Compresión de Imagen

Marcos Martín

4 de mayo de 2004

1 Introducción

Una imagen no variable con el tiempo (ya sea una fotografía, un fotograma...) se puede definir como una función de $\Re^2$ en $\Re^3$, si la imagen es en color o de $\Re^2$ en $\Re$ si es con niveles de gris. En ambos casos las dos variables independientes representan el espacio bidimensional. Nos centraremos en las imágenes con niveles de gris, ya que de este tipo son las imágenes médicas que analizaremos, pero la ampliación a tres variables para la representación del color, como en los modelos RGB, YIQ o HSI, no es excesivamente complicada.

Como ocurre para cualquier tipo de función, una digitalización supone una discretización de los valores tanto de las variables independientes como de las dependientes. De esta forma podríamos decir que una imagen digital es una función discreta de dos variables discretas de la forma:

$$y[n_1, n_2] \;\; n_1 = 1, ..., M; \;\; n_2 = 1, ..., N \tag{1}$$

donde n_1 y n_2 son las variables dimensionales discretas, $M \times N$ es el tamaño de la imagen e y es el valor de la intensidad en cada punto. A cada punto de la imagen dado por las coordenadas (n_1, n_2) se le denomina pixel.

La intensidad, por el carácter digital de la imagen, no puede tomar cualquier valor sino que está limitado a un rango discreto. Esto podría alejar una imagen digitalizada de su original continua, pero hay que tener en cuenta el funcionamiento del ojo humano. La visión humana no puede percibir cualquier variación en la intensidad ΔI sino que es necesario que dicha variación supere un mínimo para que pueda ser distinguida la intensidad I de $I + \Delta I$. De hecho, la variación mínima perceptible es proporcionalmente constante ($\Delta I / I = Cte$), es decir, que el ojo tiene una respuesta logarítmica. A esto hay que añadir que existe una pérdida de sensibilidad a la variación en los extremos, con lo que no es infinito el intervalo de intensidades que se pueden percibir. El efecto conjunto de ambas características conlleva que el ser humano sólo puede percibir un número limitado de niveles de intensidad o niveles de gris. En diversos estudios realizados con múltiples observadores tipo se concluyó que el ser humano puede distinguir una media de 100 niveles logarítmicos de intensidad. De esta forma, nos bastaría con que el conjunto de valores posibles de la intensidad discreta tuviera 100 componentes para que el ser humano no percibiera diferencia entre la imagen original y la digitalizada. Dado que se trabaja con bits,

serían suficientes 7 bits ($2^7 = 128$ niveles) por pixel para evitar pérdidas perceptibles en el proceso de digitalización. Sin embargo, normalmente se utilizan 8 bits ($2^8 = 256$ niveles) por pixel, dado que es el número de bits con el que se trabaja normalmente (8 bits = 1 byte); además, se ofrece un margen para que los posibles procesos a los que se sometan las imágenes no repercutan en una pérdida de fidelidad con respecto al original. Debemos tener en cuenta que todo lo expuesto es válido si la imagen tiene como destinataria la visión humana. Para el análisis computerizado sería necesario un número considerablemente mayor de bits por pixel. Sin embargo, en nuestro caso es bastante asumible que la imágenes médicas serán analizadas por el ojo humano, de hecho sus destinatarios serán los especialistas en la materia. En las imágenes médicas en general los distintos niveles de gris que estudiará el especialista representan alguna propiedad química o física de la estructura objeto del análisis. Por ejemplo en una radiografía de rayos-X digitalizada el valor del nivel de gris de cada punto expresa la densidad óptica del área correspondiente; en una tomografía el valor del nivel de gris está relacionado con el coeficiente de atenuación lineal de tejido, etc.

Hasta ahora hemos hablado del carácter discreto de la percepción humana en lo que se refiere a niveles de intensidad. Sin embargo, el ojo humano también realiza un procesado en frecuencia, de tal forma que lo que se percibe en un punto depende de la percibido en puntos vecinos. Mediante diversos experimentos se ha llegado a la conclusión de que el ojo realiza un filtrado paso bajo, manteniendo la componente continua. De este hecho se deducen dos conclusiones:

- Existe una frecuencia espacial máxima por encima de la cual no se perciben las variaciones, sino que se uniformiza lo observado. Será esta característica la que nos permita discretizar el espacio en pixels sin que el ojo humano detecte el salto entre puntos consecutivos, siempre que los pixels sean lo suficientemente pequeños y estén lo suficientemente juntos.

- Dentro del margen de frecuencias espaciales perceptibles no podemos despreciar las altas frecuencias, pues el ojo humano es muy sensible a los bordes, dado que los potencia.

Por último, debemos reseñar que una imagen digital con niveles de gris, dado su carácter discreto, puede definirse también como una matriz de la forma:

$$A_{MN} = \begin{pmatrix} a_{11} & a_{12} & ... & a_{1N} \\ a_{21} & a_{22} & ... & a_{2N} \\ ... & ... & ... & ... \\ a_{M1} & a_{M2} & ... & a_{MN} \end{pmatrix} \tag{2}$$

Las dimensiones de la matriz ($M \times N$) es el tamaño de la imagen en pixels y el valor en cada punto (a_{ij} $i = 1, 2, ..., M$; $j = 1, 2, ..., N$) es la intensidad del pixel especificado dentro de un rango discreto, tal como explicamos. Por tanto, una imagen digital tiene un tamaño en bits igual a $M \times N \times p$ bits donde 2^p es el número de niveles de gris que tiene la imagen (p es el número de bits por pixel).

Figura 1: Ejemplo de imagen digitalizada sin pérdidas visuales.

A continuación vamos a ver unos ejemplos de imágenes digitales y podremos constatar algunos de los puntos expuestos anteriormente. Una imagen digital que de cara al observador no ha sufrido pérdidas con respecto a la original continua se muestra en la figura 1. En esta imagen, para especificar el valor de la intensidad en cada punto, se utilizan 8 bits. Comprobemos lo que ocurre si lo reducimos a 3 bits, de tal forma que el número de niveles de cuantificación es menor que los escalones de intensidad perceptibles. El resultado se observa en la figura 2. Vemos que se hace patente la transición entre los distintos niveles de gris (de entre los 8 posibles), mientras que en la anterior dicha transición era, al menos para nuestros ojos, continua. La figura 3 es un pequeña parte de la figura 1 aumentada de tamaño. Ahora volvemos a tener 8 bits por pixel pero el tamaño de los pixels es tal que la visión humana percibe las transiciones espaciales, la frecuencia de muestreo espacial es inferior a la necesaria. Podemos observar perfectamente los bordes de cada pequeño cuadrado de intensidad distinta, dado que dichos cuadrados son lo suficientemente grandes para ser percibidos.

2 Definiciones y Clasificación

El objetivo primordial de la compresión de imágenes digitales es la reducción del número de bits que requieren dichas imágenes con la menor pérdida de calidad posible. Aunque la capacidad de los soportes para almacenamiento

Figura 2: Ejemplo de imagen digitalizada con 8 niveles de cuantificación.

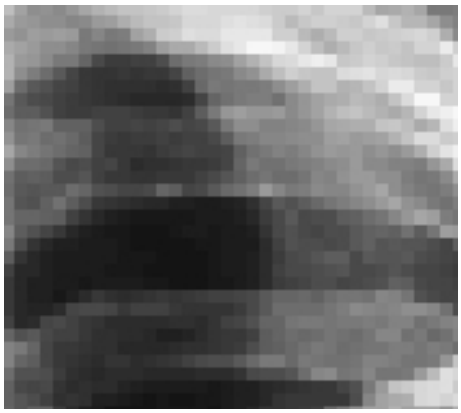

Figura 3: Ejemplo de imagen digitalizada con frecuencia espacial baja.

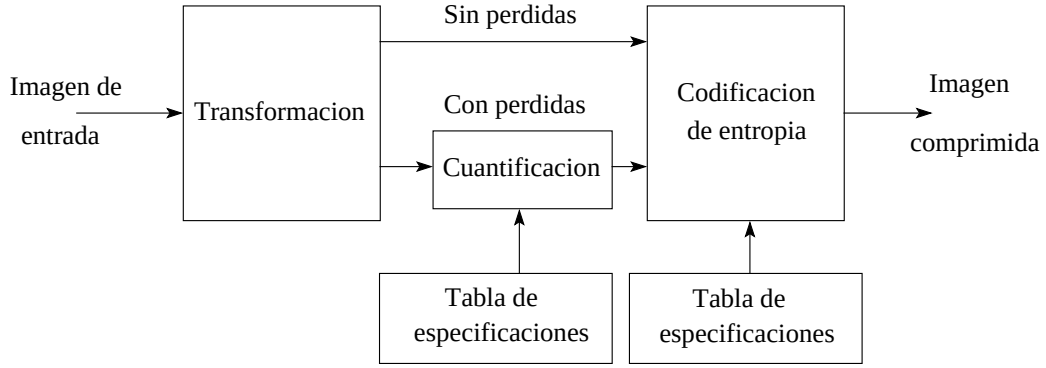

Figura 4: Diagrama general de un compresor de imágenes.

en el mercado aumenta considerablemente día a día, hay que tener en cuenta las enormes necesidades que implica un número elevado de imágenes de gran tamaño, como es el caso de los historiales radiológicos que maneja cualquier hospital. Por otro lado la compresión cobra considerable importancia en las aplicaciones que requieren la transmisión de imágenes por un medio que siempre tendrá un ancho de banda limitado, ya sea por naturaleza o por precio.

El diagrama de bloques de un compresor tipo se muestra en la figura 4. Las técnicas de compresión de imágenes, como muchas técnicas de compresión de señales digitales, constan de tres etapas más o menos definidas:

- Transformación de la imagen. Supone una decorrelación de la señal, una reducción de su rango dinámico que elimine información redundante. Los coeficientes transformados deben ser estadísticamente independientes y la energía de la imagen transformada debe compactarse en un número mínimo de ellos.

- Cuantificación. Consiste en reducir la precisión de los coeficientes transformados sin sobrepasar los límites de calidad marcados. Esta fase implica siempre algún tipo de pérdida, con lo que las técnicas sin pérdidas carecerán de ella.

- Codificación de entropía. Aumenta la compresión sin aumentar las pérdidas codificando los coeficientes transformados de forma más compacta. Tras esta codificación tendremos una ristra de bits sin separación distinguible en vez de una serie de coeficientes o símbolos.

Existen diversas formas de clasificar las técnicas de compresión de imágenes digitales, según la característica específica que se quiera diferenciar. Podemos clasificar los métodos de compresión en función de la naturaleza de la transformación que se lleva a cabo. En este caso tenemos tres grandes grupos:

- Codificadores de forma de onda. La codificación se realiza directamente sobre la intensidad de la imagen como función espacial de n_1 y n_2, número de columna y número de fila. Normalmente se lleva a cabo algún tipo de procesado antes de la fase de cuantificación, de tal forma que se elimine

parte de la información redundante, aunque no puede considerarse un verdadero cambio a un espacio transformado. Incluso se puede eliminar este procesado (como por ejemplo en el caso de la codificación PCM o de la cuantificación vectorial, que veremos más adelante).

- Codificadores de transformada. Sobre la imagen se realiza algún tipo de transformación para tratar de eliminar la dependencia estadística entre los pixels consecutivos antes de llevar a cabo la cuantificación, normalmente escalar. Por tanto, lo que se codifica no es la imagen en sí sino sus coeficientes en el espacio transformado. La transformación es equivalente a un cambio de la base en la que se expresa la imagen (o uno de los bloques de tamaño $P \times Q$ en los que se divide). Una imagen puede definirse como una matriz $M \times N$, pero también puede entenderse como un vector de dimensión MN. Si modificamos la base del espacio de forma conveniente obtendremos que la mayoría de la energía se concentra en un número pequeño de componentes, que deberemos cuantificar con mayor resolución. Por ejemplo, si la transformación es a un espacio de frecuencias espaciales, observaremos que la mayor parte de la energía de la señal transformada se concentra en las bajas frecuencias.

- Codificadores basados en modelos. Estas técnicas tratan de modelar la generación de la imagen de tal forma que lo que se codifica son los parámetros del modelo. El proceso comienza con una segmentación de la imagen en distintas zonas según el color, la textura, etc., dado que a cada una de ellas se le aplicará un modelo matemático distinto. El siguiente paso será la extracción de las características de cada una de las zonas para obtener el valor específico para la imagen de entrada de los parámetros variables de cada modelo. Son estos parámetros los que se codificarán, de tal forma que el decodificador a partir de ellos debe ser capaz de sintetizar cada región. También es necesario almacenar la segmentación que se realizó sobre la imagen para determinar las áreas en las que el sintetizador aplicará cada modelo. Estas técnicas logran unas tasas de compresión muy altas, aunque la calidad se resiente considerablemente.

Otra clasificación más general divide las técnicas de compresión en dos tipos:

- Compresión sin pérdidas. Son aquellas técnicas en las que la imagen original puede recuperarse de su versión codificada sin ningún tipo de alteración. Se trata de compresiones reversibles y en consecuencia no incluyen una fase de cuantificación, pues este proceso es intrínsecamente con pérdidas. Sin embargo, una codificación directa de las imágenes con algún tipo de codificador de entropía no supone una tasa de compresión relevante. Por ello es necesaria una procesado previo que realice una cierta decorrelación.

- Compresión con pérdidas. En el proceso de compresión se realizan modificaciones irreversibles de tal forma que existen diferencias entre la imagen original y la imagen que se obtiene tras la decodificación. Esta

modificación irreversible es básicamente una cuantificación de algún tipo. Dentro de este grupo es importante diferenciar las técnicas denominadas visualmente sin pérdidas. En este tipo de compresión, la imagen decodificada es distinta a la original pero las pérdidas sufridas no son detectables por el ojo humano; tienen por tanto en consideración las características de la visión humana a la hora de decidir que alteraciones puede sufrir la imagen durante el proceso de compresión.

3 Cuantificación

En este apartado trataremos de los método de compresión basados en la cuantificación. Como dijimos al principio de esta sección, la cuantificación es una operación que suele formar parte del proceso de compresión. Sin embargo, existen técnicas que reducen e incluso eliminan el procesado anterior a la cuantificación y basan la compresión en esta fase. La decorrelación de los pixels, si se lleva a cabo, se realizará por tanto como consecuencia de las características de la cuantificación.

Existen principalmente tres tipos de cuantificación, la cuantificación escalar, la cuantificación multiescalar y la cuantificación vectorial. Los dos primeros tipos se suelen utilizar como paso posterior a la decorrelación mientras que la cuantificación vectorial suele ser por sí sola un método de compresión. A continuación veremos cada tipo con más detalle.

3.1 Cuantificación Escalar

La cuantificación consiste en la transformación de un conjunto de valores de entrada, que puede ser finito o infinito y discreto o continuo, en otro conjunto menor, siempre discreto y finito. En el caso que nos ocupa el conjunto de valores de entrada serán directamente los niveles de gris de cada pixel o los coeficientes obtenidos por cualquier transformación. Por tanto, los valores de entrada se moverán dentro de un rango. Es ese rango el que dividimos en intervalos, a cada uno de los cuales asignamos un valor representativo, de tal forma que a cualquier valor de entrada perteneciente a un cierto intervalo i se le aproximará por el valor representativo de dicho intervalo. Cada intervalo estará limitado por unos umbrales de cuantificación. El número de intervalos vendrá condicionado por las necesidades de compresión y por el nivel de pérdida fijado aunque siempre tomará la forma $L = 2^n$, donde n es el número de bits asignado a cada valor de salida del cuantificador.

