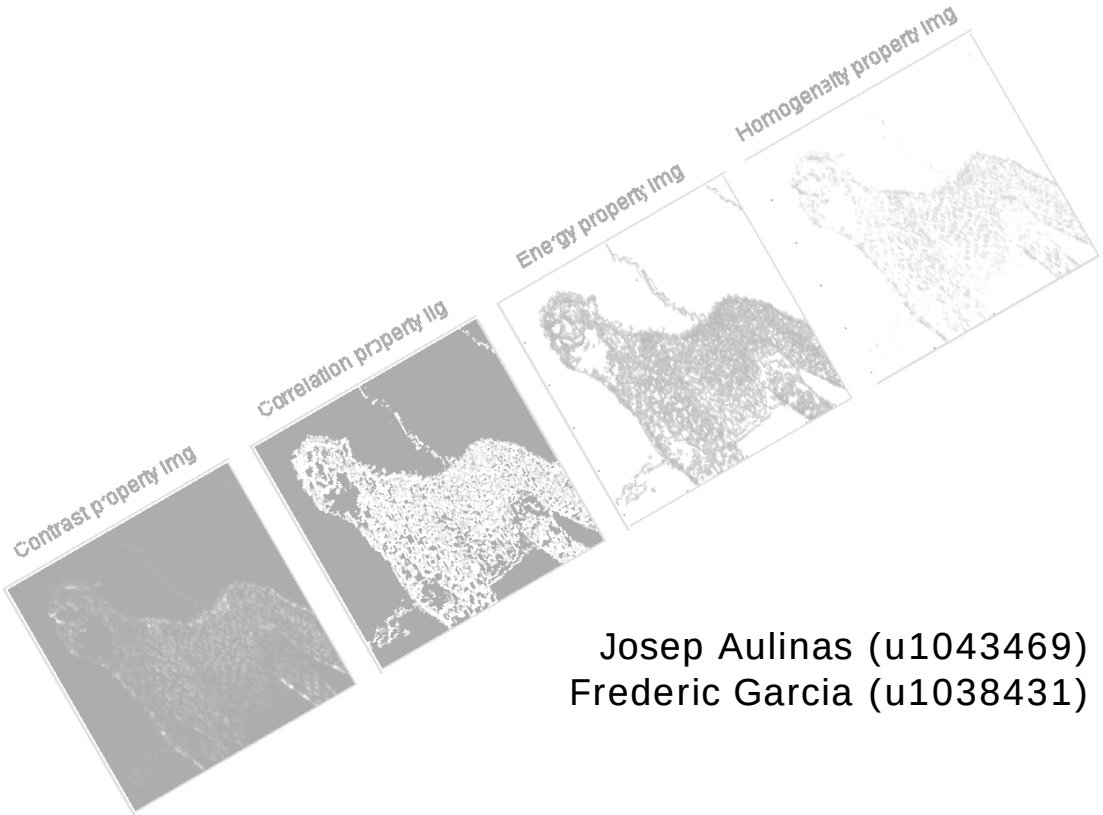
Vibot

Departament d'Electrònica, Informàtica i Automàtica

Universitat de Girona

SCENE SEGMENTATION AND INTERPRETATION

Coursework 1: Image Characterization



Josep Aulinas (u1043469)
Frederic Garcia (u1038431)

1. Introduction and problem definition

At some point in image processing a decision is made about which image points or regions of the image are relevant for further processing. In order to distinguish regions in images it is necessary to analyse their characteristics. Digital image characteristics are mainly three: colour, shape and texture. A good descriptor for an image, normally, integrates some information from all three features.

Colour is related to human's visual perception and also to several physic specifications. The colour of an object in an image depends both on physics and on perception, and so it is considerably complex to characterise it.

Shape refers to the outline or the appearance of something in the image that describes its geometrical properties. This characteristic can be used to fix patterns to classify objects with similar shapes, hence define different groups of elements with similar shapes.

Texture could be understood as the appearance of a specific surface or part of an object in an image. In the same image different textures can be identified defining some regions. Combining this characteristic with some statistic tools important information can be extracted from the image.

This assignment presents results and discussion about a set of computations implemented on several images. These computations were programmed to generate texture descriptors using co-occurrence matrices and energy filters. The results helped to determine the correctness of the input parameters for the co-occurrence matrix and the energy filters depending on the texture type.

2. Algorithm analysis

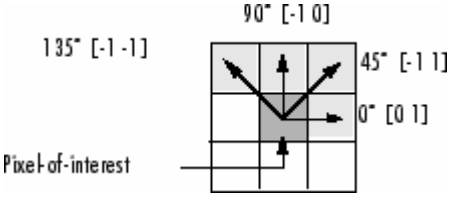
The main propose of the exercise is to implement basic principles on image texture analyses using Matlab, and more precisely co-occurrence matrices and energy filters combined with some statistics, such as homogeneity, contrast and uniformity.

Co-occurrence matrices

A co-occurrence matrix contains the frequency of a certain pair of pixels repetition in an image. In order to compute a co-occurrence matrix it is necessary to know the following values:

- Number of grey levels: a greyscale image contains 256 grey levels, which means a high computational cost because all possible pixel pairs must be taken in account. The solution is to generate the matrix reducing the number of greyscales, and so the number of possible pixel combinations. The co-occurrence matrix is always square with the same dimensionality as the number of grey-levels chosen. This value is often set to eight.

- Angle. Similarly to the distance it is necessary to define the direction of the pair of pixels. The most common directions are 0, 45º, 90º, 135º, and its symmetric equivalents.



- Distance between pixels: the co-occurrence matrix stores the number of times that a certain pair of pixels is found in the image. Normally the pair of pixels are just neighbours, but it could also be computed the matrix analysing the relation between non consecutive pixels. Thus a distance between pixels must be previously defined.

Example of a co-occurrence matrix with eight grey levels, computed using one for distance between pixels and zero degrees for the direction.

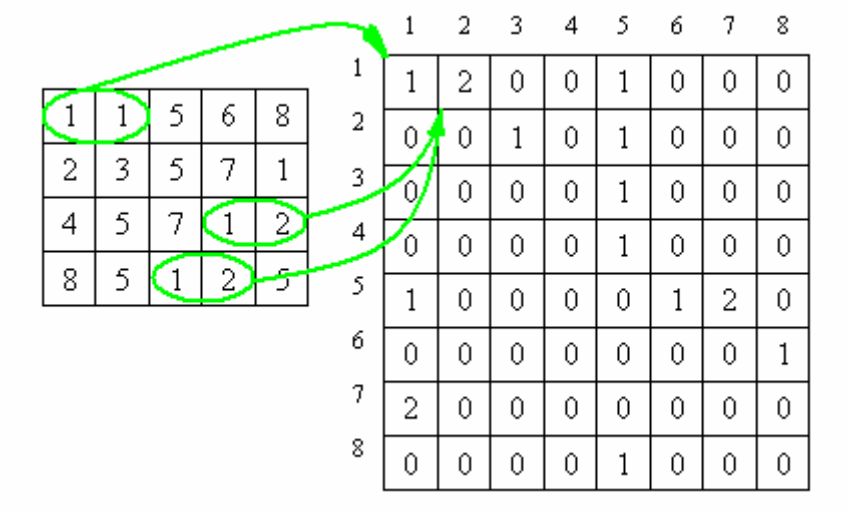


Figure 2.1. Example of co-occurrence matrix process.

The idea is to calculate the co-occurrence matrix for small regions of the image and then use this matrix to find statistic values, for instance homogeneity, contrast, correlation and energy.

$$\text{Contrast} = \sum_{i,j=0}^{N-1} (i-j)^2 \times m_{ij} \qquad \text{Homogeneity} = \sum_{i,j=0}^{N-1} \frac{m_{ij}}{1+|i-j|}$$

$$\text{Correlation} = \sum_{i,j=0}^{N-1} \frac{(i-\mu)\times(j-\mu)\times m_{ij}}{\sigma^2} \qquad \text{Uniformity} = \sum_{i,j=0}^{N-1} m_{ij}$$

Figure 2.2. Value descriptors.

Energy filters

Energy filter are computed in two steps, a law mask convolution is first applied over the image and then statistic measures are obtained from the convolution, such as mean, standard deviation and absolute mean.

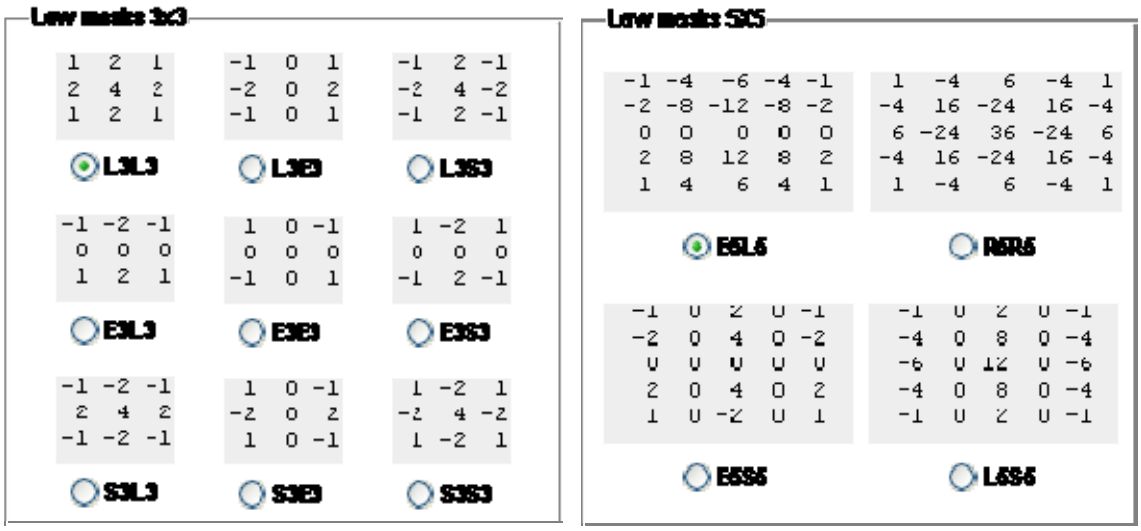


Figure 2.2. Energy filters.

The law masks are composed mainly for L (level), E (edge) and S (spot) components. And the information obtained after convolving them with the image is then used to find the mean image or standard deviation one.

Mean $= \mu = \frac{\sum_{i=1}^{N} y_i}{N}$ Standard deviation $s = +\sqrt{\sum_{i=1}^{N} \frac{(y_i - \mu)^2}{N}}$ Absolute mean $= \frac{\sum_{i=1}^{N} |y_i|}{N}$

Figure 2.3. Value descriptors.

3. Design and implementation of proposed solution

As is said in the coursework sheet, the main goal is to be able to create output images corresponding to different texture descriptors. To achieve this goal, an interface using Matlab guide has been implemented which includes both the co-occurrence matrices and energy filters at local level producing descriptors for each pixel.

Both techniques have been implemented and can be compared through the checkbox showed in the figure 3.1.

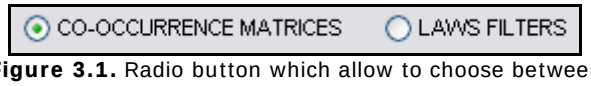


Figure 3.1. Radio button which allow to choose between co-occurrence matrices and energy filters.

