the cut value $cut(A,B) = \sum_{u \in A, v \in B} w(u,v)$, a problem arise that the sets of small cuts are favored if isolated pixels appears [138].

6.4.4. Region adjacency graphs

Region adjacency graphs (RGA) [71] provides a spatial view of the image, in order to efficiently manipulate the information contained in it, through a graph which associates a region and the arcs linking each pair of adjacent regions with each vertex [124].

A region R_i defined by a quadruple $R_i = (k, x_k, y_k, A_k) | k \in (1, \ldots, n)$, where k refers to the index number of the region, (x_k, y_k) represents the center of gravity and A_k represents the area.

A RAG is defined by a set of regions and a set of pairs of regions: $(\{R_i\}, \{(R_j, R_k)\} \mid i,j,k \in (1, \ldots, n))$. Thus, the set of parts described all regions involved. And the set of pairs of regions indicate regions which are adjacent [140].

As we mentioned above, nodes represent the regions and are compared in terms of a formula-

tion based on chains. Thus, a region can be perceived as a shape whose boundary is represented by a cyclic chain (boundary string), which consists of a sequence of simple graph arcs. According to a given algorithm, the similarities between two border chains are calculated [126].

Two important characteristics of the RAG raised by Pavlidis [71] are:

(i) The degree of a node is the number of adjacent regions to a given region, which usually is proportional to the size of the region.
(ii) If a region completely surrounds other regions, that node is a cutnode in RAG, as shown in the following figure.

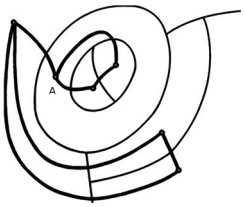

Fig. 19. Example of region adjacency graph [71].

Despite being a very useful and robust tool, this method has the disadvantage that it does not produce good results when the processed image has not perfectly defined regions or discontinuities [141].

The resulting image of RAG on Figure 2 is shown in Figure 20. The algorithm uses watershed algorithm to find structure of the regions in the image. Then, two regions are considered as neighbors if they are separated by a small number of pixels in horizontal or vertical direction.

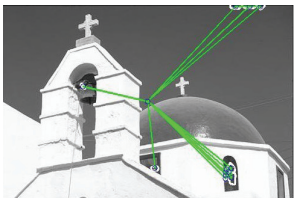

Fig. 20. Region adjacency graph's result.

Accordingly, this technique is used to provide an efficient system for manipulating implicit information in an image through a spatial view. So each node in the RAG represents a region, and two nodes (regions) are linked if they are neighbors [142].

Summarizing, graph theory has the advantage of organizing the image elements into mathematically sound structures making the formulation of the problem more understandably and enabling less complex computation [143].

The image segmentation techniques presented in this section have been categorized into four classes: thresholding methods, watershed methods, region segmentation methods and graph partitioning methods. In addition, some evaluation methods are created to evaluate quality of an image segmented. A review of these evaluation methods for image segmentation is presented in [144,145].

7. Hierarchical image segmentation

Hierarchy theory [146] has been frequently used in sociology and anthropology. These sciences study a hierarchical set of social classes or strata [147]. In this section, we present the definition of hierarchical segmentation methods, which try to group individuals according to criteria starting from one group including all individuals until creating n groups each containing a unique individual. This can be done in two ways, by agglomerative methods [148] or by divisive methods [149], where the groups are represented by a dendrogram based on a two-dimensional diagram as a hierarchical clustering [4].

The agglomerative methods are responsible for successively merge the n individuals in groups. On the other hand, the divisive methods choose to separate the n individuals into progressively smaller groupings.

First, we begin by presenting the elements needed to obtain a hierarchical image segmentation using graphs.

According to Gómez *et al.* [150,151] given an image I and its network $N(I)$, a family $\boldsymbol{S} = (S^0, S^1, \ldots, S^K)$ of segmentations of $N(I)$ constitute a hierarchical image segmentation of $N(I)$ when the following properties are verified:

- $S^t \in S^n(N(I))$ for all $t \in \{0, 1, \ldots, K\}$, (i.e., each S^t is an image segmentation of $N(I)$).
- There are two trivial partitions, the first $S^0 = \{\{v\}, v \in V\}$ containing groups with one node

(singleton), and another partition $S^K = \{V\}$ containing all pixels in the same cluster.

- $|S^t| > |S^{t+1}|$ for all $t = 0, 1, \ldots, K-1$, indicating that in each iteration the number of groups or communities increases.
- $S^t \widetilde{\subseteq} S^{t+1}$ for all $t = 0, 1, \ldots, K-1$, means that S^t is finer that $S^t - 1$ (i.e., given two partitions P and Q of the set of pixels V of a network I, it is said that P is finer than Q if for all $A \in P$ exists $B \in Q$ such that $A \subseteq B$).

One advantage of hierarchical segmentation is that it can produce an ordering of the segmentations, which may be informative for picture display. So in any hierarchical image segmentation proceedings, no apriori information about the number of clusters is required. A dendrogram helps in making the selection of the number of groups on the same levels. The partitions are achieved through the selection of one of the solutions in the nested sequence of clusters making up the hierarchy. The best partition is at a certain level, where the groupings below this height are far apart.

