Capítulo 6

Compresión de imágenes

6.1. Introducción

Una imagen digital de tamaño M×N con L tonos de gris requiere para su almacenamiento, en principio, $M \times N \times log(L)$ bits. La compresión de imágenes se ocupa de representar la imagen digital de tal forma que al almacenarla (o transferirla) ocupe menos espacio que con su representación matricial. Se aplica, por tanto, para el almacenamiento y transferencia de imágenes y, además, va a permitir la extracción de características de la imagen.

Un sistema de compresión de imágenes debe de tener las siguientes propiedades:

a) Conseguir una reducción significativa en el número de bits que utiliza para su almacenamiento.

b) Si conlleva una pérdida en la calidad de la imagen, ésta deberá ser poco significativa para el ojo humano o no suponer pérdida de información en las características relevantes de la imagen cuando se utiliza para aplicaciones en visión artificial.

c) Rapidez de cálculo tanto para la compresión como para la descompresión.

d) El formato de salida deberá permitir su almacenamiento y su transferencia.

En primer lugar hay que distinguir los *datos* de la *información*. Los datos son un soporte de la información de manera que varios conjuntos de datos pueden expresar la misma información. Si un conjunto de datos expresa la misma información que otro conjunto más reducido es porque contiene información redundante. La compresión de imágenes trata de reducir la cantidad de datos necesaria para representar una imagen digital y la idea básica del proceso de reducción de datos no es otra que la de eliminar la redundancia en la información. Por tanto, un proceso de compresión de imágenes busca una representación reducida de la imagen, mediante un conjunto de datos, que elimine la información redundante. Vamos a distinguir entre las técnicas de compresión que preservan la información, o **compresión sin pérdida** de información, donde la imagen descomprimida será idéntica a la original y, por tanto, estará libre de error, como así podemos desear en imágenes de texto, huellas dactilares o imágenes médicas, y la **compresión con pérdida** de información, donde la imagen descomprimida no será idéntica a la original y su grado de parecido dependerá de la aplicación que se vaya a realizar (vídeoconferencias, televisión, robótica, etc.).

Las técnicas para la compresión de imágenes las vamos a agrupar en cuatro clases:

a) Compresión estadística, donde la codificación de la imagen se basa en los niveles de gris de la imagen completa.
b) Compresión espacial, donde la codificación se basa en la relación espacial entre los píxeles que presentan valores similares en los niveles de gris.
c) Compresión por cuantificación, que reduce el número de niveles de gris utilizados en la representación o sustituyen cada ventana de $m \times m$ píxeles (m suele ser 3, 5 ó 7, dependiendo de la tasa de compresión que deseemos) por la más parecida entre las ventanas de un conjunto de ventanas prototipo.
d) Compresión fractal, que consiste en encontrar una regla de construcción que produzca una imagen fractal que se aproxime a la imagen original utilizando la teoría matemática de los sistemas iterados de funciones. Un codificador fractal procesa la imagen original seleccionando una función contractiva cuyo único punto fijo (atractor) aproxima la original. El descodificador desarrolla iterativamente la función para recuperar al atractor.

En general, un sistema de compresión de imágenes consta de tres fases u operaciones:

1. Transformación.
2. Cuantificación.
3. Codificación.

Los datos de salida de cada una de estas fases son los datos de entrada de la operación siguiente.

6.2. Transformaciones

Esta fase se utiliza una operación de transformación, es decir, se utiliza una función que transforma el conjunto de datos de la imagen original en otro conjunto de datos, que intenta eliminar toda la información redundante posible, pero sin que haya pérdida de información, con el fin de hacer más eficiente la cuantificación. Por lo tanto, esta fase es reversible, es decir, nos permite obtener la imagen original. Veamos alguna de las transformaciones que suelen usar los formatos clásicos de compresión.

a) Transformaciones lineales

Son las transformaciones más sencillas. Cualquier imagen digital monocroma de tamaño $M \times N$ con L niveles de gris, se puede representar por el vector

$$\left(f(1), f(2),..., f(N).f(N+1),..., f(2N), f(2N+1),..., f(M \times N)\right)$$

donde hemos colocado las filas de la matriz que representa a la imagen una a continuación de la otra. Para codificar cada componente de dicho vector necesitamos $log(L)$ bits (para L=256 necesitamos 8 bits).

