LA COMPRESIÓN JPEG

Jordi Cenzano – 09/2001

INTRODUCCIÓN

La compresión JPEG (Joint Photographic Experts Group) es una herramienta para comprimir de imágenes muy popular, se utiliza para cualquier propósito en el que se necesite transmitir o almacenar una imagen y se disponga de un ancho de banda o memoria limitada. Se empezó a gestar el año 1982, y no vio la luz hasta el 1991 donde se editaron las primeras recomendaciones del estándar.

El JPEG es un método de compresión con pérdidas, es decir nunca podremos recuperar exactamente la imagen original, aunque una de las grandes virtudes del JPEG es que estas pérdidas son variables y directamente proporcionales a la compresión, como veremos más adelante.

Este es un sistema de compresión donde conocemos de antemano el elemento a transmitir, es un sistema diseñado para la compresión de imágenes, al saber que se utilizará solo para transmitir imágenes podemos permitir que tenga pérdidas, cosa que inaceptable en archivos de datos, lo que se tiene que intentar es que estas perdidas sean lo menos perceptibles para el ojo humano, veremos como se consigue esto.

Existen diferentes variantes del sistema del JPEG: Secuencial, progresivo, jerárquico. En este articulo estudiaremos la secuencial o baseline.

EL SISTEMA JPEG BASELINE :

En este articulo explicaremos el JPEG-Baseline para una imagen de 256 niveles de gris.

El diagrama de bloques en el que nos basaremos para explicar esta compresión es el de la figura 1.

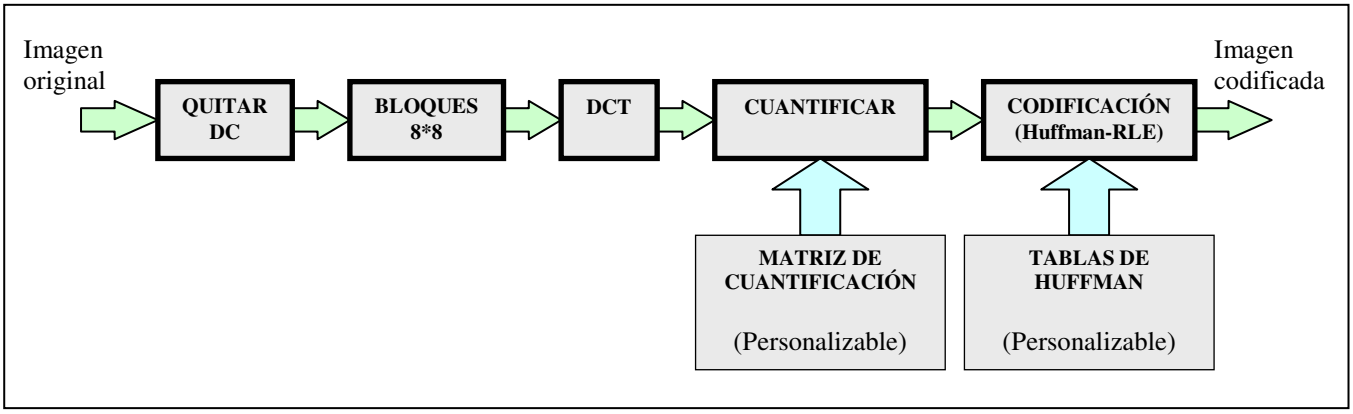

Figura 1

El primer bloque **"QUITAR DC"** resta a cada pixel de la imagen original 128, es decir pasamos de tener un rango de 0 a 255 a uno de –128 a127 (8 bits).

El bloque **"BLOQUES 8*8"** simplemente divide la imagen en bloques de 8*8 pixels. Esto impone una condición a la imagen original, que tanto la altura como la anchura sean múltiplos de 8, si esto no se cumpliera se rellenarían los pixels que falten con 0, es decir negro. Estos pixels se quitan al descomprimir.

El siguiente bloque **"DCT"** es el más importante del esquema, en el aplicamos a las subimágenes de 8*8 pixels la transformada coseno discreta (DCT). Con ella pasamos del dominio espacial de la imagen original a un dominio transformado donde podemos discernir el contenido frecuencial de la imagen original.

El resultado de la DCT de un bloque de 8*8 pixels es también un bloque de 8*8 en el que los valores cercanos al coeficiente [0,0] (arriba a la izquierda de la matriz) indican baja frecuencia y como más nos alejamos de este coeficiente mas alta frecuencia representa.

