PRACTICA 5. SERVICIOS TELEMATICOS MULTIMEDIA

```
% Guillermo Aldrey Pastor

% Fecha de Entrega: 20/11/2020

% En este archivo "practica5.m" se procede a la resolucion de la practica.
% En este caso voy a probar un nuevo metodo de presentacion en el que se
% comentara la explicacion de cada apartado en el explica en un live script
% que luego convertire a PDF.


clc;clear all;close all;
```

Analisis de Imagen

```
    % Leer y mostrar imagen
    original= imread('imagen.bmp');
    imshow(original)
```

```
Warning: Image is too big to fit on screen; displaying at 67%
```

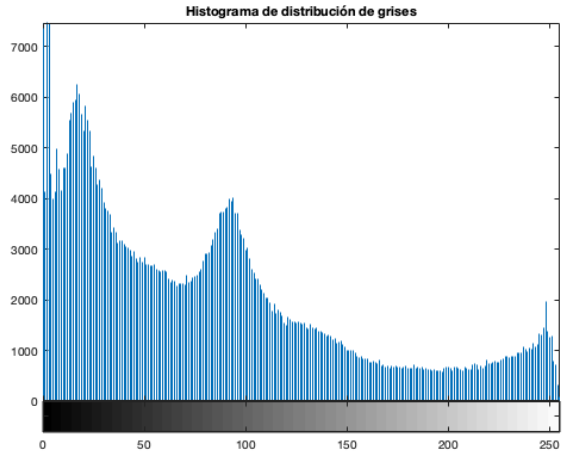
```
    pause
```

```
% Convertir de color a escala de grises
I=rgb2gray(original);
figure, imshow(I)
```

```
Warning: Image is too big to fit on screen; displaying at 67%
```

```
title('Image convertida a escalas de grises')
```


```
pause

% Mostrar histograma de distribución de grises
figure, imhist(I)
title('Histograma de distribución de grises')
```




```
pause

% Ajuste escala de grises a fondo de escala
J = imadjust(I);
figure, imshow(J)
```

```
Warning: Image is too big to fit on screen; displaying at 67%
```

```
title('Se ajusta la escala de grises a fondo de escala')
```

```
pause

% Obtener negativo de la imagen
gamma=1;
neg=imadjust(I, [0 1], [1 0], gamma);
figure, imshow(neg)
```

```
Warning: Image is too big to fit on screen; displaying at 67%
```

```
title('Negativo de la imagen')
```


```
pause

% NORMALIZACIONES

% Ajustar gamma

% Ajustar valores normalizados al intervalo 0.25 0.75 negativo de la
% imagen gamma = 1
gamma=1;
Iad=imadjust(I, [0.25 0.75], [0 1], gamma);
figure, imshow(Iad)
```

```
Warning: Image is too big to fit on screen; displaying at 67%
```

```
    title('Intervalo [0.25 0.75] negativo de la imagen, gamma = 1')
```


```
    pause

    % Ajustar valores normalizados al intervalo 0.25 0.75 gamma=2
    gamma=2;
    Iad=imadjust(I, [0.25 0.75], [0 1], gamma);
    figure, imshow(Iad)
```

```
Warning: Image is too big to fit on screen; displaying at 67%
```

```
    title('Intervalo [0.25 0.75] negativo de la imagen, gamma = 2')
```


```
    pause

    % Ajustar valores normalizados al intervalo 0.25 0.75 gamma=0.5
    gamma=0.5;
    Iad=imadjust(I, [0.25 0.75], [0 1], gamma);
    figure, imshow(Iad)
```

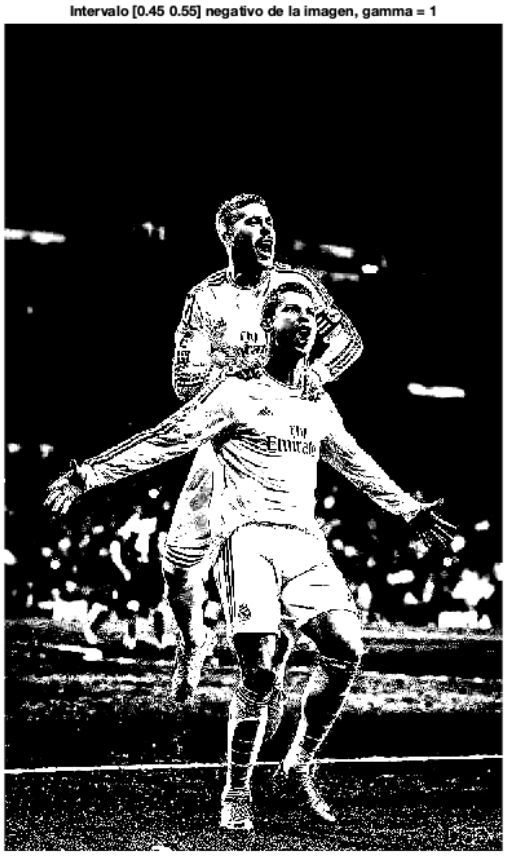
```
Warning: Image is too big to fit on screen; displaying at 67%
```

```
    title('Intervalo [0.25 0.75] negativo de la imagen, gamma = 0.5')
```


```
    pause

    % Ajustar Intervalo

    % Ajustar valores normalizados al intervalo 0.45 0.55 negativo de la
    % imagen gamma = 1
    gamma=1;
    Iad=imadjust(I, [0.45 0.55], [0 1], gamma);
    figure, imshow(Iad)
```

```
Warning: Image is too big to fit on screen; displaying at 67%
```