Another advantage of hierarchical segmentation is that it can be used to build fuzzy boundaries by a hierarchical segmentation algorithm designed by the authors [150,152,153,154,155,156].

There are some algorithms of hierarchical image segmentation, as [158,159,160,161,162] among others. In the next subsection, we present an algorithm of hierarchical image segmentation based on graphs.

7.1. Divide & link algorithm

The divide-and-link (D&L) algorithm proposed by Gómez *et al.* [155] and extended in [156] is a binary iterative unsupervised method that obtains a hierarchical partition of a network, to be shown in a dendrogram, and it is framed as a process to partition networks through graphs.

Generally speaking, D&L tries to group a set of pixels V of a graph $G = (V, E)$ through an iterative procedure that classifies nodes in V in two classes V_0 and V_1. However, the rating depends on the type of edge. There are endpoints of division edges that are assigned in different classes (split nodes) and there are endpoints of link edges that are assigned in the same class (link nodes).

The D&L algorithm consists on the following main steps[150]:

(i) Calculate the division and link weights for each edge.
(ii) Organize subgraph edges G^t sequentially according to the weights of the previous step.
(iii) Build a spanning forest $F^t \subset G^t$ (through, for instance, a Kruskal-like algorithm [163]) based on the arrangement found in the previous step.
(iv) Build a partition P^t following the binary procedure applied to F^t.
(v) Define a new set of edges E^{t+1} from E^t by removing those edges that connect different groups of P^t.
(vi) Repeat steps *i-v* while $E^{t+1} \neq \emptyset$.

The output of the D&L algorithm (with six partitions) applied to Figure 3 is shown in Figure 21.

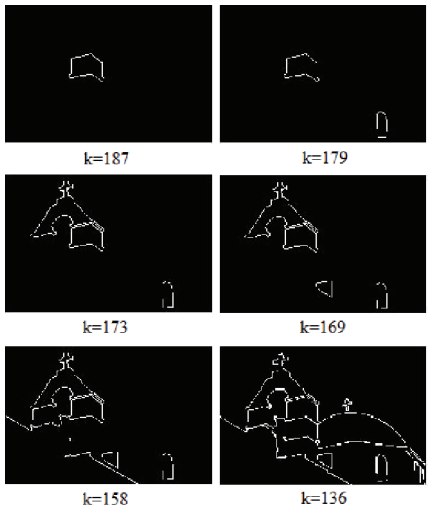

Fig. 21. D&L's result.

In summary, any hierarchical image segmentation algorithm is a divisive process because it starts with a trivial partition, formed by a single group containing all nodes, and ends with another trivial partition formed by as many groups as nodes possesses the finite set of elements you wish to group. It also has a polynomial time complexity and a low computational cost. One advantage is that it can be used in a variety of applications.

8. Fuzzy image segmentation

Fuzzy image segmentation concept was introduced in [151,156] and extended in [150,157], where they established that a fuzzy image segmentation should be a set of fuzzy regions $R_1, \ldots, R_k$ of the image. They presented a way to define this concept based on the fact that crisp image segmentation can be characterized in terms of the set of edges that separates the adjacent regions of the segmentation.

Definition 3. Given an image modeled as a network image $N(I) = \{ G = (V,E); D \}$, a subset $B \subset E$ characterizes an image segmentation if and only if the number of connected components of the partial graph generated by the edges $E - B$, denoted as $G(E-B) = (V, E-B)$, decreases when any edge of B is deleted.

In this sense, a formal definition of fuzzy image segmentation is through the fuzzyfication of the edge-based segmentation concept introduced in Definition 3 of this section (see [150] for more details).

Definition 4. Given a network image $N(I) = \{ G = (V,E); D \}$, we will say that the fuzzy set $\widetilde{B} = \{(e, \mu_B(e)),\ e \in E\}$ produces a fuzzy image segmentation if and only for all $\alpha \in [0,1]$ the crisp set $B(\alpha) = \{e \in E \ : \ \mu_B(e) \geqslant \alpha\}$ produces an image segmentation in the sense of Definition 3.

In the previous definition, the membership function of the fuzzy set $\widetilde{B}$ for a given edge represents the degree of separation between these two adjacent pixels in the segmentation process.

In [157], it has proven that there exist a relation between fuzzy image segmentation concept and the concept of hierarchical image segmentation. In that paper, it is showed that it is possible to build a fuzzy image segmentation from a hierarchical image segmentation in the following way.

Given a network image $N(I) = \{G = (V,E); D\}$, let $\mathscr{B} = \{B^0 = E, B^1, \ldots, B^K = \emptyset\}$ be a hierarchical image segmentation, and for all $t \in \{0, 1, \ldots, K\}$ let $\mu^t : E \longrightarrow \{0,1\}$ be the membership function associated to the boundary set $B^t \subset E$. Then, the fuzzy set $\widetilde{B}$ defined as:

$$\mu_B(e) = \sum_{t=0}^{K} w_t \mu^t(e) \qquad \forall e \in E. \tag{12}$$

induces a fuzzy image segmentation of $N(I)$ for any sequence $w = (w_0, w_1, \ldots, w_K)$ such that:

$$w_t \geqslant 0 \quad \forall t \in \{0, 1, \ldots, K\} \qquad \sum_{t=0}^{K} w_t = 1. \tag{13}$$

The output of fuzzy image segmentation applied to Figure 3 is shown in Figure 22. In this picture has been aggregated the output of hierarchical image segmentation of Section 7.1, in order to build a fuzzy image segmentation [157].

