Basics of Image Analysis

Chapter · January 2015

DOI: 10.1007/978-1-4939-2836-1_2

Fernando A Mendoza
AB Electrolux

Chapter 2
Basics of Image Analysis

Fernando Mendoza and Renfu Lu

2.1 Introduction

Image analysis is used as a fundamental tool for recognizing, differentiating, and quantifying diverse types of images, including grayscale and color images, multispectral images for a few discrete spectral channels or wavebands (normally less than 10), and hyperspectral images with a sequence of contiguous wavebands covering a specific spectral region (e.g., visible and near-infrared). Earlier works on image analysis were primarily confined to the computer science community, and they mainly dealt with simple images for such applications as defect detection, segmentation and classification. Nowadays, image analysis is becoming increasingly important and widespread because it can be done more conveniently, rapidly and cost effectively (Prats-Montalbán et al. 2011). Image analysis relies heavily on machine vision technology (Aguilera and Stanley 1999). The explosive growth in both hardware platforms and software frameworks has led to significant advances in the analysis of digital images.

Image analysis has been applied to many different fields of science and technology. For example, it has been used to assess or quantify the external characteristics (i.e., color, size, shape and surface texture) and internal structures (architecture and/or connectivity of the material constituents) of food products.

F. Mendoza
U.S. Department of Agriculture, Agricultural Research Service, Sugarbeet and Bean Research Unit, 524 S. Shaw Lane, Room 105A, East Lansing, MI 48824, USA

Department of Biosystems and Agricultural Engineering, Michigan State University, East Lansing, MI, USA
e-mail: fmendoza@msu.edu

R. Lu (✉)
U.S. Department of Agriculture, Agricultural Research Service, East Lansing, MI, USA
e-mail: renfu.lu@ars.usda.gov

B. Park, R. Lu (eds.), *Hyperspectral Imaging Technology in Food and Agriculture*, Food Engineering Series, DOI 10.1007/978-1-4939-2836-1_2

Commercial machine vision units are readily available to meet the requirements of automatic inspection for the food processing and packaging industries. As consumers are demanding better quality and safer food products, there is an increasing need for rapid and non-destructive quality evaluation of foods. In recent years, new imaging-based inspection techniques, such as multispectral and hyperspectral imaging, have been developed for quality assessment of a variety of foods, which have overcome some of the drawbacks of traditional human and instrumental inspection techniques (Du and Sun 2007). These methods, which are based on the automatic detection of various image features, correlate well with quality attributes of foods that are related to the sensorial, chemical, and physical properties (Valous et al. 2009a).

It is important to note that image analysis is part of a wider field known as image processing, where the main underlying idea is to improve the visual quality of an image and/or to extract useful information or features. The analysis is based on different image properties such as color, gloss, morphology of the objects, and texture. Image processing actions can be grouped into three sub-areas (Prats-Montalbán et al. 2011):

(a) *Image compression*, which reduces the memory requirements by removing the redundancy present in the image, that is, the image information which is not perceptible to the human eye.
(b) *Image preprocessing*, which consists of improving the visual quality of the image by reducing noise, pixel calibration and standardization, enhancing the edge detection, and making the image analysis step more reliable based on objective and well established criteria. The term image preprocessing, in general, is referred to all manipulations on an image, each of which produces a new image.
(c) *Image analysis*, which usually returns numeric values and/or graphical information about the image characteristics that are suited for classification, defect detection, or prediction of some of the quality properties of the imaged object. The term image analysis is used when the output is a number or decision, not an image.

These processing actions are related and may have a different effect or output for each application. The following sections describe basic concepts and characteristics of a digital image and how they are processed and transformed (improved), give an overview of typical image analysis methods and techniques, and show some application examples in food quality assessment and control.

2.2 The Digital Image

A *digital image* is a numerical 2D or 3D representation of a real physical object or scene, from which we can obtain an accurate spatial (geometric) and/or spectral (for the case of a hyperspectral image) representation with sufficient detail (resolution) for processing, compression, storage, printing and display. A *digital image* may be of *vector* or *raster* type, depending on whether or not the image resolution is fixed (Wikipedia 2012). Raster images are electronic files that consist of discrete picture elements, called *pixels* (short for picture elements). Associated with each pixel is a number that is the average radiance (or *brightness*) of a relatively small area within a scene, representing the color or gray-level at a single point in the image.

Vector graphics formats are complementary to raster graphics, which are the representation of images based on mathematical expressions, as are typically used in computer graphics for images made up of vectors (arrows of direction, points, and lines) that define shapes, as compared to the individual pixels used to represent a raster image. A vector image is resolution independent, which means the image can be enlarged or shrunken without affecting the output quality. The term *digital images* usually refers to raster images, which are also called bitmap images. Moreover, it is also applied to data associated to points scattered over a three-dimensional region, such as produced by tomographic equipment. In that case, each datum is called a *voxel*.

The most commonly used method for the creation of raster images (i.e., digital imaging or image acquisition) is digital photography with a CCD camera in a process called *digitization*. The digitization process requires the mapping of the image on a grid and a quantization of the intensity level. Digitization is the process of converting an analog signal into a digital signal, known as an A/D (analog to digital) conversion. For raster images, an analog voltage signal (from any of several types of imaging sensors), proportional to the amount of light reflected or transmitted by an item being digitized, is divided into discrete numeric values (Castleman 1979). In other words, the process converts an image into a series of small picture square elements or *pixels* that are either black or white (binary), a specific shade of gray (grayscale) or color. Each pixel is represented by a single or series of binary digits, either 1 s or 0 s. By measuring the color (or gray-level for black and white photos) of an image at a large number of points, we can create a digital approximation of the image from which a copy of the original can be reconstructed.

Figure 2.1 depicts an example of the process for creation of a digital image. This figure shows the energy from an illumination source being reflected from a

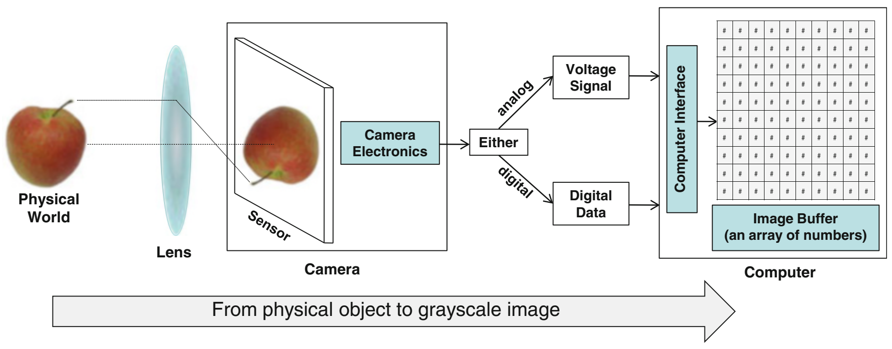


Fig. 2.1 An example of the process for creation of a digital image

scene element (apple). The imaging system collects the incoming energy and focuses it onto an image plane. The front end of the imaging system is an optical lens that projects the viewed scene onto the lens focal plane. The camera sensor is an array of photo cells which measure light at discrete points. The sensor does not directly recognize the color of the incoming light; instead, in a color camera (with 3 CCD sensors) a prism separates the light into three components or color channels i.e., R (for red), G (for green) and B (for blue). The response of each sensor, which is coincident with the focal plane, is proportional to the integral of the light energy projected onto the surface of the sensor. Once a digital image has been created and stored in any media, there is a corresponding digital to analog conversion that allows the computer to present the image in a human readable form on either a display or printer. Displaying an image on a computer monitor or printing the image on a printer are both examples of an analog representation of a digital image. In order to process and analyze images on a computer we first have to digitize them.

Figure 2.2 emphasizes the characteristics of a digital color image. The displayed color image is a two-dimensional matrix of thousands or millions of pixels, each of

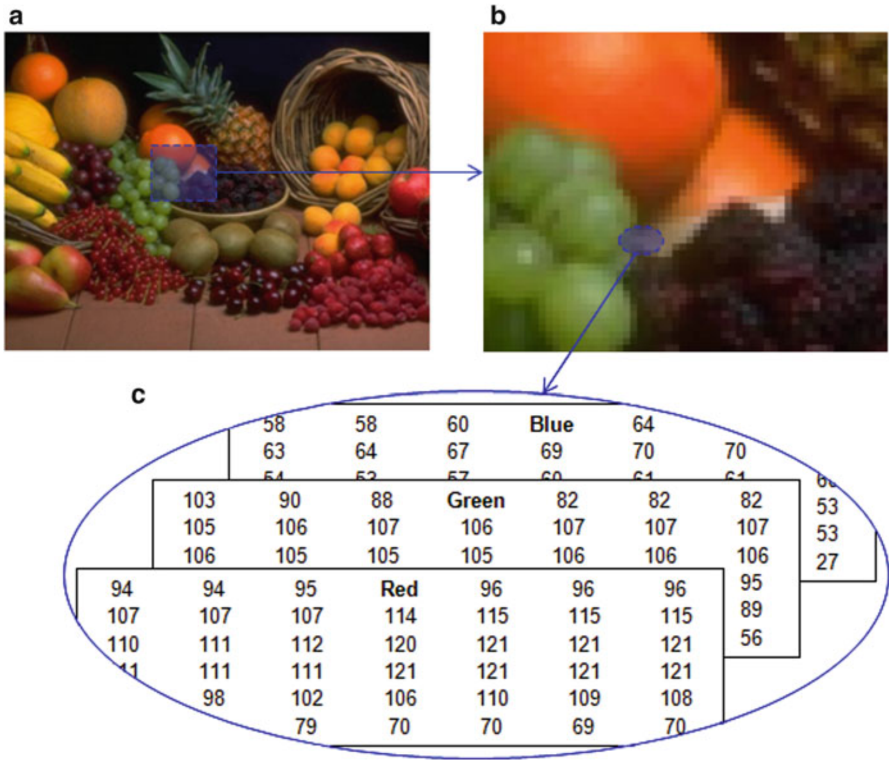


Fig. 2.2 Characteristics of a digital color image: (**a**) original image (340 × 256 pixels), (**b**) magnified color image (×1,600) to visualize the pixels, (**c**) the numerical R, G, and B color intensity values for a selected region

which has its own address, size, and grayscale or color channel representation. By zooming in on this digital image, it is noticed that the image is composed of a series of rows and columns of square pixels. Each pixel represents the brightness or intensity value of a given color channel (or gray-level) at any single specific point in the image. Typically, the pixel information is stored in computer memory as a raster image or raster map, a 2D array of integers. These values are often transmitted or stored in a compressed form.

Constraints of the digitized representation are that it contains much less information of the original scene since a 3D scene is reduced to a 2D representation. The sensors in any acquisition device (still or video cameras, scanners among other sensors) in general are not capable to capture and reproduce exactly, although not sensitive to the human vision, all color information from the real scene. Moreover, the size and location of the objects on the image are now estimates, whose precision and accuracy are dependent on the sampling resolution. The advantages of digital images are that they can be processed and analyzed, in many ways, by computers.

2.2.1 Basic Image Measurements

There are three basic measures of every static digital image: *spatial resolution*, *pixel bit depth*, and *color* (Puglia 2000). The specifications selected for each measure determine the amount of electronic information captured to represent the original photograph or scene. Generally, the higher the values are within these measures, the more data will be captured representing a greater amount of photographic detail from the original.

2.2.1.1 Resolution

Spatial resolution is defined as the rate, or number of times, at which an image is sampled during the acquisition or imaging process. More specifically, it is the frequency of pixels used to capture sample shades in the space of the object being digitized. *Spatial frequency* is synonymous for *spatial resolution*. Generally, more pixels per unit dimension means a higher resolution, but the overall image quality cannot be determined by spatial resolution alone. Typical array sizes in pixels (or *pixel resolution*) in many imaging sensors vary from 640×480 to $2{,}048 \times 1{,}536$ pixels. For reference human vision is >100 million pixels.

Quantitatively, *spatial resolution* (i.e., the number of pixels in an image) can be described in a number of ways, being the most common measures: *dpi* (dots per inch), *ppi* (pixels per inch), and *lpi* (line pairs per inch) (Gonzales and Woods 2008). The resolution of a digital image file is most appropriately referred to as pixels per inch or *ppi*. The *dpi* and *lpi* measures are considered printing terms and are most appropriate when referring to the resolution at which a computer

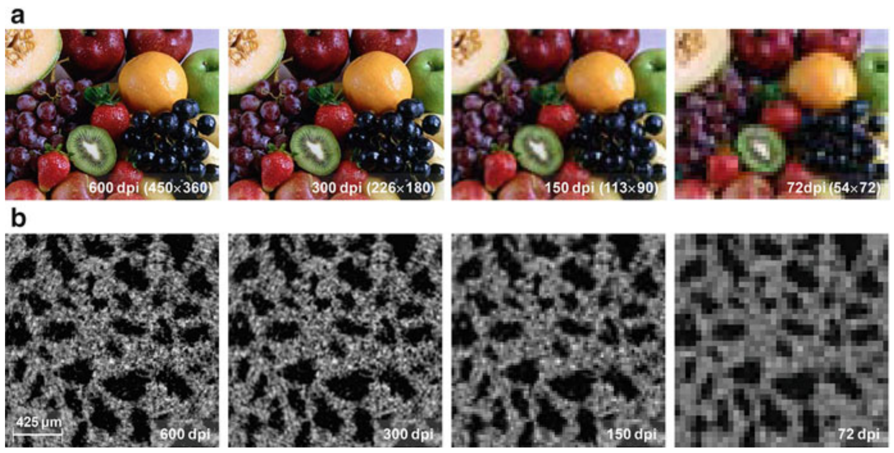


Fig. 2.3 Images saved at different *dpi* levels while keeping the same size. (**a**) typical effects on a color image; (**b**) X-ray microtomography image of 'Jonagold' apple captured with a spatial resolution of 8.5 μm/pixel, where black areas represent pores and gray areas the cellular material

printer produces a print. However, *dpi* is a more generic term and is more commonly used than *ppi* in image reproduction and photography.

In science and engineering research, however, the *dpi* term is not frequently reported when referring to the resolution of images. Generally, they are expressed by a spatial or unit distance per pixel, i.e., mm/pixel, μm/pixel, etc., and sometimes represented by a horizontal bar on the same image indicating the actual spatial length of that bar (as shown in Fig. 2.3b). Figure 2.3 presents two examples of images at different *dpi* levels while keeping their size the same. Reducing the resolution of the original image generally leads to larger pixels and less detail in the image. Images saved with the lowest *dpi* levels appear blurred and have a reduced contrast. By decreasing the image resolution of the apple tissue (Fig. 2.3b) the finest details of the tissue microstructure are lost, making it more difficult for the visualization and analytical determination of the smallest structures.