La DCT la hacemos con una precisión de 3 bits adicionales a la precisión de la imagen de entrada (11bits = 8 + 3). Es decir la DCT tiene un rango de -1024 ... +1023

Por supuesto existe una expresión, la IDCT con la cual podemos recuperar la imagen original a partir de los coeficientes obtenidos con la DCT.

En la figura 2 vemos la expresión de la DCT en 2 dimensiones.

$$C[k,l] = \alpha[k] \cdot \alpha[l] \cdot \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} x[n,m] \cdot \cos\left(\frac{(2n-1)k\pi}{2N}\right) \cdot \cos\left(\frac{(2m-1)l\pi}{2M}\right) \cdot$$

Figura 2

En la figura 3 tenemos un cuadro de 8x8 pixels al cual le hemos aplicado la DCT, el resultado de esta la hemos puesto en forma de gráfica 2D donde la altura nos marca el valor del coeficiente. En este ejemplo vemos que los coeficientes más altos están cerca del origen (continua), los demás son muy pequeños, este es el caso más común.

Tenemos que tener mucho cuidado al codificar el coeficiente [0,0] (nivel de DC) ya que es el que marca el nivel de gris global del cuadro de 8*8, si hubiera errores al codificar este coeficiente veríamos los cuadros de 8*8 de la imagen con diferentes niveles de gris.

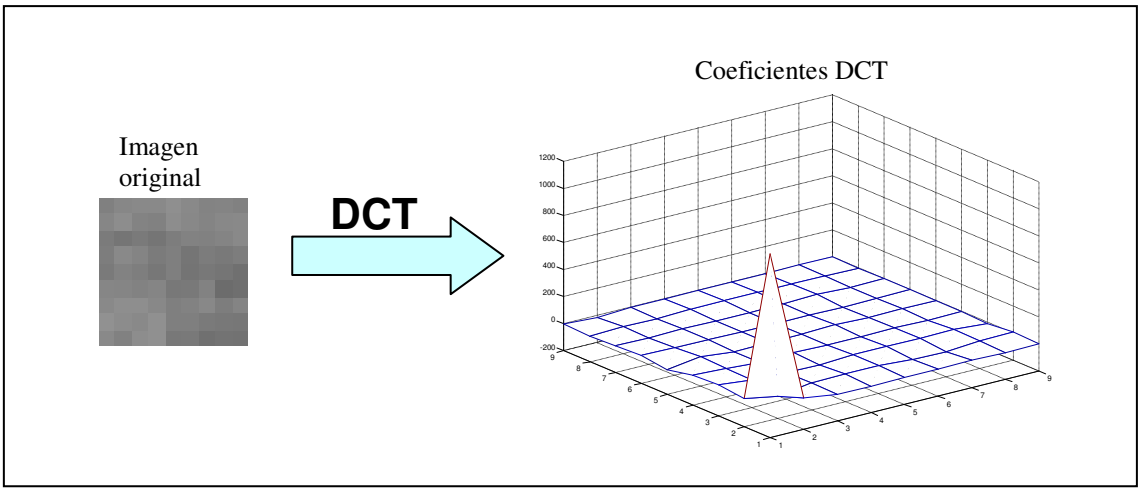

Figura 3

El bloque "**CUANTIFICAR**" es el encargado de fijar la calidad de la imagen, es decir su nivel de compresión. A más calidad menos compresión.

Una vez tenemos las matrices de 8*8 resultantes de la DCT los divido por la matriz de cuantificación Q_Y y redondeo.

Como se ha explicado antes el coeficiente más importante es el primero, este coeficiente tiene un trato especial. Se codifica de forma diferencial, es decir al valor del coeficiente [0,0] de la matriz resultante de la DCT se le resta el valor del mismo coeficiente de la matriz anterior.

En la figura 4 hemos supuesto que el valor [0,0] de la matriz resultante de la DCT anterior era 1000, el valor de la matriz de la DCT actual es 888.5(valor de la DCT de **IMAGEN**) . Entonces el valor a codificar es 888.5 – 1000 = -111.5.

Lo que estamos haciendo al codificar este coeficiente de forma diferencial es aumentar su rango dinámico, si antes podían variar entre -1024 ... +1023, el coeficiente de DC podrá variar entre -2047 ... +2047. En la figura 4 tenemos un ejemplo para clarificar un poco las ideas.