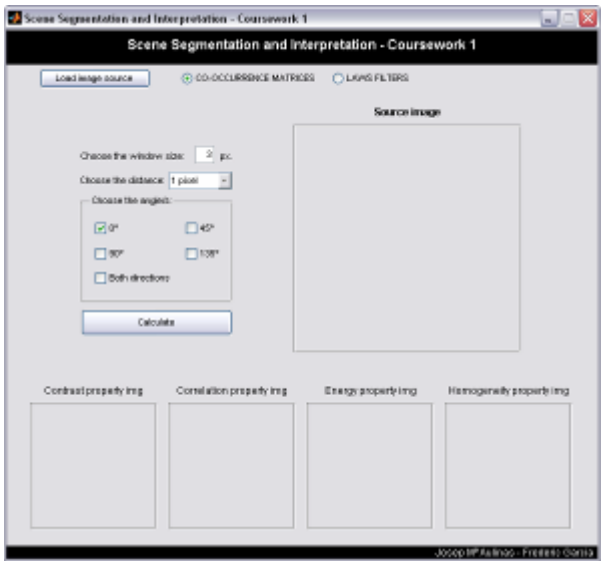


Figure 3.2a. Initial interface for co-occurrence matrices.

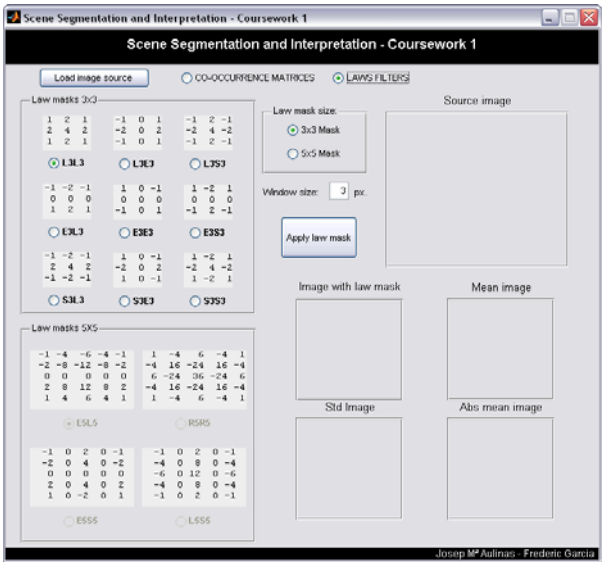


Figure 3.2b. Initial interface for energy filters.

The local level which allows producing descriptor for each pixels can be defined by the edition box which is named as '*Choose the window size*' (Figure 3.3). The smallest value entered, the most local level will be the descriptor and also, the more computation time will be taken.

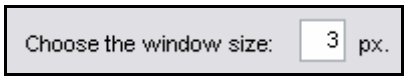
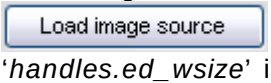


Figure 3.3. Edition box which allow to enter the window size for the descriptor size.

According to the former technique, co-occurrence matrices, the code takes the values entered by the user which corresponds with: window size, distance between pixels to be examined and the angle or angles. When these values are obtained, the code is relatively simple:

```
D = get(handles.pm_distance,'Value'); % Distance
offset = [];
if get(handles.cb_0,'Value') % Angle 0º selected
    offset = [offset; 0 D];
end
if get(handles.cb_45,'Value') % Angle 45º selected
    offset = [offset; -D D];
end
if get(handles.cb_90,'Value') % Angle 90º selected
    offset = [offset; -D 0];
end
if get(handles.cb_135,'Value') % Angle 135º selected
    offset = [offset; -D -D];
end
[w,h] = size(handles.gVar.I);
w_size = str2double(get(handles.ed_wsize,'String'));
inc = floor(w_size/2);
tic % starts a stopwatch timer
for i=1:w-w_size
    for j=1:h-w_size
        if get(handles.cb_both,'Value')
            im_co =
graycomatrix(handles.gVar.I(i:i+w_size,j:j+w_size),'NumLevels',8,'Offset',offset,'Symmetri
c',true);
        else
            im_co =
graycomatrix(handles.gVar.I(i:i+w_size,j:j+w_size),'NumLevels',8,'Offset',offset);
        end
        st = graycoprops(im_co,{'contrast','correlation','energy','homogeneity'});
        st_contrast(i+inc,j+inc) = mean(st.Contrast);
        st_correlation(i+inc,j+inc) = mean(st.Correlation);
        st_energy(i+inc,j+inc) = mean(st.Energy);
        st_homogeneity(i+inc,j+inc) = mean(st.Homogeneity);
    end
end
sec=toc; %prints the elapsed time since tic was used
```

Where,

- '*handles.pm_distance*' is the interface component (list box) which has the distance between pixels to be examined entered by the user.
- '*handles.cb_0*' is the interface component (radio button) which indicates if the user wants to examine the neighbour pixels which is situated at 0º (if it is enabled).
- '*handles.cb_45*' is the interface component (radio button) which indicates if the user wants to examine the neighbour pixels which is situated at 45º (if it is enabled).
- '*handles.cb_90*' is the interface component (radio button) which indicates if the user wants to examine the neighbour pixels which is situated at 90º (if it is enabled).
- '*handles.cb_135* is the interface component (radio button) which indicates if the user wants to examine the neighbour pixels which is situated at 135º (if it is enabled).
- '*handles.cb_both* is the interface component (radio button) which indicates if the user wants to examine the neighbour pixels in both directions, i.e. 0º and 180º.
- '*handles.gVar.I*' is the image matrix previously loaded using the button 

- '*handles.ed_wsize*' interface component (edit text) which indicates the dimension of the window size entered by the user to generate the local level descriptor.

To obtain the resulting descriptor image, can be use the Matlab function '*graycomatrix*' which returns the gray-level co-occurrence matrix from the image. The

parameters are the region of the source image which is in the window size selected, the number of levels of grey colors (8 which indicates 2^8, 256 levels) and the offset which is the angle or angles to be explored.

After obtaining the co-occurrence matrix (im_co), it is calculated a set of statistical values such as contrast, correlation, energy and homogeneity. These values are obtained using the Matlab function '*graycoprops*'. This function returns the statistical value or values, depending on the offset. If there is more than one angle, it returns a vector with the same length of the number of angles to be computed. The resulting value is the mean of these values.

With these statistical values are generated the resulting images to be examined as is shown in the next figure:

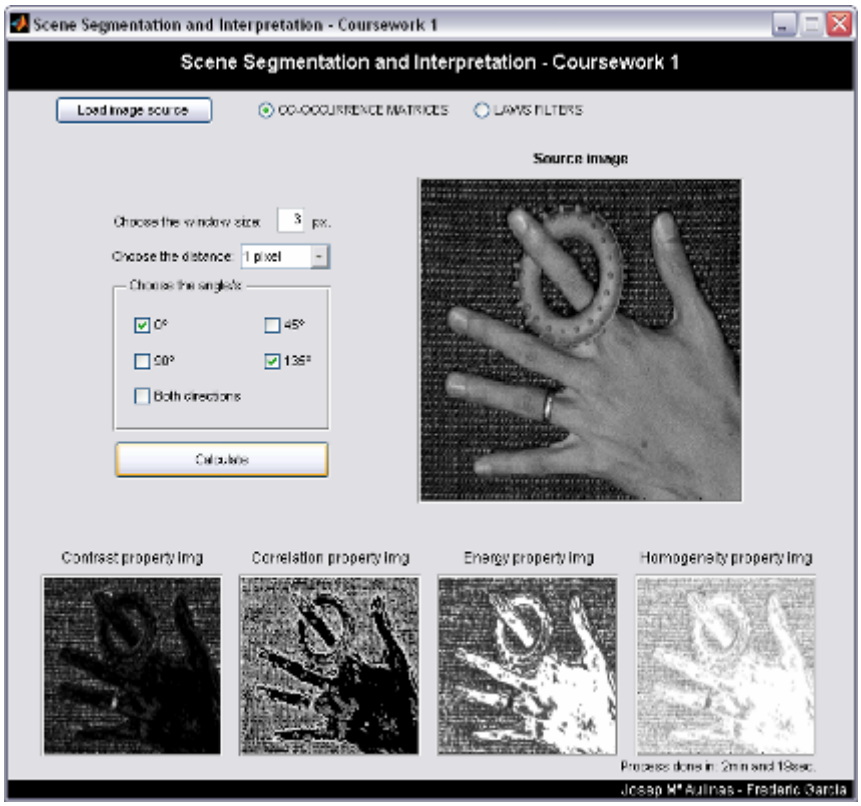
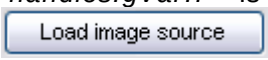


Figure 3.4. Example of the co-occurrence matrices execution.

According to the latter technique, energy filters, the code also takes the entered values and selections by the user, such as the law mask and another time, the window size. The code implementation is:

```
tic % starts a stopwatch timer
newImg = filter2(handles.gVar.mask,handles.gVar.I); % Apply the mask
[w,h] = size(newImg);
w_size = str2double(get(handles.ed_wsize2,'String'));
inc = floor(w_size/2);
for i=1:w-w_size
    for j=1:h-w_size
        st_mean(i+inc,j+inc) = mean2(newImg(i:i+w_size-1,j:j+w_size-1));
        st_std(i+inc,j+inc) = std2(newImg(i:i+w_size-1,j:j+w_size-1));
        st_absmean(i+inc,j+inc) = mean2(abs(newImg(i:i+w_size-1,j:j+w_size-1)));
    end
end
sec=toc; %prints the elapsed time since tic was used
```

Where,

- '*handles.gVar.mask*' contains the law mask according with the radio button checked by the user.
- '*handles.gVar.I*' is the image matrix previously loaded using the button 

- '*handles.ed_wsize2*' interface component (edit text) which indicates the dimension of the window size entered by the user to generate the local level descriptor.

The main function is '*filter2*' which applies the law mask to the source image. After this process it is calculated some statistical values such as the mean, the standard deviation and the absolute mean on the resulting image of the *filter2* execution according with the window size entered by the user.

With these statistical values are generated the resulting images to be examined as is shown in the next figure:

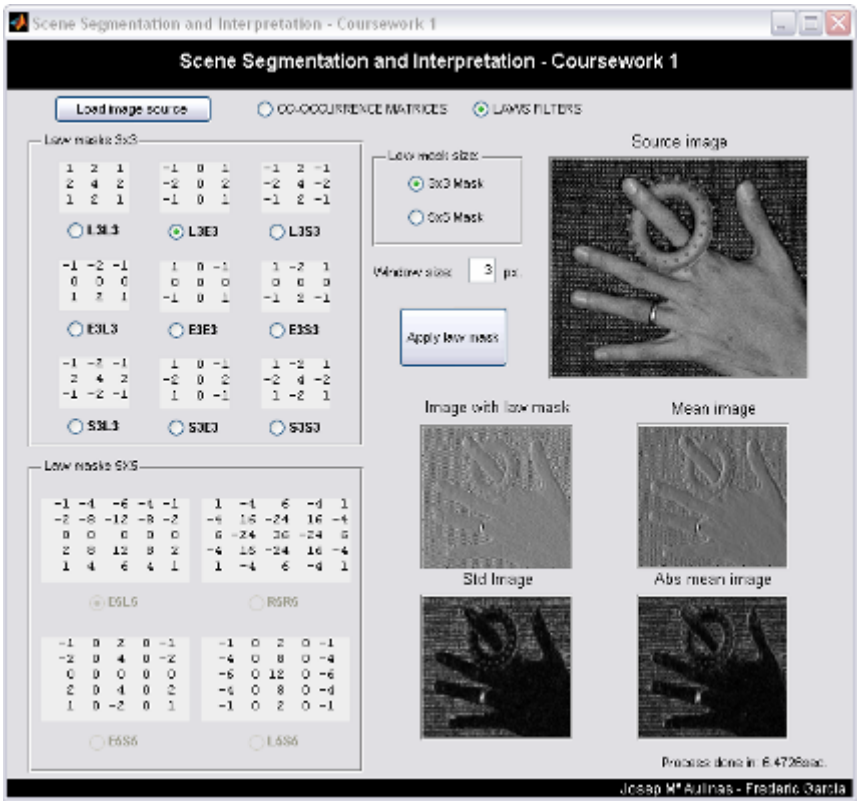


Figure 3.5. Example of the energy filters execution.