El valor representativo o nivel de cuantificación se escoge de manera que minimice la distorsión para cada intervalo por lo que dependerá de la medida de la distorsión utilizada. La forma de medición más común es la distorsión cuadrática, es decir:

$$d(y, \hat{y}) = (y - \hat{y})^2 \tag{3}$$

donde y es el valor original e $\hat{y}$ es el valor cuantificado, el valor representativo del intervalo al que pertenece y. Utilizando la distorsión cuadrática, el valor representativo óptimo es el punto medio del intervalo.

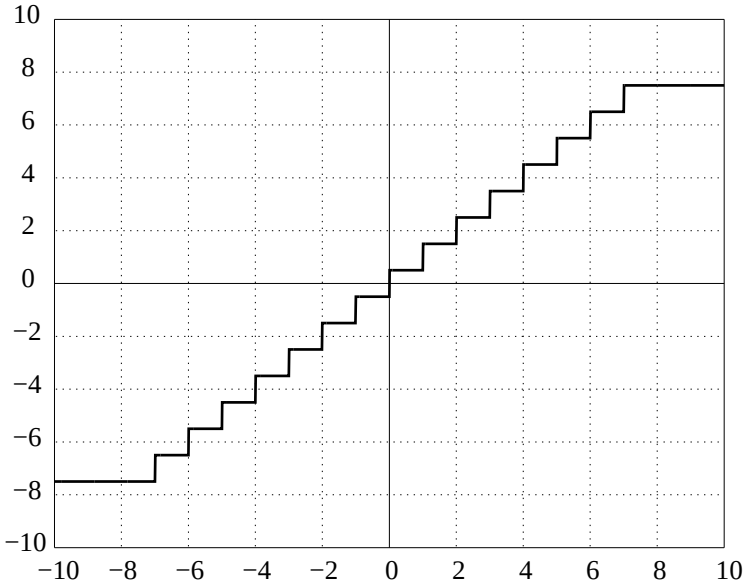

Figura 5: Cuantificador uniforme con 16 niveles.

Existe una clasificación de los cuantificadores escalares en función de la naturaleza de los escalones de cuantificación. En este sentido, el más sencillo es el cuantificador uniforme. Es aquel en el que el rango de variación es dividido en intervalos iguales y el nivel de cuantificación toma el valor del punto medio de cada uno de ellos. En la figura 5 vemos la característica entrada/salida de un cuantificador uniforme para $L = 16$ niveles. La relación señal a ruido (SNR) de un cuantificador uniforme viene dado por:

$$\text{SNR} = 6n + K \text{ dB} \tag{4}$$

siendo $2^n = L$ el número de niveles de cuantificación y K una constante que depende de las características de la señal de entrada.

El cuantificador uniforme será óptimo si la función de densidad de probabilidades del conjunto de posibles valores de entrada es uniforme. En caso contrario, la optimización se consigue con un cuantificador no uniforme, que consta de una transformación (tras la cual los valores de entrada adquirirán una función de densidad de probabilidades uniforme), una cuantificación uniforme y la necesaria transformación inversa. Este proceso se denomina compansión. El efecto conjunto es equivalente a asignar más niveles de cuantificación en la zona del rango de variación en la que la función de entrada toma valores con más frecuencia. Centrándonos en el caso que nos ocupa, el ojo humano tiene una respuesta logarítmica, es sensible a los saltos del logaritmo de la intensidad lumínica, con lo que será menos dañina perceptualmente una cuantificación logarítmica que una cuantificación uniforme.

3.2 Cuantificación Multiescalar

Este tipo de cuantificación consiste básicamente en la agrupación de múltiples cuantificadores escalares de características distintas dentro de un mismo esquema de codificación. Por tanto, está concebido para ser utilizado tras una

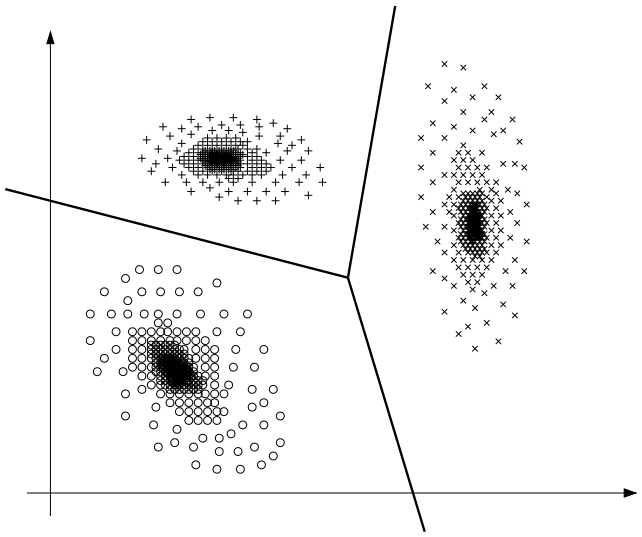

Figura 6: Ejemplo de cuantificación vectorial.

fase de transformación. Los coeficientes calculados pueden tener rangos y características probabilísticas muy distintas en función de su frecuencia o pseudo-frecuencia, por lo que es conveniente aplicar una cuantificación distinta a cada conjunto de coeficientes con características similares para una optimización del proceso.

3.3 Cuantificación Vectorial

La cuantificación vectorial consiste básicamente en agrupar k muestras de la señal de entrada (para ello habrá que dividir o segmentar la imagen original en subbloques), formando vectores k-dimensionales y representar cada uno de éstos por un vector representativo denominado centroide. En la figura 6 podemos ver un ejemplo para $k = 2$ dimensiones y tres centroides. Es decir, en vez de dividir el rango de la recta real en el que se mueven los valores de entrada en intervalos (cuantificación escalar) dividimos el espacio k-dimensional en celdas o subespacios. De esta forma se aprovechará la correlación entre pixels consecutivos que hará que los vectores se concentren en una determinada zona del espacio. La principal complicación de este tipo de métodos es la división en celdas y el consecuente cálculo de los centroides.

La cuantificación vectorial es la versión multidimensional de la cuantificación escalar. En la cuantificación escalar el análisis de la entrada se realiza elemento a elemento, es decir, el valor de la entrada en cada momento y_n se compara con un conjunto de niveles de decisión ordenados y_i hasta encontrar el par entre los que se encuentra: $y_i < y_n < y_{i+1}$. En consecuencia, el valor de salida será un nivel fijo de reconstrucción para el intervalo al que pertenece la entrada: $\hat{y}_i$.

La determinación del valor de los niveles de decisión y de los niveles de reconstrucción depende de número total de niveles disponibles, el criterio de error y las posibles restricciones que impongan factores como la entropía o la

tasa de bit de salida mínima.

La cuantificación vectorial es una extensión de este planteamiento a varias dimensiones. La entrada en cada momento no es un elemento aislado, sino un bloque de elementos, o un vector de dimensión k. De esta forma todas las componentes de dicho vector $Y = [y_1, y_2, ..., y_k]$ se cuantificarán unidas, llevándose a cabo la división no en la recta real sino en el espacio k-dimensional. Los niveles de decisión serán sustituidos por los bordes de una determinada región C_i de dicho espacio y el nivel de reconstrucción es sustituido por un punto determinado en dicha región $\hat{Y}_i = [\hat{y}_{i1}, \hat{y}_{i2}, ..., \hat{y}_{ik}]$. A estos vectores de reconstrucción se les denomina centroides o vectores código. De esta forma la cuantificación vectorial es equivalente a un mapeo del espacio euclídeo k-dimensional $\Re^k$ en un subconjunto finito formado por N_c centroides según la función dada por:

$$Y \rightarrow \hat{Y}_i \quad \text{si } Y = [y_1, y_2, ..., y_k] \in C_i \tag{5}$$

Al subconjunto finito de $\Re^k$ mencionado se le denomina librería de centroides. Será en esa librería en la que se lleve a cabo la búsqueda del centroide más cercano al vector de entrada. Lo que se transmite para cada vector de entrada es por tanto el índice i del centroide que lo representa, que es reconstruido en el decodificador. El tamaño de la librería es típicamente una potencia de 2, es decir, $N_c = 2^b$, de tal forma que el índice pueda ser representado como un número de b bits.

La complejidad de la búsqueda aumenta linealmente con el número de centroides y exponencialmente con la dimensión. Esta dimensión está determinada por el tamaño de los subbloques en los que se divida la imagen para su codificación. El método de compresión por cuantificación vectorial es por tanto un método por subbloques. Debido al aumento de complejidad con la dimensión, la división típica que se suele utilizar es en subbloques cuadrados de 4×4 pixels. Al no ser subbloques muy grandes se evita también que aparezcan efectos graves por el procesado por subbloques.

Dentro de la cuantificación, la distorsión cobra importancia en dos operaciones. Por un lado, en la fase de entrenamiento, los centroides $\hat{Y}_i$ deben ser elegidos de tal forma que minimicen la distorsión media dentro de la región C_i. Por otro lado, en la fase de codificación, para cada vector de entrada debe elegirse el centroide con el que se logre una distorsión mínima. De todo esto se desprende la importancia que tiene la elección de una correcta medida de la distorsión. La medida más utilizada sigue siendo el error cuadrático medio, dado que es una medida de la energía, no de la amplitud. En el caso de la cuantificación vectorial, siguiendo la nomenclatura, el error cuadrático medio entre el vector original y el de reconstrucción es:

$$d_2(Y, \hat{Y}_i) = \frac{1}{k} \sum_{n=1}^{k} (y_n - \hat{y}_{in})^2 \tag{6}$$

Dado que la operación expresada en la ecuación (6) será reiteradamente realizada durante la fase de búsqueda de cualquier algoritmo de cuantificación

vectorial, una manera de ahorro del coste computacional sería reescribir cada sumando de la forma:

$$(y_n - \hat{y}_{in})^2 = y_n^2 + \hat{y}_{in}^2 - 2y_n\hat{y}_{in} \tag{7}$$

De esta manera, el error cuadrático medio resulta:

$$d_2\left(Y, \hat{Y}_i\right) = \frac{1}{k}\left(\sum_{n=1}^{k} y_n^2 + \sum_{n=1}^{k} \hat{y}_{in}^2 - 2\sum_{n=1}^{k} y_n\hat{y}_{in}\right) \tag{8}$$

Dado que el primer sumatorio es fijo para cada vector de entrada (es su módulo al cuadrado), la determinación del centroide más cercano (de distorsión mínima) para cada entrada se basa únicamente en los dos últimos sumatorios. Por tanto, la búsqueda de dicho centroide es equivalente a la maximización del valor:

$$M\left(Y, \hat{Y}_i\right) = \sum_{n=1}^{k} y_n\hat{y}_{in} - \frac{1}{2}\sum_{n=1}^{k} \hat{y}_{in}^2 \qquad i = 1, ..., N_c \tag{9}$$

La diferencia principal dentro del conjunto de algoritmos basados en la cuantificación vectorial es el método empleado en el diseño de la librería de centroides. A continuación realizaremos un repaso de algunos de ellos:

- Cuantificación vectorial clasificada. Surge en repuesta al problema de los bordes que aparecen en la imagen. Como bordes entendemos las zonas de la imagen en las que se produce un cambio brusco en el nivel de luminancia, marcando el contorno de algún objeto. Dichos bordes, aunque ocupan espacios reducidos dentro de la imagen necesitan un reproducción exacta para satisfacer el criterio perceptual del ojo humano. En el algoritmo básico de la cuantificación vectorial el número de centroides adecuados a los bordes es muy pequeño, debido a su escasa frecuencia de aparición, con lo que la búsqueda para los vectores de borde puede dar como resultado un centroide que no se ajusta bien a él. Para solucionar este problema surge la clasificación. Los vectores de entrenamiento se dividen en borde y sombra y se generan centroides distintos para cada tipo. Los vectores de entrada se clasifican también para codificarlos con el tipo de centroide adecuado. Este concepto se puede ampliar aumentando las posibilidades de la clasificación a más de dos tipos. Para la clasificación se utiliza algún algoritmo de detección de bordes.

- Cuantificación vectorial de media/forma. Se basa en la cuantificación por separado de las propiedades de cada vector. El procedimiento consiste en sustraer a cada componente el valor medio del vector, de modo que el resultado de la sustracción, la forma, se cuantifica vectorialmente, y la media extraída se cuantifica escalarmente por separado. Este método deriva de la afirmación de que muchas entradas tienen formas parecidas pero difieren en su luminancia media.

- Cuantificación vectorial interpolativa. En este caso el proceso comienza con la transmisión mediante alguna codificación sencilla de un elemento

representativo de cada bloque. A partir de él, tanto en el codificador como en el decodificador se interpola un bloque reconstruido el cual se resta a la distribución de luminancia del bloque real en la parte codificadora. Es la señal diferencia lo que se cuantificará vectorialmente y se enviará.

- Cuantificación vectorial multietapa. Este método consiste en realizar una primera cuantificación gruesa a la imagen y a continuación someter a la diferencia entre la salida de esta cuantificación y la imagen original a uno o más procesos de cuantificación posteriores. Los resultados son buenos en imágenes que difieren sustancialmente de las imágenes de entrenamiento.

- Cuantificación vectorial de estado finito. Se basa en la transición entre estados que sufren el codificador y el decodificador. Para la codificación de los vectores de entrada se utilizan una colección de pequeñas librerías, de entre las que se elige una en función del estado del codificador y del decodificador, que es el mismo. El estado en cada momento depende del estado en el instante anterior y de la transición asociada al centroide que se usó en el instante anterior.

- Cuantificación vectorial predictiva. Aprovecha las ventajas de la cuantificación vectorial pero no para la codificación de la imagen sino para la codificación del residuo de una predicción, sustituyendo a la cuantificación escalar.

- Cuantificación vectorial adaptativa. Se basa en el principio fundamental de la compresión de datos que implica la obtención de mejores resultados cuando se utilizan algoritmos de codificación adaptados a las propiedades estadísticas de los datos de entrada, siempre que el diseño del sistema conlleve que la información requerida para el establecimiento de los parámetros variable sea mínima. Así, estos tipos de algoritmos en general fijan una serie de variables en el codificador según la distribución estadística de la imagen.

- Transformada/Cuantificación vectorial. Resulta de la combinación de una transformación con la cuantificación vectorial posterior de los coeficientes transformados. De esta forma se superan dos problemas de la cuantificación vectorial. Por un lado, la necesidad de utilizar subbloques pequeños para disminuir la complejidad computacional, siendo menos complicado aumentar el tamaño de estos subbloques en el dominio transformado. Por otro lado, la distribución estadística de los valores de luminancia en una imagen natural está erráticamente definida, de tal forma que son necesarios complejos procesos de diseño de librería, complejidad que se reduce en el dominio transformado.

4 Algoritmos de Entrenamiento para Cuantificación Vectorial

El principal elemento diferencial entre los distintos métodos de cuantificación vectorial es el algoritmo utilizado para la generación de la librería de centroides a utilizar. En el desarrollo de un proceso de cuantificación vectorial es necesaria la existencia de una librería de centroides, que será el conjunto de vectores k-dimensionales que se utilizarán como representativos del espacio $\Re^k$. En este punto, la cuantificación vectorial difiere de su versión unidimensional. Debido a la difícil definición de la distribución estadística de los niveles de luminancia en una imagen no se suelen utilizar las distribuciones analíticas multidimensionales como base para establecer los intervalos de decisión del cuantificador, sino que su diseño suele basarse en la aplicación de métodos de mínima distorsión aplicados a una secuencia de datos de entrenamiento. Dado que el valor de cualquier medida de la distorsión incrementa con la distancia entre los vectores de entrada y los vectores código correspondientes, estos métodos establecerán la correspondencia óptima según la regla del vecino más próximo:

$$Y \rightarrow \hat{Y}_i$$

$$\text{Si } d\left(Y, \hat{Y}_i\right) < d\left(Y, \hat{Y}_j\right) \quad \text{para todo } j \neq i \tag{10}$$

Esta regla también definirá las regiones de cuantificación C_i.

