University of Girona

Scene Segmentation and Interpretation

Coursework 2

Image characterisation using texture

Author:
Di MENG
Darja STOEVA

Supervisor:
Dr. Xavier Lladó
Dr. Arnau Oliver

March 23, 2017

1 Introduction and problem definition.

A texture is a set of texture elements or texels occurring in some regular or repeated pattern. It is an important image feature used in computer vision system. In this lab assignment we investigate the effects of using image texture in segmentation and classification.

One of the most common statistical methods in order to extract texture descriptors "co-occurrence matrices" is used. Texture can be computed globally or locally. By locally, texture is computed at each pixel by using just its neighbourhood. Hence, each pixel will have its own vector of features. Texture computed by locally is used to enhance the region growth algorithm. By globally, all pixels in the image are used to compute a final vector of features. It is used to classify the images to different classes in this lab work.

2 Texture algorithm analysis.

Extraction of texture features has been done using Gray-level Co-occurrence Matrix method(GLCM) which is a statistical method. The GLCM is a tabulation of how often different combinations of pixel brightness values (grey levels) occur in an image. The output of this method is a square matrix of how many times a particular combination of gray level values appear at a given distance and a given orientation. Mathematically, a position in the matrix (i,j) corresponds to the number of pixels of intensity i and j separated by a given distance and angle. They are always square matrices with the same dimensionality as the number of grey-levels. The Matlab provides function "graycomatrix" for computing the co-occurrence matrices.

From the co-occurrence matrix, different texture properties can be computed. Such as "Contrast", which is the Measures the local variations in the gray-level co-occurrence matrix. "Correlation", which is the Measures the joint probability occurrence of the specified pixel pairs. "Energy", which Provides the sum of squared elements in the GLCM. "Homogeneity", which is the Measures the closeness of the distribution of elements in the GLCM to the GLCM diagonal. The Matlab provides function "graycoprops" for computing these statistics from the matrices.

3 Design and implementation for the segmentation problem.

3.1 Implementation of texture features extraction

The image can be segmented based on difference of colors. Except for colors, there are different texture features in one image which are also special representations of different regions. So, extracting texture features of the image and adding them to the channels parallel to the color channels can enhance the result of segmentation.

As the GLCM algorithm specially works for gray-level images, the image is converted to gray level firstly. The textured image always shows high frequency where the region has high spacial variation. Smoothing process is applied to blur the image a little bit for helping provide a better result of segmentation. Here, we choose median filter with different window sizes for comparing the effects.

A format of image is created for saving the texture features. Before computing the co-occurrence matrix, several parameters are set. Then going through all the pixels in the image, a specific window size of neighbors centered at each pixel is considered for computing the co-occurrence matrix. Based on the co-occurrence matrix, four properties are extracted,which are contrast, correlation, homogeneity and energy. They are added to four channels of the image. Four values of the properties are normalized into range 0-1.

An empty image is created with the same size as input image, colors and texture features channels are added to the image. So the dimension of the image is expanded full of information of colors and texture features.

3.2 Implementation of segmentation

The region growth algorithm for segmentation was implemented in last lab assignment, but it has a drawback that it can only be useful for images which have one or three channels. Because only gray and rgb images were considered last time. For fitting the optimized way of segmentation with

texture features, the code is modified which can be useful for images with multiple channels.

4 Experimental section and results analysis for the segmentation problem

4.1 Effects of parameters on descriptors

Before incorporating the texture descriptors into the region growth algorithm, it is necessary to observe the effects of different parameters on the output of GLCM descriptors and select the optimal parameters to be added to the segmentation part.

4.2 Effects of window size of neighbors

The effects of using different window sizes are compared in this section. Different window sizes are applied to provided four images. All the other parameters are kept constant to better observe the effect on the chosen parameter. The effects on different window sizes are shown below, three of them are shown here.