The rest of the Matlab code which appears in the '*ssi_c1fig.m*' file is related with the interface manipulation and the user input validation.

4. Experimental section and results analysis (speed, quality, etc)

A set of eight different images were provided to study the problems and possible improvements.

Figure 4.1. Images provided for the current lab exercise.

Concerning to the input parameters that must be previously decided in order to extract information from an image, it was necessary to think carefully first which could return the best result. These parameters were: number of grey levels, angle and distance between pixels, and mask dimension. The number of grey levels was set by default to eight. The angle was chosen according to the direction of the main edges in the image. The distance between pixels was chosen depending on the width of the edges. And the mask dimension was chosen taking in account the uniformity of the image, in other words, the size of the objects or patterns in the picture.

Before analysing each picture choosing its best parameters, a series of executions where conducted in order to find some trends, to get some previous knowledge (view annex).

Image feli.tif

In this image, *feli.tif* (Figure 4.2 left) the direction is not as important as the mask dimension. The tiger has a regular pattern which is bigger than 9x9 pixels, being in some cases closer to 21x21. For this reason a proper dimension of the mask was selected at 15x15 pixels.

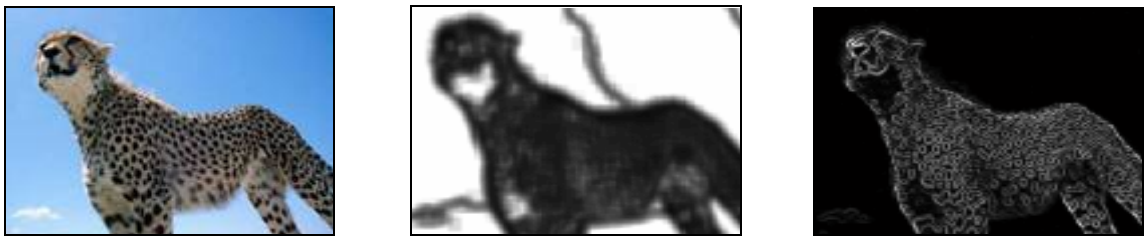


Figure 4.2. Image feli.tif (left); Co-occurrence Energy image (middle); Std. deviation L3L3 mask (right)

If the purpose of the characterization is to find a region in which a pattern keeps repeating then the energy image computed using a mask of 15x15 is a good approach (computation time: 5 min and 43 sec), accepting that the tiger shape becomes slightly diffused (Figure 4.2 middle). However, if the purpose is to get a proper edge image then it will be better to reduce the mask length or use a low mask, for instance computing the standard deviation (Figure 4.2. right) on the result given by a L3L3 mask (computation time: 11.6827 sec).

Image pingpong2.tif

This image, *pingpong2.tif* (Figure 4.3 left), contains some important regions, such as the table, the wall and the arm. The arm colour is homogeny, except from its shadows. It happen exactly the same with the table colour, while the wall texture is quite rough. These facts suggest that a good mask dimension should be small, for instance 5x5 pixels. The distance between pixels should be preferably high, for example 5 pixels, because the textures are quite homogeny, and so it is more likely to detect a change after some pixels instead of within the closest neighbours.

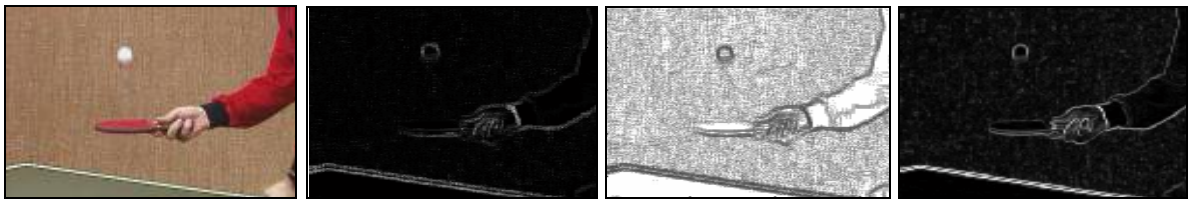


Figure 4.3. Image pingpong2.tif (left); Co-occurrence Contrast image (middle-left); Co-occurrence Homogeneity Image (middle-right); Std. deviation L3L3 mask (right)

Computing the co-occurrence matrix (computation time: 4min and 28sec) the result obtained suggests that the Contrast statistic (Figure 4.2 middle-left) and the Homogeneity statistic (Figure 4.3 middle right), as well as the standard deviation (computation time: 8.9275sec) of the L3L3 law mask (Figure 4.3 right) are correct approaches for edge detection or for regions definition. However, the wall texture could not be correctly sorted out.

Image hand2.tif

This image, *hand2.tif* (Figure 4.4 left), contains three clear textures: the skin of the hand which is considerably regular, the blue ring and the background which is composed by a regular and translation invariant pattern. The background pattern is repeated almost every five pixels, for this reason a mask of 5x5 pixels is used in order to characterize this texture. In addition, the main directions of this pattern are zero and ninety degrees, and so these are the direction between pixels used in the co-occurrence matrix. The resulting correlation image (Figure 4.4 middle, computation time: 3min and 36sec) returns a possible texture descriptor for the background.

Figure 4.4. Image hand2.tif (left); Co-occurrence correlation image (middle); absolute mean of the L3L3 mask (right)

A good descriptor for this image is the absolute mean of the law mask L3S3 (Figure 4.4 right, computation time: 16,4484sec). In this image the three textures can be easily observed.

Image dubuisson2.tif and mosaic8.tif

The images *dubuisson2.tif* and *mosaic8.tif* contain four different textures. All these textures have in common its random content, but some of them are translation invariant, while some others have some clear shapes and main directions.

The *dubuisson2.tif* (Figure 4.5 left) left textures contain a pattern which follows a clear direction, for instance, the top one have some vertical lines and the bottom one have a sort of an horizontal direction. For this reason it seems that the best texture descriptor could be the one that computes the co-occurrence matrix at zero and ninety degrees. Another texture descriptor could be found in the bottom right texture, which is randomly

granular (Figure 4.5 right), using the energy image indicator (computation time: 3min and 42sec). Also an E3L3 law mask produces a significant result (Figure 4.5 right, computation time 17,5372 sec).

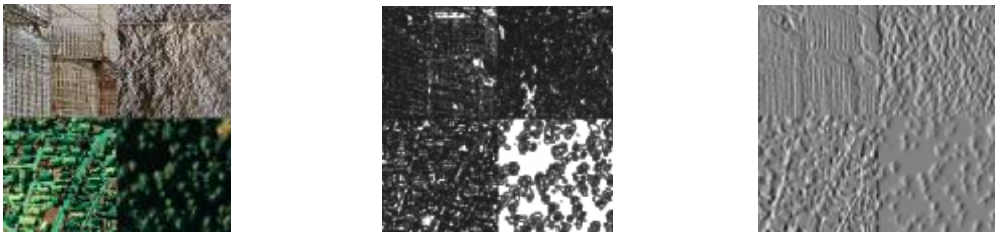


Figure 4.6. Image dubuisson.tif (left); Co-occurrence energy image 5x5 (middle); E3L3 law mask mean image (right)

For example, the top right texture in the image *mosaic8.tif* (Figure 4.6 left) is the easiest to characterize using the contrast image for the co-occurrence matrix and a mask dimension of 9x9 (Figure 4. middle, computation time: 3min and 32sec) or 5x5 (Figure 4. right, computation time: 3min and 41sec) which returns more accurate contours.

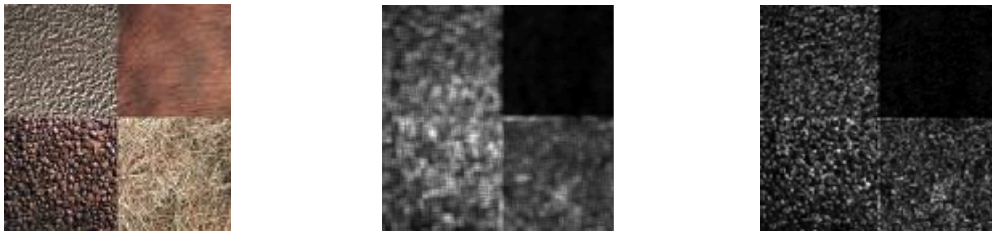


Figure 4.6. Image mosaic8.tif (left); Co-occurrence contrast image 9x9 (middle); Co-occurrence contrast image 5x5(right)

Image sq5.tif, zhu1.tif and t10.tif

According to previous simulations for images like *sq5.tif* (Figure 4.7), *zhu1.tif* (Figure 4.8) and *t10.tif* (Figure 4.9) it seems that an acceptable edge detector could be a mask of 5x5 pixels and a distance between pixels of 5 pixels. All these images contain random and translation invariant textures, such as the background grass for the squirrel's image or the Koala's skin. This fact makes considerably complicated to extract texture information from these images.

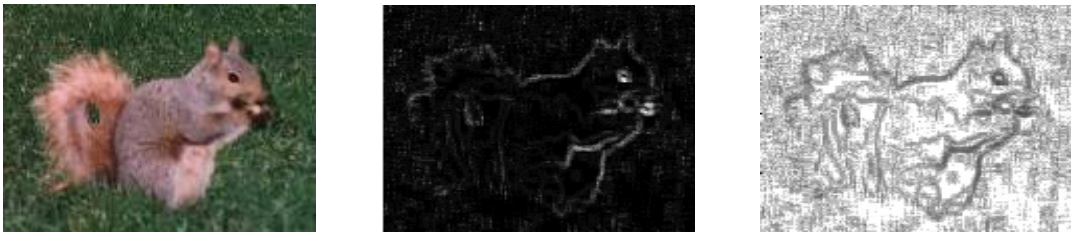


Figure 4.7. Image sq5.tif (left); Co-occurrence contrast image (middle); Co-occurrence homogeneity image(right)

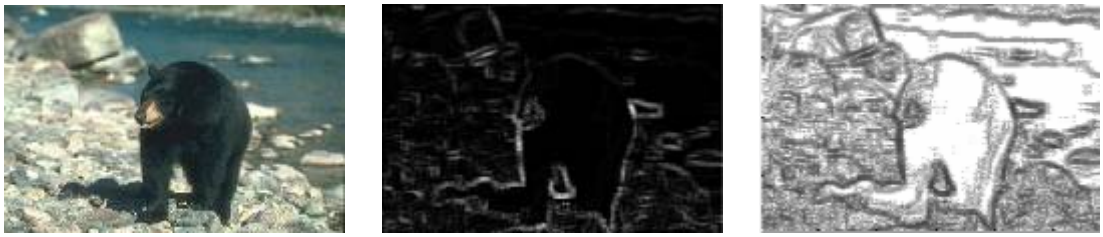


Figure 4.8. Image zhu1.tif (left); Co-occurrence contrast image 9x9 (middle); Co-occurrence contrast image 5x5(right)

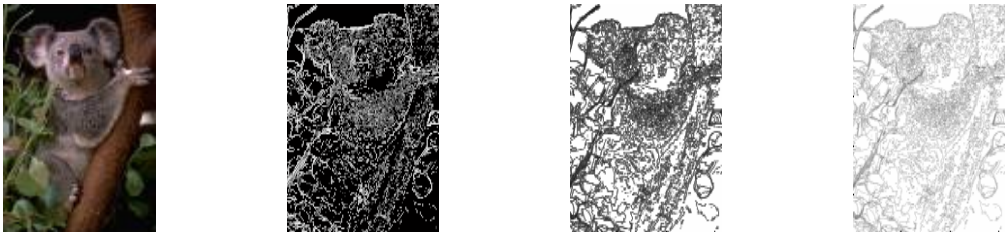


Figure 4.9. Image t10.tif (left); Different results in which no clear texture descriptor is visible.