Una transformación lineal viene dada por la expresión:

$$\begin{pmatrix} g(1) \\ g(2) \\ \dots \\ g(M\times N) \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1,M\times N} \\ a_{21} & a_{22} & \dots & a_{2,M\times N} \\ \dots & \dots & \dots & \dots \\ a_{M\times N,1} & a_{M\times N,2} & \dots & a_{M\times N,M\times N} \end{pmatrix} \begin{pmatrix} f(1) \\ f(2) \\ \dots \\ f(M\times N) \end{pmatrix}$$

Hay que elegir los coeficientes a_{ij} de la matriz de transformación de forma que los nuevos valores $g(n)$, $n=1,2,...,M\times N$, sean mucho menores que L y así necesitaremos menos bits para almacenarlos. Al mismo tiempo, dicha transformación debe ser reversible (invertible), es decir, el determinante de la matriz $((a_{ij}))$ debe ser diferente de cero, de forma que podamos obtener los valores $f(n)$ a partir de los valores $g(n)$, $n = 1, 2,\dots, M\times N$.

Una operación muy usada es la transformación lineal *basada en las diferencias* que viene dada por la siguiente matriz:

$$\begin{pmatrix} 1 & 0 & 0 & \dots & 0 & 0 & 0 \\ 1 & -1 & 0 & \dots & 0 & 0 & 0 \\ 0 & 1 & -1 & \dots & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{pmatrix}$$

Con esta transformación se tiene que $g(1) = f(1)$ y $g(n) = f(n-1)-f(n)$. Cabe esperar que los nuevos valores que vienen dados por la diferencia entre los niveles de gris de dos píxeles consecutivos sean menores que los originales. Además, dicha transformación es reversible.

b) <u>Transformaciones basadas en la longitud de la racha</u>

Consideremos una imagen digital representada en forma vectorial,

$$\left(f(1), f(2),..., f(N).f(N+1),..., f(2N), f(2N+1),..., f(M\times N)\right)$$

Una **racha** es una secuencia de píxeles consecutivos con el mismo tono de gris, precedida por un píxel con diferente tono de gris y seguida también por un píxel de diferente tono de gris.

Mediante esta operación se transforma el vector imagen

$$\left(f(1), f(2),..., f(N).f(N+1),..., f(2N), f(2N+1),..., f(M\times N)\right)$$

en la secuencia de pares

$$\left((g_1,l_1),(g_2,l_2),...,(g_k,l_k)\right)$$

donde $g_1=f(1)$ y l_1 es la longitud de la primera racha que se forma; g_2 es el nivel de gris del píxel que sigue a dicha racha y l_2 es la longitud de la nueva racha, y así sucesivamente. Sólo necesitamos almacenar el nivel de gris de cada racha y la longitud de la misma para poder reconstruir la imagen original. En algunas imágenes se consigue una reducción sustancial del número de bits con este método.

c) *La transformación coseno discreta*

La ventaja que presenta esta transformación frente a la transformada de Fourier discreta es que no necesita utilizar números complejos. La transformada coseno discreta de la imagen digital {f(m,n), m=0,1,...,M-1, n=0,1,2,...,N-1} viene dada por la expresión:

$$F(u,v)=\frac{2}{\sqrt{MN}}C(u)C(v)\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}f(m,n)\cos\left[\frac{(2m+1)u\pi}{2M}\right]\cos\left[\frac{(2n+1)v\pi}{2N}\right]$$

donde

$$C(z)=\begin{cases}\frac{1}{\sqrt{2}} & \text{si } z=0\\ 1 & \text{si } z>0\end{cases}$$

y la transformación inversa viene dada por la expresión:

$$f(u,v)=\frac{2}{\sqrt{MN}}C(u)C(v)\sum_{u=0}^{M-1}\sum_{v=0}^{N-1}F(u,v)\cos\left[\frac{(2m+1)u\pi}{2M}\right]\cos\left[\frac{(2n+1)v\pi}{2N}\right]$$

Obsérvese que la imagen se ha descompuesto en un conjunto de funciones de base que permiten reconstruir la imagen a partir de una suma ponderada de las mismas.

