

Domingo Mery

# Computer Vision for X-Ray Testing

Imaging, Systems, Image Databases and  
Algorithms

April 17, 2015

Springer

4.4.1	Gradient estimation	121
4.4.2	Laplacian-of-Gaussian	124
4.4.3	Canny edge detector	126
4.5	Segmentation	127
4.5.1	Thresholding	129
4.5.2	Region growing	133
4.5.3	Maximally stable extremal regions	136
4.6	Image restoration	138
4.7	Summary	144
<b>5</b>	<b>X-ray Image Representation</b>	<b>145</b>
5.1	Introduction	146
5.2	Geometric features	147
5.2.1	Basic geometric features	147
5.2.2	Elliptical features	150
5.2.3	Fourier descriptors	152
5.2.4	Invariant moments	153
5.3	Intensity features	156
5.3.1	Basic intensity features	156
5.3.2	Contrast	158
5.3.3	Crossing line profiles	160
5.3.4	Intensity moments	165
5.3.5	Statistical textures	165
5.3.6	Gabor	167
5.3.7	Filter banks	168
5.4	Descriptors	169
5.4.1	Local binary patterns	170
5.4.2	Binarized statistical image features	172
5.4.3	Histogram of oriented gradients	172
5.4.4	Scale-invariant feature transform	173
5.5	Sparse representations	177
5.5.1	Traditional dictionaries	178
5.5.2	Sparse dictionaries	178
5.5.3	Dictionary learning	180
5.6	Feature selection	181
5.6.1	Basics	181
5.6.2	Exhaustive search	187
5.6.3	Branch and bound	188
5.6.4	Sequential forward selection	188
5.6.5	Sequential backward selection	189
5.6.6	Ranking by class separability criteria	190
5.6.7	Forward orthogonal search	190
5.6.8	Least square estimation	191
5.6.9	Combination with principal components	191
5.6.10	Feature selection based in mutual information	192

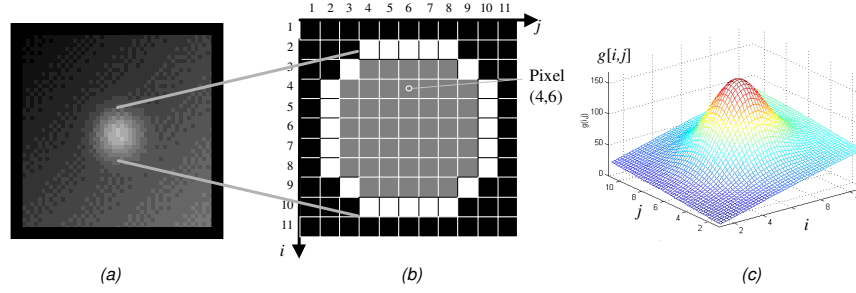


Fig. 5.1: Example of a region: a) X-ray image, b) segmented region (gray pixels), c) 3D representation of the gray values.

## 5.1 Introduction

As we learned in the previous Chapter, in image processing for X-ray testing, segmentation is used to detect (potential) regions that can be the objects of interest that we are looking for (see Section 4.5). As segmented potential regions frequently set off false detections, an analysis of the segmented regions can significantly improve the effectiveness of detection. Measuring certain characteristics of the segmented regions (*feature extraction*) can help us to distinguish the false detection, although some of the features extracted are either irrelevant or are not correlated. Therefore, a *feature selection* must be performed. Depending on the values returned for the selected features, we can try to classify each segmented potential region in one of the following two classes: *background* or *object of interest*.

In this Chapter, we will explain several features that are normally used in image analysis and computer vision for X-ray testing. In our description, features will be divided into two groups: *geometric* and *intensity* features. Furthermore, we will cover some local descriptors and sparse representations that can be used in many X-ray testing applications. In this Chapter we shall concentrate on the extraction and selection of features, whereas in the following Chapter we will discuss the classification problem itself.

We will use Fig. 5.1 as our example in the description of features. In our example, we use an X-ray image of a circular defect. The segmentation is a binary image that gives information about the pixels that belongs to our object of interest (the defect). Geometric features are extracted from this binary image. Moreover, intensity features are extracted from the intensity image considering the pixels of the segmentation. Some intensity features consider only the gray values inside the segmented region, other ones take into account both gray values inside and outside the region (*e.g.*, contrast).