2.2.1.2 Pixel Bit Depth

This measure is sometimes called *pixel depth* or *color depth*. This defines the number of shades that can actually be represented by the amount of information saved for each pixel. Computers work on a binary system; each bit of data is either 1 or 0. Each pixel in a raster image is represented by a string of binary digits and the number of digits is known as the *bit depth*. Hence, a one-bit image can assign only one of two values to a single pixel: 0 or 1 (black or white). An 8-bit (2^8) grayscale image can assign one of 256 colors to a single pixel. A 24-bit ($2^{(3\times8)}$) *RGB* image (8-bits each for red, green and blue color channels) can assign one of 16.8 million colors to a single pixel. The *bit depth* determines the number of possible combinations of 1 s and 0 s for that number of binary digits and therefore the number of gray shades or color shades that can be represented by each pixel. This is calculated by the following formula (Puglia 2000):

$$\textit{Number of Shades} = 2^x, \text{ where } x = \text{ the bit depth.} \tag{2.1}$$

The *bit depth* influences the representation of images. The greater the *bit depth*, the finer the levels of change can be recorded. Consequently, more space is needed in the computer system to handle and store the image. Although grayscale images with bit depths of 2, 4, 6, 12, 16 and 32 exist, 8 *bpp* (i.e., byte-per-pixel) grayscale images are the most common. This is for two reasons; first, the 8 *bpp* size makes it easier to manipulate with a computer, and second, since the human eye can distinguish less than 200 shades, it can faithfully represent any grayscale image because it provides 256 distinct levels of gray. Thus, the use of 8-bit grayscale image files and corresponding 24-bit *RGB* color image files, in many cases, represent a reasonable compromise for image processing and analysis purposes. Some scientific applications requiring wider or higher dynamic ranges, as in multi- and hyper-spectral imaging, frequently use camera sensors with 14-bit (16,384 shades) or 16-bit (65,536 shades) in order to reduce the noise level for a higher signal-noise-ratio.

To achieve a desired *bit depth* without any data loss, it is necessary to digitize a photograph at a higher *bit depth* and then scale down to the desired *bit depth* after any image processing has occurred. In addition to the loss of data from small fluctuations in the acquisition system, raw digital images often require minimal processing (e.g., sharpening or minimal tonal corrections). Any processing of a digital image results in some data loss. Acquiring and processing an image at a higher *bit depth* and then reducing to the desired *bit depth* will minimize the impact of the data loss and provide a file with the desired quality.

2.2.1.3 Color Representation

Several different systems are used to represent color images. The most common are *RGB* (additive color system), *CMYK* (subtractive color system), *HSV* and the *CIELAB* color space. A color space is a specific implementation of a color model. There are many *RGB* and *CMYK* color spaces defined for specific purposes and applications (such as *sRGB* and *CMY*). The terms color space and color profile can often be used interchangeably, since a profile is a description or numerical model of a specific color space.

Operating systems and programs need to have access to an embedded profile that describes the meaning of the color values in order to interpret the color correctly. There are two types of profiles: *matrix-based* and *table-based*. Matrix-based profiles use mathematical formulas to describe the 3D color spaces. They can be relatively small and most appropriate as working spaces and as embedded profiles. Table-based profiles, as the name implies, use a large table of sample points called a

Look-Up Table or *LUT* to define the 3D color space. These profiles are more customizable, and are therefore more useful when translating color information from one space to another, or in describing the color characteristics of a particular device. Because they rely on many data points, they are much larger.

RGB is a color model that uses the three primary (red, green, blue) additive colors, which can be mixed to make all other colors. It builds its model by adding different colors of light together, where the mixture of all three colors produces white light. Grayscale values follow the line from black (the origin of the coordinate system) to white, as shown in Fig. 2.4a. Digital cameras produce *RGB* images, and monitors display *RGB* images. Mathematical conversions between different color spaces for analysis and special visualizations are also possible.

CMYK (*CMY*) is a color model based on subtracting light. The cyan (*C*), magenta (*M*), and yellow (*Y*) are the basic colors for a subtractive model, and represent the complements of the three primary colors (see Fig. 2.4a). *R*, *B*, *G*, and black (*K*) inks are used in most commercial color printing (books, magazines, etc.). Inks absorb colored light, which is why the model is called a subtractive one. *CMYK* is commonly referred to as process color, and there are many individual color spaces that use the *CMYK* color model. Conversions from *RGB* to *CMY* are done by using the following simple equation (Gonzales and Woods 2008):

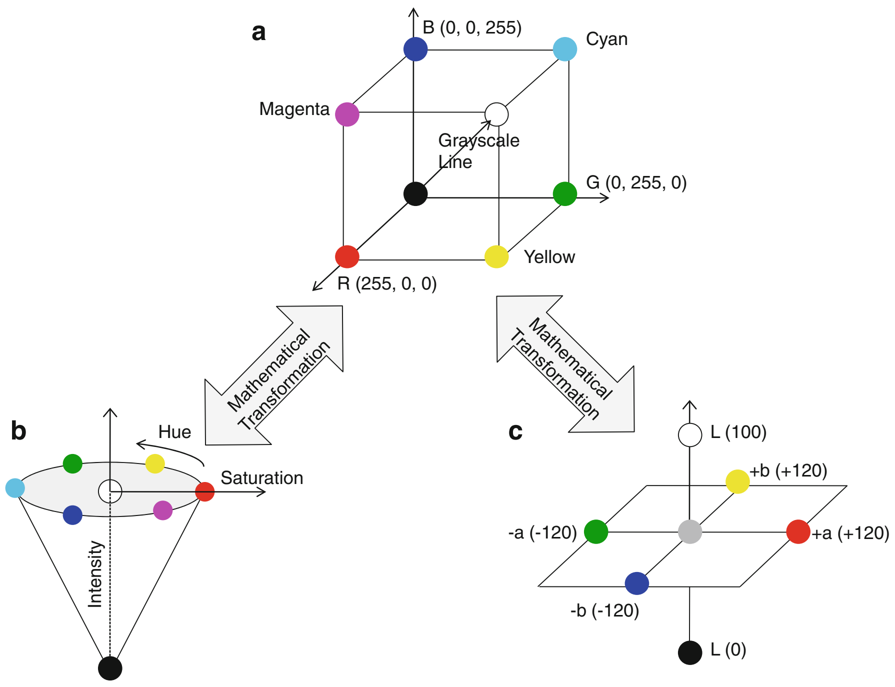


Fig. 2.4 Commonly used color spaces: (**a**) *RGB* color cube; (**b**) *HSV* color cone; and (**c**) CIELAB or $L^*a^*b^*$ color system

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1.0 \\ 1.0 \\ 1.0 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{2.2}$$

If a color image has to be printed, black (K) as a fourth color is added to the model to achieve a purer black than the simple combination of the other three colors, resulting in the *CMYK* model. Transformation from *CMY* to *CMYK* is done by:

$$K = min(C_{CMY}, M_{CMY}, Y_{CMY}) \tag{2.3}$$

$$C_{CMYK} = C_{CMY} - K \tag{2.4}$$

$$M_{CMYK} = M_{CMY} - K \tag{2.5}$$

$$Y_{CMYK} = Y_{CMY} - K \tag{2.6}$$

HSV is a user-oriented color model based on the artist's idea of tint, shade and tone. *HSV* expresses color into three components that vary from 0 to 1; H (hue) distinguishes among the perceived colors, such as red, yellow, green and blue, S (saturation) refers to how much of the light is concentrated at each specific wavelength of the hue; and V (value) represents the total brightness (see Fig. 2.4b). The computation of H, S, and V values from the *RGB* color space is made according to the following expressions (Du and Sun 2005):

$$V = max(nR, nG, nB) \tag{2.7}$$

$$S = \frac{V - min(nR, nG, nB)}{V} \tag{2.8}$$

Let

$$tR = \frac{V - nR}{V - min(nR, nG, nB)}, \tag{2.9}$$

$$tG = \frac{V - nG}{V - min(nR, nG, nB)}, \tag{2.10}$$

$$tB = \frac{V - nB}{V - min(nR, nG, nB)}, \tag{2.11}$$

Then

$$6H = \begin{cases} 5 + tB & \text{if } nR = max(nR, nG, nB) \text{ and } nG = min(nR, nG, nB) \\ 1 - tG & \text{if } nR = max(nR, nG, nB) \text{ and } nG \neq min(nR, nG, nB) \\ 1 + tR & \text{if } nG = max(nR, nG, nB) \text{ and } nB = min(nR, nG, nB) \\ 3 - tB & \text{if } nG = max(nR, nG, nB) \text{ and } nB \neq min(nR, nG, nB) \\ 3 + tG & \text{if } nB = max(nR, nG, nB) \text{ and } nR = min(nR, nG, nB) \\ 5 - tR & \text{otherwise} \end{cases} \tag{2.12}$$

where $H, S, V \in [0, \ \ldots, 1]$.

In food research, color is frequently represented using the *CIELAB* or $L^*a^*b^*$ color space since results closely match those of the human perception. L^* is the luminance or lightness component that goes from 0 (black) to 100 (white), and parameters a^* (from green to red) and b^* (from blue to yellow) are the two chromatic components, varying from −120 to +120 (see Fig. 2.4c). The definition of $L^*a^*b^*$ is based on the intermediate system *CIE XYZ* which is derived from *RGB* (Rec. ITU-R BT.709-5, 2002). Thus, L^*, a^*, and b^* are defined as:

$$L^* = 116\left(\frac{Y}{Y_n}\right)^{1/3} - 16 \tag{2.13}$$

$$a^* = 500\left[\left(\frac{X}{X_n}\right)^{1/3} - \left(\frac{Y}{Y_n}\right)^{1/3}\right] \tag{2.14}$$

$$b^* = 200\left[\left(\frac{Y}{Y_n}\right)^{1/3} - \left(\frac{Z}{Z_n}\right)^{1/3}\right] \tag{2.15}$$

where X_n, Y_n, Z_n correspond to the *XYZ* values of a reference white chart.

2.2.2 Types of Files

File types are used to encode digital images, allowing for compression and storage. Image files can be of different sizes and formats, and larger file types mean more disk usage and slower download. Compression is a term used to describe ways of reducing the size of the file. Compression schemes can by *lossy* or *lossless* (Shapiro and Stockman 2001). A *lossless* compression algorithm does not discard information. It looks for more efficient ways to represent an image, while making no compromises in accuracy. A *lossless* algorithm might, for example, look for a recurring pattern in the file, and replace each occurrence with a short abbreviation, thereby cutting the file size.

Contrarily, a *lossy* algorithm might accept some degradation in the image in order to achieve smaller file size. For example, *JPEG* or *JPG* file format works by analyzing images and discarding information that the eye is least likely to notice. Changes in luminance are more significant by the human observer than change in hue. Thus, in *JPEG* compression, factors of more than 20 are often quite acceptable. Better graphics programs, such as *Paint Shop Pro* and *Photoshop*, allow viewing the image quality and file size as a function of compression level, so that it is possible to conveniently choose the balance between qualities and file size. Higher *JPEG* compression equals lower image quality since the color information in individual pixels is compressed into *blocks* of pixels using mathematical algorithms that methodically blend all the pixel colors in each block. Increasing the compression produces smaller computer file sizes, whereas lower compression produces better quality but larger computer file sizes.

The most common digital image file types and their main characteristics are summarized in Table 2.1. Currently, *GIF* and *JPEG* are the formats used for nearly all web images. *PNG* is supported by most of the latest generation browsers. *TIFF* is not widely supported by web browsers, and should be avoided for web use. *PNG* does everything *GIF* does, and better; so it is expected to replace *GIF* in the future. *PNG* will not replace *JPEG*, since *JPEG* is capable of much greater compression of photographic images, even when set for quite minimal loss of quality.

Table 2.1 Common digital file types

File type		Description
TIFF	Tagged Image File Format	• *Lossless* uncompressed file format with 24 or 48 bit color support. File sizes are quite big
		• Supports embedded information like EXIF[a], calibrated color space and output profile information
		• There is a lossless compression for *TIFF* called LZW. LZW works like *zipping* the image file because there is no quality loss. An LZW *TIFF* decompresses with all of the original pixel information unaltered
PNG	Portable Network Graphics	• *Lossless* compression format with up to 16-bit depth for grayscale values and up to 48-bits for color values
		• In contrast with common *TIFF* files, it looks for patterns in the image that it can use to compress file size
		• The compression is exactly reversible, so the image is recovered exactly
JPEG	Joint Photographic Experts Group	• *Lossy* compression format optimized for photographs that contain many colors which store information as 24-bit color and the degree of compression is adjustable
		• *JPEG* compression ratio and resolution are not relational. It is possible to get a high resolution JPEG image with either low or high compression
		• Supports embedded information like EXIF, calibrated color space and output profile information
		• This is the best format for photographs to be shown on the web or as email attachments

(continued)

Table 2.1 (continued)

File type		Description
GIF	Graphics Interchange Format	• *Lossless* uncompressed 8-bit file format that supports only 256 distinct colors. *GIF* creates a table of up to 256 colors from a pool of 16 million
		• Not suitable for photographs because of its limited color support. Best used with web clip art and logo type images
BMP	Bitmap Format	• *Lossless* uncompressed file format that supports 24-bit color invented by Microsoft
		• Does not support embedded information like EXIF, calibrated color space and output profiles
		• *BMP* produces approximately the same file sizes as *TIFF* without any of the advantages of *TIFF*
RAW		• *Lossless* compressed file format that is proprietary for each digital camera manufacturer and model
		• Though *lossless*, it is a factor of three or four smaller than *TIFF* files of the same image
		• It contains the full range of color information from the sensor. RAW files converted to 16-bit *TIFF* produce the absolute best quality image available from any digital camera
		• Camera *RAW* supports imbedded EXIF data
PSD, PSP	Photoshop file, Pain Shop Pro	• Proprietary formats used by graphic programs
		• Preferred working formats as you edit images in the software, because only the proprietary formats retain all the editing power of the programs
		• These packages use layers building complex images, and layer information may be lost in the nonproprietary formats such as *TIFF* and *JPG*

[a]EXIF stands for Exchangeable Image File Format. This data includes technical information about each photograph including calibrated color space, output profile device information shutter speed and aperture used, whether or not flash was used, and the date the photo was taken

2.2.3 Types of Digital Images

As described above, each pixel of a *raster* image is typically associated to a specific position in the 2D region, and has a value consisting of one or more quantities (samples) related to that position. Digital images can be classified according to the number and nature of those samples in: *binary*, *grayscale*, *color*, *false-color*, *multi-spectral*, *thematic*, and *picture function* (Shapiro and Stockman 2001).

