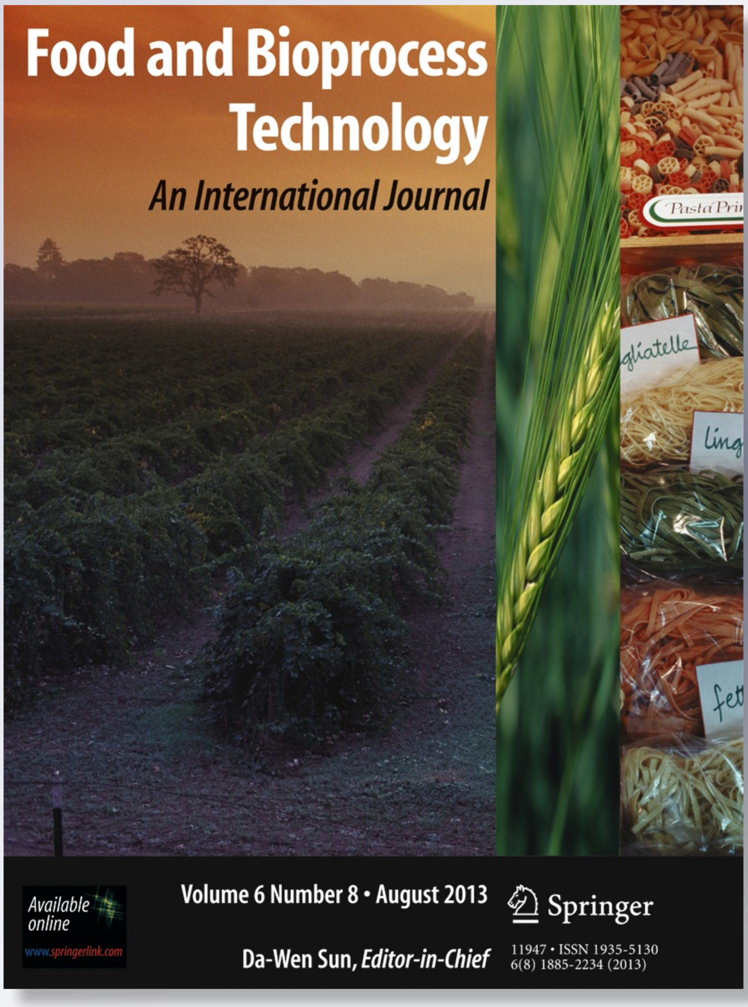
Automated Design of a Computer Vision System for Visual Food Quality Evaluation

Domingo Mery, Franco Pedreschi & Alvaro Soto

Food and Bioprocess Technology
An International Journal

ISSN 1935-5130
Volume 6
Number 8

Food Bioprocess Technol (2013)
6:2093-2108
DOI 10.1007/s11947-012-0934-2



Food Bioprocess Technol (2013) 6:2093–2108
DOI 10.1007/s11947-012-0934-2

ORIGINAL PAPER

Automated Design of a Computer Vision System for Visual Food Quality Evaluation

Domingo Mery · Franco Pedreschi · Alvaro Soto

Received: 11 January 2012 / Accepted: 10 July 2012 / Published online: 10 August 2012

Abstract Considerable research efforts in computer vision applied to food quality evaluation have been developed in the last years; however, they have been concentrated on using or developing tailored methods based on visual features that are able to solve a specific task. Nevertheless, today's computer capabilities are giving us new ways to solve complex computer vision problems. In particular, a new paradigm on machine learning techniques has emerged posing the task of recognizing visual patterns as a search problem based on training data and a hypothesis space composed by visual features and suitable classifiers. Furthermore, now we are able to extract, process, and test in the same time more image features and classifiers than before. Thus, we propose a general framework that designs a computer vision system automatically, i.e., it finds—without human interaction—the features and the classifiers for a given application avoiding the classical trial and error framework commonly used by human designers. The key idea of the proposed framework is to select—automatically—from a large set of features and a bank of classifiers those features and classifiers that achieve the highest performance. We tested our framework on eight different food quality evaluation problems yielding a classification performance of 95 % or more in every case. The proposed framework was implemented as a Matlab Toolbox available for noncommercial purposes.

Keywords Food quality evaluation · Feature extraction · Feature selection · Classification · Pattern recognition · Image analysis · Image segmentation · Image processing · Computer vision

D. Mery (✉)
Machine Intelligence Group
ASIS-UC Interdisciplinary Research Program on Tasty and Healthy Foods,
Department of Computer Science,
Pontificia Universidad Catolica de Chile,
Av. Vicuña Mackenna 4860, Santiago, Chile
e-mail: dmery@ing.puc.cl
URL: http://dmery.ing.puc.cl

F. Pedreschi
ASIS-UC Interdisciplinary Research Program on Tasty and Healthy Foods,
Department of Chemical Engineering and Bioprocesses,
Pontificia Universidad Catolica de Chile, Santiago, Chile

A. Soto
Machine Intelligence Group,
Department of Computer Science,
Pontificia Universidad Catolica de Chile, Santiago, Chile

Introduction

Computer vision has emerged in the last years as a standard technology in nondestructive food quality evaluation where relevant information can be extracted from visual features (Sun 2008; Irudayaraj and Reh 2008; Davies 2000). Typically, visual food quality evaluation is used in several applications like grading, detection of defects, quality determination, measurements, characterization, and detection of contaminants, among others. In general terms, the problem of visual evaluation of food quality can be considered as a problem of visual pattern recognition, where the goal is to detect in the food products specific patterns related to relevant quality standards. According to our literature review, many research directions have been exploited, some very different principles have been adopted, different imaging systems have been used (color, X-ray, ultrasound, computer tomography, near-infrared radiation, etc.), and a wide variety of algorithms have been appeared in the literature. The reader can find several applications in food quality evaluation, for example, in meats (Tan 2004), salmon fillets (Aguilera et al. 2007), pizzas (Sun 2000), potatoe chips (Pedreschi et al. 2004), bananas (Quevedo et al. 2008), table olives (Diaz et al. 2004), apples (Leemans and Destain 2004), corn tortillas (Mery et al. 2010), blueberries (Leiva et al. 2011), detection of foreign objects in packaged foods (Kwon et al. 2008), detection of fish bones (Mery et al. 2011), identification of insect infestation in citrus (Jiang et al. 2008), detection of codling moth larvae in apples (Haff and Toyofuku 2008), fruit quality inspection like split pits, water content distribution and internal structure (Ogawa et al. 2003), detection of larval stages of the granary weevil in wheat kernels (Haff and Slaughter 2004), and several others (Sun 2008; Irudayaraj and Reh 2008; Davies 2000).

A computer vision system for food quality evaluation typically follows a five-step schema, as shown in Fig. 1 (see for example Gonzalez and Woods 2008):

1. *Image acquisition*: A digital image of the food under test is taken (sometimes resized) and stored in a computer.
2. *Preprocessing*: The digital image is improved in order to enhance the details.
3. *Segmentation*: The food image is found and isolated from the background of the scene.
4. *Feature extraction/selection*: Significant features of the food image are quantified.
5. *Classification*: The extracted features are interpreted automatically using knowledge about the analyzed food in order to evaluate its quality.

In the last years, considerable research efforts in visual food quality evaluation have been concentrated on using or developing tailored features and classifiers that are able to solve a specific task. The traditional strategy to solve this problem is to use a low number of features and a few number of classifiers (sometimes only one classifier), probably because only a few number of them are available in common software packages and, additionally, because the cost is considerably reduced for experimenting, training, and testing. Using today's computer capabilities, however, it is now possible to extract a very large number of features (e.g., 3,000 features as shown in Fig. 2) and to test several state-of-the-art classifiers (e.g., 25 classifiers), in order to select which features (e.g., only 10 of them) are really relevant and at the same time which classifier (e.g., a support vector machine)