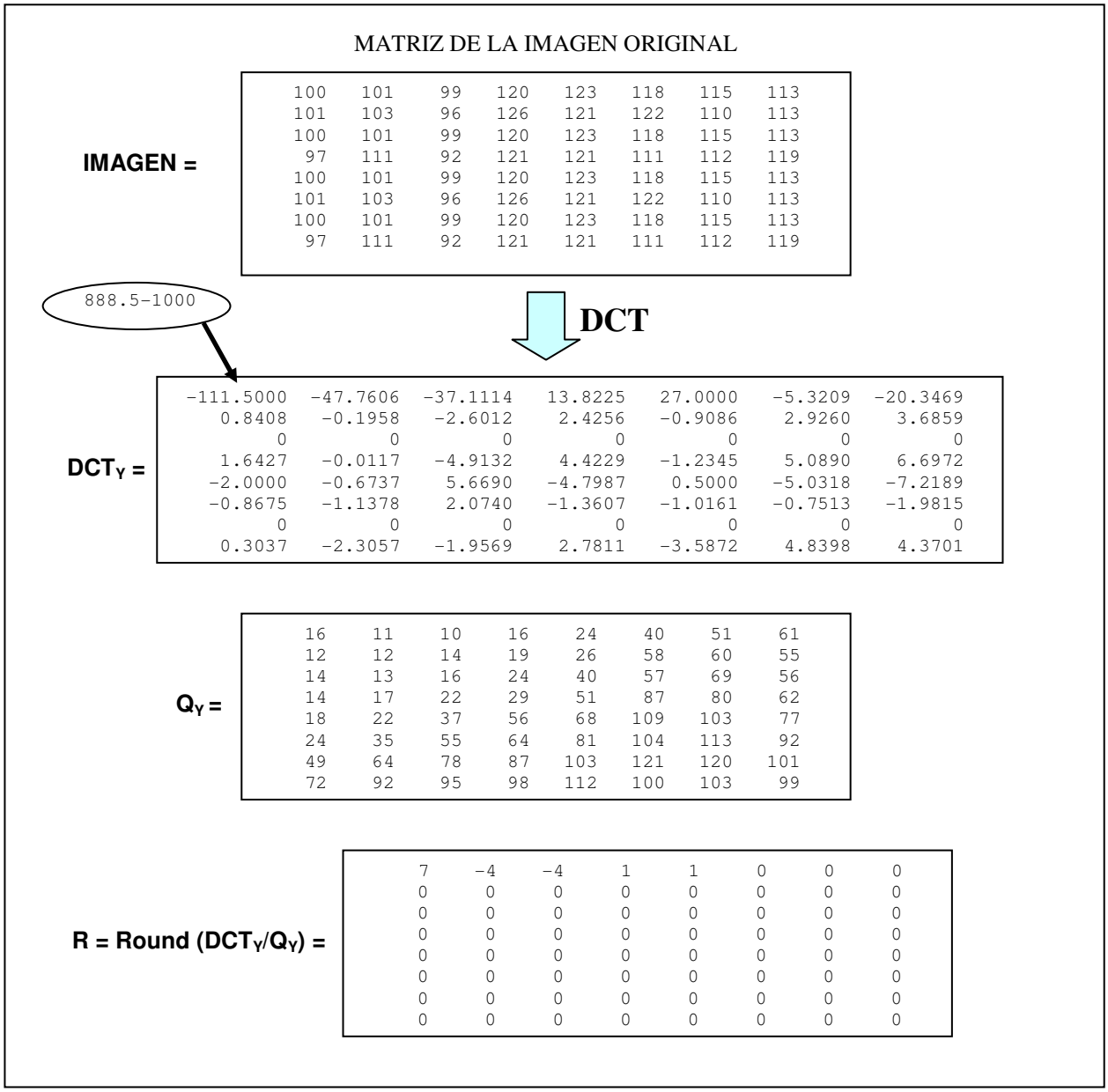

Figura 4

La matriz Q_Y es la matriz recomendada para la cuantificación de los coeficientes de la DCT(en el caso que la imagen sea de escala de grises), se llama matriz de Lohscheller.

En este paso está el "truco" de la compresión JPEG ya que dividiendo por Qy he conseguido que la mayoría de coeficientes sean 0, y utilizando un buen método de codificación puedo codificar esta matriz con muchos menos bytes de los 64 de la imagen original.