## 5.2 Geometric features

These provide information on the location, size and shape of the segmented region. Location and size features, such as center of mass, perimeter, height and width, are given in pixels. Shape features are usually coefficients without units. It is worth mentioning that we distinguish three different zones in the segmented image (see Fig. 5.1b), the segmented region (gray zone  $\mathfrak{R}$ ), the boundary (white edge pixels  $\ell$ ) and the background (black zone).

### 5.2.1 Basic geometric features

In this Section we will summarize basic geometric features that can be easily extracted.

#### Height and width

The height and width of a region can be defined as:

$$h = i_{\max} - i_{\min} + 1 \quad \text{and} \quad w = j_{\max} - j_{\min} + 1 \quad (5.1)$$

where  $i_{\max}$  and  $i_{\min}$  is the maximal and minimal value that takes coordinate  $i$  in the region. The same is valid for  $j_{\max}$  and  $j_{\min}$  in  $j$ -direction. In our example of Fig. 5.1,  $h = w = 7$  pixels.

#### Area and perimeter

We define the area  $A$  of a region as the number of pixels that belong to the region. On the other hand, the perimeter  $L$  is the number of pixels that belong to the boundary. In the region of Fig. 5.1, the area and the perimeter are  $A = 45$  and  $L = 24$  pixels, respectively. More accurate measurements for area and perimeter can also be estimated [34]: for instance, the boundary of the region can be fitted to a curve with known area and length (in our example the boundary can be fitted to a circle with radius  $r = 4$  pixels, so  $A = \pi r^2 = 50.26$  pixels and  $L = 2\pi r = 25.13$  pixels), however, the computational time of such approaches can be extremely long if there are thousands of regions to be measured. Moreover, the shape of the region can be much more complex than a simple circle as shown in Fig. 4.30. We should remember, therefore, that the goal of feature extraction is not the accurate measurement, rather it is simply the extraction of features that can be used in a classification approach to separate our classes (objects of interest from background). Thus, it is not relevant that the measurement of the area of the region is just 45 pixels and not 50.26 pixels.

#### Center of mass

This provides information about the location of the region. It is computed as the average of coordinate  $i$  and coordinate  $j$  in pixels that belong to region  $\mathfrak{R}$ :

$$\bar{i} = \frac{1}{A} \sum_{i \in \mathfrak{R}} i \quad \bar{j} = \frac{1}{A} \sum_{j \in \mathfrak{R}} j \quad (5.2)$$

where  $A$  is the area of the region, *i.e.*, the number of pixels of the region.

### Roundness

Shape features are usually attributed coefficients without units. An example is roundness that is defined as:

$$R = \frac{4 \cdot A \cdot \pi}{L^2} \quad (5.3)$$

The roundness  $R$  is a value between 1 and 0.  $R = 1$  means a circle, and  $R = 0$  corresponds to a region without an area. In our example  $R = 4 \cdot 45 \cdot \pi / 24^2 = 0.98$ .

### Other basic features

There are some useful features that can be extracted employing the Image Processing Toolbox of Matlab using command `regionprops`. According to User's Guide of the Image Processing Toolbox [144] these are defined as follows:

- **EulerNumber**: The number of objects in the region minus the number of holes in those objects.
- **EquivDiameter**: The diameter of a circle with the same area as the region.
- **MajorAxisLength** and **MinorAxisLength**: The length (in pixels) of the major and minor axis of the ellipse that has the same normalized second central moments as the region.
- **Orientation**: The angle (in degrees ranging from -90 to 90 degrees) between the x-axis and the major axis of the ellipse that has the same second-moments as the region.
- **Solidity**: The proportion of the pixels in the convex hull that are also in the region.
- **Extent**: The ratio of pixels in the region to pixels in the total bounding box.
- **Eccentricity**: The eccentricity of the ellipse that has the same second-moments as the region.