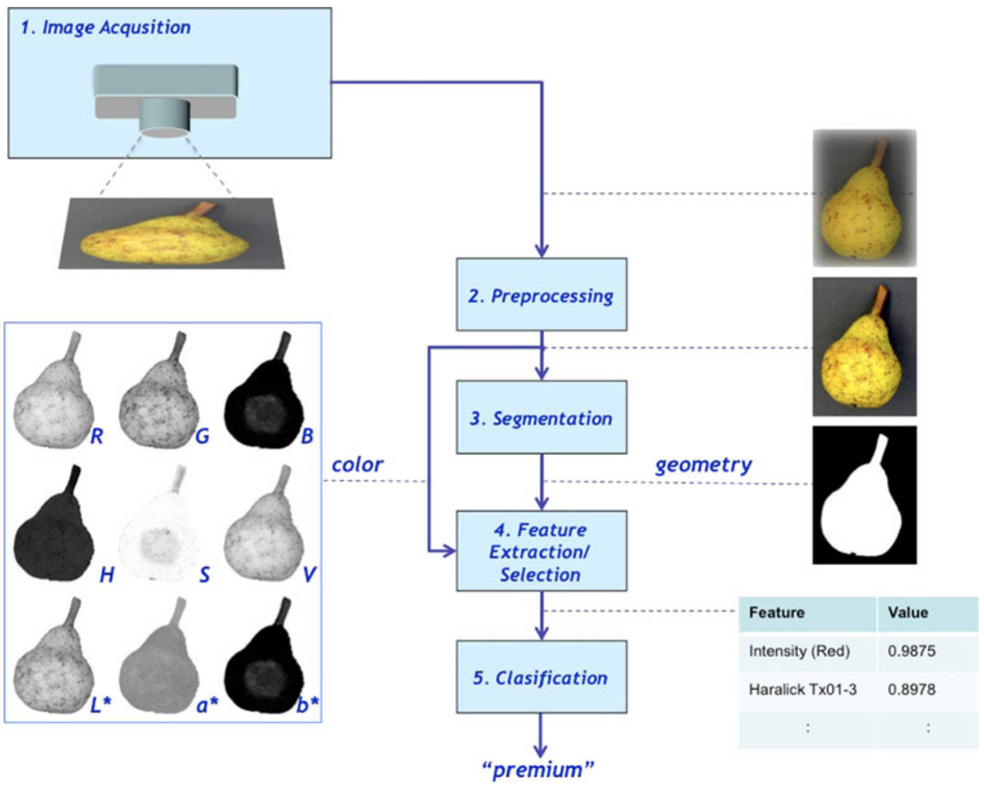
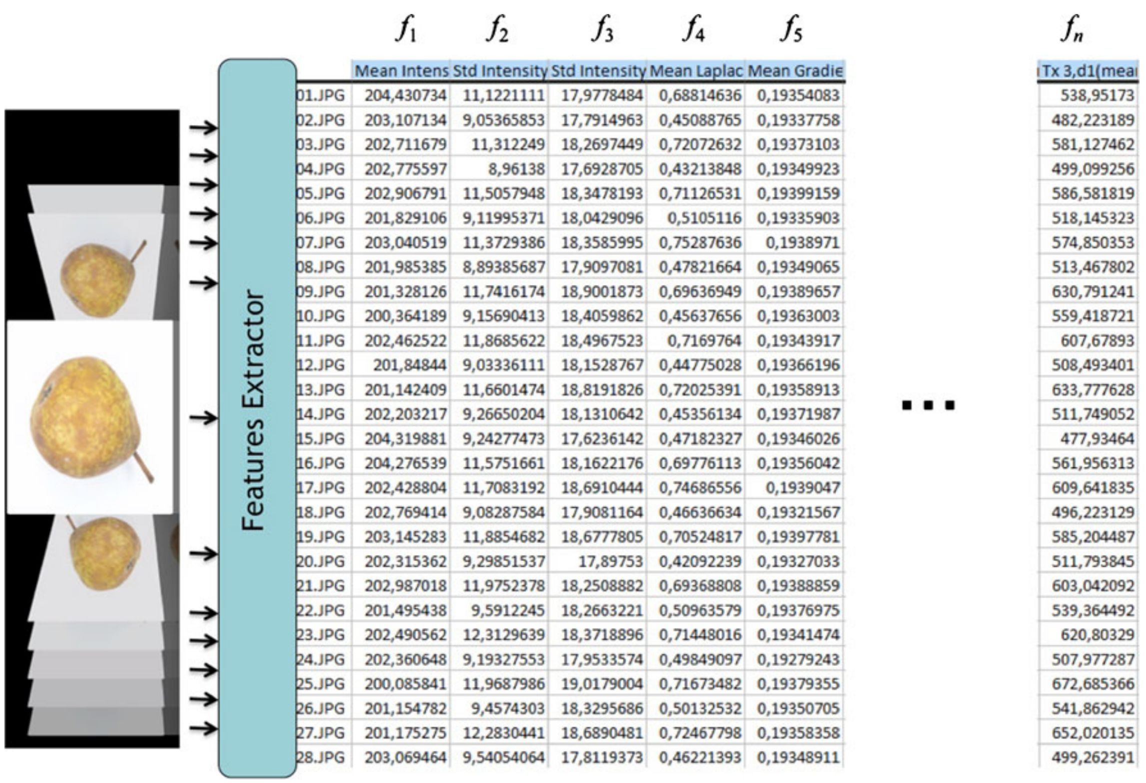


Fig. 1 Example of a computer vision system for food quality evaluation using geometry and color information obtained from a digital image. In this example, the quality of a fruit is determined automatically according to its visual attributes



	f_1	f_2	f_3	f_4	f_5	...	f_n
	Mean Intens	Std Intensity	Std Intensity	Mean Laplac	Mean Gradie		Tx 3,d1(mea
01.JPG	204,430734	11,1221111	17,9778484	0,68814636	0,19354083		538,95173
02.JPG	203,107134	9,05365853	17,7914963	0,45088765	0,19337758		482,223189
03.JPG	202,711679	11,312249	18,2697449	0,72072632	0,19373103		581,127462
04.JPG	202,775597	8,96138	17,6928705	0,43213848	0,19349923		499,099256
05.JPG	202,906791	11,5057948	18,3478193	0,71126531	0,19399159		586,581819
06.JPG	201,829106	9,11995371	18,0429096	0,5105116	0,19335903		518,145323
07.JPG	203,040519	11,3729386	18,3585995	0,75287636	0,1938971		574,850353
08.JPG	201,985385	8,89385687	17,9097081	0,47821664	0,19349065		513,467802
09.JPG	201,328126	11,7416174	18,9001873	0,69636949	0,19389657		630,791241
10.JPG	200,364189	9,15690413	18,4059862	0,45637656	0,19363003		559,418721
11.JPG	202,462522	11,8685622	18,4967523	0,7169764	0,19343917		607,67893
12.JPG	201,84844	9,03336111	18,1528767	0,44775028	0,19366196		508,493401
13.JPG	201,142409	11,6601474	18,8191826	0,72025391	0,19358913		633,777628
14.JPG	202,203217	9,26650204	18,1310642	0,45356134	0,19371987		511,749052
15.JPG	204,319881	9,24277473	17,6236142	0,47182327	0,19346026		477,93464
16.JPG	204,276539	11,5751661	18,1622176	0,69776113	0,19356042		561,956313
17.JPG	202,428804	11,7083192	18,6910444	0,74686556	0,1939047		609,641835
18.JPG	202,769414	9,08287584	17,9081164	0,46636634	0,19321567		496,223129
19.JPG	203,145283	11,8854682	18,6777805	0,70524817	0,19397781		585,204487
20.JPG	202,315362	9,29851537	17,89753	0,42092239	0,19327033		511,793845
21.JPG	202,987018	11,9752378	18,2508882	0,69368808	0,19388859		603,042092
22.JPG	201,495438	9,5912245	18,2663221	0,50963579	0,19376975		539,364492
23.JPG	202,490562	12,3129639	18,3718896	0,71448016	0,19341474		620,80329
24.JPG	202,360648	9,19327553	17,9533574	0,49849097	0,19279243		507,977287
25.JPG	200,085841	11,9687986	19,0179004	0,71673482	0,19379355		672,685366
26.JPG	201,154782	9,4574303	18,3295686	0,50132532	0,19350705		541,862942
27.JPG	201,175275	12,2830441	18,6890481	0,72467798	0,19358358		652,020135
28.JPG	203,069464	9,54054064	17,8119373	0,46221393	0,19348911		499,262391

Fig. 2 In this example, n features for dessert pears are extracted to build a features table, where each row represents a sample and each column a feature. Typically, $n > 3{,}000$ for color imaging

achieves the highest performance for these selected features.[1]

Exploiting this idea, we propose a general computer vision framework that can be used in many applications on visual food quality evaluation. The proposed framework—independent of the application—extracts initially the *same large set* of features in order to automatically select from them (1) relevant features and (2) an accuracy classifier (chosen from a set of classifiers). The key idea is that the selected features in combination with the selected classifier could solve the specific food quality evaluation problem. It is well known in machine learning community that the dimensionality reduction, i.e., finding and exploiting low-dimensional structures in high-dimensional data, is used to reduce the computational complexity, remove noise, make the results easier to understand and interpret, and enhance the generalization and accuracy and robustness of the classification (Marsland 2009). Our hypothesis is that in this large set of features and in this set of classifiers, there is a subset of features and a classifier that can be selected to perform accurately a specific classification task. Thus, our framework designs a computer vision system automatically, i.e., it finds—without human interaction—the features and the classifier for a given application avoiding the classical trial and error framework commonly used by human designers and, therefore, reducing considerably the cost for experimenting, training, and testing.

The proposed framework is deeply rooted in an emerging computer vision paradigm based on machine learning techniques. Here, the idea is to use training data to automatically explore a hypothesis space composed of visual features and classifier parameters. The goal is to find the hypothesis that best models or discriminates a relevant hidden relation from the application domain that is represented by the training data. Our framework follows this intuition by exploring a highly rich set of features and classifiers, such as a suitable hypothesis or solution can be identified after an effective exploration process. In terms of the applicability of the proposed framework, it is important to consider that the intensive exploration process is performed *offline*, while at detection time, when suitable features and classifier are selected, we only need a processing load

[1]This problem is known in the literature as *full model selection* (Escalante et al. 2009).

similar to current tailored solutions to food quality evaluation.

From the previous intuition, a relevant question arises: what is a suitable hypothesis space such as an effective exploration will deliver a proper solution to a given visual food quality evaluation task? Here, we rely on years of research on the computer vision domain which has delivered a valuable amount of knowledge, comprehension, and algorithms related to the detection of visual patterns. In particular, there is a wide agreement with respect to the fact that depending of the problem at hand, the key information to detect a visual pattern can be hidden in a diverse set of visual cues such as shape, texture, color, position, orientation, or depth disparity (Szeliski 2011). In fact, it is the high multidimendionality of visual information which provides strong discriminative capabilities to eliminate ambiguities and to provide key information to detect a target pattern. In our framework, we integrate a wide variety of off-the-shelf feature extraction algorithms, able to provide information about different visual cues. As our experiments validate, we believe that such feature space provides a hypothesis space rich enough to solve common problems related to food quality evaluation.

The rest of the paper is organized as follows: In "Materials and Methods," the proposed computer vision framework and the experiments are explained. In "Results and Discussion," the results obtained in several experiments are shown. Finally, in "Conclusions," some concluding remarks are given.

Materials and Methods

The proposed general framework used to design automatically a computer vision system for food quality evaluation consists on four steps: (1) data collection, (2) large feature extraction, (3) feature selection, and (4) classifier selection. The framework considers robust segmentation approaches, almost 3,000 features containing geometric and intensity information, seven feature selection algorithms, seven families of classifiers, five performance validation techniques, and one classifier selection technique.[2] Below, we describe each of these steps in further details using two examples on food quality evaluation of dessert pears:

- Example A: It refers to the quality determination according to external appearance.
- Example B: It refers to the detection of codling moth larvae inside of the samples.

Data Collection

According to the application, the electromagnetic spectrum must be chosen in order to obtain good quality digital images where the foods to be evaluated are visible. For example, optical and X-ray images may be used respectively to evaluate the external and internal properties of food products in a noninvasive and a nondestructive way. Afterwards, preprocessing and segmentation techniques can be applied in order to achieve digital images ready to be analyzed. In this step, N representative samples are captured, preprocessed, and segmented. We differentiate two kinds of samples: (1) food products, for evaluation of the whole food (like "example A") and (2) food parts, for evaluation of parts of the foods only (like "example B").