Como la matriz de cuantificación Qy es definible por el usuario, como mayores pongamos los elementos de esta matriz más ceros conseguiremos en la matriz R, y por consiguiente más comprimido quedará el archivo, pero más pérdidas tendremos.

El bloque **"CODIFICACIÓN"** se encarga de poner la información de la matriz R de forma que ocupe el menor "espacio" posible.

El primer paso que realiza este bloque es poner todos los coeficientes de la matriz en forma de vector de 64 elementos, para colocar los coeficientes en el vector se sigue una exploración en zig-zag de la matriz R, como se ilustra en la figura 5.

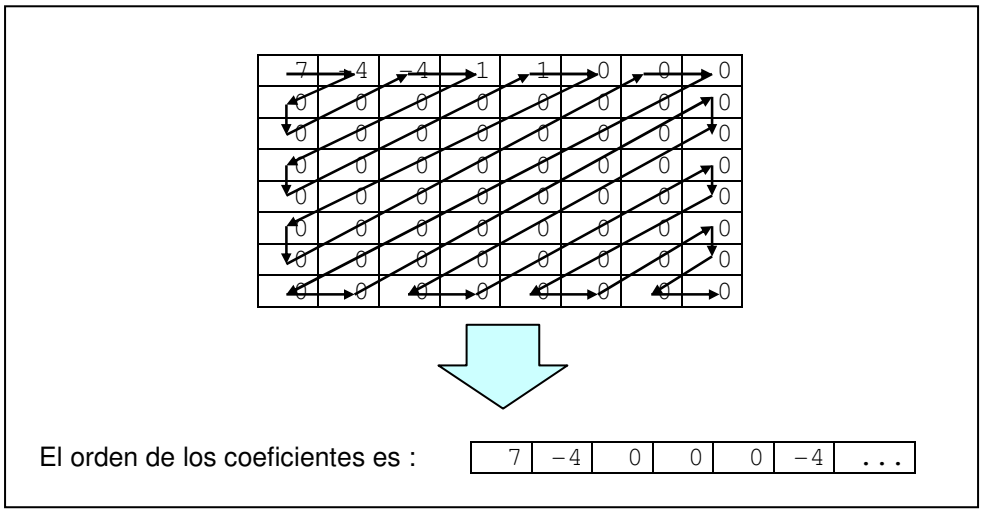

Figura 5

El coeficiente de DC en el momento de la codificación también recibe un trato especial, la codificación de los elementos de la matriz R la explicaremos en dos pasos:

Codificación del primer elemento (DC): Para codificar este elemento se utilizan 2 palabras a la primera la llamaremos N y a la segunda B. La palabra de tipo N indica el numero de bits con que se va a codificar el coeficiente de DC y la palabra B es el coeficiente codificado en complemento a 1. Para pasar de N a binario utilizamos unas tablas de Huffman que nos indica el estándard, en la figura 6 vemos las tablas de Huffman recomendadas para la luminancia, y en la figura 7 tenemos un par de ejemplos de codificación de coeficientes DC.

N	Código (C)
0	00
1	010
2	011
3	100
4	101
5	110
6	1110
7	11110
8	111110
9	1111110
10	11111110
11	111111110