All basic geometric features explained in this Section can be extracted by command `Xbasicgeo` of  $\mathbb{X}$ vis Toolbox. An example is shown in Table 5.1, where the basic 15 geometric features (divided by 1000) are presented for 10 regions of Fig. 5.2:  $f_1$ : Center of mass in  $i$  direction.  $f_2$ : Center of mass in  $j$  direction.  $f_3$ : Height.  $f_4$ : Width.  $f_5$ : Area.  $f_6$ : Perimeter.  $f_7$ : Roundness.  $f_8$ : Euler Number.  $f_9$ : Equivalent Diameter.  $f_{10}$ : Major Axis Length.  $f_{11}$ : Minor Axis Length.  $f_{12}$ : Orientation.  $f_{13}$ : Solidity.  $f_{14}$ : Extent.  $f_{15}$ : Eccentricity.



Matlab Example 5.1: In this example, we show how to extract the basic geometric features of ten apples as segmented in Fig. 5.2.

Table 5.1: Basic geometric features of apples of Fig. 5.2 [ → Example 5.1 ↗]

$k$	$f_1$	$f_2$	$f_3$	$f_4$	$f_5$	$f_6$	$f_7$	$f_8$	$f_9$	$f_{10}$	$f_{11}$	$f_{12}$	$f_{13}$	$f_{14}$	$f_{15}$
1	0.3805	0.2954	0.1380	0.1520	17.2868	0.4795	0.0009	0.0010	0.1483	0.1574	0.1413	-0.0097	0.0010	0.0008	0.0004
2	0.3922	0.4499	0.1590	0.1530	18.6500	0.4940	0.0010	0.0010	0.1540	0.1624	0.1474	0.0696	0.0010	0.0008	0.0004
3	0.1571	0.4733	0.1490	0.1500	16.3525	0.4650	0.0010	0.0010	0.1442	0.1533	0.1371	-0.0512	0.0010	0.0007	0.0004
4	0.4659	0.5945	0.1430	0.1470	15.8229	0.4567	0.0010	0.0010	0.1418	0.1513	0.1343	-0.0445	0.0010	0.0008	0.0005
5	0.2433	0.6468	0.1480	0.1370	14.7376	0.4783	0.0008	0.0010	0.1369	0.1587	0.1207	-0.0619	0.0010	0.0007	0.0006
6	0.2210	1.4405	0.1670	0.1640	19.8205	0.5170	0.0009	0.0010	0.1588	0.1678	0.1528	-0.0326	0.0010	0.0007	0.0004
7	0.3853	1.4341	0.1510	0.1270	15.4304	0.4537	0.0009	0.0010	0.1401	0.1549	0.1277	-0.0836	0.0010	0.0008	0.0006
8	0.5513	1.4840	0.1740	0.1720	21.4611	0.5393	0.0009	0.0010	0.1652	0.1790	0.1536	-0.0415	0.0010	0.0007	0.0005
9	0.4136	1.5847	0.1500	0.1530	16.7374	0.4878	0.0009	0.0010	0.1459	0.1547	0.1416	0.0536	0.0010	0.0007	0.0004
10	0.2484	1.6098	0.1690	0.1580	19.5241	0.5112	0.0009	0.0010	0.1576	0.1673	0.1507	0.0734	0.0010	0.0007	0.0004

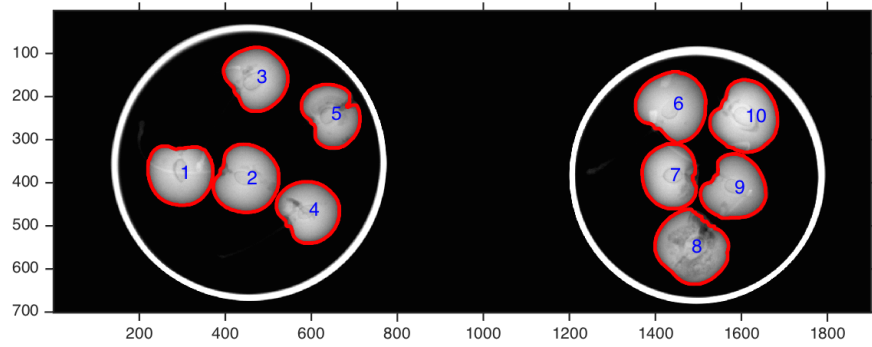


Fig. 5.2: X-ray image of 10 apples. [ → Example 5.1 ↗]