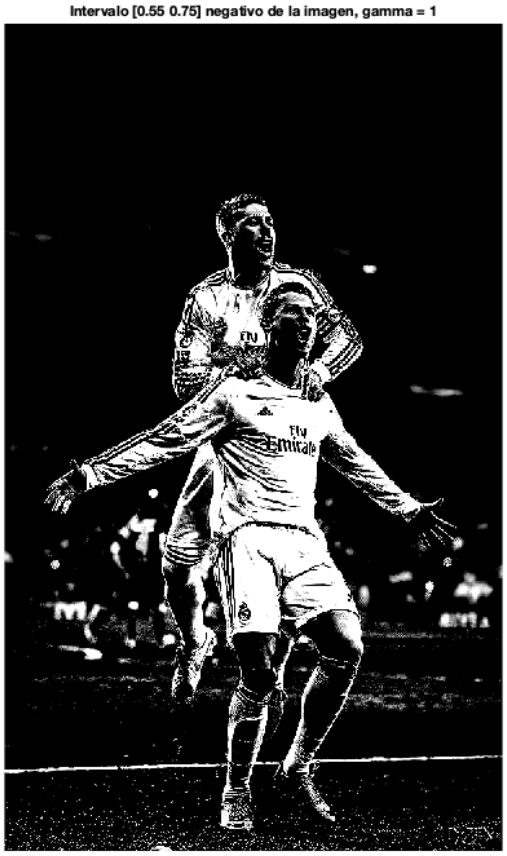
```
    title('Intervalo [0.45 0.55] negativo de la imagen, gamma = 1')
```




```
    pause

    % Ajustar valores normalizados al intervalo 0.55 0.75 negativo de la
    % imagen gamma = 1
    gamma=1;
    Iad=imadjust(I, [0.55 0.75], [0 1], gamma);
    figure, imshow(Iad)
```

```
Warning: Image is too big to fit on screen; displaying at 67%
```

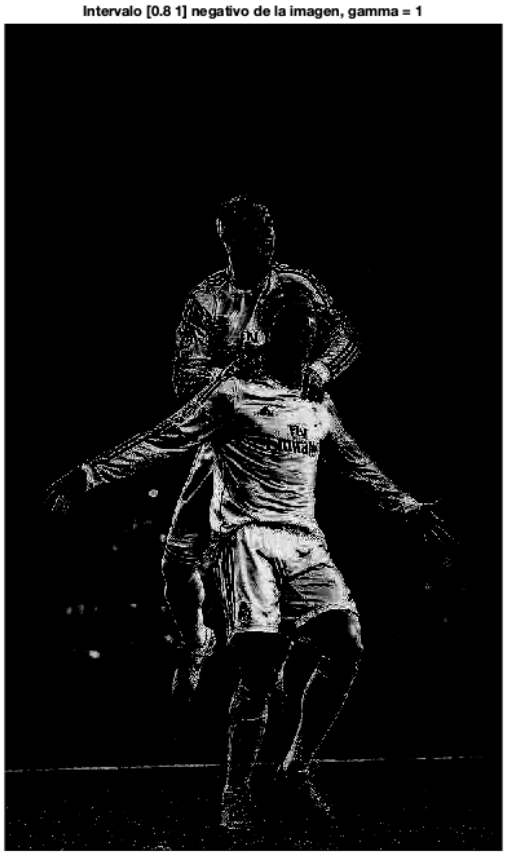
```
    title('Intervalo [0.55 0.75] negativo de la imagen, gamma = 1')
```




```
    pause

    % Ajustar valores normalizados al intervalo 0.8 1 negativo de la
    % imagen gamma = 1
    gamma=1;
    Iad=imadjust(I, [0.8 1], [0 1], gamma);
    figure, imshow(Iad)
```

```
Warning: Image is too big to fit on screen; displaying at 67%
```

```
    title('Intervalo [0.8 1] negativo de la imagen, gamma = 1')
```




```
    pause

    % Ajustar valores normalizados al intervalo 0.1 0.8 negativo de la
    % imagen gamma = 1
    gamma=1;
    Iad=imadjust(I, [0.1 0.8], [0 1], gamma);
    figure, imshow(Iad)
```

```
Warning: Image is too big to fit on screen; displaying at 67%
```