Figura 6

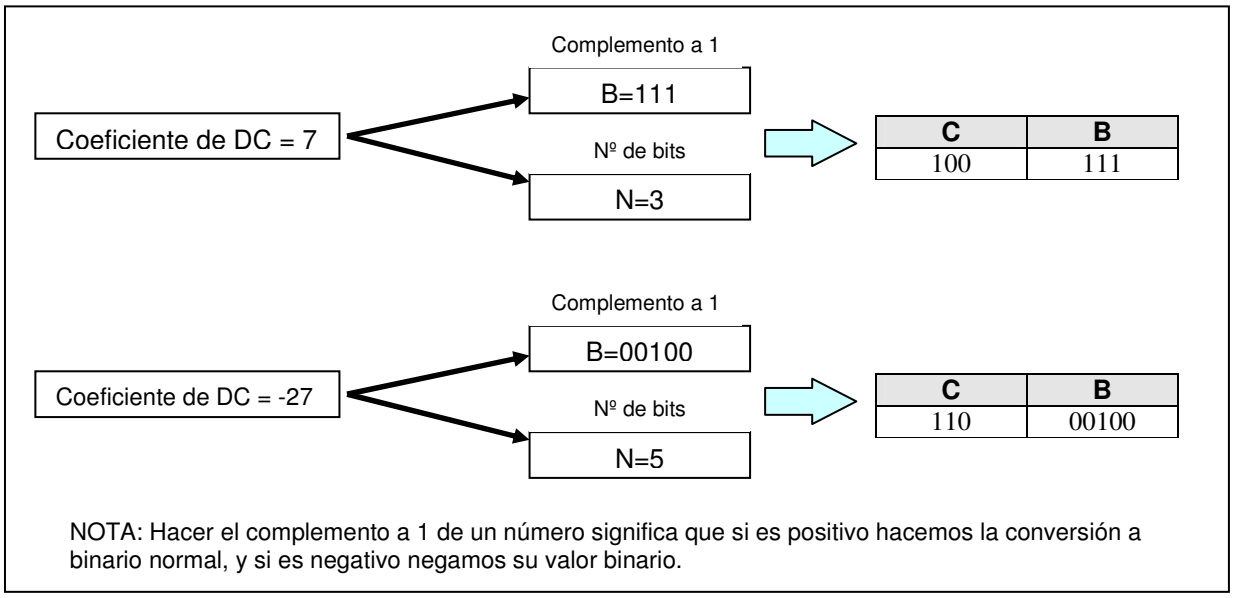

Figura 7

La codificación de los coeficientes AC es algo más compleja, también de utilizan 2 palabras, a la primera la llamaremos N_{dc} y a la segunda B. N_{dc} es una palabra compuesta que indica el número de 0 que hay antes del siguiente coeficiente no nulo y el número de bits con el que se codificará el coeficiente siguiente. El numero máximo de 0 seguidos que el protocolo permite codificar son 16, en caso que hubiera más se codificarían con 2 o mas palabras N_{dc} donde indicaríamos que el número de 0 son 15 y el número de bits son 0 (esta combinación de valores de N_{dc} indica que hay 16 ceros).

Una vez tenemos las dos componentes de la palabra Ndc, entonces buscaremos su valor binario en las tablas de Huffman para los coeficientes AC de luminancia, tenemos un fragmento de estas tablas en la figura 8.

La palabra tipo B es el coeficiente en complemento a 1,igual que en el caso de DC. En la figura 9 tenemos un ejemplo de todo lo anterior.

N_{dc}		**Código**
Nº de ceros	Bits del coef.	
0	0	(EOB)1010
0	1	00
0	2	01
0	3	100
0	4	1011
0	5	11010
...	...	...
2	7	1111111110001011
2	8	1111111110001100
2	9	1111111110001101
2	10	1111111110001110
...	...	...

