Feature detection (edges, corners, curves and blobs)

1. Edge detection

Edges are image regions where there are large intensity transitions. Thus, the detection of edges consists in determining the points of an image where the intensity varies sharply, by using one of the two general criteria:

1. the amplitude of the first derivative is above a specified threshold value.
2. the second derivative changes sign (detection of zero crossing of the second derivative value)

The main IPT function for detecting edges is the *edge* function. This function provides several methods for estimating the image derivative values, using the two criteria presented. For some of the available methods (e.g. Sobel method) it is possible to find edges with predominantly horizontal or vertical direction or edges in any direction (isotropic operator).

The general syntax of the *edge* function is:

```
[g, t] = edge(f, 'method', parameters),
```

but it is also common to use the simplified syntax:

```
g = edge(f, 'method', parameters)
```

In the instructions presented before, f is the image to be processed, 'method' is one of the methods available for edge detecting and 'parameters' corresponds to additional parameters. The possible values for the 'method' parameter are: 'sobel', 'prewitt', 'Roberts', 'log' (Laplacian of a Gaussian), 'zerocross' and 'canny'. The number of parameters and values associated with the respective parameters is variable depending on the chosen method. More information can be found in MATLAB Help.

The output of this function is a binary matrix (*logical* format) with values 1 at points where edges were detected and 0 values in the remaining points. The output parameter t (*threshold*) is optional; if included, it contains the value that is compared with the result of the method in each pixel to accept or not this pixel as an edge.

2. Corner detection

The Harris method for corner detection is available in Matlab using the *detectHarrisFeatures* (Computer Vision Systems Toolbox) function. The syntaxes for these functions are

points=detectHarrisFeatures(I)   and   points=detectHarrisFeatures(I, Name, Value)

where I is the input image, and Name, value corresponds to a list of parameter pairs, where Name is a string with the name of the parameter and Value is the value assigned to parameter specified by Name. These parameters allow the specification of the quality of the detected corners ('MinQuality'), or the dimension of the Gaussian filter used for filtering the gradient of the image ('FilterSize').The output parameter, points, is a *cornerPoints* object (see MatLab Help - Computer Vision System Toolbox - Feature Detection and Extraction – Local Feature Detection).

3. Curve fitting

Ideally, the methods for edge detection aforementioned should only select points located at the edges. In practice, this ideal situation does not occur, and mainly due to noise, non-uniform illumination and image contrast variation, lines associated to edges become fragmented; moreover, points that cannot be associated to any edge of the image are often detected.
One way to overcome some of these drawbacks is to use the Hough transform to detect and connecting line segments. In theoretical terms, the Hough transform allows adjustment of any curve that can be parameterized (lines, circles, ellipses), but when the number of parameters is greater than 2, using the Hough transform is more complex.

3.1 Detection of straight lines

The IPT provides three functions for detecting straight lines using Hough transform, namely, the *hough*, *houghpeaks* and *houghlines* functions.
The function *hough* calculates the Hough transform of a binary image, enabling the detection of straight lines. The syntax of this function is:

```
[H, theta, rho] = hough(BW)          or

[H, theta, rho] = hough(BW,param1,val1,param2,val2)
```

where BW is the input binary image and param1, param2 are the two possible parameters and val1 and val2 their values (see details in MATLAB Help). The function returns the Hough matrix, H, and two vectors (theta and rho variables) containing the specific values of the parameters (ρ,θ) (from the representation in polar coordinates of the line points) used in calculating H.
The *houghpeaks* function detects the maximum values (peaks) occurring in the matrix H. The syntax of this function is:

```
peaks = houghpeaks(H,numpeaks)
```

where 'H' is the matrix which resulted from the *hough* function, and numpeaks enables the user to specify the number of maxima to identify. peaks is an output array Q×2 (Q ranges from 0 to numpeaks) whose rows contain the positions in H of the maximum values detected.
The *houghpeaks* function can also be used as indicated below:

```
peaks = houghpeaks(...,param1,val1,param2,val2)
```

when you want to specify any of the possible parameters and their values (see details in MATLAB help).

Finally, the *houghlines* function extract line segments contained in the BW image associated with particular elements of the matrix H. This function can be used in the forms:

```
lines = houghlines(BW,theta, rho, peaks)
lines = houghlines(...,param1,val1,param2,val2)
```

where theta and rho are returned vectors by the *hough* function, and peaks is the array returned by the *houghpeaks* function.