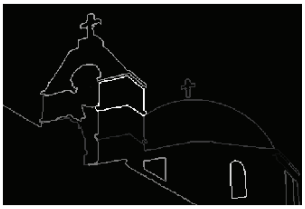

Fig. 22. Fuzzy image segmentation's result.

9. Final remarks

The main characteristics and relationships between some of the best known techniques of digital image processing have been analyzed in this paper. These techniques have been classified according to the output which they generate, and attending to the problems they face. Also, this classification depends on whether they are supervised or unsupervised methods, and whether they are crisp or fuzzy techniques. In this way we can notice that different problems should be formally distinguished when trying to recognize an object in a digital image (such as classification, edge detection, segmentation and hierarchical segmentation). For example, it has been pointed out that the output of an edge detection problem does not necessarily produce a suitable image segmentation. Edge detection only detects the border of the objects and does not necessarily separate objects in the image.

Furthermore, some of the techniques reviewed in this paper can be used as a previous step for another technique. For example, an edge detection output could be the input for an image segmentation if we find a partition of the set of pixels into connected regions. The opposite is also true, and it is possible to find the boundary of a set of objects through the

set of pixels that connect pixels of different regions. Also, in an output of image segmentation, if two different and not adjacent subsets of pixel can share the same characteristics, we can apply an image classification technique and then these regions will belong to the same class. Moreover, a solution of clustering can be a previous step to reach an image segmentation. Thus, although different, these techniques can be related to each other.

Computational intelligence is a must in a wide range of challenging real-world image processing problems. More sophisticated original approaches, variations and combinations should be expected depending on the specific characteristics of each image and the decision-maker objectives (see, for example [164]). In particular, one of the most important applications of computational intelligent techniques is the imaging process which is used in many fields as for example, digital imaging [165], medical imaging [166], industrial tomography [167], chemical imaging [168] and thermography [169].

We would like to remark that there were found no significative differences in computational time when processing all the images considered in this paper with different algorithms, being in general quite short.

Finally, it is important to point out that in the background of an edge detection process, there is a classification problem, because a pixel is classified as either edge or not edge. Similarly, a thresholding method can be a classification problem because a pixel is classified as either object or background. In this manner, the classification of the methods shown in this paper can be extended even more generally, a problem to be considered in a future research.

Acknowledgments

This research has been partially supported by the Government of Spain, grant TIN2015-66471-P, and by the Government of the Community of Madrid, grant S2013/ICE-2845 (CASI-CAM-CM).

References

1. J. Martin and D. Jurafsky, "Speech and language processing," *International Edition*, (2000).
2. S. Thrun, "Learning metric-topological maps for indoor mobile robot navigation," *Artificial Intelligence*, **99**, 1:21–71, (1998).
3. E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," *Computer Vision-ECCV. Springer Berlin Heidelberg*, 430–443, (2006).
4. J. Richards and X. Jia, "Remote Sensing Digital Image Analysis. An Introduction," *Springer*, (1999).
5. D. Marr and E. Hildreth, "Theory of Edge Detection," *Proc. Roy. Soc. London. Series B, Biological Sciences*, **207**, 1167:187–217, (1980).
6. A. Jain, M. Murty and P. Flynn, "Data Clustering: A Review," *ACM Computing Surveys*, **31**, 3:264–322, (1999).
7. M. Sezgin and B. Sankur, "Survey over image thresholding techniques and quantitative performance evaluation," *Electronic Imaging*, **13**, 1:146–165, (2004).
8. R. González, R. Woods and S. Eddins, "Digital Image Processing using MATLAB," *Chapman and Hall Computing*, (2009).
9. D. Martin, C. Fowlkes, D. Tal and J. Malik, "A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics," *Proc. 8th Intl. Conf. Computer Vision*, **2**, 416–423, (2001).
10. J. Bernd, "Digital Image Processing," *Springer*, (2002).
11. W. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, **5**, 4:115–133, (1943).
12. A. Turing, "Computing machinery and intelligence," *Mind*, **49**, 236:433–460, (1950).
13. C. Cortes and V. Vapnik, "Support-Vector Networks," *Machine Learning*, **20**, 273–297, (1995).
14. J. Pearl, "Reverend Bayes on Inference Engines: A Distributed Hierarchical Approach," *Association for the Advancement of Artificial Intelligence*, **AAAI-82**, 133–136, (1982).
15. A. Scott and M. Symons, "Clustering methods based on likelihood ratio criteria," *Biometrics*, **27**, 2:387–397, (1971).
16. A. Wacker and D. Landgrebe, "Minimum Distance Classification in Remote Sensing," *Proc. of the First Canadian Symposium on Remote Sensing*, **2**, 577–599, (1972).
17. I. Aizenberg, N. Aizenberg, J. Hiltner, C. Moraga and E. Meyer zu Bexten, "Cellular neural networks and computational intelligence in medical image processing," *Image and Vision Computing*, **19**, 4:177–183, (2001).
18. J. Jiang, P. Trundle and R. Ren, "Medical image analysis with artificial neural networks," *Computerized Medical Imaging and Graphics*, **34**, 8:617–631, (2010).
19. S. Manikandan, K. Ramar, M. Willjuice Iruthayara-