Figura 8

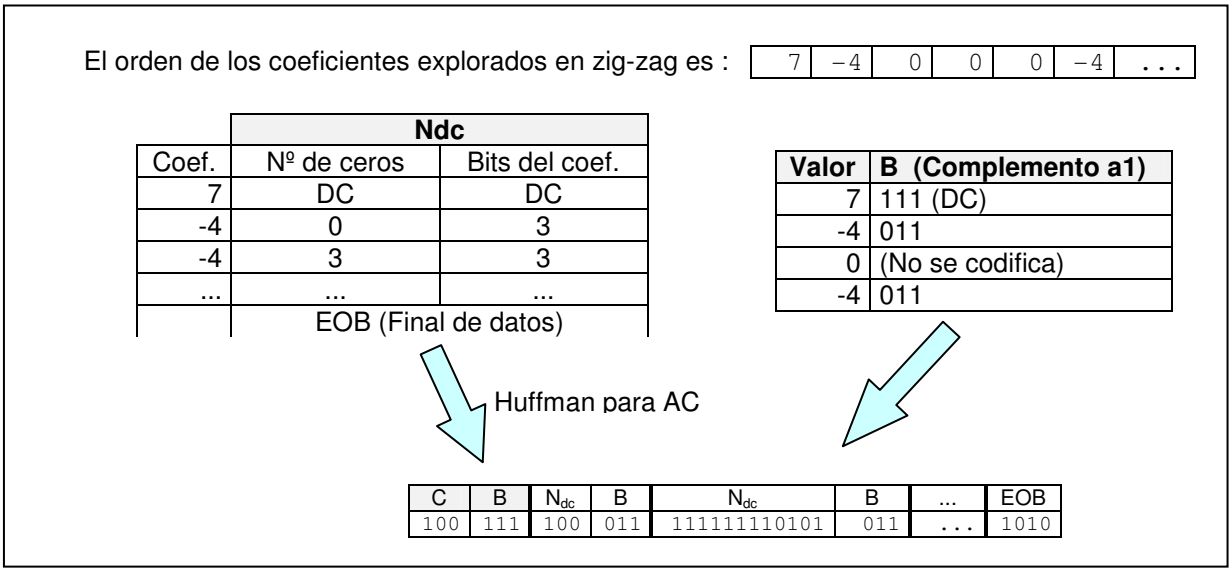

Figura 9

Las tablas de Huffman son personalizables, al igual que la matriz de cuantificación, estos dos parámetros están incluidos en la cabecera de un archivo JPEG.

CABECERA DE UN FICHERO JPEG

En la siguiente tabla describimos la estructura de un fichero JPEG donde se codifica una **imagen de escala de grises**, para imágenes en color varían algunos parámetros.

Todos los archivos JPEG empiezan con FF D8 y acaban con FF D9, dentro de estos limitadores existen diferentes secciones:

Cada celda significa 1 byte.

FF	D8	FF	E0	*LM	*LL	*ID	...	*V1	*V2	*UN	*X1	*X2	*Y1	*Y2	*XT	*YT

-FF D8: Indica el inicio de un archivo JPEG.

-FF E0: Indican el inicio de este bloque.

-*LM: Es el byte más significativo de la longitud de este bloque de datos (incluyendo *LM y *LL).

-*LL: Es el byte menos significativo de la longitud de este bloque de datos (incluyendo *LM y *LL).

-*ID: En nuestro caso serán siempre 5 bytes con estos datos 74,70,73,70,0. Esto es un string terminado en 0 que indica que el archivo tiene formato JFIF.

-*V1,*V2: Estos 2 bytes indican la versión del JFIF con que se trabaja. Pueden ser 1,1 (Versión 1.1).

-*UN: Indica las unidades de X e Y, en nuestro caso las unidades serán pixels, entonces *UN=0.

-*X1,*X2: Indican la densidad de pixels en horizontal, en nuestro caso (0,1).

-*Y1,*Y2: Indican la densidad de pixels en vertical, en nuestro caso (0,1).

*XT: Xthumbnail, en nuestro caso *XT=0.

*YT: Ythumbnail. en nuestro caso *XT=0.

FF	DB	*LM	*LL	*TN	*QD

-FF DB: Indican el inicio de una tabla de cuantificación.

-*LM: Es el byte más significativo de la longitud de la tabla de quantificación + 3.

-*LL: Es el byte menos significativo de la longitud de la tabla de quantificación + 3.

-*TN: Es el número de tabla, como nosotros trabajamos en escala de grises solo trabajaremos con 1 tabla (*TN = 0).

-*QD: Estos son los 64 bytes (si trabajamos en bloques de 8*8) de la matriz de quantificación.

FF	C4	*LM	*LL	*TC	*HC	*HS

-FF C4: Indican el inicio de una tabla de Huffman

-*LM: Es el byte más significativo de la longitud de la tabla simbolos + longitud de la tabla de bits por símbolo + 3.

-*LL: Es el byte más significativo de la longitud de la tabla simbolos + longitud de la tabla de bits por símbolo + 3.

-*TC: Los 4 bits más significativos indican el tipo de tabla(0=DC,1=AC), los 4 menos significativos marcan el número de tabla (empezando por 0).

-*HC: Esta tabla contiene 16 bytes e indica. El primer byte indica el número de palabras del código Huffman de 1 bit, el segundo indica las palabras del código Huffman de 2 bits, etc ...

-*HS: Los simbolos del código Huffmam vienen dados pero lo que podemos variar es a que corresponden. Por ejemplo, si nos fijamos en la figura 6, C no se puede variar, pero lo que podemos hacer es variar a que C corresponde cada N.
Por ejemplo si *HS = 1,0,2,3,4,5,6,7,8,9,10,11. Lo que he hecho es cambiar el código del 1 por el del 0 i viceversa, es decir l 1 = 00, y el 0 = 010.

NOTA: Tenemos que codificar 2 tablas de Huffman la de DC (TC=0x00) y la de AC (TC=0x10).

FF	C0	*LM	*LL	*PR	*HM	*HL	*WM	*WL	*NC	*N0	*S	*QT

-FF C0: Indican el inicio de la sección de parámetros.