4.1 Algoritmo LBG

En esta línea, el algoritmo más clásico es el debido a Y. Linde, A. Buzo y R. M. Gray, conocido por las iniciales de los autores como algoritmo LBG. Cada paso de este algoritmo se compone dos partes. En la primera, partiendo de una librería de vectores código obtenida en el paso anterior, se buscará la partición óptima del espacio, en la que cada vector pertenecerá a la región asociada al vector de reproducción más cercano. En la segunda, basándose en dicha partición, se hallarán los nuevos centroides de cada región, que serán los vectores de reproducción óptimos para este paso.

Cada iteración del algoritmo de entrenamiento LBG, basado en un conjunto de vectores de entrenamiento, tiene dos partes:

1. Partición del conjunto de vectores de entrenamiento en un número de categorías o regiones igual al número requerido de vectores código N_c. En esta primera parte de la iteración se utiliza el conjunto de vectores código $\hat{\mathbf{Y}}(n-1) = \left\{\hat{Y}_i(n-1),\ i = 1, ..., N_c\right\}$ que se calculó en la segunda parte de la iteración anterior. En este caso la selección para cada vector de entrenamiento Y del miembro más cercano del conjunto $\hat{\mathbf{Y}}(n-1)$ tendrá como resultado una partición $P\left(\hat{\mathbf{Y}}(n-1)\right)$ que será la óptima.

2. Determinación del vector código óptimo para cada categoría. En esta segunda fase tenemos la partición $\mathbf{S}(n) = P\left(\hat{\mathbf{Y}}(n-1)\right)$, siendo conocidos todo el conjunto de vectores de entrenamiento pertenecientes a cada región $S_i(n)$. De esta forma, estableceremos el vector código $\hat{Y}_i(n)$ de la región $S_i(n)$ como el centro de gravedad generalizado o centroide de los vectores de dicha región de tal forma que:

$$E\left[d\left(Y, \hat{Y}_i(n)\right) / \ Y \in S_i(n)\right] = \min_u E[d(Y,u) \ / \ Y \in S_i(n)] \tag{11}$$

Si asumimos la medida de la distancia euclídea y dado que en la práctica los vectores código óptimo se calculan con un conjunto finito de vectores de entrenamiento $\mathbf{Y} = \{Y_1, Y_2, ..., Y_T\}$ con $T >> N_c$, podemos aproximar el centroide de cada región por:

$$\hat{Y}_i(n) \approx \frac{1}{t_i(n)} \sum_{Y_t \in S_i(n)} Y_t, \ i = 1, ..., N_c \tag{12}$$

donde $t_i(n)$ es el número de vectores de entrenamiento pertenecientes a $S_i(n)$. Se deberá cumplir que:

$$\sum_{i=1}^{N_c} t_i(n) = T, \ \forall n \tag{13}$$

El criterio de finalización es la variación relativa de la distorsión media $D(n)$ en cada paso, dato que a medida que la distorsión converja a su valor asintótico, se irá reduciendo hasta alcanzar el umbral ϵ, valor que se considera aceptablemente bajo. La distorsión de cada vector de entrenamiento $D_t(n)$ viene dada por:

$$D_t(n) = \min_{i=1,...,N_c} \left\{ d\left(Y_t, \hat{Y}_i(n)\right) \right\}, \ t = 1, ..., T \tag{14}$$

y la distorsión media viene dada por la ecuación:

$$D(n) = \frac{1}{T} \sum_{t=1}^{T} D_t(n) \tag{15}$$

El criterio de parada es:

$$\frac{D(n-1) - D(n)}{D(n)} < \epsilon \tag{16}$$

El único punto que nos queda por determinar es el establecimiento de los valores iniciales de los vectores código que conforman el conjunto inicial $\hat{\mathbf{Y}}(0)$. Para este fin desarrollamos un proceso que comienza con la determinación del centro de gravedad global del conjunto de entrenamiento al completo. Cada una de las componentes de este único vector representativo es perturbada con pequeñas cantidades aleatorias $\pm\delta$. Esta perturbación se realiza dos veces dando lugar a dos nuevos centroides. El proceso continúa de la misma manera, dividiéndose estos dos valores iniciales en otros dos, iterándose hasta alcanzar

Figura 7: Ejemplo de cuantificación vectorial de imagen utilizando algoritmo LBG. Imagen original y reconstruida.

SNR (dB)	Tasa de compresión
22.6916	22.5987 : 1

Tabla 1: Resultados de la codificación con cuantificación LBG.

el número de centroides requerido, que será potencia de dos, de cara a su posterior codificación binaria. Al final de este proceso, la librería de centroides inicial será aproximadamente una nube de vectores alrededor del centro de gravedad del conjunto de entrenamiento

En la figura 7 recogemos un ejemplo de una imagen y su equivalente reconstruida utilizando cuantificación vectorial con librería de centroides generada mediante algoritmo LBG. En el cuadro 1 se puede ver la tasa de compresión y la calidad lograda.

Al algoritmo LBG se han ido añadiendo una serie métodos que pretenden generar librerías de forma más eficiente. A continuación veremos algunos de ellos.

4.2 Librería en Árbol.

Se trata de un algoritmo que define la librería mediante una estructura en árbol. El algoritmo parte del centroide o centro de gravedad de la totalidad del conjunto de vectores de entrenamiento. Este vector representativo único es perturbado mediante la suma a cada una de sus componentes de una cantidad diferencial aleatoria δ. Mediante dos perturbaciones obtenemos dos nuevos vectores representativos que serán utilizados para establecer una partición del espacio en dos regiones. Siguiendo el algoritmo LBG, a continuación se hallan

los centroides de cada una de estas regiones, que a su vez se vuelven a dividir en dos dando lugar a una nueva partición con el doble de regiones. El proceso se repite hasta alcanzar el número N_c de centroides fijado, siendo N_c una potencia de dos.

Para el establecimiento de la estructura en árbol es necesario que no sólo se almacenen los N_c centroides finales, sino también los que aparecen en los pasos intermedios, estableciendo una sucesión de ramas. De cada centroide calculado en un paso intermedio parten dos centroides pertenecientes a su región asociada, que a su vez dividen dicha región en dos subregiones. El nodo inicial del árbol es el centro de gravedad del conjunto de vectores de entrenamiento, cuya región asociada es todo el espacio. De esta forma se simplifica considerablemente la búsqueda durante el proceso de cuantificación, ya que ésta se realiza siguiendo un camino en el árbol, realizando decisiones binarias en cada nivel de la estructura: cada entrada pertenece a una de las dos regiones en las que se divide el espacio en el primer nivel y dentro de su región pertenece a una de las dos subregiones en las que se dividen las regiones primarias y así sucesivamente.

En su forma más sencilla, cada rama más baja se divide en dos más altas. Los algoritmos de división binaria en cada nodo tiene la ventaja de producir una estructura uniforme pero los resultados que se obtiene son peores a los que se conseguirían si relajamos este requerimiento, dividiendo cada nodo en función de la distorsión que aparece en la región que representa. Surge así un algoritmo de estructura en árbol no uniforme, basado en la división sucesiva sólo de los nodos con más distorsión. Los resultados que se obtienen en este caso se acercan más a los de la búsqueda exhaustiva (comparación con todos los centroides). El número de centroides sólo tiene que ser un entero y no una potencia de dos como en el caso anterior.

Otra modificación de la estructura en árbol añade el concepto de podado. La idea básica consiste en la eliminación de ciertas ramas para compensar la creación de otras nuevas. Como criterio de eliminación en el proceso se analiza la relación λ_p asociada a cada nodo, definida como la relación entre el incremento de la distorsión con respecto al decremento de la tasa de transmisión que se produce como consecuencia de la eliminación (podado) de la rama que cuelga de dicho nodo. En la dirección de crecimiento (en los nodos finales del árbol, susceptibles de generar nuevas ramas), lo que se analiza es λ_g, es decir, a la inversa, la relación entre el decremento de la distorsión con respecto al incremento de la tasa de transmisión. De esta forma, el árbol es generado mediante la división en dos nuevas ramas de los nodos con los valores más altos de λ_g (con los que se obtienen mejores resultados) y el podado de aquellos cuyo valor de λ_p sea menor (cuya eliminación implica una pérdida menor). Incluso existen modificaciones que permiten la división de los nodos en tres o cuatro ramas, no sólo en dos.

4.3 Algoritmo PNN

Con un planteamiento totalmente diferente encontramos el algoritmo vecinos más cercanos por parejas o PNN (Pairwise Nearest Neighbour). Este algorit-

mo opera en sentido inverso a la estructura en árbol, ya que agrupa los vecinos más próximos dentro del conjunto de vectores de entrenamiento hasta que es alcanzado el tamaño de librería requerido. En un principio se agrupan los vectores de entrenamiento por parejas, siendo la pareja de cada uno el vector de entrenamiento vecino más cercano, y se calculan los centroides de cada región formada por dos vectores de entrenamiento. Tenemos entonces el espacio dividido en un número de regiones igual a la mitad del número de vectores de entrenamiento, estando asociada a cada región un vector representativo. En cada paso se lleva a cabo la unión de dos regiones en una nueva, y la obtención del centroide de esta nueva región. Las dos regiones vecinas C_i y C_j que se unirán en una sola son elegidas de tal forma que se minimice el error introducido por la unión de dichas regiones dado por la ecuación:

$$e_{ij} = \frac{n_i n_j}{n_i + n_j} \left| \hat{Y}_i - \hat{Y}_j \right|^2 \tag{17}$$

En esta ecuación, los enteros n_i y n_j son el número de vectores que contiene cada región e $\hat{Y}_i$ e $\hat{Y}_j$ son los centroides de dichas regiones. Podemos observar que el criterio que se aplica es el de aunar una región C_i con la región vecina C_j cuyo centroide $\hat{Y}_j$ sea el más cercano al centroide de la primera región $\hat{Y}_i$. La unión de regiones termina cuando se alcanza un determinado número de centroides o cuando la distorsión alcanza un umbral determinado.

4.4 Relajación Estocástica

La relajación estocástica es un método general para la optimización de problemas complejos. En el caso que estudiamos, para una determinada librería se selecciona una temperatura como parámetro de control del algoritmo y la cantidad $\exp(-\Delta E/T)$ es calculada cuando la librería es sometida a alguna perturbación tal como mover un vector de entrenamiento aleatoriamente de una partición a otra o añadir una variación aleatoria a algún centroide. En estos casos, la variación de la energía (distorsión) es calculada y si es negativa, la nueva asignación se acepta. En caso contrario, la nueva asignación será aceptada con probabilidad $\exp(-\Delta E/T)$. Por tanto, a altas temperaturas, las variaciones serán probablemente aceptadas incluso aunque supongan un aumento de la distorsión.

Cuando el proceso se considera terminado a una determinada temperatura, bien sea a causa de una limitación del número de iteraciones o porque la distorsión a disminuido en una cantidad aceptable, la temperatura se reduce y el proceso se repite. A medida que la temperatura disminuye también lo hace la probabilidad de aceptar una nueva asignación, alcanzándose el final del algoritmo cuando se consigue una librería estable durante sucesivas reducciones de temperatura.

La principal ventaja de este algoritmo es su capacidad de converger a un mínimo global, no local, de la distorsión, pero su coste computacional es muy alto.

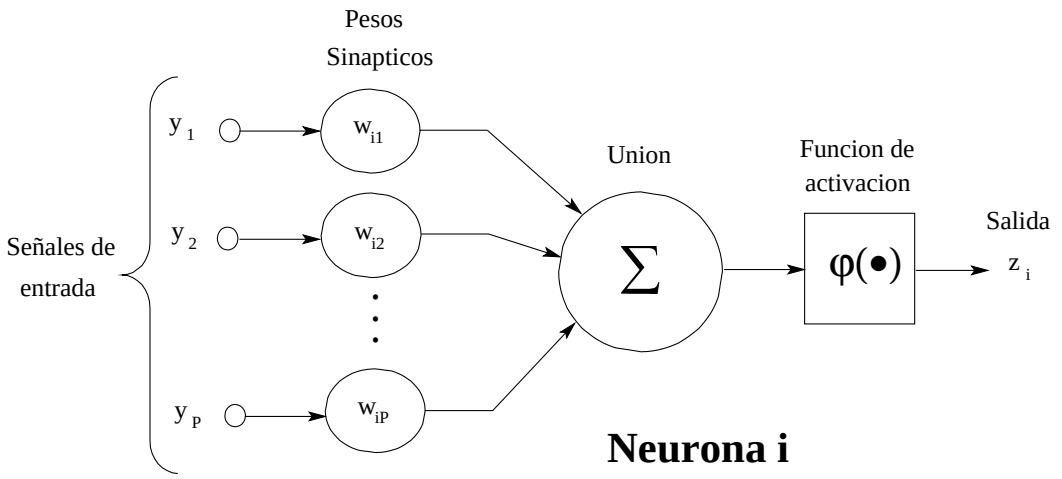

Figura 8: Modelo no lineal de una neurona.

4.5 Redes Neuronales

Una de las últimas soluciones para el diseño de librerías de centroides consiste en la aplicación de técnicas basadas en redes neuronales, de amplia aplicación en el procesado de señales.

Una red neuronal podría definirse como un procesador distribuido masivamente en paralelo, compuesto por unidades de procesamiento simples, que tiene una tendencia natural a almacenar conocimiento experimental y tenerlo disponible para su uso. Esta red se asemeja en su funcionamiento al cerebro humano en dos aspectos:

- El conocimiento es adquirido por la red de su entorno a través de un proceso de aprendizaje.
- Las fuerzas de interconexión entre las neuronas, denominadas pesos sinápticos, son usadas para almacenar el conocimiento adquirido.

El procedimiento usado para llevar a cabo el proceso de aprendizaje se denomina algoritmo de aprendizaje. Será un procedimiento de este tipo el que se empleará para el entrenamiento de la librería de vectores código.

La unidad de procesado de información fundamental para la realización de una red neuronal es la neurona. El diagrama de bloques del modelo genérico de una neurona se reproduce en la figura 8. Los tres elementos básicos del modelo neuronal son los siguientes:

- Un conjunto de sinapsis, cada una de las cuales está caracterizada por un peso propio. Específicamente, una señal y_p a la entrada de la sinapsis p conectada a la neurona i es multiplicada por un peso w_{ip}.
- Un sumador para sumar las señales de entrada ponderadas por la respectiva sinapsis de la neurona.
- Una función de activación que limita la amplitud de la salida de la neurona a un valor finito.

La teoría expuesta hasta ahora responde al modelo genérico de redes neuronales. Para su adaptación a la cuantificación vectorial de imágenes es necesario redefinir algunos aspectos. De entrada, consideraremos la imagen dividida en subbloques como un conjunto de vectores del espacio k-dimensional. Serán estos vectores Y los que constituyan el conjunto de señales de entrada de la red neuronal.

Suponemos que el tamaño de la librería de vectores código es N_c, y dichos vectores serán $\hat{Y}_i, \quad i = 1, ..., N_c$. De esta forma, consideraremos una red neuronal con N_c neuronas o unidades neuronales, cada una de las cuales tiene una sola entrada y por tanto, una sola sinapsis. Haremos que el vector código i-ésimo $\hat{Y}_i$ sea el peso asociado a la sinapsis de la neurona i, W_i. Por otro lado, en la definición de neurona vista, la ponderación de la entrada para cada sinapsis se realizaba mediante una simple multiplicación de la entrada por el peso correspondiente. En el caso de la cuantificación vectorial, para esta ponderación es necesario una operación más compleja, el cálculo de la distorsión $d\left(Y, \hat{Y}_i\right)$ entre el vector Y y el vector código $\hat{Y}_i$ que lo representa. Será sobre esta distorsión sobre la que se aplicará la función de activación. Podemos observar que en la estructura definida no aparece el sumador, lo cual es debido a que cada neurona tiene una única sinapsis.