The *houghlines* function has as output the variable lines, that is a vector whose length is equal to the number of connected line segments which have been identified. Each vector element is a structure with a set of fields that specify the start and end points of the line segment and the parameters (ρ, θ) of the line to which the segment belongs.

3.2 Detection of circles

A circle may be characterized by three parameters: the two center coordinates and the radius. Accordingly, the representation of Hough curves for this kind of curve is carried out in a three-dimensional space. To avoid working with three-dimensional arrays, implementations of the Hough transform for circles usually use a value or a range of limited values for the radius and use only the center coordinates as variables in the Hough array, which under these conditions becomes two-dimensional.
The *imfindcircles* function implements the Hough transform using a two-dimensional version. The simplest way to use this function is the instruction

```
centers=imfindcircles(A, radius)
```

where A is the input image, radius is value of the radius of the circles to find and centers is an array with the coordinates of the centers of the circumferences detected.

In case if you want to use a range of values, we can draw the following call

```
[Centers, radii] = imfindcircles (A, radiusRange)
```

or

```
[Centers, radii, metric] = imfindcircles (A, radiusRange)
```

where radiusRange lets you specify the range of radii for the circles, radii is a vector with the rays associated with each center, and metric contains the value of the accumulator (ordered in decreasing order) that gave rise to each of the circumferences detected.
Each of the above syntaxes admits a set of input parameters in the form Name, Value, which can be found in MatLab Help (Image Processing Toolbox Image Analysis - Object Analysis). Contrary to the function for detecting straight line that takes as input a binary image, *imfindcircles* function does not require an outline image. By default, this function detects light objects on a dark background. The pair of parameters 'ObjectPolarity', 'Dark', should be used for detecting dark objects.

4. Blob detection

The detection of blobs in images can be performed using a variety of operators, among which we highlight the LoG operator (Laplacian of Gaussian). The kernel of a LoG operator can be generated using the *fspecial* function using the following syntax:

```
H = fspecial ('log', hsize, sigma)
```

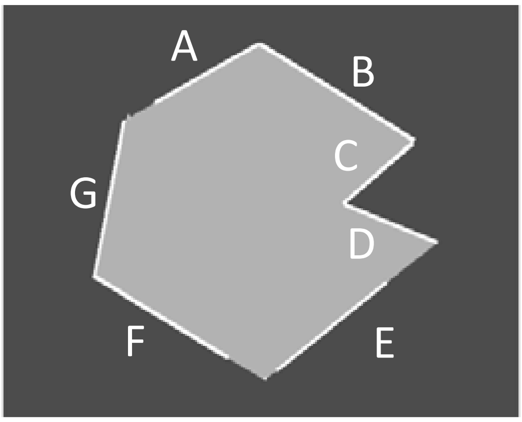
where hsize defines the size of the matrix H and sigma is the value of the standard deviation of the Gaussian filter.
The use of this type of operator requires that the adaptation of the sizes of both of the filter (hsize) and the value of sigma to the size of the blobs to detect, often requiring the use of multiple filters with variable size whose responses should then be combined.

Work proposal

1. (Pen and paper) Compute the magnitude of the Sobel operator on the highlighted points, and comment the results.

8	1	22	29	9	1	1
9	21	21	25	22	5	3
7	23	22	3	5	6	7
8	26	21	2	1	4	9
10	23	24	20	2	5	10
11	2	25	21	26	2	5
5	3	5	21	25	0	1
4	5	9	10	11	13	6

2. (Pen and paper) The following figure contains line segments (marked in white) that were detected through the application of the Hough transform. Their corresponding values ρ and θ, computed via the Hough matrix, are indicated in the following table for some of the segments. Complete the table, justifying your results using a drawing illustrating the ρ and θ measurements on the image space.



θ	11		-59	-68	60	51	
ρ	222	164	-293	-89	253	720	588
Segment		B					C

3. Figure A ("blobs2.tif") represents an original image that was subjected to a sequence of operations aimed at detecting the contours of the darker objects. Figure B shows the result of the enhancement of the edges using a Sobel operator (after smoothing the original image) and Figure C the edges that resulted from the figure B.