The obtained images are defined as $\mathbf{I}_i$, for $i = 1, ..., N$. The N representative samples must be labeled by experts or a sensorial panel to establish the class of each sample. This set of samples represents the labeled training set that allows us to perform a *supervised training* (Mitchell 1997; Duda et al. 2001). The label (or class) for each sample is defined as $y_i \in \{1, ..., C\}$, for C possible classes. For instance, in example A, the quality of the fruit can be categorized into one of three classes: "rejected," "accepted," and "premium," i.e., $y_i \in \{1, 2, 3\}$. On the other hand, in example B, the inner part of the fruit can be categorized into one of two classes: "with insect" and "without insect," i.e., $y_i \in \{1, 2\}$.

In this subsection, the reader can find some guidelines for color imaging, where image acquisition, preprocessing, and segmentation techniques are addressed in a general way. For other parts of the electromagnetic spectrum, other techniques should certainly be used. For color imaging, a computer vision system, as shown in Fig. 1, similar to the one described in Pedreschi et al. (2004) can be employed to capture food images. The food samples can be illuminated using a constant light source such as fluorescent lamps with a color temperature of 6,500 K (D65, standard light source commonly used in food research). The digital images can be captured using a commercial digital camera with the required resolution, e.g., 4 megapixels or more. In order to obtain food images in $L^*a^*b^*$ color space, a similar framework that was reported in Leon et al. (2006) can be used, where the color values $L^*a^*b^*$ of different color charts are measured using a colorimeter, and R, G, and B color values of the corresponding charts are obtained using RGB color images. Thus, a

[2] A previous version of this paper can be found in Mery and Soto (2008) where preliminary results are presented.

linear model is used to estimate the L^*a^*b values from the RGB values in each pixel of the image.

Independent of the electromagnetic spectrum of the imaging system, the captured images can be resized and segmented. In order to reduce the computational cost, the images can be resized to x times the size of the original images using interpolation (Gonzalez and Woods 2008).

In order to analyze only the information required for the food quality evaluation, those parts of the image (foreground) must be isolated from the rest of the image (background). In example A, the pixels of the pear must be segmented from the background pixels. In example B, the candidate pixels of a larvae must be separated from the pixels of the pear. For this segmentation task, two general approaches can be used: a traditional image segmentation or a *sliding-window* approach. In our framework, we provide tools to detect the relevant visual patterns using both alternatives.

Traditional Segmentation It is the typical approach used to identify the relevant part of the input image, and there is an extensive list of algorithms and applications based on this approach (Cheng et al. 2001). For food images, however, we suggest the algorithm developed in Mery and Pedreschi (2005). The method has the following three steps:

1. For color images, a high contrast gray value image **J** is computed from an optimal linear combination of the RGB color components that ensures a high standard deviation in transformed image. In case that the images are gray value images, **J** can be defined as the equalized images of them (Gonzalez and Woods 2008).
2. An initial binary image **K** by thresholding **J** is computed with a global threshold estimated from the histogram of **J**.
3. A final binary image **R** is computed using morphological operations on **K** to remove isolated pixels in background and to fill holes in foreground.

An example for this segmentation is shown in Fig. 3.

Sliding-Window Approach Segmentation basically acts as a focus of attention mechanism that filters the information that is fed to the following steps of the systems, as such failure in the segmentation is catastrophic for the performance of the system. Inherent limitations of traditional segmentation algorithms for complex tasks and increasing computational power have fostered the emergence of an alternative approach based on the so-called sliding-window paradigm. Sliding-window approaches have established themselves as state of the art in computer vision problems where a visually complex object must be separated from the background (see, for example, successful applications in face detection (Viola and Jones 2004) and human detection (Dalal and Triggs 2005)). In sliding-window framework, a detection window is moved over an input image in both horizontal and vertical directions, and for each localization of the detection window, a classifier decides to which class the corresponding portion of the image belongs according to its features. Here, a set of candidate image areas is selected and all of them are fed to the subsequent parts of the image analysis algorithm. This resembles

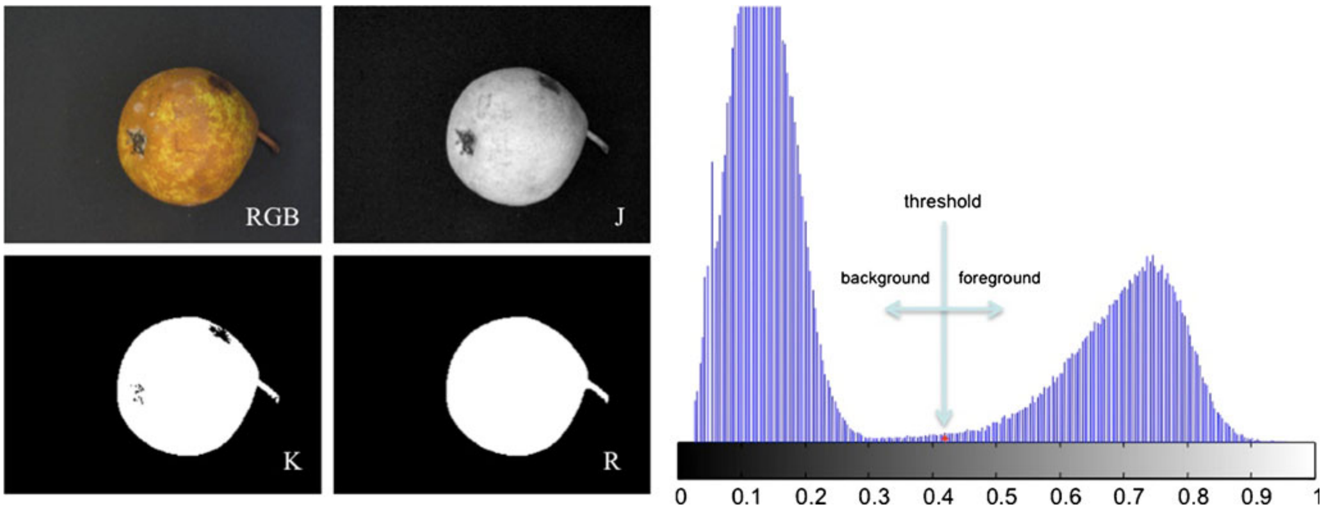


Fig. 3 Segmentation of a RGB food image using high contrast image **J**, global thresholding **K** obtained from its histogram (see *right*), and morphological operations **R**

a brute force approach where the algorithm explores a large set of possible segmentations, and at the end, the most suitable is selected by the classification steps. An example for this approach is shown in Fig. 4. The segmented region, enclosed in this example by the red line, will define a binary image **R**.

If no segmentation is required, the binary image **R** will be the whole image. With or without segmentation, the idea is to have a binary image **R**, in which pixels equal to "1" represent the pixels where the food is and "0" where not.

Large Feature Extraction

In this subsection, we explain how a large set of features is obtained. For each sample i, an image $\mathbf{I}_i$ is captured and a segmented image $\mathbf{R}_i$ is computed. From them, n features are extracted building a large feature matrix $\mathbf{F}$ with $N \times n$ elements (features). The element F_{ij} of $\mathbf{F}$ corresponds to the jth feature of ith sample, for $i = 1, ..., N$ and $j = 1, ..., n$. Features extracted for each food sample are divided into two families: *geometric* and *intensity* features, as shown in Fig. 5. An illustration of example A is given in Fig. 2. It is worth noting that if no segmentation is required, the geometric information will have no sense because all shapes will be the rectangles of the whole images. Additionally, if food quality does not depend on location, rotation, and background color, features related to these characteristics must not be extracted in order to avoid false correlations.

Geometric Features They provide information on the location, size, and shape of the food region. Location and size features, such as center of gravity, perimeter, height, and width, are given in pixels. Shape features are usually coefficients without units. The following geometric features can be extracted for each region:

1. *Basic*: Simple location, orientation, size, and shape information like center of gravity, area, perimeter, orientation, Euler number, and solidity, among others (MathWorks 2003). There are 18 basic geometric features.
2. *Elliptical*: Location, size, and shape information extracted from a fitted ellipse to the boundary of the region (Fitzgibbon et al. 1999). There are seven elliptical features.
3. *Fourier descriptors*: Shape information invariant to scale, orientation, and position based on Fourier descriptors (Zahn and Roskies 1971; Chellappa and Bagdazian 1984; Persoon and Fu 1977). We use 16 Fourier descriptors.
4. *Invariant moments*: Shape information invariant to scale, orientation, and position based on Hu (1962) moments and invariant to affine transformation based on Flusser and Suk (1993) and Gupta and Srinath (1987) moments. There are 14 invariant moments.

In this family, the number of geometric features obtained is $n_g = 55$. Nevertheless, in our experiments, $n_g = 49$ because location and orientation features are not considered in order to avoid false correlations.