For image processing and analysis, the input image is supposed to be grayscale or *RGB*. However, there are four basic types of digital images frequently used as intermediate steps in the processing of an image, which allow to identify, enhance, quantify and also represent specific characteristics or regions of interest on an image. They are: *indexed color images*, *intensity images*, *binary*, and *labeled* images.

2.2.3.1 Indexed Color Images

A digital color image is an image that includes the *R*, *G*, and *B* color channel information or other color representation for each pixel. An *indexed color* image consists of two arrays: an *image matrix* and *colormap*. The *colormap* (also called a *palette*) is an ordered set of values that represent the colors in the image. For each image pixel, the *image matrix* contains a value that is an index into the *colormap*. In computing, the encoding of the color image data in this way allows for saving computer memory and file storage, while speeding up refreshing and file transfers. It is a form of vector quantization compression.

2.2.3.2 Intensity Images

An *intensity image* or *digital grayscale image* is an image in which the value of each pixel is a single sample, that is, it carries only intensity information. These images are composed exclusively of various shades of gray, varying within a given range from 0 (for black) to 1 (or 255 for white), with any fractional values in between. However, it must be noted that this does not define what black or white is in terms of colorimetry. Another convention is to employ percentages, so the scale is then from 0 to 100 %. This is used for a more intuitive approach, but if only integer values are used, the range encompasses a total of only 101 intensities, which are insufficient to represent a broad gradient of grays.

Grayscale images are often the result of measuring the intensity of light at each pixel in a single band of the electromagnetic spectrum (e.g., infrared, visible light, ultraviolet, etc.), and in such cases they are monochromatic when only a given frequency is captured. Multispectral or hyperspectral images have numerous bands or a finer spectral resolution, and they are examples of this type of intensity images. But they also can be synthesized from a full color image by converting to grayscale. Grayscale images are also called monochromatic because of the absence of any chromatic variation (i.e., one color).

If a color image has to be converted into an *intensity* or *grayscale* image, the following equations can be used. One alternative is the simple average of the *R*, *G*, *B* color channels:

$$I = 0.333 \cdot R + 0.333 \cdot G + 0.333 \cdot B \tag{2.16}$$

Another equation, which takes into account the luminance perception of the human eye, is

$$Y = 0.2162 \cdot R + 0.7152 \cdot G + 0.0722 \cdot B \tag{2.17}$$

The weights used to compute luminance are related to the monitor's phosphors. The explanation of these weights is due to that for equal amounts of color, the eye is more sensitive to green, then red, and then blue. This means that for equal amounts of green and blue light, the green will, nevertheless, be much brighter. Thus the image obtained by the normal averaging of an image's three color channels produces a grayscale brightness that is not perceptually equivalent to the brightness of the original color image. The weighted sum that defines Y (Eq. 2.17), however, does.

2.2.3.3 Binary

A *binary image* has only two possible values (0 and 1) for each pixel. Typically the two colors used for a binary image are black and white, though any two colors can be used. The color used for the object(s) in the image is the foreground color while the rest of the image is the background color. Thus, the set of all white pixels in a binary image is a complete morphological description of the image (Gonzales and Woods 2008). Binary images are also called *one-bit*, *bi-tonal*, *bi-level* or *two-level*. This means that each pixel is stored as a single bit (0 or 1, see Sect. 2.2.1.2). In digital image processing, binary images often arise as masks or result from certain operations such as segmentation, thresholding, and dithering. A binary image is usually stored in memory as a bitmap, a packed array of bits (Wikipedia 2012).

2.2.3.4 Labeled Images

A *labeled image* is a digital image whose pixel values are symbols from a finite alphabet. The symbol value of a pixel denotes the outcome of some decision made for that pixel. An example is the labeling of objects in a *binary image*, which means that these objects are classified and numbered. Related concepts are *thematic image*, *false-color image* and *pseudo-colored image*. In a *false-color* image this close correspondence between subject color and image color is altered. The term *false-color* is typically used to describe images whose colors represent measured intensities outside the visible portion of the electromagnetic spectrum. *Pseudo-coloring* can make some details more visible, by increasing the distance in color space between successive gray levels. *Pseudo-color* images differ from *false-color* images in that they are made from only one original grayscale image, rather than two or three.

2.3 Steps for Image Processing and Analysis

Image processing involves a series of image operations to enhance the quality of a digital image so as to remove defects such as geometric distortion, improper focus, repetitive noise, non-uniform lighting and camera motion. *Image analysis* is the process of distinguishing the objects (regions of interest or ROIs) from the background and producing quantitative information, which is subsequently used for decision making. Processing and analysis can be performed on many different types of image data. These include, in an increasing order of complexity: binary images, grayscale, color, polarized-light, multi-spectral and hyper-spectral, 3D images, multi-sensor and multimedia systems, and image sequences and video.

Gunasekaran and Ding (1994) defined three levels of image processing, named, *low level processing* which includes image acquisition and pre-processing of images, *intermediate level processing* which involves image segmentation, image representation and description, and *high level processing* which involves recognition of ROIs and interpretation for quality sorting and grading. The terms *machine vision* or *computer vision* is often used for the entire subject, including *image processing* and *analysis* and *pattern recognition* techniques. Hence, the process of making a decision involves a number of steps in sequential order. Not all situations require all of these steps or operations, but all are potentially available to deal with particular problems.

Machine vision generally consists of the following five steps or operations (Fig. 2.5): (1) *image acquisition* operations to convert images into digital form, as explained in Sect. 2.2; (2) *pre-processing* operations to obtain an improved image with the same dimensions as the original image; (3) *image segmentation* operations to partition a digital image into disjoint and non-overlapping regions; (4) *object measurement* operations to measure the characteristics of objects, such as size, shape, color and texture; and (5) *classification* or *sorting* operations to identify objects by classifying them into different groups.

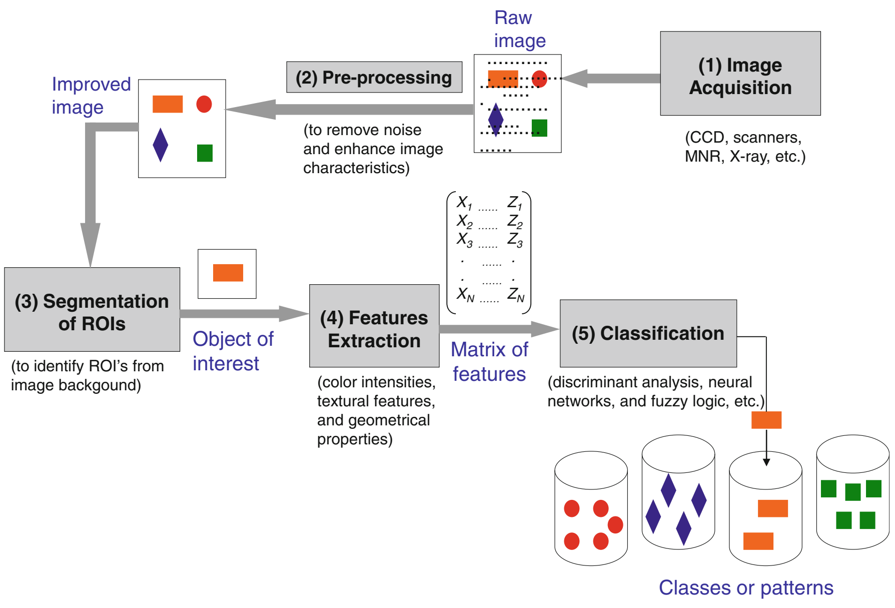


Fig. 2.5 An overview of the operational steps for a machine vision system

2.4 Image Processing

Image processing or *pre-processing* encompasses a broad range of operations, which may be treated as an end in themselves, or are intended to simplify or enhance subsequent analysis. *Pre-processing* improves the image data by removing unintended distortions or enhancing some image features that are important for further processing and creating a more suitable image than the original for a specific application. The operations that can be performed on digital images include *point*, *local* or *neighborhood*, and *global operations*.

Point operations transform pixels without regard to neighboring pixels. The gray value of the output image at a particular pixel depends only on the gray value of the same pixel in the input image. They map the pixels in one image to form another using a single mapping function. Point operations do not consider the spatial organization of the image, which forms the fundamental character of images as opposed to other types of data. Examples of these operations include contrast stretching, segmentation based on gray value, and histogram equalization (Marchant 2006). The term *contrast* refers to the amplitude of gray level variations within an image.

A *local* or *neighborhood operation* or *mask operation* generates an output pixel whose value depends on the pixel values in a neighborhood of the corresponding input point. Examples include convolution (as for image smoothing or sharpening) and spatial features detection (e.g., line, edge, and corner detection). A large and powerful class of non-linear neighborhood operations is morphological methods; they extend naturally to gray-level (and multiband) images (Soille 1999).

Finally, an operation is a *global operation* if the output value at specific coordinate is dependent on all the values in the input images. Spatial domain processing methods include all three types, but frequency domain operations, by nature of the frequency (and sequence) transforms, are global operations. Of course, frequency domain operations can become mask operations, based only on a local neighborhood, by performing the transform on small image blocks instead of the entire image. This section presents a brief description of the types of algorithms commonly utilized for digital image processing. We intentionally limit the discussion to the types of image processing algorithms that are widely used in applications for foods.

2.4.1 Grayscale Operations for Image Enhancement

Once the grayscale or color image is obtained, one of several different techniques may be used to improve the quality of the image. Image enhancement techniques are used to emphasize and sharpen image features for further analysis in order to facilitate the development of a solution to a specific application problem. Consequently, the enhancement methods are application specific and are often developed empirically.

2.4.1.1 Arithmetic Operations

All arithmetic operations performed on matrices may be performed on images. Arithmetic operations between images are array operations carried out between corresponding pixel pairs. Hence, the images normally have to be of the same size. These operators are frequently used for reducing noise and image enhancement. The four arithmetic operations are as follows:

$$s(x, y) = f(x, y) + g(x, y) \tag{2.18}$$

$$d(x, y) = f(x, y) - g(x, y) \tag{2.19}$$

$$p(x, y) = f(x, y) \times g(x, y) \tag{2.20}$$

$$v(x, y) = f(x, y) \div g(x, y) \tag{2.21}$$

where $x = 0, 1, 2, \ldots, M-1$ and $y = 0, 1, 2, \ldots, N-1$ for images with the size of M rows and N columns.

Addition is a discrete version of continuous integration, and it is an operation frequently used to create double-exposures or composites. A common variant of this operator simply allows a specified constant to be added to every pixel in an image, so as to brighten that image. *Image subtraction* is frequently used to enhance differences between images; such as finding changes between two images.

The *multiplication* or *division* of two images is performed in two main forms. The first form takes two input images and produces an output image in which the pixel values are just those of the first image, multiplied (or divided) by the values of the corresponding values in the second image. The second form takes a single input image and produces output in which each pixel value is multiplied (or divided) by a specified constant. This latter form is probably the more widely used and is generally called *scaling*.

Image averaging is an application for correcting noisy images. Averaging of multiple pictures of the same scene helps to reduce this noise. In practice, however, the images must be previously *registered* (aligned) in order to avoid the introduction of blurring and other artifacts in the output image.

Other related operators are the *logical operators* such as *And*, *Or*, *Not*, *If. . .Then*, *If and only If*, among others which are often used to combine (mostly binary) two images. An example is when a mask is used for selecting a region of interest from an image.

2.4.1.2 Histogram Equalization

The *histogram* of a digital image gives the numeric (quantitative) information about the distribution of the number of pixels per gray-level value. Histograms are the basis for numerous spatial domain processing techniques, and their manipulation can be used for image enhancement. In addition to providing useful image statistics, the information inherent in histograms also is quite useful in other image processing applications, such as image compression and segmentation. Mathematically, the *histogram* of a digital image is a discrete function $h(k) = n_k/n$, where $k = 0, 1, \ldots, L-1$ is the *kth* gray-level, n_k is the number of pixels in the image having gray-level k, and n is the total number of pixels in the image.

Frequently, an image is acquired in such a way that the resulting brightness values do not make full use of the available dynamic range. *Histogram equalization* is a common point operation method for *spreading* the histogram of pixel levels more evenly. The gray levels are transformed with a function such that all gray values are equally represented in the histogram. In the method each original gray-level k is mapped into new gray-level i by:

$$i = \sum\nolimits_{i=0}^{k} h(j) = \sum\nolimits_{j=0}^{k} \frac{n_j}{n} \tag{2.22}$$

where the sum counts the number of pixels in the image with gray-level equal to or less than k. Thus, the new gray-level is the cumulative distribution function of the original gray-levels, which is always monotonically increasing. The resulting image will have a histogram that is flat in a local sense, since the operation of histogram equalization spreads out the peaks of the histogram while compressing other parts of the histogram. In more complicated cases, the global histogram may not be a good representation for local statistics in two parts of the image. In such cases it is best to use an *adaptive histogram equalization* where you can divide the image into several rectangular domains, compute an equalizing histogram and modify levels so that they match across boundaries (Macaire and Postaire 1989). Figure 2.6 illustrates the image enhancement of a loaf of bread using global histogram equalization. In general, the function is non-linear. It is important to realize that whatever point operation is used, the separation into ROIs is still not possible. Thus, although the image in Fig. 2.6c appears easier to separate into components, a machine vision system will still not be able to do it on the basis of point operations only.