La cuantificación ya la estudiamos en un capítulo anterior y la codificación se ha estudiado en otras asignaturas.

6.3. La Modulación Delta (DM)

En muchos métodos de compresión no se saca provecho de la correlación entre los niveles de gris de los píxeles. Con la Modulación Delta se tiene en cuenta la diferencia entre los niveles de gris de los píxeles consecutivos y esta se cuantifica con sólo dos niveles de reconstrucción (que se codifican con un sólo bit).

Supongamos que la imagen digital viene representada en forma vectorial,

$$\left(f(1), f(2),..., f(N).f(N+1),..., f(2N), f(2N+1),..., f(M\times N)\right)$$

y sea $f^*(n)$ la imagen reconstruida. La Modulación Delta utiliza la transformación,

$$e(n) = f(n) - f^*(n-1), \qquad n=1,2,...,M\times N.$$

donde $f^*(1) = f(1)$.

Los valores $e(n)$ se cuantifican según la expresión:

$$e^*(n) = \begin{cases} \Delta & \text{si} \quad e(n)\geq 0 \\ -\Delta & \text{si} \quad e(n)<0 \end{cases}$$

y los valores de reconstrucción vienen dados por:

$$f^*(n) = f^*(n-1) + e^*(n), \qquad n = 1, 2,..., M\times N.$$

El tamaño del paso, Δ, juega un papel determinante a la hora de conseguir una compresión de buena calidad. En las zonas de la imagen donde los tonos de gris varían poco la imagen reconstruida puede ir variando mucho con respecto a la imagen original produciendo lo que se llama un *ruido granular*. En estos casos sería conveniente haber elegido un paso Δ pequeño. Por otro lado, en las zonas de la imagen donde se producen grandes variaciones en lo niveles de gris, si elegimos un paso Δ pequeño entonces la imagen reconstruida presentará un reducción de contraste (bordes menos perfilados) y será más borrosa (suave). En este caso decimos que se ha producido *distorsión por sobrecarga de pendiente* y deberíamos haber elegido un paso Δ mayor (ver la figura 1). Tanto el ruido granular como la distorsión por sobrecarga de pendiente son efectos contrapuestos y dependen del paso elegido. Cuando Δ es el 5% del rango dinámico de la imagen, L, se reduce el ruido granular pero la imagen es borrosa. Cuando Δ es el 15% del rango dinámico, aparece más granularidad en las zonas más homogéneas pero los contornos están más perfilados y la imagen es menos borrosa. La única forma de reducir simultáneamente dichos inconvenientes es eligiendo 4 niveles de reconstrucción, en lugar de dos, pero ello va a suponer una menor tasa de compresión.

La imagen se reconstruye a partir del valor $f(1)$ y se le va sumando Δ o -Δ a los valores que se van reconstruyendo iterativamente,

$$f^*(1) = f(1), \quad f^*(2) = f^*(1) + e^*(2), \quad f^*(3) = f^*(2) + e^*(3),.....$$

Por ello, el transmisor sólo debe emitir los valores $f(1)$, $e^*(n)$, $n=2,3,...,M\times N$, y el receptor reconstruye la imagen añadiendo $e^*(n)$ a $f^*(n-1)$. Como $e^*(n)$ vale Δ o $-\Delta$ sólo necesitamos 1 bit para codificar esta información y $log(L)$ bits para codificar $f(1)$. Por ejemplo, para $L=256$, cada píxel de la imagen requiere 8 bits, mientras que para la imagen comprimida son necesitamos un bit por píxel, salvo para el primer píxel que necesitamos también 8 bits. Por lo tanto se obtiene una tasa de compresión aproximada de 8:1 (ocho a uno).