Intensity Features They provide information about the intensity of a food region. For gray value images, e.g., X-ray images, there is only one intensity channel; however, for color images, there are originally three intensity channels (R, G, and B), and other ones, like L^*, a^*, and b^*, for example, if other color spaces are used. The following five groups of intensity features are used:

1. *Basic*: Simple intensity information related to the mean intensity in the region; standard deviation, kurtosis, and skewness of the intensity in the region; and in the image, mean first derivative in the boundary of the region (gradient) and second derivative (laplacian) in the region. Additionally, five contrast measurements can be extracted in order to analyze the intensity difference between object and background (Mery and Filbert 2002), e.g., burned spots in a tortilla (see Fig. 4). There are 11 basic intensity features.
2. *Statistical textures*: Texture information extracted from the distribution of the intensity values based on the Haralick approach (Haralick 1979). They are computed utilizing *co-occurrence matrices* that represent second-order texture information (the joint probability distribution of intensity pairs of neighboring pixels in the image), where mean and range—for five different pixel distances in eight directions—of the following variables were measured: (1) angular second moment, (2) contrast, (3) correlation, (4) sum of squares, (5) inverse difference moment, (6) sum average, (7) sum entropy, (8) sum variance, (9) entropy, (10) difference variance, (11) difference entropy, (12) information measures of correlation, and (14) maximal correlation coefficient. There are $2 \times 14 \times 5 = 140$ statistical features.
3. *Local binary patterns*: Texture information extracted from occurrence histogram of *local binary patterns* (LBP) computed from the relationship between each pixel intensity value with its eight neighbors. The features are the frequencies of each one of the histogram bins. LBP are very robust in terms of gray-scale and rotation variations (Ojala

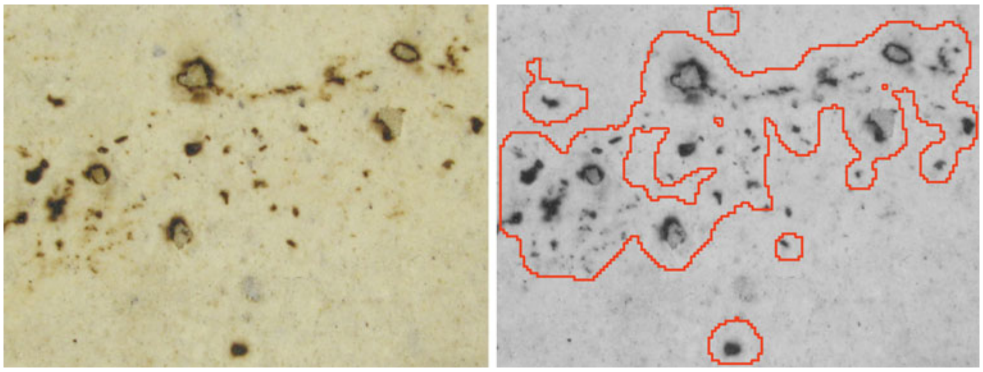


Fig. 4 Detection of burned surface on a tortilla using sliding windows: original image (*left*) and segmentation (*right*)

et al. 2002). Other LBP features like *semantic* LBP (sLBP) (Mu et al. 2008) can be used in order to bring together similar bins. We use 59 uniform LBP features (LBP-u2), 36 rotation invariant LBP features (LBP-ri), and 31 semantic LBP features (sLBP).

4. *Filter banks*: Texture information extracted from image transformations like discrete Fourier transform (DFT)—magnitude and phase—, discrete cosine transform (DCT) (Gonzalez and Woods 2008), and Gabor features based on 2D Gabor functions, i.e., Gaussian-shaped bandpass filters, with dyadic treatment of the radial spatial frequency range and multiple orientations, which represent an appropriate choice for tasks requiring simultaneous measurement in both space and frequency domains (usually eight scales and eight orientations). Additionally, the maximum, the minimum, and the difference between both are computed (Kumar and Pang 2002). We use 16 DCT features, 16 Fourier features, and $8 \times 8 + 3$ Gabor features, i.e., $16 + 2 \times 16 + 67 = 115$ filter bank features.
5. *Invariant moments*: Shape and intensity information based on Hu (Hu 1962) moments. There are seven Hu-intensity moments.

In this family, the number of features that can be extracted is $n_t = 399$. As mentioned above, in order to avoid false correlations, features that depend on contrast or orientation must not be considered when food quality is not related to these characteristics. In this set of features, 5 of them depend on the background color (contrast features) and 202 of them depend on the orientation of the object (certain LBP, DCT, Fourier, and Gabor features). For this reason, in example A, the number of intensity features per channel is $n_t = 192$. Thus, it is possible to obtain $10n_t = 1,920$ intensity features, as shown in Fig. 1, by using red, green, and blue (from RGB color space); hue, saturation, and value (from HSV color space); L^*, a^* and b^* (from $L^*a^*b^*$ color space); and gray. On the other, in example B with X-ray images (gray value image), we could include the contrast features because the background is part of the food product, i.e., $n_t = 197$.

All features used in the proposed framework are summarized in Table 1. Totally, for n_c intensity channels, there are $n = n_g + n_c n_t$ features. This number can be $n = 55 + 10 \times 399 = 4,045$ features for color images with 10 intensity channels and $n = 55 + 399 = 454$ for gray value images. Nevertheless, in example A and example B, the total number of extracted features

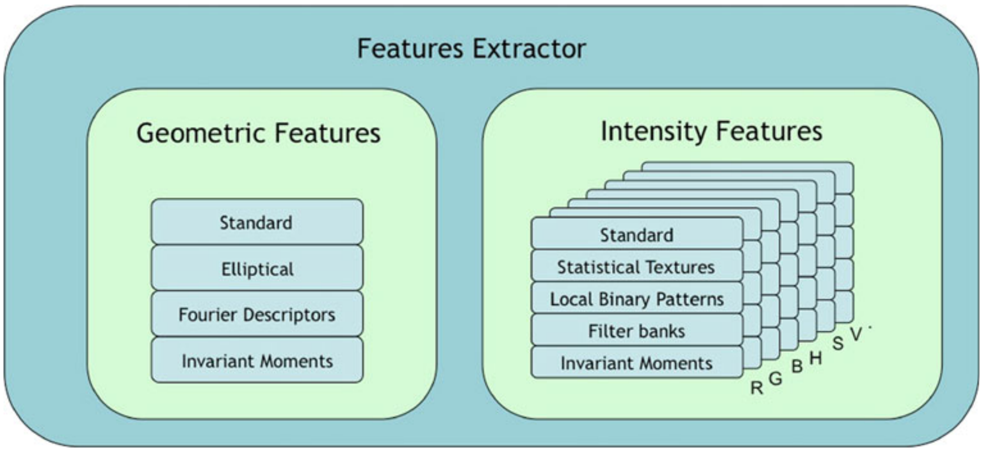


Fig. 5 In the training process, a large number of geometric and intensity features for each sample are extracted. The same set of intensity features can be extracted for red, green, blue, etc. channels. If the image is a gray value image, e.g., an X-ray image, only the gray value channel is used

Table 1 Extracted features

Family	Group	Name and references
Geometric	1. Basic	Center of gravity *i*, center of gravity *j*, height, width, area, perimeter, roundness, Danielsson factor, Euler number, equivalent diameter, major axis length, minor axis length, orientation, solidity, extent, eccentricity, convex area and filled area (MathWorks 2003; Nixon and Aguado 2008; Danielsson 1978).
	2. Elliptical	Major axis, minor axis, eccentricity, orientation, center of gravity *i*, center of gravity *j* and area (Fitzgibbon et al. 1999).
	3. Fourier descriptors	FourierDes (0,..., 15) (Zahn and Roskies 1971; Chellappa and Bagdazian 1984; Persoon and Fu 1977).
	4. Invariant moments	Hu (1,..., 7) (Hu 1962). Flusser (1,..., 4) (Flusser and Suk 1993). Gupta (1,..., 3) (Gupta and Srinath 1987).
Intensity	1. Basic	Mean intensity, standard deviation intensity, kurtosis, skewness, standard deviation intensity with neighbor, mean gradient, mean Laplacian, and contrasts (Mery and Filbert 2002).
Channels 1: Gray, 2: RGB, 3: HSV, 4: $L^*a^*b^*$	2. Statistical textures	$Tx(k, p)$ (mean/range) for k=1. angular second moment, 2. contrast, 3. correlation, 4. sum of squares, 5. inverse difference moment, 6. sum average, 7. sum entropy, 8. sum variance, 9. entropy, 10. difference variance, 11. difference entropy, 12 and 13. information measures of correlation, and 14. maximal correlation coefficient, and p=1,..., 5 pixels (Haralick 1979).
	3. Local binary patterns	LBPu2 (1,..., 59), LBPri (1,..., 36) (Ojala et al. 2002) and sLBP (1,..., 31) (Mu et al. 2008).
	4. Filter banks	DFT (1, 4; 1, 4) and DCT (1, 4; 1, 4) (Gonzalez and Woods 2008). Gabor (1,..., 8; 1,..., 8), max(Gabor), min(Gabor), Gabor-J (Kumar and Pang 2002).
	5. Invariant moments	Int-Hu (1,..., 7) (Hu 1962).

is $n^{(A)} = 49 + 10 \times 192 = 1,969$ and $n^{(B)} = 49 + 1 \times 197 = 246$, respectively, because location, orientation, and contrast features are removed in order to avoid false correlations. Thus, two different feature matrices are obtained: $\mathbf{F}^{(A)}$ with $N^{(A)} \times n^{(A)}$ elements and $\mathbf{F}^{(B)}$ with $N^{(B)} \times n^{(B)}$ elements, where $N^{(A)}$ and $N^{(B)}$ are the number of samples in each example.

In the proposed framework, independent of the application, a large number of features are extracted. Many of them are probably not used by the implemented classifier; however, the decision on which are the relevant features takes place automatically in the next step called feature selection.

Feature Selection

As explained in the previous section, n features are extracted from each food image. Afterwards, the features must be selected in order to decide on the relevant features for the classification tasks, namely, the automatic determination of the given classes by experts or a sensorial panel.

The n extracted features for sample i are arranged in the ith row of matrix $\mathbf{F} : [F_{i1} \ldots F_{in}]$ that corresponds to a point in the n-dimensional measurement feature space. The features are normalized yielding a $N \times n$ matrix W which elements are defined as

$$W_{ij} = \frac{F_{ij} - \mu_j}{\sigma_j} \tag{1}$$

for $i = 1, \ldots, N$ and $j = 1, \ldots, n$, where F_{ij} denotes the jth feature of the ith feature vector, N is the number of samples, and μ_j and σ_j are the mean and standard deviation of the jth feature, i.e., the jth column of $\mathbf{F}$. Thus, the normalized features have 0 mean and a standard deviation equal to 1. Those constant features (e.g., Euler number when foods have no holes) and the high correlated features (e.g., intensity features extracted from hue channel and gray value) can be eliminated because they do not give relevant information about the food evaluation quality.