Listing 5.1 : Basic geometric features

```

% AppleBasicGeoFeatures.m
I = Xloading('N',1,4);           % X-ray of apples
I = I(300:1000,100:2000);       % input image
J = and(I>60,I<110);
K = imerode(bwfill(J,'holes'),ones(11,11)); % segmentation
[L,n]=bwlabel(K,4);             % regions

X = zeros(n,15);                % features
E = zeros(size(I));              % edges of the regions
imshow(I);hold on
for i=1:n
    Ri = imdilate(L==i,ones(11,11));
    E = or(E,bwperim(Ri));
    X(i,:) = Xbasicgeo(Ri);      % basic geo-features
    text(X(i,2),X(i,1),sprintf('%d',i),... % output
         'color','b','fontsize',12)
end
[ii,jj] = find(E==1); plot(jj,ii,'r.') % display edges
X/1000                                % features divided by 1000

```

The output of this code is shown in Fig. 5.2 and Table 5.1. The basic geometric features are extracted by command `Xbasicgeo` of `Xvis` Toolbox. □

### 5.2.2 Elliptical features

Elliptical features can be used to extract information about location, size and shape of a region. They are extracted from a fitted ellipse to the boundary of the region [55]. From this ellipse we can extract the center, the length of the axes, the orientation and the eccentricity.

The pixels of the boundary are defined as  $(x_i, y_i)$  for  $i = 1 \dots L$ . It is well known that an ellipse is defined as:

$$ax^2 + bxy + cy^2 + dx + ey + f = 0, \quad (5.4)$$

that can be written as  $\mathbf{a}^T \mathbf{x} = 0$ , where  $\mathbf{a} = [a \ b \ c \ d \ e \ f]^T$  is a vector that includes the parameters of the ellipse and  $\mathbf{x} = [x^2 \ xy \ y^2 \ x \ y \ 1]^T$  is a vector that includes the coordinates of a point  $(x, y)$  that lies on the ellipse.

If our region is elliptical, then for each point  $(x_i, y_i)$  we have  $\mathbf{a}^T \mathbf{x}_i = 0$  with  $\mathbf{x}_i = [x_i^2 \ x_i y_i \ y_i^2 \ x_i \ y_i \ 1]^T$ . Nevertheless, in practice the regions are not perfectly elliptical, not only because real regions have different shapes, but also there is a discretization error when forming a digital image. For this reason, we look for a vector  $\mathbf{a}$  so that  $\mathbf{a}^T \mathbf{x}_i \rightarrow \min$  for every point  $i = 1 \dots L$ . That is, we can formulate the estimation of the parameters of the ellipse as an optimization problem as follows:

$$\|\mathbf{X}\mathbf{a}\| \rightarrow \min \quad (5.5)$$

where  $\mathbf{X}$  is matrix with  $L$  rows whose  $i$ -th row is  $\mathbf{x}_i^T$ . Usually, a solution can be found by minimizing (5.5) subject to  $\|\mathbf{a}\| = 1$ . In this case,  $\mathbf{a}$  is the last column of matrix  $\mathbf{V}$ , where  $\mathbf{X} = \mathbf{U}\mathbf{S}\mathbf{V}^T$  is the singular-value-decomposition (SVD) of  $\mathbf{X}$  [81].

The elliptical features can be extracted by writing (5.4) as follows:

$$\left(\frac{x - x_0}{a_e}\right)^2 + \left(\frac{y - y_0}{b_e}\right)^2 = 1 \quad (5.6)$$

where

$$a_e = \frac{1}{\sqrt{s} a_p}, \quad b_e = \frac{1}{\sqrt{s} b_p} \quad (5.7)$$

with

$$s = \frac{1}{v - f} \quad v = \mathbf{t}^T \mathbf{T} \mathbf{t}$$

$$\mathbf{T} = \begin{bmatrix} a & b/2 \\ b/2 & c \end{bmatrix} \quad \mathbf{t} = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = \frac{1}{2} \mathbf{T}^{-1} \begin{bmatrix} d \\ e \end{bmatrix}$$

$$\begin{aligned} a_p &= a \cos^2(\alpha) + b \cos(\alpha) \sin(\alpha) + c \sin^2(\alpha) \\ b_p &= a \sin^2(\alpha) - b \cos(\alpha) \sin(\alpha) + c \cos^2(\alpha) \end{aligned}$$