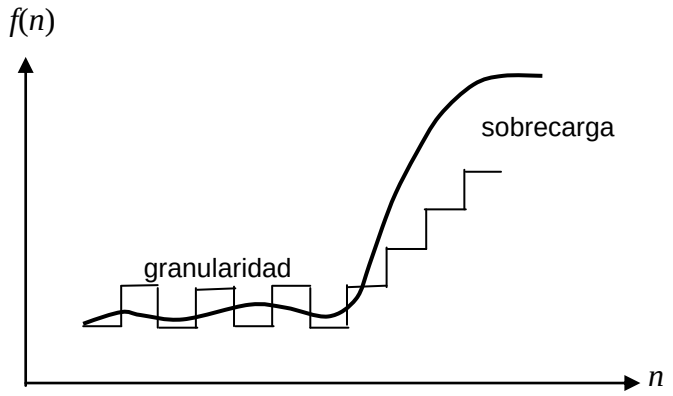

Figura 1. Imagen con ruido granular y distorsión por sobrecarga de pendiente.

6.4 El formato JPEG

JPEG es una familia de técnicas de compresión (29 procesos de codificación diferentes) estandarizadas por el grupo *Joint Photographic Experts Group*. La palabra inglesa "joint" (grupo conjunto) se debe a la cooperación conjunta realizada por ISO (International Organization for Standardization) y CCITT (International Telegraph and Telephone Consultative Committee, que condujo al primer formato estándar internacional para la compresión de imágenes digitales. La compresión JPEG tiene en cuenta una limitación del sistema visual humano, y es que el ojo humano puede percibir mejor pequeños cambios en el brillo de una imagen que pequeños cambios en el color. Por ello, la compresión JPEG perderá alguna información sobre el color. Con esta técnica se pueden conseguir tasas de compresión de 20 a 1 sin que apenas se note.

Se puede controlar la tasa de compresión de una imagen especificando el valor de un parámetro Q; cuando se elige un valor de Q grande la imagen tiene una mayor calidad y ocupa mayor espacio que cuando se elige un valor pequeño. Conforme la calidad de la imagen empieza a degradarse irá apareciendo una estructura de bloques

que degenera en un conjunto de cuadrados cuyo nivel de gris corresponde al valor medio de los píxeles que forman cada cuadrado.

La técnica JPEG básica (baseline JPEG) consta de cinco pasos:

1. Transformación de la imagen RGB a una imagen en el espacio de colores YC_rC_b.
2. Reducción de las componentes de color (opcional).
3. Partición de la imagen en bloques (ventanas) de 8×8 píxeles y determinación de la transformada del coseno discreta (TCD) para cada bloque.
4. Cuantificación de los coeficientes de la TCD.
5. Codificación sin pérdidas de los coeficientes reducidos utilizando el algoritmo de Huffman modificado.

En el primer paso se realiza la transformación de la imagen RGB al espacio de representación del color YC_rC_b (no es necesaria si la imagen es monocromática). Una cámara de vídeo recibe la luz de una escena y la divide en sus componentes roja, verde y azul. La intensidad luminosa (luminancia o brillo) Y viene dada por la expresión:

$$Y = 0.2999 \times Rojo + 0.587 \times Verde + 0.114 \times Azul$$

A partir de la luminancia se puede obtener dos señales o valores que contienen la información del color (cromática); el valor (R−Y) es un matiz naranja-cian que es muy importante para crear el color del cutis, y el valor (B−Y) que es un verde-magenta. Estos tres valores, Y, (R−Y) y (B−Y), forman la representación básica de colores de todas las televisiones actuales. Esta representación se consigue, a partir de la representación RGB, mediante la transformación:

$$\begin{pmatrix} Y \\ (R-Y) \\ (B-Y) \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.117 \\ 0.701 & -0.587 & -0.114 \\ -0.299 & -0.587 & 0.886 \end{pmatrix} \begin{pmatrix} Rojo \\ Verde \\ Azul \end{pmatrix}$$

donde los valores Rojo, Verde y Azul, están dentro del intervalo [0,1], y, como se puede comprobar, la luminancia también. Sin embargo, los valores de (R−Y) están en el intervalo [-0.701, 0.701] y los valores de (B−Y) están en el intervalo [-0.886, 0.886]. Pero estos valores no son convenientes cuando perseguimos una representación binaria. Por ello, es preciso normalizarlos en el intervalo [−0.5, 0.5] con la siguiente transformación:

$$C_R = 0.714\ (R–Y)$$

$$C_B = 0.564\ (B–Y)$$

Finalmente, los valores de esta representación de colores se pueden convertir en una codificación binaria mediante las expresiones:

$$Y = redondear\ (219 \times Y + 16)$$

$$C_R = redondear\ (224 \times 0.713 \times (R–Y) + 128)$$

$$C_B = redondear\ (224 \times 0.564 \times (B–Y) + 128)$$

Esta representación tiene dos grandes ventajas sobre la representación RGB.

a) El valor de luminancia, Y, es compatible con las televisiones en blanco y negro.

b) Como el ojo humano es más sensible a pequeños cambios en la luminancia que en la saturación o el matiz de los colores, y percibe los detalles más finos (que no se pueden distinguir con la información del color) por su intensidad luminosa, entonces los valores cromáticos, (R−Y) y (B−Y), se pueden codificar a niveles más pequeños de resolución y precisión, manteniéndose un nivel razonable de calidad en la imagen.

El segundo paso es opcional y conlleva pérdida de información. Se deja la componente Y como está y se muestrean las componentes de color (hay diferentes esquemas de muestreo). Una forma de hacer el muestreo consiste en dividir la imagen en bloques de 4 píxeles y se extrae uno o se promedian los cuatro píxeles de cada bloque.

El tercer paso consiste en dividir las componentes de la imagen en bloques (ventanas) 8×8 (si lo valores de los píxeles varían de 0 a 255 se les resta 128 para conseguir valores enteros positivos y negativos) a los que se les aplica la transformada discreta del coseno. El elemento (0,0) de la TDC nos da el valor medio de los 64 píxeles del bloque y lo representamos por *DC*, el resto de elementos por AC_{ij} siendo (*i*,*j*) la posición del píxel en el bloque.

En el cuarto paso se cuantifican los coeficientes de la TDC de cada bloque. Para ello, dichos coeficientes se dividen por su correspondiente *coeficiente de cuantificación* y se redondean al valor entero más próximo. Este paso reduce muchos elementos a cero favoreciendo la compresión. El valor de Q determina los coeficientes de cuantificación. Así, se dispone de varias tablas de coeficientes de cuantificación. Una tabla de coeficientes de cuantificación para la luminancia es:

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Una tabla de coeficientes de cuantificación para las componentes cromáticas es:

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

En el quinto paso se realiza una codificación sin pérdidas. Los valores *DC* se codifican como por diferencia con el valor del bloque previo (adyacente). Esto se hace porque dichos valores están muy correlacionados. Los demás valores (AC_{ij}) se colocan en fila siguiendo un orden en zigzag (AC_{00}, AC_{01}, AC_{10}, AC_{20}, AC_{11}, AC_{02}, AC_{03}, AC_{12}, AC_{21}, AC_{30}, AC_{40},....., AC_{77}) para conseguir que los términos de bajas frecuencias estén juntos (figura 2).

