Grado en Ingeniería de Tecnologías de Telecomunicación
Fundamentos de Sonido e Imagen
Bloque 2: Fundamentos de Imagen

Práctica 3 - Compresión de imagen y video

Objetivos

El objetivo de la práctica consiste en comprender una serie de principios operativos básicos aplicables a la compresión de imagen y video con Matlab. Para ello se programarán rutinas sencillas que permitan profundizar en los principios de compresión mediante transformada y cuantificación selectiva del estándar JPEG. Del mismo modo, se estudiarán técnicas básicas de estimación y compensación de movimiento que constituyen la base de la compresión temporal en secuencias de video.

1. Compresión con pérdidas de imágenes estáticas

En esta sección se pretende implementar una versión sencilla del modo secuencial de JPEG. Para ello, se tendrán en cuenta las siguientes ideas:

1. Trabajaremos con una imagen monocroma de tamaño suficiente. Cargue la imagen *cameraman.tif* y calcule su DCT por bloques tal y como especifica el estándar. El tamaño de los bloques será de 8×8. A los valores de intensidad de dichos bloques se les resta una componente continua genérica de valor 128. Sobre los valores resultantes realiza una DCT. En Matlab, esto se programa fácilmente haciendo uso de las rutinas `blockproc` y `dct2`.

2. Considere dos formas de selección de coeficientes:

   - La primera (digamos, modo 1) de ellas permitirá seleccionar un porcentaje de coeficientes a determinar, ordenando los coeficientes según el modo zig-zag (es decir, si decide escoger sólo K coeficientes, seleccione los K primeros en ordenación zig-zag). Para ello puede hacer uso de la función `zigzag.m`.
   - La segunda (modo 2) llevará a cabo la cuantificación y selección de coeficientes prevista en el estándar, acorde con los niveles de cuantificación disponibles en el fichero `tabla.dat`.

3. Observe el efecto que produce el recorte de coeficientes de la DCT en la imagen reconstruida empleando el modo 1. Extraiga las oportunas conclusiones.

4. Observe estos efectos empleando el modo 2.

5. Para el modo 1, obtenga una curva que represente la relación señal a ruido de pico (PSNR) como función del porcentaje de coeficientes seleccionados. La Relación Señal a Ruido de Pico (*Peak Signal to Noise Ratio*, PSNR) es una medida de calidad que se obtiene como

$$\mathrm{PSNR}(I, J) = 10 \log_{10} \left(\frac{\mathrm{máx}(I[m, n])^2}{\mathrm{MSE}(I, J)} \right)$$

con $I[m,n]$ la imagen original y $J[m,n]$ su versión degradada. $\mathrm{MSE}(I,J)$ es el error cuadrático medio de ambas imágenes, y puede estimarse como

$$\mathrm{MSE}(I,J) = \frac{1}{MN}\sum_{m=1}^{M}\sum_{n=1}^{N}\left(I[m,n]-J[m,n]\right)^2 .$$

Determine la tasa de compresión– si es que existe– a partir de la cual, a su juicio, la compresión sería visualmente sin pérdidas.

6. Calcule la PSNR obtenida con el método 2. Compare el valor de PSNR obtenido con este método con el valor de PSNR correspondiente al punto seleccionado por Vd.–en su caso– en el apartado anterior.

2. Compresión mediante predicción por compensación de movimiento

En esta sección vamos a simular un predictor de movimiento simple para comprender las bases de este tipo de predicción temporal. Para ello:

1. Utilice el ejemplo de código proporcionado en el siguiente listado para cargar y visualizar un video con datos *raw* en formato YUV 4:2:0. La función `yuv_import` se encuentra disponible en el campus virtual.

```
clear all
close all
%%Lectura de video YUV RAW
%Formato QCIF europeo, 4:2:0
height=144;
width=176;
nframes=300;
%Lectura de las componentes de video
[Y,U,V]=(yuv_import('akiyo_qcif.yuv',[width height],nframes));

%Conversión video apto para visualización

vid=zeros(height,width,3,nframes);

for k=1:nframes
        %Resampling de croma para visualización
        U{k}=imresize(U{k},[height width]);
        V{k}=imresize(V{k},[height width]);
        vid(:,:,1,k)=(Y{k});
        vid(:,:,2,k)=(U{k});
        vid(:,:,3,k)=(V{k});
        vidrgb(:,:,:,k)=(ycbcr2rgb(uint8(vid(:,:,:,k))));
        video(k)=im2frame(uint8((vidrgb(:,:,:,k))));
end

%Reproducción del video
        implay(video,30)
```

2. Para la estimación y compensación de movimiento, trabajaremos inicialmente sobre la señal de luminancia. Almacene en una imagen `Iref` la imagen de luminancia

correspondiente al *frame* número 10. Como *frame* actual `Iact`, consideraremos inicialmente el número 15. Obtenga el error cuadrático medio entre ambas imágenes. Puede utilizar la función proporcionada `costFuncMSE`. Visualice ambas imágenes en la misma figura.