jan and K. Srinivasagan, "Multilevel thresholding for segmentation of medical brain images using real coded genetic algorithm," *Measurement*, **47**, 0:558–568, (2014).
20. H. Yang, H. Wang, Q. Wang and X. Zhang, "LS-SVM based image segmentation using color and texture information," *JVCIR*, **23**, 7:1095–1112, (2012).
21. E. Alpaydin, "Introduction to Machine Learning," *The MIT Press*, (2010).
22. N. Nilsson, "Introduction to Machine Learning," *Ebook*, (1996).
23. H. Driver and A. Kroeber, "Quantitative expression of cultural relationships," *University of California Publications in American Archeology and Ethnology*, **31**, 211–256, (1932).
24. A. Rencher, "Methods of Multivariate Analysis," *Wiley Interscience*, (2002).
25. W. Hardle and L. Simar, "Applied Multivariate Statistical Analysis," *Springer*, (2003).
26. R. Xu and D. Wunsch II, "Survey of Clustering Algorithms," *IEEE Transactions on Neural Networks*, **16**, 3:645–678, (2005).
27. J. MacQueen, "Some methods for classification and analysis of multivariate observations," *Proc. of 5th Berkeley Symposium on Mathematical Statistics and Probability 1. University of California Press*, **31**, 281–297, (1967).
28. S. Lloyd, "Least square quantization in PCM," *Bell Telephone Laboratories Paper*, (1957).
29. S. Lloyd, "Least square quantization in PCM," *IEEE Transactions on Information Theory*, **28**, 2:129–137, (1982).
30. T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, R. Silverman and A. Wum, "An Efficient k-Means Clustering Algorithm: Analysis and Implementation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **24**, 7:881–892, (2002).
31. J. Wang, "Encyclopedia of Data Warehousing and Mining," *Idea Group Reference*, (2006).
32. A. Wacker. "A cluster approach to finding spatial boundaries in multispectral imagery," *Laboratory for Applications of Remote Sensing, Purdue University* **LARS Info. Note 122969**, (1969).
33. R. Dass, Priyanka and S. Devi, "Image Segmentation Techniques," *IJECT*, **3**, 1:66–70, (2012).
34. G. Toussaint, "The use of context in Pattern Recognition," *Pattern Recognition*, **10**, 3:189–204, (1978).
35. J. Lu and Q. Weng, "A survey of image classification methods and techniques for improving classification performance," *IJRSA*, **28**, 5:823–870, (2007).
36. L. Zadeh, "Fuzzy Sets," *Information and Control*, **8**, 338–353, (1965).
37. Z. Chi, H. Yan and T. Pham, "Fuzzy Algorithms: With Applications to Image Processing and Pattern Recognition," *World Scientific*, **10**, (1996).
38. V. Nokák, I. Perfilieva and J. Mockor, "Mathematical principles of fuzzy logic," *Springer Science & Business Media*, **517**, (2012).
39. V. Novák, I. Perfilieva and J. Mockor, "Mathematical Principles of Fuzzy Logic," *Boston: Kluwer Academic Publishers*, (1999).
40. D. Dubois, W. Ostasiewicz and H. Prade, "Fuzzy sets: history and basic notions," *Fundamentals of fuzzy sets. Springer*, 21–124, (2000).
41. D. Dubois, H. Prade and H. Prade, "Fundamentals of fuzzy sets," *Springer & Business Media*, **7**, (2000).
42. G. Klir and B. Yuan, "Fuzzy sets and fuzzy logic," *New Jersey: Prentice Hall.*, **4**, (1995).
43. E. Kerre, "Introduction to the Basic Principles of Fuzzy Set Theory and some of its Applications, second revised edition," *Gent: Communication and Cognition.*, (1993).
44. A. Amo, J. Montero, G. Biging and V. Cutello, "Fuzzy classification systems," *EJOR*, **156**, 495–507, (2004).
45. L. Zadeh, "A rationale for fuzzy control," *J. Dynamic Syst. Meas.Control*, **94**, 3–4, (1972).
46. F. Herrera, M. Lozano and J.L. Verdegay, "A learning process for fuzzy control rules using genetic algorithms," *Fuzzy Sets and Systems*, **100**, 143–158, (1998).
47. M. Galar, J. Fernandez, G. Beliakov and H. Bustince, "Interval-valued fuzzy sets applied to stereo matching of color images," *IEEE Trans. on Image Processing*, **20**, 7:1949–1961, (2011).
48. E. Mamdani, "Application of fuzzy algorithms for control of simple dynamic plant," *Proc. of the Institution of Elec. Engineers*, **121**, 12:1585–1588, (1974).
49. D. Ruan and E. Kerre, "Fuzzy IF-THEN Rules in Computational Intelligence: Theory and Applications," *Boston: Kluwer Academic Publishers*, (2000).
50. I. Perfilieva, "Logical foundations of rule-based systems," *Fuzzy Sets and Systems*, **157**, 5:615–621, (2006).
51. D. Dubois and H. Prade, "What are fuzzy rules and how to use them," *Fuzzy sets and systems*, **84**, 2:169–185, (1996).
52. W. van Leekwijck and E. Kerre, "Defuzzification: criteria and classification," *Fuzzy Sets and Systems*, **108**, 2:159–178, (1999).
53. O. Cordón, M. del Jesús and F. Herrera "A proposal on reasoning methods in fuzzy rule-based classification systems," *Intl. Journal of Approximate Reasoning*, **20**, 21–45, (1999).
54. J. Dunn, "A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters," *Journal of Cybernetics*, **3**, 3:32–57, (1973).
55. J. Bezdek, "Pattern Recognition with Fuzzy Objective Function Algorithms," *Plenum Press*, (1981).
56. J. Bezdek, R. Ehrlich and W. Full, "FCM: The fuzzy c-means clustering algorithm," *Computers & Geo-*