In feature selection, a subset of m features ($m \leq n$) that leads to the smallest classification error is selected. The selected m features are arranged in a new matrix $\mathbf{X}$ with $N \times m$ elements obtained from m selected columns of the large set of normalized features $\mathbf{W}$. In our examples, a set $\mathbf{X}^{(\mathrm{A})}$ of $m^{(\mathrm{A})}$ selected features can be determined from $\mathbf{W}^{(\mathrm{A})}$ for example A and a different set $\mathbf{X}^{(\mathrm{B})}$ of $m^{(\mathrm{B})}$ selected features from $\mathbf{W}^{(\mathrm{B})}$ for example B.

The features can be selected using several state-of-the-art algorithms reported in the literature. In the proposed framework, the following feature selection algorithms have achieved high performance:[3]

1. *Sequential forward selection* (SFS): This method selects the best single feature and then adds one feature at a time that, in combination with the selected features, maximizes a classification performance objective function. The objective function can be (1) the Fisher criteria that ensure simultaneously a small intraclass variation and a large interclass variation in the space of the selected features (SFS-Fisher) or (2) the performance obtained using a specific classifier as a wrapper method (e.g., SFS-SVM when a classifier based on support vector machine is used, Jain et al. 2000).
2. *Ranking by class separability criteria* (rank): Features are ranked using an independent evaluation criterion to assess the significance of every feature for separating two labeled groups. The absolute value two-sample Student's t test with pooled variance estimate is used as an evaluation criterion (MathWorks 2007).

In our framework, there are s feature selection algorithms available (including the above-mentioned algorithms and their variations): $\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_s$ with $m_1, m_2, ..., m_s$ features.

Classifier Selection

This section describes the design of a classifier once the proper features are selected. A classifier h assigns a feature vector $\mathbf{x}$ (a row of the selected features stored in matrix $\mathbf{X}$) to one of the determined classes $c \in \{1, ..., C\}$, where C is the number of classes. According to the selected features, different state-of-the-art classifiers are tested in order to obtain the highest performance. The idea is to obtain the highest accuracy defined as the proportion of true results. In the proposed framework, the following well-known classifiers have achieved high performance:[4]

1. *Mahalanobis distance*: It assigns a test feature vector $\mathbf{x}$ to class c if the Mahalanobis distance between $\mathbf{x}$ and $\bar{\mathbf{x}}_c$ is minimal, where $\bar{\mathbf{x}}_c$ is the mean value of the training samples for class c. The Mahalanobis classifier takes into account errors associated with prediction measurements, such as noise, by using the feature covariance matrix to scale features according to their variances (Duda et al. 2001).
2. *Linear discriminant analysis* (LDA): It computes linear decision boundaries in the feature space in order to separate it into a set of regions according to the classification. The boundaries are computed assuming that the classes have Gaussian densities with a common covariance matrix. The test feature vector $\mathbf{x}$ is assigned to the class determined by the boundaries (Hastie et al. 2001; Webb 2005).
3. *Nearest neighbors* (k-NN): It is a nonparametric approach, in which the k most similar training samples to a given test feature vector $\mathbf{x}$ are determined. The assigned class is the most frequent class from those k samples (Duda et al. 2001).
4. *Support vector machines* (SVM): It transforms a two-class feature space, where the classes overlap, into a new enlarged feature space where the classification boundary is linear. Thus, a simple linear classification can be designed in the transformed feature space in order to separate both classes (Shawe-Taylor and Cristianini 2004). The original feature space is transformed using a function $f(\mathbf{x})$; however, for the classification only, the kernel function $K(\mathbf{x}, \mathbf{x}') = \langle f(\mathbf{z}), f(\mathbf{z}')\rangle$ that computes inner products in the transformed space is required. The following kernels can be used: (1) linear, (2) quadratic, (3) polynomial, (4) gaussian radial basis function, and (5) multilayer perceptron kernel.
5. *Neural networks* (NN): It consists of artificial neurons connected in a network that is able to classify a test feature vector $\mathbf{x}$ evaluating a linear or nonlinear weighted sum of functions. The weights, the functions, and the connections are estimated

[3] In our framework, other methods were implemented, however, their performance and their computational time were not satisfactory in our experiments. The other methods are: Forward Orthogonal Search (Wei and Billings 2007), Least Square Estimation (Mao 2005), Combination with Principal Components, (Duda et al. 2001), Future Selection based in Mutual Information (Peng 2005).

[4] In our framework, other methods were implemented; however, their performance and their computational time were not satisfactory in our experiments. The other methods are minimal distance (Duda et al. 2001), quadratic discriminat analysis (Hastie et al. 2001; Webb 2005), boosting and AdaBoost (Polikar 2006; Viola and Jones 2004), and probabilistic neural networks (MathWorks 2009; Duda et al. 2001).

in a training phase by minimizing the classification error (Bishop 2005, 2006).

In our framework, there are r classifiers available (including the above-mentioned algorithms and their variations): $h_1, \dots, h_r$. In order to find the *best* classifier, we evaluate the performance of the r classifiers on the s subsets of selected features using an exhaustive search as shown in algorithm 1. The performance η is defined as the proportion of true results. As we can see, the performance of the classification is evaluated using *cross-validation* (widely used in machine learning problems, Kohavi 1995). In cross-validation, the data are divided into v folders. A portion $(v-1)/v$ of the whole data is used to train and the rest $(1/v)$ for test. This experiment is repeated v times rotating train and test data to evaluate the stability of the classifier. Then, when training is performed, the samples that were initially removed can be used to test the performance of the classifier on these test data. Thus, one can evaluate the generalization capabilities of the classifier by testing how well the method will classify samples that have not already examined. The estimated performance, η, is calculated as the mean of the v percentages of the true classifications that are tabulated in each case. In our experiments, we use $v = 10$ folders.[5] Confidence intervals, where the classification performance η expects to fall, are obtained from the test sets. These are determined by the cross-validation technique, according to a Student's t test (Mitchell 1997). Thus, the performance and also the confidence can be assessed. According to algorithm 1, the highest achieved performance (using all classifiers and all selected features) is computed as $\hat{\eta}$, and this performance will be reported in our experiments.

Algorithm 1 Classifier selection

```
Input: Subsets $\{\mathbf{X}_i, m_i\}_{i=1}^{s}$ of selected features , and
    labels $\mathbf{y}$ of the samples.
 1: $\hat{\eta} = 0$//Initialization of the highest performance
 2: for $i = 1$ to $s$ do
 3:   for $j = 1$ to $m_i$ do
 4:     $\mathbf{X}$ = LeftColumns($\mathbf{X}_i$, $j$) //First $j$ selected features of $\mathbf{X}_i$
 5:     for $k = 1$ to $r$ do
 6:       $\eta_k$ = CrossValidation($h_k, \mathbf{X}, \mathbf{y}$) //Performance of classifier $h_k$ on data $\mathbf{X}$
 7:       if $\eta_k > \hat{\eta}$ then
 8:         $\hat{\eta} = \eta_k$ //Highest performance
 9:         $\hat{\mathbf{X}} = \mathbf{X}$ //Selected features
10:         $\hat{i} = i$ //Best feature selection approach
11:         $\hat{j} = j$ //Number of selected features
12:         $\hat{k} = k$ //Best classifier
13:       end if
14:     end for
15:   end for
16: end for
Output: $\hat{\eta}, \hat{\mathbf{X}}, \hat{i}, \hat{j}, \hat{k}$
```

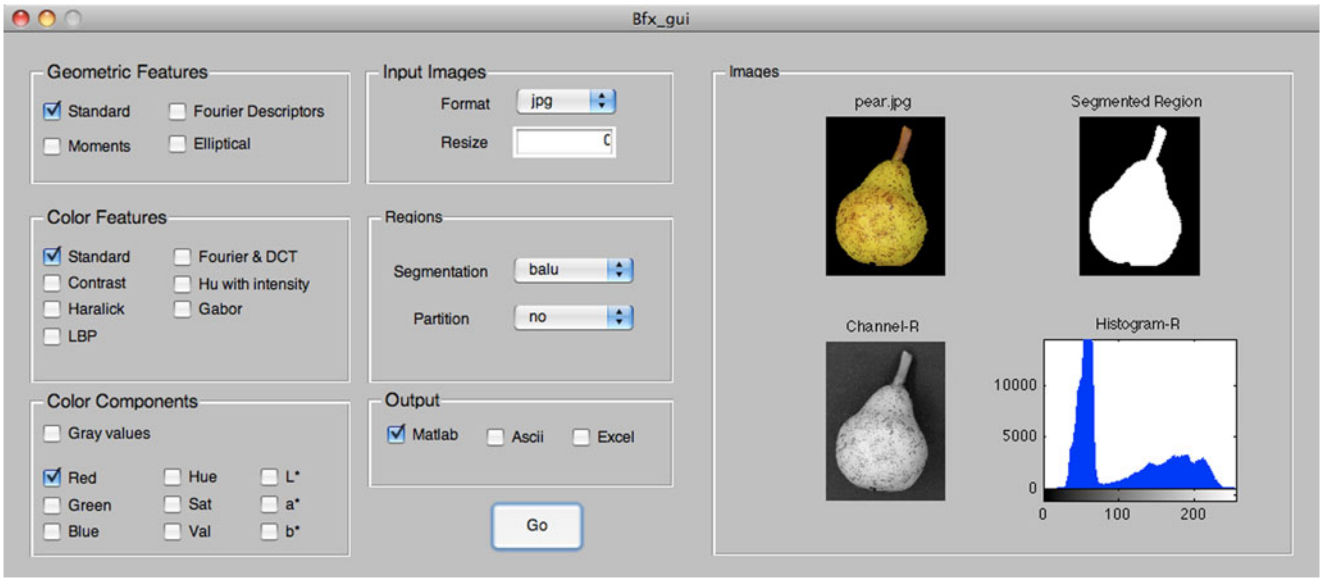
In the mentioned example A and example B, the selected classifiers could be, respectively, a neural network trained using the first 10 features selected after SFS algorithm and a linear support vector machine trained using the first 12 features selected after rank algorithm.