2.4.2 *Spatial Image Filtering*

With *spatial image filtering technique*, a window of finite size and shape is scanned across the entire image, transforming the local intensities in that image. The window with its weights is called the *convolution kernel* or *filter mask*. Thus, filtering creates a new pixel with coordinates equal to the coordinates of the center of the neighborhood, and whose value is the result of the filtering operation. Two main linear spatial filtering methods are *correlation* and *convolution*. Correlation is the process of moving a filter mask over the image and computing

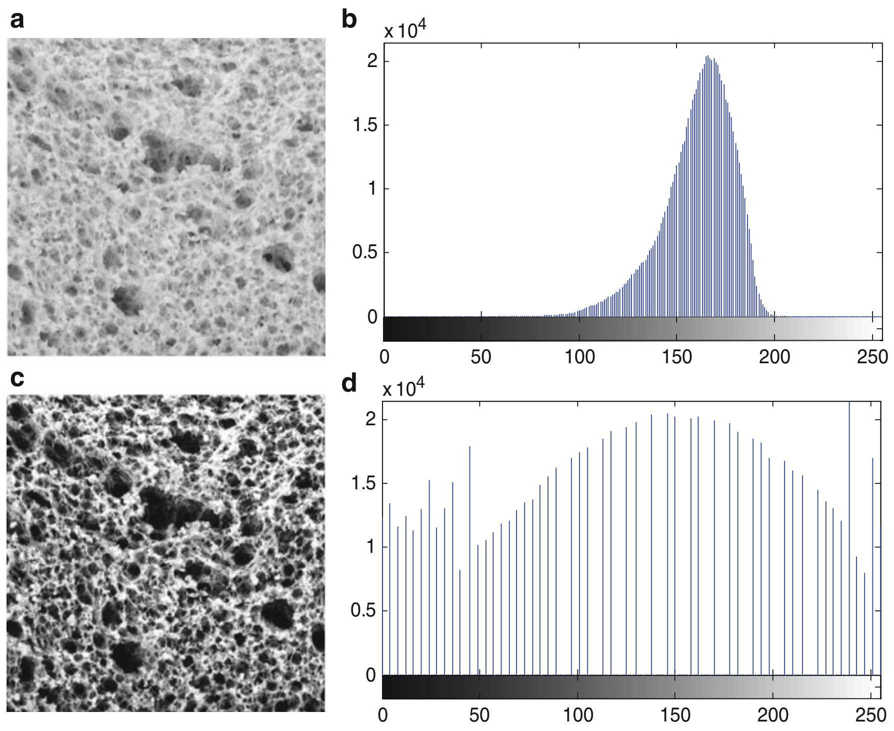


Fig. 2.6 Image quality enhancement using histogram equalization: (**a**) grayscale image of a loaf bread; (**b**) histogram of the image in (**a**); (**c**) resulting image obtained from image (**a**) by histogram equalization; (**d**) histogram of the image in (**c**)

the sum of products at each location. The mechanisms of convolution are the same, except that the filter is first rotated by 180° (Gonzales and Woods 2008). Correlation is often used to measure the similarity between images or parts of images (e.g., pattern matching). Correlation and convolution yield the same result when the filter mask is symmetric. Nonetheless, basic image processing techniques are mainly based on *convolution*.

The convolution is performed by sliding the kernel over the image, generally starting at the top left corner and moving the kernel through all the positions within the boundaries of the image. Thus, in a convolution, a *convolution kernel* is applied to every pixel to create a filtered image, $I_{out}(x, y)$:

$$I_{out}(x, y) = I_{in}(x, y)*W(x, y) = \sum_{u=-\infty}^{\infty} \sum_{u=-\infty}^{\infty} I_{in}(u, v)W(u - x, v - y) \tag{2.23}$$

where the minus signs on the right flip W (i.e., rotate it by 180°). Flipping and shifting W instead of the input image, $I_{in}(u,v)$, is done for notational simplicity and also to follow convention, which is the same as flipping and shifting the input image. It should be noticed that a special treatment is needed for calculating the border of the new image during spatial filtering. Klinger (2003) suggests four possibilities for handling the border of an image. First, if the convolution filter has a size of 3 × 3, get an image smaller by the border of one pixel. Second, keep the same gray-level values of the original image for the new border pixels. Third, use special values for the new border pixels; for example, 0, 127, or 255. Finally, use pixels from the opposite border for the calculation of the new values.

2.4.2.1 Image Smoothing and Blurring

All smoothing filters build a weighted average of the surrounding pixels, and some of them also use the center pixel itself. *Averaging* and *Gaussian* filters are linear filters often used for noise reduction with their operation causing a smoothing in the image but having the effect of blurring edges.

The *average* or *low-pass* filter is a linear filter and one of the simplest types of neighborhood operation. In the *average filtering* the new value is calculated as the average of all nine pixels (for a [3 × 3] kernel) using the same weight. The elements of the mask must be positive and the coefficients for the center pixel are either 0 or 1. With the averaging filter, the mean value of the pixels in the neighborhood is used to form a pixel in a new image at the same location as the center pixel in the original image. Like all averaging operations, this can be used to reduce some types of noise at the expense of a loss of sharpness as it is shown in Fig. 2.7. The averaging operation is represented by,

$$I_{out}(x,y) = \frac{1}{9}\sum_{u=x-1}^{x+1}\sum_{v=y-1}^{y+1} I_{in}(u,v) \tag{2.24}$$

Rather than weight all input pixels equally, it is better to reduce the weight of the input pixels with increasing distance from the center pixel. The *Gaussian filter* does this and is perhaps the most commonly used of all filters (Shapiro and Stockman 2001). The center pixel coefficient of a Gaussian kernel is always greater than 1, and thus greater than the other coefficients because it simulates the shape of a Gaussian curve according to,

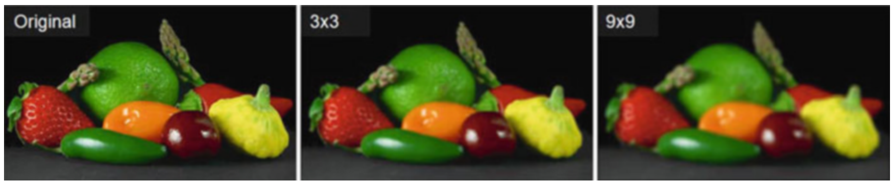


Fig. 2.7 Examples of averaging filter using masks [3 × 3] and [9 × 9]

$$G_{out}(x, y) = \frac{1}{\sqrt{2\pi}\sigma} exp\left(-\frac{d^2}{2\sigma^2}\right) \tag{2.25}$$

where $d = \sqrt{(x - x_c)^2 + (y - y_c)^2}$ is the distance of the neighborhood pixel [x,y] from the center pixel [x_c,y_c] of the output image to which the filter is being applied. Convolution with this kernel forms a weighted average which stresses the point at the center of the convolution window, and incorporates little contribution from those at the boundary. As σ increases, more samples must be obtained to represent the Gaussian function accurately. Therefore, σ controls the amount of smoothing. A second derivative filter based on the Laplacian of the Gaussian is called a *LOG filter*. Examples of Gaussian filter using masks [3×3] and [9×9] with $\sigma = 3$ are given in Fig. 2.8.

Sometimes, non-linear operations on neighborhoods yield better results. An example is the use of a *median filter* to remove noise. *Median filtering* replaces each pixel by the median in a neighborhood around the pixel. Consider an area of a scene of reasonably uniform gray-level; the resulting image may have a single pixel with a very different gray-level due to some random noise. The output of an averaging filter would contain a proportion of this outlier pixel value. However, the *median filter* would set the output pixel to the median value (the 5th largest gray-level in a 3×3 neighborhood) and so it would be unaffected by the outlier. The method is very effective for removing salt and pepper noise (i.e., random occurrences or impulses of black and white pixels) (Gonzales and Woods 2008), as shown in Fig. 2.9. A disadvantage is that the median filter may change the contours of objects in the image.

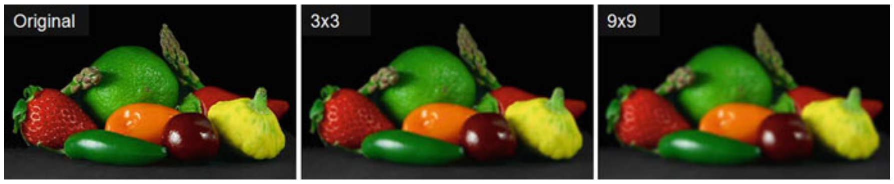


Fig. 2.8 Examples of Gaussian filter using masks [3×3] and [9×9]

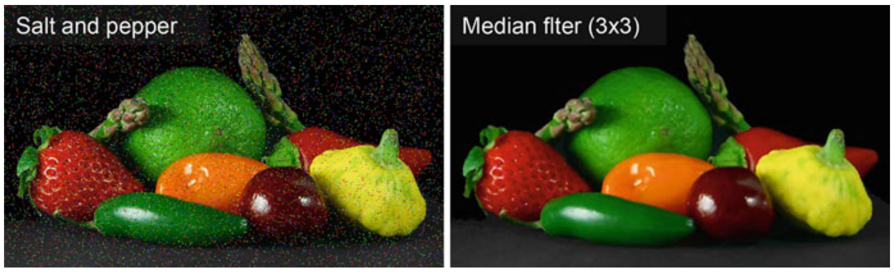


Fig. 2.9 Example of median filter using a kernel [3×3]: the input image (*left*) contains Gaussian noise, and the noise is removed in the resultant image (*right*) after 3×3 median filtering

Computing the median requires more computation time than computing a neighborhood average, since the neighborhood values must be partially sorted. Moreover, median filtering is not so easily implemented in special hardware that might be necessary for real-time processing. However, in many image analysis tasks, its value in image enhancement is worth the time spent.

2.4.2.2 Edge Detection and Enhancement

The detection of edges in an image is an important area of study. An *edge* is an area of an image characterized by sharp changes in gray-level or brightness. The process of edge detection attenuates high fluctuations in color, i.e., dramatic change in intensity. In the frequency domain, this process refers to the attenuation of high frequencies. Among the families of *edge detection* filters are: *gradient filters*, *Laplacian*, and *wavelet transform* (Klinger 2003). Both *gradient* and *Laplacian kernels* are of the *high-pass filter*, which operates by differencing adjacent pixels, because the sharp edges can be described by high frequencies. However, as in other areas of signal processing, *high-pass* filtering amplifies noise, if an appropriate attempt is made to find object boundaries in the image with a simple edge detector (Marchant 2006).

A *gradient filter* extracts a significant brightness change in a specific direction and is thus able to extract edges perpendicular to this direction. These filters are known as *Prewitt filter masks*. The Prewitt operator is based on convolving the image with a small, separable, and integer valued filter in horizontal and vertical direction and is therefore relatively inexpensive in terms of computations. Because the components of the gradient vector are derivatives, they are linear operators.

Another group of gradient masks are the *Sobel filters* or *Sobel kernels*. A Sobel operator gives the specified filter direction a stronger weight. In general, the gradient specifies the amount of change of a value in a certain direction. First derivatives in image processing are implemented using the magnitude of the gradient. The simplest filter kernels, used in two orthogonal directions, are:

$$G_x = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{2.26}$$

and

$$G_y = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.27}$$

resulting in two images, $I_x = (S_x(x,y))$ and $I_y = \left(S_y(x,y)\right)$. The mask coefficients sum to zero, as expected of a derivative operator. The value and direction of the gradient are therefore calculated as follows:

$$I = \sqrt{S_x(x,y)^2 + S_y(x,y)^2} \tag{2.28}$$

and

$$\theta = arctan\left(\frac{S_x(x,y)}{S_y(x,y)}\right) \tag{2.29}$$

The Laplacian operator is an example of a second order or second derivative method of enhancement. It is particularly good at finding the fine detail in an image. All masks of the *Laplacian filter* group are omni-directional, meaning that they provide edge information in each direction. The *Laplacian operators* show two interesting effects. First, if the sum of all coefficients is equal to 0, the filter kernel shows all image areas with a significant brightness change; that means it works as an isotropic or omni-directional edge detector. In other words, isotropic filters are *rotation invariant* (Gonzales and Woods 2008), in the sense that rotating the image and then applying the filter gives the same result as applying the filter to the image first and then rotating the result. Second, if the center coefficient is greater than the sum of the absolute values of all other coefficients, the original image is superimposed over the edge information (Klinger 2003).

The simplest isotropic derivative operator is the *Laplacian*, which for a image *f*(*x*,*y*) of two variables, is defined as

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{2.30}$$

Laplacian as derivative operator highlights intensity discontinuities in an image and deemphasizes regions with slowly varying intensity levels. This will tend to produce images that have grayish edge lines and other discontinuities, all superimposed on a dark, featureless background. Hence, it is important to define the type of Laplacian operator used. If the filter mask has a negative center coefficient, then it subtracts, rather than adds, the Laplacian image to obtain a sharpened result. Typical Laplacian masks are:

$$L_{subtract} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{2.31}$$

and

$$L_{add} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \tag{2.32}$$

The basic representation using these Laplacian masks for image sharpening is,

$$g(x,y) = f(x,y) + c\left[\nabla^2 f(x,y)\right] \tag{2.33}$$

where *f*(*x*,*y*) and *g*(*x*,*y*) are the input and sharpened images, respectively. The constant $c = -1$ if the Laplacian filter $L_{subtract}$ is used and $c = 1$ if L_{add} is used. Since derivative filters are very sensitive to noise, it is common to smooth the image (e.g., using a Gaussian filter) before applying the Laplacian. This two-step process is call the Laplacian of Gaussian (LoG) operation.

2.4.3 Frequency Filtering

Filtering in the frequency domain consists of modifying the Fourier transform of an image and then computing the inverse transform to obtain the processed result. Thus, the Fourier basis can be used to remove high frequency noise from the image of signal, to extract texture features that can be used to classify the type of objects in an image region, and also for image compression. Formally, the Fourier transform is defined by

$$F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{-i2\pi(xu+yv)}dx.dy \tag{2.34}$$

with *F*(*u*,*v*) as the frequencies and *f*(*x*,*y*) as the pixel intensities. The letter $i = \sqrt{-1}$ denotes the *imaginary unit* of complex numbers. The exponential function of Eq. 2.34 satisfies the *Eulerian* formula

$$e^{-i2\pi\alpha} = \cos 2\pi\alpha - i \sin 2\pi\alpha \tag{2.35}$$

for any real α. Thus, the entire set of pixel intensities can be described by a sum of sine and cosine functions but the result is complex.

The *Fast Fourier Transform* (FFT) is more efficient than Fourier transform since it saves computational time by sharing common operations for different pairs of *u*, *v* and is usually used on square images of size $2^m \times 2^m$. It can be described by

$$F(u,v) = \frac{1}{m^2}\sum_{x=0}^{m-1}\sum_{y=0}^{m-1} f(x,y)e^{-i2\pi\left(\frac{xu+yv}{m}\right)} \tag{2.36}$$

where *m* are the numbers of pixels in *x* and *y* directions assuming square images. Despites its common use in image processing, the Fourier transform can cause unwanted degradation in local features of images because it is a global transform that uses all image pixels in every computation of *F*(*u*,*v*) (Shapiro and Stockman 2001). An example of FFT spectrum of a potato chip image and frequency filtering is shown in Fig. 2.10.