3. Para crear una imagen con movimiento compensado, vamos a utilizar dos funciones disponibles en el campus virtual. La función `estmov` realiza lo siguiente:

   a) Considera para cada macrobloque de tamaño a definir (usaremos 16×16) en la imagen de destino una región de búsqueda en la imagen de referencia que contenga $(2p+32) \times (2p+32)$ píxeles. El centro de esa región coincidirá con la posición del centro del macrobloque en la imagen de referencia (ver figura 1). Para esta práctica usaremos $p = 3$.

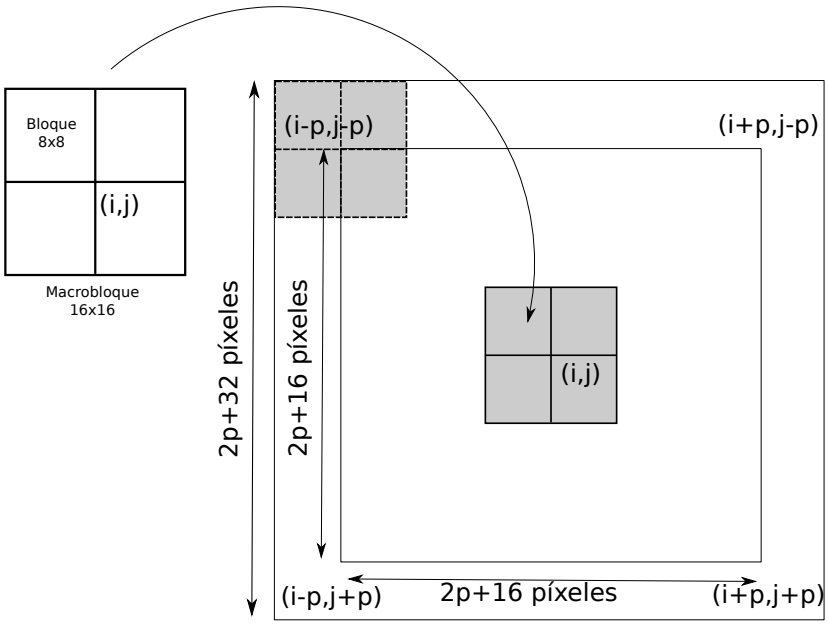

Figura 1: Región de búsqueda en imagen de referencia

   b) Compara dicho macrobloque con **todos** los macrobloques de la región de búsqueda en términos de MSE. Para ello, utiliza la fucnión `costFuncMSE` referenciada más arriba.

   c) Obtiene la posición del macrobloque de la imagen destino cuya comparación con el macrobloque de referencia ha dado lugar al menor MSE. A este macrobloque nos referiremos como macrobloque objetivo. A partir de dicha posición, obtiene las coordenadas del vector de movimiento.

   d) Realizado lo anterior para todos los macrobloques, la función devuelve los vectores de movimiento para todos ellos como salida única.

   Por otra parte, la función `compmov` recibe como parámetros la imagen de referencia, el tamaño de macrobloque, y los vectores de movimiento devueltos por `estmov` y devuelve una imagen con movimiento compensado.

   Utilice las funciones descritas con los parámetros indicados para crear una imagen con movimiento compensado `Icomp`. Visualice esta imagen conjuntamente con las dos del ejercicio anterior.

4. Visualice de forma conjunta la diferencia entre la imagen actual y: 1) La imagen referencia; 2) La imagen con movimiento compensado. Llámelas `imdiff` e `imdiffcomp` respectivamente. Calcule el valor cuadrático medio de ambas imágenes y comente los resultados obtenidos.

5. Comprima ambas imágenes diferencia utilizando el modo 2 de JPEG implementado en la sección anterior. Utilice la imagen referencia y los vectores de movimiento para reconstruir la imagen actual en ambos casos a partir de las diferencias comprimidas. Visualice los resultados de forma paralela con la imagen actual. Calcule la PSNR en ambos casos y comente los resultados.

6. Repita los ejercicios 3–5 empleando en este caso el *frame* 200 para la imagen actual. Mantenga la misma referencia.

7. (*Opcional.*) Comprima el video proporcionado utilizando una distancia $N = 12$ entre *frames* de tipo I y asumiendo que el resto de los *frames* son de tipo P. Los *frames* I deben comprimirse tal cual con el modo 2 de JPEG implementado en la sección anterior. Los *frames* P se codificarán a partir del I o P anterior utilizando la metodología de estimación y compensación de movimiento desarrollada en los ejercicios anteriores. Visualice el video comprimido y calcule la PSNR promedio a partir de la PSNR calculada en todos los *frames*. Nota: Utilice para las señales de crominancia los mismos vectores de movimiento que para las de luminancia.