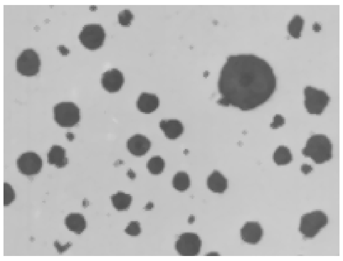


Figure A

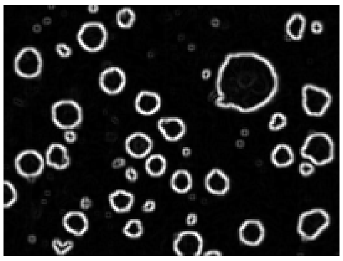


Figure B

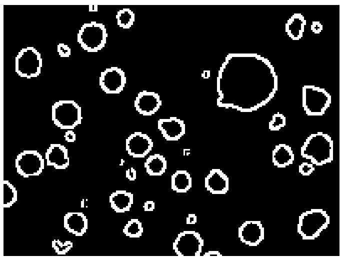


Figure C

- **A.** Write a set of instructions for obtaining the images that are shown in Figures B and C. The original image was previously smoothed with a Gaussian filter 7x7 (sigma = 1).
- **B.** Add instructions for detecting the edges using the Canny method. Evaluate the influence of changing the values of the filter parameters, and in particular the value of the standard deviation of the Gaussian filter.
- **C.** Compare the result obtained in the preceding paragraph with edges that can be obtained using the Marr-Hildreth method.

4. The image shown in Figure 2 resulted from the application of a processing sequence to the image of Figure 1 ("root.tif"), mainly consisting of binarization followed by thinning. The image in Figure 2 is available in the file "root_thin.tif".

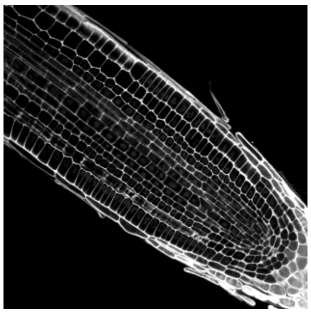


Figure 1

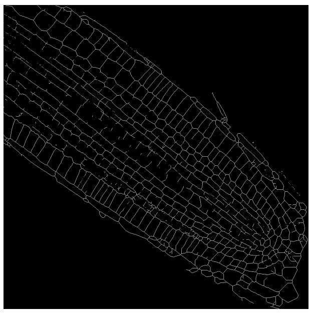


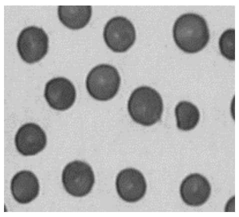
Figure 2

- **A.** Use the function detectHarrisFeatures for detecting the intersection points in the image of Figure 1.
- **B.** Repeat A. with the image in Figure 2.

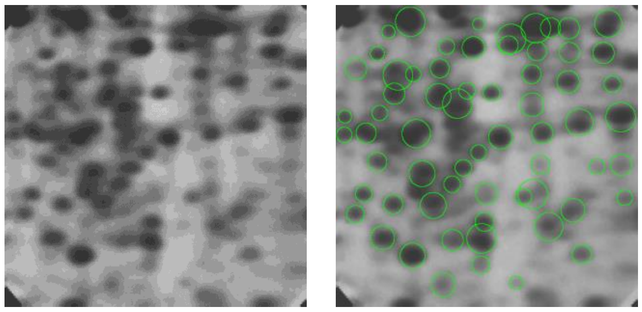
5. Consider once again the image of Figure 2

- **A.** Use the Hough transform to identify and visualize the lines visible in the image.
- **B.** Change the code in order to restrict the line detection only to lines with the root dominant direction.
- **C.** Change the code once again to detect only the 5 largest lines (corresponding to the 5 highest elements of the Hough matrix).

5. Use the following image ("celulas_2.tif") and Hough transform for detecting the contours of the cellules (you can consider that an approximate value for the cellular diameter is less than 40 pixels).



6. The image of the following figure ("BlackBlobs.tif") needs to be processed for detecting the black areas and get information related with their size.



Note: For representing a circle (center (*cx,cy*); radius *radius*; colour green 'g') you can use the following set of instructions or use the function *viscircle* :

```
imshow(var_img);
hold on;
t=linspace(0,2*pi);
plot(radius*cos(t)+cx, radius*sin(t)+cy,'g')
```