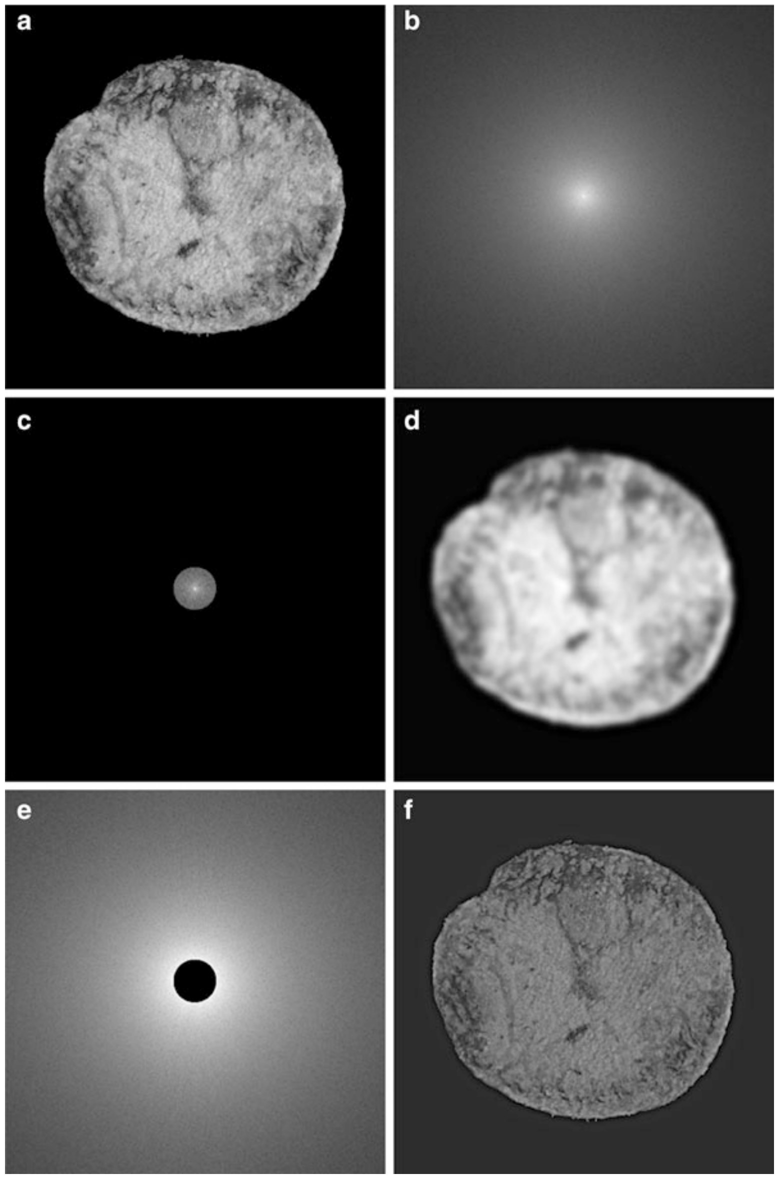


Fig. 2.10 Example of fast Fourier transform (FFT) and frequency filtering: (**a**) *grayscale* image of a potato chip; (**b**) FFT spectrum of (**a**) where low frequency components are grouped in the centre (*brighter pixels*) and high frequencies near to the corners (*darker pixels*); (**c**) low-pass filtered transform defined by a radius = 100 pixels around the central point and zero-out every point in the Fourier image that is beyond that radius; (**d**) low-pass filtered image after applying an inverse Fourier transform to smooth regions of dark and bright, but lose the sharp contours and crisp edges; (**e**) high-pass filtered transform using the same spatial frequency threshold of (**c**), where only the higher spatial frequency components are preserved; (**f**) high-pass filtered image after applying an inverse Fourier transform, which preserves all of the sharp crisp edges from the original, but loses the larger regions of dark and bright

2.4.4 Wavelet Transform

The *wavelet transform* is used primarily for smoothing, noise reduction and lossy compression. In all cases the procedure is to first perform a forward wavelet transform, and then perform some operations on the transformed image followed by an inverse wavelet transform. The reconstructed image exhibits a number of compression-related artifacts, but it is worth noting that unlike an FFT based low-pass filter, the advantage of the wavelet transform is that the image contains a fair amount of high-frequency content.

The wavelet transform is a convolution product of a signal with a scaled and translated kernel (usually a *n-th* derivative of a smoothing kernel in order to precisely detect singularities)

$$Wf(u,s) = \frac{1}{s}\int_{-\infty}^{\infty} f(x)\psi\left(\frac{x-u}{s}\right) \cdot dx \tag{2.37}$$

where s and u are real numbers (s and $u > 0$) which are discretized for computational purposes. The wavelet transform performs a transformation of a function $f(x)$ into a function defined over the scale-space plane (pair of values u and s). For implementation, typical wavelet bases are: *Daubechies* of order 4, *Symlet* of order 2 and 4, first derivative of a *Gaussian* and *Mexican Hat* wavelets.

Two simple multi-scale wavelet transforms can be used for image processing: the discrete wavelet transform (DWT) and the continuous wavelet transform (CWT). For DWT computations, a number of discrete levels can be computed from the images, and four sets of coefficients are extracted for each level: horizontal, vertical, diagonal and approximation coefficients (Mallat 1989). For CWT computations, the extension of the previous concept from 1D (Eq. 2.25) to 2D is given by Piñuela et al. (2007):

$$Mf(u,v,s) = \sqrt{\left|W^1 f(u,v,s)\right|^2 + \left|W^2 f(u,v,s)\right|^2} \tag{2.38}$$

with u and v denoting the 2D coordinates and the scale parameter being usually used as $s = 2j$. Now, two separate wavelet transforms are computed: W^1 refers to the wavelet transform performed along the horizontal dimension and W^2 refers to the vertical one.

2.4.5 Binary Morphological Operations

Morphological operations are methods for processing binary images based on shapes. Basically, morphological operations change the structure of the elements or particles in an image. In a binary image, the elements or particles are defined as

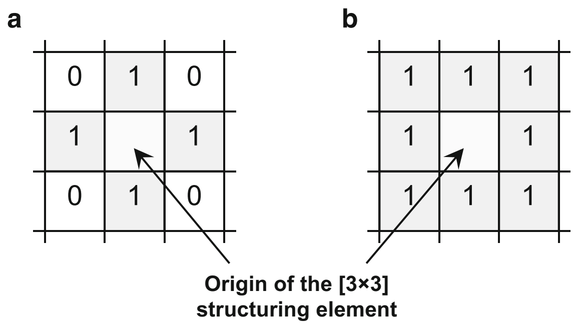


Fig. 2.11 Examples of square structuring elements (a) connectivity 4; (b) connectivity 8

the segmented regions (or ROIs) in which the pixel value is 1. The rest of the image (pixel value 0) is called background. In these operations, the value of each pixel in the output image is based on the corresponding input pixel and its neighbors. Morphological operations can be used to construct filters similar to the spatial filters discussed above. The basic operations of binary morphology are *dilation*, *erosion*, *closing*, and *opening*.

Morphological operations use a structuring element to calculate new pixels, which plays the role of a neighborhood or convolution kernel in other image processing operations (as shown in filter mask operations). Figure 2.11 shows two typical examples of structuring elements. The shape of the structuring element could be rectangular, square, diamond, octagon, disk, etc. The connectivity defines whether four or all eight surrounding pixels are used to calculate the new center pixel value (in the case of a [3×3] structuring element) (Klinger 2003).

2.4.5.1 Erosion and Dilation

These two operators are fundamental for almost all morphological operations (Gonzales and Woods 2008). Opening and closing are also duals of each other with respect to set complementation and reflection. Thus, *Erosion* is an operator that basically removes objects smaller than the structuring element and removes perimeter pixels from the border of larger image objects (sets the pixel value to 0). If I is an image and M is the structuring element (mask), the erosion (operator $\ominus$) is defined as:

$$erosion(I) = I \ominus M = \cap_{a \in M} I_{-a} \tag{2.39}$$

where I_a indicates a basic shift operation in the direction of element a of M and I_{-a} would indicate the reverse shift operation.

Contrarily, a *dilation* operation enlarges a region. A dilation adds pixels to the perimeter of each image object (sets their values to 1), filling in holes and broken areas, and connecting areas that are separated by spaces smaller than the size of the structuring element. The dilation (operator $\oplus$) is defined as:

$$dilation(I) = I \oplus M = \cup_{a \in M} I_a \tag{2.40}$$

2.4.5.2 Opening and Closing

Both operations generate a certain amount of smoothing on an object's contour. An *opening* operation (erosion then dilation) can separate objects that are connected in a binary image. Opening generally smoothes the contour of an object, breaks narrow isthmuses, and eliminates thin protrusions. Mathematically, the opening function can be described by

$$opening(I) = dilation(erosion(I)) \tag{2.41}$$

or, using the operator $\circ$,

$$I \circ M = (I \ominus M) \oplus M \tag{2.42}$$

The closing operation is defined as dilation followed by an erosion using the same structuring element. A *closing* operation can close up internal holes and gaps in a region and eliminate bays along the boundary.

$$closing(I) = erosion(dilation(I)) \tag{2.43}$$

or, using the operator $\bullet$,

$$I \bullet M = (I \oplus M) \ominus M \tag{2.44}$$

It should be noted that multiple openings or closings have no effect after the operator has been applied once.

2.5 Image Segmentation

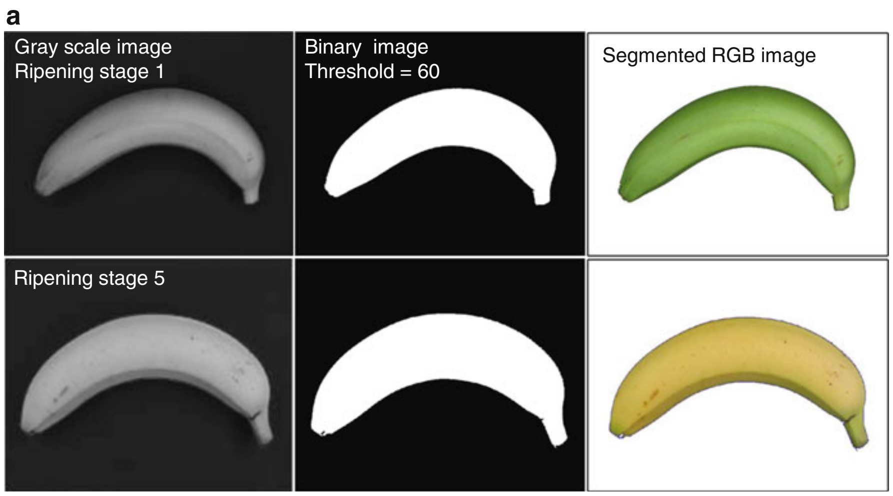
Image segmentation is one of the most important steps in the entire image processing technique, as subsequent extracted data are highly dependent on the accuracy of this operation. Its main aim is to divide an image into regions that have a strong correlation with objects or areas of interest (i.e., ROIs). Segmentation can be achieved by three different techniques: *thresholding*, *edge-based* segmentation, and *region-based* segmentation (Sonka et al. 1999; Sun 2000). These algorithms, their modifications, and combinations with other approaches are frequently used in food quality applications.

Thresholding or global thresholding is a simple and fast technique for characterizing image regions based on constant reflectivity or light absorption of their surfaces. Edge-based segmentation relies on edge detection by edge operators. Edge operators detect the sharp discontinuities in gray level, color, texture, etc., in the image. Region-based segmentation involves the grouping together of similar pixels to form regions representing single objects within the image (such as in the seeded region growing method). The segmented image may then be represented as a boundary or a region. Boundary representation is suitable for analysis of size and shape features, while region representation is used in the detection and evaluation of texture, defects or simply ROIs (Brosnan and Sun 2004).

Two examples are presented to illustrate the segmentation process. Figure 2.12 shows the image segmentation process for green and yellow bananas using a simple global thresholding. In general, successful results are found for the identification of the true area of both bananas since the intensity distribution of the banana peel and background pixels are sufficient distinct. The histogram is bimodal with the peaks for background being tall and narrow, and separated from the banana peaks by deep valleys (Fig. 2.12b).

Ideally a segmentation process should be fully automatic so that it can provide fully objective and consistent data. Several methods have been proposed for automatic threshold determination. An interesting alternative for that is the *Otsu method*, which selects the threshold based on the minimization of the within-group variance of the two groups of pixels separated by the thresholding operator. So, if the histogram is bimodal, the thresholding problem is to determine the best threshold t separating the two modes of the histogram from each other (as shown in Fig. 2.12). Each threshold t determines a variance for the group of values that are less than or equal to t ($\leq t$) and a variance for the group of values greater than t ($>t$). The definition for the best threshold suggested by Otsu is that threshold for which the weighted sum of within-group variances is minimized. The weights are the probabilities of the respective groups (Shapiro and Stockman 2001). The application of the Otsu method for segmenting the same images of bananas in ripening stages 1 and 5 (shown in Fig. 2.12) gave threshold values of 56 and 63, respectively.

Removal of the background should be fairly straight-forward but removing non-useful sub-regions of an object can be far more difficult. Difficulty in segmentation can vary with each task. Figure 2.13 compares the results of segmenting pores and cellular material of apple tissue from X-ray computed tomography (CT) images using a *global thresholding* and a combination with the *kriging based segmentation* method (Oh and Lindquist 1999). For automatic segmentation of apple pores from a stack of CT images, a common practice in using a simple global threshold is to choice a threshold value that would match a predetermined bulk measurement of porosity (e.g., around 28 % for 'Jonagold' apples). However, this procedure is very subjective and may lead to biases when one is trying to segment and extract reproducible quantitative information from multiple images (e.g., for binary 3D reconstructions from stacks of CT images). The distinction between the void and solid phases in radiographic images is frequently not sharp (i.e., do not show a bimodal distribution) due to the amount of peak overlapping in



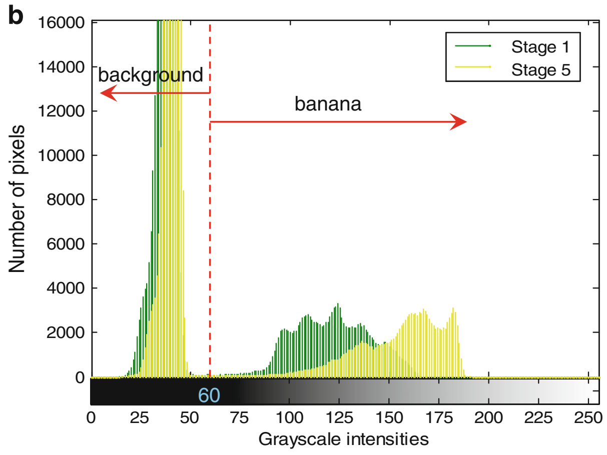
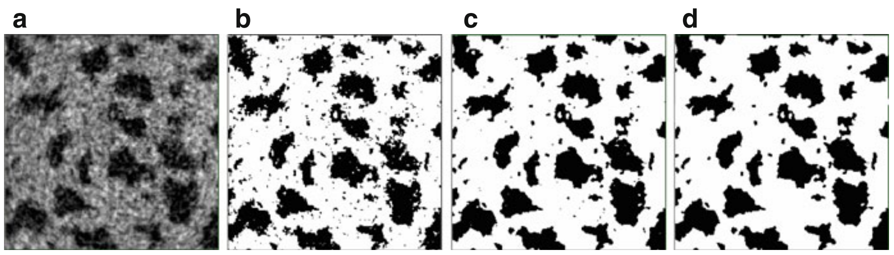


Fig. 2.12 Image segmentation process for bananas using a simple global thresholding: (**a**) pre-processing of a color image of bananas in ripening stages 1 and 5, including previous *grayscale* transformation and image smoothing using a *Laplacian-of-Gaussian filter* (LOG filter) [3 × 3] for easing the edge detection, binarization, and segmentation; (**b**) histogram of the grayscale images for both bananas showing the chosen threshold value of 60

the attenuation coefficient histogram and the nature of X-ray tomography (where processing and analysis are based on voxels instead of pixels). Moreover, the resulting binary image of apple tissue using global thresholds of 60 is noisy (Fig. 2.13b), and the average porosity is highly dependent on the selected threshold value. Figure 2.13e plots the distribution of the transmitted X-ray intensities (solid line, right axis) measured for the reconstructed tomography images of apple tissue, and it also shows the typical dependence of the porosity (open circles, left axis) of the resulting segmented image when a simple global threshold is chosen.



