Universidad Carlos III de Madrid

Ingeniería Técnica de Telecomunicación: Imagen y Sonido

PROYECTO FIN DE CARRERA

DISEÑO DE UN CODIFICADOR DE IMÁGENES ADAPTATIVO MULTITRANSFORMADA MEDIANTE EL USO DE LA TRANSFORMADA KARHUNEN-LOÈVE

Autor: Valentín Cruz Rodríguez
Tutor: Manuel de Frutos López

Octubre 2012

DEPARTAMENTO DE TEORÍA DE LA SEÑAL Y COMUNICACIONES

ÍNDICE DE CONTENIDOS

LISTA DE FIGURAS

LISTA DE TABLAS

CAPÍTULO 1 - INTRODUCCIÓN Y OBJETIVOS

1.1 INTRODUCCIÓN

El desarrollo de este proyecto se da en un contexto de codificación adaptativa de imágenes mediante transformada. A continuación se introducen los conceptos principales de la codificación de imágenes.

Las capacidades de almacenamiento y sobre todo la de transmisión de la información que poseen los sistemas son limitadas, sin embargo, los datos multimedia suponen volúmenes de información importante. Por este motivo, antes del almacenamiento o de la transmisión de datos multimedia, prácticamente se hace imprescindible el empleo de técnicas de compresión que permitan reducir el volumen de dicha información.

A grandes rasgos, los tipos de compresión se pueden clasificar en dos: sin pérdidas y con pérdidas. La principal diferencia entre ambos, es que a partir de los datos codificados sin pérdidas es posible recuperar de forma exacta los datos originales, mientras que a partir de los datos codificados con pérdidas se recupera una aproximación, cuyo error con respecto a los originales dependerá de la información eliminada. Ambos tipos de compresión tienen características contrapuestas: en el caso de sin pérdidas, la compresión de los datos no conlleva pérdida de calidad, lo que limita en gran medida la tasa de compresión ya que se conserva toda la información necesaria para recuperar los datos originales; sin embargo, la compresión con pérdidas es capaz de obtener mayores tasas de compresión ya que elimina mayor cantidad de información, a costa de que se degrade la calidad. La elección de un codificador u otro, dependerá del escenario y de las distintas necesidades.

En la compresión multimedia nos encontramos con dos clases de técnicas de codificación: las basadas en la entropía y las basadas en la fuente. Las técnicas de codificación basadas en la entropía, son técnicas de compresión sin pérdidas, las cuales comprimen los datos sin tener en cuenta la naturaleza de los mismos, por lo tanto se pueden aplicar a cualquier tipo de dato. Tratan a los datos como un mensaje compuesto por símbolos pertenecientes a un alfabeto, el cual está formado por un conjunto finito de símbolos distintos. La compresión se aplica basándose en las probabilidades de aparición de los símbolos, asignando a los símbolos más frecuentes las cadenas binarias más cortas y a los menos frecuentes las más largas. Por otro lado, las técnicas de codificación basadas en la fuente, están diseñadas teniendo en cuenta las propiedades de la fuente de datos a codificar, por lo tanto, son técnicas aplicadas a determinados tipos de datos. Existen tanto sin pérdidas como con pérdidas y gracias a que se diseñan según las cualidades de los datos, por término general obtienen mayores prestaciones que las

técnicas de codificación basadas en la entropía, sobre todo las técnicas con pérdidas cuyos errores pasan inadvertidos perceptualmente para el usuario.

Una cualidad de los datos audiovisuales es la similitud que presentan muestras vecinas. En audio, la frecuencia que se utiliza para muestrear las señales es muy alta, por lo que el intervalo de tiempo que transcurre entre dos muestras es muy pequeño, provocando que los valores de dos muestras consecutivas difieran en muy poco. Por otro lado, en imágenes, los píxeles que forman los distintos objetos que aparecen tienen valores parecidos; en este caso, si nos centramos en un píxel, el valor de dicho píxel y el de sus vecinos en cualquier dirección serán parecidos. Por último, en video, al estar formado por una sucesión de planos nos encontramos con una doble redundancia ya que, por un lado, en cada uno de los planos existe la misma redundancia espacial que en las imágenes, pero por otro, dado que la variación entre planos consecutivos suele ser pequeña, los píxeles correspondientes a una misma posición en planos consecutivos tomarán por regla general valores similares, existiendo por tanto también cierta redundancia temporal.

Algunas técnicas de compresión basadas en la fuente utilizadas en imágenes son: la codificación diferencial, la codificación predictiva y la codificación por transformada. La codificación diferencial lo que hace es codificar la diferencia entre dos muestras consecutivas; si dichas muestras poseen valores similares, su diferencia tendrá un valor mucho más reducido y por lo tanto, necesitará menos bits para codificarse. Por otro lado, la codificación predictiva se basa en predecir el valor de un píxel utilizando sus píxeles vecinos, para una vez calculada dicha predicción, restársela al valor original y codificar el error de predicción obtenido en lugar del valor original del píxel; cuanto más parecidos sean los píxeles vecinos mejor será la predicción, por lo que el error de predicción disminuirá y se necesitarán menos bits para su codificación. Por último, la codificación mediante transformada aprovecha las cualidades que poseen algunas transformadas lineales, para representar datos cuya dependencia estadística es muy alta, en otro dominio en el que dicha dependencia es mucho menor. Las transformadas de imagen se pueden agrupar según la forma de aplicarse en dos categorías: por un lado están las transformadas de bloque que son las que se aplican a subdivisiones de la imagen original (como la DCT) y por otro están las transformadas de imagen que son las que se aplican a la imagen completa (como la transformada Wavelet). Las que tienen mayor interés para nosotros son las transformadas de bloque, ya que el procesado por bloques está ampliamente extendido tanto en codificación de imágenes como de vídeo y son las que se utilizan en este proyecto.

En imágenes, los píxeles próximos entre sí tienen una gran dependencia estadística, pero según aumenta la distancia entre píxeles, dicha dependencia disminuye. Por ello, los codificadores por transformada de bloque dividen la imagen en bloques cuadrados de un tamaño no demasiado grande, para que los píxeles contenidos en cada bloque mantengan una dependencia estadística alta y aplican a cada uno de los bloques una transformada lineal (normalmente la misma) obteniendo, a partir de cada uno, un

bloque de coeficientes transformados en el que la dependencia estadística de los coeficientes es mucho menor que la que existía entre los píxeles del bloque original.

La transformada por sí sola no aplica compresión a los píxeles, pero sí hace posible que se consiga en procesos posteriores; la transformada, al decorrelar los datos, lo que provoca es una compactación de la energía en el dominio transformado, concentrando la mayoría de la energía en unos cuantos coeficientes y resultando el resto con muy poca energía. Con esta nueva distribución energética, se pueden identificar los coeficientes que aportan más información a la imagen (aquellos en los que se ha compactado la energía) y los que aportan menos (los energéticamente poco significativos) y de esta forma codificar con elevada precisión los primeros y con poca precisión los segundos, consiguiendo así la compresión.

Los codificadores por transformada, tras la transformación, aplican una cuantificación escalar individual a cada coeficiente del bloque, que no es más que dividir cada coeficiente por un escalón de un tamaño determinado, utilizando escalones pequeños para los coeficientes energéticamente significativos y escalones de mayor tamaño para los coeficientes con poca energía. De esta forma, el tamaño de los coeficientes con mucha energía disminuye considerablemente y además, un porcentaje alto de los coeficientes que tienen poca energía resultan nulos, disminuyendo así el número de bits necesarios para codificar los valores de los coeficientes. Por otro lado, tras la cuantificación, se suele hacer un reordenamiento de los coeficientes cuantificados con el objetivo de situar a los coeficientes nulos de forma consecutiva, y así aplicar técnicas de codificación entrópica que conseguirán comprimir más los datos.

Un gran inconveniente de la cuantificación, es que provoca pérdida de información irreversible y como consecuencia, al aplicar la transformada inversa a los coeficientes recuperados, no se obtendrán los valores de los píxeles originales, sino valores aproximados y por lo tanto, la imagen decodificada presentará distorsión con respecto a la imagen original.

Un codificador será tanto mejor cuanto mayor es la tasa de compresión que obtiene para una calidad dada o bien cuanto menor es la distorsión que presenta la imagen decodificada a una tasa dada. Para encontrar el mejor equilibrio, la transformada es una herramienta crucial.

Por otro lado, dado que en una imagen completa las estadísticas de los datos no son constantes, ya que diferentes regiones de la imagen poseen características espaciales distintas y puesto que la eficiencia de un codificador de imágenes mediante transformada depende en gran medida de la compactación que produce la transformada que usa, dicha eficiencia a lo largo de la imagen variará dependiendo de las distintas regiones que la conforman.

Una transformada ampliamente utilizada en codificación de imagen es la transformada discreta de coseno o DCT. Esta transformada en particular, compacta de forma más eficiente las zonas en donde la imagen es más homogénea, es decir, el valor

de los píxeles a lo largo de estas regiones varía de forma lenta, o lo que es lo mismo, en las regiones en las que predominan las frecuencias espaciales bajas. Sin embargo, la compactación por parte de la DCT es más limitada en las zonas donde el valor de los píxeles varía de forma rápida, o lo que es lo mismo, en las regiones de la imagen en las que predominan las altas frecuencias espaciales.

Por el contrario la transformada Kahurnen-Loève o KLT es capaz de adaptarse a las estadísticas que presentan los datos a transformar, ya que para su construcción utiliza los propios datos. La KLT trata a cada píxel del bloque como si fuera una muestra de una variable aleatoria y se adapta según la correlación existente entre las variables que forman el bloque, de esta forma al transformar el bloque, obtiene un bloque de coeficientes en el que cada uno se corresponde también con una muestra de una variable aleatoria, pero ahora estas variables transformadas estarán totalmente decorreladas entre sí, provocando en el bloque transformado una compactación óptima de la energía. Para construir una KLT especializada, hay que hacerlo a partir de un conjunto de bloques con características espaciales similares, es decir, que las estadísticas que presentan los píxeles de un bloque sean similares a las de los otros bloques; de esta forma, cuanto más parecidos sean los bloques en este sentido, mayor especialización obtendrá la KLT y mayor será la compactación de la energía que provoque en dichos bloques.

1.2 OBJETIVOS

El principal objetivo de este proyecto es aprovechar el carácter adaptativo de la KLT para así, intentar obtener una compactación óptima de la energía en las diferentes regiones que existan en una imagen. Para ello, se partirá de un codificador de imágenes basado en la DCT (codificación no adaptativa) y se modificará de forma que en lugar de transformar todos los bloques de la imagen con la DCT, el codificador utilice un conjunto de transformadas formado por la DCT y un número determinado de KLTs especializadas de acuerdo con las características espaciales de las diferentes regiones existentes en la imagen.

Aparte del diseño completo del proceso de codificación multitransformada, también es objetivo de este proyecto el diseño de la parte del codificador encargada de la obtención de las transformadas KLT. Para ello, se ha buscado la manera de caracterizar espacialmente a los bloques de la imagen para, mediante un algoritmo de agrupamiento, agrupar los bloques con características espaciales similares y obtener una KLT especializada para los bloques de cada grupo.

El estándar de codificación de imágenes que se toma como referencia en este proyecto, es el estándar JPEG, y en concreto su modo secuencial basado en la DCT (este estándar se describe con detalle en el apartado 2.4.2.1). El motivo de elegir JPEG es que es un algoritmo de escasa complejidad y cuyas prestaciones son válidas para un gran número de escenarios diferentes de uso, además de por ser uno de los estándares de codificación de imágenes mas extendido.

CAPÍTULO 2 - ESTADO DEL ARTE

2.1 INTRODUCCIÓN

En este capítulo se recogen las principales técnicas usadas en compresión de imágenes mediante transformada, explicando con mayor detalle los fundamentos teóricos y las propiedades de la transformada discreta de coseno (DCT) y de la transformada Karhunen-Loève (KLT), que son las transformadas que se utilizan en este proyecto. También recoge un resumen de los principales estándares de compresión de imágenes, realizando un análisis más profundo del estándar JPEG ya que es el que se usa en este proyecto. Por último se repasan algunas técnicas de codificación adaptativa aplicadas a la codificación de imágenes y de vídeo.

2.2 REDUNDANCIA EN IMÁGENES

Una propiedad de las imágenes naturales es que la redundancia espacial entre muestras próximas es muy alta. El ejemplo que se desarrolla en [Salomon, 2007] pretende mostrar la alta correlación existente entre un píxel y su vecino. Para ello utiliza la imagen 'lenna.bmp' en escala de grises de 256 niveles (cada píxel puede tomar un valor entre 0 y 255) y cuyo tamaño es de 512×512 píxeles.

Figura 2.1 'lenna.bmp'.

Para visualizar la correlación, se extraen bloques de 2×1 píxeles no solapados, en orden de izquierda a derecha y de arriba abajo. Cada par de píxeles de cada bloque se puede interpretar como un punto en un espacio de dos dimensiones, correspondiendo así los píxeles de las líneas impares de la imagen con la coordenada x y los píxeles de las líneas pares con la coordenada y. Al representar los puntos en un espacio de dos dimensiones, dado que los valores de dos píxeles cercanos van a ser parecidos, es lógico pensar que dichos puntos estarán concentrados alrededor de la línea $y = x$ (45°). Observando la figura 2.2, en la que se representan dichos puntos, se puede apreciar dicha concentración alrededor de la línea $y = x$.

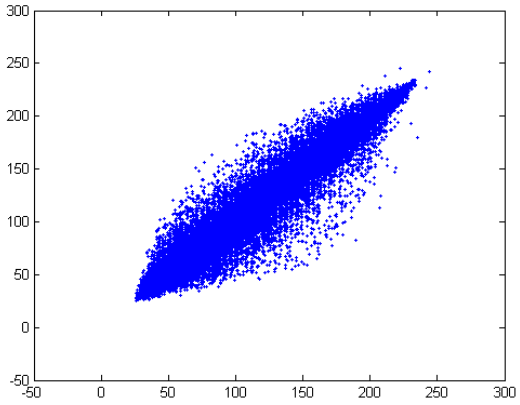


Figura 2.2 Representación en un espacio bidimensional de los puntos de 'lenna.bmp'.

Como se puede observar, la mayoría de los puntos forman una nube alrededor de esta línea, y sólo unos pocos puntos están localizados fuera de ella. Dicho de otro modo, las variables x e y están altamente correladas.

2.3 CODIFICACIÓN DE IMÁGENES MEDIANTE TRANSFORMADA

Un sistema de compresión consiste en dos bloques estructurales distintos: un codificador y un decodificador. El codificador es alimentado con una imagen de entrada $p(x, y)$, a partir de la cual obtiene un conjunto de símbolos. Después de la transmisión o del almacenamiento de los símbolos, la representación codificada es procesada por el decodificador, donde se genera una imagen de salida reconstruida $\hat{p}(x, y)$. Si $\hat{p}(x, y)$ es una réplica exacta de $p(x, y)$, la codificación recibe el nombre de codificación sin pérdidas. Sin embargo, si la imagen reconstruida presenta un cierto nivel de distorsión, la codificación se denomina como codificación con pérdidas [Gonzalez, 2002]. La figura 2.1 muestra el diagrama de bloques simplificado de un sistema de compresión de imágenes.

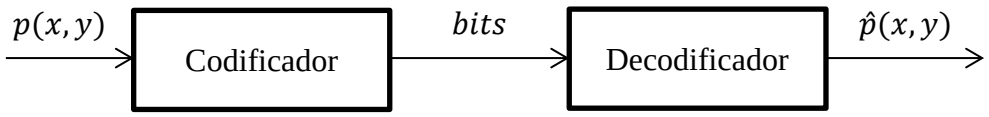


Figura 2.3 Diagrama de bloques básico de un sistema de compresión de imágenes.

Los modelos típicos de codificación de imagen poseen tres fases: fase de transformación, fase de cuantificación y fase de codificación entrópica. En la figura 2.4 se representa el diagrama de bloques típico de un codificador de imágenes mediante transformada [Goyal, 2001].

Figura 2.4 Diagrama de bloques típico de un codificador de imágenes.

La primera fase consiste en transformar los datos linealmente mediante una transformada diseñada principalmente con el objetivo de producir coeficientes incorrelados y reducir así la redundancia entre píxeles vecinos de la imagen de entrada, permitiendo de esta forma la compresión de los datos en posteriores fases del proceso de codificación. Esta operación generalmente es reversible.

La segunda fase consiste en una cuantificación escalar independiente de cada coeficiente reduciendo así la precisión de los datos transformados. El escalón de cuantificación que se aplica a cada coeficiente, varía en función de la sensibilidad del sistema de visión humano, al efecto subjetivo del error de cuantificación de los mismos. Esta operación es irreversible, y por esta razón no es posible recuperar de forma exacta la imagen de entrada original (codificación con pérdidas).

Por último, la tercera fase consiste en la codificación entrópica de los coeficientes cuantificados formando una secuencia de bits. Normalmente, antes de aplicar la codificación entrópica, se aplica un reordenamiento de los coeficientes cuantificados con el objetivo de agrupar los datos con valores significativos y obtener una codificación entrópica más eficiente. En la mayoría de los casos el codificador entrópico asigna códigos de longitud variable a los coeficientes cuantificados e indexados, de forma que los valores que ocurren con más frecuencia son representados con palabras de código más pequeñas. La operación, por supuesto, es reversible.

El decodificador por su parte invierte los pasos del codificador para reconstruir una aproximación de la imagen. La acción del codificador entrópico puede ser invertida de forma que se recuperan los mismos coeficientes cuantificados que se produjeron en

la fase de cuantificación del codificador. Posteriormente, tras la cuantificación escalar inversa se obtienen estimaciones $\hat{c}$ de los coeficientes transformados originales. Para completar la reconstrucción, se vuelve a aplicar una transformada lineal obteniendo una aproximación $\hat{p}$ de los datos iniciales. La figura 2.5 muestra el diagrama típico de bloques de un decodificador mediante transformada.

Figura 2.5 Diagrama de bloques típico de un decodificador de imágenes.

2.3.1 TRANSFORMACIÓN

Una transformada de imagen aplica una transformación lineal a los píxeles modificando su representación, de forma que obtiene a partir de un conjunto de píxeles estadísticamente dependientes, un conjunto de coeficientes "más independientes".

Es deseable que una transformada de imagen consiga una decorrelación óptima de los datos, de forma que la compactación de la energía en el dominio transformado sea alta y se concentre la gran mayoría en unos cuantos coeficientes transformados. Del mismo modo, la transformación debe ser reversible, por lo tanto es requisito que exista la transformada inversa y que ni en el proceso de transformación ni en el de transformación inversa exista pérdida de información. Por último, la transformada tiene que ser computacionalmente manejable. Un tipo de transformada que cumple estos requisitos son las denominadas transformadas lineales, y más en concreto las transformadas ortogonales.

Lo que hace una transformada lineal es descomponer el bloque imagen en una combinación lineal de un conjunto de funciones base, denominadas imágenes base, las cuales han sido generadas mediante matrices unitarias (matriz cuya inversa es igual a la transpuesta de su conjugada).

2.3.1.1 Transformada Lineal Unidimensional

La expresión de la transformada directa unidimensional, o ecuación de análisis, viene dada por:

$$c[u] = \sum_{x=0}^{N-1} b[u,x]p[x]\,, \qquad u = 0, 1, \dots, N-1 \tag{2.1}$$

donde $p[k]$ denota los píxeles de un vector imagen de tamaño N × 1, $c[u]$ denota a los coeficientes transformados y $b[u,x]$ denota a los pesos de la combinación lineal. De acuerdo con la ecuación (2.1) cada coeficiente $c[u]$ es una combinación lineal de todos los píxeles del vector imagen.

De forma similar, la transformada inversa unidimensional o ecuación de síntesis viene dada por la siguiente expresión:

$$p[x] = \sum_{u=0}^{N-1} i[x,u]c[u]\,, \qquad x = 0, 1, \dots, N-1 \tag{2.2}$$

donde $i[x,u]$ denota los pesos de la combinación lineal. De acuerdo con la ecuación (2.2) cada píxel $p[x]$ es una combinación lineal de todos los coeficientes transformados [Wintz, 1972].

Las ecuaciones (2.1) y (2.2) pueden expresarse de forma matricial como

$$\bar{C} = \bar{\bar{B}}\bar{P} \tag{2.3}$$

$$\bar{P} = \bar{\bar{I}}\bar{C} \tag{2.4}$$

donde P es un vector de tamaño $N \times 1$ y representa los píxeles de un vector imagen, C es un vector de tamaño $N \times 1$ y representa los coeficientes transformados, B es una matriz de tamaño $N \times N$ y se denomina núcleo de la transformada directa e I es otra matriz de tamaño $N \times N$ y se denomina núcleo de la transformada inversa. Ambos núcleos están relacionados del siguiente modo:

$$\bar{\bar{I}} = \bar{\bar{B}}^{-1} \tag{2.5}$$

donde B^{-1} denota la matriz inversa de B.