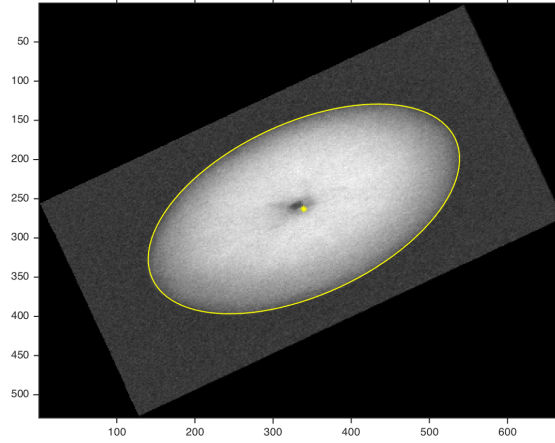



Fig. 5.3: Elliptical features of a fruit. In this example, the coordinates of the center of the ellipse correspond to the yellow cross ( $i = 262.99, j = 339.42$ ). The estimated length of each axis are 109.71 and 213.39 pixels. The orientation (with respect to vertical axis in a counterclockwise direction) is 1.1396 rad, *i.e.*,  $65.30^\circ$ . The eccentricity is 0.5141. [  $\rightarrow$  Example 5.2  ].

and

$$\alpha = \frac{1}{2} \arctan\left(\frac{b}{a-c}\right) \quad (5.8)$$

The axes of the ellipse are defined by  $a_e$  and  $b_e$ , the center of the ellipse is located on  $(x_0, y_0)$  and the orientation is  $\alpha$ . Thus, the eccentricity is defined by

$$e_x = \frac{\min(a_e, b_e)}{\max(a_e, b_e)} \quad (5.9)$$

For circular shapes, the eccentricity as the roundness (5.3), takes values between 0 and 1, where 1 means a perfect circle.



Matlab Example 5.2: In this example, we show how to extract elliptical features of a shape. We test this approach on an X-ray of an apple with a circular shape that was transformed and rotated as shown in Fig. 5.3.

#### Listing 5.2 : Elliptical boundary of a fruit

```
% EllipticalBoundary.m
I = double(Xloading('N',5,9)); % input image
I = imrotate(I(1:2:end,:),25); % shape transformation and rotation
R = Xsegbimodal(I); % segmentation
[X,Xn] = Xfitellipse(R); % ellipse features
Xprintfeatures(X,Xn) % features
imshow(I,[]); hold on
```



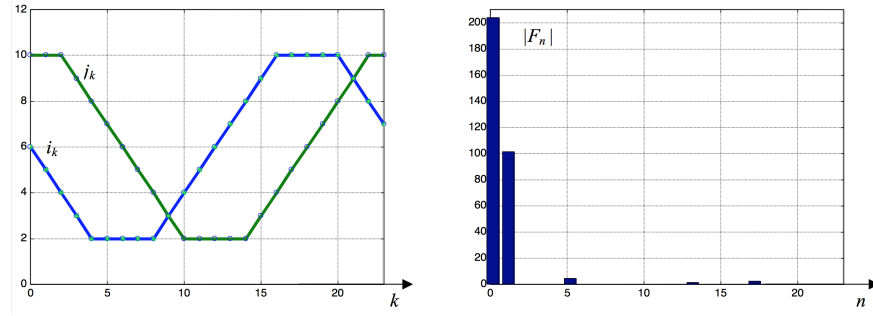


Fig. 5.4: Coordinates of the boundary of region of Fig. 5.1 and the Fourier descriptors.

```

Xdrawellipse(X,'y')           % ellipse drawing
plot(X(2),X(1),'y*')          % center of ellipse