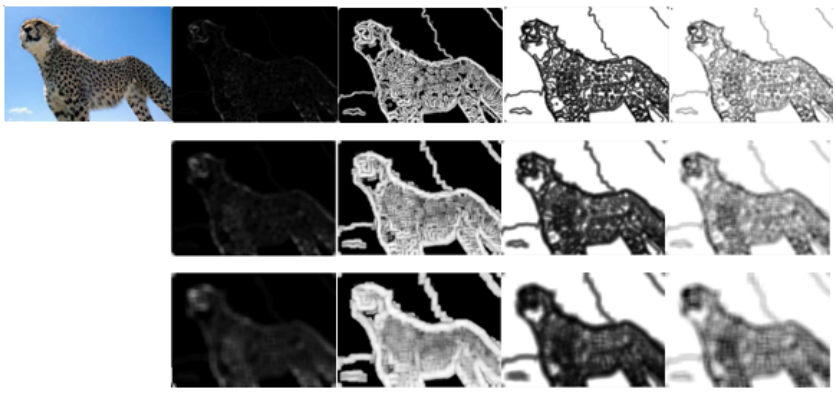


Figure 1: Effect of window size on feli image. NumLevels=8, Distance=2, offset=[0 D;-D D;-D 0;-D -D]. From left to right-Original image, Contrast, Correlation, Energy, Homogeneity. From top to bottom, window size=7,13,19

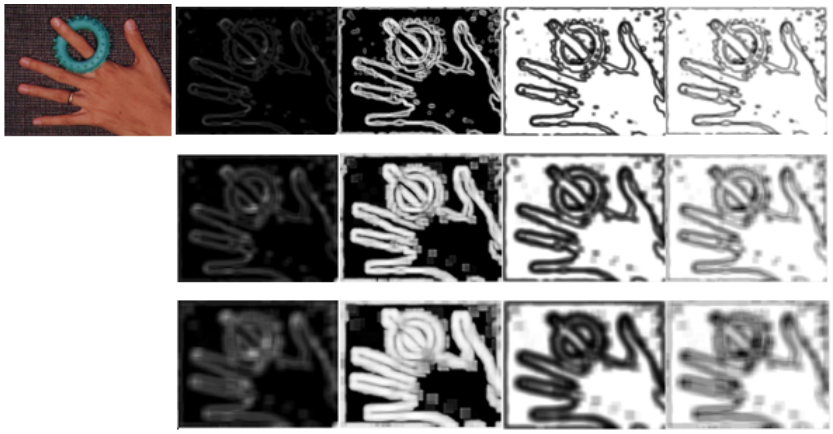


Figure 2: Effect of window size on hand image. NumLevels=8, Distance=2, offset=[0 D;-D D;-D 0;-D -D]. From left to right-Original image, Contrast, Correlation, Energy, Homogeneity. From top to bottom, window size=7,13,19

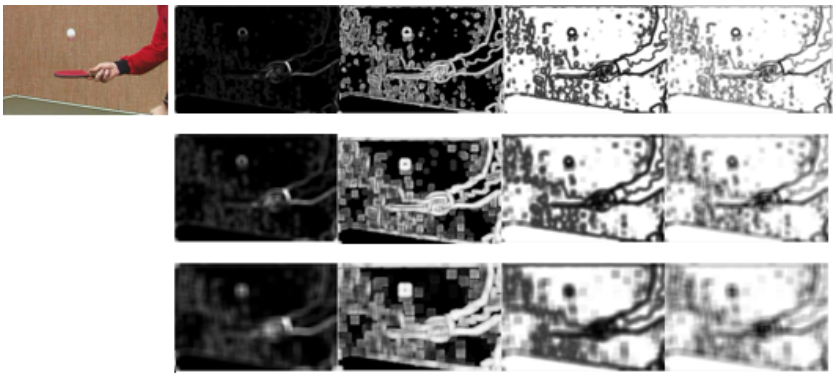


Figure 3: Effect of window size on pingpong image. NumLevels=8, Distance=2, offset=[0 D;-D D;-D 0;-D -D]. From left to right-Original image, Contrast, Correlation, Energy, Homogeneity. From top to bottom, window size=7,13,19

As we can see, the larger the window size, the more blurred the descriptors. A general conclusion is that a certain window size should be chosen which is not too small. For instance, if the window size is smaller than one spot of the tiger, the feature vector we get would not describe the texture well.