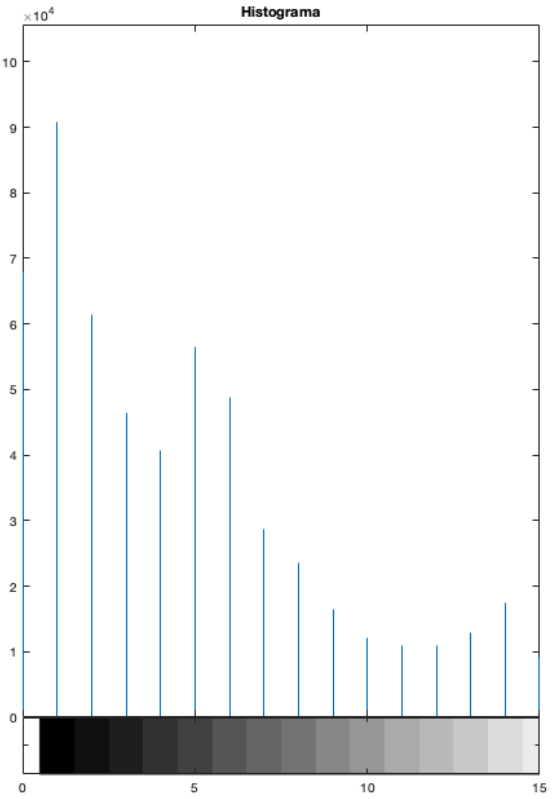
```
    title('Intervalo [0.1 0.8] negativo de la imagen, gamma = 1')
```


```
    pause

    % Disminucion niveles de gris a 16 y cuenta de valores
    [X, map] = gray2ind(I, 16);
    figure, imshow(X, map);
```

```
Warning: Image is too big to fit on screen; displaying at 67%
```

```
    title('Disminución niveles de gris a 16')
```


```
imhist(X,map)
title('Histograma')
```




```
    [cuenta, y]=imhist(X, map)
```

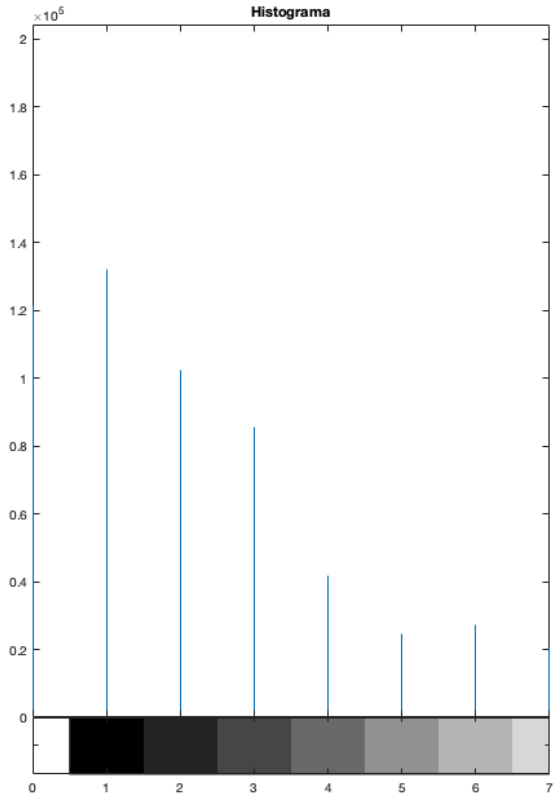
```
cuenta = 16×1
       67854
       90818
       61333
       46526
       40731
       56418
       48745
       28758
       23512
       16547
      ⋮
y = 1×16
     0     1     2     3     4     5     6     7     8     9    10    11    12 ···
```

```
    pause

    % Disminucion niveles de gris a 8 y cuenta de valores
    [X, map] = gray2ind(I, 8);
    figure, imshow(X, map);
```

```
Warning: Image is too big to fit on screen; displaying at 67%
```

```
    title('Disminución niveles de gris a 8')
```


```
    imhist(X,map)
    title('Histograma')
```




```
    [cuenta, y]=imhist(X, map)
```