```

The output of this code is shown in Fig. 5.3. The elliptical features are extracted by command `Xfitellipse` of `XVIS` Toolbox.  $\square$

### 5.2.3 Fourier descriptors

Shape information –invariant to scale, orientation and position– can be measured using *Fourier descriptors* [32, 203, 273]. The coordinates of the pixels of the boundary are arranged as a complex number  $i_k + j \cdot j_k$ , with  $j = \sqrt{-1}$  and  $k = 0, \dots, L-1$ , where  $L$  is the perimeter of the region, and pixel  $k$  and  $k+1$  are connected. The complex boundary function can be considered as a periodical signal of period  $L$ . The Discrete Fourier Transformation [30] gives a characterization of the shape of the region. The Fourier coefficients are defined by:

$$F_n = \sum_{k=0}^{L-1} (i_k + j \cdot j_k) e^{-j \frac{2\pi kn}{L}} \quad \text{for } n = 0, \dots, L-1. \quad (5.10)$$

The Fourier descriptors correspond to the coefficients  $F_n$  for  $n > 0$ . The Fourier coefficient  $F_0$  is not used because it gives information about the location of the region. The magnitude and phase of Fourier descriptors give information about orientation and symmetry of the region. In general, only the magnitude  $|F_n|$  is used. Fourier descriptors are invariant under rotation. The Fourier descriptors of our example in Fig. 5.1a are illustrated in Fig. 5.4. The first pixel of the periodic function is  $(i_0, j_0) = (6, 10)$ . In case the region is a perfect circle,  $|F_n| = 0$  for  $1 < n < L$  because  $(i_k, j_k)$  represent a perfect sinusoid. In our example, the region is not a perfect circle, however, as we can see the Fourier descriptors are very small for  $2 < n < L$ .

Fourier descriptors can be extracted using command `Xfourierdes` of `ℳVIS` Toolbox.

### 5.2.4 Invariant moments

The statistical moments are defined by:

$$m_{rs} = \sum_{i,j \in \mathfrak{R}} i^r j^s \quad \text{for } r, s \in \mathbb{N} \quad (5.11)$$

where  $\mathfrak{R}$  is the set of pixels that belong to the region (see gray pixels in Fig. 5.1b). In this example, pixel  $(i = 4, j = 6) \in \mathfrak{R}$ . The parameter  $r + s$  corresponds to the order of the moment. The reader can demonstrate that the zero-th moments  $m_{00}$  is equal to the area  $A$  of the region. Moreover, the center of mass of the region is easily defined by:

$$\bar{i} = \frac{m_{10}}{m_{00}} \quad \bar{j} = \frac{m_{01}}{m_{00}} \quad (5.12)$$

The reader can compare this definition with (5.2). The coordinates of the center of mass can be computed using command `Xcentroid` of `ℳVIS` Toolbox.

The center of mass and statistical moments of higher order, however, are not invariant to the location of the region. This can be useful for detecting objects that must be in certain locations. Nevertheless, when objects of interest may be everywhere in the image we must use features that are invariant to the position. Using the center of mass, the central moments are defined. They are invariant to the position:

$$\mu_{rs} = \sum_{i,j \in \mathfrak{R}} (i - \bar{i})^r (j - \bar{j})^s \quad \text{for } r, s \in \mathbb{N}. \quad (5.13)$$

Other known moments that can be used are the well-known Hu-moments [95, 244]. These were developed using the central moments as follows:

$$\begin{aligned} \phi_1 &= \eta_{20} + \eta_{02} \\ \phi_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\ \phi_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\ \phi_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \\ \phi_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + \\ &\quad (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\ \phi_6 &= (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + \\ &\quad 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) \\ \phi_7 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] - \\ &\quad (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \end{aligned} \quad (5.14)$$

with

$$\eta_{rs} = \frac{\mu_{rs}}{\mu_{00}^t} \quad t = \frac{r+s}{2} + 1.$$

Hu-moments are invariant to translation, rotation and scale. That means that regions that have the same shape, but have a different size, location and orientation, will have similar Hu-moments.