Propiedad de Ortogonalidad

Se dice que una matriz es unitaria, si su inversa es igual a la transpuesta de la propia matriz conjugada, es decir, dado A una matriz compleja de tamaño $N \times N$, se dice que A es unitaria si $A^{-1} = A^{*T}$, donde A^* denota la matriz conjugada de A y A^T denota la matriz transpuesta de A. Una matriz ortogonal es una matriz unitaria real, de forma que $A^{-1} = A^T$. Una propiedad necesaria para que una matriz cuadrada sea unitaria (u ortogonal) es que sus vectores columna y sus vectores fila formen sistemas de vectores ortonormales respectivamente.

La propiedad de ortonormalidad tiene una cualidad valiosa en las transformadas de imágenes, y es la de la conservación de la energía ya que al cumplirse asegura que la transformación sea reversible:

$$E = \sum_{x=0}^{N-1} |p[x]|^2 = \sum_{u=0}^{N-1} |c[u]|^2 \tag{2.6}$$

Resumiendo, si el núcleo de la transformada es ortonormal, la transformada directa e inversa se puede calcular de forma matricial así:

$$\bar{C} = \bar{\bar{B}}\bar{P} \tag{2.7}$$

$$\bar{P} = \bar{\bar{B}}^T\bar{C} \tag{2.8}$$

Interpretación Estadística

La mejor transformada reversible sería aquella cuyos resultados fueran variables independientes, pero esta transformada no puede determinarse en la práctica. Sin embargo la solución más cercana es una transformada lineal que produzca coeficientes incorrelados pero no necesariamente independientes.

Con las transformadas lineales se consigue que las variables transformadas estén menos correladas que las originales o incluso totalmente incorreladas. Este es el caso de la transformada KLT, cuyo núcleo se calcula a partir de la matriz de covarianza de los píxeles [Wintz, 1972]. Esta transformada y otras, se explican con más detenimiento en el apartado 2.3.

Interpretación Geométrica

Una transformación lineal también se puede interpretar geométricamente como la aplicación de una rotación a los datos. Para demostrarlo retomaremos el ejemplo con la imagen ‘lenna.bmp’ comentado en el apartado 2.1.

Dada la correlación que presentan las variables x e y mostrada en la figura 2.2, una posible transformación invertible que redujera dicha correlación, sería aplicar una rotación a los datos de forma que la nube de puntos coincidiera con el eje x, es decir, aplicar una rotación de 45° a todos los puntos del siguiente modo:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} cos\,45° & sin\,45° \\ -sin\,45° & cos\,45° \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = A \begin{pmatrix} x \\ y \end{pmatrix} \tag{2.9}$$

En (2.9) A se denomina matriz de transformación y es una matriz ortonormal, es decir, sus filas son vectores con norma 1 y a su vez entre ellos son ortogonales (el producto escalar de dos cualesquiera de ellos es nulo). Gracias a esta propiedad, la matriz de transformación inversa se puede obtener directamente ya que coincide con su traspuesta:

$$\begin{pmatrix} x \\ y \end{pmatrix} = A^{-1} \begin{pmatrix} x' \\ y' \end{pmatrix} = A^T \begin{pmatrix} x' \\ y' \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} x' \\ y' \end{pmatrix} \tag{2.10}$$

A continuación, en la figura 2.6, se han representado los puntos transformados e inmediatamente se puede apreciar que tras la transformación, en la mayoría de los puntos la coordenada y es igual o próxima a cero, mientras que los valores de la coordenada x no han variado sustancialmente. Lo que ha conseguido la transformación ha sido reducir la varianza drásticamente en la coordenada y, pero ha aumentado la varianza de la coordenada x. Sin embargo, debido a que la matriz de transformación es ortonormal, la varianza total de los píxeles no ha variado. Dicho de otra forma, la transformación ha concentrado la energía en la coordenada x en detrimento de la coordenada y, lo cual se podría aprovechar para conseguir compresión utilizando una codificación que ignorara la coordenada y debido a su baja varianza.

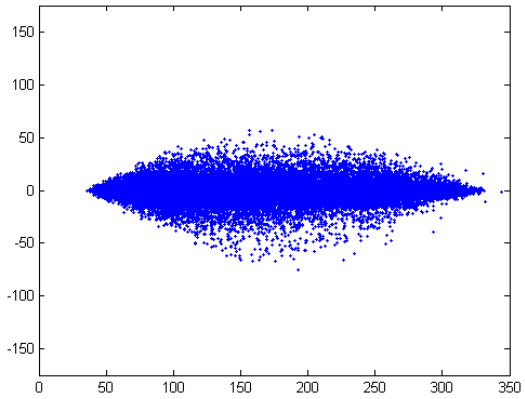


Figura 2.6 Representación en un espacio bidimensional de los puntos transformados de 'lenna.bmp'.

Resumiendo, aplicar una transformación lineal a un vector de N dimensiones (subimagen de N pixels) no es más que aplicar una rotación en los ejes del sistema de coordenadas N-dimensional del vector, y calcular los coeficientes con respecto a este nuevo sistema de coordenadas. Esta transformación simple puede ser extendida a cualquier número de dimensiones, el único inconveniente es que no podemos visualizar espacios de más de 3 dimensiones, pero matemáticamente no hay ningún problema. Por ejemplo, en lugar de seleccionar parejas de píxeles adyacentes se pueden seleccionar tripletes, los cuales se pueden representar como puntos en un espacio tridimensional. La rotación ahora tendría que hacerse teniendo en cuenta los ejes x, y, z, por lo que habría que utilizar una matriz ortonormal de tamaño 3×3 para transformar cada vector.

Interpretación de Vectores Base

Otra interpretación que se le puede dar a la transformación lineal de un vector es la de descomposición en vectores base.

Dado un vector $\vec{p}$ perteneciente a un espacio vectorial $\mathbb{R}^N$, suponemos un conjunto de vectores ortonormales $\{\vec{b}_u\}$ con $u = 0,1,\dots,N-1$, el cual es base

generadora del propio espacio, entonces, cualquier vector perteneciente a este espacio vectorial se puede escribir de forma única como una combinación lineal de los elementos de dicha base de la siguiente forma:

$$\vec{p} = \sum_{u=0}^{N-1} c_u \vec{b}_u \tag{2.11}$$

donde cada coeficiente c_u con $u = 0,1,\dots,N-1$, viene dado por $c_u = \frac{\langle \vec{b}_u, \vec{p} \rangle}{\langle \vec{b}_u, \vec{b}_u \rangle}$, siendo $\langle \vec{m}, \vec{n} \rangle$ el producto escalar de los vectores $\vec{m}$ y $\vec{n}$. Dado que los vectores $\vec{b}_u$ son ortonormales se cumple que $\langle \vec{b}_u, \vec{b}_u \rangle = 1$, por lo que:

$$c_u = [b_{u,0} \quad b_{u,1} \quad \cdots \quad b_{u,N-1}] \begin{bmatrix} p_0 \\ p_1 \\ \vdots \\ p_{N-1} \end{bmatrix} \tag{2.12}$$

luego el conjunto de coeficientes c se puede calcular de forma matricial como:

$$\begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{N-1} \end{bmatrix} = \begin{bmatrix} b_{0,0} & b_{0,1} & \cdots & b_{0,N-1} \\ b_{1,0} & b_{1,1} & \cdots & b_{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ b_{N-1,0} & b_{N-1,1} & \cdots & b_{N-1,N-1} \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ \vdots \\ p_{N-1} \end{bmatrix} \tag{2.13}$$

que es equivalente al cálculo matricial que se ha utilizado anteriormente para aplicar una rotación a los vectores extraídos de la imagen.

Finalmente, la expresión matricial (2.13) puede expresarse de forma algebraica del siguiente modo:

$$c[u] = \sum_{x=0}^{N-1} g[x] b[u,x] \,, \qquad u = 0,1,\dots,N-1 \tag{2.14}$$

correspondiendo esta última expresión a la forma general de la transformada directa unidimensional, denominada ecuación de análisis.

2.3.1.2 Transformación Lineal Bidimensional

Una imagen digital (en escala de grises) puede tratarse como una señal dependiente de dos variables, por ejemplo x e y, donde x correspondería con las filas de la imagen, e y con las columnas, y p haría referencia al nivel de gris de cada píxel, entonces dada una imagen de $N \times N$ píxeles, la transformada directa bidimensional se define como:

$$c[u,v] = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} b[u,v,x,y]\, p[x,y], \qquad u,v = 0,1,\dots,N-1 \tag{2.15}$$

y del mismo modo la transformada inversa bidimensional se puede definir como:

$$p[x,y] = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} i[x,y,u,v]c[u,v]\,, \qquad x,y = 0,1,\dots,N-1 \tag{2.16}$$

$b[x,y,m,n]$ y $i[x,y,m,n]$ se denominan núcleos de transformación de la transformada directa y de la transformada inversa, respectivamente.

Propiedad de Separabilidad

Un núcleo de transformación es separable si cumple la siguiente expresión:

$$b[x,y,u,v] = b_c[x,u]b_f[y,v] \tag{2.17}$$

Una transformada bidimensional separable se puede descomponer en dos transformadas unidimensionales, es decir, la transformación en dos dimensiones puede implementarse por una transformación unidimensional de las columnas, seguida por otra transformada unidimensional de las filas como se muestra en (2.18) y (2.19).

$$c_c[x,v] = \sum_{y=0}^{N-1} p[x,y]b_c[y,v]\,, \qquad x,v = 0,1,\dots,N-1 \tag{2.18}$$

$$c[u,v] = \sum_{x=0}^{N-1} c_c[x,v]b_f[x,u]\,, \qquad u,v = 0,1,\dots,N-1 \tag{2.19}$$

y sustituyendo (2.18) en (2.19):

$$c[u,v] = \sum_{x=0}^{N-1}\left(\sum_{y=0}^{N-1} p[x,y]b_c[y,v]\right)b_f[x,u]\,, \qquad u,v = 0,1,\dots,N-1 \tag{2.20}$$

Propiedad de Simetría

Un núcleo de transformación se dice que es simétrico si el núcleo es separable y además se cumple la siguiente condición:

$$b_f[y,v] = b_c[y,v] \tag{2.21}$$

es decir, que b_f es funcionalmente equivalente a b_c.

La representación en forma matricial ayuda a la comprensión de estas dos propiedades. Siendo P una matriz de tamaño $N \times N$ que representa a una imagen digital en escala de grises, donde sus elementos se denotan por $p_{x,y}$ con $x, y = 0,1, \dots, N-1$, y B_c y B_f la pareja de matrices de transformación unidimensional de tamaño $N \times N$ correspondientes al núcleo de transformación de una transformada bidimensional separable, entonces la transformación bidimensional de P puede expresarse como:

$$C = B_c P B_f^T \tag{2.22}$$

donde C representa la matriz de coeficientes transformados de tamaño $N \times N$.

Y la expresión de la transformada inversa resulta:

$$P = (B_c^{-1}) C (B_f^{-1})^T \tag{2.23}$$

Finalmente, si ambos núcleos de transformación cumplen la propiedad de simetría, entonces la expresión de la transformación directa puede escribirse como:

$$C = BPB^T \tag{2.24}$$

Y la expresión de la transformación inversa puede expresarse como:

$$P = B^{-1} C (B^{-1})^T \tag{2.25}$$

Propiedad de Ortogonalidad

Como se ha explicado para el caso de la transformada unidimensional, esta propiedad es necesaria para conservar la energía tras la transformación, y asegurar así la transformación inversa. Si una matriz cumple la propiedad de ortonormalidad, entonces $B^{-1} = B^T$. Luego, para el caso bidimensional, si el núcleo de transformación cumple las propiedades de separabilidad, simetría y ortogonalidad, la transformación directa e inversa en su forma matricial puede expresarse como:

$$C = BPB^T \tag{2.26}$$

$$P = B^T C B \tag{2.27}$$

Imágenes Base

Al igual que la transformada unidimensional se puede interpretar como la descomposición en vectores base, la transformada bidimensional se puede interpretar como la descomposición en imágenes base.

Dada la ecuación de la transformada inversa bidimensional, ecuación (2.16), ésta se puede expresar como combinación lineal de matrices, o lo que es lo mismo, de imágenes bidimensionales del siguiente modo:

$$P = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} c_{u,v} H_{u,v} \tag{2.28}$$

donde $c_{u,v}$ representa cada uno de los coeficientes transformados de la imagen, y $H_{u,v}$ cada una de las imágenes base que forman el núcleo de transformación de la transformada bidimensional.

Del mismo modo que para el caso unidimensional y debido a que se ha seleccionado un conjunto de imágenes base ortonormal, los pesos de la combinación lineal (coeficientes transformados) vienen dados por el producto escalar entre la imagen sin transformar y cada una de las imágenes base:

$$c_{u,v} = \langle H_{u,v}, P \rangle \tag{2.29}$$

Según [Wintz, 1972] ahora se puede apreciar que si se usa un set de imágenes base ortogonales de forma que los coeficientes sean más independientes que los píxeles, entonces las varianzas de los coeficientes serán desiguales y por lo tanto, si únicamente se utilizaran los primeros q coeficientes para obtener una aproximación de la imagen original tal que:

$$\hat{P} = \sum_{u=0}^{q-1} \sum_{v=0}^{q-1} c_{u,v} H_{u,v} \tag{2.30}$$

el Error Cuadrático Medio (MSE) con el que resultaría $\hat{P}$ sería:

$$MSE = E\left\{\left\|P - \hat{P}\right\|^2\right\}$$

$$= E\left\{\left\|\sum_{u=0}^{N-1} \sum_{v=0}^{N-1} c_{u,v} H_{u,v} - \sum_{u=0}^{q-1} \sum_{v=0}^{q-1} c_{u,v} H_{u,v}\right\|^2\right\} \tag{2.31}$$

$$= E\left\{\left\|\sum_{u=q}^{N-1} \sum_{v=q}^{N-1} c_{u,v} H_{u,v}\right\|^2\right\}$$

y debido a que las imágenes base forman un conjunto ortonormal, el MSE resultaría ser igual a la suma de las varianzas de los coeficientes descartados:

$$MSE = \sum_{u=p}^{N-1} \sum_{v=p}^{N-1} \sigma_{c_{u,v}} \tag{2.32}$$

2.3.2 CUANTIFICACIÓN

En el campo de la compresión de imágenes mediante transformada, la cuantificación se usa para reducir el rango de los valores de los coeficientes transformados, convirtiéndolos además en números enteros, permitiendo una codificación más eficiente. Si los coeficientes tienen valores grandes, el cuantificador los convierte en valores pequeños, y si los coeficientes tienen valores cercanos a cero, a la salida del cuantificador estos coeficientes serán cero. Esta reducción en el rango de los valores de los coeficientes permite que sean necesarios menos bits para su representación, generando así compresión. La cuantificación es un proceso irreversible, por lo que no es posible recuperar la pérdida de información que provoca, resultando en una compresión con pérdidas.

Hay dos tipos de cuantificación según el número de datos cuantificados conjuntamente. Si únicamente se cuantifica un dato escalar, la cuantificación recibe el nombre de cuantificación escalar. Mientras que si es un vector de escalares lo que se cuantifica, la cuantificación recibe el nombre de cuantificación vectorial.

Cuantificación Escalar

Un ejemplo simple de cuantificación escalar es el proceso de redondear un número real al entero más cercano. El proceso es con pérdidas dado que no es posible determinar el valor exacto del número real original a partir del entero redondeado.

A continuación se muestra la forma de general de un cuantificador escalar uniforme:

$$x^Q = round\left(\frac{x}{\Delta}\right)$$
$$\hat{x} = x^Q \cdot \Delta \tag{2.33}$$

donde Δ es el tamaño del escalón de cuantificación. Los niveles de los valores cuantificados están espaciados a intervalos uniformes de Δ (figura 2.7).

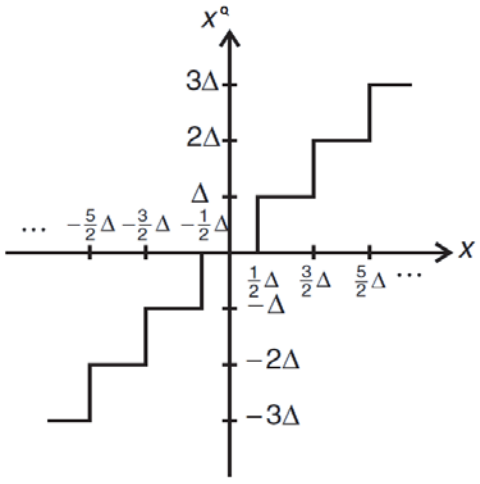


Figura 2.7 Cuantificación escalar.

En compresión de imágenes la operación de cuantificación normalmente se lleva a cabo en dos partes: una cuantificación directa en el codificador en el que a cada coeficiente se le aplica una división por un escalón de cuantificación junto con un redondeo al entero más próximo y una cuantificación inversa en el decodificador en la que los coeficientes cuantificados son multiplicados por los respectivos escalones que se utilizaron en el codificador, recuperando finalmente aproximaciones de los coeficientes originales.

Un parámetro crítico en la cuantificación es el tamaño del escalón Δ utilizado para dividir los coeficientes. Si el tamaño del escalón es grande, el rango de los valores cuantificados resulta pequeño lo cual permite que se puedan representar eficientemente durante la transmisión (gran compresión), pero provoca que los valores recuperados en la cuantificación inversa sean una burda aproximación de los valores originales. Sin embargo, si el tamaño del escalón es pequeño, los valores recuperados se aproximarán mucho a los valores originales, pero el rango de los valores cuantificados será grande y la eficiencia de la compresión se verá reducida.

La cuantificación en compresión de imágenes se usa para reducir la precisión de los datos de una imagen después de aplicar una transformada (como la DCT o la wavelet) eliminando los valores cercanos a cero. El cuantificador de un codificador de imágenes está diseñado para asignar un cero a los valores insignificantes y para conservar un reducido número de valores significativos. A la salida de un cuantificador, si la transformada que usa está correctamente diseñada se obtendrá un conjunto de coeficientes cuantificados de los cuales la mayoría de ellos serán cero.

2.3.3 CODIFICADOR ENTRÓPICO

En este apartado se describe la función del codificador entrópico dentro de la compresión de imágenes, así como se resumen dos de las técnicas más ampliamente usadas en codificación entrópica, que son la codificación Huffman y la codificación aritmética, ambas incluidas en el estándar internacional de codificación de imágenes JPEG. En [Shi, 2000, Cap.5], [Salomon, 2008, Cap.2,4] y [Richardson, 2003, Cap.3], se puede encontrar esta información con mayor detalle.

El codificador entrópico convierte una serie de símbolos, que representan a los elementos de una imagen, en una cadena binaria comprimida lista para su transmisión o para ser guardada.

Dada una fuente de información representada por un alfabeto fuente *S*

$$S = \{s_1, s_2, \dots, s_m\}$$

donde s_i, $i = 1,2,\dots,m$, son símbolos fuente, se define el mensaje como una secuencia de uno o varios símbolos fuente. Para el caso de compresión de imágenes mediante

transformación, los símbolos que forman el alfabeto serán los posibles valores cuantificados que puedan tomar los coeficientes resultantes de la transformación.

2.3.3.1 Codificación Huffman

Los códigos Huffman son códigos de longitud variable (VLC). En dichos códigos el codificador mapea cada símbolo del alfabeto fuente con una cadena de símbolos de código denominada palabra de código. Sea un alfabeto de código A con

$$A = \{a_1, a_2, \dots, a_r\}$$

donde a_j, con $j = 1,2, \dots, r$, son símbolos de código, se define una palabra de código como una secuencia de símbolos de código que representan a un símbolo fuente, es decir,

$$s_i \rightarrow A_i(a_{i1}, a_{i2}, \dots, a_{ik})$$

donde la palabra de código A_i es una cadena de k símbolos de código asignados al símbolo fuente s_i.

En codificación binaria, el número de símbolos de código r es igual a 2: los dígitos binarios “0” y “1”.

En los VLC la longitud de las palabras de código es variable, pero siempre contienen un número entero de bits. Normalmente, a los símbolos que ocurren más frecuentemente se les asigna las palabras de código más cortas, mientras que los menos frecuentes se representan con las palabras de código más largas.

Un código Huffman además de ser un VLC, posee otras propiedades valiosas como la de que es un código decodificable de forma única, es decir, que no hay ambigüedad en la decodificación de ninguna palabra de código; también es un código instantáneo, o lo que es lo mismo, ninguna palabra de código es prefijo de otra; y también es un código compacto ya que minimiza la longitud media de las palabras de código.

Procedimiento

El primer paso de la codificación Huffman es organizar todos los símbolos fuente de forma que sus probabilidades de ocurrencia estén en orden decreciente.

El segundo paso es asignar un “0” binario y un “1” binario a los dos símbolos del alfabeto con menor probabilidad, para después combinarlos y formar un nuevo símbolo fuente con una probabilidad igual a la suma de sus probabilidades. Este paso se repite hasta que el alfabeto resulte con un único símbolo fuente con probabilidad 1.

El tercer paso es formar las palabras de código de cada símbolo. Para ello se utilizan los 0s y 1s binarios que se han ido asignando a los dos símbolos con menor probabilidad de cada fase, comenzando desde la última fase y hacia atrás. La longitud de un símbolo dependerá del número de veces en las que el símbolo compuesto al que pertenece haya sido uno de los dos símbolos menos probable. En la figura 2.8 se representa un ejemplo.