sciences, **10**, 2-3:191–203, (1984).
57. J. Yang, S. Hao and P. Chung, "Color image segmentation using fuzzy c-means and eigenspace projections," *Signal Processing*, **82**, 461–472, (2002).
58. D. Zhong and H. Yan, "Color image segmentation using color space analysis and fuzzy clustering," *Proceedings of the 2000 IEEE Signal Processing Society Workshop*, **2**, 624–633, (2000).
59. M. Nachtegael, D. Van der Weken, E. Kerre and W. Philips, "Soft Computing in Image Processing," *Studies in Fuzziness and Soft Computing. Springer*, **210**, (2007).
60. E. Kerre and M. Nachtegael, "Fuzzy techniques in image processing," *Physica-Verlag Springer*, **52**, (2000).
61. E. Ruspini, "A new approach to clustering," *Information and Control*, **15**, 22–32, (1969).
62. M. Sonka, V. Sonka and R. Boyle, "Image Processing, Analysis and Machine Vision," *Chapman and Hall Computing*, (1993).
63. D. Phillips, "Image Processing in C," *R & D Publications*, (2000).
64. L. Liang and C. Looney, "Competitive fuzzy edge detection," *Applied Soft Computing*, **3**, 123–137, (2003).
65. I. Sobel, "History and Definition of the Sobel Operator," (2014).
66. J. Prewitt, "Object Enhancement and Extraction," *Picture processing and Psychopictorics, Academic Press*, (1970).
67. L. Roberts, "Machine perception of three-dimensional solids," *Optical and Electro-optical Information Processing*, 159–197, (1965).
68. A. Reeves, "A moment based two-dimensional edge operator," *Proc. IEEE Comput. Vision & Patt. Recogn*, 312–317, (1983).
69. R. Duda and P. Hart, "Pattern Classification and Scene Analysis," *John Wiley and Sons*, 271–272, (1973).
70. J. Canny, "A Computational Approach To Edge Detection," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **PAMI-8**, 6:679–698, (1986).
71. T. Pavlidis, "A graphics and image processing," *Springer*, (1982).
72. C. López-Molina, "The breakdown structure of edge detection: Analysis of individual components and revisit of the overall structure," *Ph.D Thesis, Navarra Public University*, (2012).
73. F. Russo, "Edge detection in noisy images using fuzzy reasoning," *Instrumentation and Measurement Technology Conference*, **1**, 369–372, (1998).
74. I. Bloch, "Fuzzy sets for image processing and understanding," *Fuzzy Sets and Systems*, doi:10.1016/j.fss.2015.06.017, (2015).
75. S. Pal and R. King, "On Edge Detection of X-Ray Images Using Fuzzy Sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **5**, 1:69–77, (1983).
76. F. Russo, "A user-friendly research tool for image processing with fuzzy rules," *Proc. of the First IEEE International Conference on Fuzzy Systems*, 561–568, (1992).
77. F. Russo and G. Ramponi, "Edge extraction by FIRE operators," *IEEE Conf. on Fuzzy Systems*, **1**, 249–253, (1994).
78. H. Bustince, E. Barrenechea, J. Fernández, M. Pagola, J. Montero and C. Guerra, "Contrast of a fuzzy relation," *Information Sciences*, **180**, 8:1326–1344, (2010).
79. H. Bustince, E. Barrenechea, M. Pagola and J. Fernández, "Interval-valued fuzzy sets constructed from matrices: Application to edge detection," *Fuzzy Sets and Systems*, **160**, 13:1819–1840, (2009).
80. E. Barrenechea, H. Bustince and B. De Baets, "Construction of interval-valued fuzzy relations with application to the generation of fuzzy edge images," *IEEE Trans. On Fuzzy Systems*, **19**, 819–830, (2011).
81. C. López-Molina, B. De Baets, H. Bustince, J. Sanz and E. Barrenechea, "Multiscale edge detection based on gaussian smoothing and edge tracking," *Knowledge-Based Systems*, **44**, 101–111, (2013).
82. H. Bustince, V. Mohedano, E. Barrenechea and M. Pagola, "Definition and construction of fuzzy DI-subsethood measures," *Information Sciences*, **176**, 21:3190–3231, (2006).
83. N. Senthilkumaran and R. Rajesh, "Edge detection techniques for image segmentation a survey of soft computing approaches," *International Journal of Recent Trends in Engineering*, **1**, 2:250–254, (2009).
84. B. Basavaprasad and H. Ravindra, "A Survey on Traditional and Graph Theoretical Techniques for Image Segmentation," *IJCA Proc. on Nat. Conf. on Recent Advances in Information Technology*, **NCRAIT**, 1:38–46, (2014).
85. N. Pal and S. Pal, "A review on image segmentation techniques," *Pattern Recognition*, **26**, 1277–1294, (1993).
86. J. Bruce, T. Balch and M. Veloso, "Fast and inexpensive color image segmentation for interactive robots," *International Conference on Intelligent Robots and Systems (IROS 2000). Proceedings*, **3**, 2061–2066, (2000).
87. Y. Al-Kofahi, W. Lassoued, W. Lee and B. Roysam, "Improved automatic detection and segmentation of cell nuclei in histopathology images," *IEEE Trans. on Biomedical Engineering*, **57**, 4:841–852, (2010).
88. V. Grau, A. Mewes, M. Alcaniz, R.Kikinis and S. Warfield, "Improved watershed transform for medical image segmentation using prior information," *IEEE Trans. on Medical Imaging*, **23**, 4:447–458, (2004).
89. W. Pratt, "Digital Image Processing," *Wiley - Interscience*, (2001).
90. Y. Ma, K. Zhan and Z. Wang, "Applications of Pulse-