In addition, there are similar invariant features, called Gupta moments, that are derived from the pixels of the boundary (instead of the region) [70]. They are invariant to translation, rotation and scale.

Sometimes, it is necessary to have features that are invariant to affine transformation as well (see Section 3.2.2). For this reason Flusser moments, *i.e.*, features invariant to translation, rotation, scale and affine transformation were derived from second and third order central moments [56, 236]:

$$\begin{aligned} I_1 &= \frac{\mu_{20}\mu_{02} - \mu_{11}^2}{\mu_{00}^4} \\ I_2 &= \frac{\mu_{30}^2\mu_{03}^2 - 6\mu_{30}\mu_{21}\mu_{12}\mu_{03} + 4\mu_{30}\mu_{12}^3 + 4\mu_{21}^3\mu_{03} - 3\mu_{21}^2\mu_{12}^2}{\mu_{00}^{10}} \\ I_3 &= \frac{\mu_{20}(\mu_{21}\mu_{03} - \mu_{12}^2) - \mu_{11}(\mu_{30}\mu_{03} - \mu_{21}\mu_{12}) + \mu_{02}(\mu_{30}\mu_{12} - \mu_{21}^2)}{\mu_{00}^7} \quad (5.15) \\ I_4 &= \frac{(\mu_{20}^3\mu_{03}^2 - 6\mu_{20}^2\mu_{11}\mu_{12}\mu_{03} - 6\mu_{20}^2\mu_{02}\mu_{21}\mu_{03} + 9\mu_{20}^2\mu_{02}\mu_{12}^2 \\ &\quad + 12\mu_{20}\mu_{11}^2\mu_{21}\mu_{03} + 6\mu_{20}\mu_{11}\mu_{02}\mu_{30}\mu_{03} - 18\mu_{20}\mu_{11}\mu_{02}\mu_{21}\mu_{12} \\ &\quad - 8\mu_{11}^3\mu_{30}\mu_{03} - 6\mu_{20}\mu_{02}^2\mu_{30}\mu_{12} + 9\mu_{20}\mu_{02}^2\mu_{21} \\ &\quad + 12\mu_{11}^2\mu_{02}\mu_{30}\mu_{12} - 6\mu_{11}\mu_{02}^2\mu_{30}\mu_{21} + \mu_{02}^3\mu_{30}^2)/\mu_{00}^{11}}{\mu_{00}^{11}} \end{aligned}$$



**Matlab Example 5.3:** In this example, we show how to measure invariant moments that can be used as a shape feature of objects of interest. We tested this approach on an X-ray containing 10 apples. We superimpose onto this image 6 rectangles the size of which is  $a \times b$  pixels (where  $b = 2a$ ). The rectangles are located in horizontal and vertical directions as shown in Fig. 5.5. Thus, we can simulate an input X-ray image containing apples and rectangles. The idea is to separate them. We see that the first Hu-moment can be used to effectively discriminate apples from rectangles.

Listing 5.3 : Detection using invariant moments

```
% AppleMoments.m
I = Xloading('N',1,4);
I( 50: 249,1500:1599) = 90;
I(1500:1599,1550:1749) = 90;

% X-ray of apples
```

```

I( 50: 449,1700:1899) = 90;
I(1800:1899,2050:2249) = 90;
I( 350: 749,2000:2199) = 90;
I(1050:1249,2000:2099) = 90;
imshow(I); hold on
J = and(I>60,I<110);
K = imerode(bwfill(J,'holes'),ones(11,11));           % segmentation
[L,n]=bwlabel(K,4);                                   % regions
X = zeros(n,7);
for i=1:n
    Ri = imdilate(L==i,ones(11,11));
    c = Xcentroid(Ri);                                % mass center
    X(i,:) = Xhugo(Ri);                               % Hu moments
    text(c(2)-55,c(1),sprintf('%4.0f',X(i,1)*1000),... % output
         'color','b','fontSize',12)
end
E = bwperim(K);
[ii,jj] = find(E==1); plot(jj,ii,'y.')               % display edges

```

The output of this code is shown in Fig. 5.5. In this example, command `Xcentroid` was used to compute the center of mass of each segmented object. The coordinates were used to print the first Hu-moment ( $\times 1000$ ) in blue. The Hu-moments are