Símbolo Fuente	Probabilidad de Ocurrencia	Palabra de Código Asignada	Longitud Palabra de Código
s_1	0.3	00	2
s_2	0.1	101	3
s_3	0.2	11	2
s_4	0.05	1001	4
s_5	0.1	1000	4
s_6	0.25	01	2

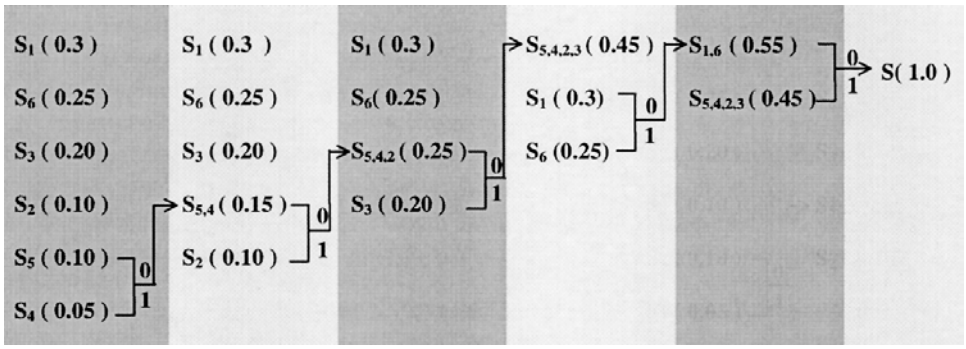


Figura 2.8 Ejemplo de código Huffman [Shi, 2000].

2.3.3.2 Codificación Aritmética

La codificación Huffman tiene el inconveniente de asignar una palabra de código con un número entero de bits para cada símbolo ya que el número de bits óptimo para un símbolo, depende del contenido de la información y normalmente es un número fraccionario. Por este motivo, la eficiencia en la compresión de los códigos de longitud variable es particularmente pobre para símbolos con probabilidades mayores de 0’5.

Sin embargo, la codificación aritmética asigna un código binario (normalmente largo) a todo un mensaje, resultando un número fraccionario de bits por símbolo. El método lee todos los símbolos de la cadena que representa al mensaje y calcula la probabilidad de cada símbolo. Con dichas probabilidades va estrechando consecutivamente un cierto intervalo inicial de valores reales. Especificar un intervalo más estrecho requiere más bits, por lo que los intervalos construidos por el algoritmo requieren cada vez más y más números para especificar sus límites. El algoritmo está diseñado de forma que un símbolo con probabilidad alta estreche el intervalo menos que

un símbolo con baja probabilidad, con lo que resulta que los símbolos con alta probabilidad contribuyen con pocos bits a la salida, consiguiendo así la compresión.

El rango de valores reales inicial es $[0,1)$ (la notación $[a,b)$ significa el rango de números reales desde a hasta b, incluyendo a pero no incluyendo b) y la salida del codificador aritmético se interpreta como un número dentro de ese rango. Sólo se tiene en cuenta la parte decimal de dicho código, por lo que el código 9746509 se interpreta como 0.9746509.

Procedimiento

Para entender cómo se divide el intervalo $[0,1)$ en subintervalos, a continuación se representa un ejemplo de codificación aritmética del mismo mensaje que se codificó en el ejemplo de codificación Huffman, y que también aparece en [Shi, 2000, Cap.5].

Al igual que en el ejemplo anterior, el alfabeto fuente está formado por seis símbolos cuyas probabilidades de ocurrencia se recogen en la tabla 2.1 junto con los subintervalos de $[0,1)$ que se asignan a cada símbolo dependiendo de su probabilidad de ocurrencia. El mensaje a codificar es la serie de símbolos $s_1 s_2 s_3 s_4 s_5 s_6$.

Símbolos Fuente	Probabilidad de Ocurrencia	Subintervalos Asociados
s_1	0.3	$[0, 0.3)$
s_2	0.1	$[0.3, 0.4)$
s_3	0.2	$[0.4, 0.6)$
s_4	0.05	$[0.6, 0.65)$
s_5	0.1	$[0.65, 0.75)$
s_6	0.25	$[0.75, 1)$

Tabla 2.1 Ejemplo de codificación aritmética: probabilidades de los símbolos.

Inicialmente se parte del intervalo $[0,1)$, y este intervalo se va reduciendo según los subintervalos asociados a los símbolos que forman la cadena a codificar. Al codificar un nuevo símbolo, la longitud del intervalo inicial se reduce de forma proporcional al valor de la longitud del subintervalo asociado a dicho símbolo del siguiente modo:

$$Long_{nueva} = Long_{actual} \cdot p(s_i) \tag{2.34}$$

donde $p(s_i)$ es la probabilidad de ocurrencia del símbolo i.

El límite inferior del intervalo resultante después de codificar un símbolo, se calcula mediante la siguiente expresión:

$$Lim\_inf_{nuevo} = Lim\_inf_{actual} + Lim\_inf_{símbolo} \cdot Long_{actual} \tag{2.35}$$

Y finalmente el límite superior del nuevo intervalo se calcula mediante la suma del nuevo límite inferior más la nueva longitud calculada:

$$Lim\_sup_{nuevo} = Lim\_inf_{nuevo} + Long_{nueva} \tag{2.36}$$

En la figura 2.9 se muestra gráficamente todo el proceso.

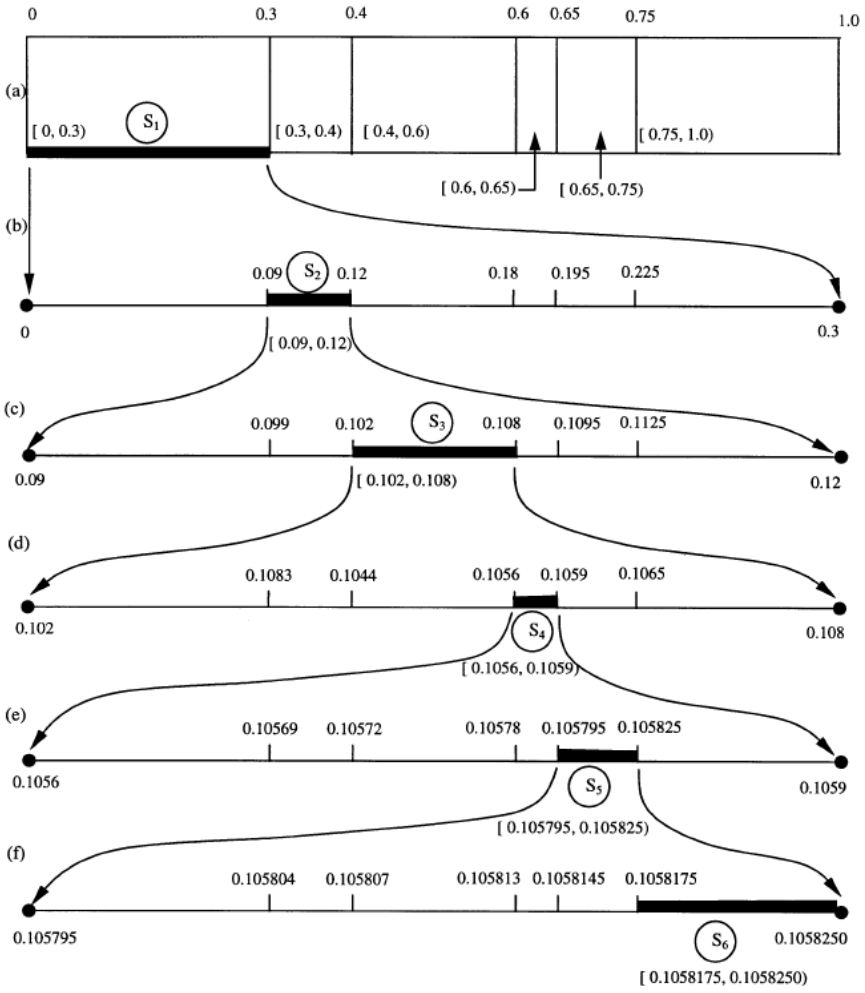


Figura 2.9 Ejemplo de codificación aritmética: cálculo de subintervalos [Shi, 2000].

Como se puede observar, el intervalo final que resulta tras codificar la secuencia de símbolos que forman el mensaje es $[0.1058175, 0.1058250)$, por lo que dicho mensaje se podría codificar con un único número perteneciente a este intervalo.

2.3.4 Rendimiento De Un Codificador De Imágenes Con Pérdidas

El rendimiento de un codificador de imágenes con pérdidas viene determinado por dos factores: la distorsión que introduce en la imagen decodificada y el grado de compresión de la imagen codificada.

Como resultado del proceso de cuantificación, aparece distorsión en la imagen decodificada. Según [Wu, 2006] las medidas de distorsión están divididas en dos categorías: medidas subjetivas y medidas objetivas. Las medidas subjetivas de calidad son las que están evaluadas por humanos.

Sin embargo en las medidas objetivas, la distorsión se calcula como la diferencia entre la imagen original P, y la imagen reconstruida $\hat{P}$. Todos los cambios son considerados como distorsiones, sin importar cómo aparecen para el observador humano. La cantidad de distorsión de la imagen reconstruida se mide normalmente con el error cuadrático medio (MSE) y con la relación señal a ruido de pico a pico ($PSNR$)

$$MSE = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left[P(x,y) - \hat{P}(x,y)\right]^2 \tag{2.37}$$

$$PSNR = 10 \cdot log_{10}\left(\frac{MAX_I^2}{MSE}\right) = 20 \cdot log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right) \tag{2.38}$$

donde MAX_I denota el máximo valor de intensidad que puede tomar una muestra (o píxel) en la imagen, que para precisiones de 8 bits es de 255.

Por otro lado, la medida que se utiliza para evaluar el tamaño es el número de bits por píxel (bpp), que no es más que el número de bits esperado a la salida del codificador dividido por el número de muestras codificadas.

Es usual presentar gráficamente el rendimiento de un codificador mediante una gráfica en la que se representa la evolución del nivel de PSNR para distintos grados de compresión medidos en bits por píxel, o lo que es lo mismo, la relación entre la tasa y la distorsión.

2.4 TRANSFORMADAS USADAS EN COMPRESIÓN DE IMÁGENES

Se han propuesto muchas transformadas para la compresión de imágenes, pero las más populares se pueden agrupar en dos categorías según la forma de aplicarse a la imagen: transformadas de bloque y transformadas de imagen. Las transformadas de bloque no se aplican a la imagen completa, sino que la imagen se divide en subimágenes denominadas bloques, y se transforma individualmente cada una de estas subimágenes. Algunos ejemplos de transformadas de bloque típicamente usadas en codificación de imágenes son, la Transformada de Walsh-Hadamard (WHT), la Transformada Discreta de Coseno (DCT) y la Transformada de Karhunen-Loève (KLT).

Sin embargo, las transformadas de imagen operan con la imagen entera. La más popular es la Transformada Wavelet Discreta (DWT). Está demostrado que las transformadas de imagen, como la DWT, obtienen mejores resultados que las

transformadas de bloque, pero tienden a necesitar mayores requerimientos de memoria ya que la imagen completa es procesada como una unidad.

2.4.1 TRANSFORMADAS DE BLOQUE

Las transformadas de bloque son transformadas lineales y como ya se ha comentado, el objetivo de las lineales en imágenes, es compactar la mayor energía posible en los primeros coeficientes, para poder eliminar los coeficientes energéticamente poco significativos, minimizando así el MSE de la imagen reconstruida y consiguiendo de esta forma la compresión. La mayor o menor compactación de la energía en los primeros coeficientes, depende del conjunto de funciones base que forme el núcleo de transformación de la transformada.

2.4.1.1 Transformada Walsh-Hadamard (WHT)

La transformada Walsh-Hadamard (WHT) es poco eficiente compactando la energía en los primeros coeficientes, por lo que es poco usada, pero sin embargo es muy sencilla de calcular ya que su matriz de transformación está formada sólo por $+1$ y -1.

Su núcleo de transformación cumple las propiedades de separabilidad, simetría y ortogonalidad.

WHT unidimensional

El conjunto de funciones base $\{b_u\}$ con $u = 0,1, \dots N-1$, que forman el núcleo de transformación de la transformada directa unidimensional vienen dados por

$$b_u[x] = \frac{1}{N}(-1)^{\sum_{k=0}^{N-1} bit_k(x)bit_k(u)}, \qquad x = 0,1, \dots, N-1 \tag{2.39}$$

donde la suma del exponente se realiza en módulo 2, siendo $bit_k(x)$ el bit k-ésimo de la representación binaria de x, luego la expresión de la transformada directa unidimensional resulta:

$$c[u] = \frac{1}{N}\sum_{x=0}^{N-1} p[x](-1)^{\sum_{k=0}^{N-1} bit_k(x)bit_k(u)} \tag{2.40}$$

Dado que el núcleo de transformación de la transformada directa cumple la propiedad de ortogonalidad, el núcleo de transformación de la transformada inversa es igual que el de la directa salvo en el factor $1/N$:

$$i_x[u] = (-1)^{\sum_{k=0}^{N-1} bit_k(x)bit_k(u)}, \qquad u = 0,1, \dots, N-1 \tag{2.41}$$

expresándose la transformada inversa Walsh-Hadamard unidimensional como:

$$p[x] = \sum_{u=0}^{N-1} c[u](-1)^{\sum_{k=0}^{N-1} bit_k(x)bit_k(u)} \tag{2.42}$$

WHT bidimensional

Por otro lado, el conjunto $\{b_{u,v}\}$ con $u, v = 0,1, \dots, N-1$, de funciones base (imágenes) que forman el núcleo de transformación de la transformada directa bidimensional viene dado por:

$$b_{u,v}[x,y] = \frac{1}{N}(-1)^{\sum_{k=0}^{N-1}[bit_k(x)bit_k(u)+bit_k(y)bit_k(v)]}, \qquad x, y = 0,1, \dots, N-1 \tag{2.43}$$

y el de la transformada inversa, $\{i_{x,y}\}$ con $x, y = 0,1, \dots, N-1$, viene dado por:

$$i_{x,y}[u,v] = \frac{1}{N}(-1)^{\sum_{k=0}^{N-1}[bit_k(x)bit_k(u)+bit_k(y)bit_k(v)]}, \qquad u, v = 0,1, \dots, N-1 \tag{2.44}$$

y como su núcleo cumple las propiedades de separabilidad, simetría y ortogonalidad:

$$\begin{aligned} b_{u,v}[x,y] = b_u[x]b_v[y] = b_u[x]b_u[y] = i_x[u]i_x[v] = \\ = \left[\frac{1}{\sqrt{N}}(-1)^{\sum_{k=0}^{N-1} bit_k(x)bit_k(u)}\right]\left[\frac{1}{\sqrt{N}}(-1)^{\sum_{k=0}^{N-1} bit_k(y)bit_k(v)}\right] \end{aligned} \tag{2.45}$$

En la figura 2.10 se muestra el conjunto de imágenes base para $N = 4$, dónde blanco denota $+1$ y negro denota -1 (el factor $1/N$ se ha ignorado). Las filas y las columnas en la figura, corresponden respectivamente con los valores de u y v desde 0 a 3, ordenadas en modo creciente. Las filas y las columnas dentro de cada bloque corresponden respectivamente con los valores de x e y desde 0 a 3.

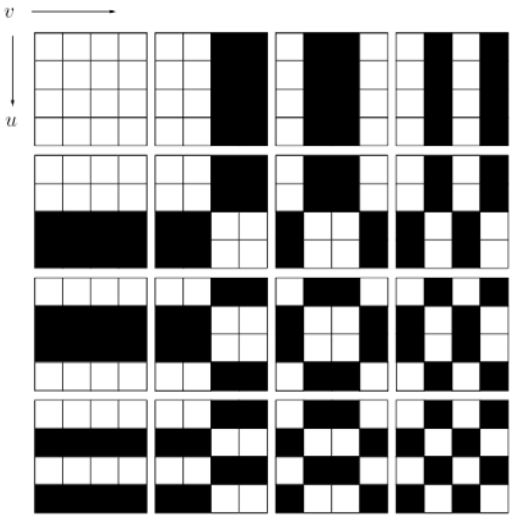


Figura 2.10 Imágenes base dela WHT bidimensional 4 × 4.

2.4.1.2 Transformada Discreta De Coseno (DCT)

La DCT ha sido ampliamente utilizada tanto en estándares modernos de codificación de vídeo como por ejemplo los desarrollados por MPEG o JVT, como en estándares de codificación de imágenes tan extendidos como por ejemplo JPEG.

DCT unidimensional

La expresión de la transformada directa de la DCT unidimensional viene dada por:

$$c[u] = \alpha[u] \sum_{x=0}^{N-1} p[x] cos\left[\frac{(2x+1)u\pi}{2N}\right], \qquad u = 0,1,\dots,N-1 \tag{2.46}$$

y la de la transformada inversa unidimensional por:

$$p[x] = \sum_{u=0}^{N-1} \alpha[u]c[u] cos\left[\frac{(2x+1)u\pi}{2N}\right], \qquad x = 0,1,\dots,N-1 \tag{2.47}$$

donde:

$$\alpha[u] = \begin{cases} \sqrt{\frac{1}{N}}, & u = 0 \\ \sqrt{\frac{2}{N}}, & u \neq 0 \end{cases} \tag{2.48}$$

El conjunto de funciones base $\{b_u\}$ con $u = 0,1,\dots N-1$, que forma el núcleo de transformación de la DCT viene dado por:

$$b_u[x] = \alpha[u] cos\left[\frac{(2x+1)u\pi}{2N}\right], \qquad x = 0,1,\dots N-1 \tag{2.49}$$

Como se puede observar, las funciones base son ortogonales, por lo que ninguna de ellas puede ser representada como una combinación de las otras funciones base, es decir, son independientes. Esto provoca que el conjunto de funciones base (vectores) que forman el núcleo de transformación de la DCT inversa sea el mismo que el de la DCT directa.

Estas son las expresiones matriciales de las DCT directa e inversa unidimensionales:

$$C = BP \tag{2.50}$$

$$P = B^T C \tag{2.51}$$

donde P es un vector de tamaño $N \times 1$ que representa un vector de píxeles de una imagen, C es un vector de tamaño $N \times 1$ que contiene los coeficientes transformados, y B es una matriz de tamaño $N \times N$ que representa el núcleo de transformación de la DCT y sus filas corresponden con los vectores base de la DCT unidimensional $\{B_u\}$:

$$B = \begin{bmatrix} B_0^T \\ B_1^T \\ \vdots \\ B_{N-1}^T \end{bmatrix} \tag{2.52}$$

DCT bidimensional

La DCT directa bidimensional es una extensión del caso unidimensional, y viene dada por:

$$c[u,v] = \alpha[u]\alpha[v] \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} p[x,y] cos\left[\frac{(2x+1)u\pi}{2N}\right] cos\left[\frac{(2y+1)v\pi}{2N}\right], u,v = 0,1,\dots,N-1 \tag{2.53}$$

y del mismo modo la DCT inversa bidimensional viene dada por:

$$p[x,y] = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \alpha[u]\alpha[v]c[u,v] cos\left[\frac{(2x+1)u\pi}{2N}\right] cos\left[\frac{(2y+1)v\pi}{2N}\right], x,y = 0,1,\dots,N-1 \tag{2.54}$$

El núcleo de transformación de la DCT cumple las propiedades de separabilidad, simetría y ortogonalidad, por lo que la DCT bidimensional, directa o inversa, puede calcularse como dos DCT unidimensionales del siguiente modo

$$C = BPB^T \tag{2.55}$$

$$P = B^T CB \tag{2.56}$$

donde P es una matriz de tamaño $N \times N$ que representa un bloque de píxeles de una imagen en escala de grises, C es una matriz de tamaño $N \times N$ que contiene los coeficientes del bloque transformado, y B es una matriz de tamaño $N \times N$ que se corresponde con la matriz de transformación de la DCT unidimensional.

Como ya se ha comentado, las transformadas bidimensionales se pueden interpretar como la descomposición de una imagen en imágenes base en el caso de la transformada directa o como la composición de imágenes base en el de la inversa. Las imágenes base de la DCT se pueden generar fácilmente a partir de los vectores base de la DCT unidimensional, mediante la multiplicación de dichos vectores con sus traspuestos:

$$B_{u,v} = B_u B_v^T = \begin{bmatrix} b_{u,0} \\ b_{u,1} \\ \vdots \\ b_{u,N-1} \end{bmatrix} [b_{v,0} \quad b_{v,1} \quad \cdots \quad b_{v,N-1}] \tag{2.57}$$

En la figura 2.11 se muestra las imágenes base para $N = 8$.