Coupled Neural Networks," *Springer*, (2010).
91. K. Fu and J. Mui, "A survey on image segmentation," *Pattern Recognition*, **13**, 3–16 (1981).
92. J. Weszka, R. Nagel and A. Rosenfeld, "A Technique for Facilitating Threshold Selection for Object Extraction from Digital Pictures," *University of Maryland Computer Science Center*, **TR-243**, (1973).
93. T. Wu, "Image data field-based framework for image thresholding," *Optics & Laser Technology*, **62**, 1–11, (2014).
94. R. Adams and L. Bischof, "Seeded Region Growing," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **16**, 6:641–647, (1994).
95. S. Lee and S. Chung, "A comparative study of several global thresholding techniques for segmentation," *Comput. Vision. Graphics Image Process*, **52**, 171–190, (1990).
96. J. Kittler and J. Illingworth, "Minimum error thresholding," *Pattern Recognition*, **19**, 1:41–47, (1986).
97. I. Daubechies, M. Defrise and C. Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Comm. Pure Appl. Math*, **57**, 1413–1457, (2004).
98. T. Pun, "A new method for grey-level picture thresholding using the entropy of the histogram," *Signal Processing*, **2**, 3:223–237, (1980).
99. N. Otsu, "A threshold selection method from gray level histograms," *IEEE Trans. Systems Man Cybernet*, **9**, 62–66, (1979).
100. J. Weszka and A. Rosenfeld, "Threshold Evaluation Techniques," *IEEE Trans. on Systems, Man and Cybernetics*, **SMC-8**, 8:622–629, (1978).
101. T. Ridler and S. Calvard, "Picture Thresholding Using an Iterative Selection Method," *IEEE Trans. on Systems, Man and Cybernetics*, **SMC-8**, 8:630–632, (1978).
102. J. Kapur, P. Sahoo and A. Wong, "A New Method for Gray-Level Picture Thresholding Using the Entropy of the Histogram," *Computer Vision, Graphics and Image Processing*, **29**, 273–285, (1985).
103. J. Russ and C. Russ, "Automatic discrimination of features in gray-scale images," *Journal of Microscopy*, **148**, 3:263–277, (1987).
104. L. Hertz and R. Schafer, "Multilevel thresholding using edge matching," *Computer Vision, Graphics, and Image Processing*, **44**, 3:279–295, (1988).
105. R. Kirby and A. Rosenfeld, "A Note on the Use of (Gray Level, Local Average Gray Level) Space as an Aid in Threshold Selection," *IEEE Trans. on Systems, Man and Cybernetics*, **SMC-9**, 12:860–864, (1979).
106. Y. Nakagawa and A. Rosenfeld, "Some experiments on variable thresholding," *Pattern Recognition*, **11**, 3:191–204, (1979).
107. P. Sahoo, S. Soltani, A. Wong and Y. Chen, "A Survey of Thresholding Techniques," *Comput. Vision, Graphics, and Image Processing*, **41**, 233–260, (1988).
108. C. Glasbey, "An Analysis of Histogram-Based Thresholding Algorithms," *CVGIP: Graphical Models and Image Processing*, **55**, 6:532–537, (1993).
109. H. Bustince, M. Pagola, E. Barrenechea, J. Fernndez, P. Melo-Pinto, P. Couto, H. Tizhoosh and J. Montero, "Ignorance functions. An application to the calculation of the threshold in prostate ultrasound images," *Fuzzy sets and Systems*, **161**, 1:20–36, (2010).
110. S. Beucher and C. Lantujoul, "Use of watersheds in contour detection," *Intl. workshop on image processing, real-time edge and motion detection*, (1979).
111. S. Collins, "Terrain parameters directly from a digital terrain model," *Canadian Surveyor*, **29**, 5:507–518, (1975).
112. T. Puecker, and D. Douglas, "Detection of surface-specific points by local parallel processing of discrete terrain elevation data," *Comput. Vision, Graphics, Image Processing*, **4**, 375–387, (1975).
113. D. Marks, J. Dozier and J. Frew, "Automated basin delineation from digital elevation data," *Geoprocessing*, **2**, 299–311, (1984).
114. L. Band, "Topographic partition of watersheds with digital elevation models," *Water Resources Res.*, **22**, 1:15–24, (1986).
115. L. Vincent and P. Soille, "Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **13**, 6:583–598, (1991).
116. A. Mangan and R. Whitaker, "Partitioning 3D Surface Meshes Using Watershed Segmentation," *IEEE Trans. on Visualization and Computer Graphics*, **5**, 4:308–321, (1999).
117. E. Rashedi and H. Nezamabadi-pour, "A stochastic gravitational approach to feature based color image segmentation," *Engineering Applications of Artificial Intelligence*, **26**, 4:1322–1332, (2013).
118. A. Moga and M. Gabbouj, "A Parallel Watershed Algorithm Based on the Shortest Paths Computation," *4th NTUG'95 & ZEUS'95 Parallel Programming and Applications*, (IOS Press), 316–324, (1995).
119. F. Meyer and S. Beucher, "Morphological segmentation," *Journal of Visual Communication and Image Representation*, **1**, 21–46, (1990).
120. L. Vincent and S. Beucher, "The morphological approach to Segmentation: An Introduction," *School of Mines, Paris*, (1989).
121. S. Beucher and F. Meyer, "The morphological approach to segmentation: the watershed transformation," *Mathematical Morphology in Image Processing*, (Ed. E.R. Dougherty), 433–481, (1993).
122. J. Roerdink and A. Meijster, "The Watershed Transform: Definitions, Algorithms and Parallelization Strategies," *Fundamenta Informaticae*, (IOS Press), **41**, 187–228, (2001).