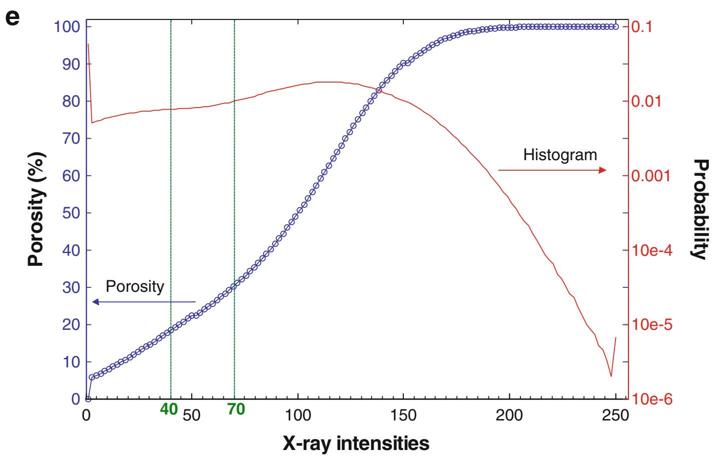


Fig. 2.13 Image segmentation process for apple tissue images using a simple global thresholding and krigking based segmentation methods: (**a**) original grayscale image; (**b**) segmented image using a simple global threshold of 60; (**c**) segmented image after using a thresholding window T0 (=40) and T1 (=70) and indicator krigking for edge detection; (**d**) segmented image after cleaning (**c**) by erosion and dilation operations; and (**e**) the distribution of the transmitted X-ray intensities (*solid line*, *right axis*) measured for the tomography images of apple tissue, and the typical dependence of the porosity (*open circles*, *left axis*) of the resulting segmented image when a simple global threshold is chosen (Reproduced from Mendoza et al. 2007b with permission from Springer-Verlag)

Alternatively, the thresholding algorithm developed by Oh and Lindquist (1999) is a non-parametric formulation able to analyze regions of uncertainty based on the estimation of the spatial covariance of the image in conjunction with indicator kriging to determine object edges (Mardia and Hainsworth 1988). Use of indicator kriging makes the thresholding local based on two threshold values, T_0 and T_1, and guarantees smoothness in the threshold surface. Implementation of the method requires a-priori population identification of some percentage of the image. Thus, for the thresholding step of apple tissue images, the gray threshold values were set at 40 and 70. According to this thresholding window, pores were identified as those voxels with gray values less than 40, and voxels of gray value greater than 70 were classified as cellular material (in general, non-edge). Finally, the remaining voxels of the population were identified by indicator kriging (Fig. 2.13c).

Small disconnected void or solid phase components which may be either physical or arise from segmentation error, can be cleaned up from each image using a morphological procedure called *opening* which involves a simple step of erosion followed by dilation (Fig. 2.13d).

2.6 Quantitative Analysis

A large part of computer vision is concerned with extracting features from images. Techniques by which numerical measurements are extracted from images vary considerably in technological complexity. The process of image measurement involves an enormous reduction in the amount of data, by selecting from the original image those objects and parameters that are important for further characterization and sorting of samples. This selection and reduction is at the heart of image analysis and measurement, and it is achieved by ignoring irrelevant information.

In food images, the most frequently occurring features are color, texture and morphology. These are the features that reveal information on palatability, quality and defects. They are easy to measure but can be difficult to characterize succinctly. After processing of the image and with the region of interest identified, the region must be described with measurable features based on standard chemical and physical measurements (e.g., color parameters by colorimeter or spectrophotometer, firmness by penetrometer, soluble solids content by refractometer, etc.), or empirical standards developed by experts (standard charts for color, gloss and texture of food surfaces, and sensorial evaluations). Many image processing techniques developed for 2D images can also be extended to analyzing multidimensional images, such as multispectral or hyperspectral images and tomographic images. In this section, we discuss in some detail the three main approaches and methods used in image analysis: color analysis, image texture analysis, and geometrical or morphological analysis, accompanied with specific application examples in food quality inspection.

2.6.1 Color Analysis

Color is specified by the geometry and spectral distributions of three elements: the light source, the reflectivity of the sample and the visual sensitivity of observer. Each of these was defined by the *Commission Internationale de I'Eclairage* (CIE) in 1931. CIE also defined, for the purpose of color measurement, the cone of spectral sensitivities of an average observer and introduced several methods to describe color objectively. The definition was aimed at stimulating the human color perception based on a 2° field of view, a set of primaries (red, green, blue), and color-matching functions (CIE 1986). Several standard illuminants which are specified by their color temperatures were also defined by CIE. The most common one is standard illuminant D_{65}, corresponding to the radiation of a black body at 6,500°K, which is intended to represent average daylight (Hunt 1991).

2.6.1.1 Color Evaluation

To accurately specify object colors and color differences, CIE recommended two color spaces that are less illumination-dependent, namely, CIELAB or $L^*a^*b^*$ and CIELUV $L^*u^*v^*$ (Robertson 1976). The CIELAB space has a function of correcting for chromatic adaptation to the white point, and is intended for object color displays. The CIELUV space is defined in a similar manner, and the coordinate (L^*, u^*, v^*) is calculated from the Y and (u', v') of the given light stimulus and the white point.

The color difference in the CIELAB space is calculated as the Euclidean distance between the points in this three-dimensional space, and is given by,

$$\Delta E^*_{ab} = \left[\left(\Delta L^*\right)^2 + \left(\Delta a^*\right)^2 + \left(\Delta b^*\right)^2\right]^{1/2} \tag{2.45}$$

Equation 2.45 is called the CIE 1976 (L^*, a^*, b^*) color difference formula. Chroma $C^*{}_{ab}$ and hue angle $h^*{}_{ab}$ are also calculated from (L^*, a^*, b^*) by

$$C^*_{ab} = \left(a^{*2} + b^{*2}\right)^{1/2} \tag{2.46}$$

$$h^*_{ab} = tan^{-1}\left(b^*/a^*\right) \tag{2.47}$$

Procedures for calibrated color measurements and color reproduction using machine vision systems with color digital cameras were proposed and discussed in Mendoza et al. (2006) for the analysis of fruits and vegetables and in Valous et al. (2009a) for the analysis of pre-sliced hams.

2.6.1.2 Color Measurement on Curved Surfaces

Computer vision systems based on color digital cameras can quantify the color of any food easily and quickly (Segnini et al. 1999; Papadakis et al. 2000). Different color channels give different color information about the pixel intensity distribution of an image, and therefore, evaluations using different color components represent an advantage in further quality characterization and classification tasks. However, average color results from curved and/or uneven surfaces, as they are typical in many fresh foods, should be interpreted with caution. A sensitivity analysis of color measurements by a common machine vision system for samples with curved surfaces (bananas and red pepper) revealed that $L^*a^*b^*$ is more appropriate for

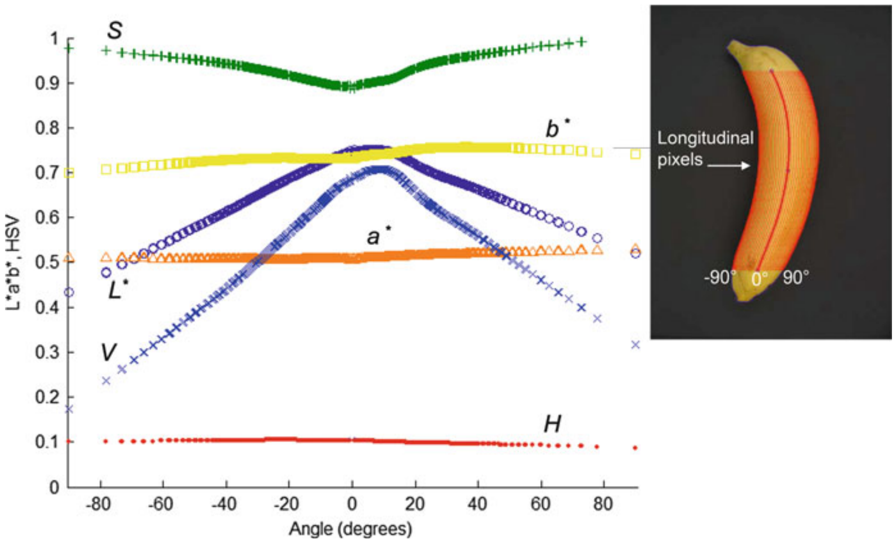


Fig. 2.14 Color profiles expressed in *L*a*b** and *HSV* color scales for a yellow banana. The profiles were constructed using the average values of the pixels in the longitudinal direction of the segmented image and its angular positions varying from – 90° to 90° (Modified from Mendoza et al. 2006)

color representation of surfaces or materials illuminated by a light source. These color profiles are less affected by the degree of curvature, shadows and glossiness of the surfaces than the *RGB* and *HSV* color systems, and, therefore, more appropriate for color measurement of food surfaces (Mendoza et al. 2006). An example of the effect of the curvature on a yellow banana using the *L*a*b** and *HSV* color scales is illustrated in Fig. 2.14. *L** and *V* color scales were highly sensitive to the curvature of the banana surface, while the *S* scale was also surface-curvature sensitive but to a lesser extent. For the *a**, *b** and *H* scales no or minimum color variations were observed.

2.6.2 Texture Analysis

The meaning of the term *texture* in image processing is completely different from the usual meaning of texture in foods. Image texture can be defined as the spatial organization of intensity variations in an image at various wavelengths, such as the visible and infrared portions of the electromagnetic spectrum (Haralick et al. 1973). Image texture is an important aspect of images and textural features play a major role in image analysis. In some images, it can be the defining characteristic of regions and critical in obtaining a correct analysis (Shapiro and Stockman 2001). These features provide summary information defined from the intensity maps of the scene, which may be related to visual characteristics (coarseness of the texture, regularity, presence of a privileged direction, etc.), and also to characteristics that cannot be visually differentiated.

Texture is a property of areas, so there is no texture for a point. Texture involves the spatial distribution of gray levels and there is a need for a significant number of intensity units (i.e., pixels) in a region to detect texture features. This is linked to the fact that texture involves gray levels in a spatial neighborhood. There are many texture analysis techniques that can be applied to images. Among the most popular methods used for the characterization and evaluations of food surfaces and biological structures are: *First-order Statistics*, *Gray Level Co-occurrence* and *Run Length Matrices*, and *Fractals*.

2.6.2.1 First-Order Statistics (FOS)

Image histogram gives primarily the global description of the image. The histogram of a gray-level image represents the relative frequency of occurrence of each gray-level in the image. The features of FOS are commonly derived from the normalized gray-level histogram of the digital image, which is built by counting the number of pixels (N) with the gray value of i (I) and can be written as:

$$H(i) = N\langle (x, y) | I(x, y) = i \rangle \tag{2.48}$$

The histogram, $H(i)$, is normalized using the function given below:

$$H^{'}(i) = \frac{H(i)}{\sum_i H(i)} \tag{2.49}$$

The extracted statistical features for further partial least squares analysis included: *mean* of the pixel histogram (MV), *variance* (VA), *entropy* (ET), and *energy* (EN), which are defined as follows (Cernadas et al. 2005):

$$MV = \sum_i iH^{'}(i) \tag{2.50}$$

$$VA = \sum_i (i - MV)^2 H^{'}(i) \tag{2.51}$$

$$ET = -\sum_i H^{'}(i) \log\left(H^{'}(i)\right) \tag{2.52}$$

$$EN = \sum_i i^2 H^{'}(i) \tag{2.53}$$

These FOS features, however, do not provide any insight about the possible textural differences in an image, because they do not extract any information about the relative position of pixels, and the correlation of their intensities.

2.6.2.2 Gray Level Co-occurrence Matrix (GLCM)

When a non-random spatial distribution (or more than one texture in an image) has to be characterized, second-order statistics are more suitable to describe these types of relationships. A GLCM is a matrix with a number of rows and a number of columns equal to the number of gray level intensities of the image, which contains textural features (spatial relationships) from an image obtained using second order statistics. Each element of the gray level co-occurrence matrix $P(i,j|\Delta x, \Delta y)$ is the relative frequency or probability of occurrence linked to the combination of intensities $I(i,j)$ in all pixel pairs of the image located at a distance $(\Delta x, \Delta y)$. This relative position is defined by a distance (d) and an angle ($\theta = 0°$, 45°, 90°, 135°). For a given directional orientation and distance of the patterns, 14 textural features can be extracted from a grayscale image using this matrix as proposed by Haralick et al. (1973), the most common being *energy*, *entropy*, *contrast*, *correlation*, *local homogeneity* and *variance*. They are computed by:

$$Energy = \sum_i \sum_j P_{d\theta}(i,j)^2 \tag{2.54}$$

$$Entropy = -\sum_i \sum_j P_{d\theta}(i,j) \cdot \log(P_{d\theta}(i,j)) \tag{2.55}$$

$$Contrast = \sum_i \sum_j (i-j)^2 \cdot P_{d\theta}(i,j) \tag{2.56}$$

$$Correlation = \frac{\sum_i \sum_j (i \cdot j) p(i,j) - \mu_i \cdot \mu_j}{\sigma_i \cdot \sigma_j} \tag{2.57}$$

$$Homogeneity = \sum_i \sum_j \frac{1}{1+(i-j)^2} \cdot P_{d\theta}(i,j) \tag{2.58}$$

$$Variance = \sum_i \sum_j \left(i - u_{ij}\right)^2 \cdot P_{d\theta}(i,j) \tag{2.59}$$

where μ_i and μ_j are the means, and σ_i and σ_j are the standard deviations. *Energy* measures the textural uniformity of the image, i.e., the repetition of pixel pairs.