La cuantificación seguiría el siguiente proceso: cada vector Y a cuantificar es alimentado en paralelo a las N_c neuronas de la red. Cada una de ellas proporciona una salida que resulta de ponderar la entrada con el peso asociado a su sinapsis mediante el cálculo de la distorsión entre ambos. Esta salida a su vez será la entrada de la función de activación de cada neurona.

Con respecto a la función de activación, el tipo que utilizaremos es el tipo umbral, es decir, la salida z_i sólo toma el valor 1 ó 0. Existen otros tipos de funciones de activación que, para ciertos valores de entrada, pueden tomar valores intermedios, pero no son adecuadas en este caso. La definición exacta de esta función puede ser distinta en la fase de codificación y en la fase de entrenamiento. En la fase de codificación, el valor 1 significa que la unidad neuronal corresponde al vector código que representará al vector de entrada. Así, para cada entrada, la función de activación sólo puede ser igual a 1 en una neurona. Lógicamente, el valor 1 lo toma la neurona cuya $d\left(Y, W_i\right)$ sea menor, es decir, la que corresponde al vector código más cercano al vector de entrada. Matemáticamente, para la neurona i, la salida de la función de activación sería de la forma:

$$z_i = \begin{cases} 1 & d[Y, W_i(n)] \leq d[Y, W_j(n)], \qquad j = 1, ..., N_c \\ 0 & \text{en otro caso} \end{cases} \tag{18}$$

En lo que se refiere a la fase de entrenamiento, los algoritmos basados en redes neuronales hacen uso de la capacidad de almacenar conocimiento de estas estructuras. Como dijimos, el conocimiento se almacena en los pesos sinápticos, los cuales serán actualizados con cada nueva entrada. La actualización de los pesos de las sinapsis estará en función de la entrada, del peso actual y de la salida de la neurona. Traducido en términos de cuantificación vectorial, tras el paso de un vector de entrada por la red neuronal se actualizan los vectores

código (que son los pesos) en función de su diferencia con el vector de entrada si así lo indica la salida de la función de activación correspondiente. Este proceso lo veremos con más detalle en la descripción de los algoritmos que veremos a continuación.

4.5.1 Red de Aprendizaje Competitivo

Para la ejecución de este algoritmo asumimos que la red neuronal será entrenada por un conjunto amplio de vectores de entrenamiento. Partiremos de la inicialización de las N_c unidades neuronales con los vectores peso $W_i(0), \;\; i = 1, ..., N_c$. Estos pesos pueden ser generados aleatoriamente o ser los primeros N_c vectores del conjunto de entrenamiento. Partiendo de esta base, el algoritmo de entrenamiento realizará diversas pasadas por los vectores de entrenamiento, ajustando los vectores peso de las unidades neuronales después de la entrada de cada vector de entrenamiento. El algoritmo de ajuste está basado en el aprendizaje competitivo. Comienza con la presentación del vector de entrada Y a todas la unidades neuronales cada una de las cuales calcula la distorsión entre su peso y el vector de entrada. La unidad de distorsión menor es designada como *ganadora* y su peso es ajustado hacia el vector de entrada. Expresado matemáticamente, partiremos del peso $W_i(n)$ de la unidad i en el paso n antes de la entrada del nuevo vector, siendo la salida z_i de la función de activación calculada de la forma:

$$z_i = \begin{cases} 1 & d[Y, W_i(n)] \leq d[Y, W_j(n)], \qquad j = 1, ..., N_c \\ \\ 0 & \text{en otro caso} \end{cases} \tag{19}$$

Los nuevos vectores peso $W_i(n+1)$ serán calculados por lo tanto como:

$$W_i(n+1) = W_i(n) + \eta(n)[Y - W_i(n)]z_i \tag{20}$$

El parámetro $\eta(n)$ es la relación de aprendizaje y suele decaer monótonamente a cero a medida que avanza el proceso de aprendizaje. El principal problema que surge con este algoritmo es la infrautilización de algunas unidades neuronales que no resultan ganadoras nunca.

4.5.2 Mapas Autoorganizado de Kohonen

Los mapas autoorganizados o KSFM (Kohonen Self-organizing Feature Map) conciben una estructura de red parecida al aprendizaje competitivo, pero donde cada unidad neuronal tiene asociada una vecindad topológica con otras unidades neuronales. Durante el proceso de entrenamiento, son actualizadas tanto las unidades neuronales ganadoras como aquellas que se encuentran en su vecindad. El tamaño de esta vecindad decrece a medida que avanza el entrenamiento hasta alcanzar la unidad, es decir, la red KSFM se transforma en una red de aprendizaje competitivo.

Sea $W_i(n)$ el peso asociado a la i-ésima unidad neuronal y sea Y el vector de entrada. Calculada la distorsión $d[Y, W_i(n)], \;\; i = 1, ..., N_c$ se establecerá que la neurona de índice i^* es la de menor distorsión. Por otro lado, estará

definido el subespacio $N_{i^*}(n)$ como la vecindad topológica asociada a la unidad i^*. La vecindad topológica incluirá a aquellos vectores que se encuentren a una distancia inferior a un umbral (que irá disminuyendo) de la unidad ganadora. Con todos estos elemento la ecuación de actualización del peso, de la que se puede deducir la función de activación utilizada, será:

$$W_i(n+1) = \begin{cases} W_i(n) + \eta(n)[Y - W_i(n)] & i \in N_{i^*}(n) \\ \\ W_i(n) & \text{en otro caso} \end{cases} \tag{21}$$

Es lógico deducir que la estructura KSFM implica una aumento considerable de la carga computacional con respecto al aprendizaje competitivo, pues las actualizaciones afectan a toda la vecindad. Sin embargo, es este mismo hecho el que permite solucionar el problema de la infrautilización de unidades neuronales visto. Por otro lado, también supone un notable incremento de la complejidad el cálculo de las vecindades en cada paso del algoritmo.

5 Codificación

En la sección anterior hemos discutido el problema de cuantificar escalar o vectorialmente una fuente de información. Como resultado de esta cuantificación, se obtiene un nivel específico de reconstrucción. Para transmitir hasta el receptor esos posibles $L = 2^n$ niveles de cuantificación (o para representar en la imagen comprimida) se necesita asignar una palabra código a cada nivel. Así, el decodificador podrá identificar el nivel de reconstrucción correspondiente mirando a la entrada apropiada de tabla de palabras código. Para que esto sea posible a cada nivel de cuantificación se le debe asignar una palabra código diferente. Además, como se transmitirá (o se tendrá almacenado) muchas palabras código de forma secuencial, se deben diseñar de forma que se puedan identificar cada una de ellas en cualquier secuencia. Un código con estas características se denomina decodificable de forma única.

5.1 Códigos de Longitud Fija

Supongamos que el resultado de la cuantificación escalar o vectorial es un mensaje con L posibles valores $a_i, i = 0, \ldots, L-1$ cada uno de ellos correspondiendo a un nivel de reconstrucción. El método más sencillo de obtener el conjunto de palabras código es utilizando un código de longitud fija. Usando este método cada elemento del mensaje se codifica mediante una palabra código que tiene la misma longitud que el resto. La forma más sencilla de construir estos códigos es representando cada a_i pasando el valor del índice i a base binaria usando $n = \log_2(L)$ dígitos binarios o bits. En la tabla 2 podemos ver un ejemplo para $L = 8$ niveles.

5.2 Códigos de Longitud Variable y Entropía

La codificación Huffman es una codificación de longitud variable que consiste en asignar a cada posible salida de un codificador de datos un símbolo de canal

Nivel de reconstrucción	Palabra código
a_0	0 0 0
a_1	0 0 1
a_2	0 1 0
a_3	0 1 1
a_4	1 0 0
a_5	1 0 1
a_6	1 1 0
a_7	1 1 1

Tabla 2: Ejemplo de un código de longitud fija.

según su probabilidad de aparición. Se trata de un codificador de entropía, pues trata de aproximar la longitud media de símbolo a la entropía del sistema.

La entropía (h) de un sistema de comunicaciones caracteriza la información que los símbolos o señales de dicho sistema contiene y viene dada por:

$$h = -\sum_{sistema} p(i) \log_2 p(i) \tag{22}$$

El sumatorio se aplica a todo símbolo i posible del sistema, cuya probabilidad de ocurrencia es $p(i)$. La entropía constituye el límite inferior de tamaño medio de código alcanzable.

El algoritmo de codificación Huffman (usada en el estándar JPEG), diseñado por D. A. Huffman, trata de acercarse al límite impuesto por la entropía del sistema. Para entenderlo es conveniente estudiar un ejemplo sencillo, el que mostramos en la figura 9. Partimos de un conjunto de 5 símbolos, los cuales ordenamos en orden descendente según su probabilidad de ocurrencia, tal como vemos en la figura. A continuación agrupamos los dos símbolos inferiores creando uno nuevo de probabilidad igual a la suma de las probabilidades de dichos símbolos y reordenamos. Esto se repite hasta llegar a la probabilidad unidad. En este punto recorremos la cadena hacia atrás, asignando valores distintos (uno o cero) a cada rama de una unión, concatenándolo con los valores asignados anteriormente, de tal forma que el código para cada símbolo inicial se forma recorriendo su trayectoria de derecha a izquierda.

En este ejemplo, si la codificación hubiera sido de tamaño de código fijo, este tamaño hubiera sido igual 3 bits por símbolo. Con la codificación Huffman, multiplicando el tamaño de cada símbolo por la probabilidad de que aparezca y sumándolo todo, obtenemos un tamaño medio igual a 1.98 bits por símbolo. Este número se acerca considerablemente a la entropía, que en este caso es igual a 1.955, según la ecuación (22). Resulta por tanto notable la conveniencia de la utilización de este tipo de codificación.

Debemos destacar además que tal como se forman las palabras códigos, es imposible que coincida un segmento inicial de una de ellas con otra completa. Si el cero es utilizado como símbolo aislado, no será utilizado como principio

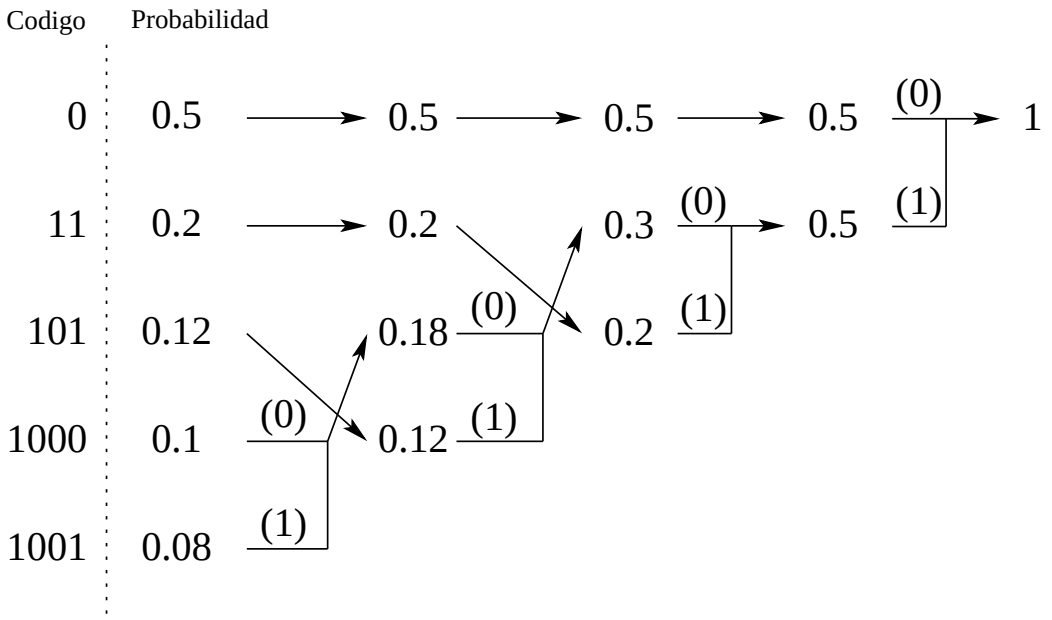

Figura 9: Ejemplo de generación de códigos Huffman.

de ningún otro símbolo y así con cualquier combinación. Esto evita en la decodificación cualquier confusión, aunque no se indique explícitamente el límite entre símbolos, siempre que se conozcan las palabras códigos fijadas.

6 Codificadores de Forma de Onda

En la codificación de forma de onda se codifica directamente los niveles de intensidad de la imagen o alguna variación sencilla de estos niveles de intensidad como la diferencia entre pixels consecutivos. La ventaja más importante de este tipo de codificación es su simplicidad. Los codificadores de forma de onda no pretenden explotar las características específicas de una clase particular de señales y por lo tanto se van a poder utilizar para un rango muy amplio de señales (voz e imagen). En algunos casos la tasa de compresión alcanzada puede ser similar a los codificadores de transformada, pero en otros casos las tasas alcanzadas son significativamente menores.

En principio, se podría utilizar diferentes tipos de cuantificación y codificación en los codificadores de forma de onda, sin embargo, en general, se utiliza cuantificación escalar y codificación de longitud fija.

6.1 PCM

Es la forma más sencilla de codificación de forma de onda. Simplemente, la imagen de intensidad se pasa a través de un cuantificador uniforme. En la figura 10 podemos ver el esquema de este tipo de codificación. El sistema PCM no sólo se puede utilizar para codificar los niveles de intensidad, sino que se puede utilizar para codificar los coeficientes transformados (en el caso

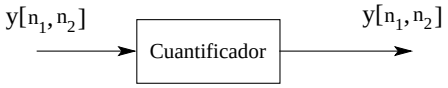

Figura 10: Sistema PCM.

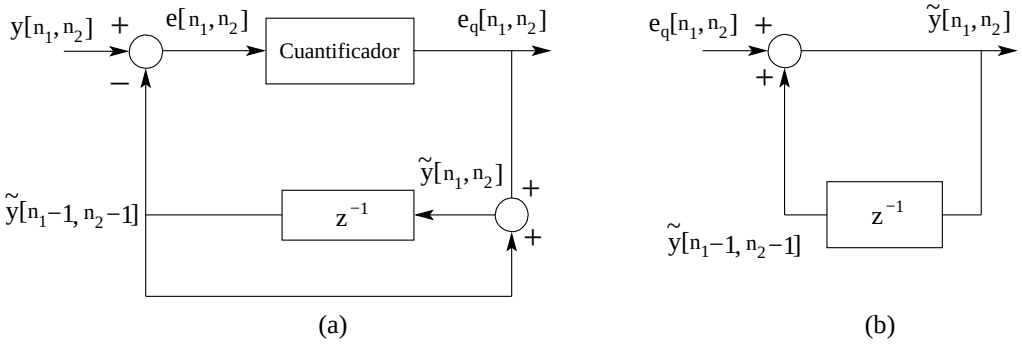

Figura 11: Sistema DM. (a) Codificador. (b) Decodificador.

de codificación de transformada) y los parámetros (en el caso del codificación basada en modelos).

Una forma sencilla de mejorar el sistema PCM básico es utilizando un cuantificador no uniforme en lugar del uniforme ya que los niveles de intensidad no suelen estar distribuidos de forma uniforme. Para ello se suele utilizar la técnica de compansión ya explicada.