4.3 Effects of offset - Distance

In this section different distances of the offset are tested on four provided images. All the other parameters are kept constant to better observe the effect on the chosen parameter. Only two of them are displayed below in order to fill up the pages with images.

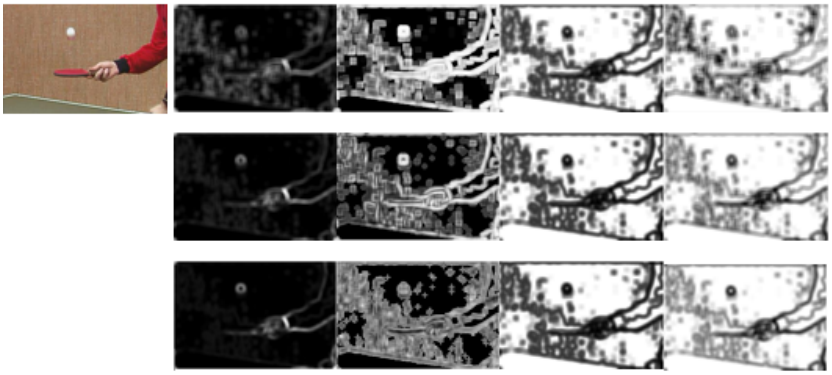


Figure 4: Effect of window size on pingpong image. NumLevels=8, Window size=13, offset=[0 D;-D D;-D 0;-D -D]. From left to right-Original image, Contrast, Correlation, Energy, Homogeneity. From top to bottom, distance=1,3,5

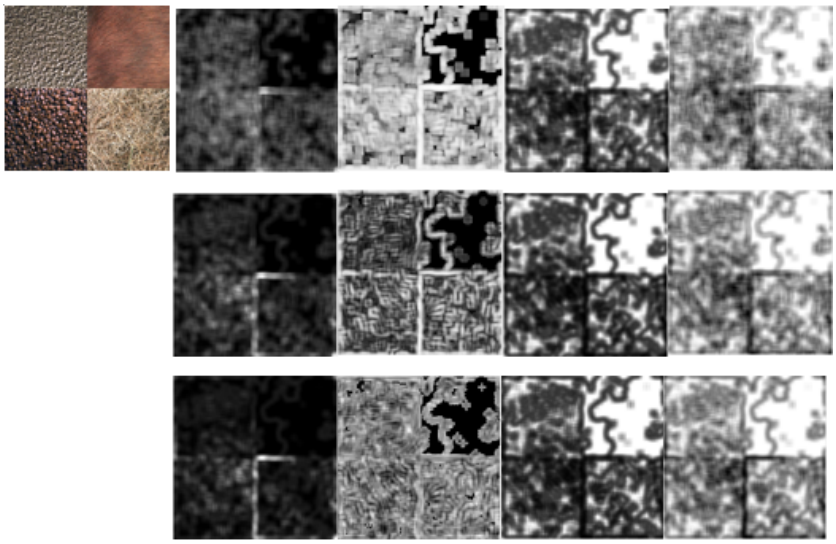


Figure 5: Effect of window size on mosaic image. NumLevels=8, Window size=13, offset=[0 D;-D D;-D 0;-D -D]. From left to right-Original image, Contrast, Correlation, Energy, Homogeneity. From top to bottom, distance=1,3,5

As we can see, the effects on property "energy" and "homogeneity" are almost zero. There is no noticeable effects on this two output. The regions with same textures are seen to be more homogeneous when increasing the distance, which can be used to better integrate texture features with region growth algorithm.

4.4 Effects of offset - Orientation

In this section different orientations of the offset are tested on four provided images. All the other parameters are kept constant to better observe the effect on the chosen parameter. Only two of them are displayed below in order to fill up the pages with images.

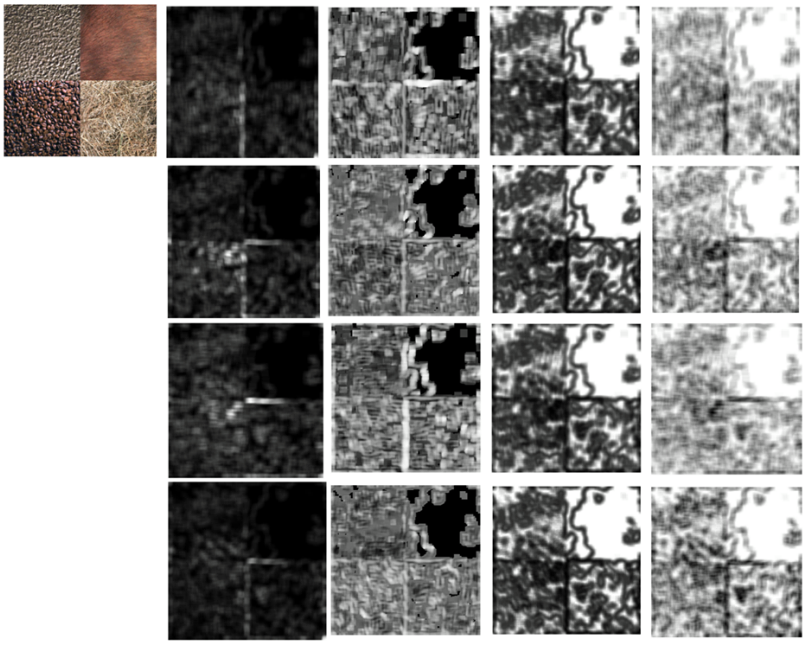


Figure 6: Effect of window size on mosaic image. NumLevels=8, Window size=13, Distance=3. From left to right-Original image, Contrast, Correlation, Energy, Homogeneity. From top to bottom, offset=[0 D],[-D D],[-D 0][-D -D]

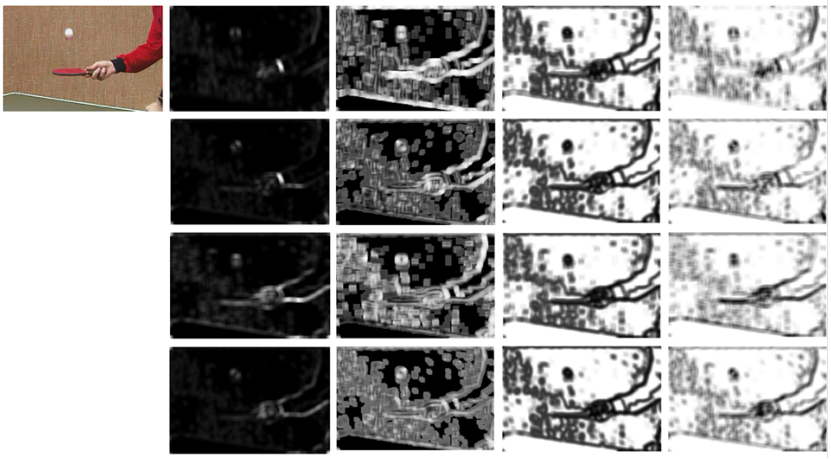


Figure 7: Effect of window size on pingpong image. NumLevels=8, Window size=13, Distance=3. From left to right-Original image, Contrast, Correlation, Energy, Homogeneity. From top to bottom, offset=[0 D],[-D D],[-D 0][-D -D]

Pingpong and mosaic images are observed to give better distinction in 90 degrees whereas the other two images give better results in 45 degrees. It indicates that the effects on orientations are highly dependent on the types of textures.

4.5 Effects of NumLevels

In this section different gray levels of the offset are tested on four provided images. All the other parameters are kept constant to better observe the effect on the chosen parameter. Only two of them are displayed below in order to fill up the pages with images.

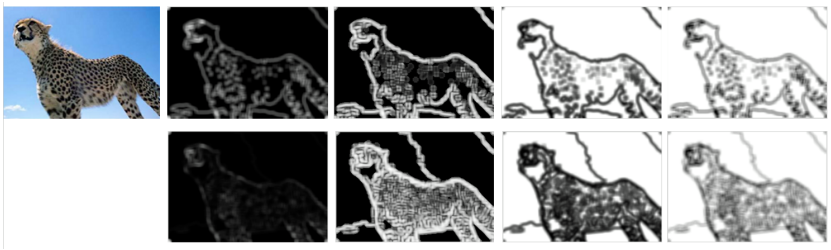


Figure 8: Effect of window size on feli image. Distance=3, Window size=13, offset=[0 D;-D D;-D 0;-D -D]. From left to right-Original image, Contrast, Correlation, Energy, Homogeneity. From top to bottom, gray levels=4,8

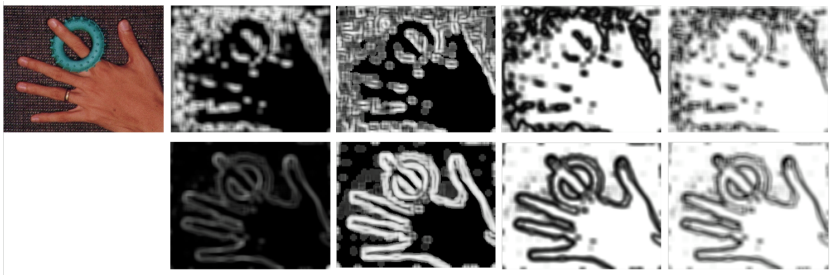


Figure 9: Effect of window size on hand image. Distance=3, Window size=13, offset=[0 D;-D D;-D 0;-D -D]. From left to right-Original image, Contrast, Correlation, Energy, Homogeneity. From top to bottom, gray levels=4,8

The gray level of 8 shows more distinctive than 4. And all the results of 8 gray levels are observed better.

5 Segmentation using texture

After observing the effects on different parameters, different descriptors of textures are added to the region growth algorithm. The optimal parameters for different image could be very different, we try to keep some of the parameters same for implementing a more adaptive algorithm.

For feli image, all the descriptors are added to the segmentation algorithm. The parameters chose are: offset=[0 D; -D D; -D 0; -D -D],

Distance=2, numLevels=8, window size=19, median filter size=[7,7]. The threshold chose for segmentation is 1.4.

For pingpong image, all the descriptors are added to the segmentation algorithm. The parameters chose are: offset=[-D 0], Distance=2, numLevels=8, window size=19, median filter size=[13,13]. The threshold chose for segmentation is 0.97.

For hand image, all the descriptors are added to the segmentation algorithm. The parameters chose are: offset=[0 D; -D D; -D 0; -D -D], Distance=2, numLevels=8, window size=19, median filter size=[13,13]. The threshold chose for segmentation is 0.97.

For mosaic image, all the descriptors are added to the segmentation algorithm. The parameters chose are: offset=[0 D; -D D; -D 0; -D -D], Distance=2, numLevels=8, window size=7, median filter size=[15,15]. The threshold chose for segmentation is 0.8.

Figure 10: Segmentation with texture on feli

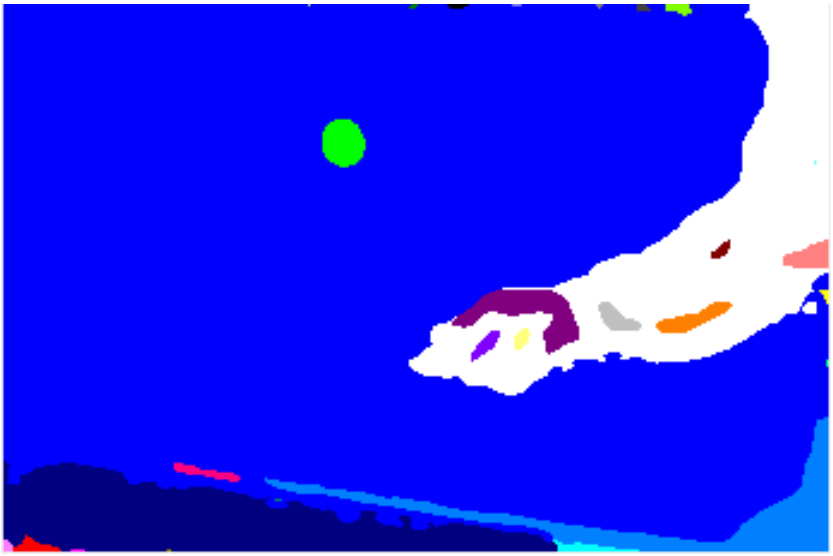


Figure 11: Segmentation with texture on pingpong

Figure 12: Segmentation with texture on hand

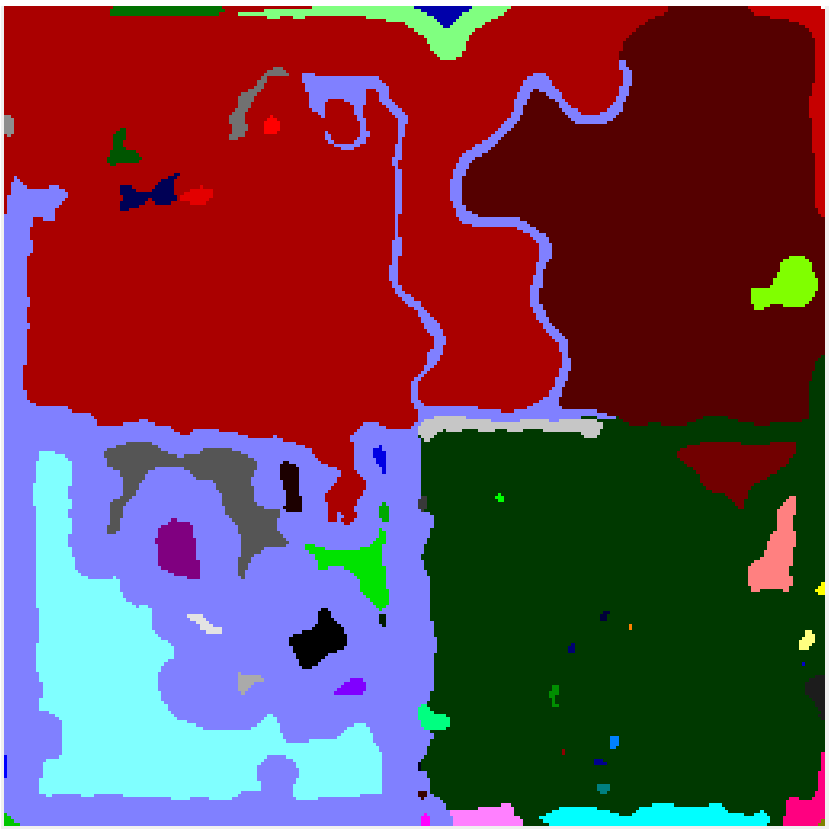


Figure 13: Segmentation with texture on mosaic

6 Design and implementation of the proposed solution for the classification problem

In order to analyse and understand the texture computed globally, classification using feature extraction was implemented. To classify the dataset Textures using texture features, a feature vector has to be computed. For this part of the lab 20 different datasets of images were used, each containing six images, two used for training and four used for testing. The script `scr_classifyPR` uses the library PRTools to classify the textures of the dataset using a k-nearest neighbor classifier. However, before being able to classify the images, the feature vector for both, training and test images is computed.

The `computeFeatureVector` function was modified to increase the accuracy of the classifier. The function takes an image as an input and returns the feature vector using the build-in MATLAB functions: `graycomatrix` and `graycoprops`. The `graycomatrix` was used t to generates a grey-level co-occurence matrix which was then used for calculating the statistical properties such as: contrast, correlation energy and homogeneity.

The accuracy of the classifier strongly depends on the parameters passed when calculating the co-matrix. In order to improve the accuracy the `'Offsets'` was used, to calculate the frequency of a pixel occuring horrizontally in multiple directions.It was noticed that with the increase of the number of directions used the accuracy also increased. Several directions were tried and the value giving the highest classifier accuracy was chosen.

Once the feauture vector was computed, for all images, a classifier was build using the PRTools and the feauture vectors from the train images. The feature vectors of the test images were tested and the final confusion matrix was computed with an error estimation and percentage of accuracy.

7 Experimental section and results analysis for the classification problem

Figure 14 shows the final confusion matrix, calculated using feature vector with 14 different offsets, which gives 91.25% of accuracy. The accuracy was achived by trying different values for the offsets. It was noticed that there is a threshold which spans the highest accuracy and when a higher or lower value than the threshold was used the accuracy starts to decrease.

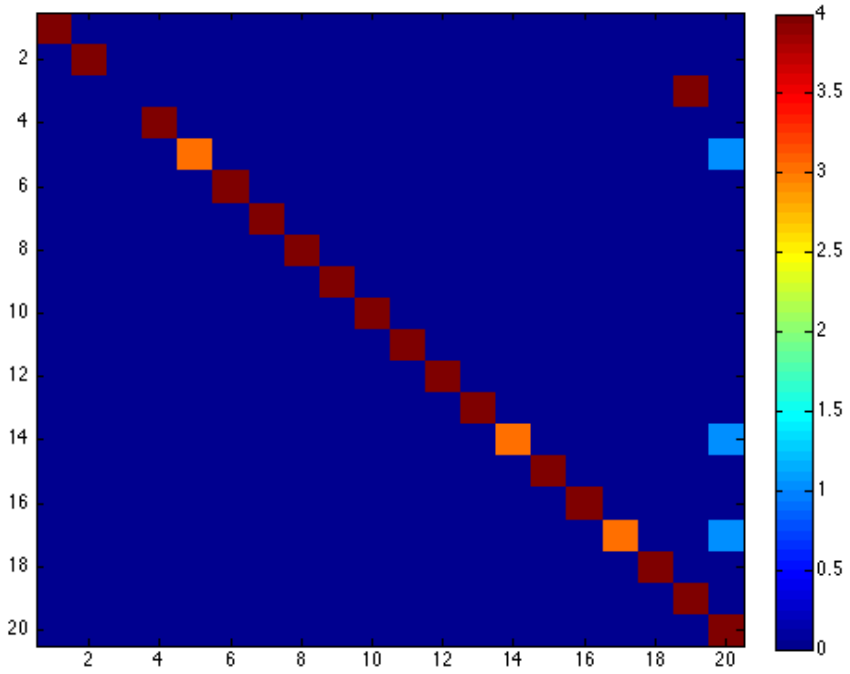


Figure 14: Confusion matrix and accuracy of the classifier

Figure 15 shows the change of accuracy depending on the distance used to compute the properties of the co-matrix. As the graph shows, if low distance values are taken into account the accuracy is low, or if high distance values are used since the values compared are too far therefore very likely to be different.

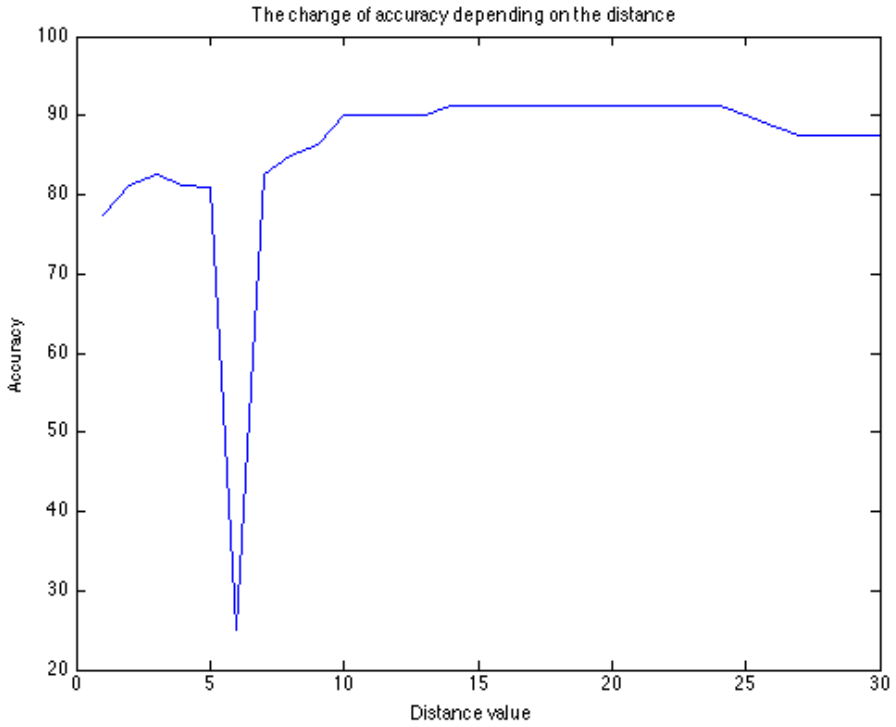
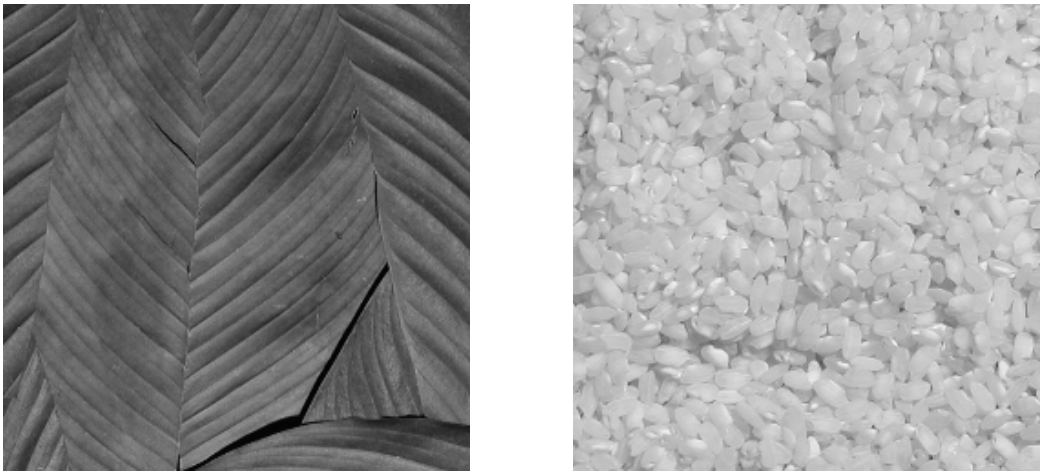


Figure 15: Graph of the accuracy changing with the change of distance value

From the matrix it can be seen that 7 test images were wrongly classified. The test images from dataset 3 have the highest inaccuracy since all of them were calssified as images belonging to the dataset 19. This could be due to very similar texture features, like contrast, homogeneity and energy. For example, Figure 16 shows an image from dataset 3 (a) and an image from dataset 19. Although the images are very different, the variation in intesity values and uniformity of the images are very similar, so when computing the properties, it is highly likely that they will be similar. This is the case because the contrast is computed by the local variation in the image and energy indicates the uniformity of an image. The same can be said for the other three wrongly classified images.



(a) (b)

Figure 16: Image from dataset 3 (a) and dataset 19 (b)

One way of improving this classifier could be including the color, since the current algorithm transforms the image into grayscale image. When comparing the texture feature properties using a grayscale image impacts the propertiy values since grayscale images give less information.

8 Conclusion

Texture is an useful feature which can be computed locally and globally.

For segmentation problem, locally computing texture features method is used and effects of different parameters are analysed. Texture features can be used for enhancing segmentation results but they are highly dependent on the type of textures and sensitive with the parameters.

For the classification problem it can be concluded that the texture features can be used for classifying images. Although using more distance values is more costly, it increases the accuracy of the classifier. Another way to improve the performance of the algorithm would be to use the colored image for retriving more information.