These thin textures suggest that the mask dimension should be small in order to find a proper texture descriptor. However, any acceptable indicator could be found for these images.

5. Organization and development of the coursework

The Gannt diagram was first though as follows, but some of the tasks could not keep the schedule mostly for computational reasons.

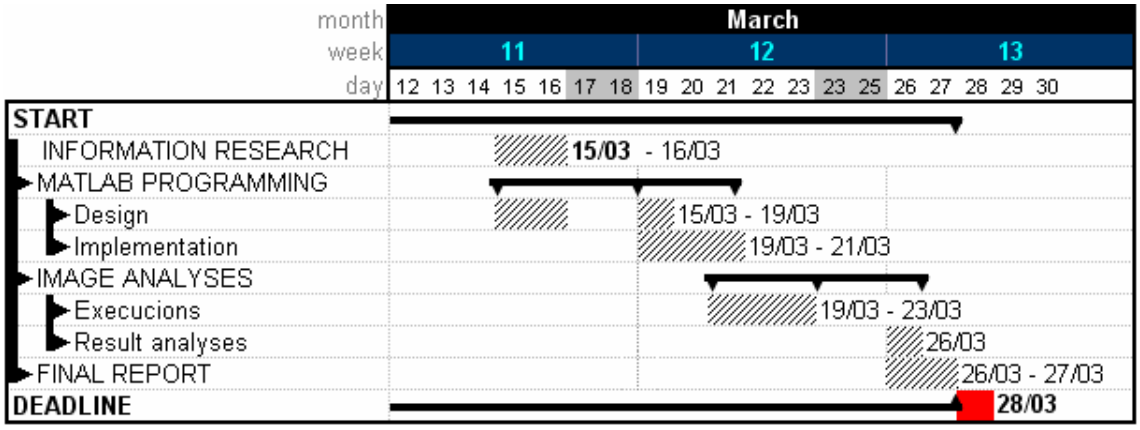


Figure 5.1. Gannt diagram.

The following flow diagram shows the sequence of tasks developed for this lab exercise:

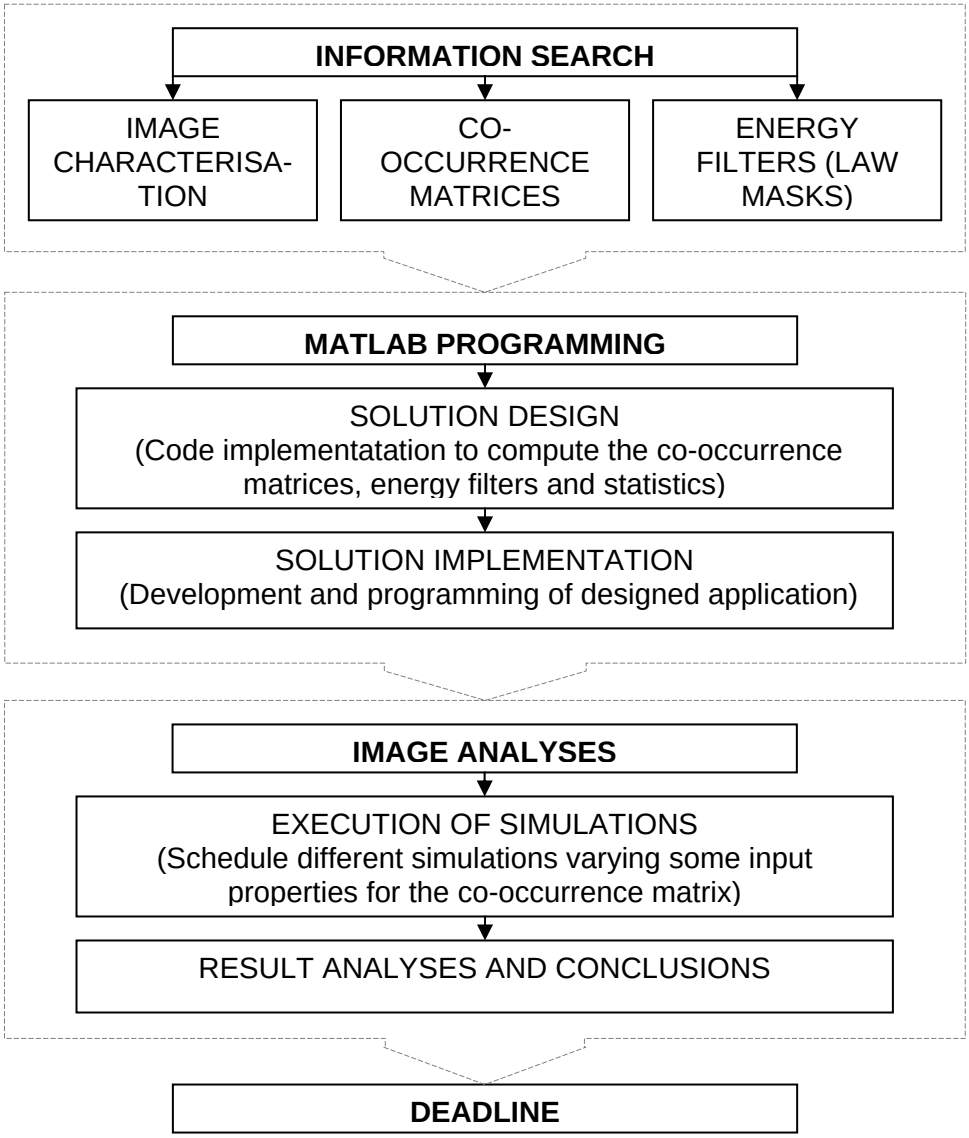


Figure 5.2. Sequence of tasks developed for this lab exercise.

A set of computations were programmed for each image, for each simulation only one property of the co-occurrence matrix was changed. These simulations produced important data which then was used to analyse and extract some conclusions of the application developed and so related with texture descriptors.

SIMULATIONS	Grey levels	Direction (degrees)	Distance (pixels)	Mask dimension (pixels)
1.	8	0	1	3
2.	"	90	"	"
3.	"	0, 45, 90, 135	"	"
4.	"	"	"	9
5.	"	"	"	15
6.	"	"	"	25
7.	"	"	2	"dependent"
8.	"	"	4	"

Figure 5.3. Comparison between simulations.

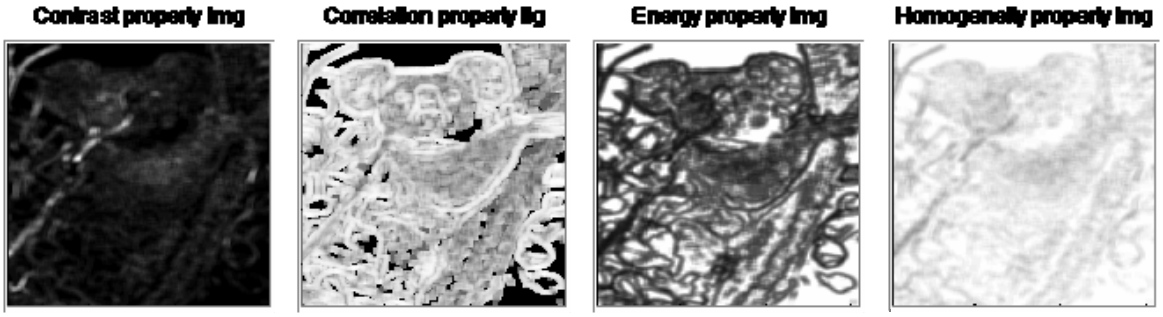
6. Discussion

In this paper some texture descriptors have been analysed, such as co-occurrence matrices and energy filters. The results suggests that an appropriate overall conclusion should say that it is considerably complex to characterize an image only looking at its textures, and so, that a texture descriptor is only a part of the information which then can be used to extract some features from the image.

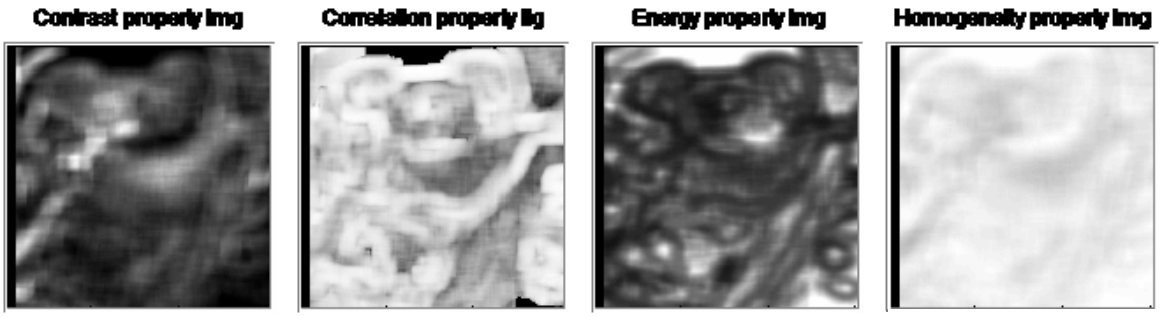
Another fact to take into account is the dependence between the problem to solve and its solution, therefore if the image's texture is a clear pattern, the mask dimensions to characterise it will be according to the pattern properties. Then, it seems not to be easy to implement a texture descriptor software for any type of images.