6.2 Modulación Delta (DM)

En un sistema PCM la intensidad de la imagen se cuantifica escalarmente, pero no es explota para nada la correlación existente entre los pixels de la imagen. Una forma de hacer esto manteniendo la cuantificación escalar es la modulación delta (DM). En un sistema DM, se codifica con un bit (2 niveles) la diferencia entre la intensidad de dos pixels consecutivos. Aunque el rango dinámico de la señal diferencia es el doble, la varianza de la señal diferencia es significativamente menor debido a la elevada correlación existente entre los niveles de intensidad de dos pixels cercanos en la imagen. En la figura 11 podemos ver el esquema del codificador y del decodificador DM. La señal error $e[n_1, n_2]$ del pixel actual con respecto al anterior se cuantifica con un bit de forma que $e_q[n_1, n_2]$ toma el valor $\Delta/2$ si $e[n_1, n_2]$ es positivo y $-\Delta/2$ si es negativo. Esta señal cuantificada se codifica con un bit y se transmite. En el receptor la señal recuperada se obtiene sumando el error cuantificado $e_q[n_1, n_2]$ a la señal reconstruida para el pixels anterior.

Un parámetro importante de diseño del sistema DM es el tamaño del escalón Δ. Cuando la señal varía lentamente, la señal recuperada varía rápidamente en torno de la señal deseada ($\pm\Delta/2$ en torno a ella). Este tipo de error se conoce como error granular. Un valor grande de Δ dará lugar a un error granular mayor, por lo que se prefieren valores de Δ pequeños. Cuando la señal varía (crece o decrece) rápidamente, si Δ es pequeño la señal reconstruida

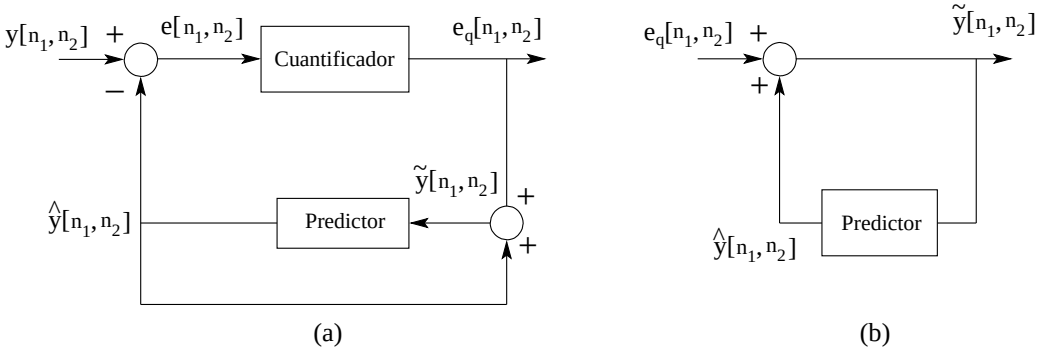

Figura 12: Diagrama de un codificador predictivo. (a) Codificador. (b) Decodificador.

tardará bastante en alcanzar a la señal deseada. Este error se conoce como error de sobrecarga de pendiente. Para este caso valores de Δ grandes son deseables. Es por tanto necesario tomar una solución de compromiso entre el error granular y el de sobrecarga de pendiente.

6.3 DPCM

DPCM es un método predictivo que se puede considerar una generalización del método DM: se utilizan más de un pixel para predecir el actual y se utiliza un cuantificador de más de un bit. Los métodos de codificación predictivos pretenden eliminar la correlación existente entre pixels consecutivos antes de la cuantificación. Para ello tanto en el codificador como en el decodificador se realizan predicciones de la muestra actual en función de las muestras anteriores, de tal manera que el valor que se cuantifica es la diferencia entre la predicción y el valor real del pixel. Mediante esta resta suprimimos en cada punto la información redundante dada ya por los puntos anteriores. El diagrama de bloques genérico de este tipo de compresores se muestra en la figura 12, tanto la parte del codificador como la parte del decodificador.

La señal de partida está constituida por una secuencia de valores discretos que forman bidimensionalmente la imagen digitalizada original, $y[n_1, n_2]$. A cada valor se le sustrae la predicción calculada, la señal $\hat{y}[n_1, n_2]$. Esta predicción no se realiza directamente a partir de la señal $y[n_1, n_2]$, sino que el predictor tiene como entrada la señal que se obtendrá en el decodificador, $\tilde{y}[n_1, n_2]$. Esto se hace para evitar la acumulación de errores en el decodificador. El error de predicción será por lo tanto:

$$e[n_1, n_2] = y[n_1, n_2] - \hat{y}[n_1, n_2] \tag{23}$$

Sobre esta secuencia se realiza una cuantificación reduciendo el número de niveles, lo que añadirá cierto ruido de cuantificación $q[n_1, n_2]$. Es este error de predicción cuantificado lo que constituirá la imagen comprimida. La señal que se recibirá en el lado del decodificador será por tanto:

$$e_q[n_1, n_2] = e[n_1, n_2] + q[n_1, n_2] \tag{24}$$

En el decodificador vemos que se lleva a cabo exactamente la misma predicción que en el codificador a partir de la misma señal $\tilde{y}[n_1, n_2]$, añadiéndole la secuencia $e_q[n_1, n_2]$ a la señal predicha. La señal reconstruida es por tanto:

$$\begin{aligned} \tilde{y}[n_1, n_2] &= e_q[n_1, n_2] + \hat{y}[n_1, n_2] \\ &= e[n_1, n_2] + q[n_1, n_2] + \hat{y}[n_1, n_2] \\ &= y[n_1, n_2] + q[n_1, n_2] \end{aligned} \tag{25}$$

El principal problema de este tipo de compresores es la determinación de los coeficientes del predictor. Sabemos que la señal predicha será una combinación lineal fija de muestras anteriores ponderadas, lo cual se puede expresar matemáticamente de la forma:

$$\hat{y}[n_1, n_2] = \sum_{i=1}^{p_1} \sum_{j=1}^{p_2} a_{i,j} \tilde{y}[n_1 - i, n_2 - j] \tag{26}$$

donde $a_{i,j}$ serán los coeficientes del predictor y p_1, p_2 los niveles del predictor en ambas direcciones. Normalmente para el cálculo de coeficientes que optimicen el predictor se suele aproximar $\tilde{y}[n_1, n_2]$ por $y[n_1, n_2]$ para evitar la componente no lineal $q[n_1, n_2]$. Esta aproximación es válida si el número de niveles del cuantificador del error de predicción no es demasiado pequeño. Los pesos del predictor se determinan mediante un proceso de minimización del error cuadrático medio del error de predicción.

7 Codificadores de Transformada

El objetivo principal de los codificadores de transformada es alterar la distribución de los valores que representan el nivel de intensidad para que muchos de ellos puedan ser eliminados o por lo menos cuantificados con muy pocos bits. De esta forma conseguimos, al igual que en los métodos que utilizan predicciones, una disminución de la dependencia estadística de los pixels antes de pasar a la fase de cuantificación.

La transformación no se realiza sobre toda la imagen sino que ésta es dividida en bloques de tamaño fijo (normalmente de 8×8 o 16×16 pixels) y sobre ellos se realiza la transformación. De esta forma reducimos el coste computacional considerablemente.

Una forma de entender el resultado de la transformación es asimilar cada bloque de 8×8 puntos de la imagen original como un vector en un espacio de 64 dimensiones, expresado en los ejes canónicos. La transformación realizará un cambio a otra base con el objetivo de que un número pequeño de los nuevos vectores base sigan las direcciones principales de los datos, de tal forma que en las nuevas coordenadas tengamos dos tipos de componentes: unas (en menor número) tomarán valores grandes y otras (en mayor número) se aproximarán a cero. Por lo tanto el número de niveles con los que se cuantificará cada una de las nuevas coordenadas variará de unas a otras.

Matemáticamente, la transformación consiste en expresar la función bidimensional original $y[n_1, n_2]$ como combinación lineal de una familia de fun-

ciones $a^*_{k_1,k_2}[n_1,n_2]$ (con el símbolo * nos referimos a la conjugación) ponderada cada una por un coeficiente $TU[k_1,k_2]$. La secuencia bidimensional de estos coeficientes constituyen la señal transformada. Hablamos de transformaciones unitarias, de tal forma que las funciones utilizadas en la transformación deben ser ortonormales entre sí. Todo esto podemos expresarlo mediante las siguientes igualdades (suponiendo las señales de tamaño $N \times N$):

$$TU[k_1,k_2] = \sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1} y[n_1,n_2]a_{k_1,k_2}[n_1,n_2] \tag{27}$$

$$y[n_1,n_2] = \sum_{k_1=0}^{N-1}\sum_{k_2=0}^{N-1} TU[k_1,k_2]a^*_{k_1,k_2}[n_1,n_2] \tag{28}$$

La ortonormalidad de las familia de funciones implica lo siguiente:

$$\sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1} a_{k_1,k_2}[n_1,n_2]a^*_{k'_1,k'_2}[n_1,n_2] = \begin{cases} 0 & si\ k_1 \neq k'_1\ o\ k_2 \neq k'_2 \\ \\ 1 & si\ k_1 = k'_1\ y\ k_2 = k'_2 \end{cases} \tag{29}$$

Como dijimos anteriormente, los subbloques de las imágenes pueden entenderse como matrices de tamaño $N \times N$. Por otro lado, en la mayoría de los casos, la familia de funciones ortonormales de la transformación podrán formar una matriz separable. Por tanto, las transformaciones equivalen a la premultiplicación de cada bloque de la imagen original por una matriz de transformación de tamaño $N \times N$ y la postmultiplicación por la traspuesta de dicha matriz de transformación. La operación matricial de la transformación será de la forma :

$$[TU] = [A][Y][A]^T \tag{30}$$

Algunas de las transformaciones que veremos no parten directamente de una familia de funciones sino que están basadas en matrices de transformación cuyas filas son ortonormales entre sí. Sin más, pasamos a ver algunas de las transformaciones más utilizadas.

7.1 Transformada de Fourier Discreta (DFT)

Es la particularización de la transformada de Fourier para las funciones discretas bidimensionales. La familia de funciones ortonormales será de la forma:

$$a_{k_1,k_2}[n_1,n_2] = \frac{1}{N} e^{-jk_1\frac{2\pi}{N}n_1} e^{-jk_2\frac{2\pi}{N}n_2} \tag{31}$$

Una de las características de la imagen es que en el dominio transformado la energía se concentra en los coeficientes con k_1 y k_2 pequeños (bajas frecuencias espaciales), lo cual es muy conveniente para la compresión. Sin embargo no se pueden obviar los coeficientes de las frecuencia más altas pues contiene la información de los bordes a los cuales es muy sensible el ojo humano. Por lo tanto se opta por una cuantificación con un número de niveles variable según el coeficiente, disminuyendo conforme aumenta la frecuencia.

Una de las ventajas que presenta esta transformación es la separabilidad de cada función base bidimensional en dos unidimensionales idénticas. Además, existe un algoritmo que permite calcular la transformada de una función con una reducción considerable del coste computacional, el algoritmo FFT. Por el contrario presenta dos grandes inconvenientes. Uno de ellos es la posibilidad de que, aunque partamos de una imagen con valores reales, los coeficientes transformados puedan ser números complejos. Además, esta transformación equivale a la construcción de una imagen que es la repetición periódica de la original, por lo que cada punto del marco de la imagen original pasa a colindar con los puntos del borde opuesto, apareciendo variaciones bruscas que se traducen en un aumento de la energía que se encuentra en las altas frecuencias. Debido a este último efecto, la DFT no es la transformada óptima para la compresión de imágenes.

7.2 Transformada del Coseno (DCT)

La transformada del coseno equivale a una DFT realizada sobre una imagen $y_a[n_1, n_2]$ de tamaño $2N \times 2N$ que se forma mediante dos reflexiones (una según un eje horizontal y otra según un eje vertical) de la imagen original:

$$\begin{aligned} DCT\{y[n_1,n_2]\} &= DFT\{y_a[n_1,n_2]\} \\ &= \frac{1}{2N}\sum_{n_1=0}^{2N-1}\sum_{n_2=0}^{2N-1} y_a[n_1,n_2]e^{-jk_1\frac{2\pi}{2N}n_1}e^{-jk_2\frac{2\pi}{2N}n_2} \end{aligned} \quad (32)$$

Si sobre esta fórmula reagrupamos y realizamos un cambio de variable obtenemos:

$$\begin{aligned} DCT\{y[n_1,n_2]\} &= \frac{2}{N}\sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1} y[n_1,n_2] \\ &\quad \cos\left[\frac{k_1\pi}{2N}(2n_1+1)\right]\cos\left[\frac{k_2\pi}{2N}(2n_2+1)\right]e^{j(k_1+k_2)\frac{\pi}{2N}} \end{aligned} \quad (33)$$

Vemos que la transformada no es real porque aparece el término $e^{j(k_1+k_2)\frac{\pi}{2N}}$. Esta exponencial compleja desaparecería si consideráramos el origen en el centro de la imagen formada $y_a[n_1, n_2]$, que realmente es el origen de la imagen original. De hecho, dado que es un término de fase que no afecta a la imagen, podemos obviarlo. Así, podemos expresar cada una de las funciones base de la forma:

$$a_{k_1,k_2}[n_1,n_2] = \frac{2}{N}\cos\left[\frac{k_1\pi}{2N}(2n_1+1)\right]\cos\left[\frac{k_2\pi}{2N}(2n_2+1)\right] \quad (34)$$

La ventaja principal de la DCT frente a la DFT es que los pixels de los bordes de la imagen original se convierten en adyacentes de sí mismos al realizar las reflexiones, eliminando los cambios abruptos que aparecían en la DFT y por tanto no aparecen nuevas componentes en frecuencias altas. A esto hay que añadir que la DCT tiene todas la ventajas de la DFT y siempre es real, lo cual la convierte en una buena opción como fase de transformación en un algoritmo de compresión. De hecho es la transformación utilizada en el conocido estándar JPEG.

7.3 Transformada de Walsh-Hadamard

Esta transformación normalmente es definida de forma matricial. Las matrices de transformación separables de distinto orden parten de la matriz de orden más bajo (2×2) que es de la forma:

$$[WHT]_1 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{35}$$

La matrices de orden superior se generan sustituyendo cada uno de la matriz $[WHT]_1$ por la matriz misma, obteniéndose por tanto:

$$\begin{aligned} [WHT]_2 &= \frac{1}{\sqrt{2}} \begin{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} & \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \\ \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} & -\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \end{pmatrix} \\ &= \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \end{aligned} \tag{36}$$

y así sucesivamente, generando matrices mayores aunque siempre de tamaño $2n \times 2n$. Tal como dijimos, cada fila de las matrices son los vectores base de la transformada. Dado que definimos la transformación mediante matrices $[A]$ los coeficientes transformados se obtiene de la forma descrita en la ecuación (30).

La transformada de Walsh-Hadamard al igual que la DCT produce un pseudo-espectro espacial de la imagen. En la DCT cada fila de la matriz de transformación es el producto de dos funciones coseno con una frecuencia ascendente. En el caso de la transformada de Walsh-Hadamard, la frecuencia de cada vector fila se refleja en los cruces por cero. Para que la frecuencia de los vectores base sea ascendente con el número de fila se suele realizar una reordenación de la matriz de la ecuación (36) de la forma:

$$[WHT]_2 = \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{pmatrix} \begin{matrix} \text{cambios de signo} \\ 0 \\ 1 \\ 2 \\ 3 \end{matrix} \tag{37}$$

De esta forma, los coeficientes transformados se relacionan con la rapidez de cambio en los vectores de datos, lo cual corresponde a la idea intuitiva de frecuencia.

Es notable la sencillez de cálculo de esta transformación dado que no se producen multiplicaciones para cada pixel, sólo hay sumas (+1) y restas (-1) entre ellos.

7.4 Transformada de Karhunen-Loewe

En esta transformación el objetivo es la máxima adaptación a un conjunto de imágenes. La matriz de transformación será la matriz de varianza-covarianza del conjunto de imágenes, específica por tanto para dicho conjunto.