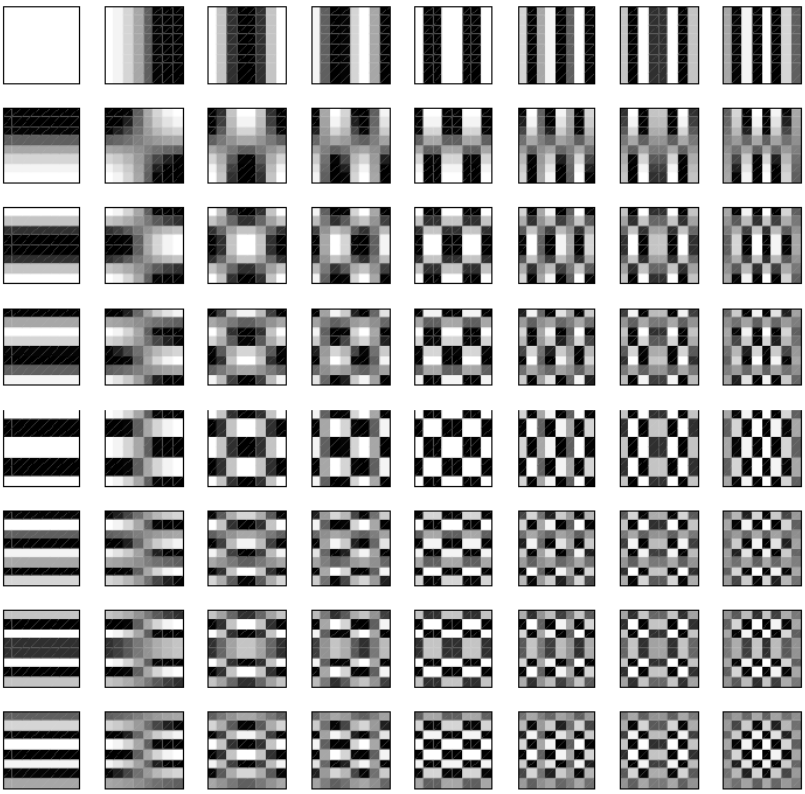


Figura 2.11 Imágenes base de la DCT bidimensional 8×8 [Richardson, 2003].

Espectro de la DCT

Aplicar la DCT a un bloque es equivalente a representarlo con el conjunto de imágenes base del núcleo de transformación de la DCT, lo que es equivalente a aislar las distintas frecuencias contenidas en el bloque, indicando cada coeficiente transformado el peso de dicha frecuencia en la imagen. Las bajas frecuencias contienen la información importante (psicovisualmente hablando), mientras que las altas frecuencias corresponden con los detalles de la imagen por lo que son menos importantes. Gracias a esta distinción en las frecuencias, se utilizan distintos escalones de cuantificación para cuantificar los coeficientes transformados, de forma que el escalón es más pequeño para las bajas frecuencias y va aumentando según crece la frecuencia [Salomon, 2007].

Por lo general, si los datos de entrada están altamente correlados, entonces la mayoría de los coeficientes transformados resultantes tras la DCT son cero o cercanos a cero, y sólo unos pocos son valores altos (normalmente los primeros), por lo que tras la cuantificación la mayoría de los coeficientes de la DCT serán cero.

2.4.1.3 Transformada De Karhunen-Loève

La transformada Karhunen-Loève se basa en el análisis por componentes principales de Hotelling (PCA). La KLT es por definición la transformada óptima por dos razones principalmente: la primera de ellas es que tras transformar los datos iniciales correlados, obtiene coeficientes incorrelados y la segunda es que compacta la máxima energía posible en los primeros coeficientes, minimizando así el error cuadrático medio (MSE) al reconstruir los datos mediante transformación inversa sin utilizar los coeficientes de menor energía.

Decorrelación de los datos [Britanak, 2007]

Consideramos un vector $\bar{X}$ de N variables aleatorias correladas, su vector de medias denotado por $\bar{M}_X$, y su matriz de covarianza denotada por $\bar{\bar{V}}_X$, donde:

$$\bar{X} = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{bmatrix}, \qquad \bar{M}_X = \begin{bmatrix} m_0 \\ m_1 \\ \vdots \\ m_{N-1} \end{bmatrix} = \begin{bmatrix} E[x_0] \\ E[x_1] \\ \vdots \\ E[x_{N-1}] \end{bmatrix} \tag{2.58}$$

y $E[\cdot]$ denota al operador esperanza. Por otro lado, la matriz de covarianza de $\bar{X}$, denotada por $\bar{\bar{V}}_X$, se calcula como:

$$\bar{\bar{V}}_X = E[(\bar{X} - \bar{M}_X)(\bar{X} - \bar{M}_X)^T] \tag{2.59}$$

Por simplicidad en los cálculos supondremos que el vector de medias $\bar{M}_X$ es nulo, resultando:

$$\bar{\bar{V}}_X = E[\bar{X}\bar{X}^T] \tag{2.60}$$

Como se ha indicado anteriormente, la KLT es óptima, entre otras razones, porque es capaz de obtener coeficientes incorrelados mediante transformación lineal:

$$\bar{Y} = \bar{\bar{B}}\bar{X} \tag{2.61}$$

La matriz de covarianza de un conjunto de variables aleatorias incorreladas es una matriz diagonal perfecta, luego inspeccionando la matriz de covarianza del vector de coeficientes transformados podemos hallar la matriz de transformación de la KLT. Suponiendo que el vector de coeficientes transformados también tiene media nula, la matriz de covarianza $\bar{\bar{V}}_Y$ del vector de coeficientes transformados resulta:

$$\bar{\bar{V}}_Y = E[\bar{Y}\bar{Y}^T] = diag(\lambda_0, \lambda_1, \dots, \lambda_{N-1}) = \begin{bmatrix} \lambda_0 & 0 & \cdots & 0 \\ 0 & \lambda_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_{N-1} \end{bmatrix} \tag{2.62}$$

donde λ_u con $u = 0,1, \dots, N-1$, denota las varianzas de los coeficientes transformados.

Finalmente, sustituyendo (2.61) en (2.62):

$$E[\bar{Y}\bar{Y}^T] = E\left[\bar{\bar{B}}\bar{X}\bar{X}^T\bar{\bar{B}}^T\right] = \bar{\bar{B}}E[\bar{X}\bar{X}^T]\bar{\bar{B}}^T = diag(\lambda_0, \lambda_1, \dots, \lambda_{N-1}) \qquad (2.63)$$

o lo que es lo mismo:

$$\bar{\bar{V}}_Y = \bar{\bar{B}}\bar{\bar{V}}_X\bar{\bar{B}}^T \qquad (2.64)$$

y puesto que la transformación tiene que ser reversible, la matriz de transformación $\bar{\bar{B}}$ tiene que cumplir la propiedad de ortogonalidad, es decir, que $\bar{\bar{B}}^{-1} = \bar{\bar{B}}^T$, resultando

$$\bar{\bar{V}}_Y\bar{\bar{B}}^T = \bar{\bar{V}}_X\bar{\bar{B}}^T \qquad (2.65)$$

y dado que $\bar{\bar{V}}_Y = diag(\lambda_0, \lambda_1, \dots, \lambda_{N-1})$, se obtiene el siguiente problema de autovalores

$$\lambda_u\bar{B}_u = \bar{\bar{V}}_X\bar{B}_u\,, \qquad u = 0,1, \dots, N-1 \qquad (2.66)$$

siendo $\bar{B}_n$ el vector columna n en la matriz $\bar{\bar{B}}^T$.

Resumiendo, las filas de la matriz de transformación de la KLT corresponden con los autovectores de la matriz de covarianza de los datos, correspondiendo los autovalores asociados a dichos autovectores, con las varianzas de los coeficientes transformados.

Minimización del MSE

Desde el punto de vista de la información, cuanta más varianza tiene un coeficiente transformado mayor información contiene, por lo que la magnitud de cada autovalor es un indicativo de la información que aporta el autovector asociado en la reconstrucción del vector original. Por ello, para formar la matriz de transformación de la KLT, los autovectores se ordenan en orden decreciente según el autovalor asociado. De esta forma, se minimiza el MSE tras aplicar truncado en los coeficientes y reconstruir el vector original [Wintz, 1972]:

Sea $\bar{Y}_p$ un vector de tamaño $N \times 1$ cuyos p primeros elementos coinciden con los p coeficientes transformados con mayor varianza y el resto son nulos, el vector reconstruido $\bar{X}$ se calcula como se muestra a continuación:

$$\hat{X} = \bar{\bar{B}}^T\bar{Y}_p = \sum_{u=0}^{p-1} y_u\,\bar{B}_u \qquad (2.67)$$

Finalmente, el error cuadrático medio resulta como:

$$
\begin{aligned}
MSE &= E\left\{\left\|\bar{X}-\hat{\bar{X}}\right\|^2\right\} = \\
&= E\left\{\left(\sqrt{\langle(\bar{X}-\hat{\bar{X}}),(\bar{X}-\hat{\bar{X}})\rangle}\right)^2\right\} = E\left\{\left(\bar{X}-\hat{\bar{X}}\right)^T\left(\bar{X}-\hat{\bar{X}}\right)\right\} = \\
&= E\left\{\left[\bar{\bar{B}}^T\left(\bar{Y}-\bar{Y}_p\right)\right]^T\bar{\bar{B}}^T\left(\bar{Y}-\bar{Y}_p\right)\right\} = E\left\{\left(\bar{Y}-\bar{Y}_p\right)^T\left(\bar{Y}-\bar{Y}_p\right)\right\} = \\
&= E\left\{\sum_{u=p}^{N-1} y_u^2\right\} = \sum_{u=p}^{N-1} E\{y_u^2\} = \sum_{u=p}^{N-1}\lambda_u
\end{aligned}
\tag{2.68}
$$

Matriz de transformación KLT dependiente de los datos

La KLT es una transformada dependiente de los datos, es decir, que su núcleo (o matriz) de transformación dependerá de los datos que se quieran transformar, al contrario que la transformada WHT o la DCT, que tienen un núcleo de transformación fijo. Esto hace que la KLT sea la transformada más apropiada para utilizar en codificación adaptativa, ya que según varían las propiedades estadísticas de los datos, la KLT se adapta a estas variaciones, mientras que la DCT no. Por otro lado, esto supone un inconveniente, y es que el decodificador necesita, para aplicar la transformada inversa, la misma matriz de transformación que utilizó el codificador para transformar los datos, por lo que es absolutamente necesario hacerle llegar dicha matriz, aumentando de esta forma la cantidad de bits enviados al decodificador y por consiguiente, disminuyendo la compresión total de los datos.

KLT en imágenes

En compresión de imágenes, la WHT y la DCT se aplican normalmente en su forma bidimensional a los bloques de las imágenes, atacando primero la correlación existente entre las columnas del bloque y seguidamente la existente entre las filas, mediante dos transformadas unidimensionales respectivamente.

Por el contrario la KLT, aunque también se aplica a los bloques de la imagen, ataca la correlación de los píxeles del bloque con una única transformación unidimensional. El motivo de esto es porque el núcleo de transformación de la KLT, en general, no es separable, por lo que cuando se aplica a bloques, estos han de ser vectorizados. En (2.69) se muestra como se vectoriza un bloque de tamaño $N \times N$ mediante la concatenación de sus filas, obteniendo un vector de tamaño $N^2 \times 1$.

$$\bar{\bar{P}} = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1N} \\ p_{21} & p_{22} & \cdots & p_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ p_{N1} & p_{N2} & \cdots & p_{NN} \end{bmatrix} \quad \rightarrow \quad \bar{P} = \begin{bmatrix} p_{11} \\ p_{12} \\ \vdots \\ p_{1N} \\ p_{21} \\ \vdots \\ p_{2N} \\ \vdots \\ p_{N1} \\ \vdots \\ p_{NN} \end{bmatrix} \tag{2.69}$$

Para calcular la matriz de transformación KLT para un conjunto de bloques de tamaño $N \times N$, se vectorizan dichos bloques obteniendo vectores de tamaño $N^2 \times 1$. Estos vectores pueden considerarse como realizaciones de una variable aleatoria N^2-dimensional, por lo que es factible calcular su matriz de covarianza, y a partir de ella obtener la matriz de transformación KLT de tamaño $N^2 \times N^2$.

El problema de aplicar la KLT de forma unidimensional a los bloques de tamaño $N \times N$, es el elevado tamaño de la matriz de transformación, puesto que es necesario enviarla al decodificador para aplicar la transformada inversa, incrementando considerablemente la tasa binaria y en consecuencia reduciendo la compresión.

Sin embargo, es posible calcular un par de KLTs separables, de forma que una de ellas atacaría la correlación entre las columnas de los bloques y la otra la de las filas. La combinación de este par de transformadas es una transformada sub-óptima, aunque se aproxima al rendimiento de la KLT y reduce los requisitos de memoria que supone la transformada en un factor de 32 [Feng, 1999] y en [Feng, 2002].

El cálculo de las matrices de transformación KLT separables para un conjunto de bloques de tamaño $N \times N$, se hace de forma independiente. Para calcular la matriz de transformación de tamaño $N \times N$ que decorrela las columnas, denotada por B_C, se utilizan todas las columnas de los bloques, considerando dichas columnas de tamaño $N \times 1$ realizaciones de una variable aleatoria N-Dimensional. Del mismo modo, para calcular la matriz de transformación de tamaño $N \times N$ que decorrela las filas, denotada por B_F, también se utilizan todas las filas de los bloques considerándolas como realizaciones de una variable aleatoria N-Dimensional, obteniendo la matriz de transformación a partir de su matriz de covarianza.

Finalmente la expresión matricial de la transformada KLT separable queda del siguiente modo:

$$\bar{\bar{C}} = \bar{\bar{B}}_C \bar{\bar{P}} \bar{\bar{B}}_F \tag{2.70}$$

siendo P una matriz de $N \times N$ píxeles, y C la matriz de coeficientes transformados de tamaño $N \times N$.

Relación entre KLT y DCT

La DCT tiene un rendimiento sub-óptimo si se compara contra la KLT, aunque muy cercano al de ésta para datos con valores significativamente altos de correlación, cosa que ocurre entre píxeles adyacentes en imágenes. Ambas transformadas pueden ser relacionadas a partir de una clase particular de matriz de covarianza perteneciente a las señales de primer orden de Markov [Clarke, 1981].

Los datos de una imagen pueden aproximarse mediante una señal de primer orden de Markov cuya matriz de correlación es la siguiente

$$C = \begin{bmatrix} 1 & \rho & \rho^2 & \cdots & \cdots & \rho^{N-1} \\ \rho & 1 & \ddots & \ddots & & \vdots \\ \rho^2 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \rho^2 \\ \vdots & & \ddots & \ddots & 1 & \rho \\ \rho^{N-1} & \cdots & \cdots & \rho^2 & \rho & 1 \end{bmatrix} \tag{2.71}$$

donde ρ es la correlación entre píxeles adyacentes ($0 \leq \rho \leq 1$).

Tal como se desarrolla en [Ray, 1970], para el caso de una señal estacionaria de Markov de primer orden, sus autovalores se pueden calcular de forma analítica del siguiente modo:

$$\lambda_n = \frac{(1-\rho^2)}{1 - 2\rho\cos(\mu_n) + \rho^2} \tag{2.72}$$

donde μ_n son las raíces reales positivas de la siguiente ecuación:

$$\tan(N\mu) = \frac{-(1-\rho^2)\sin\mu}{(1+\rho^2)\cos\mu - 2\rho} \tag{2.73}$$

y los elementos de los autovectores vienen dados por:

$$w_n(m) = \sqrt{\frac{2}{N+\lambda_n}} \sin\left\{\mu_n\left[(m+1) - \frac{N+1}{2}\right] + \frac{(n+1)\pi}{2}\right\} \tag{2.74}$$

Para el caso de que ρ tienda a uno $w_n(m)$ resultan ser los elementos de la matriz de transformación de la DCT [Britanak, 2007]:

$$\mu_n = \frac{n\pi}{N}, \qquad n = 0,1,\ldots,N-1 \tag{2.75}$$

resultando:

$$w_n(m) = \sigma_n \sqrt{\frac{2}{N}} \cos\left\{n\left(m + \frac{1}{2}\right)\frac{\pi}{N}\right\}, \qquad m,n = 0,1,\ldots,N-1 \tag{2.76}$$

donde:

$$\sigma_n = f(x) = \begin{cases} \frac{1}{\sqrt{2}}, & para\ n = 0 \\ 1, & en\ otro\ caso \end{cases} \quad (2.77)$$

2.4.1.4 Selección Del Tamaño De Bloque

Con la transformada lo que se busca es, a partir de unos píxeles altamente correlados, obtener unos coeficientes con la menor correlación posible, para que de esta forma se produzca una alta compactación de la energía en unos cuantos coeficientes y se pueda conseguir una mayor compresión en los procesos posteriores de cuantificación y codificación entrópica. Pero, si la correlación entre los píxeles es débil, la compactación de la energía que la transformada provoca, disminuirá considerablemente, por lo que aumentará la energía eliminada en el proceso de cuantificación y con ello el MSE.

Como ya se ha visto anteriormente, la correlación entre píxeles cercanos es alta pero, dicha correlación disminuye según aumenta la distancia entre los píxeles. Por este motivo, la imagen se subdivide en bloques cuadrados de $\mathrm{N} \times \mathrm{N}$ píxeles, y es a los bloques a los que se les aplica la transformada.

Según [Wintz, 1972], en la mayoría de las imágenes la correlación entre píxeles es significativa sólo entre 20 píxeles adyacentes, aunque este número depende en gran medida de la cantidad de detalle de la imagen, de hecho, no se puede garantizar que para tamaños con $N > 16$ el rendimiento no se vea reducido.

Sin embargo, para las técnicas adaptativas, un tamaño de bloque demasiado grande no permite la adaptación a las características locales de la imagen, mientras que un tamaño muy pequeño de bloque hace que no se tengan en cuenta suficientes correlaciones entre píxeles. Por ello, el tamaño de bloque elegido debe ser un compromiso entre el número de correlaciones tenidas en cuenta y la capacidad de adaptación a las estadísticas locales.

El tamaño adoptado en el estándar de codificación de imágenes JPEG, así como en los estándares de codificación de vídeo H.261, H.263, MPEG 1 y MPEG 2 es $N = 8$.

2.4.2 TRANSFORMADAS DE IMAGEN

Como se ha visto, en la codificación tradicional de imágenes por transformada (transformadas ortogonales), un problema es la elección del tamaño de bloque, ya que éste debe seleccionarse de forma que las estadísticas que contiene, proporcionen un buen modelo del comportamiento de la imagen. Sin embargo, la transformada wavelet se aplica a la imagen completa, descomponiéndola en bandas de frecuencia.

2.4.2.1 Transformada Wavelet

Tal y como se explica en [Usevitch, 2001], la forma genérica para una transformada wavelet unidimensional se muestra en la figura 2.12.

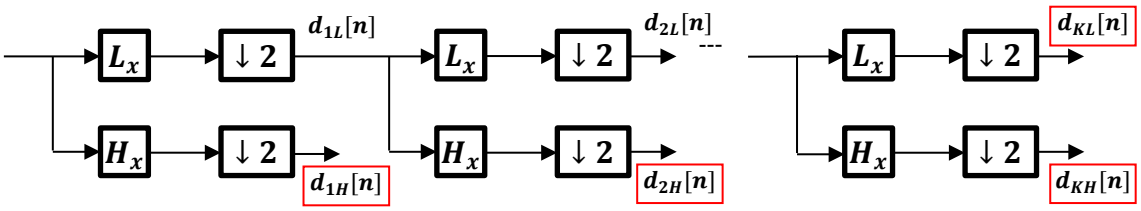


Figura 2.12 Transformada Wavelet: Diagrama de bloques para el caso unidimensional.

La señal se pasa a través de dos filtros, uno paso-bajo y uno paso-alto, L_x y H_x, respectivamente, y las dos señales que se obtienen se diezman por un factor de 2, constituyendo la transformada de nivel uno. Los siguientes niveles se consiguen repitiendo el proceso de filtrado y de diezmado sólo en la señal filtrada paso-bajo. El proceso normalmente se lleva a cabo para un número finito de niveles K, y los coeficientes resultantes, $d_{iH}[n]$ con $i = 1,2,\dots,K$ y $d_{KL}[n]$, se llaman coeficientes wavelet.

La transformada wavelet unidimensional se puede extender a dos dimensiones usando filtros wavelet separables. Con filtros separables las transformadas bidimensionales pueden calcularse primero aplicando una transformada unidimensional a las filas, y después repitiéndolo en todas las columnas, tratando filas y columnas como señales unidimensionales.

Tal y como podemos encontrar en [Richardson, 2003], cada fila de la imagen es filtrada con un filtro paso-bajo y uno paso-alto (L_x y H_x) y la salida de cada filtro es diezmada por un factor de dos para producir imágenes intermedias L y H. L es la imagen original filtrada paso-bajo y H es la imagen original filtrada paso-alto, ambas imágenes se diezman en la dirección horizontal por un factor de 2, de forma que el número de columnas de L y H es la mitad que en la imagen original. Después, cada columna de las nuevas imágenes es filtrada también con filtros paso-bajo y paso-alto (L_y y H_y) y diezmada por un factor de 2 resultando 4 subimágenes (LL_1, LH_1, HL_1 y HH_1). Estas cuatro imágenes sub-banda pueden ser combinadas para crear una imagen con el mismo número de muestras que la original. En la figura 2.13 se puede observar el diagrama de bloques que resume el proceso.