Annex

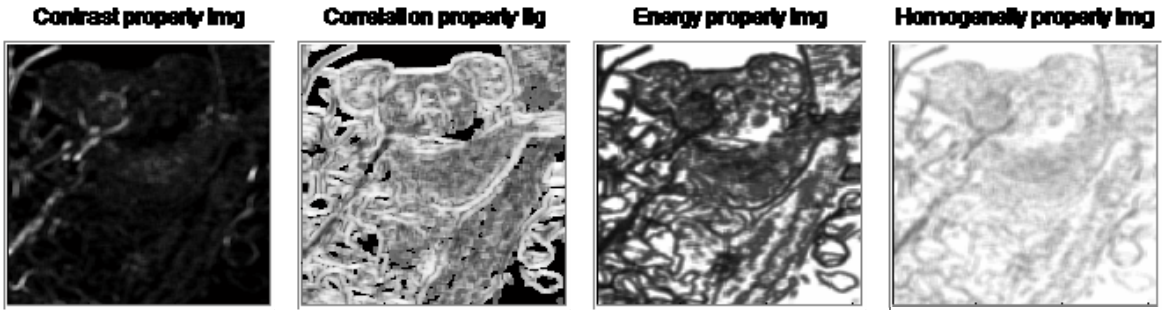
Window size 3, distance 1, angle 0



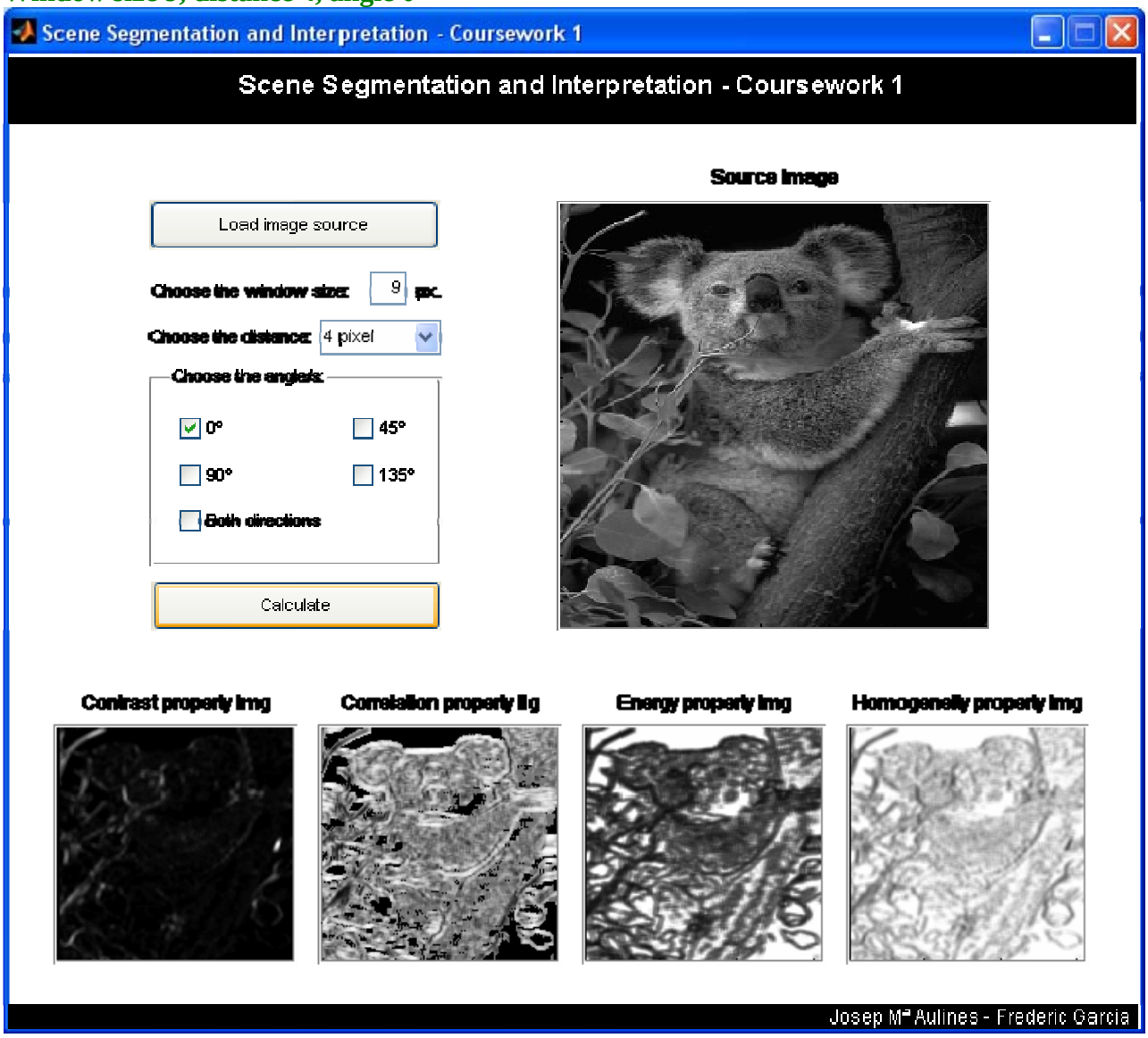
Window size 25, distance 1, angle 0

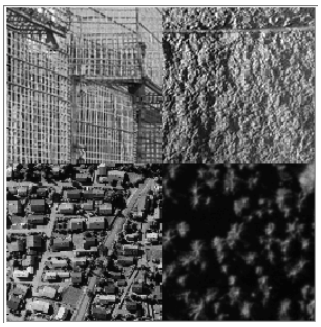


Window size 9, distance 2, angle 0

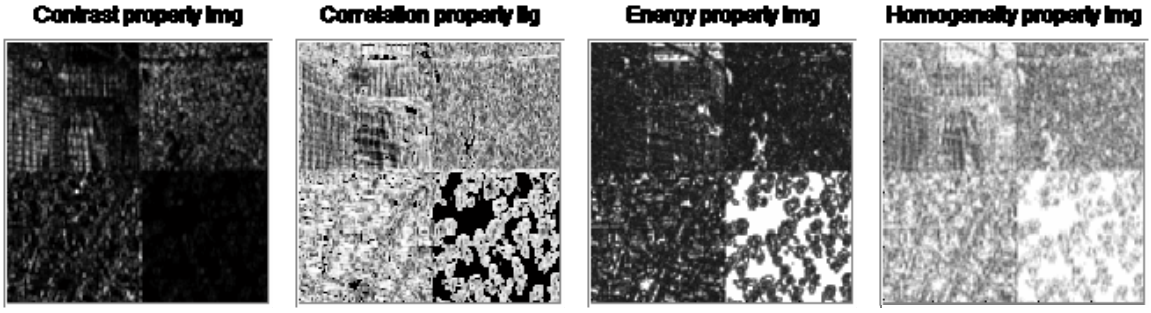


Window size 9, distance 4, angle 0

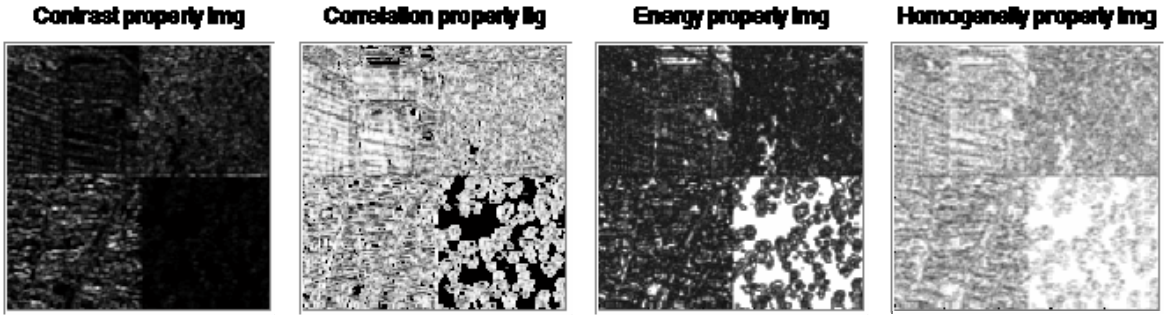




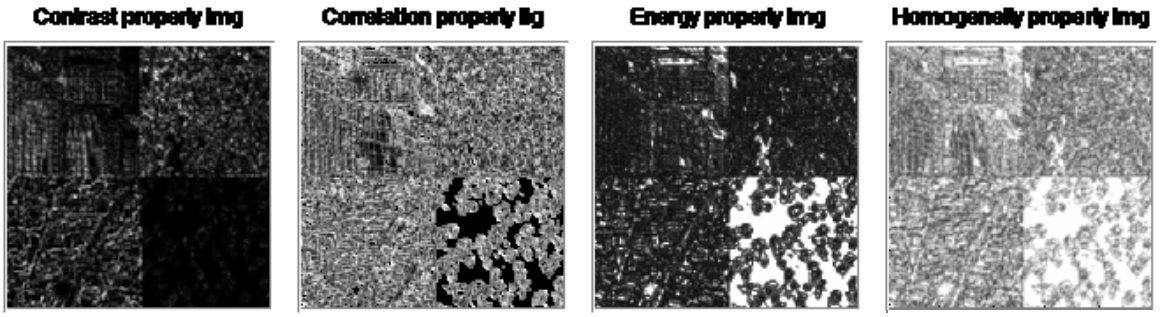
Window size 3, distance 1, angle 0



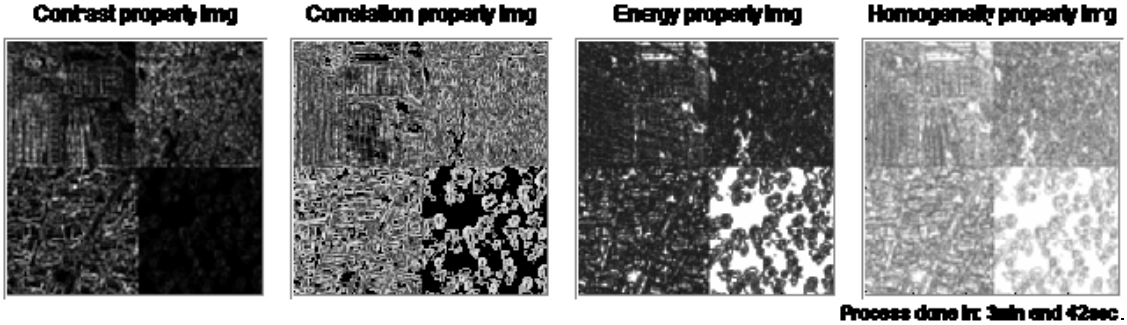
Window size 3, distance 1, angle 90



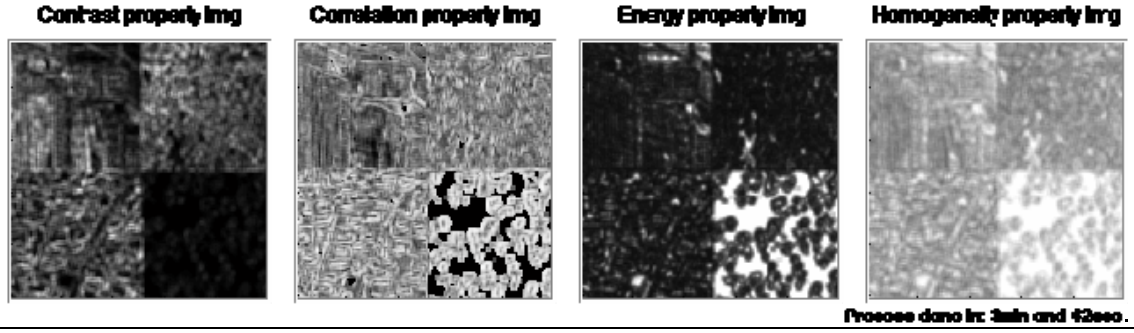
Window size 3, distance 1, angle 135



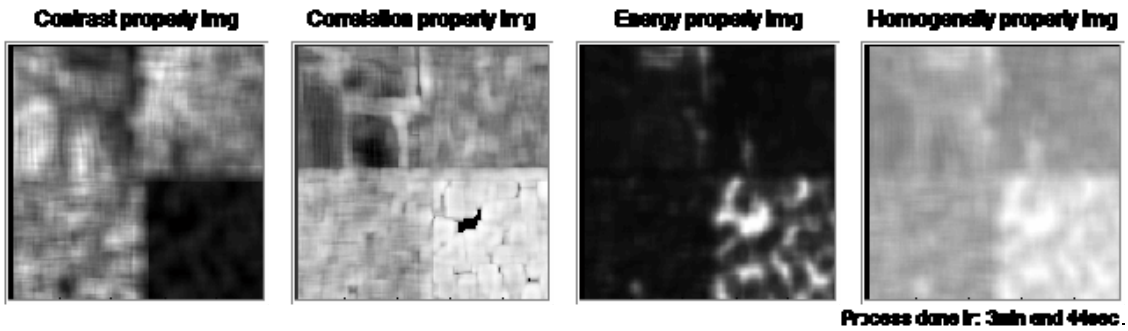
Window size 3, distance 1, angle all



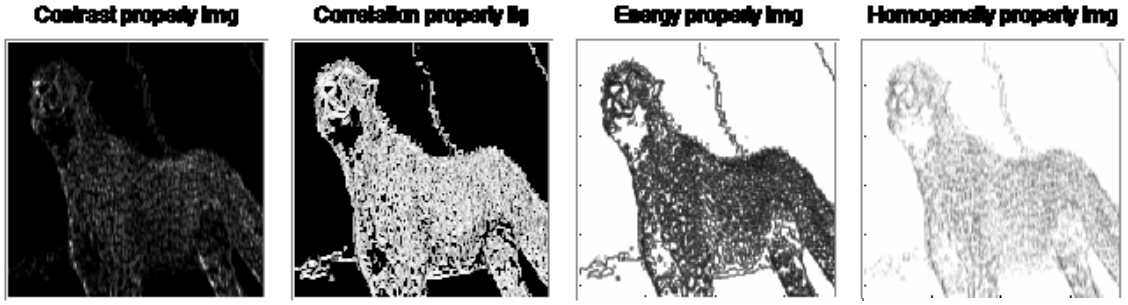
Window size 5, distance 1, angle all



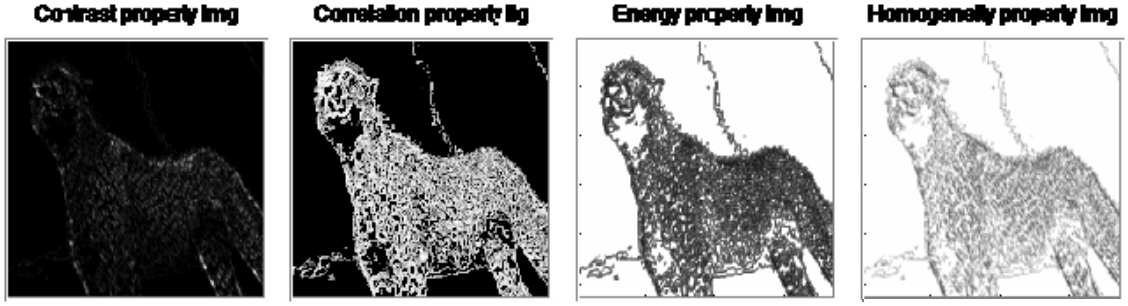
Window size 15, distance 1, angle all



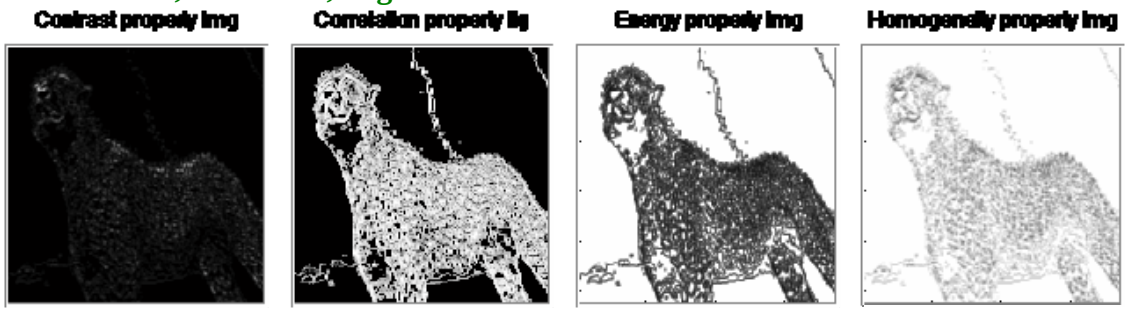
Window size 3, distance 1, angle 0



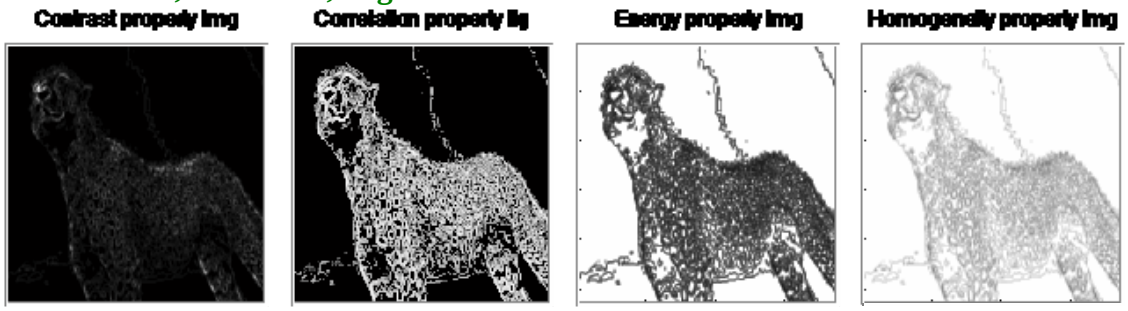
Window size 3, distance 1, angle 45