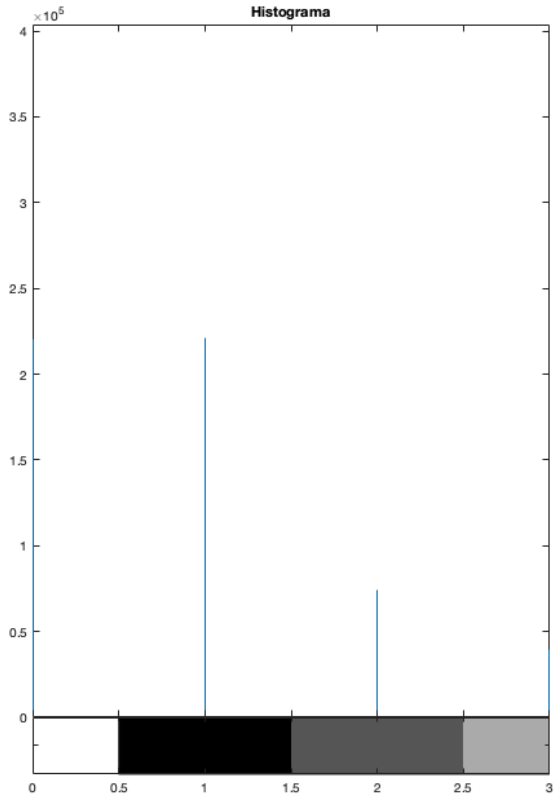
```
cuenta = 8×1
      121495
      132020
      102222
       85446
       41609
       24624
       27080
       20384
y = 1×8
     0     1     2     3     4     5     6     7
```

```
    pause

    % Disminucion niveles de gris a 4 y cuenta de valores
    [X, map] = gray2ind(I, 4);
    figure, imshow(X, map);
```

```
Warning: Image is too big to fit on screen; displaying at 67%
```

```
    title('Disminución niveles de gris a 4')
```


```
imhist(X,map)
title('Histograma')
```



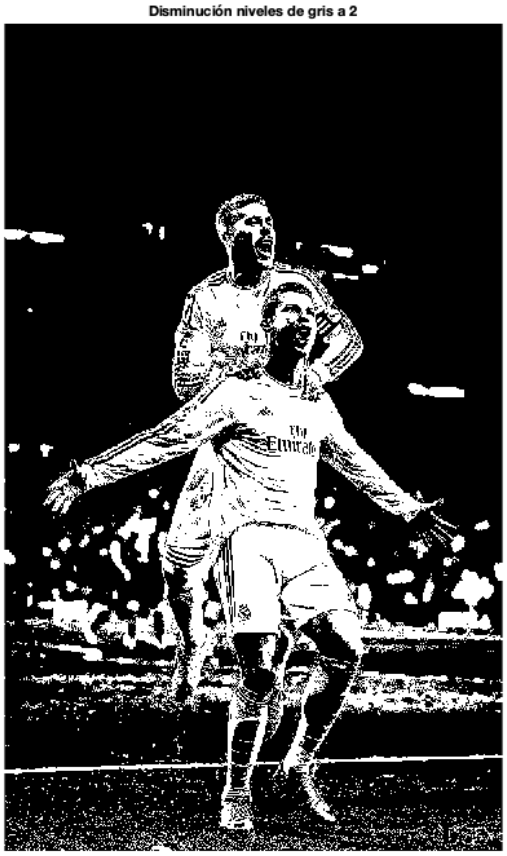

```
    [cuenta, y]=imhist(X, map)
```

```
cuenta = 4×1
      220005
      221178
       74019
       39678
y = 1×4
     0     1     2     3
```

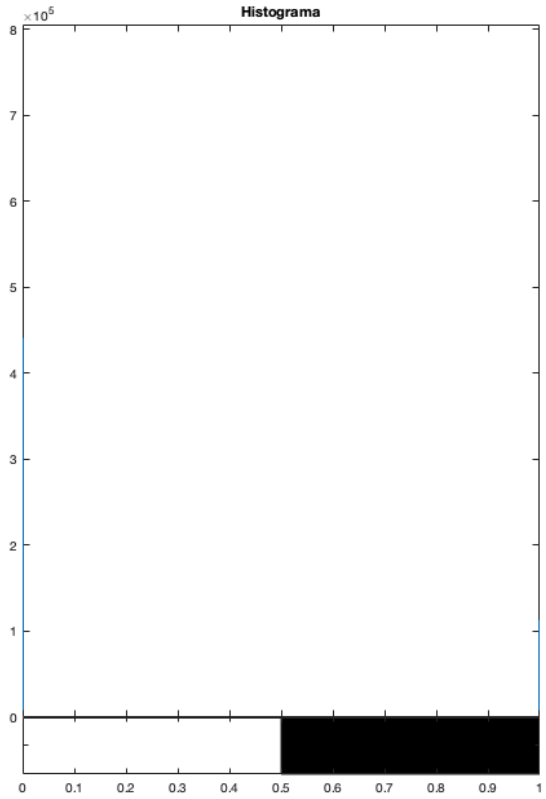
```
    pause

    % Disminucion niveles de gris a 2 y cuenta de valores
    [X, map] = gray2ind(I, 2);
    figure, imshow(X, map);
```

```
Warning: Image is too big to fit on screen; displaying at 67%
```

```
    title('Disminución niveles de gris a 2')
```




```
imhist(X,map)
title('Histograma')
```




```
    [cuenta, y]=imhist(X, map)
```