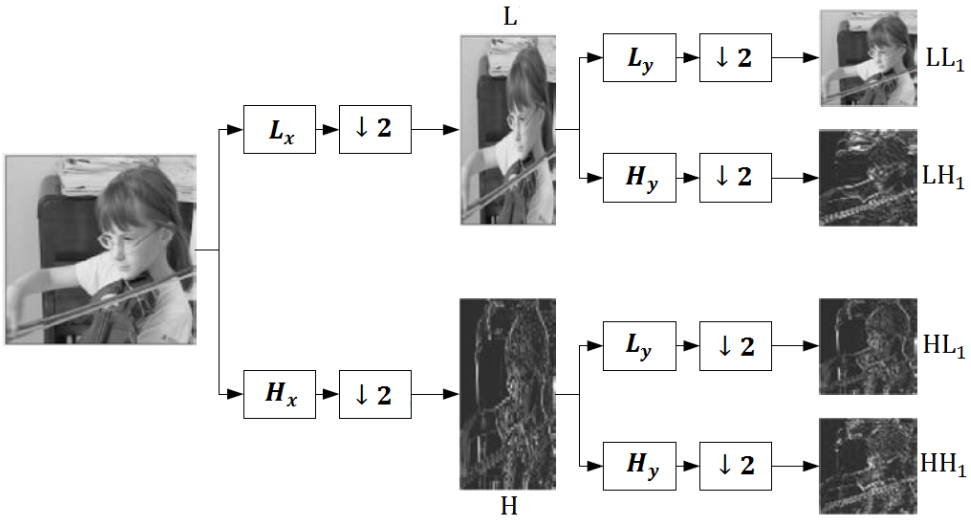


Figura 2.13 Transformada Wavelet: Diagrama de bloques para el caso bidimensional [Richardson, 2003].

LL_1 es la imagen original filtrada paso-bajo en las direcciones horizontal y vertical y submuestreada por un factor de 2. HL_1 está filtrada paso-alto en la dirección vertical (filas) y contiene frecuencias verticales residuales, LH_1 está filtrada paso-alto en la dirección horizontal (columnas) y contiene frecuencias horizontales residuales y HH_1 está filtrada paso-alto en ambas direcciones. Las cuatro subbandas contienen toda la información presente en la imagen original, y forman el primer nivel de la descomposición wavelet. Las bajas frecuencias representan un ancho de banda $0 < |\omega| < \pi/2$, mientras que las altas frecuencias representan la banda $\pi/2 < |\omega| < \pi$. Para obtener el siguiente nivel de la descomposición, se aplica el mismo proceso a la subimagen LL_1 formando cuatro nuevas subimágenes. La imagen filtrada paso-bajo resultante es iterativamente filtrada para crear un árbol de imágenes subbanda. Muchas de las muestras (coeficientes) en las imágenes subbanda filtradas paso-alto son cercanas a cero y es posible conseguir compresión eliminando estos coeficientes insignificantes.

La figura 2.14 muestra cuatro niveles de subbandas, donde el nivel 1 contiene los detalles de la imagen, es decir, los coeficientes wavelet de alta frecuencia o de resolución fina, y el nivel superior, el nivel 4, contiene la información más general de la imagen (coeficientes de baja frecuencia). Está claro que los niveles inferiores pueden cuantificarse de forma más agresiva sin perder demasiada información importante de la imagen, mientras que los niveles superiores deberían ser cuantificados más finamente. La estructura en subbandas es la base de todos los métodos de compresión que emplean la transformada wavelet.

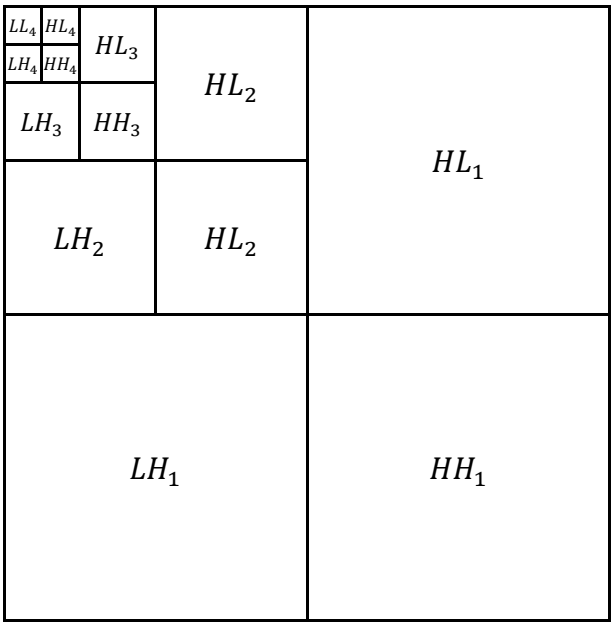


Figura 2.14 Transformada Wavelet: descomposición en subbandas.

2.5 ESTÁNDARES DE COMPRESIÓN DE IMÁGENES

2.5.1 DPCM

El método de compresión DPCM es un método de codificación diferencial. Aprovecha el hecho de que en imágenes, píxeles consecutivos tienen valores muy parecidos y en lugar de codificar los valores de los píxeles, codifica las diferencias entre ellos, consiguiendo así compresión. En las líneas siguientes se muestran las principales características de este esquema de compresión recogidas en [Shi, 2000, Cap.3] y [Salomon, 2007, Cap.4].

Los métodos de codificación diferencial calculan las diferencias d_i entre muestras consecutivas y las codifican como indica la siguiente expresión:

$$d_i = a_i - a_{i-1} \tag{2.78}$$

La primera muestra (a_0), al no precederle ninguna otra, no se codifica diferencialmente sino que se codifica de forma separada. Esto hace que el decodificador la recupere de forma exacta y pueda ser usada para recuperar la muestra siguiente.

Normalmente para codificar las diferencias se utiliza codificación con pérdidas mediante cuantificación, por lo que la cantidad codificada $\hat{d}_i$ no es la diferencia exacta, sino un valor parecido:

$$\hat{d}_i = d_i + q_i \tag{2.79}$$

siendo q_i el error de cuantificación de esa diferencia.

El decodificador para recuperar la muestra actual $\hat{a}_i$ utiliza la muestra decodificada anterior $\hat{a}_{i-1}$ y la diferencia cuantificada actual $\hat{d}_i$ como indica la siguiente expresión:

$$\hat{a}_i = \hat{a}_{i-1} + \hat{d}_i \tag{2.80}$$

Analizando las expresiones (2.79) y (2.80) se puede comprobar que el error de cuantificación de muestras anteriores afecta a la muestra recuperada actual, ya que se acumula dicho error:

$$\hat{a}_n = a_n + \sum_{i=1}^{n} q_i \tag{2.81}$$

Para evitar que el error de cuantificación de muestras anteriores afecte a la recuperación de la muestra actual, la diferencia d_i se calcula entre la muestra actual a codificar y la muestra anterior decodificada como se indica en (2.80):

$$d_i = a_i - \hat{a}_{i-1} \tag{2.82}$$

de esta forma, cuando el decodificador recupere una muestra, ésta sólo contendrá el error de cuantificación correspondiente a dicha muestra:

$$\hat{a}_i = a_i + q_i \tag{2.83}$$

En la figura 2.15 se muestran los diagramas de bloques correspondientes al codificador y decodificador DPCM.

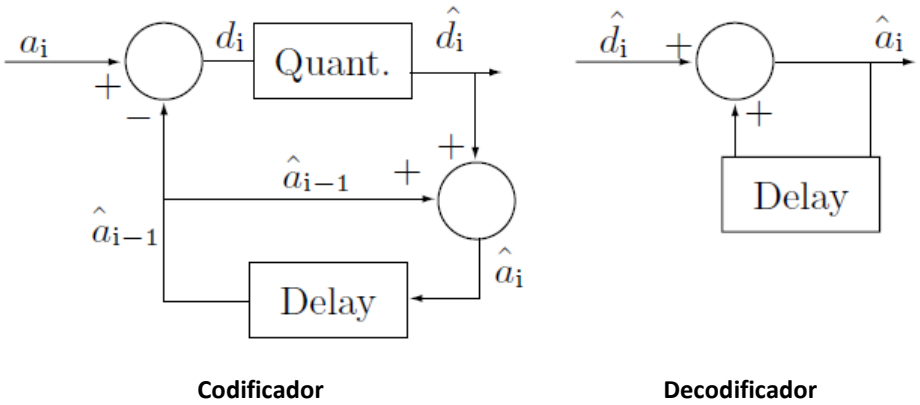


Figura 2.15 DPCM: Diagramas de bloques del codificador y decodificador [Salomon, 2007].

2.5.2 JPEG

En este capítulo se recogen las principales características del estándar, profundizando en el modo secuencial basado en la DCT, puesto que es el modo más usado, y es el que se utilizará para comparar las imágenes codificadas con el algoritmo

propuesto. Gran parte de la información que aquí se recoge procede de [Salomon, 2008], [Wallace, 1991] y [Shi, 2000], en donde se hace una descripción muy amplia del estándar.

JPEG es el acrónimo en inglés de Joint Photographic Experts Group. Fue desarrollado conjuntamente por el CCITT y por la ISO (the International Standards Organization) quienes comenzaron en 1987 y produjeron la primera propuesta de borrador en 1991. El estándar JPEG ha demostrado su eficacia y su uso en compresión de imágenes ha sido muy extendido.

El estándar JPEG permite la codificación de imágenes mediante compresión con pérdidas o sin pérdidas. La codificación con pérdidas sigue un esquema basado en la DCT y se corresponde con el modo básico de JPEG, el cual es apropiado para un gran número de aplicaciones. Para aplicaciones que no toleran pérdidas, como por ejemplo la compresión de imágenes médicas, este modo no es válido; para dichas aplicaciones JPEG dispone de un esquema de codificación sin pérdidas basado en codificación predictiva. Desde el punto de vista de los algoritmos, JPEG incluye cuatro modos distintos de operación denominados, modo secuencial (basado en DCT), modo progresivo (basado en DCT), modo sin pérdidas y modo jerárquico.

2.5.2.1 Modo Secuencial (Basado En La DCT)

En el modo secuencial basado en DCT, se divide primero una imagen en bloques de 8×8 píxeles. Los bloques son procesados de izquierda a derecha y de arriba abajo. A cada bloque se le aplica la DCT directa y los 8×8 coeficientes DCT son cuantificados. Finalmente, los coeficientes DCT cuantificados son codificados entrópicamente, obteniendo los datos de la imagen comprimida.

Las figuras 2.16 y 2.17 muestran los pasos clave de la codificación y de la decodificación. Estas figuras muestran el caso especial de compresión de una imagen en escala de grises. El proceso de compresión de una imagen puede verse esencialmente como la compresión de una cadena de bloques de píxeles de tamaño 8×8.

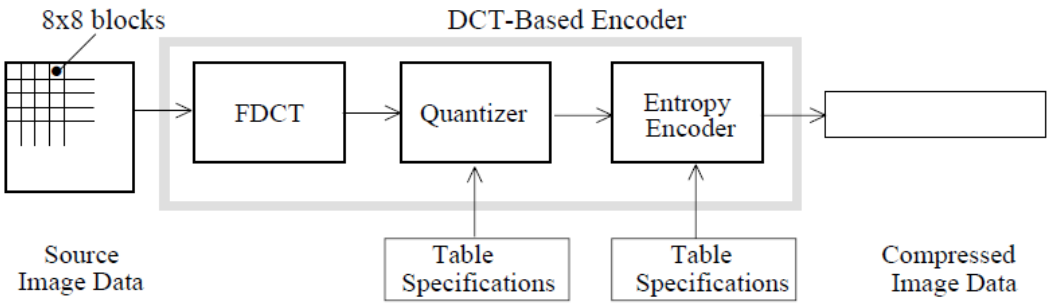


Figura 2.16 JPEG: Diagrama de bloques del codificador.

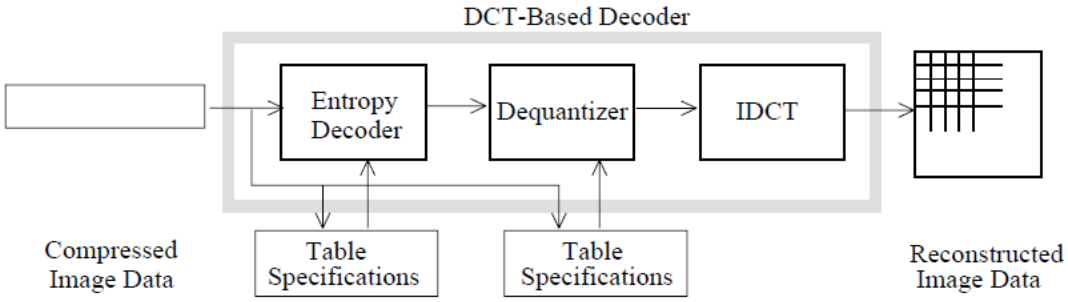


Figura 2.17 JPEG: Diagrama de bloques del decodificador.

Transformadas Discretas de Coseno Directa e Inversa

Inicialmente, los valores de las muestras de la imagen que pertenecen al rango de enteros sin signo $[0, 2^P - 1]$ ($[0, 255]$ con $P = 8$), son desplazadas al rango de enteros con signo $[-2^{P-1}, 2^{P-1} - 1]$ ($[-128, 127]$ con $P = 8$). Seguidamente, a la entrada del codificador, las muestras son agrupadas en bloques de 8×8 y transformadas con la DCT directa (FDCT de su acrónimo en inglés *Fordward Discrete Cosine Transform*). A la salida del decodificador, la DCT inversa (IDCT de su acrónimo en inglés *Inverse Discrete Cosine Transform*) obtiene como salida bloques de 8×8 muestras con los que reconstruye la imagen.

Cada bloque de 8×8 píxeles se puede considerar como una señal discreta de 64 muestras. La DCT directa descompone la imagen en 64 imágenes base ortogonales, correspondiendo cada una de ellas con una de las 64 frecuencias espaciales bidimensionales que componen el espectro de la DCT. La salida de la DCT directa es el conjunto de amplitudes de estas 64 imágenes base calculados para la señal de entrada, ampliamente conocidos como “coeficientes DCT”. Por consiguiente, los valores de los coeficientes DCT pueden ser considerados como la cantidad relativa de las frecuencias espaciales bidimensionales contenidas en la señal de entrada. El coeficiente con frecuencia cero en ambas dimensiones se llama “Coeficiente de DC” y los 63 coeficientes restantes reciben el nombre de “Coeficientes de AC”. Debido a que los valores de los píxeles normalmente varían de forma lenta de un punto a otro del bloque, la DCT directa concentra la mayoría de la señal en las frecuencias espaciales más bajas. Para un bloque típico, la mayoría de las frecuencias espaciales tienen amplitud cero o próxima a cero y no necesitan ser codificadas.

En el decodificador se invierte este proceso, cogiendo los 64 coeficientes de la DCT (los cuales han sido previamente cuantificados) y reconstruyendo una imagen de 64 píxeles mediante la suma de las señales base. En principio, la DCT no introduce pérdidas a las muestras de la imagen fuente; simplemente las transforma a un dominio en el que pueden ser codificadas más eficientemente.

Cuantificación

Después de la salida de la DCT directa, cada uno de los 64 coeficientes de la DCT es cuantificado uniformemente de acuerdo con una tabla de cuantificación de 64 elementos, la cual debe ser especificada por la aplicación (o por el usuario) como un parámetro en el codificador. Cada elemento de la tabla puede ser un valor entero desde 1 a 255, y especifica el tamaño del escalón de cuantificación del coeficiente de la DCT asociado. Con la cuantificación se reduce la precisión de la representación de los coeficientes de la DCT, haciendo nulos los coeficientes con valores próximos a cero, favoreciendo así al codificador entrópico el cual obtiene mayor compresión. Dado que la cuantificación es la principal fuente de pérdidas en los codificadores basados en la DCT, la información eliminada debe ser la menos significativa visualmente.

La cuantificación se define como la división de cada coeficiente de la DCT por su correspondiente escalón, seguido de un redondeo al entero más cercano:

$$T^Q(u,v) = round\left(\frac{T(u,v)}{QP(u,v)}\right) \tag{2.84}$$

La función de cuantificación inversa, revierte la normalización llevada a cabo durante la cuantificación, multiplicando los coeficientes cuantificados por sus escalones de cuantificación correspondientes, resultando unos valores aproximados a los valores de los coeficientes originales, a los que finalmente se les aplica la DCT inversa:

$$\hat{T}(u,v) = T^Q(u,v)QP(u,v) \tag{2.85}$$

Idealmente, cada escalón de cuantificación debería ser escogido en función de la contribución visual de la frecuencia espacial del coeficiente asociado a dicho escalón (umbral perceptivo). El experimento descrito en [Lohscheller, 1984] obtuvo unas tablas de cuantificación para la recomendación CCIR-601 para imágenes y pantallas. Dichas tablas han sido probadas experimentalmente por los miembros del JPEG y aparecen en el estándar de la ISO con carácter informativo, pero no como un requerimiento. En la tabla 2.2 se muestran los escalones de cuantificación para los coeficientes de la componente de luminancia y para los de la de crominancia.

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

Luminancia

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

Crominancia

Tabla 2.2 JPEG: Tablas de cuantificación recomendadas.

Como se puede observar en la tabla 2.2 los escalones de cuantificación generalmente crecen según nos movemos desde la esquina superior izquierda hacia la esquina inferior derecha, es decir, JPEG aplica una cuantificación más agresiva a los coeficientes DCT de alta frecuencia que a los de baja. Esto es así, porque el sistema visual humano es más sensible a las bajas frecuencias espaciales que a las altas.

Después de la cuantificación, el coeficiente de DC es tratado de forma separada a los 63 coeficientes de AC. El coeficiente de DC es una medida del valor medio de las 64 muestras de la imagen, y normalmente contiene una fracción importante del total de la energía de la imagen. Dado que hay una gran correlación entre coeficientes de DC de bloques adyacentes, el coeficiente de DC se codifica diferencialmente con el DC del bloque anterior en el orden de codificación de los bloques (de izquierda a derecha y de arriba ©).

Finalmente, todos los coeficientes cuantificados son ordenados siguiendo una secuencia de zig-zag como se muestra en la figura 2.18. Este orden ayuda al cuantificador entrópico agrupando los coeficientes no nulos (normalmente los de baja frecuencia) al principio y finalizando la secuencia con los coeficientes de baja frecuencia, los cuales suelen ser nulos.

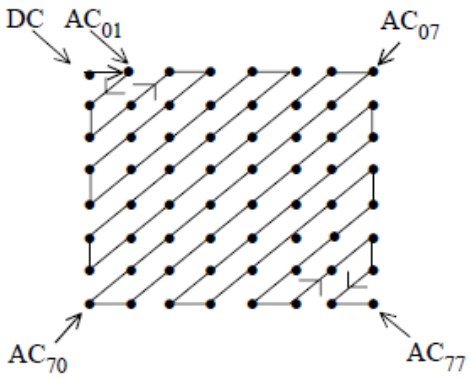


Figura 2.18 JPEG: Escaneo en zig-zag de los coeficientes AC.

Codificador Entrópico

Este es el paso que añade compresión a los datos mediante la codificación de los coeficientes DCT según sus características estadísticas. Es un proceso totalmente reversible y por lo tanto sin pérdidas. JPEG especifica dos tipos de codificadores entrópicos: codificación Huffman y codificación aritmética. El modo Secuencial usa codificación Huffman.

El codificador entrópico en JPEG tiene dos fases, la primera convierte la secuencia de coeficientes cuantificados y escaneados en zig-zag, en una secuencia de símbolos y la segunda fase convierte la secuencia de símbolos en una secuencia binaria.

La codificación Huffman requiere que la aplicación especifique una o más tablas de códigos Huffman. Las mismas tablas usadas para comprimir la imagen son necesarias para descomprimirla. Las tablas Huffman pueden estar predefinidas por defecto en la aplicación o calculadas específicamente para una imagen dada.

Por otro lado, el método particular de codificación aritmética especificado en JPEG no requiere tablas, ya que es capaz de adaptarse a las estadísticas de los datos. Aunque para algunas imágenes, la codificación aritmética produce entre un 5% y un 10% más de compresión que la codificación Huffman, para ciertas aplicaciones es más compleja.

Codificación Huffman

Para convertir la secuencia de coeficientes en una secuencia de símbolos, cada coeficiente de AC distinto de cero se representa junto con el número de coeficientes AC consecutivos iguales a cero que le preceden en la secuencia. Esta representación se hace mediante dos símbolos consecutivos, símbolo UNO y símbolo DOS. El símbolo UNO está formado por dos partes: (RUNLENGTH, SIZE), y el símbolo DOS sólo por una: (AMPLITUDE). RUNLENGTH es el número de coeficientes AC iguales a cero consecutivos que preceden al coeficiente AC actual, SIZE es el número de bits usados para codificar AMPLITUDE, y AMPLITUDE es la amplitud del coeficiente de AC actual.