123. C. Brice and C. Fennema, "Scene analysis using regions," *Artificial Intelligence*, **1**, 3-4:205–226, (1970).
124. A. Trémeau and P. Colantoni, "Regions Adjacency Graph Applied to Color Image Segmentation," *IEEE Trans. on Image Processing*, **9**, 4:735–744, (2000).
125. P. Felzenszwalb and D. Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision*, **59**, 2:167–181, (2004).
126. J. Lladós, E. Martí and J. Villanueva, "Symbol Recognition by Error - Tolerant Subgraph Matching between Region Adjacency Graphs," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **23**, 10:1137–1143, (2001).
127. L. Grady, "Random Walks for Image Segmentation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **28**, 11:1768–1783, (2006).
128. H. Wechsler and M. Kidode, "A random walk procedure for texture discrimination," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **PAMI-1**, 3:272–280, (1979).
129. P. Tetali, "Random walks and the effective resistance of networks," *Journal of Theoretical Probability*, **4**, 1:101–109, (1991).
130. L. Grady and G. Funka-Lea, "Multi-Label Image Segmentation for Medical Applications Based on Graph-Theoretic Electrical Potentials," *ECCV*, 230–245, (2004).
131. D. Greig, B. Porteous and A. Seheult, "Exact Maximum A Posteriori Estimation for Binary Images," *Journal of the Royal Statistical Soc. Series B*, **51**, 271–279, (1989).
132. Y. Boykov and O. Veksler, "Graph Cuts in Vision and Graphics: Theories and Applications," *Handbook of Mathematical Models in Computer Vision*, 76–96, (2005).
133. Y. Boykov and V. Kolmogorov, "An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **26**, 9:1124–1137, (2004).
134. Y. Boykov and G. Funka-Lea, "Graph Cuts and Efficient N-D Image Segmentation," *Intl. Journal of Computer Vision*, **70**, 2:109–131, (2006).
135. S. Sinha, "Graph Cut Algorithms in Vision, Graphics and Machine Learning. An Integrative Paper," *UNC Chapel Hill*, (2004).
136. L. Ford and D. Fulkerson, "Maximal Flow Through a Network," *Canadian Journal of Mathematics*, **8**, 399–404, (1956).
137. P. Elias, A. Feinstein and C. Shannon, "Note on maximum flow through a network," *IRE Trans. on Inf. Theory*, **IT-2**, 117–119, (1956).
138. S. Tatiraju and A. Mehta, "Image Segmentation using k-means clustering, EM and Normalized Cuts," *University of California, Department of Eecs*.
139. J. Shi and J. Malik, "Normalized Cuts and Image Segmentation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **22**, 8:888–905, (2000).
140. S. Flavell, S. Winter and D. Wilson, "Matching Region Adjacency Graphs," *Microprocessing and Microprogramming*, **31**, 15:31–33, (1991).
141. A. Dutta, J. Llads, H. Bunke and U. Pal, "Near Convex Region Adjacency Graph and Approximate Neighborhood String Matching for Symbol Spotting in Graphical Documents," *12th International Conference on Document Analysis and Recognition (ICDAR)*, 1078–1082, (2013).
142. P. Colantoni and B. Laget, "Color Image Segmentation using region adjacency graphs," *Sixth Intl. Conf. on Image Proc. and Its App.*, **2**, 698–702, (1997).
143. B. Peng, L. Zhang and D. Zhang, "A survey of graph theoretical approaches to image segmentation," *Pattern Recognition.*, **46**, 3:1020–1038, (2013).
144. Y. Zhang, "Image Segmentation," *Science Press*, (2001).
145. Y. Zhang, "A review of recent evaluation methods for image segmentation," *IEEE Signal Proc. Society Press. Malaysia: Proc. of the VI Intl. Symposium on Signal Proc. and Its Appl.*, (2001).
146. H. Simon, "The architecture of complexity," *Proc. of the American Philosophical Soc.*, **106**, 467–482, (1962).
147. R. Lynd and H. Lynd, "Middletown in Transition: A study in Cultural Conflicts," *Harcourt, Brace and Company, New York.*, (1937).
148. M. Newman, "Fast algorithm for detecting community structure in networks," *Physical Review E*, **69**, 6:066133, (2004).
149. M. Girvan and M. Newman, "Community structure in social and biological networks," *Proc. of the National Academy of Sciences of the USA*, **99**, 12:7821–7826, (2002).
150. D. Gómez, E. Zarrazola, J. Yáñez and J. Montero, A Divide-and-Link Algorithm for Hierarchical Clustering in Networks, *Information Sciences*, **316**, 308–328, (2015).
151. D. Gómez, E. Zarrazola, J. Yáñez, J. Rodríguez and J. Montero, "A new concept of fuzzy image segmentation," *The 11th Intl. FLINS Conference*, (2014).
152. D. Gómez, J. Montero and J. Yáñez, "A divide-link algorithm based on fuzzy similarity for clustering networks," *International Conference on Intelligent Systems Design and Applications, ISDA*, 1247-1252, (2011).
153. D. Gómez, J. Montero and G. Biging, "Improvements to remote sensing using fuzzy classification, graphs and accuracy statistics," *Pure and Applied Geophysics*, **165**, 1555–1575, (2008).
154. D. Gómez, J. Montero and J. Yáñez, "A coloring algorithm for image classification," *Information Sciences*,

