1er Desafío tecnológico

Universidad de Málaga

Jesús Hermoso de Mendoza Hernández

Introducción

Una primera aproximación al problema nos muestra que la imagen tiene grandes áreas de colores muy similares.
Por tanto, lo primero que podemos hacer es convertir la imagen a modo indexado, reduciendo el número de colores a 256. Con este fin, podemos usar el comando:

[imagen, mapa] = rgb2ind(imagen_original, 256)

Previamente hemos leido la imagen en Matlab con el comando imread.
Dado que la imagen original está en modo RGB, tiene 3 canales (rojo, azul y verde) cada uno con una matriz asociada. Con esto conseguimos pues reducir el tamaño de la imagen a una tercera parte aproximadamente (podemos considerar el tamaño del mapa despreciable en relación a lo que ocupa la "imagen real"), entorno a los 11-12 MB.
El código en MATLAB quedaría de la siguiente manera:

```
function comprime
RGB = imread('Imagendesafio.tif');
[imagen,mapa] = rgb2ind(RGB,colores);
clear RGB

save('comprimida', 'imagen',  'mapa')
msgbox('La imagen se ha comprimido correctamente')
```

```
function descomprime
load ('comprimida')
imwrite(imagen,mapa, 'descomprimida.tif')

msgbox('La imagen se ha descomprimido correctamente')
```

Comprimiendo la imagen

Una vez obtenida la imagen indexada, algo que llama la atención es que hay muchos píxeles consecutivos que tienen exactamente el mismo valor, por ello podemos comprimir usando Run Length Encoding.
El formato NUMERO/REPETICIONES, eneste caso en concreto es muy ineficiente, porque el mínimo de bits que MATLAB necesita para almacenar un dato son 8 bits, y por otro lado está el problema que en los casos de píxeles aislados (255 – 255 – 255 – **240** -252 – 252 por ejemplo), el byte original se transforma en 2 bytes, lo cual no es deseable.

Para ello, nos ayudamos de una matriz auxiliar **indice**, que marca el inicio y final de cada racha con un "1". Y en el interior, los "0" se pone el mismo numero, el cual se guarda en el **vector imagen,** que esta comprimido.

Al descomprimir la imagen, se realiza el proceso inverso, y se van trasponiendo las filas, hasta tener la imagen original.

Reduciendo los colores

Si reducimos aún mas los colores, vemos que las rachas son más largas y por tanto podemos cmprimir más, por tanto debemos buscar un equilibrio entre tamaño y calidad de imagen descomprimida. Aqui vemos una tabla comparativa viendo el resultado de ejecutar con varios valores de comprime(colores)

	comprimida(MB)
comprime(256)	6,59
comprime(128)	5,22
comprime(64)	4,22
comprime(32)	3,24
comprime(16)	2,46
comprime(8)	1,97

Como solución que no compromete la calidad de la imagen, podemos coger la que genera **comprime(32)**, ya que tiene una buena relacion entre calidad y tamaño, unos 3 MB.
A partir de ahí, la calidad empieza a empeorar y se nota bastante la degradación de colores.

Bibliografía utilizada

[1] Digital Image Processing Using Matlab. Rafael C. Gnzalez, Richard Woods. (Capítulos 1, 6 y 8)

[2] Tratamiento digital de voz e imagen. Marcos Faundez Zauny

[3] http://ocw.innova.uned.es/mm2/tcm/contenidos/pdf/tema3.pdf

[4] http://es.wikipedia.org/wiki/Compresi%C3%B3n_de_imagen