RUNLENGTH varía de 0 a 15. Si en un bloque entre un coeficiente AC distinto de cero y el siguiente distinto de cero hay más de 15 coeficientes AC consecutivos nulos, se utiliza un único símbolo UNO con valor (15, 0), indicando que además de haber 15 coeficientes de AC nulos consecutivos precediendo al coeficiente actual, el propio coeficiente también es cero por lo que no se escribe el símbolo DOS. Si después del último coeficiente AC distinto de cero, todos los coeficientes AC hasta el 64 son nulos, toda esta cadena se representa con un único símbolo UNO con valor (0,0) indicando que desde ese coeficiente y hasta el 64 todos son nulos. El símbolo UNO con valor (0,0) indica el final de un bloque y para su representación se suelen utilizar las siglas EOB (acrónimo de su nombre en inglés “End Of Block”).

Del análisis numérico de la DCT directa se puede comprobar que si los valores de la señal de entrada están representados con P bits, será necesario como máximo para representar la parte entera de los coeficientes transformados $P + 3$ bits, por lo que si el rango de valores entre los que varían las muestras de entrada es $[-2^7, 2^7 - 1]$, el rango de los coeficientes AC será $[-2^{10}, 2^{10} - 1]$, tomando (SIZE) valores entre 0 y 10.

La codificación del coeficiente de DC se hace de forma similar a la de los coeficientes de AC, pero en este caso el símbolo UNO está formado sólo por una parte (SIZE) y representa el número de bits usados para codificar la amplitud del DC, y el símbolo DOS (AMPLITUDE) representa la amplitud.

Por otro lado con $P = 8$ bits, la amplitud máxima del DC nunca superará el valor 2040, por lo que SIZE tomará valores entre 0 y 11. En la tabla 2.3 se recogen los rangos de amplitud de los coeficientes con sus correspondientes valores de SIZE.

Amplitud de los Coeficientes	SIZE
0	0
−1, 1	1
−3, −2, 2, 3	2
−7, …, −4, 4, …, 7	3
−15, …, −8, 8, …, 15	4
−31, …, −16, 16, …, 31	5
−63, …, −32, 32, …, 63	6
−127, …, −64, 64, …, 127	7
−255, …, −128, 128, …, 255	8
−511, …, −256, 256, …, 511	9
−1023, …, −512, 512, …, 1023	10
−2047, …, −1024, 1024, …, 2047	11

Tabla 2.3 JPEG: Rangos de los valores de amplitud de los coeficientes DCT.

Una vez que los coeficientes cuantificados de un bloque se han representado en una secuencia de símbolos, se asignan códigos de longitud variable a dicha secuencia del siguiente modo: a cada símbolo UNO se le asigna mediante codificación Huffman y a cada símbolo DOS se le asigna mediante codificación en Complemento a 2.

Los códigos Huffman (VLCs) deben ser especificados externamente como una entrada a los codificadores JPEG. El estándar JPEG incluye un juego de tablas, pero puesto que las tablas son específicas para el tipo de aplicación, no es obligatorio su uso. Dichos códigos se representan en las tablas 2.4 y 2.5. Para la codificación de las imágenes que se lleva a cabo en este proyecto se usan las recomendadas en el estándar.

SIZE	VLC	Longitud total del coeficiente
0	00	2
1	010	4
2	011	5
3	100	6
4	101	7
5	110	8
6	1110	10
7	11110	12
8	111110	14
9	1111110	16
10	11111110	18
11	111111110	20

Tabla 2.4 JPEG: Códigos Huffman para el símbolo UNO de los coeficientes de DC.

S / R	1 6	2 7	3 8	4 9	5 10
0	00 1111000	01 11111000	100 1111110110	1011 1111111110000010	11010 1111111110000011
1	1100 1111111110000100	11011 1111111110000101	11110001 1111111110000110	111110110 1111111110000111	11111110110 1111111110001000
2	11100 111111110001010	11111001 111111110001011	1111110111 111111110001100	111111110100 111111110001101	111111110001001 111111110001110
3	111010 1111111110010001	111110111 1111111110010010	111111110101 1111111110010011	1111111110001111 1111111110010100	1111111110010000 1111111110010101
4	111011 1111111110011001	1111111000 1111111110011010	1111111110010110 1111111110011011	1111111110010111 1111111110011100	1111111110011000 1111111110011101
5	1111010 1111111110100001	11111110111 1111111110100010	1111111110011110 1111111110100011	1111111110011111 1111111110100100	1111111110100000 1111111110100101
6	1111011 1111111110101001	111111110110 1111111110101010	1111111110100110 1111111110101011	1111111110100111 1111111110101100	1111111110101000 1111111110101101
7	11111010 1111111110110001	111111110111 1111111110110010	1111111110101110 1111111110110011	1111111110101111 1111111110110100	1111111110110000 1111111110110101
8	111111000 1111111110111001	111111111000000 1111111110111010	1111111110110110 1111111110111011	1111111110110111 1111111110111100	1111111110111000 1111111110111101
9	111111001 1111111111000010	1111111110111110 1111111111000011	1111111110111111 1111111111000100	1111111111000000 1111111111000101	1111111111000001 1111111111000110
10	111111010 1111111111001011	1111111111000111 1111111111001100	1111111111001000 1111111111001101	1111111111001001 1111111111001110	1111111111001010 1111111111001111
11	1111111001 1111111111010100	1111111111010000 1111111111010101	1111111111010001 1111111111010110	1111111111010010 1111111111010111	1111111111010011 1111111111011000
12	1111111010 1111111111011101	1111111111011001 1111111111011110	1111111111011010 1111111111011111	1111111111011011 1111111111100000	1111111111011100 1111111111100001
13	11111111000 1111111111100110	1111111111100010 1111111111100111	1111111111100011 1111111111101000	1111111111100100 1111111111101001	1111111111100101 1111111111101010
14	1111111111101011 1111111111110000	1111111111101100 1111111111110001	1111111111101101 1111111111110010	1111111111101110 1111111111110011	1111111111101111 1111111111110100
15	11111111001 1111111111111001	1111111111110101 1111111111111010	1111111111110110 1111111111111011	1111111111110111 1111111111111101	1111111111111000 1111111111111110

Tabla 2.5 JPEG: Códigos Huffman para el símbolo UNO de los coeficientes AC.

2.5.2.2 Modo Progresivo (Basado En DCT)

En el modo progresivo, el proceso de división en bloques y el de la DCT directa es igual que en el modo secuencial, pero en el modo progresivo los datos de los coeficientes cuantificados de la DCT, son codificados mediante múltiples escaneos, codificando en cada escaneo una porción de los datos de los coeficientes transformados. Los datos codificados parcialmente pueden ser utilizados para obtener una imagen del mismo tamaño que la original pero con menor calidad. Cada escaneo adicional mejora la calidad de la imagen reconstruida. La codificación parcial realizada en cada escaneo se puede llevar a cabo mediante el método de selección espectral o bien mediante el método de aproximaciones sucesivas.

En el método de selección espectral, se ordenan primero los coeficientes DCT en una secuencia siguiendo un escaneo en zig-zag y se dividen en varias bandas de frecuencia. En cada escaneo se codifica una banda, siendo la banda que contiene el DC la primera.

En el método de aproximaciones sucesivas, se codifican primero los bits más significativos de los coeficientes cuantificados y en sucesivos escaneos se codifican el resto de bits menos significativos.

El objetivo de la codificación progresiva, es que el decodificador se capaz de crear rápidamente una imagen de baja calidad utilizando una pequeña parte de los datos de los coeficientes DCT, permitiendo una previsualización aproximada de la imagen, sin necesidad de procesar todos los datos codificados.

2.5.2.3 Modo Sin Pérdidas

El modo de codificación sin pérdidas, en lugar de basarse en la DCT, se lleva a cabo en el espacio mediante un esquema de codificación predictiva. En este esquema, se utilizan tres píxeles vecinos del píxel a codificar para calcular una predicción del mismo, codificando finalmente el residuo de predicción (diferencia entre el píxel original y la predicción) pero sin cuantificar, de ahí "sin pérdidas". La codificación se lleva a cabo indistintamente mediante codificación Huffman o codificación aritmética.

2.5.2.4 Modo Jerárquico

En el modo jerárquico la imagen es submuestreada espacialmente varias veces obteniendo distintas resoluciones. Excepto la imagen de menor resolución, el resto de subimágenes se codifican siguiendo un esquema de codificación predictivo, es decir, a partir de una imagen de menor resolución se predice la imagen de resolución mayor siguiente, codificando el residuo de predicción (la diferencia entre dicha imagen y su predicción). El residuo de predicción puede codificarse mediante el método basado en la DCT, o el método sin pérdidas. El modo jerárquico permite una presentación progresiva similar a la del modo progresivo, pero también es útil en aplicaciones con requerimientos multirresolución.

2.5.3 JPEG 2000

El estándar internacional JPEG 2000, desarrollado conjuntamente por la ISO y por la ITU, representa avances en la tecnología de compresión de imágenes, optimizando la eficiencia del sistema de codificación considerablemente respecto a otros estándares de codificación de imágenes, como por ejemplo JPEG. A continuación se recoge un resumen de sus principales características, las cuales se desarrollan en profundidad en [Usevitch, 2001] y [Skodras, 2001].

El funcionamiento del estándar JPEG 2000 se representa en forma de diagrama de bloques en la figura 2.19.

En el codificador, primero se aplica la transformada discreta en los datos de la imagen fuente. Los coeficientes transformados son entonces cuantificados y codificados entrópicamente formando una cadena binaria. El decodificador revierte el proceso del

codificador. El código binario primero se decodifica entrópicamente, obteniendo coeficientes transformados cuantificados, a los que se les aplica la cuantificación inversa y la transformada discreta inversa, resultando la imagen reconstruida. Aunque este diagrama de bloques general se parece al del JPEG convencional, existen grandes diferencias en los procesos de cada bloque del diagrama.

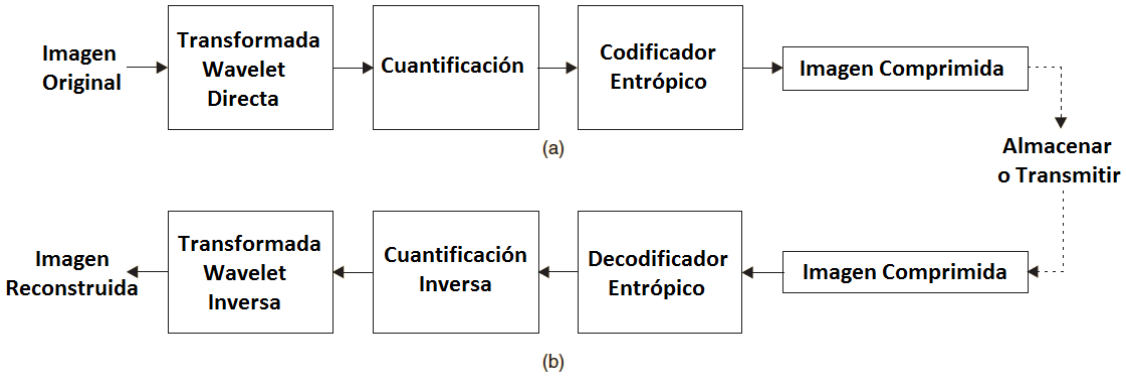


Figura 2.19 Diagrama de bloques del proceso de codificación/decodificación en JPEG 2000.

Estos son los principales pasos en el proceso de codificación de JPEG 2000:

División en *tiles*

La imagen original se divide en recuadros rectangulares sin solapar o *tiles*, los cuales son codificados independientemente como si fueran imágenes distintas. Todas las operaciones, incluyendo mezclado de componentes, transformada wavelet, cuantificación y codificación entrópica se aplican independientemente a cada uno de los *tiles*. Un *tile* se considera como la unidad básica de la imagen original así como de la reconstruida. El tamaño del *tile* no está prefijado permitiendo incluso llegar al tamaño de la imagen, pero su elección afecta tanto a la calidad objetiva de la imagen como a la calidad subjetiva, obteniéndose mejores rendimientos visuales con tamaños mayores.

Sustracción del offset de las muestras

Antes de aplicar la transformada discreta wavelet (DWT) a cada *tile*, a todas sus muestras se les sustrae la misma cantidad 2^{P-1}, donde P es la precisión usada en bits para representar los valores de las muestras de entrada, luego el rango de valores de las muestras pasa de ser $[0, 2^P - 1]$ a ser $[-2^{P-1}, 2^{P-1} - 1]$.

Transformada Wavelet

La transformada wavelet se utiliza para descomponer en niveles a un *tile*. Estos niveles contienen una serie de sub-bandas que describen las características frecuenciales espaciales, horizontales y verticales, del *tile* original.

La DWT puede ser reversible o irreversible. La transformada irreversible por defecto es implementada mediante el filtro 9/7 de Daubechies. La transformada reversible por defecto se implementa mediante un filtro 5/3 de Le Gall.

Los coeficientes entre subbandas están poco correlados y gracias a ello, los coeficientes de cada subbanda pueden ser cuantificados independientemente de los coeficientes de las otras subbandas sin que se produzca una pérdida significativa en el rendimiento. Al igual que ocurre en los coeficientes de la DCT, la varianza de los coeficientes en cada subbanda normalmente es diferente, por lo que para obtener el rendimiento óptimo en la codificación cada subbanda requiere una cantidad diferente de bits.

Cuantificación

Después de aplicar la transformada, todos los coeficientes son cuantificados. En la Parte I del estándar se utiliza cuantificación escalar. El proceso de cuantificación reduce la precisión de los coeficientes. Esta operación es con pérdidas, a no ser que el escalón de cuantificación sea 1 y los coeficientes sean enteros, que es lo que genera la wavelet entera con filtro 5/3.

El estándar JPEG 2000 soporta escalones de cuantificación diferentes para cada subbanda. Aun así, sólo se permite un escalón de cuantificación por subbanda. El rango dinámico depende del número de bits usados para representar la imagen original y en la elección de la transformada wavelet. Todos los coeficientes cuantificados son expresados en una representación de signo-magnitud para su posterior codificación.

Posición de los coeficientes significativos

Una de las propiedades de la transformada wavelet es que tiende a compactar la energía en un número pequeño de coeficientes, de hecho, para imágenes naturales, tras las transformada wavelet mucha de la energía se concentra en la banda LL_k. Además, la energía en las bandas de alta frecuencia (HL_i, LH_i, HH_i) se concentra en un número relativamente pequeño de coeficientes. Dado que sólo se necesita codificar un número reducido de coeficientes, el problema se presenta en que el codificador necesita enviar la información de la posición junto con la información de la magnitud de cada uno de estos coeficientes. Dependiendo del método usado, la cantidad de recursos requeridos para codificar la información de la posición puede suponer una fracción importante del total, restando muchos de los beneficios de la compactación de la energía. Se han propuesto varias formas de reducir el coste de codificación de la posición asociada a los coeficientes significativos. Muchos métodos están basados en la idea de la predicción entre bandas basándose en la naturaleza jerárquica de la transformada wavelet. La idea es usar la posición de coeficientes significativos en una banda de frecuencia para predecir la posición y la magnitud de coeficientes significativos en otras bandas de frecuencia, reduciendo así el coste de la codificación de la posición.

Codificación Entrópica

La codificación entrópica se lleva a cabo mediante un sistema de codificación aritmética que comprime los símbolos según un modelo de probabilidad adaptativo con 18 contextos diferentes de codificación.

2.5.4 JPEG XR

En este apartado se hace una breve descripción del nuevo estándar de compresión de imágenes conjuntamente desarrollado por la ITU y por la ISO (ITU-T T.832 | ISO/IEC 29199-2). En [Dufaux, 2009] se puede encontrar esta información con mayor detalle.

JPEG XR es un nuevo formato de compresión de imágenes que soporta imágenes con rango dinámico alto. El formato de codificación tradicional JPEG usa una profundidad de bit de 8 para cada una de los tres canales de color rojo, verde y azul (RGB), resultando 256 niveles por canal o 16.777.216 colores posibles por píxel. Sin embargo, las aplicaciones que últimamente más se demandan requieren una profundidad de bit de 16, resultando 65.536 niveles posibles por canal.

JPEG XR está diseñado para usarse en una amplia gama de aplicaciones de tratamiento de imágenes digitales y de fotografía digital. Sus principales ámbitos de actuación son estos: tecnologías de adquisición de imágenes de alta fidelidad, trabajos con imágenes de rango dinámico alto o entornos con capacidades de procesamiento de señal limitadas (como dispositivos móviles o aplicaciones embebidas).

JPEG XR está basado en un diseño de transformación de bloques, y usa algunos de los bloques funcionales típicos de la mayoría de los esquemas de compresión de imágenes, como la conversión de color, la transformación espacial, cuantificación escalar, escaneo de coeficientes y codificador entrópico.

El algoritmo da soporte de forma nativa a los tipos de color RGB y CMYK mediante conversión de estos formatos a un formato interno de luminancia-crominancia mediante el uso de una transformada de color reversible. También soporta los formatos YUV y monocromo (escala de grises).

Soporta múltiples profundidades de bits. Los formatos de 8 y 16 bits, así como otros formatos especiales, son soportados en compresión con pérdidas o sin pérdidas. El formato de 32 bits es soportado sólo para el modo con pérdidas. Mientras que para el procesamiento interno sólo se usa codificación aritmética, están soportadas la codificación sin pérdidas y con pérdidas para datos de punto fijo y de coma flotante, así como para datos enteros.

La transformada espacial convierte los datos de la imagen a una representación en el dominio de la frecuencia. La transformada es jerárquica reversible del tipo *Lapped*

Biorthogonal Transform (LBT). Esta transformada requiere un número pequeño de operaciones enteras, tanto como para codificar como para decodificar. Se puede invertir de forma exacta con aritmética entera y de hecho soporta compresión sin pérdidas. La transformada está basada en dos operadores básicos: el núcleo de la transformación y el filtro de solapamiento opcional. El núcleo de transformación es similar al de la DCT y ataca la correlación espacial mediante la división en bloques. Por el contrario, el filtro de solapamiento está diseñado para atacar la correlación en los límites (bordes) de los bloques para mitigar el efecto de bloque.

Para optimizar la cuantificación basada en la sensibilidad del sistema visual humano y en las estadísticas de los coeficientes, JPEG XR utiliza una cuantificación flexible, donde la selección de un escalón de cuantificación (QP) puede variar a través de diferentes regiones espaciales, bandas de frecuencia y canales de color. Para mejorar aún más la capacidad de compresión, se aplica predicción entre los coeficientes transformados de los bloques, para eliminar la redundancia entre bloques. El escaneo adaptativo de los coeficientes se usa para convertir la matriz bidimensional de coeficientes transformados en un vector unidimensional para ser codificado. Los patrones de escaneo son adaptados dinámicamente basándose en las estadísticas locales de los coeficientes codificados. Finalmente, los coeficientes transformados son codificados entrópicamente. Para ello se utiliza codificación de longitud variable que selecciona de forma adaptativa una tabla de códigos de longitud variable de entre un pequeño juego de tablas predefinidas, basándose en las estadísticas locales.

En el estándar sólo está definida la sintaxis de codificación y el proceso de decodificación. La figura 2.20 ilustra los varios estados del proceso de decodificación en JPEG XR, el cual básicamente revierte los pasos del proceso de codificación esperado.

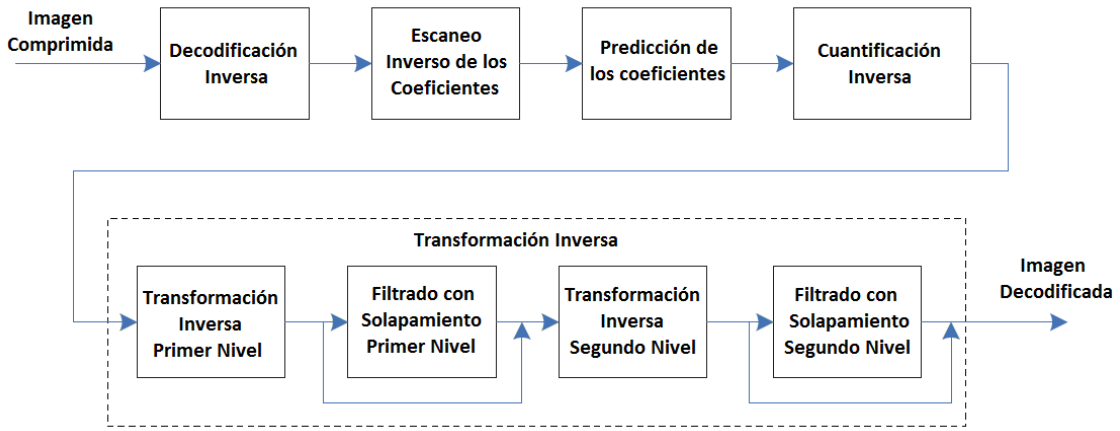


Figura 2.20 JPEG XR: Diagrama de bloques del codificador/decodificador.

2.6 CODIFICACIÓN ADAPTATIVA

La codificación eficiente de los datos se puede conseguir mediante la adaptación a las estadísticas de los mismos. En una imagen completa, las estadísticas de los datos no son constantes, sino que diferentes regiones de la imagen poseen características espaciales distintas, por lo que si el codificador es capaz de identificar estas características locales y adaptarse para codificarlas de forma óptima, se estará consiguiendo una codificación eficiente.