Experiments

Our experiments uses a framework implemented in *Balu Matlab Toolbox* (Mery 2011). In the developed toolbox, there are two main graphic user interfaces: (1) For feature extraction, called Bfx_gui (Fig. 6), the user can chose the feature groups that will be extracted and the format (Matlab, Excel, or Ascii) that will be used to store the large set of extracted features. (2) For feature and classifier selection, called Bcl_gui (Fig. 7), the user can chose the feature selection algorithms to be used, the maximal number of features to be selected, the classifiers that will be evaluated, and the number of folders of the cross-validation technique. Using only these two graphic user interfaces, it is possible to design in an easy way the computer vision system automatically according to the general computer vision framework explained in the previous section.

For advanced users, however, the toolbox provides a suite of helpful commands. As illustrated in Fig. 8, a very simple program can be used to design automatically a computer vision system for an application according to the images stored in a directory (line 2). The manual labeling is performed displaying each image and asking for the corresponding class (line 5). The user can choose the features to be extracted (lines 8), the feature selection algorithms (line 13), and the classifiers (line 16) to be tested. All features for all images are extracted using only one command (line 19). The feature and classifier selection is done using again

[5]The number of folders v can be another number, for instance, 5- or 20-fold cross-validation estimates very similar performances. In our experiments, we use 10-fold cross-validation because it has become the standard method in practical terms (Witten and Frank 2005).



Geometric Features: [✓ Standard] Fourier Descriptors, Moments, Elliptical

Input Images: Format jpg; Resize

Color Features: [✓ Standard] Fourier & DCT, Contrast, Hu with intensity, Haralick, Gabor, LBP

Regions: Segmentation balu; Partition no

Color Components: Gray values, [✓ Red] Hue, L*, Green, Sat, a*, Blue, Val, b*

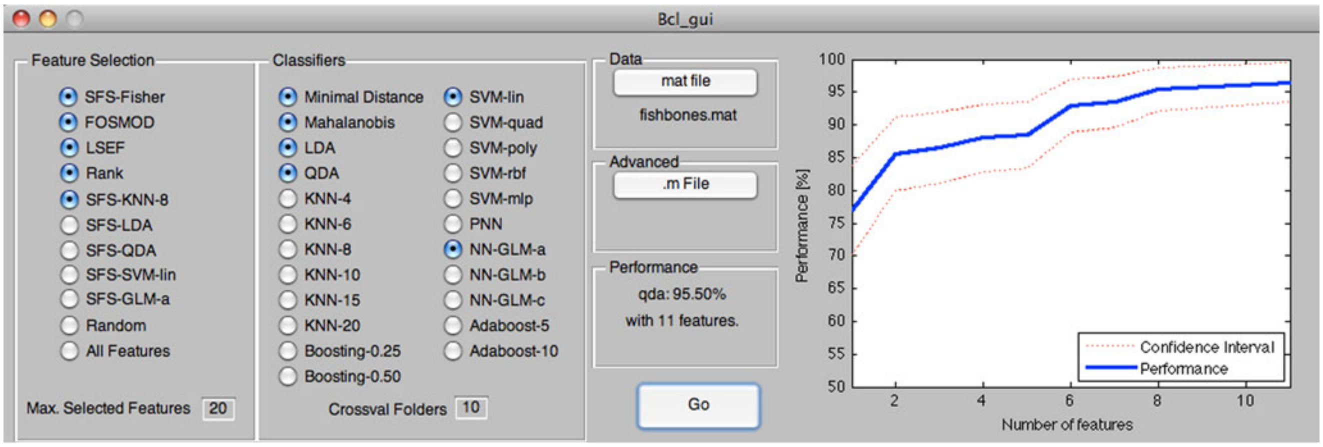
Output: [✓ Matlab] Ascii, Excel

Fig. 6 Bfx_gui user interface for feature extraction with *Balu Matlab Toolbox*: the user can select the features to be extracted (in this example standard geometric features and the standard intensity features for the red channel only)

only one command (line 24), in this example for a cross-validation with 10 folders (default) and for 20 features maximal (line 23). The selected features are given in variable fs, and the selected classifier, in variable cs. In this example, the geometric features are extracted for the segmented image using the traditional approach explained in "Data Collection" (line 10). Similarly, the intensity features are extracted for the segmented region for seven different intensity channels (line 9). As shown in this code, it is really easy to define the features and classifiers to be tested. The user can edit this file in order to add or delete other features, features selection algorithms, classifiers, and the number of features to be selected.

Three different strategies were tested on visual food data in order to compare our framework. The strategies use different combinations of (1) features, (2) feature selection algorithms, and (3) classifiers. The following three *strategies* were implemented:

- *Strategy 1—Standard*: The goal of this strategy is to test how accurate is a standard and well-known methodology in this kind of problems as shown by Inen et al. (2011). In this case, we used (1)



Feature Selection: [✓ SFS-Fisher] [✓ FOSMOD] [✓ LSEF] [✓ Rank] [✓ SFS-KNN-8] SFS-LDA, SFS-QDA, SFS-SVM-lin, SFS-GLM-a, Random, All Features

Max. Selected Features 20

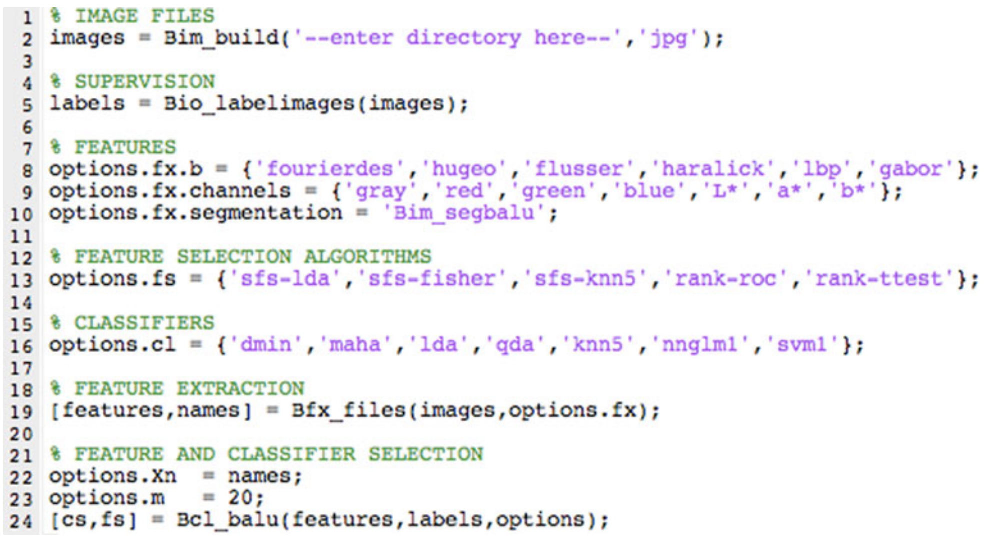
Classifiers: [✓ Minimal Distance] [✓ Mahalanobis] [✓ LDA] [✓ QDA] KNN-4, KNN-6, KNN-8, KNN-10, KNN-15, KNN-20, Boosting-0.25, Boosting-0.50, [✓ SVM-lin] SVM-quad, SVM-poly, SVM-rbf, SVM-mlp, PNN, [✓ NN-GLM-a] NN-GLM-b, NN-GLM-c, Adaboost-5, Adaboost-10

Crossval Folders 10

fishbones.mat

Performance: qda: 95.50% with 11 features.

Fig. 7 Bcl_gui user interface for feature and classifier selection with *Balu Matlab Toolbox*: the user can select the feature selection algorithms, the classifiers, the features file (computed by previous step), and the number of folders used by cross-validation technique. In this example, a performance of 95.5 % was achieved using a QDA classifier with 11 features



```
% IMAGE FILES
images = Bim_build('--enter directory here--','jpg');

% SUPERVISION
labels = Bio_labelimages(images);

% FEATURES
options.fx.b = {'fourierdes','hugeo','flusser','haralick','lbp','gabor'};
options.fx.channels = {'gray','red','green','blue','L*','a*','b*'};
options.fx.segmentation = 'Bim_segbalu';

% FEATURE SELECTION ALGORITHMS
options.fs = {'sfs-lda','sfs-fisher','sfs-knn5','rank-roc','rank-ttest'};

% CLASSIFIERS
options.cl = {'dmin','maha','lda','qda','knn5','nnglm1','svm1'};

% FEATURE EXTRACTION
[features,names] = Bfx_files(images,options.fx);

% FEATURE AND CLASSIFIER SELECTION
options.Xn   = names;
options.m    = 20;
[cs,fs] = Bcl_balu(features,labels,options);
```

Fig. 8 A simple Matlab code using *Balu* commands that performs labeling (manually), feature extraction, feature selection, and classification using the proposed framework for all files contained in a directory

local binary patterns of the original channels of the images (R, G, and B for color images and gray for gray value images), (2) no feature selection, and (3) classifiers LDA, *k*-NN with one and three neighbors, and SVM with linear and radial basis function kernels.

- *Strategy 2—Simplified*: The goal of this strategy is to test how accurate is our methodology using a simplified configuration in order to reduce the computational time. In this case, we used (1) basic geometric and intensity features and local binary patterns of RGB and gray channels for color images and gray channel only for gray value images, (2) feature selection SFS using Fisher criteria, and (3) classifiers LDA, *k*-NN with one and three neighbors, and SVM with linear and radial basis function kernels.
- *Strategy 3—Full*: The goal of this strategy is to test how accurate is our methodology using the full version. In this case, we used (1) all features explained in "Large Feature Extraction" in all available intensity channels (gray, RGB, HSV, $L^*a^*b^*$ for color images, and gray for gray value images), (2) all feature selection algorithms explained in "Feature Selection", and (3) all classifiers explained in "Classifier Selection."