La principal ventaja de esta transformación es su adaptación total a los datos, demostrándose matemáticamente que es la transformación que alcanza la máxima decorrelación posible. Sin embargo es poco viable pues la matriz de transformación cambia con cada conjunto de imágenes, y cuanto más amplio sea este conjunto menos eficaz es la transformación. Además, se trata de una transformada no separable y que no cuenta con un algoritmo para una computación rápida como en el resto de los casos.

8 Codificadores de Resolución Variable

Agrupamos bajo este epígrafe a un conjunto de métodos de compresión basados esencialmente en la generación progresiva de la imagen reconstruida a partir de versiones de distinta resolución de la imagen original. Las diferencias entre los métodos surgen de su forma de obtener dichas versiones de distintas resolución y del posterior procesado al que son sometidas.

8.1 Interpolación Jerárquica

La interpolación jerárquica (HINT) es un método piramidal basado en el submuestreo. Partiendo de una versión de baja resolución de la imagen original (S_0), obtenida submuestreando dicha imagen, genera versiones sucesivamente de más resolución utilizando interpolación. De esta forma, lo que se transmitirá o almacenará tras ser codificado con algún codificador de entropía será en primer lugar la imagen de menor resolución S_0. De ella se obtiene la versión con el siguiente nivel de resolución situando nuevos pixels entre los anteriores. Para hallar el valor de estos nuevos pixels se utiliza algún método de interpolación aplicada sobre los pixels ya conocidos (interpolación lineal, exacta, etc). En transmisión esta operación también se realiza y el valor calculado es sustraído al valor del pixel correspondiente en la imagen original y es la diferencia obtenida la que se transmitirá para la reconstrucción final de la imagen. Se trata por tanto de un método concebido en principio para una compresión sin pérdidas.

8.2 Pirámide Diferencia

Este método de resolución variable se basa en la construcción de una pirámide de medias y el cálculo de una pirámide diferencia que contiene las diferencias entre los niveles de la pirámide de medias.

El algoritmo parte de la división de la imagen original, que será la base de la pirámide de medias, en grupos de cuatro pixels sobre los que se halla la media redondeada al entero más cercano, conformando estos valores medios el siguiente nivel de la pirámide, de tamaño cuatro veces menor. Este proceso

se repite de nuevo para cada nivel progresivamente de menor resolución. Paralelamente se forma la pirámide diferencia siendo igual en cada punto de cada nivel a la diferencia que se obtiene al sustraer a cada uno de los pixels de cada grupo del mismo nivel de la pirámide de medias el valor medio correspondiente, recogido en el siguiente nivel de la pirámides de media. De esta forma, la imagen original se reconstruirá a partir del vértice de la pirámide de medias y de la pirámide diferencia. Al vértice de la pirámide de medias, que es un sólo valor, se le suma cada uno de los cuatro valores del último nivel de la pirámide diferencia, obteniéndose los cuatro valores del penúltimo nivel de la pirámide de medias. El proceso se repite con cada uno de estos cuatro valores y los 16 valores del siguiente nivel de la pirámide diferencia, obteniéndose los 16 valores del antepenúltimo nivel de la pirámide de medias, y así sucesivamente hasta recuperar la imagen original. De nuevo estamos ante un método sin pérdidas.

De este método existen variantes como la pirámide diferencia reducida (RDP) o la Transformada-S.

8.3 Codificación por Plano de Bits

La codificación por planos de bits se basa en la separación de la imagen original con p bits por pixel en p imágenes de 1 bit por pixel. Partiendo de la representación en binario de los valores de los pixels, seleccionamos un sólo bit de la misma posición para todos los pixels. De esta manera formaremos una nueva imagen de tamaño $M \times N$ con la particularidad de que es binaria, es decir, todos sus pixels toman el valor 1 ó 0. Esta imagen se denomina plano de bits. Por ejemplo, el plano de bits más significativo es una imagen $M \times N$ que contiene los bits más significativos de los valores de los pixels en la imagen original. Repitiendo este proceso para el resto de las posiciones de los bits, se llega a la descomposición de la imagen en los p planos de bits, imágenes binarias de tamaño $M \times N$. En las figura 13 tenemos un ejemplo de esta descomposición. La primera imagen corresponde a la radiografía original y las siguientes son sus 8 planos de bits, del más significativo al menos.

Como podemos observar, los planos de bits más significativos (MSB) contienen la información estructural y a medida que los bits son menos significativos la naturaleza del plano es cada vez más ruidosa, añadiendo menos información para la reconstrucción de la imagen. Sin embargo, estos últimos planos no pueden obviarse porque aparecerían falsos bordes.

La descomposición en planos de bits facilita la transmisión progresiva. Cada plano, de más a menos significativo, se codifica y transmite secuencialmente. En recepción, la imagen reconstruida inicialmente es una imagen binaria, el plano MSB, pero a medida que se reciben más planos se van añadiendo más niveles de grises. Este método de compresión es en un principio sin pérdidas, pero la reconstrucción puede detenerse en cualquier plano, obteniendo tasas de compresión notablemente mayores.

Para la codificación se aprovecha la existencia de grandes áreas uniformes en los planos de bits más significativos. El método típicamente utilizado para la compresión de datos con grupos amplios de ceros y unos consecutivos es la codificación runlength o RLE, utilizada ampliamente en la codificación de

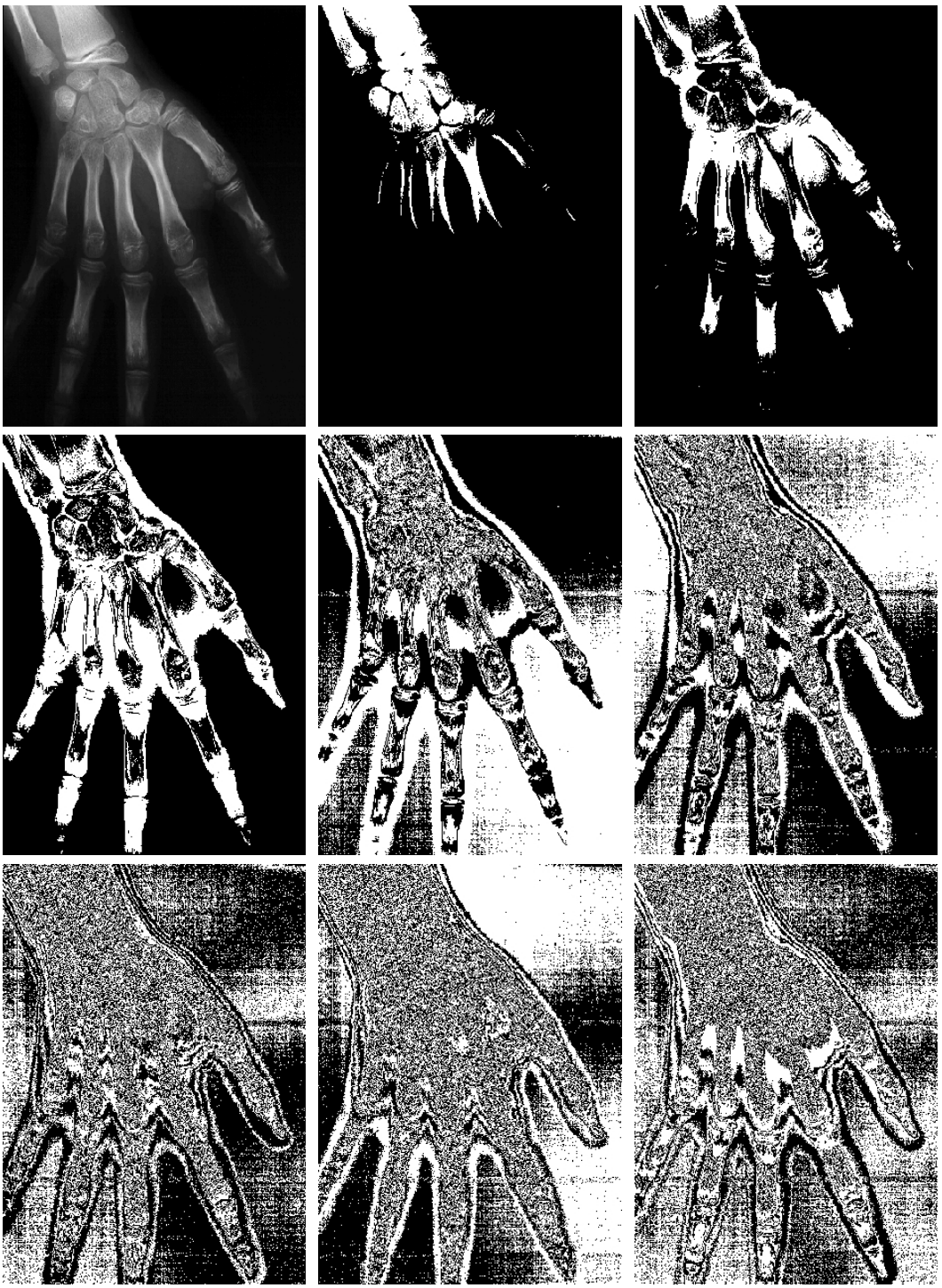

Figura 13: Ejemplo de descomposición de una imagen en sus planos de bits.

documentos en el campo de los facsímiles. Básicamente consiste en codificar sólo el primer bit y las longitudes en sistema binario de los grupos de unos y ceros, formando una serie de símbolos RLE. La naturaleza de la ristra de bits que cada símbolo codifica se establece mediante una regla de alternancia, es decir, tras una ristra de unos siempre se codifica una ristra de ceros y viceversa. Sin embargo, la codificación se complica por la limitación del número representable de componentes de cada ristra al establecer un número de bits por símbolo. No podemos codificar con un determinado número de bits una ristra de un número ilimitado de componentes al no ser posible la traducción a binario de su longitud utilizando sólo dichos bits. Es por esto que algunos grupos especialmente largos deberán ser codificados en dos o más símbolos consecutivos, rompiéndose la regla de la alternancia de ristras de unos y ceros. Existirá entonces un símbolo especial que indique por un lado la codificación de una ristra de tamaño igual al máximo permitido y por otro que el símbolo siguiente corresponde a otra ristra del mismo tipo, rompiéndose la alternancia.

La codificación RLE pierde gran parte de su utilidad en los planos menos significativos ya que, tal como se observa en la figura 13 la áreas uniformes van haciéndose cada vez más pequeñas, tendiendo a una estructura ruidosa. Estos planos menos significativos deberán codificarse de forma distinta o, debido a que no aportan información estructural, eliminarse, aunque eso implicaría que el método de codificación dejaría de ser sin pérdidas. Además, a la imagen reconstruida habría que añadirle ruido aleatorio para evitar la aparición de falsos bordes, tal como dijimos. La codificación RLE por sí sola no es por tanto adecuada para estos últimos planos. Sin embargo, gracias a la codificación de longitud variable que se aplicará sobre los símbolos RLE se consigue un tamaño medio de código pequeño, de tal manera que, en conjunto, se logra una cierta compresión, aunque mínima, sobre los planos menos significativos.

Como decíamos, tras la codificación RLE tenemos una serie de símbolos en los que hemos codificado cada uno de los planos de bits de más a menos significativo de tal forma que la reconstrucción será multirresolutiva. A cada uno de estos símbolos RLE es conveniente asignarles un código adecuado para su transmisión. El método más sencillo sería asignar a cada símbolo una palabra de longitud fija, pero este método es muy ineficiente. El conocimiento de las probabilidades de cada símbolo RLE nos permite alcanzar codificaciones de canal más eficientes, las denominadas codificaciones de longitud variable o codificaciones de entropía. La codificación Huffman es la codificación de entropía más utilizada.

Para la aplicación de la codificación de entropía es necesaria una fase de adquisición de experiencia en la que obtengamos las probabilidades de cada símbolo RLE. Estas probabilidades se aproximan realizando un muestreo sobre un universo de imágenes de entrenamiento con las características del tipo de imagen objetivo, aplicándoles RLE y promediando la frecuencia de aparición de cada símbolo.

Un ejemplo de las distintas versiones que se generan durante la reconstrucción progresiva que se lleva a cabo se muestra en la figura 14. En ella vamos observando que la imagen reconstruida tiene cada vez más resolución, pues está formada por un número sucesivamente mayor de planos. En el punto

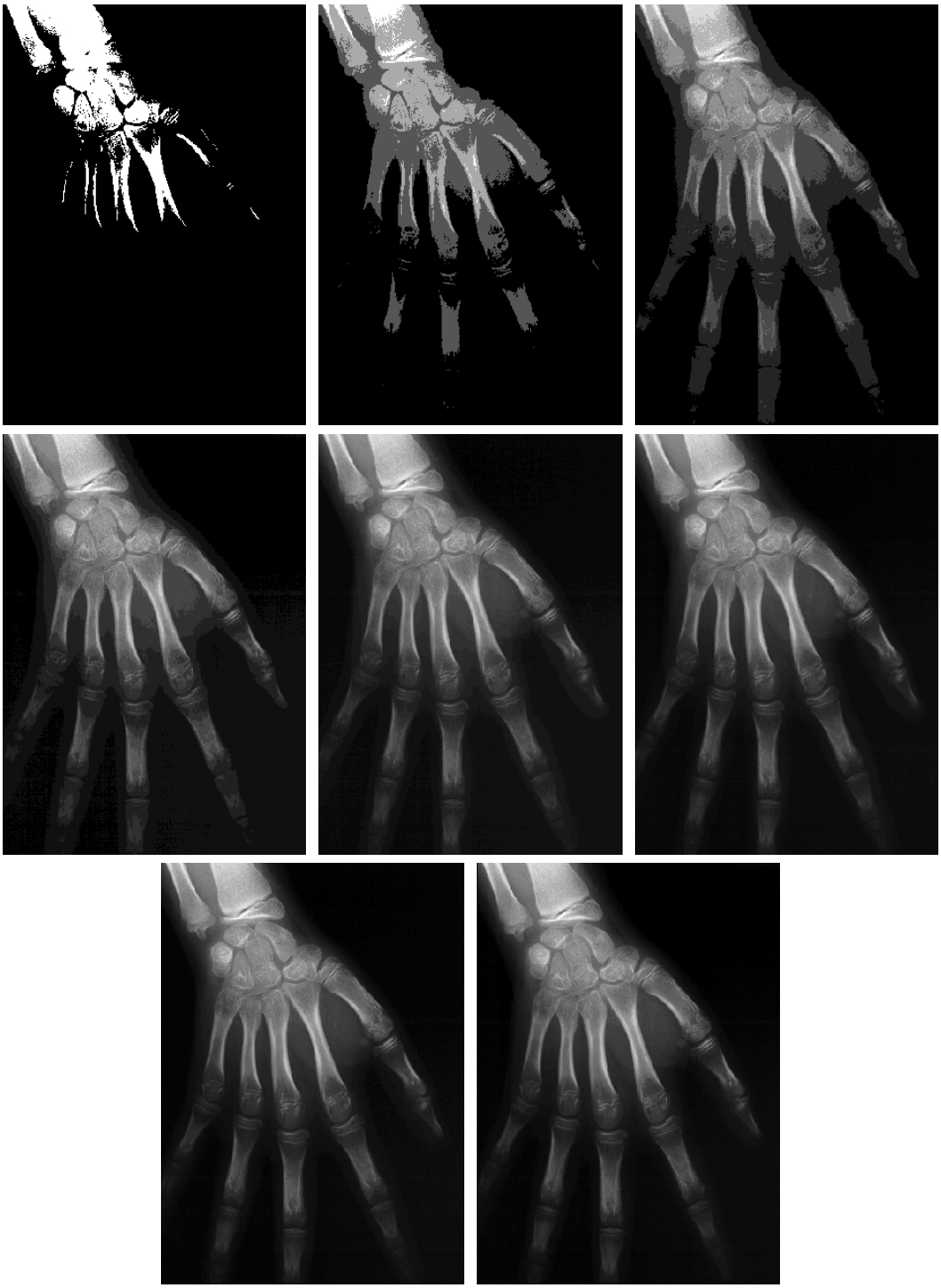

Figura 14: Ejemplo de reconstrucción progresiva de una imagen codificada. Cada radiografía tiene un plano más de resolución.