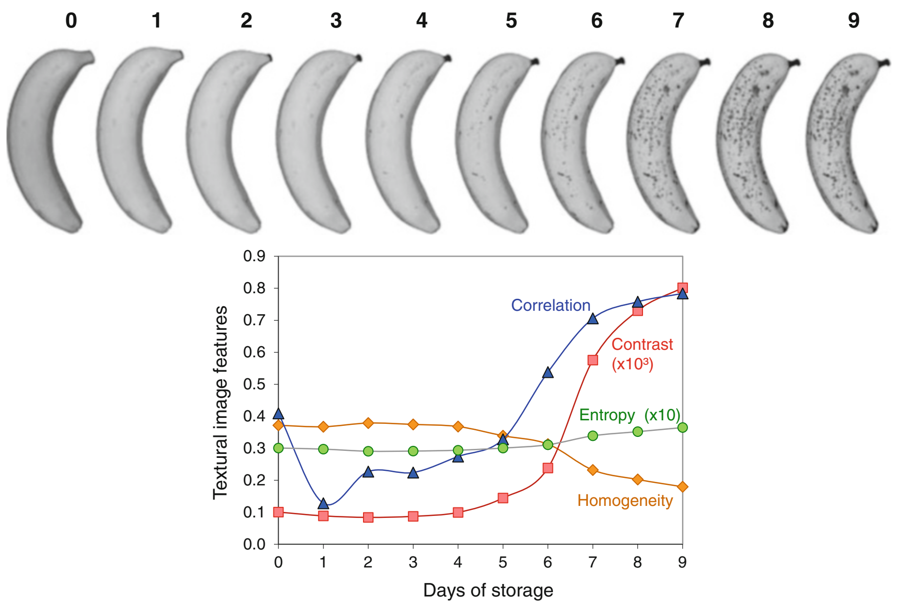


Fig. 2.15 Image texture analysis using the gray level co-occurrence matrix (GLCM) method for quantifying the ripening process of a banana (*Musa cavendish*)

Entropy measures the disorder or randomness of the image and it is an indication of complexity within the image, i.e., more complex images have higher entropy values. *Contrast* is a measure of the local variations present in the image; so higher contrast values indicate larger local variations. *Homogeneity* (also called an inverse difference moment) is inversely proportional to contrast at constant energy. Similarly at constant contrast, homogeneity is inversely proportional to energy. Finally, *correlation* is a measure of image linearity among pixels (Mendoza et al. 2007a). An application of GLCM features for the characterization of the surface appearance of a banana (*Musa cavendish*) during ripening is presented in Fig. 2.15.

2.6.2.3 Run Length Matrix (RLM)

First introduced by Galloway (1975), the gray-level RLM approach characterizes texture by the gray-level run, which is a set of consecutive pixels with the same gray-level. Run length is the number of pixels in a run. Therefore, the run length of coarse textures will be longer than that of fine textures. In RLM method, a matrix containing the information about the run length of images is constructed in terms of the brightness value and length of the runs (Fardet et al. 1998).

The run-length matrix $P(i, j)$ is defined by specifying direction (i.e., 0°, 45°, 90°, 135°) and then counting the occurrence of runs for each gray levels and length in this direction. i-dimension corresponds to the gray level (bin values) and has a length equal to the maximum gray-level (bin values), *j*-dimension corresponds to the run length and has a length equal to the maximum run length (bin values). Five features were proposed by Galloway (1975), namely, *short run emphasis* (SRE) which measures the distribution of short runs, *long run emphasis* (LRE) which measures the distribution of long runs, *gray-level non-uniformity* (GLNU) which measures the similarity of gray-level values throughout the image, *run length non-uniformity* (RLNU) which measures the similarity of the length of runs throughout the image, and *run length percentage* (RLP) which measures the homogeneity and the distribution of runs of an image in a given direction. These RLM features are calculated as follows:

$$SRE = \frac{1}{n_r}\sum_{i=1}^{M}\sum_{j=1}^{N}\frac{P(i,j)}{j^2} \tag{2.60}$$

$$LRE = \frac{1}{n_r}\sum_{i=1}^{M}\sum_{j=1}^{N}P(i,j)\cdot j^2 \tag{2.61}$$

$$GLNU = \frac{1}{n_r}\sum_{j=1}^{N}\left(\sum_{i=1}^{M}P(i,j)\right)^2 \tag{2.62}$$

$$RLNU = \frac{1}{n_r}\sum_{i=1}^{M}\left(\sum_{j=1}^{N}P(i,j)\right)^2 \tag{2.63}$$

$$RLP = \frac{n_r}{n_p} \tag{2.64}$$

in which n_r is the total number of runs and n_p is the number of pixels in the image, M is the number of gray levels (bins) and N is the number of run length (bins).

2.6.2.4 Fractal Methods

A fractal describes a rough or fragmented geometric shape that can be subdivided into parts, each of which is, at least approximately, a reduced-size copy of the whole. This means that they are generally self-similar and independent of scale (Mandelbrot 1983). Contrary to the classical Euclidean geometry, fractals are not regular and may have an integer or non-integer dimension. Thus, fractal dimensions offer a systematic approach in quantifying irregular patterns that contain an internal structure repeated over a range of scales. Self-similarity is not visually obvious, but there may be numerical or statistical measures that are preserved across scales.

Due to their statistical scaling invariance, natural objects may exhibit statistical fractality (Klonowski 2000).

The methods to calculate fractal dimensions can be divided into two types: spatial and spectral. The first type operates in the spatial domain, while the second type operates in the frequency domain, using the Fourier power spectrum. The two types are unified by the principle of fractional Brownian motion (Dougherty and Henebry 2001).

Box-Counting Method

Estimations of *fractal dimension* are all based on the *box-counting method*. The technique allows to obtain the scaling properties of two dimensional fractal objects (such as from binary images) by covering a measure with a range of boxes of size ε and counting the number of boxes containing at least one pixel representing the object under study. This means that the technique does not consider the amount of mass (density of pixels) inside each box. Hence, in a homogeneous system, the number N of features of a certain size ε varies as (Evertsz and Mandelbrot 1992):

$$N(\varepsilon) \propto \varepsilon^{-D_0} \tag{2.65}$$

where the fractal dimension D_0,

$$D_0 = \lim_{\varepsilon \to 0} \frac{log N(\varepsilon)}{log \frac{1}{\varepsilon}} \tag{2.66}$$

can be measured by counting number N of boxes needed to cover the object under investigation for increasing box sizes ε and estimating the slope of a *log*-*log* plot. Figure 2.16 illustrates the scaling properties of a binary image for apple tissue.

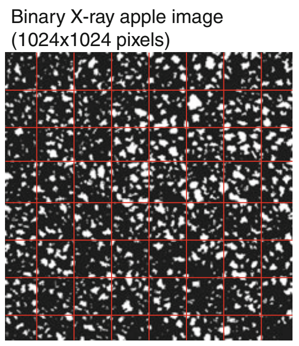
Variogram Method

The *directional fractal dimension* (DFD) of grayscale images using the *variogram* model is computed as follows (Kube and Pentland 1988; Mandelbrot 1983):

$$v(d) = c \cdot d^a \tag{2.67}$$

$$v(d) = \frac{1}{N_d} \sum_{N(d)} [y(s+d) - y(s)]^2 \tag{2.68}$$

$$DFD = 3 - 0.5\hat{a} \tag{2.69}$$



$\varepsilon = Lx2^{-k}$	$N(\varepsilon)=2^k$	Actual $N(\varepsilon)$
$1024\text{x}2^{-1}$	4	4
$1024\text{x}2^{-2}$	16	16
$1024\text{x}2^{-3}$	64	64
$1024\text{x}2^{-4}$	256	250
$1024\text{x}2^{-5}$	1024	995
$1024\text{x}2^{-6}$	4096	3084
$1024\text{x}2^{-7}$	16384	7236
$1024\text{x}2^{-8}$	65536	18282
$1024\text{x}2^{-9}$	262144	53050

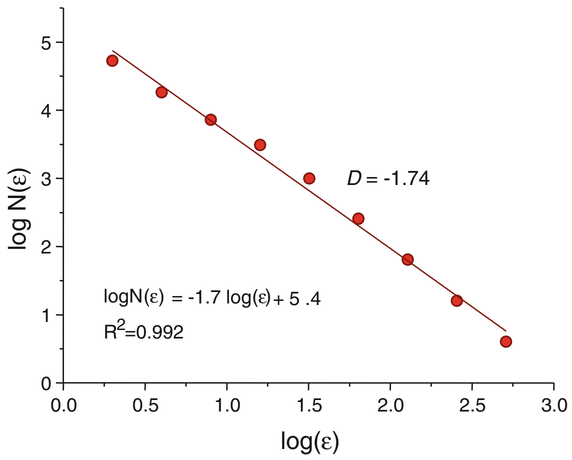


Fig. 2.16 Illustration of fractal *box-counting* theory applied to a binary image of fresh apple tissue (pores are represented by *white pixels*)

where $v(d)$ is the *variogram* of the image, a is termed the fractal index, c is a constant, d is the separation distance in pixels, $y(s)$ denotes the gray-level of the pixel at location s, and $N(d)$ denotes the cardinality of the set of pairs of observations. The image *variogram* $v(d)$ represents the variance or dispersion of the difference between two random variables. Thus, the relationship between a and $v(d)$ can also be represented using a linear regression model by applying log function to both sides of this equation to get an estimate of fractal index $\hat{a}$. Then, the directional fractal dimension (DFD) of the image can be directly computed.

The *variogram* of the image and hence the fractal dimension are estimated at a fixed image resolution level without specifying any spatial direction along which the set of pairs of observations is constructed. This means that the image is assumed to be isotropic. Since the fractal texture parameters of many images of foods and biological materials do not have isotropic patterns, four variograms should be computed along the directions $d = 0°$, $45°$, $90°$, and $135°$ (namely horizontal, first diagonal, vertical and second diagonal) respectively. These variograms should be analyzed and averaged for a given pixel location s of an image, and then be used in further analysis and image characterization.

This approach has been recently applied to the appearance characterization and classification of commercial pork, turkey and chicken ham slices (Mendoza et al. 2009). DFD features were extracted from the digitalized intensity images in grayscale, and R, G, B, L^*, a^*, b^*, H, S, and V color components were calculated for three image resolution levels (100, 50, and 25 %). Simulation results showed that in spite of the complexity and high variability in color and texture appearance, the modeling of ham slice images with DFD allowed the capture of differentiating textural features between the four commercial ham types. Independent DFD features entailed better discrimination than that using the average of four directions. However, the DFD features revealed a high sensitivity to color channel, orientation and image resolution for the fractal analysis. The classification accuracy using six DFD features was 93.9 % for the training data and 82.2 % for the testing data.

Fourier Fractal Texture (FFT)

In the calculation of FFT, the 2D Fourier transform of the grayscale image is first calculated and the 2D power spectrum is then derived. The 2D power spectrum is reduced to a 1D radial power spectrum (direction-independent mean spectrum, i.e., the average of all possible directional power spectra) by averaging values over increasingly larger annuli for each of the radial increments. The power spectrum, $P(f)$, varying with frequency f, is calculated as follows (Dougherty and Henebry 2001):

$$P(f) = k \cdot f^{(-1-2H)} \tag{2.70}$$

where k is a constant and H is the *Hausdorff-Besicovitch dimension*. When the log $[P(f)]$ is plotted against log$[f]$, a straight line can be fitted. According to the Fourier slice theorem, the 1D Fourier transform of a parallel projection of an image along a line with the direction h is identical to the value along the same line in the 2D Fourier transform of the image. This means that the line through the spectrum gives the spectral information obtained from a projection with the same orientation in the spatial domain. FFT dimensions, D_f, are calculated as a function of orientation based on this theorem, with 24 being the frequently number of directions that the frequency space is uniformly divided. The data of magnitude vs. frequency are plotted in *log-log* scale and its slope is determined using linear least-squares regression. Thus, Hausdorff-Besicovitch dimension H is computed from the slope c of the straight line, $c = (-1-2H)$. The D_f dimension of the grayscale image is related to the slope c of the *log–log* plot by the equation below, with $H = D_f - 3$, $2 < D_f < 3$ and $3 < c < 1$ (Geraets and Van der Stelt 2000):

$$D_f = \frac{7}{2} + \frac{c}{2} \tag{2.71}$$

For analysis the slope and intercept for all directions can be computed from each image and used for further image characterization and classification. This algorithm was proposed by Russ (2005) and was modified by Valous et al. (2009b) for hams processing.

Fractal Lacunarity (FL)

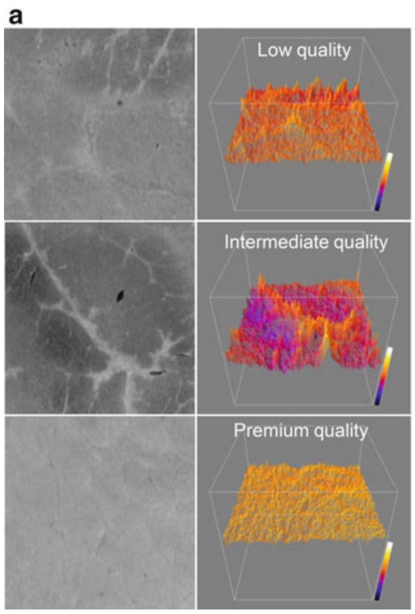
Various studies have shown that the fractal dimension alone is not a sufficient metric for the characterization of most textures, because fractal dimension only measures how much space is filled. A second-order fractal metric such as *Lacunarity* complements fractal dimension by measuring how the data fill the space, enabling the parsimonious analyses of textures (Tolle et al. 2008). In this method, a square structuring element or moving window of side length b is placed in the upper left-hand corner of the ham slice image of side length T (pixels), such that $b \leq T$. The algorithm records the number or mass m of pixels that are associated with the image underneath the moving window. The window is then translated by one pixel to the right and the underlying mass is again recorded. When the moving window reaches the right-hand side of the image, it is moved back to its starting point at the left-hand side of the image and is translated by one pixel downward. The computation proceeds until the moving window reaches the lower right-hand edge of the image, at which point it has explored every one of its $(T - b + 1)^2$ possible positions. Allain and Cloitre (1991) defined *lacunarity* Λ measured with a square moving window of side length b on an image of side length T (pixels), such that $b \leq T$, as:

$$\Lambda = \frac{\sigma^2}{\mu^2} + 1 \tag{2.72}$$

where the ratio of σ (standard deviation) to μ (mean) changes with window size which signifies that lacunarity depends on the scale (moving window size relative to image size). Lacunarity may assume any value between 1 and ∞. A lacunarity value of 1 indicates that the textural content is uniformly dispersed at a given scale, while values $\neq 1$ indicate non-uniformity.