In order to evaluate the strategies, the following eight *sets* of food images were used:

- *Set 1—Potato chip quality recognition*: In this set, there are 60 different color images with two different pretreatments and three different temperatures used in the production of potatoes chips. The images are large ($1,536 \times 2,048$ pixels), and they were resized to 154×205 pixels.[6] The goal is to recognize pretreatment and temperature (six classes) of the production (Pedreschi et al. 2004).
- *Set 2—Potato chip temperature recognition*: In this set, we use the images of set 1. The goal is to recognize the pretreatment (two classes) of the production (Pedreschi et al. 2004).
- *Set 3—Corn tortilla quality recognition*: In this set, there are 1,500 different color pictures with three different qualities of tortillas. The images are very large ($2,304 \times 3,072$ pixels), and they were resized to 130×173 pixels (footnote 6). The goal is to recognize the level of the production (Mery et al. 2010).
- *Set 4—Fungal decay recognition in blueberries using calyx side*: In this set, there are 120 different color pictures. The color images are small (100×100 pixels).[7] The goal is to detect fungal decay (Leiva et al. 2011).
- *Set 5—General damage recognition in blueberries using calyx side*: In this set, we use the images of set 4. The goal is to detect general damage (Leiva et al. 2011).
- *Set 6—Fungal decay recognition in blueberries using pedicel side*: In this set, there are 105 different color pictures. The color images are small (100×100 pixels).[7] The goal is to detect fungal decay (Leiva et al. 2011).
- *Set 7—General damage recognition in blueberries using pedicel side*: In this set, we use the images of

[6] In order to convert RGB images into $L^*a^*b^*$, we use a methodology based on a calibration step with known color charts (Leon et al. 2006).

[7] No $L^*a^*b^*$ information was available.

Table 2 Details of the three tests for each data set

Set	Classes C	Samples N	Size [Kpixels]	Channels[a] 1 2 3 4	Geometric[b] 1 2 3 4	Intensity[c] 1 2 3 4 5	Extracted features n	Selected features m	Feature selection	Classifier
1	6	60	350	□⧆□□	□□□□	□□⧆□□	108	108	None	k-NN-1
				⧆⧆□□	⧆□□□	⧆□⧆□□	183	20	SFS-Fisher	LDA
				⧆⧆⧆⧆	⊡⊡⊡⊡	⧆⧆⧆⧆⧆	1969	41	Rank3	SVM1
2	2	60	350	□⧆□□	□□□□	□□⧆□□	108	108	None	k-NN-1
				⧆⧆□□	⧆□□□	⧆□⧆□□	183	19	SFS-Fisher	LDA
				⧆⊡⧆⧆	⊡⊡⧆⊡	⧆⧆⧆⊡⊡	1969	10	SFS-Fisher	LDA
3	3	1,500	22.5	□⧆□□	□□□□	□□⧆□□	108	108	None	k-NN-3
				⧆⧆□□	⧆□□□	⧆□⧆□□	183	43	SFS-Fisher	SVM1
				⧆⧆⧆⧆	⊡⊡⧆⊡	⧆⧆⧆⧆⧆	1969	40	SFS-Fisher	SVM1
4	2	120	100	□⧆□□	□□□□	□□⧆□□	108	108	None	SVM1
				⧆⧆□□	⧆□□□	⧆□⧆□□	183	20	SFS-Fisher	LDA
				⧆⧆⧆□	⊡⊡⧆⊡	⊡⧆⧆⧆⊡	1393	9	SFS-Fisher	LDA
5	2	120	100	□⧆□□	□□□□	□□⧆□□	108	108	None	k-NN-1
				⧆⧆□□	⊡□□□	⧆□⧆□□	183	25	SFS-Fisher	LDA
				⧆⧆⧆□	⊡⊡⊡⊡	⊡⊡⧆⧆⊡	1393	14	SFS-Fisher	NN1
6	2	105	100	□⧆□□	□□□□	□□⧆□□	108	108	None	SVM4
				⧆⧆□□	⊡□□□	⧆□⧆□□	183	9	SFS-Fisher	LDA
				⧆⧆⧆□	⊡⊡⊡⊡	⊡⧆⧆⧆⊡	1393	7	SFS-Fisher	NN1
7	2	105	100	□⧆□□	□□□□	□□⧆□□	108	108	None	SVM4
				⧆⧆□□	⧆□□□	⧆□⧆□□	183	30	SFS-Fisher	LDA
				⧆⧆⧆□	⧆⊡⧆⊡	⊡⧆⧆⧆⊡	1393	22	SFS-Fisher	SVM1
8	2	7,697	1.6	⧆□□□	□□□□	□□⧆□□	36	36	None	k-NN-3
				⧆□□□	□□□□	⧆□⧆□□	57	14	SFS-Fisher	SVM4
				⧆□□□	□□□□	⧆⧆⧆⧆⊡	241	40	Rank5	SVM2

Each set has three rows for strategy 1 (standard), strategy 2 (simplified), and strategy 3 (full, proposed). □no extracted, ⧆extracted and selected, ⊡extracted and not selected

[a]Channels: (1) gray, (2) RGB, (3) HSV, (4) $L^*a^*b^*$

[b]Geometric features: (1) basic, (2) elliptical, (3) Fourier descriptors, (4) invariant moments

[c]Intensity features: (1) basic, (2) statistical textures, (3) local binary patterns, (4) filter banks, (5) invariant moments

set 6. The goal is to detect general damage (Leiva et al. 2011).

- *Set 8 – Salmon fish fillets quality recognition*: In this set, there are 7,700 different X-ray images with and without fish bones. The gray value images are small (10 × 10 pixels), they were resized to 40 × 40 pixels. The goal is to detect fish bones (Mery et al. 2011).

The reader can find the details of the experimental conditions of each set in the mentioned references.

Results and Discussion

In this Section, we report the results obtained on different food evaluation quality tasks using the proposed framework. The details of our strategies and sets (explained in "Experiments") are summarized in Table 2. For each set, three rows are presented, one for each strategy. In this table, we present the number of classes C, the number of samples N, the image size in thousand of pixels, the intensity channels used, the total number of extracted features n, the number of selected features m, the feature selection algorithm and the classifier chosen in the test, and the family names of the selected features (the reader is referred to Table 1 to see a description of the features). It is interesting to

Table 3 Performance of each strategy and comparison with published results

Set	Published (%)	Strategy (%)		
		1	2	3
1	90.00	63.30	78.30	95.00
2	93.00	85.00	99.10	99.30
3	99.50	81.80	96.10	99.50
4	96.00	85.00	98.30	100.00
5	90.00	84.20	95.80	97.50
6	96.00	87.00	98.00	98.70
7	90.00	71.00	93.00	97.00
8	94.70	85.80	95.30	96.80
Mean	93.65	80.39	94.23	97.98

Strategy 1 (standard), strategy 2 (simplified), and strategy 3 (full, proposed)

Table 4 Computational time in minutes for each strategy

Set ↓ Strategy →	Feature extraction			Model selection			Total		
	1	2	3	1	2	3	1	2	3
1	1.72	3.12	171.63	0.03	1.38	2.36	1.75	4.50	173.99
2	1.73	3.12	171.63	0.00	0.10	0.60	1.73	3.22	172.23
3	31.87	35.27	239.41	0.64	28.58	32.24	32.51	63.85	271.65
4	0.33	0.55	4.01	0.01	0.12	0.25	0.34	0.67	4.26
5	0.33	0.55	4.01	0.01	0.20	0.27	0.34	0.75	4.28
6	0.34	0.51	3.72	0.01	0.09	0.23	0.35	0.60	3.95
7	0.34	0.51	3.72	0.01	0.20	0.53	0.35	0.72	4.25
8	15.12	15.16	33.72	8.45	260.28	3626.00	23.57	275.44	3659.72

Strategy 1 (standard), strategy 2 (simplified), and strategy 3 (full, proposed)

observe which features are selected: for instance, we can see that geometric features are not relevant. Additionally, texture features (statistical, LBP, and filter bank) play a very important role in the classification.

The achieved performances are summarized in Table 3. As reported in the references (Pedreschi et al. 2004; Mery et al. 2010; Leiva et al. 2011; Mery et al. 2011), we have already tested on these data achieving the performance shown in the column "published" (93.65 % in average). Related to the experiments performed in this work, we observe that using the standard strategy (strategy 1), the achieved performance is poor (80.39 % in average). A better performance is achieved by simplified and full strategies (strategy 2 and strategy 3, respectively) where more features and more classifiers are available (94.23 and 97.98 % in average, respectively). Although a huge number of features are used in strategies 2 and 3, only a few number of them are selected. Additionally, using these strategies, we increase the performance of each experiment by testing more features, more feature selection algorithms, and more classifiers. We can see that in every data set, strategy 1 is able to achieve a very high performance (95 % or more).

The time computing depends on the application; however, in order to have a reference, Table 4 was presented to show the computational time on a Mac OS X 10.7.3, processor 3.06-GHz Intel Core 2 Duo, 4-GB RAM memory. The computational time is divided into three groups: feature extraction, model selection (feature and classifier selection), and total (sum of both). We observe, on the one hand, that in strategy 1 and strategy 2—depending on the amount of data—the solution can be achieved in a reasonable time (in minutes). On the other hand, strategy 3 requires hours and in the last set a couple of days; however, it is important to consider that the intensive exploration process of strategy 3 is performed offline, while at detection time, when suitable features and classifier are selected, we only need a processing load similar to the other approaches. In our experiments, strategy 2 could be chosen in terms of performance and computational time in certain sets; however, it is worth to consider strategy 3 when dealing with more complex visual data.

We could not test our framework on other data sets (used by other researchers) because they are not available. Thus, an objective comparison is very difficult; however, in order for other researchers of the community to make contributions in this field, not only all computer programs but also all images used in this papers are available in our website.

Conclusions

The results outlined in the previous section show that using a very large number of features combined with a feature and classifier selection approach allow us to achieve high classification rates on a wide variety of visual food quality evaluation tasks. This demonstrates the generality of the proposed framework. The key idea of the proposed method is to select, from a large universe of features, only those features that are relevant for the separation of the classes. We tested our method in eight different recognition problems (with two, three, and six classes) yielding a high performance of 95 % or more for every case. We believe that the proposed framework opens up new possibilities in the field of computer vision in food quality evaluation.

The disadvantage of our method could be the computational time in the feature extraction and model selection phase; however, it is important to consider that the intensive exploration process is performed once and offline, while at detection time, when suitable features and classifier are selected, we only need a processing load similar to the other approaches.

It is clear that the recent progress in computer technology allows the handling of various theoretical and experimental problems in science and technology which were inaccessible before. Currently, the processing of many large images, the use of sophisticated filters in digital image processing, the extraction of a large set of features in different color spaces, and the test of many optimization algorithms—to cite a few—are possible. However, in order to asses the performance objectively, it will be necessary to analyze a broader and public databank.