	SNR (dB)	Tasa de compresión
1 Plano	3.190	76.211 : 1
2 Planos	7.097	25.056 : 1
3 Planos	11.510	11.889 : 1
4 Planos	17.556	6.432 : 1
5 Planos	24.317	4.007 : 1
6 Planos	30.378	2.756 : 1
7 Planos	39.072	2.125 : 1
8 Planos	∞	1.681 : 1

Tabla 3: Resultados de la codificación por planos de bits.

en que se reciben los 5 primeros planos, la imagen reconstruida apenas tiene pérdidas perceptibles. La imagen reconstruida con 7 planos es visualmente sin pérdidas. Esto es debido a la limitación del ojo humano, que sólo percibe un número de niveles de gris aproximadamente igual a 100. Con 7 bits se tienen ya 128 niveles, con lo que la escisión de cada uno de ellos en otros dos, al añadir el último plano, no será percibida por un receptor humano. En el cuadro 3 se recogen las tasas de compresión y la calidad lograda para las imágenes de la figura 14.

9 Codificación por Subbandas y Wavelets

Los codificadores por subbandas y los basados en wavelets aprovechan la existencia de diferentes densidades de energía en cada frecuencia espacial, aplicando el hecho de que para una codificación más adecuada es conveniente procesar cada intervalo de frecuencias de forma distinta.

9.1 Codificación por Subbandas

En los codificadores por subbandas el proceso de codificación comienza con un banco de M filtros paso-banda que se aplica a la imagen original (M suele ser una potencia de dos). Esto podría parecer un aumento considerable de la información, pues de una señal pasamos a M señales. Sin embargo, por los principios del muestreo, cada señal de ancho de banda M veces menor extraída de cada filtro puede ser diezmada por un factor igual a dicho M sin que se produzca ningún tipo de pérdidas, de tal forma que el conjunto global de información a codificar permanece inalterado, no aumenta. El banco de filtros que descompone la señal debe satisfacer ciertos requerimientos de error de reconstrucción, retraso del procesado e interferencias entre bandas.

Para ver el proceso global gráficamente utilizamos el ejemplo más sencillo, consistente en una separación entre bajas y altas frecuencias. En la figura 15

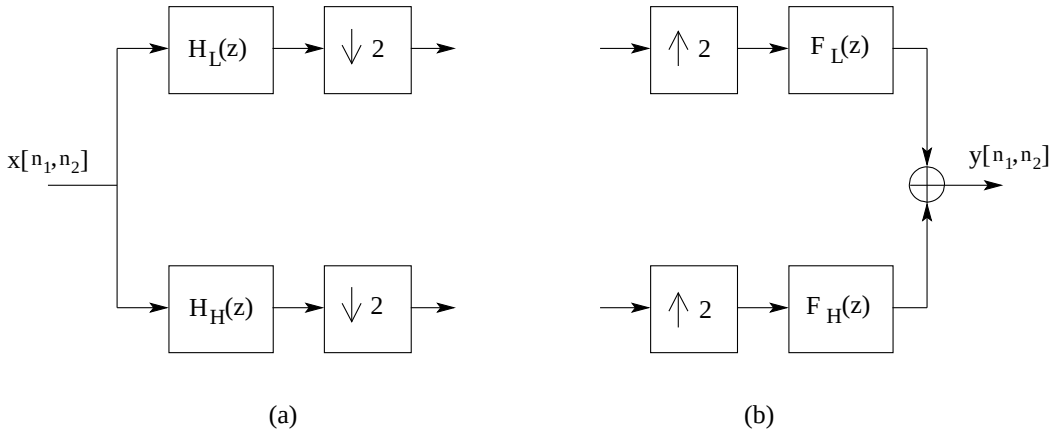

Figura 15: Ejemplo sencillo ($M = 2$) de codificación por subbandas. (a) Análisis. (b) Síntesis.

tenemos una visión del proceso completo.

La mayor parte de la energía en las imágenes reales se encuentra en las bajas frecuencias, de tal forma que el cuantificador asignado a esta banda contará con un número mayor de niveles que el asignado a frecuencias altas. Por otro lado no podemos obviar la energía a altas frecuencias pues en ellas se recoge la información de bordes. Esta sería la principal base de la asignación de bits a cada subbanda. Normalmente se lleva a cabo, para el grupo de imágenes a comprimir, un estudio más detallado de su distribución espectral, deduciendo las características más adecuadas para los cuantificadores de cada banda de frecuencia.

Es importante reseñar que en nuestro caso el espectro espacial de una imagen es bidimensional, mientras que la explicación y el diagrama dado es unidimensional. Para dos dimensiones, la descomposición se complica pues no dividimos un rango de frecuencias en intervalos, sino que dividimos un área de frecuencias bidimensionales, normalmente rectangular, en áreas más pequeñas, normalmente cuadrantes, de tal forma que el filtrado se deberá llevar a cabo para las dos componentes de las frecuencias.

9.2 Codificación por Wavelets.

La utilización de wavelets para la codificación de imagen no dista demasiado de la división por subbandas, pero dicha división se realiza de forma más conveniente.

La razón de la aparición de las transformadas wavelet surge de las limitaciones en tiempo o frecuencia de las señales reales. Debido a la interdependencia temporal (o espacial) y frecuencial de las señales es imposible tener una señal continua limitada temporalmente, dado que esto implicaría un espectro infinito. Sin embargo, en la digitalización, se registra una función continua a través de una ventana limitada, lo cual se traduce, en el dominio frecuencial, en una convolución con la transformada de esta ventana y la consecuente expansión del espectro. En general, no es posible lograr de forma independiente

exactitudes arbitrarias en el tiempo y en la frecuencia simultáneamente, sino que un aumento de exactitud en un dominio disminuirá la exactitud en el otro.

Partiendo de este hecho, la transformación mediante wavelets se realiza a partir de una función $\Psi(x)$ apropiada para ser escalada y desplazada de tal forma que, al aplicar la convolución de la función temporal o espacial con ella, podamos centrar nuestro análisis en cierta área rectangular dentro del espacio bidimensional que forman los ejes de la variable independiente x (ya sea espacio o tiempo) y su equivalente frecuencial f. Dicho rectángulo satisfará la necesidades específicas del proceso global que utiliza este tipo de transformaciones. Matemáticamente queda reflejado en la siguiente ecuación, aplicando la transformación a la función $f(x)$:

$$Wf(s,u) =< f(x), \Psi_s(x-u) > \tag{38}$$

donde

$$\Psi_s(x) = \sqrt{s}\Psi(sx) \tag{39}$$

Según modifiquemos los parámetros s y u, variaremos la forma y la posición de la ventana rectangular en el plano formado por lo ejes de tiempo (o espacio) y frecuencia. El producto base (intervalo en x) por altura (intervalo en f) de este rectángulo se mantendrá constante, pues depende de la función wavelet aplicada en la transformación, es decir, una expansión en tiempo supondrá una compresión en frecuencia y viceversa.

En las aplicaciones para la compresión de imágenes materializamos toda esta teoría en una descomposición de la imagen de entrada en versiones de distinta resolución al aplicarle una familia de transformadas wavelets tal como la familia ortonormal y discreta de bases wavelet de Meyer, teniendo en cuenta que los parámetros de escalado (s) y de translación (u) no deben tomar cualquier valor arbitrario. Proyectar la función según el diferente escalado y translación de la wavelet es equivalente a filtrarla con un banco de filtros. De esta forma conseguimos separar la función en subbandas manteniéndose la mejor relación posible entre la precisión espacial y la frecuencial, dadas las características de las transformadas wavelets.

Análogamente al proceso seguido en la codificación por subbandas, se realiza una descomposición en ambas direcciones (una versión separable en dos dimensiones de la transformación unidimensional descrita), separando las bajas frecuencias de las altas. Este filtrado se repetirá el número de veces necesario para alcanzar el número final de subbandas establecido, obteniéndose nuevas imágenes de distintas características, cada una de las cuales será tratada de forma diferente, pero manteniéndose el número total de elementos. De hecho, la codificación mediante wavelets no es sino un caso particular de codificación por subbandas.

10 Codificadores Basados en Modelos

En la codificación basada en modelos, la imagen o una parte de ésta se representa según un modelo y se emplean los parámetros del modelo para sintetizar la imagen posteriormente. En el extremo transmisor se estiman los parámetros

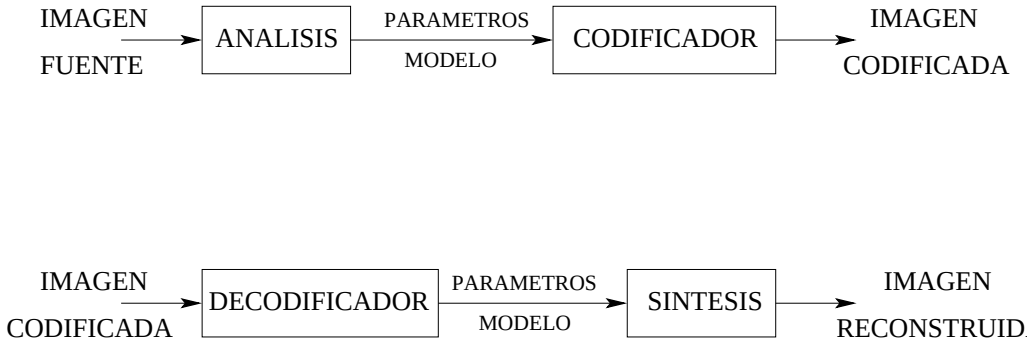

Figura 16: Esquema de codificación basada en modelos.

del modelo analizando la imagen y en el extremo receptor se sintetiza la imagen a partir de los parámetros del modelo estimados y cuantificados. En la figura 16 se puede ver esquemáticamente el transmisor y el receptor de este tipo de codificación. Se puede ver como un proceso de análisis y síntesis. Los codificadores basados en modelos logran sintetizar imágenes empleando una tasa binaria considerablemente menor que en los codificadores de forma de onda o de transformada. Sin embargo, están todavía en desarrollo y es necesario seguir en esta línea durante algún tiempo para que sean viables en la práctica. Desarrollar un modelo sencillo de imagen a partir del cual se pueda sintetizar una imagen con significado es un problema muy complicado. Además, estimar los parámetros del modelo y sintetizar la imagen a partir de estos parámetros es computacionalmente muy caro.

La codificación basada en modelos tiene su razón de ser en la noción de que para sintetizar imágenes con significado no es necesario reproducir de forma detallada las intensidades de la imagen. Por ejemplo, el fondo de la imagen, como por ejemplo hierba, el cielo, una pared, etc. no son esenciales en la inteligibilidad de la imagen de forma que se pueden reemplazar por otros fondos con características similares que se puedan sintetizar con un modelo sencillo. En un modelo de imagen, se pretende mantener las características esenciales de la imagen y aproximar de forma menos exacta las características no esenciales utilizando para ello modelos sencillos. Este método contrasta enormemente con las codificaciones de forma de onda y de transformada para las que se pretende reconstruir la imagen de intensidad. En estos casos la diferencia entre la imagen original y la reconstruida depende de la cuantificación de los parámetros. Si esta cuantificación no se lleva a cabo la imagen reconstruida es exactamente igual a la original. En la codificación basada en modelos, la diferencia entre la imagen sintetizada y la original es debida a la cuantificación de los parámetros y al error en el modelo. No es posible reconstruir la imagen original incluso aunque los parámetros del modelo no se cuantificaran. El número de parámetros para la codificación basada en modelos es mucho menor que el caso de codificación de forma de onda o codificación de transformada. De esta forma se logra una tasa tan baja en la imagen codificada.

Una imagen típica consiste en varias regiones con diferentes características. Será conveniente modelar cada una de las clases de regiones con modelos difer-

entes. Regiones tales como hierba, agua, cielo, pared, etc. tienen propiedades de orden o estructuras repetitivas denominadas textura. Hay dos técnicas básicas para modelar texturas:

- Utilizar un patrón elemental básico y repetirle de acuerdo a una regla determinística o probabilística.
- Modelar la textura como un campo aleatorio con ciertas propiedades estadísticas. Dos texturas con estadística de segundo orden similar aparecen como texturas muy similares para el sistema visual humano. Se han desarrollado muchos modelos de campos aleatorios de segundo orden para modelar texturas.

Para poder utilizar este método es necesario emplear una técnica que permita segmentar la imagen en regiones con textura similar para poder modelar cada una de ellas por separado con un modelo diferente. Sin embargo, no es posible modelar adecuadamente regiones de la imagen que contienen o forman parte de objetos como texturas puras. Para modelar estas regiones podemos utilizar la idea de que un conjunto reducido de contornos situados adecuadamente pueden representar la mayor parte de la inteligibilidad presente en estos objetos. Así se puede representar una región de la imagen con un conjunto de contornos y, si se desea, rellenar las regiones entre estos contornos empleando las texturas adecuadas.

11 Compresión de Imágenes Médicas

Hoy en día el desarrollo de nuevas tecnologías para el tratamiento, almacenamiento y transmisión de imágenes médicas en formato digital está siendo muy estudiado ya que han surgido problemas legales con respecto a los codificadores que actualmente están en el mercado. Es evidente que el codificador ideal para este tipo de imágenes ha de ser sin pérdidas, debido a la importancia de la calidad de la imagen decodificada a la hora de hacer un diagnóstico. Sin embargo, existen codificadores con pérdidas que logran tasas de compresión bastante elevadas sin sacrificar mucho la calidad de la imagen. A estos últimos se les denomina visualmente sin pérdidas, ya que no presentan pérdidas visuales bajo condiciones normales de observación.

El estudio sobre este tipo de técnicas está chocando actualmente con todo tipo de problemas legales que se plantean para su implementación. Evidentemente, los compresores sin pérdidas no tienen problemas legales, pero ofrecen una modesta tasa de compresión de 2:1 aproximadamente. Por esto, se está llevando a cabo un gran esfuerzo de investigación centrado en técnicas de compresión con pérdidas de baja tasa binaria (compresiones de 10:1), que descarten la información de poca relevancia para el diagnóstico. El estándar DICOM está trabajando en esta línea. La política reguladora en esta materia se ocupa menos de las patentes y de los estándares que de los dispositivos médicos que ofrecen software de compresión.