La adaptación se puede dar en cualquiera de los tres procesos principales del codificador típico mediante transformada: en la transformación, en la cuantificación de los coeficientes transformados o en la codificación entrópica.

Codificación adaptativa de imágenes

En [Feng, 1999] se mencionan dos algoritmos adaptativos, WUBA y WUTC:

El algoritmo *Weighted Universal Bit Allocation* (WUBA) es un codificador basado en la DCT que reemplaza la matriz de cuantificación de JPEG por una colección de matrices de cuantificación. Las matrices de cuantificación son optimizadas mediante un proceso de entrenamiento previo al proceso de codificación. El codificador elige entre las matrices de cuantificación de su colección, una para toda la imagen, o una para cada bloque. Dado que elegir entre una colección de matrices de cuantificación es más eficiente computacionalmente que diseñar una nueva matriz para cada imagen, se obtiene un buen resultado. Además, dado que el número de matrices de la colección es fijo, la descripción de la matriz de cuantificación requiere muy poca tasa, puesto que la matriz de cuantificación puede ser descrita por su índice en la lista de matrices de cuantificación.

Pero el algoritmo WUBA, al igual que JPEG, se basa únicamente en la DCT, la cual no es la transformada óptima para todos los posibles tipos de datos. En particular, mientras que la DCT consigue decorrelar satisfactoriamente los datos que pertenecen a imágenes más lisas, para imágenes que contienen altas frecuencias espaciales su rendimiento es pobre. Para aplicaciones que trabajan con conjuntos de datos más amplios, se consiguen beneficios considerables en el rendimiento usando transformadas alternativas a la DCT, como la KLT. La KLT, como ya se ha explicado en el apartado 2.3.1.3, es una transformada dependiente de los datos que consigue una decorrelación óptima, compactando al mismo tiempo la energía de los datos.

El algoritmo *Weighted Universal Transform Code* (WUTC) es un algoritmo en dos fases que añade la transformada KLT al algoritmo de codificación WUBA. El algoritmo resultante contiene una colección de parejas formadas por una KLT y una matriz de cuantificación, optimizadas para algún subconjunto de los datos de entrenamiento. Tanto la KLT como la matriz de cuantificación se diseñan de forma previa al proceso de codificación. Como en el algoritmo WUBA, el codificador puede

asignar una pareja (KLT + matriz de Cuantificación) a nivel de imagen o a nivel de bloque. Dado que la colección de KLTs, con sus matrices de cuantificación asociadas, es un número fijo, la descripción de la transformada requiere poca información puesto que cada transformada puede ser descrita por su índice en la lista de transformadas.

WUTC logra ganancias significativas sobre WUBA y sobre JPEG, aunque la mejora en el rendimiento de la tasa-distorsión se logra a expensas de una mayor complejidad computacional y unos requerimientos de memoria superiores. En particular, la implementación de WUTC sobre vectores de datos de 64 dimensiones (bloques de 8×8 píxeles como en el algoritmo JPEG) usa una colección de KLTs cuyas matrices de transformación son de tamaño 64×64.

Codificación Adaptativa de Vídeo

En [Gao, 2008], utilizan una transformada KLT especializada para cada uno de los 9 modos de predicción Intra que posee el estándar de codificación de vídeo H264.

El modo Intra 4×4 en H264, está basado en la predicción de cada bloque 4×4 en el dominio espacial, tomando como referencias las muestras decodificadas de los bloques vecinos, que se encuentran a la izquierda y encima del bloque que se está prediciendo. Dependiendo de la dirección espacial de las muestras en el bloque, existen nueve modos posibles de predicción Intra. Una vez que se le ha asignado un modo de predicción Intra a un bloque 4×4, se calcula la predicción de dicho bloque, y se codifica la diferencia con el bloque real o residuo. El estándar H264 utiliza la DCT para compactar la energía de los residuos que se obtienen de la predicción espacial, pero como los residuos con diferente modo de predicción poseen características frecuenciales diferentes, el rendimiento en la decorrelación por parte de la DCT es poco satisfactorio. El uso de la KLT en este caso es preferible al de la DCT, pero para que la matriz de transformación KLT sea óptima, o bien se entrena antes para un tipo de datos en concreto, o se calcula durante el proceso de codificación y se comunica al decodificador.

En este documento, se basan en que las distribuciones de energía de los bloques residuales con el mismo modo, son relativamente coincidentes, lo cual hace posible entrenar una matriz KLT universalmente buena para cada uno de los nueve modos de predicción. En el documento proponen un algoritmo *optimal frequency matching* (OFM) para entrenar las matrices KLT universales. Primero almacenan un gran número de bloques residuales de diferentes secuencias de video. Para cada modo, seleccionan aleatoriamente 100.000 bloques residuales y los vectorizan, para seguidamente concatenarlos y obtener un vector cadena. Después, mediante la aplicación de la transformada de Fourier, extraen las características de la distribución de la energía del vector cadena. La matriz KLT óptima para cada modo es finalmente seleccionada usando un algoritmo OFM, y en el que de forma adicional, y en pos de obtener el mejor rendimiento posible en la decorrelación, permutan cada matriz KLT candidata. Finalmente, la matriz KLT seleccionada para cada modo, será la matriz KLT permutada

que concentre la mayoría de la energía en las bandas de más baja frecuencia, ya que consideran que los coeficientes de las bandas de más baja frecuencia son más importantes para la codificación entrópica.

Sus experimentos demuestran que el rendimiento en la decorrelación con matrices KLT entrenadas previamente es persistente y mucho mejor que en la DCT.

CAPÍTULO 3 - CODIFICACIÓN DE IMÁGENES CON TRANSFORMADAS ADAPTATIVAS

3.1 INTRODUCCIÓN

A lo largo del capítulo anterior se ha tratado de explicar las bases en las que se sustenta la teoría de codificación de imágenes mediante transformada, así como de presentar un resumen de los estándares de codificación actuales.

En este capítulo, se expone minuciosamente el proceso de codificación multitransformada diseñado y se desarrollan los procesos creados para la obtención de las matrices KLT a partir de los bloques de una imagen. Para ello, el capítulo consta de cinco partes.

En la primera parte se hace una breve descripción del funcionamiento del codificador, mostrando las dos etapas que posee y las fases de las que están formadas dichas etapas, así como de los parámetros de configuración externos al codificador (número de KLTs y tamaño de bloque).

En la segunda parte se detallan los procesos que intervienen en la obtención de las matrices KLT, explicando el método creado para caracterizar espacialmente a los bloques y el algoritmo de agrupamiento utilizado para localizar los bloques con características espaciales similares.

En la tercera parte se explica la forma utilizada para evaluar un conjunto de transformadas. El objetivo es obtener una medida que refleje de forma aproximada, el comportamiento que tendría el codificador al usar un conjunto de transformadas concreto, sin necesidad de realizar los procesos de codificación y decodificación completos utilizando dicho conjunto de transformadas.

En la cuarta parte se plasman las modificaciones introducidas en los procesos de obtención de las matrices KLT intentando por un lado, aumentar la especialización de las KLTs mejorando la caracterización espacial de los bloques y por otro lado, optimizar el algoritmo de agrupamiento reduciendo el número de bloques que participan en el proceso de creación de las KLTs.

Por último, en la quinta parte, se desarrolla el procedimiento completo de codificación usando el conjunto de transformadas formado por las KLTs obtenidas junto con la DCT.

3.2 DESCRIPCIÓN DEL CODIFICADOR ADAPTATIVO

El codificador adaptativo consta de dos etapas diferenciadas: Etapa I y Etapa II. El propósito de la Etapa I es obtener el juego de matrices de transformación KLT especializadas, mientras que la función de la Etapa II consiste en la propia codificación de la imagen, pero usando para la fase de transformación el juego de matrices obtenido en la Etapa I junto con la DCT.

3.2.1 ETAPA I

Como se ha comentado, el objetivo de esta etapa es, una vez fijado el número de matrices KLT que se quieren utilizar en la Etapa II, identificar un número igual de características espaciales en la imagen y obtener para cada una de ellas una matriz de transformación KLT especializada. Para ello, esta etapa se subdivide en 4 fases: fase de División en Bloques, fase de Extracción de Características, fase de Agrupamiento y fase de Cálculo de KLTs. En la figura 3.1 se representa el diagrama de bloques de la Etapa I.

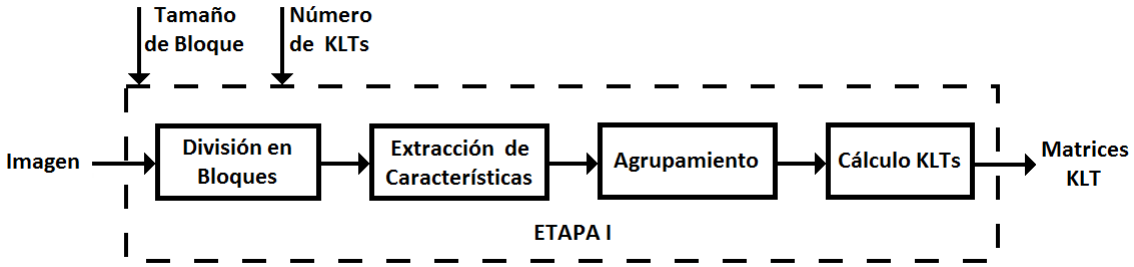


Figura 3.1 Representación en diagrama de bloques de la Etapa I del codificador adaptativo.

En primer lugar, con el objetivo de identificar las estructuras espaciales locales existentes en la imagen, se divide la imagen en regiones cuadradas o bloques de $N \times N$ píxeles, cuyo tamaño se considera un parámetro de configuración por lo que vendrá impuesto por el usuario o la aplicación.

Una vez que se ha dividido la imagen en bloques de $N \times N$ píxeles, el siguiente paso en la Etapa I es la identificación de las características espaciales que presentan los bloques. Esta identificación se hace de forma indirecta en el dominio de la frecuencia, por lo que en esta fase, cada bloque de $N \times N$ píxeles es transformado al dominio de la frecuencia mediante la DCT directa bidimensional. Una vez en este dominio, se subdivide el espectro de la DCT en bandas de frecuencia y se calcula la energía de cada una de estas bandas. Son los valores normalizados de las energías asociadas a las bandas de un bloque, los que se utilizan para caracterizar espacialmente a dicho bloque.

La siguiente fase de la Etapa I es la del agrupamiento de los bloques con características espaciales similares. Para ello, se usa el algoritmo de agrupamiento K-Means utilizando las distribuciones energéticas de cada bloque a lo largo de las bandas de frecuencia especificadas en la fase anterior. El número de grupos que se obtienen de esta fase es igual al número de transformadas KLT asignado.

Finalmente, en la fase de Cálculo de KLTs, para cada uno de los conjuntos de bloques obtenidos en la fase anterior de Agrupamiento, se calcula la matriz de covarianza de los píxeles que constituyen dichos bloques y a partir de la misma, se obtiene una matriz de transformación KLT especializada para las características espaciales predominantes en los bloques de ese grupo, tal y como se ha explicado en el apartado 2.3.1.3.

3.2.2 ETAPA II

En esta etapa se lleva a cabo la codificación de la imagen mediante un algoritmo similar al utilizado en el modo secuencial del estándar JPEG, pero con la principal diferencia de que en la fase de Transformación, la transformada no está restringida únicamente a la DCT, sino que existe un conjunto de transformadas candidatas formado por las KLTs obtenidas en la Etapa I junto con la DCT. Para poder seleccionar del conjunto disponible la transformada óptima para cada bloque, se han modificado las fases de Transformación y Cuantificación del estándar JPEG y se ha añadido una nueva fase denominada Selección. La figura 3.2 muestra el diagrama de bloques que representan las fases de la etapa de codificación.

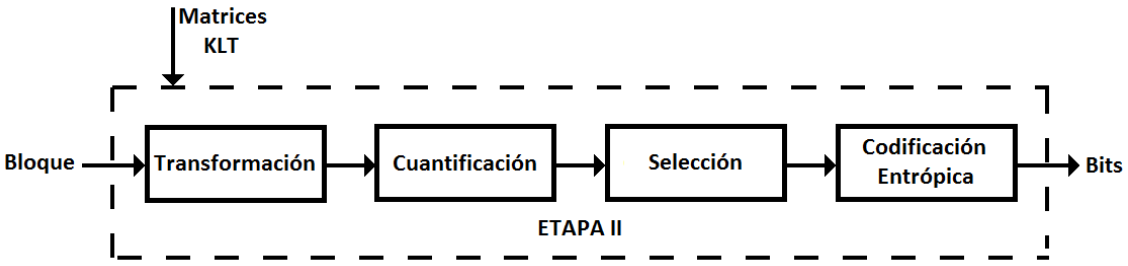


Figura 3.2 Representación en diagrama de bloques de la Etapa II del codificador adaptativo.

Las funciones de las fases de Transformación y Cuantificación de nuestro codificador son exactamente las mismas que las del codificador JPEG, es decir, mediante una transformación lineal compactar la energía del bloque en unos cuantos coeficientes transformados y mediante la cuantificación escalar individual de los coeficientes, reducir el rango de aquellos que resulten energéticamente significativos y anular los coeficientes con poca energía. Pero a diferencia del estándar JPEG en el que los bloques son transformados exclusivamente por la DCT, en la fase de

Transformación de nuestro codificador, cada bloque es transformado con cada una de las matrices disponibles del conjunto de transformadas candidatas (formado por las matrices KLT resultantes de la Etapa I junto con la matriz de transformación de la DCT), resultando así un número de bloques transformados distintos igual al número de transformadas candidatas, los cuales son transferidos en su totalidad a la fase de Cuantificación.

En la fase de Cuantificación, a los coeficientes de cada uno de los bloques transformados procedentes de la fase de Transformación, se les aplica una cuantificación escalar individual, obteniendo un número de bloques cuantificados distintos igual al número de transformadas candidatas, los cuales son transferidos íntegramente a la fase de Selección. Al igual que en el estándar JPEG, los distintos escalones de cuantificación pueden ser especificados externamente al codificador, aunque en el desarrollo de las pruebas de este proyecto siempre se utilizan las tablas de cuantificación usadas en la recomendación CCIR-601 mostradas en la tabla 2.2.

La fase de Selección es la encargada de elegir la transformada óptima para el bloque que se está codificando de entre las transformadas candidatas. Para ello, invierte los procesos de cuantificación y transformación en todos los bloques cuantificados procedentes de la fase anterior de Cuantificación, obteniendo un número de reconstrucciones del bloque original igual al número de transformadas candidatas, las cuales serán diferentes entre sí y por lo tanto presentarán niveles de distorsión con respecto al bloque original distintos. La reconstrucción que presente menor nivel de distorsión indicará la transformada que finalmente se le asigne a dicho bloque. Una vez identificada la transformada ganadora, los coeficientes cuantificados obtenidos a partir de ella, junto con la señalización de dicha transformada, son entregados a la fase de Codificación Entrópica.

La fase de Codificación Entrópica del codificador propuesto, es muy similar a la fase de Codificación Entrópica del modo secuencial del estándar JPEG, cuyo objetivo es añadir compresión adicional sin pérdidas mediante la codificación de los coeficientes basada en las estadísticas de aparición de los mismos. Al igual que en el modo secuencial de JPEG, el tipo de codificación entrópica que se usa en esta fase es la Codificación Huffman, la cual se realiza en dos pasos: un primer paso en el que la secuencia de coeficientes cuantificados se convierte en una cadena de símbolos y un segundo paso en el que cada uno de los símbolos es codificado mediante códigos de longitud variable, o códigos Huffman, asociando las palabras de código más cortas a los símbolos que ocurren más frecuentemente y las palabras de código más largas a los símbolos que ocurren con menor frecuencia. El estándar JPEG incluye un juego de tablas de códigos Huffman, aunque su uso no es obligatorio. Aun así, para todas las pruebas de este proyecto se ha decidido usar las tablas recomendadas en el estándar (tablas 2.4 y 2.5), tanto si la transformación de los coeficientes se llevan a cabo mediante la DCT como si se hace mediante la KLT.

3.2.3 PARÁMETROS EXTERNOS AL CODIFICADOR: NÚMERO DE KLTS Y TAMAÑO DE BLOQUE

En el contexto en el que nos encontramos, el número de transformadas KLT a utilizar en la codificación es variable ya que depende de varios factores como son: el tipo de imagen a codificar, la calidad con la que se esté trabajando y los requisitos de tasa (en el caso de que los hubiese).

La relación entre el número de transformadas KLT y el tipo de imagen es directa, ya que el número de características espaciales locales presentes en una imagen, puede variar considerablemente de un tipo de imagen a otro, por lo que las imágenes más complejas que posean una gran variedad de características espaciales diferentes, requerirán un mayor número de transformadas KLT especializadas.

Por otro lado, la calidad con la que se esté trabajando también es un factor determinante para seleccionar el número de KLTs. El grado de calidad depende de la cantidad de información eliminada tras el proceso de cuantificación, ya que cuanto mayor sea la cantidad de información eliminada, más diferirá la imagen recuperada con respecto de la original y menor calidad se obtendrá. La mayor o menor pérdida de información en el proceso de cuantificación, además de depender de los escalones de cuantificación utilizados, depende en gran medida de la capacidad de compactación de la energía de la transformada usada. Cuanto más especializada esté una transformada a los datos que va a transformar, mayor será la compactación de la energía que provoque y por lo tanto, menor será la pérdida de información que resulte tras la cuantificación. De esta forma, cuando se trabaje en calidades altas, se deberá utilizar una transformada KLT especializada para cada una de las características espaciales predominantes en la imagen y por el contrario, cuando se esté trabajando con calidades más bajas, se podrá utilizar una misma transformada para varias de estas características, resultando transformadas más genéricas.

El último factor del que depende el número de KLTs es la tasa binaria. Debido a que una matriz de transformación KLT para bloques de 8×8 píxeles posee un tamaño de 64×64 elementos y a que es imprescindible que las mismas matrices que utiliza el codificador se envíen al decodificador para que pueda revertir el proceso y recuperar la imagen original, cuanto más restrictivos sean los requisitos de tasa, menor número de matrices KLT se podrá utilizar ya que codificadas resultan en un tamaño elevado.

Por estos motivos, el número de matrices KLT que se usan para codificar una imagen, se considera un parámetro de configuración del codificador que debe ser indicado por el usuario o por el tipo de aplicación, en función de los factores mencionados.

Otro parámetro de configuración que vendrá impuesto por el usuario o la aplicación, es el tamaño de bloque que se utiliza para identificar las características espaciales locales existentes en la imagen. Como ya se ha comentado en el apartado 2.3.1.4, el tamaño de bloque debe de ser lo suficientemente grande como para contener

el mayor número de píxeles altamente correlados, pero no tan grande como para que en la región de un bloque aparezcan varias características espaciales diferentes, ya que esto perjudicaría a la adaptación de la transformada. El tamaño de bloque que se ha considerado para el codificador adaptativo propuesto es un tamaño de 8×8 píxeles ya que en imágenes, los píxeles contenidos en bloques de este tamaño generalmente están altamente correlados entre sí y a partir de este tamaño dicha correlación disminuye y además es el tamaño de bloque adoptado en JPEG.

3.3 OBTENCIÓN DE LAS MATRICES KLT

La obtención de las matrices KLT que formarán parte del juego de transformadas candidatas en la fase de Transformación Selectiva de la Etapa II, tiene lugar durante la Etapa I del codificador. Como se ha comentado en el punto 3.2.1, el objetivo de la Etapa I es identificar un número determinado de características espaciales predominantes en la imagen y obtener para cada una de estas características una matriz KLT lo más especializada posible, para que de esta forma la eficiencia en la codificación se vea mejorada con respecto a la del estándar JPEG, en el que únicamente se utiliza la DCT independientemente de las características espaciales presentes en la imagen.

3.3.1 ESPECIALIZACIÓN DE LAS MATRICES KLT

La eficiencia de un codificador de imágenes mediante transformación con pérdidas depende principalmente de dos factores: del nivel de compresión que aplica a los datos y del nivel de distorsión que presenta la imagen reconstruida tras la decodificación de los datos comprimidos. Ambos factores están directamente relacionados, ya que la compresión es mayor cuantos más coeficientes transformados resulten nulos tras la cuantificación, pero cuanta más energía se haya eliminado de forma irreversible al hacer nulos dichos coeficientes, mayor será la distorsión que presente la imagen reconstruida. Por todo ello, para que la eficiencia del codificador aumente, será necesario utilizar una transformada que sea capaz de compactar la mayor parte de la energía del bloque en un número muy reducido de coeficientes transformados, para de esta forma poder eliminar la energía del resto de coeficientes sin que repercuta en un aumento considerable del nivel de distorsión de la imagen reconstruida.

El grado de compactación de la energía que una KLT puede aplicar sobre un bloque, depende directamente de cuan especializada está su matriz de transformación a las características espaciales del bloque. Para obtener una KLT especializada para un bloque con un tipo de características espaciales en concreto, su núcleo de transformación debe ser calculado a partir de un conjunto de bloques que compartan dichas características espaciales. De este modo, la matriz KLT resultante será capaz de compactar eficientemente la energía de los bloques que han participado en la obtención

de la KLT, así como la de bloques que no hayan participado pero que compartan dichas características espaciales. Sin embargo, si se utiliza para calcular la matriz KLT un conjunto de bloques en el que no todos comparten las mismas características espaciales, la especialización de la KLT resultante disminuirá y con ella la compactación de la energía que produce en esos bloques.