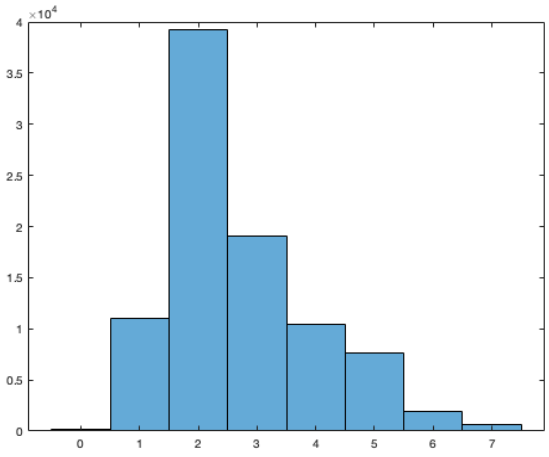
```
cuenta = 2×1
      441183
      113697
y = 1×2
     0     1
```

```
    % CONCLUSION:


    % Como podemos observar, segun aumenta el valor de gamma, empeora la
    % calidad de la imagen. En el mejor de los casos gamma es 0.5.
    % Por otro lado, cuanto mayor sea el numero de niveles de gris (maximo
    % en 16), mejor va a ser la calidad de la imagen.
```

Analisis Huffman

```
    clc; clear all; close all;

    % Primero paso mi imagen a 8 valores en escala de grises

    [imagen8,map] = gray2ind((rgb2gray(imread('imagen2.bmp'))),8);
```

```
% Obtengo filas y columnas de mi imagen

[f,c] = size(imagen8);

% Analizo el histograma
H_imagen8 = imhist(imagen8,map);
histogram(imagen8)
```




```
% Obtengo numero de muestras
muestras = f*c;

% Calculo probabilidades
for i=1:8
    p(i)=H_imagen8(i)/muestras;
end

% Usando el alfabeto de 8 valores se calcula el arbol y la tabla
alfabeto={'0' '1' '2' '3' '4' '5' '6' '7'};
[arbol, tabla]=hufftree(alfabeto, p);

% Empezamos la codificacion
array = reshape(imagen8,1,muestras);
bitString = strtrim(cellstr(num2str(array'))');

imagen_codificada = huffencode(bitString, tabla);

% Decodificamos

imagen_decodificada = huffdecode(cell2mat(imagen_codificada), arbol);

% Ratio de Compresion

II = cell2mat(bitString);
length_inicio = length(II);

IF = cell2mat(imagen_codificada);
```

```
length_final = length(IF);
%

Cr = length_inicio*3/length_final;

% Redundancia
redundancia = 1 - (1/Cr);

% CONCLUSION:
disp(tabla.val)
```

```
'2'    '0'    '7'    '6'    '5'    '4'    '1'    '3'
```

```
disp(tabla.code)
```

```
'0'    '100000'    '100001'    '10001'    '1001'    '101'    '110'    '111'
```

```
% Hemos llevado a cabo todo el proceso Huffman. Como podemos observar a
% la hora de codificar, el 2 es el valor que menos bits necesita ya que
% es a su vez el mas probable, algo que se veia ya con el histograma.
% Se puede decir que el 0 era el menos probable por lo tanto es el que
% mas bits necesita.
% En cuanto al ratio de compresion vemos que tiene un valor de 1.3
% Antes de codificar necesitamos 3 bits para cada valor, y aunque la
% longitud de salida es mayor, se codifica en binario con 1 y 0.
% La redundancia es el valor obtenido de restar a 1 la inversa del
% ratio de compresion.
```

Analisis de Transformada Discreta del Coseno

```
clc;clear all;close all;

% Primero paso mi imagen a 8 valores en escala de grises
[imagen8,map] = gray2ind((rgb2gray(imread('imagen2.bmp'))),8);

I1 = im2double(imagen8);

I1(225,:)=[];
%Elimino la ultima fila ya que tanto el numero de filas como el numero
% de columnas deben ser multiplo de 8.

T = dctmtx(8); %bloque 8x8 para aplicar la DCT
dct = @(block_struct) T * block_struct.data * T';
B = blockproc(I1,[8 8],dct); % aplico la estructura en bloques de 8x8

% Mascara que especifica los coeficientes que queremos conservar
mask = [1   1   1   1   0   0   0   0
        1   1   1   0   0   0   0   0
        1   1   0   0   0   0   0   0
        1   0   0   0   0   0   0   0
```

```
        0   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0
        0   0   0   0   0   0   0   0];

B2 = blockproc(B,[8 8],@(block_struct) mask .* block_struct.data);

% Hacemos la transformada inversa para recuperar la imagen
invdct = @(block_struct) T' * block_struct.data * T;
I2 = blockproc(B2,[8 8],invdct);

I1 = im2uint8(I1);
I2 = im2uint8(I2);

% Ratio de compresion y redundancia de datos
Cr = nnz(I1)/nnz(B2);
redundancia = 1 - (1/Cr);

% Mostramos los resultados
figure
subplot(2,1,1)
imshow(I1,map)
title('Imagen antes de la transformación DCT');
subplot(2,1,2)
imshow(I2,map)
title('Imagen recuperada tras la transformación DCT');
```


```
% CONCLUSION:

% Antes de llevar a cabo la transformacion DCT hemos tenido que
% eliminar la ultima fila de nuestra imagen, a fila 225, debido a que
% no es un numero multiplo de 8. Esto hace que la imagen no tenga la
% mayor calidad posible.
% Tras la transformacion usando la DCT vemos como la calidad es mala,
% no se aprecia muy bien la imagen.
% Nos ha salido un ratio de compresion de unos 6.4, bastante elevado,
```