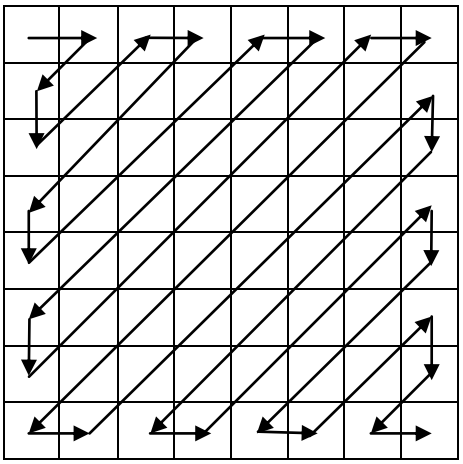

Figura 2. Secuencia de los valores AC.

Es más probable que los términos de bajas frecuencias sean diferentes de cero. Muchos de los coeficientes de alta frecuencia serán nulos y se codificarán fácilmente mediante el **código de longitud de la racha**. Los coeficientes no nulos y las longitudes de las rachas se codifican usando una **codificación de Huffman** o una **codificación aritmética**. Cada

término en la codificación basada en la longitud de la racha consta de tres valores: una LONGITUD DE RACHA que nos da el número de ceros que preceden al término, un TAMAÑO que nos da el número de bits utilizados para representar el valor del término (ver la Tabla 1) y un VALOR DEL DATO que es el valor (número) que representa dicho término.

TAMAÑO	Rango de Valores
1	–1,1
2	–3, –2, 2, 3
3	–7, –6, –5, –4, 4, 5, 6, 7
4	–15, –14,….., –7, –8, 8, 7,.....,14, 15

Tabla 1. Correlación entre el TAMAÑO y el Rango de Valores Posibles.

El VALOR DEL DATO se codifica solamente con el número de bits requeridos para su representación binaria y se almacena utilizando sólo los bits de orden inferior (tantos como especifica el TAMAÑO) en su representación binaria de complemento a 1. Por ejemplo, en una representación con 8 bits, el valor 6 tiene como representación binaria de complemento a 1,

00000110

y su codificación del VALOR DEL DATO serían los tres últimos dígitos (como especifica el TAMAÑO), es decir,

110.

Para un valor negativo, como, por ejemplo –6, su representación binaria de complemento a 1, es

11111001

y se codificaría como 001.

Obsérvese que en la representación binaria de complemento a l el dígito de orden superior especifica el signo del número (1 los negativos y cero los positivos). Asimismo, en la representación reducida de tres dígitos, el dígito de orden superior especifica también el signo (ahora 1 indica que es positivo y cero negativo). La secuencia de bits utilizada para representar los 8 valores de TAMAÑO 3 son:

–7	–6	–5	–4	4	5	6	7
000	001	010	011	100	101	110	111

Finalmente, a la secuencia de valores código que se obtiene se codifica mediante el algoritmo de Huffman. Ello está motivado porque los valores de LONGITUD DE LA RACHA y del TAMAÑO tienden a repetirse con más frecuencia unos pares que otros (los VALORES DE LOS DATOS, sin embargo, suelen ser aleatorios).

PEG es un algoritmo simétrico pues la descompresión la hace con el mismo número de operaciones pero en sentido inverso.

6.4. Criterios de fidelidad de la imagen

Se trata ahora de comparar la imagen original con la imagen comprimida para ver la degradación o distorsión de la imagen comprimida con respecto a la original. Vamos a estudiar dos criterios, uno basado en la tasa señal-ruido y el otro basado en el error cuadrático medio.

Si *f(m,n)* representa la imagen original y *f*(m,n)* la imagen reconstruida, el error de reconstrucción para el píxel (*m,n*) viene dado por

$$e(m,n) = f(m,n) - f^*(m,n)$$

y el *error cuadrático medio de reconstrucción* para toda la imagen viene dado por la expresión:

$$e^2 = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left[f(m,n) - f^*(m,n)\right]^2$$

La raíz cuadrada de la expresión anterior suele ser más utilizada pues viene expresada en las mismas unidades que *f*(*m,n*) (niveles de gris).

La *tasa de señal-ruido* viene dada por la expresión:

$$SNR = \frac{\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left[f^*(m,n)\right]^2}{\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left[f(m,n) - f^*(m,n)\right]^2}$$