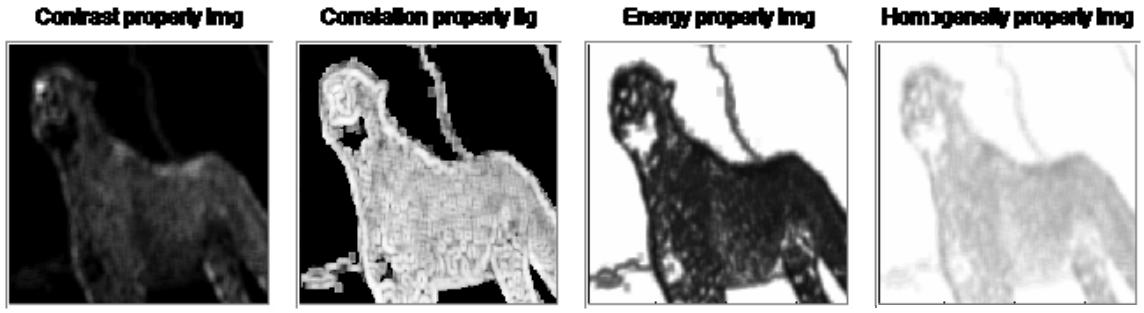
Window size 3, distance 1, angle 90



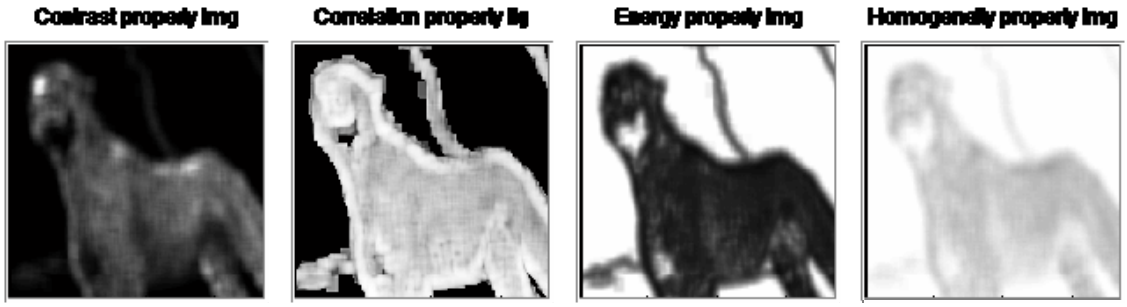
Window size 3, distance 1, angle all



Window size 3, distance 1, angle all



Window size 15, distance 1, angle all



Window size 3, distance 1, angle 0

Contrast property img | Correlation property img | Energy property img | Homogeneity property img

Process done in: 2min and 3sec.

Window size 3, distance 1, angle 90

Contrast property img | Correlation property img | Energy property img | Homogeneity property img

Process done in: 2min and 1sec.

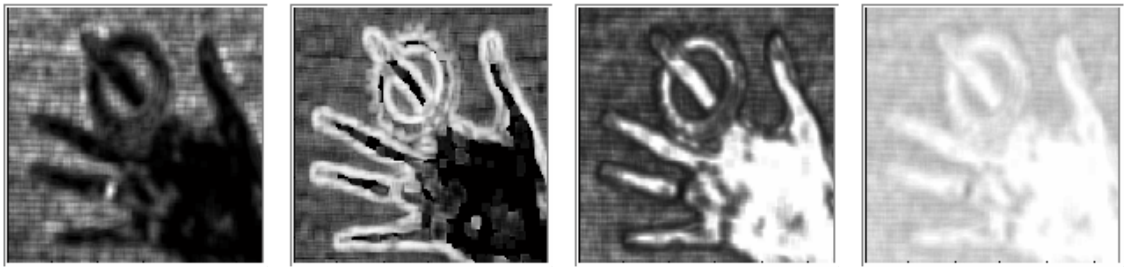
Window size 3, distance 1, angle all

Contrast property img | Correlation property img | Energy property img | Homogeneity property img

Process done in: 3min and 40sec.

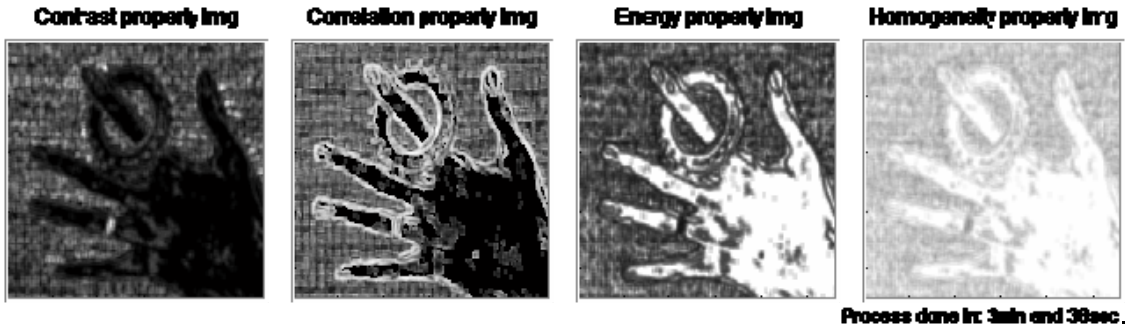
Window size 9, distance 1, angle all

Contrast property img | Correlation property img | Energy property img | Homogeneity property img

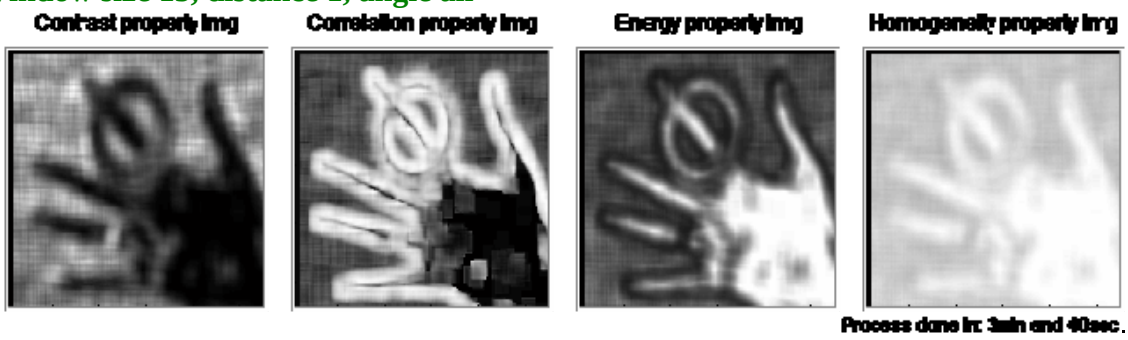


Process done in: 3min and 37sec.