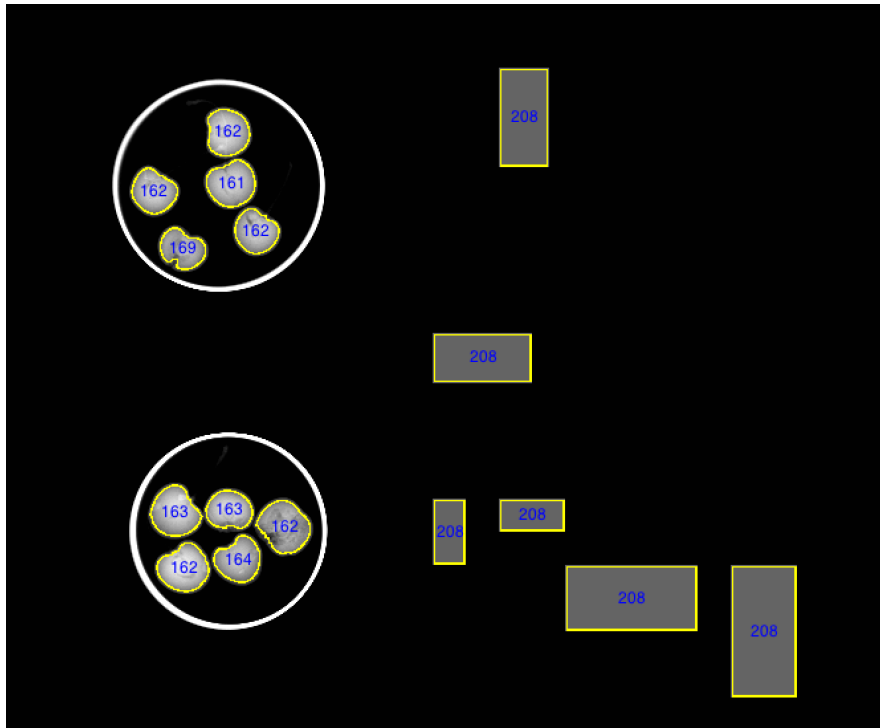


Fig. 5.5: First Hu moment ( $\phi_1$ ) of apples and rectangles. Since  $\phi_1$  for apples is approximately 163, and for these rectangles is 208, it is evident that this feature can be used to discriminate them from each other. [  $\rightarrow$  Example 5.3  $\blacktriangle$  ]

extracted by command `Xhugeo` of `Xvis` Toolbox. The reader can test Flusser and Gupta moments using commands `Xflusser` and `Xgupta`.  $\square$

### 5.3 Intensity features

These provide information about the intensity of a region. For gray value images, *e.g.*, X-ray images, there is only one intensity channel. The following features are computed using the gray values in the image, where  $x(i, j)$  denotes the gray value of pixel  $(i, j)$ .

#### 5.3.1 Basic intensity features

In this Section we summarize basic intensity features that can be easily extracted.

##### Mean gray value

The mean gray value of the region is computed as:

$$G = \frac{1}{A} \sum_{i,j \in \mathfrak{R}} x(i, j) \quad (5.16)$$

where  $\mathfrak{R}$  is the set of pixels of the region and  $A$  the area. A 3D representation of the gray values of the region and its neighborhood of our example is shown in Fig. 5.1. In this example  $G = 121.90$  ( $G = 0$  means 100% black and  $G = 255$  corresponds to 100% white).

##### Mean gradient in the boundary

This feature gives information about the change of the gray values in the boundary of the region. It is computed as:

$$C = \frac{1}{L} \sum_{i,j \in \ell} x'(i, j) \quad (5.17)$$

where  $x'(i, j)$  means the gradient of the gray value function in pixel  $(i, j)$  (see Section 4.4.1) and  $\ell$  the set of pixels that belong to the boundary of the region. The number of pixels of this set corresponds to  $L$ , the perimeter of the region. Using a Gaussian gradient operator in our example in Fig. 5.1, we obtain  $C = 35.47$ .

##### Mean second derivative

This feature is computed as: