TP 1 : Spectral characterization

I. Frequency filtering

1. Under matlab, produce the image of the Fourier spectrum in conventional representation of the Lena image. The functions abs, fft2 and log should be used.

2. Invert the spectrum using the ifft2 function. What Root Mean Square Error do you obtain between the original image and the image obtained after the inverse transform?

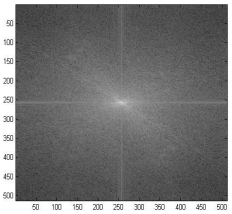
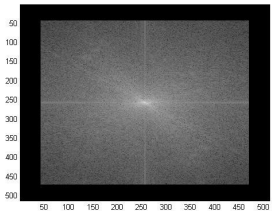
3. Apply the following algorithm:

```
n = 1;
While n < largeur_image/2-1 do
     Set to zero the first n rows, the last n rows, the first
n columns and the last n columns of the spectrum in its
conventional representation

     Compute the inverse transform to produce the filtered
image

     Compute the Mean Square Error between the starting image
and the new filtered image
     n = n + 1;
End while.
```

Produce on the same graph the curve describing the evolution of the Mean Square Error (in percentage compared to the maximum calculated error) and the curve describing the percentage of coefficients set to 0 at each iteration.

			
Starting image	Spectrum in conventional representation	Cancellation of high frequency coefficients	Inversion of the filtered spectrum

II Characterization

A collection of 50 images can be downloaded at the following link : bibimage.zip or bibimage.tar

1. For each image of the collection below, produce a feature vector characterizing the texture. To do this, calculate the half-spectrum in conventional representation of the image converted to gray levels. Calculate the sum of the moduli of the frequency coefficients taken from 18 zones defined by:

	Columns 1 to 50	Columns 51 to 75	Columns 76 to 100	Columns 101 to 125	Columns 126 to 150	Columns 151 to 200
Lines 1 to 50						
Lines 1 to 50						
Lines 1 to 50						

Note: the 101st row is deliberately ignored.

Create a matrix with 50 rows (one per image) and 18 columns (one per feature). Arrange the 18 values obtained for each image in this matrix.

You can use the code below as a guide:

```
nbimg=50;
path='C:\...\';
signature=zeros(nbimg,18);
for i=1:nbimg
   unit=num2str(mod(i,10));
   hundred=floor(i/100);
   ten=num2str(floor((i-hundred*100)/10));
   file=strcat(path,ten,unit)
   clear img;
   img=imread(file,'jpg');
   ...
end
```

2. Choose an image in the collection. Compute the Manhattan distance between the vector associated with the chosen image and those of all other images. Identify the most similar image in the sense of this distance.
Perform several tests. For each one, give the value of the minimum distance and comment on your result.