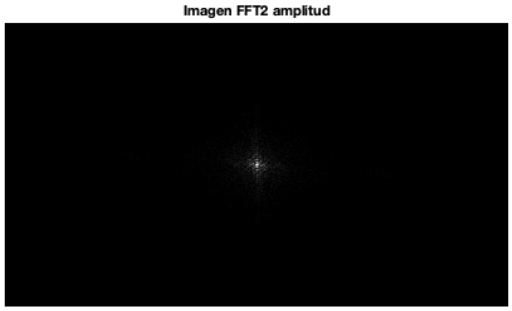
```
% significando que la imagen se ha comprimido en gran cantidad,
% necesitamos menos bits para representarla.
```

Analisis con otras transformadas

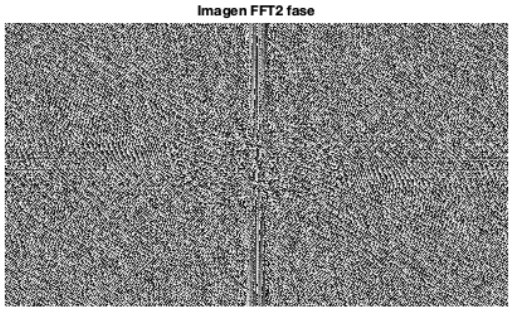
```
clc;clear all;close all;
```

Transformada de Fourier

```
original = imread('imagen2.bmp');
I=rgb2gray(original);
fftI= fft2(double(I));

%Muestra la amplitud y fase de la TTF 2D

figure, imshow(abs(fftshift(fftI)), [24 1000000]), colormap gray
title('Imagen FFT2 amplitud');
```




```
figure, imshow(angle(fftshift(fftI)),[-pi pi]), colormap gray
title('Imagen FFT2 fase');
```




```
%Inversa 2D FFT
imagen_rec = ifft2(fftI);
```

```
%Calcular los límites para representar
cmin = min(min(abs(imagen_rec)));
cmax = max(max(abs(imagen_rec)));

%Display switched images
figure, imshow(abs(imagen_rec), [cmin cmax]), colormap gray
title('Image recuperada amplitud');
```


```
% CONCLUSION:

% Hemos obtenido la fase y la ampitud de mi transformada discreta de
% fourier aplicada a la imagen original (convertida a escala de
% grises), y luego hemos recuperado la imagen mediante la inversa de la
% FFT.
```

Wavelet

```
clc;clear all;close all;

X=imread('imagen2.bmp');

I=rgb2gray(X);

% inputting the decomposition level and name of the wavelet
% See Wavelet Families in MATLAB

n=input('Enter the decomposition level:   ');
wname = input('Enter the name of the wavelet:   ');
 x = double(I);
NbColors = 255;
map = gray(NbColors);
 x = uint8(x);
%Conversion of RGB to Grayscale
% x = double(X);
% xrgb  = 0.2990*x(:,:,1) + 0.5870*x(:,:,2) + 0.1140*x(:,:,3);
% colors = 255;
% x = wcodemat(xrgb,colors);
% map = pink(colors);
% x = uint8(x);
```

```
    % A wavelet decomposition of the image

    %wname ? Name of orthogonal or biorthogonal wavelet
    %Names to use:   'haar' | 'db1' | 'db2' | 'coif1' | 'coif2' | ...

    [c,s] = wavedec2(x,n,wname); % More details in definition of wavedec2
    % wdcbm2 for selecting level dependent thresholds
    alpha = 1.5; m = 2.7*prod(s(1,:));
    [thr,nkeep] = wdcbm2(c,s,alpha,m)
```

```
thr = 3×3
   61.0000   71.7500   44.6250
   61.0000   71.7500   44.6250
   61.0000   71.7500   44.6250
nkeep = 1×3
         489         753        1384
```