-*LM: Es el byte más significativo de la longitud de la sección de parámetros+2.

-*LL: Es el byte menos significativo de la longitud de la sección de parámetros+2.

-*PR: Indican los bits de precisión por pixel de la imagen. En nuestro caso *PR=8.

-*HM: Indica el byte más significativo de la altura de la imagen.

-*HL: Indica el byte menos significativo de la altura de la imagen.

-*WM: Indica el byte más significativo de la anchura de la imagen.

-*WL: Indica el byte menos significativo de la anchura de la imagen.

-*NC: Aquí se codifica el número de componentes. En nuestro caso *NC=1.

-*N0: Indica que el siguiente dato corresponde a la componente 0. En nuestro caso *N0=1.

-*S: Indica el método de sampleo de la componente *N0. Los 4 bits más significativos marcan el sampleo en horizontal, los 4 menos significativos el sampleo en vertical. En nuestro caso *S = 0x11.

-*QT: Indica el número de tabla de cuantificación que se utilizará para la componente *N0. En nuestro caso *QT=0.

FF	DA	*LM	*LL	*NC	*N0	*H0	*FD	*LD	*TC	...	FF	D9

-FF DA: Indican el inicio de la sección de datos.

-*LM: Es el byte más significativo de la longitud de la sección de datos. Incluyendo *LM,*LL, y sin contar los datos de la imagen ni el indicador de final de datos.

-*LL: Es el byte menos significativo de la longitud de la sección de datos. Incluyendo *LM,*LL, y sin contar los datos de la imagen ni el indicador de final de datos.

-*NC: Número de componentes de la imagen, en nuestro caso solo hay una componente *NC=1.

-*N0: Indica que los datos siguientes se corresponden a este número de componente, en nuestro caso solo hay una componente *N0=1.

-*H0: Significa que los siguientes datos están codificados con las tablas de Huffman que aquí se indica.

-*FD: El primer coeficiente de la DCT. (Normalmente 0)

-*LD: Indica el último coeficiente de la DCT. (Normalmente 63)

-*TC: Indica el tipo de codificación, en nuestro caso *TC=0.

-FF D9: Indican el fin de la sección de datos.

ANALISIS DE UN FICHERO JPEG

Para finalizar este articulo generaremos un archivo JPEG con Matlab y luego lo analizaremos mediante un editor hexadecimal, así podremos experimentar con la teoría que hemos explicado en este articulo.

Con el programa de la figura 10 escrito en el Matlab conseguimos generar una imagen en escala de grises de 8*8 pixels, todos de valor 128 (gris medio).

```
%Generamos imagen
imagen_oiginal=ones(8,8) *128

%Mostramos imagen
imshow(imagen_oiginal)

%Guardamos imagen como JPEG
imwrite(imagen_oiginal,'d:\articles\jpg_mat1.jpg','jpg')
```

Figura 10

En la figura 11 vemos el archivo JPEG generado con el Matlab abierto con un editor hexadecimal.