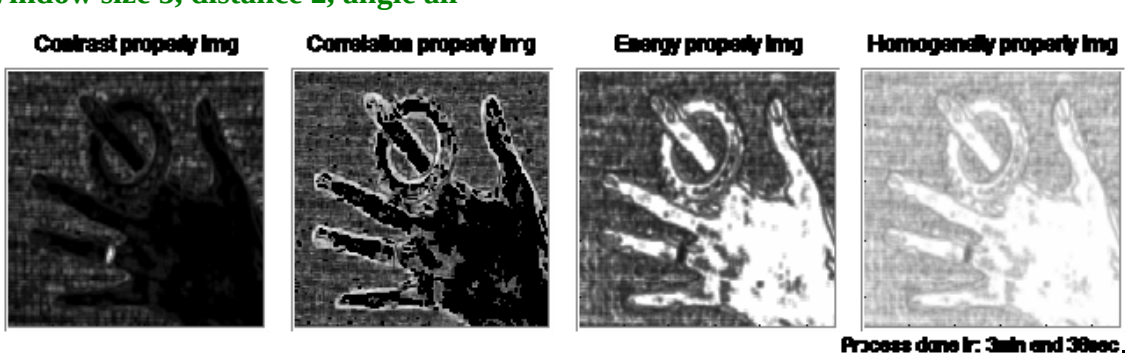
Window size 5, distance 1, angle all



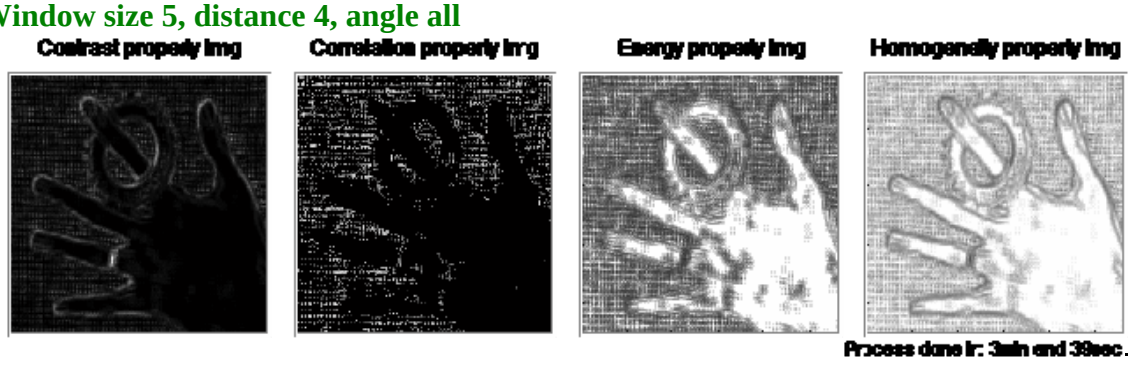
Window size 15, distance 1, angle all

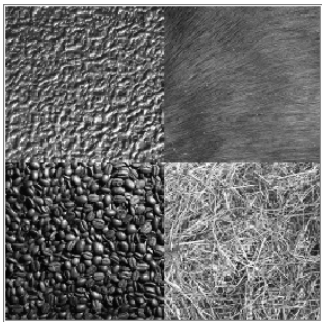


Window size 5, distance 2, angle all

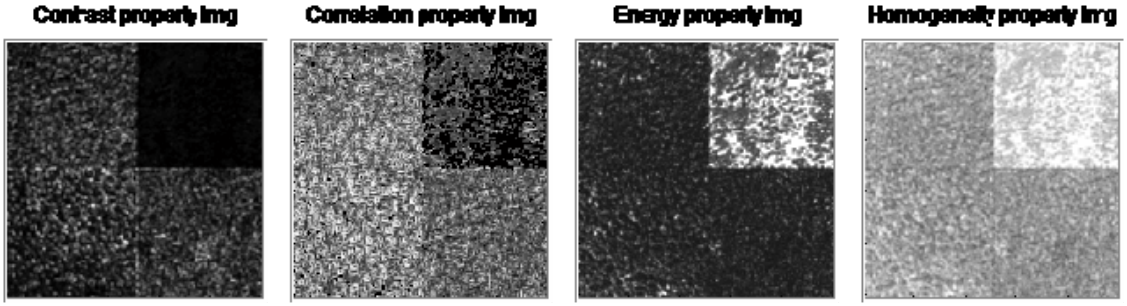


Window size 5, distance 4, angle all

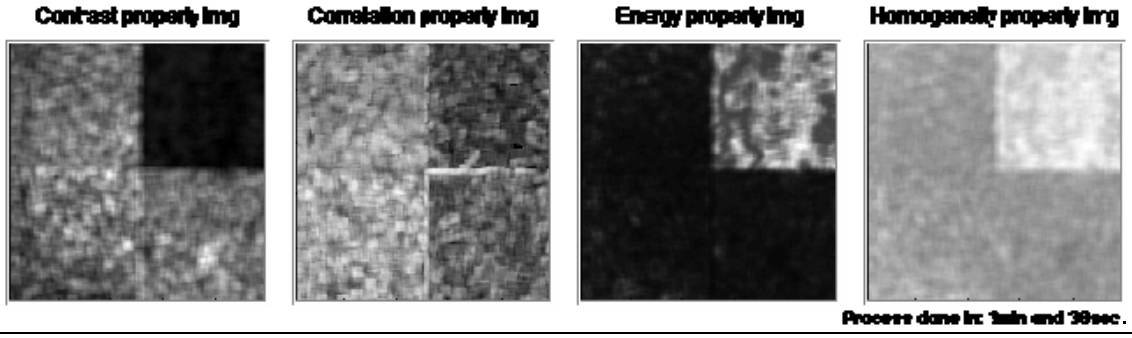




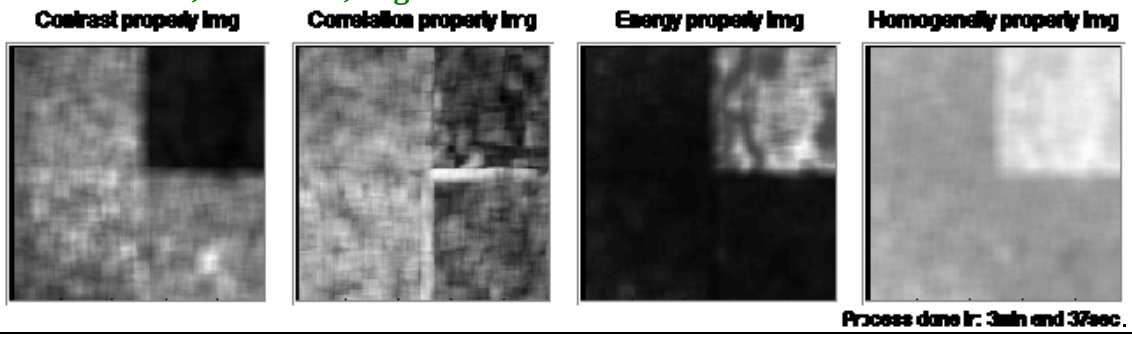
Window size 3, distance 1, angle all



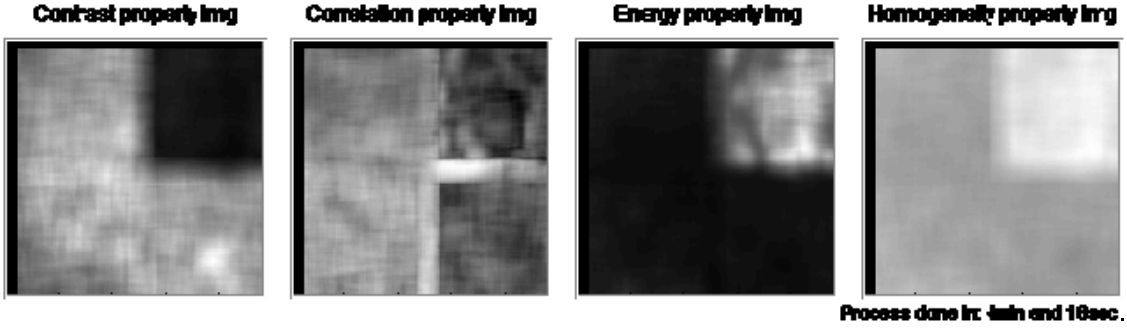
Window size 9, distance 1, angle all



Window size 15, distance 1, angle all

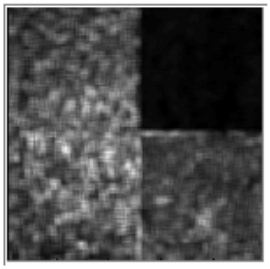


Window size 25, distance 1, angle all

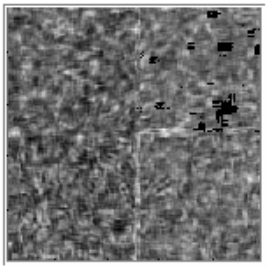


Window size 9, distance 4, angle all

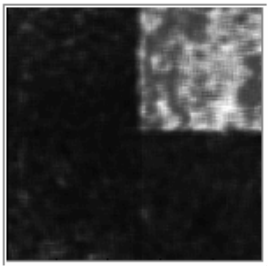
Contrast properly Img



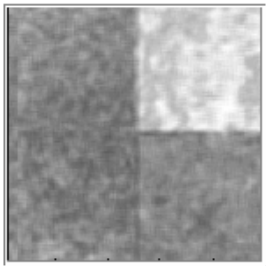
Correlation properly Img



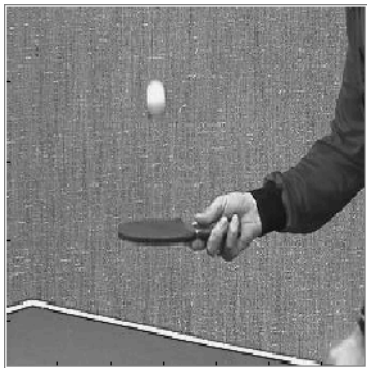
Energy properly Img



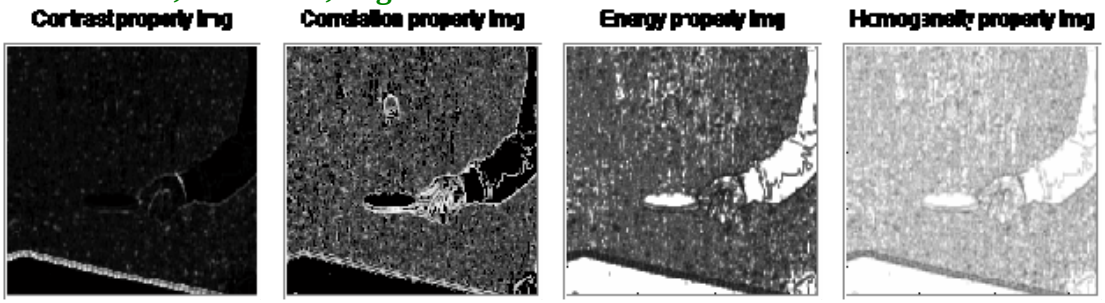
Homogeneity properly Img



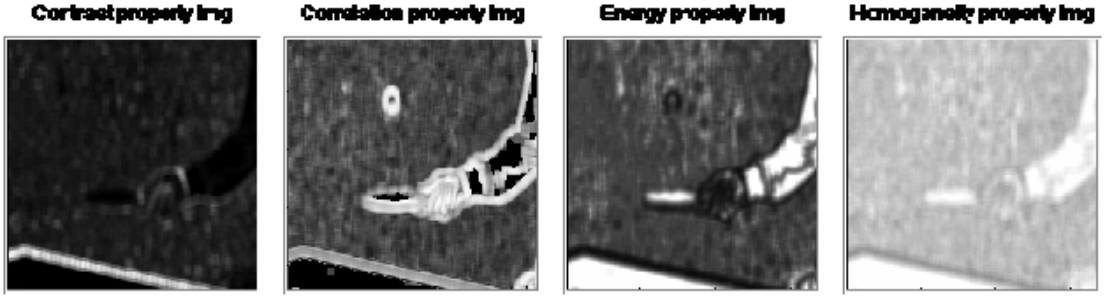
Process done in: 3min and 32sec.