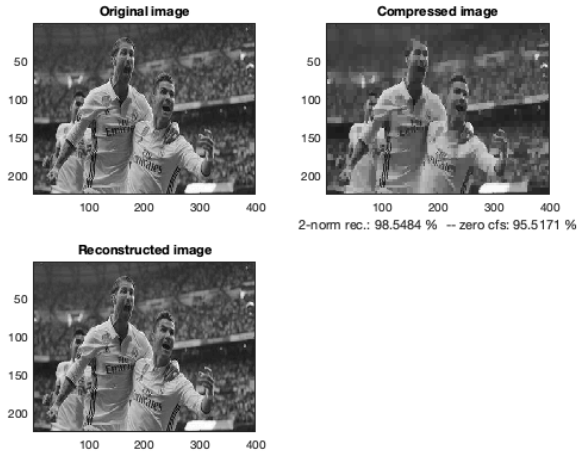
```
    % Compression
    [xd,cxd,sxd,perf0,perfl2] = wdencmp('lvd',c,s,wname,n,thr,'h');
    disp('Compression Ratio');
```

```
Compression Ratio
```

```
    disp(perf0);
```

```
   95.5171
```

```
    % Decompression
    R = waverec2(c,s,wname);
     rc = uint8(R);

    % Plot original and compressed images.
    subplot(221), image(x);
    colormap(map);
    title('Original image')
    subplot(222), image(xd);
    colormap(map);
    title('Compressed image')

    % Displaying the results
    xlab1 = ['2-norm rec.: ',num2str(perfl2)];
    xlab2 = [' %  -- zero cfs: ',num2str(perf0), ' %'];
    xlabel([xlab1 xlab2]);
    subplot(223), image(rc);
    colormap(map);
    title('Reconstructed image');
```




```
%Computing the image size

disp('Original Image');
```

```
Original Image
```

```
imwrite(x,'original.tiff');
imfinfo('original.tiff')
```

```
ans = struct with fields:
                    Filename: '/Users/guillermoaldreypastor/Desktop/CUARTO/SERVICIOS TELEMATICOS/P5/orig
                FileModDate: '20-nov-2020 22:57:07'
                    FileSize: 89426
                      Format: 'tif'
               FormatVersion: []
                       Width: 400
                      Height: 225
                    BitDepth: 8
                   ColorType: 'grayscale'
             FormatSignature: [73 73 42 0]
                   ByteOrder: 'little-endian'
              NewSubFileType: 0
               BitsPerSample: 8
                 Compression: 'PackBits'
    PhotometricInterpretation: 'BlackIsZero'
                 StripOffsets: [8 7436 15223 23239 31299 39359 47370 55216 63248 71269 79229 87184]
             SamplesPerPixel: 1
                RowsPerStrip: 20
             StripByteCounts: [7428 7787 8016 8060 8060 8011 7846 8032 8021 7960 7955 1956]
                 XResolution: 72
                 YResolution: 72
              ResolutionUnit: 'Inch'
                    Colormap: []
          PlanarConfiguration: 'Chunky'
                   TileWidth: []
                  TileLength: []
                 TileOffsets: []
              TileByteCounts: []
                 Orientation: 1
                   FillOrder: 1
           GrayResponseUnit: 0.0100
```

```
              MaxSampleValue: 255
              MinSampleValue: 0
                Thresholding: 1
                      Offset: 89140
```

```
disp('Compressed Image');
```

```
Compressed Image
```

```
imwrite(xd,'compressed.tiff');
imfinfo('compressed.tiff')
```

```
ans = struct with fields:
                    Filename: '/Users/guillermoaldreypastor/Desktop/CUARTO/SERVICIOS TELEMATICOS/P5/comp
                FileModDate: '20-nov-2020 22:57:07'
                    FileSize: 2100
                      Format: 'tif'
               FormatVersion: []
                       Width: 400
                      Height: 225
                    BitDepth: 8
                   ColorType: 'grayscale'
             FormatSignature: [73 73 42 0]
                   ByteOrder: 'little-endian'
              NewSubFileType: 0
               BitsPerSample: 8
                 Compression: 'PackBits'
    PhotometricInterpretation: 'BlackIsZero'
                 StripOffsets: [8 168 328 488 648 808 968 1128 1290 1450 1614 1774]
            SamplesPerPixel: 1
               RowsPerStrip: 20
            StripByteCounts: [160 160 160 160 160 160 160 162 160 164 160 40]
                XResolution: 72
                YResolution: 72
             ResolutionUnit: 'Inch'
                   Colormap: []
          PlanarConfiguration: 'Chunky'
                  TileWidth: []
                 TileLength: []
                TileOffsets: []
             TileByteCounts: []
                Orientation: 1
                  FillOrder: 1
           GrayResponseUnit: 0.0100
              MaxSampleValue: 255
              MinSampleValue: 0
                Thresholding: 1
                      Offset: 1814
```

```
disp('Decompressed Image');
```

```
Decompressed Image
```

```
imwrite(rc,'decompressed.tiff');
imfinfo('decompressed.tiff')
```

```
ans = struct with fields:
                    Filename: '/Users/guillermoaldreypastor/Desktop/CUARTO/SERVICIOS TELEMATICOS/P5/deco
                FileModDate: '20-nov-2020 22:57:07'
```

```
                FileSize: 89426
                  Format: 'tif'
           FormatVersion: []
                   Width: 400
                  Height: 225
                BitDepth: 8
               ColorType: 'grayscale'
         FormatSignature: [73 73 42 0]
               ByteOrder: 'little-endian'
          NewSubFileType: 0
           BitsPerSample: 8
             Compression: 'PackBits'
PhotometricInterpretation: 'BlackIsZero'
            StripOffsets: [8 7436 15223 23239 31299 39359 47370 55216 63248 71269 79229 87184]
         SamplesPerPixel: 1
             RowsPerStrip: 20
         StripByteCounts: [7428 7787 8016 8060 8060 8011 7846 8032 8021 7960 7955 1956]
             XResolution: 72
             YResolution: 72
          ResolutionUnit: 'Inch'
                Colormap: []
      PlanarConfiguration: 'Chunky'
               TileWidth: []
              TileLength: []
             TileOffsets: []
          TileByteCounts: []
             Orientation: 1
               FillOrder: 1
        GrayResponseUnit: 0.0100
          MaxSampleValue: 255
          MinSampleValue: 0
            Thresholding: 1
                  Offset: 89140
```

```
% CONCLUSION:

% En este apartado en el que hemos llevado a cabo Wavelet, empezamos
% introduciendo por pantalla el valor de 3 como nivel de
% descomposicion, y la familia de wavelet 'haar'.
% Vemos como al comprimir se pierde un poco de calidad, y que al final
% la descompresion es bastante buena pues se asemeja bastante a la
% imagen original.

% Cabe mencionar que hemos pasado la imagen original a escala de
% grises, pero que si no llevabamos a cabo este apartado, la foto de la
% imagen comprimida aparecia en blanco con una serie de puntitos, algo
% parecido a lo que salia en la explicacion de la practica en clase.
```