176, 3645–3657, (2006).
155. D. Gómez, J. Montero, J. Yáñez and C. Poidomani, "A graph coloring algorithm approach for image segmentation," *Omega*, **35**, 173-183, (2007).
156. D. Gómez and J. Montero, "Fuzzy sets in remote sensing classification," *Soft Computing*, **12**, 243-249, (2008).
157. D. Gómez, J. Yáñez, C. Guada, J. Rodríguez, J. Montero, and E. Zarrazola, Fuzzy image segmentation based upon hierarchical clustering, *Knowledge-Based Systems*, **87**, 26–37, (2015).
158. P. Arbelaez, M. Maire, C. Fowlkes and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **33**, 5:898–916, (2011).
159. S. Lhermitte, J. Verbesselt, I. Jonckheere, K. Nackaerts, J. van Aardt, W. Verstraeten and P. Coppin, "Hierarchical image segmentation based on similarity of NDVI time series," *Remote Sensing of Environment*, **112**, 2:506-521, (2008).
160. P. Schroeter and J. Bigün, "Hierarchical image segmentation by multi-dimensional clustering and orientation-adaptive boundary refinement," *Pattern Recognition*, **28**, 5:695-709, (1995).
161. H. Cheng and Y. Sun, "A hierarchical approach to color image segmentation using homogeneity," *IEEE Trans. on Image Processing*, **9**, 12:2071–2082, (2000).
162. J. Chamorro-Martinez, D. Sanchez, B. Prados-Suarez, E. Galan-Perales and M. Vila, "A hierarchical approach to fuzzy segmentation of colour images," *IEEE Intl. Conference on Fuzzy Systems*, **2**, 966–971, (2003).
163. J. Kruskal, "On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem," *Proc. of the American Mathematical Soc.*, **7**, 48–50, (1956).
164. A. Chatterjee and P. Siarry, "Computational Intelligence in Image Processing," *Springer*, (2013).
165. G. Sharma and R. Bala, "Digital color imaging handbook," *CRC press*, (2002).
166. J. Beutel, H. Kundel and R. Van Metter, "Handbook of Medical Imaging volume 1: Physics and Psychophysics," *SPIE Press*, (2000).
167. M. Wang, "Industrial Tomography. Systems and Applications," *Woodhead Publishing*, (2015).
168. C. Evans, E. Potma, M. Puoris'haag, D. Côté, C. Lin and X. Xie, "Chemical imaging of tissue in vivo with video-rate coherent anti-Stokes Raman scattering microscopy," *Proc. of Natl. Acad. of Sciences of USA*, **102**, 46:16807–16812, (2005).
169. E. Ng, "A review of thermography as promising non-invasive detection modality for breast tumor," *Intl. Journal of Thermal Sciences*, **48**, 5:849–859, (2009).