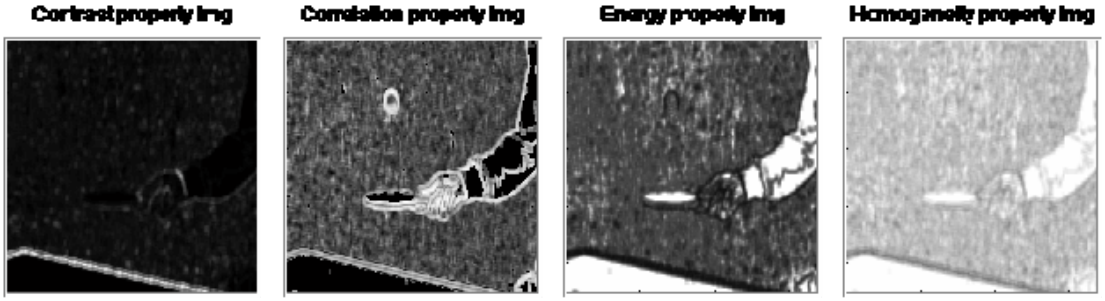
Window size 3, distance 1, angle all



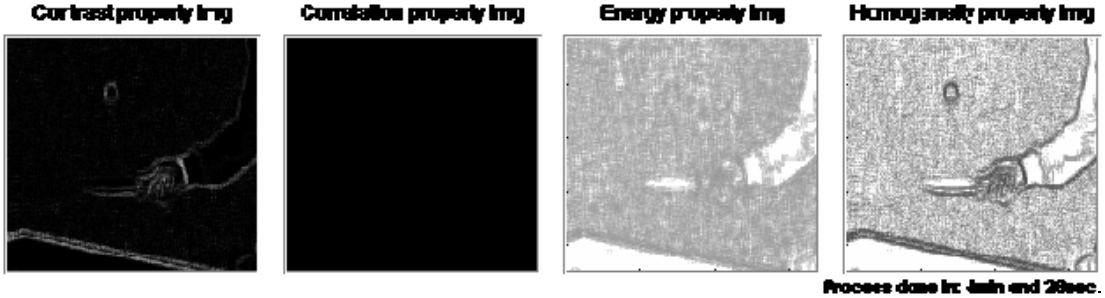
Window size 9, distance 1, angle all



Window size 5, distance 1, angle all

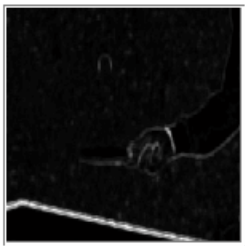


Window size 5, distance 5, angle all

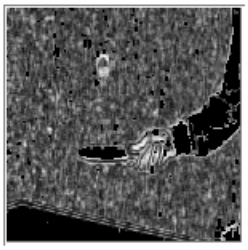


Window size 5, distance 2, angle all

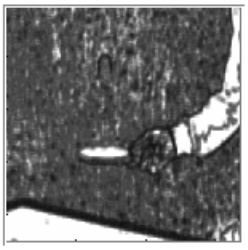
Contrast properly Img



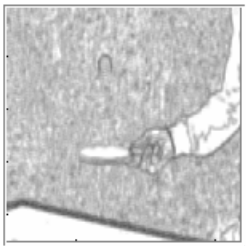
Constallion properly Img



Energy properly Img

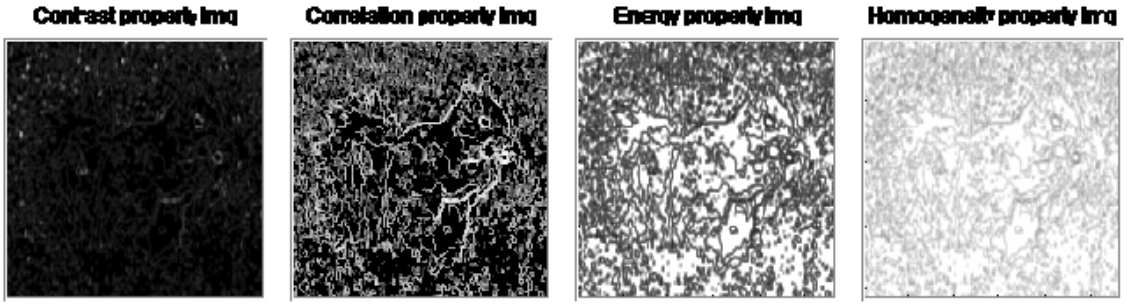


Homogenelly properly Img



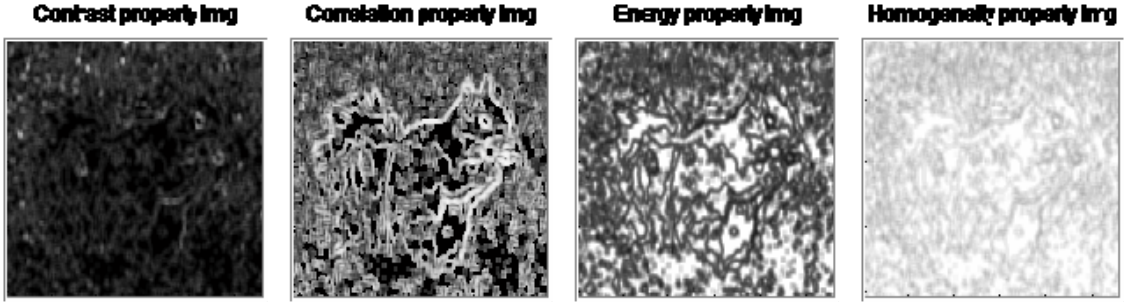
Process done in: 4min and 28sec.

Window size 3, distance 1, angle all



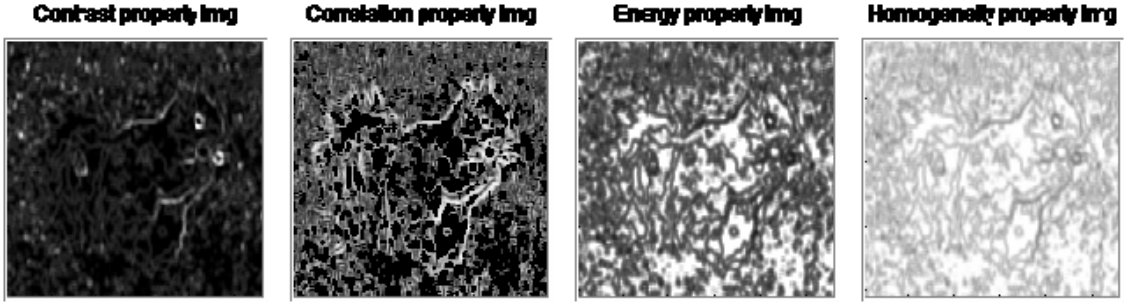
Process done in: 3min and 33sec.

Window size 5, distance 1, angle all



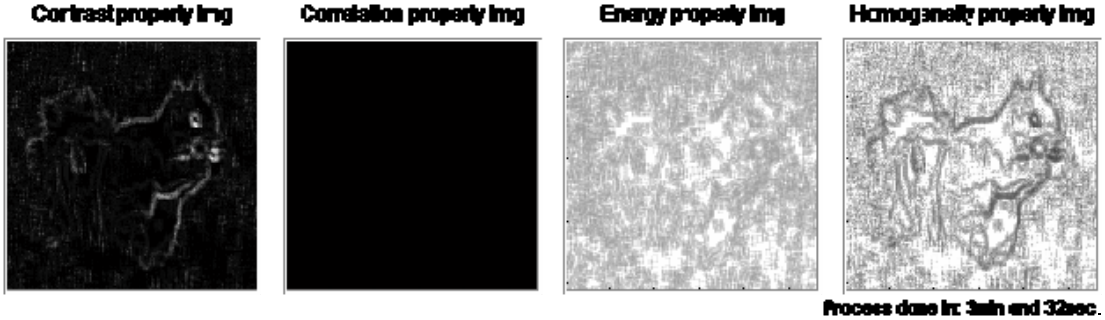
Process done in: 3min and 23sec.

Window size 5, distance 2, angle all



Process done in: 3min and 25sec.

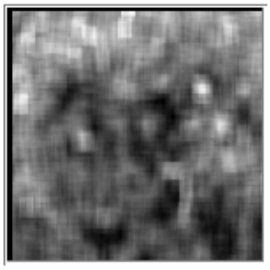
Window size 5, distance 5, angle all



Process done in: 3min and 32sec.

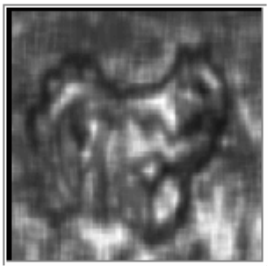
Window size 15, distance 1, angle all

Contrast properly img

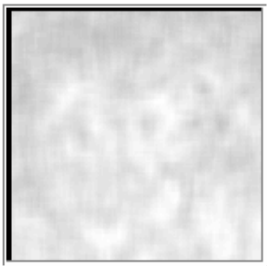


Correlation properly img

Energy properly img



Homogenity properly img



Process done in: 3min and 23sec.

Window size 3, distance 1, angle all

Contrast property img

Correlation property img

Energy property img

Homogenaity property img

Process done in: 3min and 22sec.

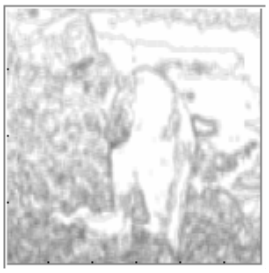
Window size 5, distance 1, angle all

Contrast property img

Correlation property img

Energy property img

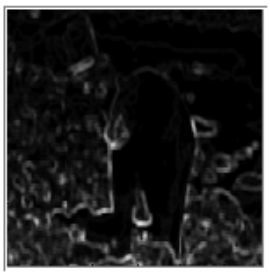
Homogenaity property img



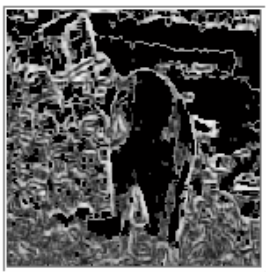
Process done in: 3min and 21sec.

Window size 5, distance 2, angle all

Contrast property img



Correlation property img



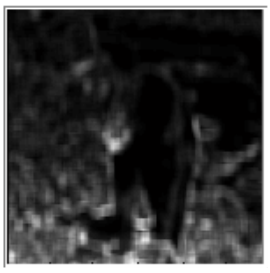
Energy property img

Homogenaity property img

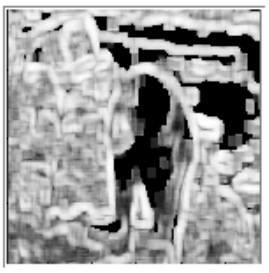
Process done in: 3min and 21sec.

Window size 9, distance 1, angle all

Contrast property img



Correlation property img



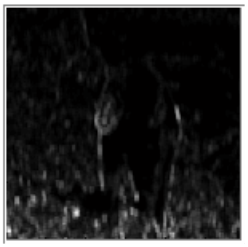
Energy property img

Homogenaity property img

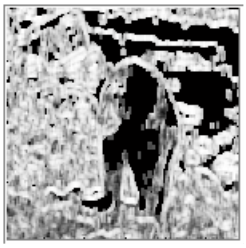
Process done in: 3min and 21sec.

Window size 5, distance 1, angle 0

Contrast property Img

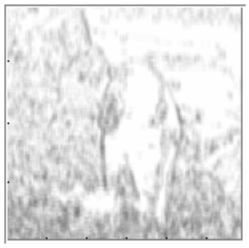


Correlation property Img



Energy property Img

Homogeneity property Img



Process done in: 1min and 52sec.