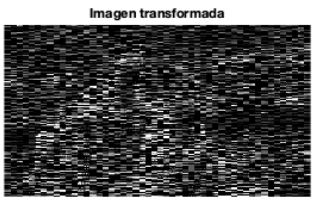
Opcional - KLT

```
clc; close all; clear all;

% Es color
original = imread('imagen2.bmp');
figure(1), imshow(original)
```

```
I=rgb2gray(original);

% Si ya es gris
%original=imread('cameraman.tif');
%I=original;

figure(2), imshow(I)
```

```
I=im2double(I);
I(225,:) = [];
[alto,ancho]=size(I);


m=1;
for i=1:8:alto
    for j=1:8:ancho
        for x=0:7
            for y=0:7
            img(x+1,y+1)=I(i+x,j+y);
            end
        end
            k=0;
            for l=1:8
                img_expect{k+1}=img(:,l)*img(:,l)';
                k=k+1;
            end
            imgexp=zeros(8:8);
            for l=1:8
```

```
                imgexp=imgexp+(1/8)*img_expect{l};%expectation of E[xx']
            end
            img_mean=zeros(8,1);
            for l=1:8
                img_mean=img_mean+(1/8)*img(:,l);
            end
            img_mean_trans=img_mean*img_mean';
            img_covariance=imgexp - img_mean_trans;
            [v{m},d{m}]=eig(img_covariance);
            temp=v{m};
             m=m+1;
            for l=1:8
                v{m-1}(:,l)=temp(:,8-(l-1));
            end
             for l=1:8
           trans_img1(:,l)=v{m-1}*img(:,l);
             end
           for x=0:7
               for  y=0:7
                   transformed_img(i+x,j+y)=trans_img1(x+1,y+1);
               end
           end
mask=[1 1 1 1 1 1 1 1
      1 1 1 1 1 1 1 1
      1 1 1 1 1 1 1 1
      1 1 1 1 1 1 1 1
      1 1 1 1 1 1 1 1
      1 1 1 1 1 1 1 1
      1 1 1 1 1 1 1 1
      1 1 1 1 1 1 1 1 ];
  trans_img=trans_img1.*mask;
           for l=1:8
           inv_trans_img(:,l)=v{m-1}'*trans_img(:,l);
           end
            for x=0:7
               for  y=0:7
                  inv_transformed_img(i+x,j+y)=inv_trans_img(x+1,y+1);
               end
           end


        end
end

figure
subplot(2,1,1)
imshow(transformed_img);
title('Imagen transformada');
subplot(2,1,2)
imshow(inv_transformed_img);
title('Inversa de la trnasformada');
```



Conclusiones Generales

```
% Me ha parecido una practica interesante porque es bastante dinamica y no
% resulta pesada de hacer. Se puede entender que sucede en los diferentes
% aspectos de la compresion de una imagen bmp de forma sencilla.

% El apartado de Huffman me ha gustado porque como no teniamos codigo dado,
% cada uno hemos tenido que aplicar lo esudiado en clase y tirar de
% imaginacion para sacar los puntos que se pedian.

% En resumen, el hecho de poder tratar con imagenes creo que es mas
% divertido y dinamico que cuando trabajabamos con audio.
```