Acknowledgements This work was supported partially by grant no. R0308LAC003 from LACCIR Virtual Institute and grant no. 1070031 from FONDECYT, Chile. Blueberry images were provided by G. Leiva, G. Mondragón, and C. Arrieta from Pontificia Universidad Catolica de Chile. Corn Tortilla images were provided by J.J. Chanona-Pérez, N. Veléz-Rivera, I. Arzate-Vázquez, and G.F. Gutiérrez-López from Instituto Politécnico Nacional de México.

References

Aguilera, J., Cipriano, A., Eraña, M., Lillo, I., Mery, D., Soto, A. (2007). Computer vision for quality control in Latin American food industry, a case study. In *Int. conf. on computer vision (ICCV2007): workshop on computer vision applications for developing countries* (pp. 1–8).

Bishop, C. (2005). *Neural networks for pattern recognition*. London: Oxford University Press.

Bishop, C.M. (2006). *Pattern recognition and machine learning*. New York: Springer.

Chellappa, R., & Bagdazian, R. (1984). Fourier coding of image boundaries. *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-6*(1), 102–105.

Cheng, H., Jiang, X., Sun, Y., Wang, J. (2001). Color image segmentation: advances and prospects. *Pattern Recognition, 34*(12), 2259–2281. ISSN 0031-3203.

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Conference on computer vision and pattern recognition (CVPR2005)* (Vol. 1, pp. 886–893).

Danielsson, P.-E. (1978). A new shape factor. *Computer Graphics and Image Processing, 7*, 292–299.

Davies, E. (2000). *Image processing for the food industry*. Singapore: World Scientific.

Diaz, R., Gil, L., Serrano, C., Blasco, M., Moltó, E., Blasco, J. (2004). Comparison of three algorithms in the classification of table olives by means of computer vision. *Journal of Food Engineering, 61*(1), 101–107.

Duda, R., Hart, P., Stork, D. (2001). *Pattern classification* (2nd ed.). New York: Wiley.

Escalante, H., Montes, M., Sucar, L. (2009). Particle swarm model selection. *The Journal of Machine Learning Research, 10*, 405–440.

Fitzgibbon, A., Pilu, M., Fisher, R. (1999). Direct least square fitting ellipses. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 21*(5), 476–480.

Flusser, J., & Suk, T. (1993). Pattern recognition by affine moment invariants. *Pattern Recognition, 26*(1), 167–174.

Gonzalez, R., & Woods, R. (2008). *Digital image processing* (3rd ed.). Upper Saddle River: Prentice-Hall.

Gupta, L., & Srinath, M.D. (1987). Contour sequence moments for the classification of closed planar shapes. *Pattern Recognition, 20*(3), 267–272.

Haff, R., & Slaughter, D. (2004). Real-time X-ray inspection of wheat for infestation by the granary weevil, sitophilus granarius (l.). *Transactions of the American Society of Agricultural Engineers, 47*, 531–537.

Haff, R., & Toyofuku, N. (2008). X-ray detection of defects and contaminants in the food industry. *Sensing and Instrumentation for Food Quality and Safety, 2*(4), 262–273.

Haralick, R. (1979). Statistical and structural approaches to texture. *Proceedings of IEEE, 67*(5), 786–804.

Hastie, T., Tibshirani, R., Friedman, J. (2001). *The elements of statistical learning: Data mining, inference, and prediction* (corrected ed.). New York: Springer.

Hu, M.-K. (1962). Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory, IT*(8), 179–187.

Inen, M.P., Pietikäinen, M., Hadid, A., Zhao, G., Ahonen, T. (2011). *Computer vision using local binary patterns* (Vol. 40). London: Springer.

Irudayaraj, J., & Reh, C. (eds.) (2008). *Nondestructive testing for food quality*. Ames: IFT Press.

Jain, A., Duin, R., Mao, J. (2000). Statistical pattern recognition: a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 22*(1), 4–37.

Jiang, J., Chang, H., Wu, K., Ouyang, C., Yang, M., Yang, E., Chen, T., Lin, T. (2008). An adaptive image segmentation algorithm for X-ray quarantine inspection of selected fruits. *Computers and Electronics in Agriculture, 60*, 190–200.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *International joint conference on artificial intelligence* (Vol. 14, pp. 1137–1145).

Kumar, A., & Pang, G. (2002). Defect detection in textured materials using gabor filters. *IEEE Transactions on Industry Applications, 38*(2), 425–440.

Kwon, J., Lee, J., Kim, W. (2008). Real-time detection of foreign objects using X-ray imaging for dry food manufacturing line. In *Proceedings of IEEE international symposium on consumer electronics (ISCE 2008)* (pp. 1–4). doi:10.1109/ISCE.2008.4559552.

Leemans, V., & Destain, M.F. (2004). A real-time grading method of apples based on features extracted from defects. *Journal of Food Engineering, 61*(1), 83–89.

Leiva, G., Mondragón, G., Mery, D., Aguilera, J. (2011). The automatic sorting using image processing improves postharvest blueberries storage quality. In *Proceedings of international congress on engineering and food*.

Leon, K., Mery, D., Pedreschi, F., Leon, J. (2006). Color measurement in l*a*b* units from rgb digital images. *Food Research International, 39*(10), 1084–1091.

Mao, K. (2005). Identifying critical variables of principal components for unsupervised feature selection. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 35*(2), 339–344.

Marsland, S. (2009). *Machine learning: an algorithmic perspective*. London: Chapman & Hall.

MathWorks (2003). *Image processing toolbox for use with MATLAB: user's guide*. The MathWorks Inc.

MathWorks (2007). *Matlab toolbox of bioinformatics: user's guide*. Mathworks Inc.

MathWorks (2009). *Neural network toolbox for use with MATLAB: user's guide*. The MathWorks Inc.

Mery, D. (2011). BALU: a toolbox Matlab for computer vision, pattern recognition and image processing. http://dmery.ing.puc.cl/index.php/balu.

Mery, D., & Filbert, D. (2002). Classification of potential defects in automated inspection of aluminium castings using statistical pattern recognition. In *8th European conference on non-destructive testing (ECNDT 2002), Barcelona, 17–21 June 2002* (pp. 1–10).

Mery, D., & Pedreschi, F. (2005). Segmentation of colour food images using a robust algorithm. *Journal of Food Engineering, 66*(3), 353–360.

Mery, D., & Soto, A. (2008). Features: the more the better. In *ISCGAV'08: Proceedings of the 8th conference on signal processing, computational geometry and artificial vision* (pp. 46–51). Wisconsin: Stevens Point.

Mery, D., Chanona-Pérez, J., Soto, A., Aguilera, J., Cipriano, A., Veléz-Rivera, N., Arzate-Vázque, I., Gutiérrez-López, G. (2010). Quality classification of corn tortillas using computer vision. *Journal of Food Engineering, 101*(4), 357–364.

Mery, D., Lillo, I., Loebel, H., Riffo, V., Soto, A., Cipriano, A., Aguilera, J. (2011). Automated fish bone detection using X-ray imaging. *Journal of Food Engineering, 105*(2011), 485–492.

Mitchell, T. (1997). *Machine learning*. Boston: McGraw-Hill.

Mu, Y., Yan, S., Liu, Y., Huang, T., Zhou, B. (2008). Discriminative local binary patterns for human detection in personal album. In *IEEE conference on computer vision and pattern recognition (CVPR 2008)* (pp. 1–8)

Nixon, M., & Aguado, A. (2008). *Feature extraction and image processing* (2nd ed.). New York: Academic.

Ogawa, Y., Kondo, N., Shibusawa, S. (2003). Inside quality evaluation of fruit by X-ray image. In *2003 IEEE/ASME international conference on advanced intelligent mechatronics, 2003. AIM 2003. Proceedings* (Vol. 2, pp. 1360–1365).

Ojala, T., Pietikainen, M., Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 24*(7), 971–987.

Pedreschi, F., Mery, D., Mendoza, F., Aguilera, J. (2004). Classification of potato chips using pattern recognition. *Journal of Food Science, 69*(6), E264–E270.

Persoon, E., & Fu, K., (1977). Shape discrimination using Fourier descriptors. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-7*(3), 170–179.

Polikar, H., Long, F., Ding, C. (2005). Criteria of max-dependency, max-relevance and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 27*(8), 1226-1238.

Polikar, R. (2006). Ensemble based systems in decision making. *IEEE Circuits and Systems Magazine, 3rd Quarter*, 21–45.

Quevedo, R., Mendoza, F., Aguilera, J., Chanona, J., Gutiérrez-López, G. (2008). Determination of senescent spotting in banana (*Musa cavendish*) using fractal texture fourier image. *Journal of Food Engineering, 84*(4), 509–515.

Shawe-Taylor, J., & Cristianini, N. (2004). *Kernel methods for pattern analysis*. Cambridge: Cambridge University Press.

Sun, D. (2000). Inspecting pizza topping percentage and distribution by a computer vision method. *Journal of Food Engineering, 44*(4), 245–249.

Sun, D.-W. (ed.) (2008). *Computer vision technology for food quality evaluation*. New York: Academic.

Szeliski, R. (2011). *Computer vision: algorithms and applications*. New York: Springer.

Tan, J. (2004). Meat quality evaluation by computer vision. *Journal of Food Engineering, 61*(1), 27–35.

Viola, P., & Jones, M. (2004). Robust real-time object detection. *International Journal of Computer Vision, 57*(2), 137–154.

Webb, A. (2005). *Statistical pattern recognition*. England: Wiley.

Wei, H.-L., & Billings, S. (2007). Feature subset selection and ranking for data dimensionality reduction. IEEE Transactions on Pattern Analysis and Machine Intelligence, 29(1), 162–166, Jan.

Witten, I., & Frank, E., (2005). *Data mining: practical machine learning tools and techniques* (2nd. ed.). San Mateo, CA: Morgan Kaufmann.

Zahn, C., & Roskies, R., (1971). Fourier descriptors for plane closed curves. *IEEE Transactions on Computers, C-21*(3), 269–281.