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000000	FF	D8	FF	E0	00	10	4A	46	49	46	00	01	01	00	00	01	ÿØÿà..JFIF.....
00000010	00	01	00	00	FF	DB	00	43	00	08	06	06	07	06	05	08	....ÿÛ.C.....
00000020	07	07	07	09	09	08	0A	0C	14	0D	0C	0B	0B	0C	19	12	.....
00000030	13	0F	14	1D	1A	1F	1E	1D	1A	1C	1C	20	24	2E	27	20	..... $.'
00000040	22	2C	23	1C	1C	28	37	29	2C	30	31	34	34	34	1F	27	",#..(7),01444.'
00000050	39	3D	38	32	3C	2E	33	34	32	FF	C0	00	0B	08	00	08	9=82<.342ÿÀ.....
00000060	00	08	01	01	11	00	FF	C4	00	1F	00	00	01	05	01	01	......ÿÄ.....
00000070	01	01	01	01	00	00	00	00	00	00	00	00	01	02	03	04	.....
00000080	05	06	07	08	09	0A	0B	FF	C4	00	B5	10	00	02	01	03	.....ÿÄ.µ.....
00000090	03	02	04	03	05	05	04	04	00	00	01	7D	01	02	03	00	.....}....
000000A0	04	11	05	12	21	31	41	06	13	51	61	07	22	71	14	32	....!1A..Qa."q.2
000000B0	81	91	A1	08	23	42	B1	C1	15	52	D1	F0	24	33	62	72	•'¡.#B±Á.RÑð$3br
000000C0	82	09	0A	16	17	18	19	1A	25	26	27	28	29	2A	34	35	,.....%&'()*45
000000D0	36	37	38	39	3A	43	44	45	46	47	48	49	4A	53	54	55	6789:CDEFGHIJSTU
000000E0	56	57	58	59	5A	63	64	65	66	67	68	69	6A	73	74	75	VWXYZcdefghijstu
000000F0	76	77	78	79	7A	83	84	85	86	87	88	89	8A	92	93	94	vwxyzƒ„…†‡ˆ‰Š'""
00000100	95	96	97	98	99	9A	A2	A3	A4	A5	A6	A7	A8	A9	AA	B2	•—˜™š¢£¤¥¦§¨©ª²
00000110	B3	B4	B5	B6	B7	B8	B9	BA	C2	C3	C4	C5	C6	C7	C8	C9	³´µ¶·¸¹ºÂÃÄÅÆÇÈÉ
00000120	CA	D2	D3	D4	D5	D6	D7	D8	D9	DA	E1	E2	E3	E4	E5	E6	ÊÒÓÔÕÖ×ØÙÚáâãäåæ
00000130	E7	E8	E9	EA	F1	F2	F3	F4	F5	F6	F7	F8	F9	FA	FF	DA	çèéêñòóôõö÷øùúÿÚ
00000140	00	08	01	01	00	00	3F	00	2B	FF	D9						......?.+ÿÙ

Figura 11

Ahora podemos comprobar que la estructura del archivo se corresponde con lo anteriormente explicado.

Comentaremos los puntos más interesantes. Desde el byte 0x6 al 0xA indica que se trata de un fichero estructura JFIF, y en los bytes 0xB y 0xC no indica que la versión del JFIF es 1.1.

La matriz de cuantificación que ha utilizado Matlab para esta imagen la tenemos desde el byte 0x19 al byte 0x58.

Desde el byte 0x59 hasta el 0x65 tenemos los datos de la imagen que podemos ver que son: Precisión de 8 bits por pixel, una altura y anchura de 8 pixels, una sola componente con un sampleo vertical y horizontal de 1.

A partir del byte 0x66 hasta el byte 0x13C están las tablas de Huffman para DC y AC.

Desde el byte 0x13D hasta el final está la sección de datos, pero de estos datos vemos que **toda la imagen esta codificada en 1 solo byte**, el de la posición 0x148. El valor de este byte es: 0x2B o en binario 00101011.

Si al bloque original (nivel 128) le restamos 128 tenemos una matriz de 0, si hacemos la DCT sobre una matriz de 0 nos da otra matriz de 8x8 con todo 0. Ahora si dividimos coeficiente a coeficiente una matriz de 0 por cualquier otra (cuantificamos) nos da igualmente una matriz de 8x8 llena de 0.

Finalmente codificamos esta matriz, el coeficiente de DC es 0 entonces N=0, según la figura 6 los bits que guardamos en el archivo son 00, como el coeficiente es 0 no hace falta guardarlo, es decir no ponemos B (es el único caso donde no hace falta guardar el valor del coeficiente). Una vez guardado el valor del coeficiente DC ya podemos guardar el valor de todos los AC, como todos son 0 ya podemos indicar final de bloque, es decir guardar EOB = 1010. Y finalmente rellenar los bits que quedan hasta 8 con 1.

Vemos que la forma en que Matlab ha codificado la imagen se corresponde exactamente con lo explicado en el articulo.

Una vez hemos llegado aquí si alguien quiere profundizar en el tema le recomiendo que intente programar en C un compresor JPEG, es más fácil de lo que parece. La parte más pesada son los códigos Huffman para coeficientes AC.

BIBLIOGRAFIA

TARRÉS RUIZ, Francesc. Sistemas audiovisuales. Edicions UPC.

SOLOMON, David. Data compression. Ed. Springer

JPEG File Interchange Format. ftp://ftp.uu.net/graphics/jpeg/jfif.ps.gz

JPEG Codec source code. ftp://ftp.uu.net/graphics/jpeg/jpegsrc.v5.tar.gz

JPEG Data structure. www.geocities.com/tapsemi/datastruct.html

Image compression course.
www-ict.its.tudelft.nl/html/education/courses/et10-38/hc/hc12/sld002.htm