The technique can be performed on binary and grayscale images. However, computing the lacunarity for characterizing binary images would result in the counting of every gliding box, thus the boxes would all be full and the output would be rather uninteresting. In contrast, when the algorithm measures the average intensity of pixels per gliding box in most cases the results of this textural analysis is more than adequate (Krasowska et al. 2004).

Valous et al. (2009b) applied the gliding box algorithm to characterize the texture appearance of ham slices based on segmented images of pores/defects and fat-connective tissue. Later, Valous et al. (2010) used the same approach to characterize ham slices based on grayscale information. Lacunarity was computed for each value of b between $b_{\min} = 2$ and $b_{\max} = 256$ with a step size of 1 for the 256×256 pixel intensity images. Once the computation was complete, the lacunarity as a function of moving window b was presented as a spatial response function with a 2D scatter plot, which illustrated the scale dependency of spatial non-stationary in the intensity images. As shown in Fig. 2.17, lacunarity plots revealed important textural content information that corresponds to degree of



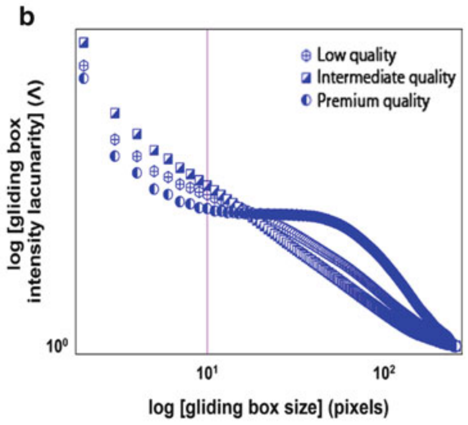


Fig. 2.17 Visualization of surface of three wet-cured cooked pork ham qualities (high yield or low quality, medium yield or intermediate quality, and low yield or premium quality ham) using different muscle sections, percentages of brine solutions and processing regimes. (**a**) intensity images of *B* color channel and their three-dimensional mesh plots. (**b**) two-dimensional *log–log* scatter plot of averaged values of intensity *lacunarity* as a function of moving window *b*. (Reproduced from Valous et al. 2010)

spatial heterogeneity of intensities and level of self-similar behavior. The results of intensity lacunarity suggested that window sizes up to 10 pixels may be adequate to cover textural features and produce meaningful results.

2.6.3 Morphology Analysis

Once a set of regions-of-interest has been identified by image segmentation, the individual regions (objects) or silhouettes can be described by their geometric and shape properties. Most image analysis systems offer a number of measures related to the size and shape and produce a numeric output suitable for further image characterization and classification. Size and shape are common object measurements for food quality evaluation; and compared with other features such as color and image texture, they are easier to measure using image processing techniques. Among these morphological properties are the counting of objects (such as particles, holes, granules, droplets, bubbles, among others) and defining their position, surface roughness and orientation in the image.

2.6.3.1 Particle Size

Many image analysis applications need to characterize or describe the distribution of particle sizes on binary images. The usual way of summarizing data for a particular system is to draw a frequency histogram of the number of particles in a certain size class. Furthermore, since many natural samples are broadly shaped like a *normal* or *Gaussian* distribution, this allows to apply statistical methods to evaluate confidence parameters and make recommendations on the minimum number of particles to analyze in order to achieve a reasonable level of statistical significance. Another widely used method of depicting data is to calculate a cumulative distribution, which shows what percentage of material lies above or below a particular size (Aguilera and Stanley 1999).

In food quality evaluations, there are three commonly used features for size measurement of an object: area, perimeter, and length and width. The most basic measurement for size is the area, which is represented by the number of pixels within the area and is straightforwardly determined by counting. The perimeter of an object is the length of its boundary, and it is particularly useful for discriminating between objects with simple and complex shapes. Area and perimeter are easily computed from a segmented image, however the quality of the measurements is highly dependent of the complexity of the measured objects and the efficiency of the segmentation method. The length and width of an object can also be used to measure the size of an object. It is necessary to locate the major axis of the object and measure its relative length and width (Du and Sun 2004).

Although relative measurements of the percentage of pixels of selected areas on an image could be done for various particle size evaluations, in some applications we need to know the x and y dimensions of an image in its real dimensional units. Hence, before making any measurements, the relationships between the size of a pixel to a size of an object of known length (in mm for example) that is visible in the image must be first specified. A scale calibration factor (f) for the x and y directions can be determined by

$$f = \frac{real\ distance\ (mm)}{image\ distance\ (pixels)} \tag{2.73}$$

2.6.3.2 Shape Descriptors

Measuring size alone is sometimes insufficient to detect important but subtle differences between samples. This is because particles or grains with similar composition and/or structure could measure the same area or perimeter, but have different shapes.

Frequently, the objects of one class can be distinguished from the others by their shapes, which are physical dimensional measurements that characterize the appearance of an object. Shape features can be measured independently and by combining

Table 2.2 Common shape descriptors

Statistical parameter	Calculation
Area ratio	$= \dfrac{Area}{Max\ Diameter \cdot Min\ Diameter}$
Aspect ratio	$= \dfrac{Max\ Diameter}{Min\ Diameter}$
Compactness	$= \dfrac{Perimeter^2}{Area}$
Circularity	$= \dfrac{4\pi \cdot Area}{Perimeter^2}$
Diameter range	$= Max\ Diameter - Min\ Diameter$
Eccentricity	$= \sqrt{1 - \dfrac{Semi\ Minor^2}{Semi\ Major^2}}$
Elongation	$= 1 - Aspect\ Ratio$
Roundness	$= \dfrac{4\pi \cdot Area}{\pi \cdot Max\ Diameter^2}$
*Shape factor*1	$= \dfrac{4\pi \cdot Area}{Perimeter^2}$
*Shape factor*2	$= \dfrac{Max\ Diameter}{Area}$

size measurements. Table 2.2 summarizes some of the most widely used shape features with combinations of size measurements for food products.

Of particular interest in particle analysis is *circularity*, which is a good measure of deviation from a perfect circle. However it is important to remember that it is unlikely that one single shape descriptor will perfectly discriminate and characterize all applications and different combinations of shapes. *Circularity* has values in the range of $0 - 1$. A perfect circle has a circularity of 1, while a very *spiky* or irregular object has a circularity value closer to 0. Circularity is sensitive to both overall form and surface roughness.

2.7 Concluding Remarks

The field of image processing and analysis has experienced dramatic growth and its application has become increasingly widespread in recent years. The development of new and more effective algorithms for image processing and analysis, along with advances in image acquisition, computer, data storage, and the Internet, has made it possible to handle an increasingly large volume of image data. Digital image processing has become economical in many fields of research and in industrial applications. While each application is different or unique from the others, they are all concerned about speed, affordability and performance or accuracy. More and more research activities and applications in machine vision are being focused on real-time and interactive operations, where image acquisition, processing, and analysis and decision-making are almost carried out simultaneously or in parallel.

For the past decade, we have also witnessed an exponential increase in research and development activities in multispectral and hyperspectral imaging technology for food and agriculture applications. While many of the basic image processing and analysis methods described in this chapter are still useful for processing and analyzing two or three-dimensional multispectral and hyperspectral images, it also entails new challenges in handling these types of images. Methods and techniques specific to processing and analyzing multispectral and hyperspectral images are presented in Chaps. 6 and 7.

References

Aguilera JM, Stanley DW (1999) Microstructural principles of food processing and engineering, 2nd edn. Aspen, Gaithersburg

Allain C, Cloitre M (1991) Characterizing the lacunarity of random and deterministic fractal sets. Phys Rev A 44(6):3552–3558

Brosnan T, Sun D-W (2004) Improving quality inspection of food products by computer vision—a review. J Food Eng 61:3–16

Castleman KR (1979) Digital image processing. Prentice-Hall, Englewood Cliffs

Cernadas E, Carrión P, Rodriguez PG, Muriel E, Antequera T (2005) Analyzing magnetic resonance images of Iberian pork loin to predict its sensorial characteristics. Comput Vis Image Und 98:344–360

CIE (1986) Colorimetry, Official recommendations of the International Commission on Illumination, CIE Publication No. 15.2. CIE Central Bureau, Vienna

Dougherty G, Henebry GM (2001) Fractal signature and lacunarity in the measurement of the texture of trabecular bone in clinical CT images. Med Eng Phys 23:369–380

Du C-J, Sun D-W (2004) Recent developments in the applications of image processing techniques for food quality evaluation. Trends Food Sci Technol 15:230–249

Du C-J, Sun D-W (2005) Comparison of three methods for classification of pizza topping using different color space transformations. J Food Eng 68:277–287

Du C-J, Sun D-W (2007) Quality measurement of cooked meats. In: Sun D-W (ed) Computer vision technology for food quality evaluation. Elsevier/Academic, London, pp 139–156

Evertsz CJG, Mandelbrot BB (1992) Multifractal measures. In: Peitgen H-O, Jurgens H, Saupe D (eds) Chaos and fractals. New frontiers of science. Springer, New York, pp 921–953

Fardet A, Baldwin PM, Bertrand D, Bouchet B, Gallant DJ, Barry J-L (1998) Textural images analysis of pasta protein networks to determine influence of technological processes. Cereal Chem 75:699–704

Galloway MM (1975) Texture analysis using grey level run lengths. Comput Graph Image Process 4:172–179

Geraets WGM, Van der Stelt PF (2000) Fractal properties of bone. Dentomaxillofac Radiol 29:144–153

Gonzales RC, Woods RE (2008) Digital image processing. Prentice-Hall, Englewood Cliffs

Gunasekaran S, Ding K (1994) Using computer vision for food quality evaluation. Food Technol 6:151–154

Haralick RM, Shanmugan K, Dinstein I (1973) Textural features for image classification. IEEE Trans Syst Man Cybern 3:610–621

Hunt RWG (1991) Measuring of color, 2nd edn. Ellis Horwood, New York

Klinger T (2003) Image processing with LabVIEW and IMAQ vision. Prentice Hall Professional Technical Reference, Upper Saddle River

Klonowski W (2000) Signal and image analysis using chaos theory and fractal geometry. Mach Graphics Vis 9(1/2):403–431

Krasowska M, Borys P, Grzywna ZJ (2004) Lacunarity as a measure of texture. Acta Phys Pol B 35:1519–1534

Kube P, Pentland A (1988) On the imaging of fractal surfaces. IEEE Trans Pattern Anal Mach Intell 10:704–707

Mallat S (1989) A theory for multiresolution signal decomposition: the wavelet representation. IEEE Trans Pattern Anal Mach Intell 11:674–693

Mandelbrot BB (1983) The fractal geometry of nature. W.H. Freeman, New York

Marchant JA (2006) Machine vision in the agricultural context. In: Munack A (ed CIGR) Precision agriculture, CIGR handbook of agricultural engineering, vol VI. Information technology. The International Commission of Agricultural Engineering, St. Joseph, MI (Chapter 5)

Mardia KV, Hainsworth TJ (1988) A spatial thresholding method for image segmentation. IEEE Trans Pattern Anal Mach Intell 6:919–927

Macaire L, Postaire JG (1989) Real-time adaptive thresholding for on-line evaluation with line-scan cameras. In: Proceedings of computer vision for industry, Society of Photooptical Instrumentation Engineers, Boston, MA, pp 14–25

Mendoza F, Dejmek P, Aguilera JM (2006) Calibrated color measurements of agricultural foods using image analysis. Postharvest Biol Tech 41:285–295

Mendoza F, Dejmek P, Aguilera JM (2007a) Color and texture image analysis in classification of commercial potato chips. Food Res Int 40:1146–1154

Mendoza F, Verboven P, Mebatsion HK, Kerckhofs G, Wevers M, Nicolaï B (2007b) Three-dimensional pore space quantification of apple tissue using X-ray computed microtomography. Planta 226:559–570

Mendoza F, Valous NA, Allen P, Kenny TA, Ward P, Sun D-W (2009) Analysis and classification of commercial ham slice images using directional fractal dimension features. Meat Sci 81:313–320

Oh W, Lindquist W (1999) Image thresholding by indicator kriging. IEEE Trans Pattern Anal Mach Intell 21:590–602

Piñuela JA, Andina D, McInnes KJ, Tarquis AM (2007) Wavelet analysis in a structured clay soil using 2-D images. Nonlin Process Geophys 14:425–434

Puglia S (2000) Technical primer. In: Sitts MK (ed) Handbook for digital projects: a management tool for preservation and access, 1st edn. Northeast Document Conservation Center, Andover

Papadakis S, Abdul-Malek S, Kamdem RE, Jam KL (2000) A versatile and inexpensive technique for measuring color of foods. Food Technol 54:48–51

Prats-Montalbán JM, de Juan A, Ferrer A (2011) Multivariate image analysis: a review with applications. Chemometr Intell Lab 107:1–23

Rec. ITU-R BT.709-5 (2002) Parameter values for the HDTV standards for production and international programme exchange (1990, revised 2002). International Telecommunication Union, 1211 Geneva 20, Switzerland

Robertson AL (1976) The CIE 1976 color difference formulae. Color Res Appl 2:7–11

Russ JC (2005) Image analysis of food microstructure. CRC, New York

Segnini S, Dejmek P, Öste R (1999) A low cost video technique for color measurement of potato chips. Lebensm-Wiss U-Technol 32:216–222

Shapiro LG, Stockman GC (2001) Computer vision. Prentice-Hall, Upper Saddle River

Soille P (1999) Morphological image analysis. Springer, Berlin

Sonka M, Hlavac V, Boyle R (1999) Image processing, analysis, and machine vision. PWS, Pacific Grove

Sun D-W (2000) Inspecting pizza topping percentage and distribution by a computer vision method. J Food Eng 44:245–249

Tolle CR, McJunkin TR, Gorsich DJ (2008) An efficient implementation of the gliding box lacunarity algorithm. Phys D 237:306–315

Valous NA, Mendoza F, Sun D-W, Allen P (2009a) Colour calibration of a laboratory computer vision system for quality evaluation of pre-sliced hams. Meat Sci 42:353–362

Valous NA, Mendoza F, Sun D-W, Allen P (2009b) Texture appearance characterization of pre–sliced pork ham images using fractal metrics: Fourier analysis dimension and lacunarity. Food Res Int 42:353–362

Valous NA, Sun D-W, Mendoza F, Allen P (2010) The use of lacunarity for visual texture characterization of pre-sliced cooked pork ham surface intensities. Food Res Int 43(3):87–395

Wikipedia (2012) Binary image. http://en.wikipedia.org/wiki/Binary_image. Accessed 12 Dec 2012