Actualmente, los radiólogos son muy escépticos respecto al uso de compresión con pérdidas para diagnóstico primario debido a la posible pérdida de

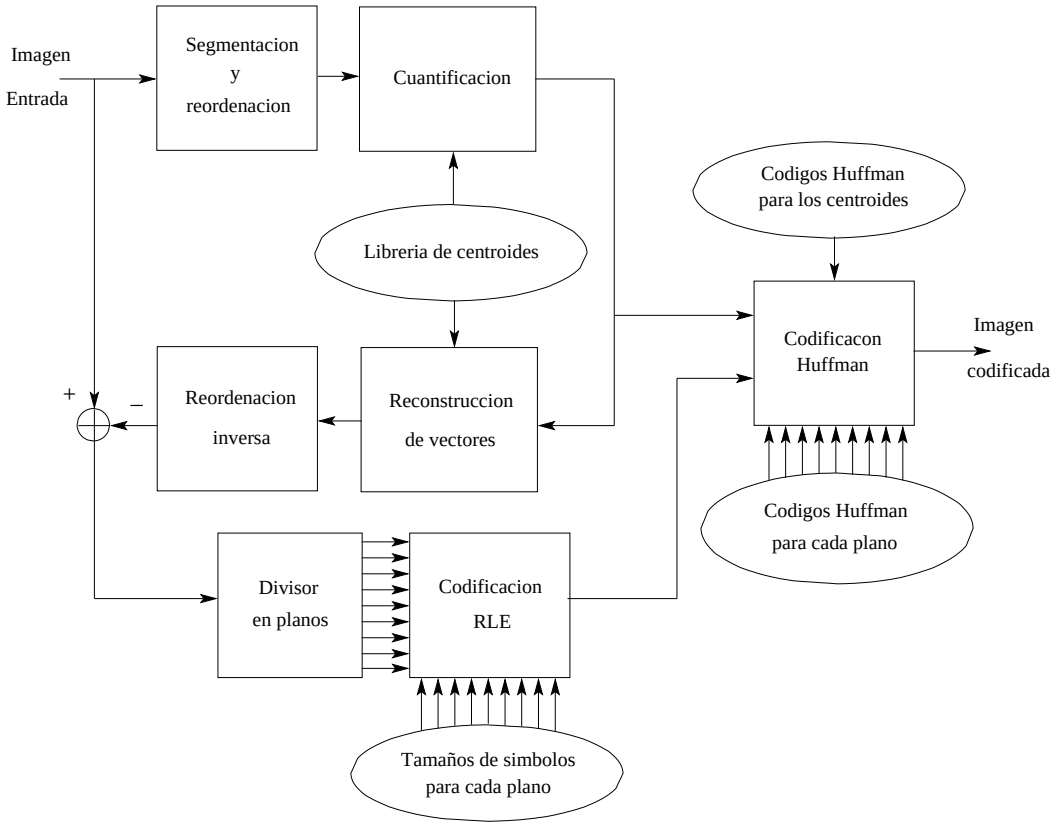

Figura 17: Diagrama de bloques del codificador del sistema multirresolutivo.

detalles importantes. Sin embargo, estos algoritmos si que se están empleando para almacenamiento de imágenes ya diagnosticadas. Una línea de estudio interesante es catalogar el grado de compresión necesario para los diferentes tipos de aplicaciones radiológicas.

Actualmente, no existe ningún estándar legal de compresión de imágenes médicas. Esto significa que no hay ninguna referencia clínica para un tribunal a la hora de juzgar un caso en el que intervenga un dispositivo que use un método de compresión con pérdidas. Se traslada la responsabilidad a los usuarios de estos dispositivos y no a los fabricantes. Los usuarios deben conocer las especificaciones de los aparatos y escoger aquellos que les interesa.

Un estándar sería muy importante sobre todo en la transmisión de imágenes a distancia debido al desconocimiento de los dispositivos médicos del otro extremo de la línea de transmisión. En cualquier caso no es lógico que este estándar dependa de la decisión de un tribunal que cree jurisprudencia en todo este asunto, sino que esta tarea se debería llevar a cabo por un profesional de la medicina y del tratamiento de imagen. Esto puede requerir un gran esfuerzo pero evitará problemas en el futuro.

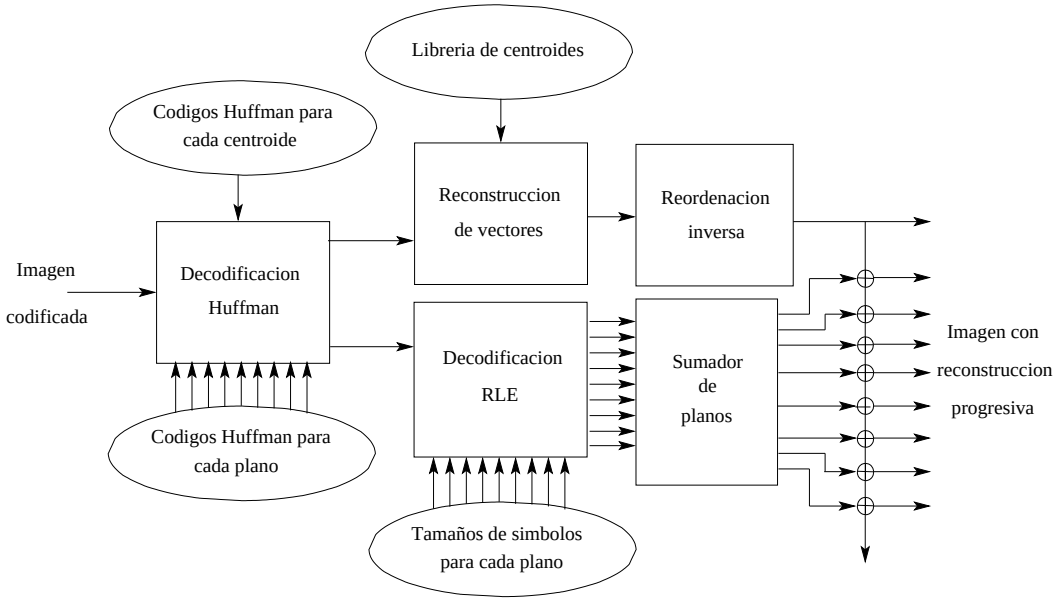

Figura 18: Diagrama de bloques del decodificador del sistema multirresolutivo.

12 Implementación de un Sistema de Codificación Multirresolución

La fusión de un algoritmo sin pérdidas (la codificación por planos de bits) y un algoritmo con pérdidas (la cuantificación vectorial) se basa en la idea de la estratificación de la codificación de la imagen. En una primera fase se realiza la codificación con pérdidas, obteniéndose una imagen reconstruida con unas determinada características de distorsión. Para las siguientes fases es necesaria la extracción en el lado del codificador de la imagen diferencia o imagen error que aparece entre la versión original y la reconstruida de la imagen de entrada. Es esta imagen diferencia la que se somete a las sucesivas etapas de la codificación sin pérdidas, durante la que se generan una serie de símbolos que codificados son añadidos a la trama de bits que constituirá la imagen codificada. En dicha trama se unifican por tanto las distintas fases de la codificación. En la figura 17 puede observarse en el diagrama de bloques para el codificador y en la figura 18 para el decodificador. En la figura 19 podemos ver un ejemplo de la reconstrucción de la imagen a partir de la codificación con cuantificación vectorial y los planos de bits de la imagen diferencia. En la figura 20 se puede ver la imagen diferencia entre la imagen original y la imagen cuantificada y en la figura 21 podemos ver la imagen diferencia representada como el plano de signo y sus 8 planos de bits. En la tabla 4 se recogen los resultados de las tasas de compresión y la calidad lograda para cada imagen de la figura 19.

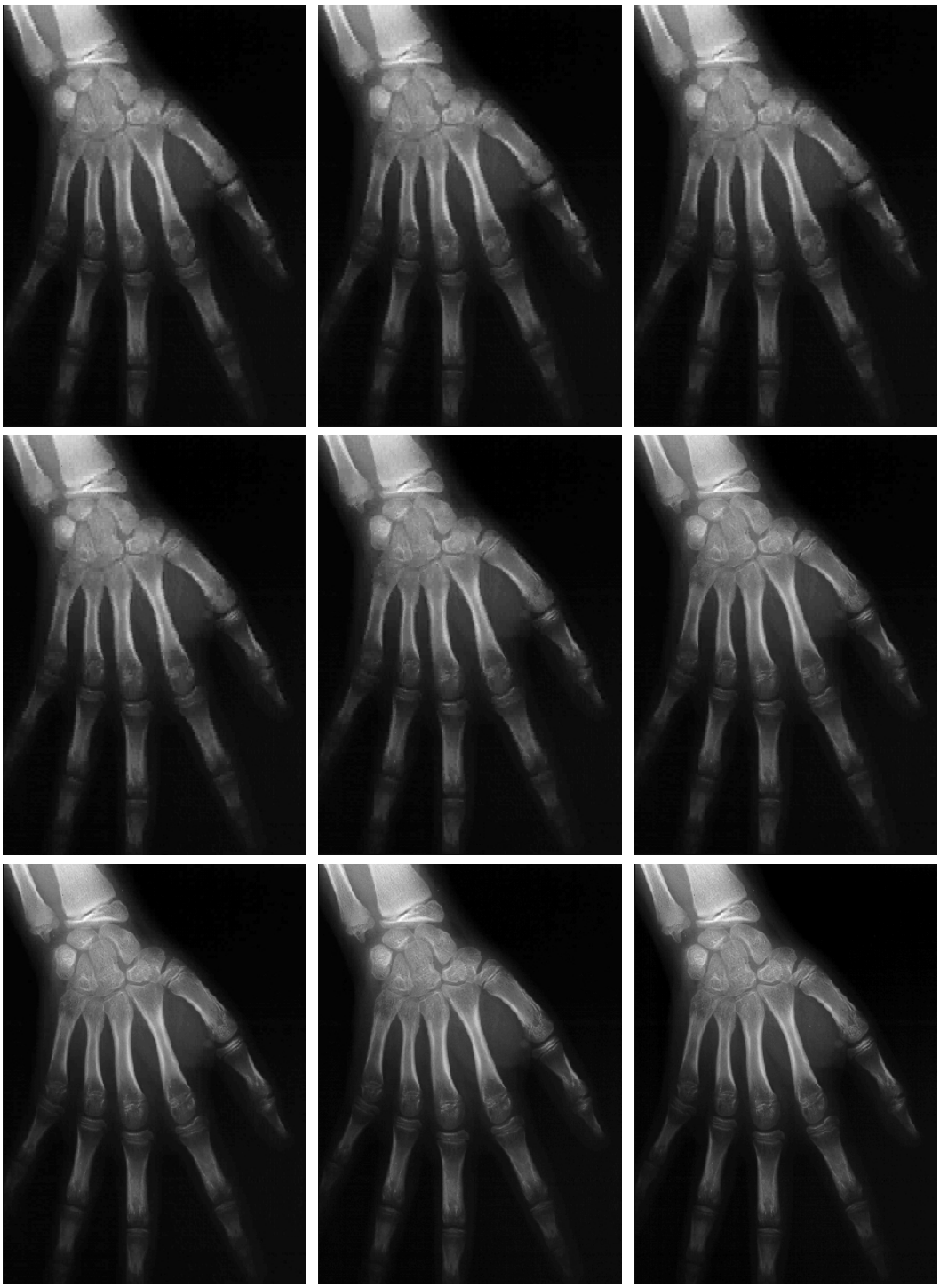

Figura 19: Ejemplo de reconstrucción de una imagen codificada.

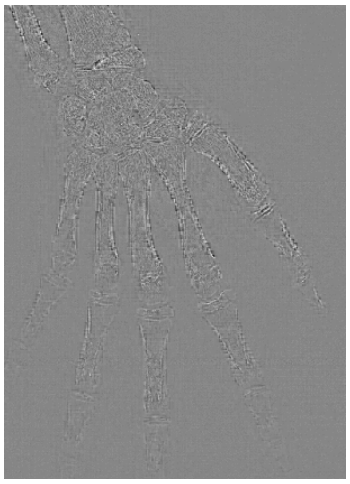

Figura 20: Imagen diferencia entre la imagen cuantificada y la original.

	SNR (dB)	Tasa de compresión
Imagen cuantificada	22.857	17.433 : 1
Imagen cuantificada + 1 Plano	22.857	5.785 : 1
Imagen cuantificada + 2 Planos	22.863	5.739 : 1
Imagen cuantificada + 3 Planos	23.248	5.682 : 1
Imagen cuantificada + 4 Planos	25.069	5.229 : 1
Imagen cuantificada + 5 Planos	28.472	4.381 : 1
Imagen cuantificada + 6 Planos	32.477	3.447 : 1
Imagen cuantificada + 7 Planos	39.650	2.441 : 1
Imagen cuantificada + 8 Planos	∞	1.878 : 1

Tabla 4: Resultados de la codificación.

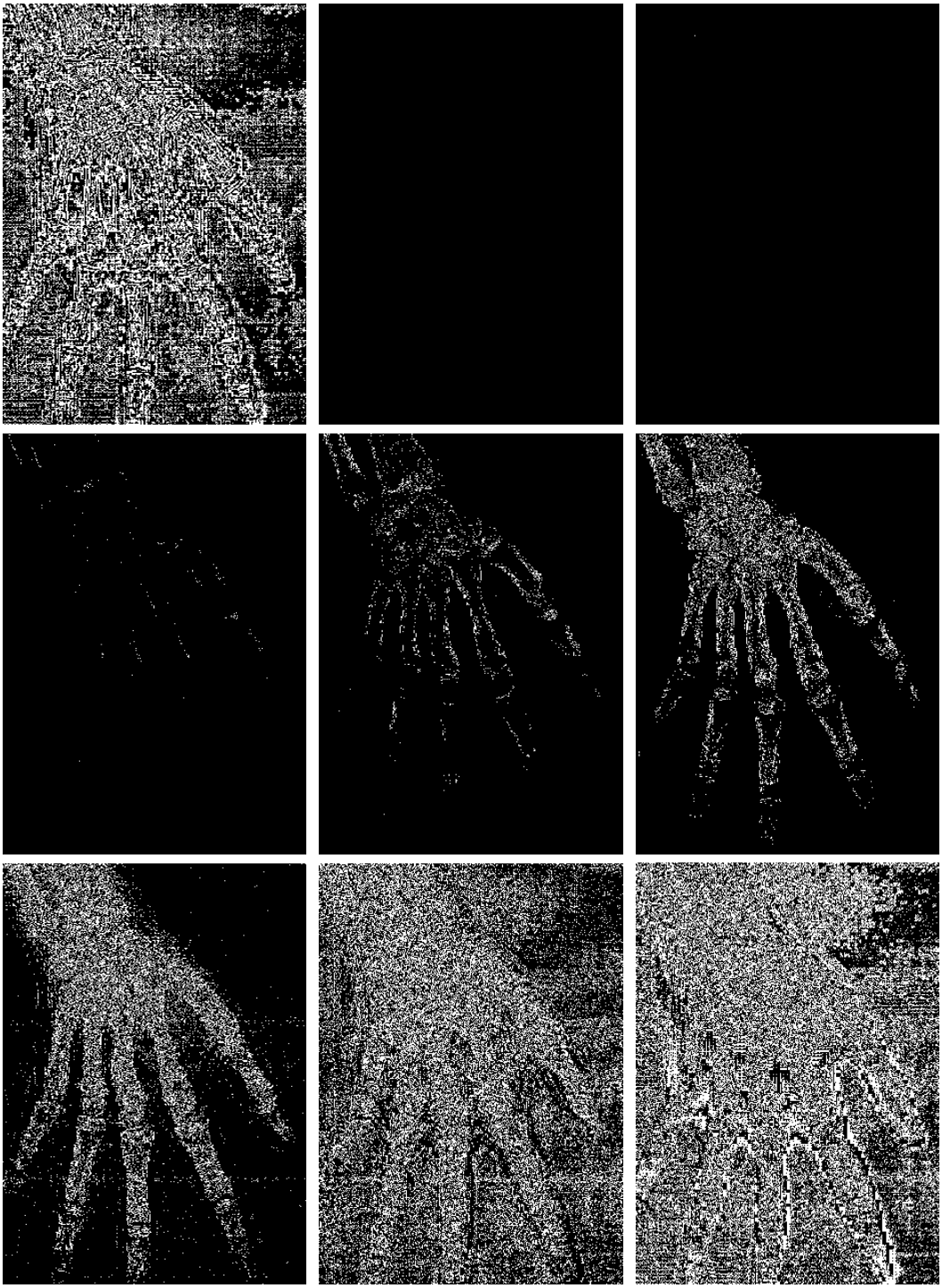

Figura 21: Ejemplo de descomposición de una imagen diferencia en sus planos de bits (plano de signo y planos del 7 al 0).