En definitiva, para poder obtener una matriz KLT especializada hay que utilizar un conjunto de bloques con características espaciales similares, es decir, la correlación existente entre los píxeles propios de un bloque, debe ser similar a la que presentan los píxeles de cada bloque del conjunto.

Por todo lo anterior se puede afirmar que el agrupamiento de los bloques según la correlación de sus píxeles, es un factor determinante para la eficiencia del codificador adaptativo propuesto, por lo que habría que utilizar un método para agrupar aquellos bloques que posean correlaciones similares en la imagen. En cierto modo, si las correlaciones de los píxeles de dos bloques son similares, se podría decir que la correlación entre estos dos bloques es alta, por lo que si se pudiera hallar la correlación entre los bloques de una imagen, se podrían agrupar los que estuvieran altamente correlados. Pero el cálculo de estas correlaciones prácticamente es inviable para un codificador de imágenes ya que, para el caso de bloques de 8×8 píxeles, cada uno de los bloques se consideraría como una variable de 64 dimensiones y el número de variables total sería igual al número de bloques que forman la imagen, por lo que el cálculo de la matriz de correlaciones requeriría una capacidad de cómputo excesiva.

Otro método que se podría utilizar directamente con los bloques sería aplicar un algoritmo de agrupamiento. Para ello, los bloques se considerarían como vectores de 64 dimensiones y se aplicaría un procedimiento de agrupamiento a los vectores de acuerdo con un criterio de cercanía. Por ejemplo, dicha cercanía se podría definir en términos de una determinada función de distancia como la euclídea. Pero igual que en el caso anterior, al trabajar con un número de dimensiones tan elevado, el cómputo necesario es muy alto, llegando a ser excesivo para nuestro codificador.

Una posible técnica de clasificar los bloques en el dominio espacial que no necesita tanto cómputo como el cálculo de la matriz de correlación de todos los bloques de la imagen, es la Predicción Intra que se realiza en el estándar de codificación de vídeo H-264. En este estándar se definen nueve modos de predicción diferentes para un bloque de 4×4 píxeles (figura 3.3). La forma de predecir un bloque se hace utilizando los píxeles adyacentes de los bloques codificados anteriormente. Cada modo utiliza los píxeles vecinos de una forma diferente para predecir el bloque mediante una determinada dirección de extrapolación o interpolación. Hay un modo que predice el bloque mediante una dirección vertical, otro modo mediante una dirección horizontal, otro con una estructura plana y el resto con direcciones diagonales de diferentes ángulos. La predicción que más se aproxime al bloque original indicará el modo de Predicción Intra seleccionado, clasificando implícitamente al bloque en una de estas 9 direcciones posibles.

En principio, utilizar la Predicción Intra para clasificar los bloques según su estructura espacial, podría parecer una técnica apropiada para usar en nuestro codificador, ya que se podrían clasificar fácilmente los bloques de una imagen dentro del conjunto de estas nueve direcciones y así, obtener nueve KLTs, cada una especializada para una dirección en concreto. Incluso se podrían definir otros tipos de direcciones diferentes a las de la Predicción Intra, que fueran más acordes con las características espaciales presentes en algún tipo de imagen determinado.

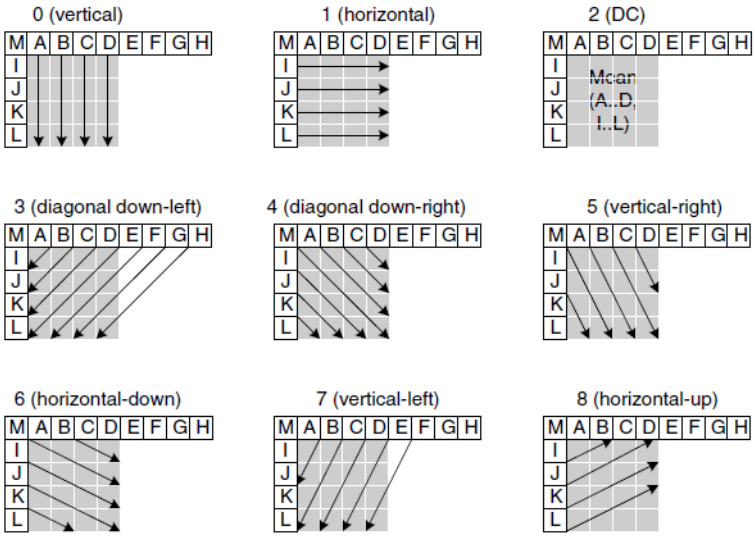


Figura 3.3 Modos de predicción Intra del estándar H-264 [Richardson, 2003].

Sin embargo, dado que el objetivo de este proyecto es aprovechar la propiedad de adaptabilidad de la transformada KLT a los datos y aumentar así la eficiencia en el codificador de imágenes propuesto, no podemos basarnos exclusivamente en unas características espaciales concretas, sino que hay que utilizar un método para identificar las características espaciales que presentan los bloques de la imagen y agrupar los bloques según estas características y el número de KLTs que se deseen usar, para finalmente obtener las correspondientes KLTs especializadas.

Dado que analizar la correlación entre los bloques en el dominio espacial nos obliga a trabajar con variables de 64 dimensiones y aplicar un algoritmo de agrupamiento en el mismo dominio nos obliga a trabajar también con vectores de 64 dimensiones, sería conveniente reducir el número de dimensiones para que disminuya la cantidad de cómputo necesaria para agruparlos según su correlación espacial. La técnica que se ha usado finalmente en este proyecto ha sido la de extraer de cada bloque un número reducido de características que le definen espacialmente y aplicar un algoritmo de agrupamiento utilizando dichas características. Para ello, cada bloque es transformado linealmente mediante la DCT y es en el dominio de la DCT en el que se extraen las características que se utilizarán en el algoritmo de agrupamiento.

3.3.2 TRANSFORMACIÓN DEL ESPACIO Y REDUCCIÓN DIMENSIONAL

Como se detalla en el punto 2.2.1.2, la transformada bidimensional de una imagen de $N \times N$ píxeles se puede interpretar como una descomposición en imágenes base de $N \times N$ píxeles. La figura 3.4 muestra el conjunto de imágenes base para el caso de la DCT y de imágenes de 8×8 píxeles (bloques).

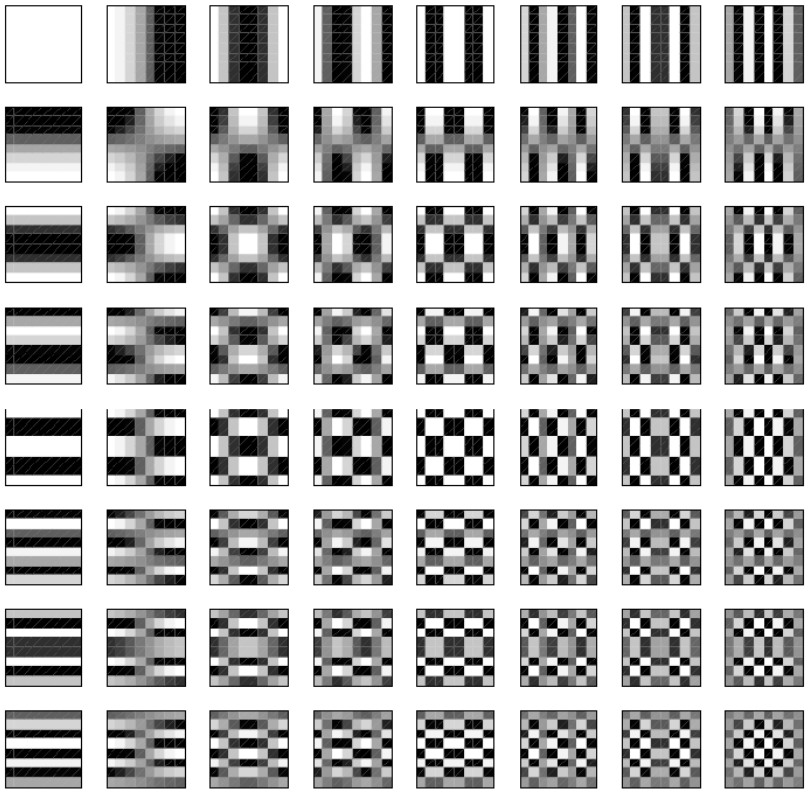


Figura 3.4 Imágenes base de la DCT bidimensional 8×8 [Richardson, 2003].

Como se puede apreciar en la figura 3.4, el valor de los píxeles dentro de cada imagen base varía de forma diferente. Según nos desplazamos por las imágenes base desde arriba hacia abajo, la frecuencia con la que varían los píxeles en sentido vertical aumenta y según nos desplazamos desde la izquierda hacia la derecha también aumenta la frecuencia con la que varían los píxeles pero de forma horizontal. Dependiendo del desplazamiento hacia la derecha y hacia abajo, cada imagen base tiene unas frecuencias de variación vertical y horizontal diferentes.

La DCT lo que hace es aislar las diferentes frecuencias bidimensionales (verticales y horizontales) que contiene una imagen, que para el caso de imágenes de 8×8 píxeles, son las mostradas en la figura 3.4. De esta forma, tras transformar un bloque de $N \times N$ píxeles mediante la DCT directa bidimensional, se obtiene un bloque de $N \times N$ coeficientes, donde cada coeficiente indica la cantidad presente en la imagen

de cada una de estas frecuencias bidimensionales, luego a partir de los coeficientes DCT y de las imágenes base, es posible obtener el bloque original mediante combinación lineal de las mismas, en donde los pesos de cada una de las imágenes serán los coeficientes.

El coeficiente DC, que está asociado a la imagen base plana en la que el valor de los píxeles no varía, realmente no contribuye a las características espaciales del bloque, sino a la luminosidad media de sus píxeles. Son los coeficientes AC los que verdaderamente aportan información sobre las características espaciales de un bloque. Las imágenes base asociadas a los coeficientes AC con mayor energía, serán las que contribuyan en mayor medida a las características espaciales del bloque imagen y las imágenes base asociadas a los coeficientes AC con menor valor no contribuirán apenas, aportando únicamente los detalles más finos.

Por todo lo anterior, podemos asegurar que hay una relación directa entre las características espaciales de un bloque y el valor de sus coeficientes DCT, por lo que para agrupar bloques con características espaciales similares, habrá que examinar la distribución energética de los coeficientes DCT de dichos bloques y agrupar aquellos que tengan distribuciones similares. En este punto, nos encontramos con el mismo problema dimensional que surgía a la hora de agrupar en el dominio del espacio, los bloques con correlaciones entre sus píxeles similares, ya que todavía el número de dimensiones se sigue manteniendo en 64. Pero gracias a la distribución particular de los coeficientes DCT que suele darse para la mayoría de los bloques de una imagen, seremos capaces de reducir las dimensiones a un número mucho menor.

Distribución de los coeficientes DCT

Normalmente, en los bloques de una imagen, los valores de los píxeles varían de forma lenta de un punto a otro, es decir, predominan las frecuencias espaciales más bajas. Es por ello que al aplicar la DCT, se obtendrá un bloque de coeficientes en el que, típicamente, los más significativos energéticamente hablando serán los cercanos al coeficiente de DC, que son los correspondientes a las bajas frecuencias, mientras que los más alejados serán cero o próximos a cero. Para comprobarlo visualmente, en la figura 3.5 se muestra la distribución energética normalizada media de los bloques DCT de todas las imágenes del set utilizado para las pruebas de este proyecto.

Tal y como se puede apreciar en la figura 3.5, el coeficiente de DC es el que más porcentaje de energía contiene ya que supone en torno al 96 % del total, distribuyéndose el resto de la energía en los coeficientes AC, conteniendo más energía los más cercanos al DC (bajas frecuencias) y disminuyendo según nos alejamos del DC hacia los coeficientes de alta frecuencia. Según esta distribución podemos concluir que en imágenes, los coeficientes AC de más baja frecuencia (cercanos al DC) son los que más contribuyen a las características espaciales del bloque, mientras que según nos alejamos del DC (coeficientes de media-alta frecuencia) la contribución disminuye, haciéndolo más rápidamente cuanto más alta es la frecuencia.

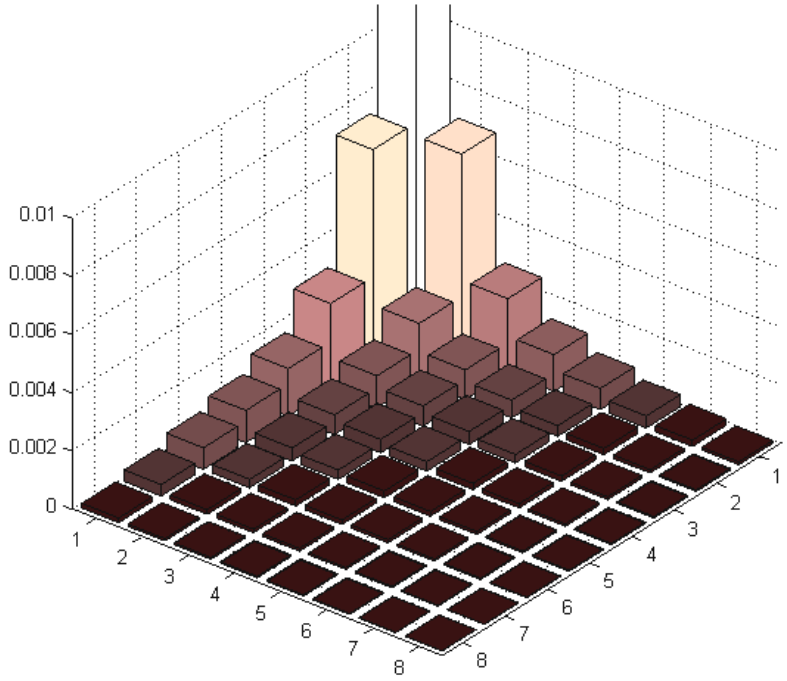


Figura 3.5 Distribución energética normalizada media de los bloques DCT de las imágenes.

Según la distribución energética típica en los bloques DCT de imágenes, podemos dividir el espectro DCT en tres zonas de forma que la primera contenga a los coeficientes que contribuyen en mayor grado a las características espaciales del bloque (coeficientes de más baja frecuencia), otra zona con los coeficientes que contribuyen en un grado medio (coeficientes de baja-media frecuencia) y por último la zona de coeficientes que menos contribuyen (coeficientes de alta frecuencia). En la figura 3.6 se pueden apreciar estas divisiones, mostrando en rojo la zona de coeficientes de baja frecuencia, en verde la zona de coeficientes de media frecuencia y en azul la zona de coeficientes de alta frecuencia.

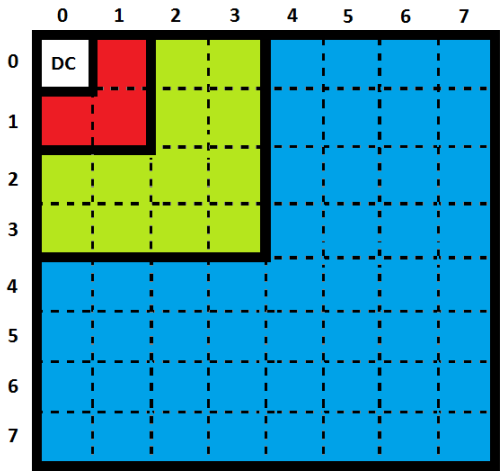


Figura 3.6 División del espectro DCT en zonas.

Una posible forma de caracterizar espacialmente a un bloque podría ser mediante el uso de estas zonas ya que, por ejemplo, podríamos calcular la energía que suman los coeficientes en cada zona y utilizar los tres valores resultantes para dicha caracterización. Pero estos tres valores en concreto no serían válidos para nuestro propósito de identificar bloques con características espaciales similares, ya que no se estaría haciendo ninguna distinción entre frecuencias horizontales y verticales, lo cual no es correcto ya que no poseen las mismas características espaciales un bloque en el que predominen las frecuencias verticales, que otro en el que predominen las horizontales, aunque ambas frecuencias pertenezcan a la misma zona. Por ello, se ha decidido usar esta forma de caracterizar espacialmente a los bloques, pero haciendo distinción entre frecuencias horizontales y verticales, para lo cual, cada zona se ha dividido en tres bandas de frecuencia de igual tamaño, de forma que una contiene los coeficientes asociados a las frecuencias verticales, otra los asociados a las horizontales y una tercera intermedia que se podría decir que contiene los coeficientes asociados a las frecuencias diagonales. En la figura 3.7 se muestran las nueve bandas.

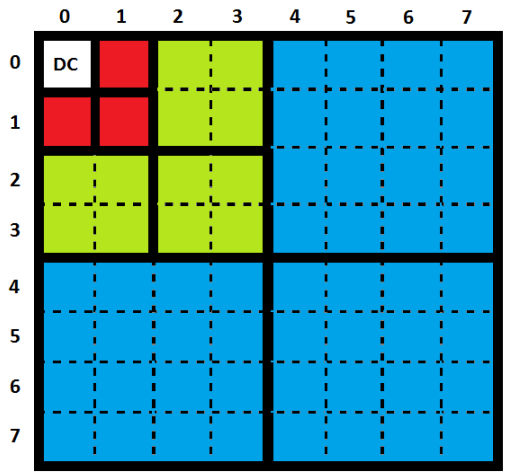


Figura 3.7 Bandas de frecuencia usadas para caracterizar espacialmente a los bloques.

Tal y como se puede apreciar en la figura 3.7, el número de coeficientes que contiene cada banda de frecuencia depende de la zona a la que pertenece: si la banda pertenece a la zona de los coeficientes que más contribuyen a las características espaciales del bloque (bajas frecuencias) únicamente contiene un coeficiente; si por el contrario la banda pertenece a la zona de los coeficientes que menos contribuyen (altas frecuencias) la banda contiene 16 coeficientes. Y por último, si la banda pertenece a la zona de coeficientes con contribución media (frecuencias medias) la banda contiene 4 coeficientes.

Una vez definidas las nueve bandas de frecuencia, podemos extraer de cada una un valor o característica que sirvan para caracterizar espacialmente al bloque. En nuestro caso, se ha utilizado la energía contenida en cada banda pero normalizada por la

energía total de las nueve bandas. Esta normalización es necesaria puesto que, aunque dos bloques contengan distinta cantidad de energía, pueden poseer características espaciales similares si la distribución de la energía en sus coeficientes DCT es parecida. Por este motivo, al normalizar por la energía contenida en todas las bandas, obtenemos valores que siguen aportando información sobre la distribución de la energía pero que son independientes de la cantidad de energía. Por la misma razón, en este proceso se obvia la energía contenida en el coeficiente DC, porque su valor no aporta información sobre las características espaciales del bloque, sino de la luminosidad media de sus píxeles y dos bloques pueden tener características espaciales similares independientemente de la luminosidad media de sus píxeles.

Con las 9 características que se extraen de cada bloque se forma un vector (vector de características) y a partir del conjunto total de vectores y mediante un algoritmo de agrupamiento, se pueden clasificar todos los bloques de la imagen según un número determinado de clases, donde el número de clases se corresponde con el número de KLTs que se desea utilizar y cada una de las clases se corresponde con un tipo de características espaciales diferente.

3.3.3 Algoritmo De Agrupamiento: K-Means

El objetivo del agrupamiento es, dado un conjunto de puntos en un espacio n-dimensional, dividirlo en k grupos de forma que los puntos dentro de un grupo guarden una mayor similitud entre sí que con los puntos de los otros grupos.

Existen multitud de algoritmos de agrupamiento englobados en varios tipos, pero en nuestro caso nos interesan los algoritmos particionales que son los que realizan una división inicial de los datos en grupos y luego, mueven los objetos de un grupo a otro según se optimice una función objetivo. Estos algoritmos asumen un conocimiento a priori del número de grupos en que debe ser dividido el conjunto de datos, que en nuestro caso es el número de KLTs que se define de forma externa a nuestro codificador. Dentro de los algoritmos de agrupamiento particionales se encuentra el algoritmo K-Means, que es uno de los más simples y de los más extendidos, razones por las que se utiliza en este proyecto para agrupar los vectores de características de los bloques.

El algoritmo K-Means trata a los vectores n-dimensionales como puntos en un espacio euclídeo e inicialmente, clasifica de forma aleatoria cada punto con una de las posibles clases, para calcular después el centro geométrico de los conjuntos de puntos obtenidos de cada clase, denominado centroide. Una vez situados los centroides, pasa a calcular la distancia de todos los puntos a todos los centroides, asignando a cada punto el centroide más cercano y recalculando de nuevo los centroides según los grupos resultantes. Iterativamente, se van actualizando los centroides en función de las asignaciones de puntos a los grupos, hasta que los centroides dejan de cambiar.

La figura 3.8 muestra el funcionamiento del K-Means con un conjunto de puntos de dos dimensiones que se desea dividir en tres grupos, consiguiéndolo tras 6 iteraciones.

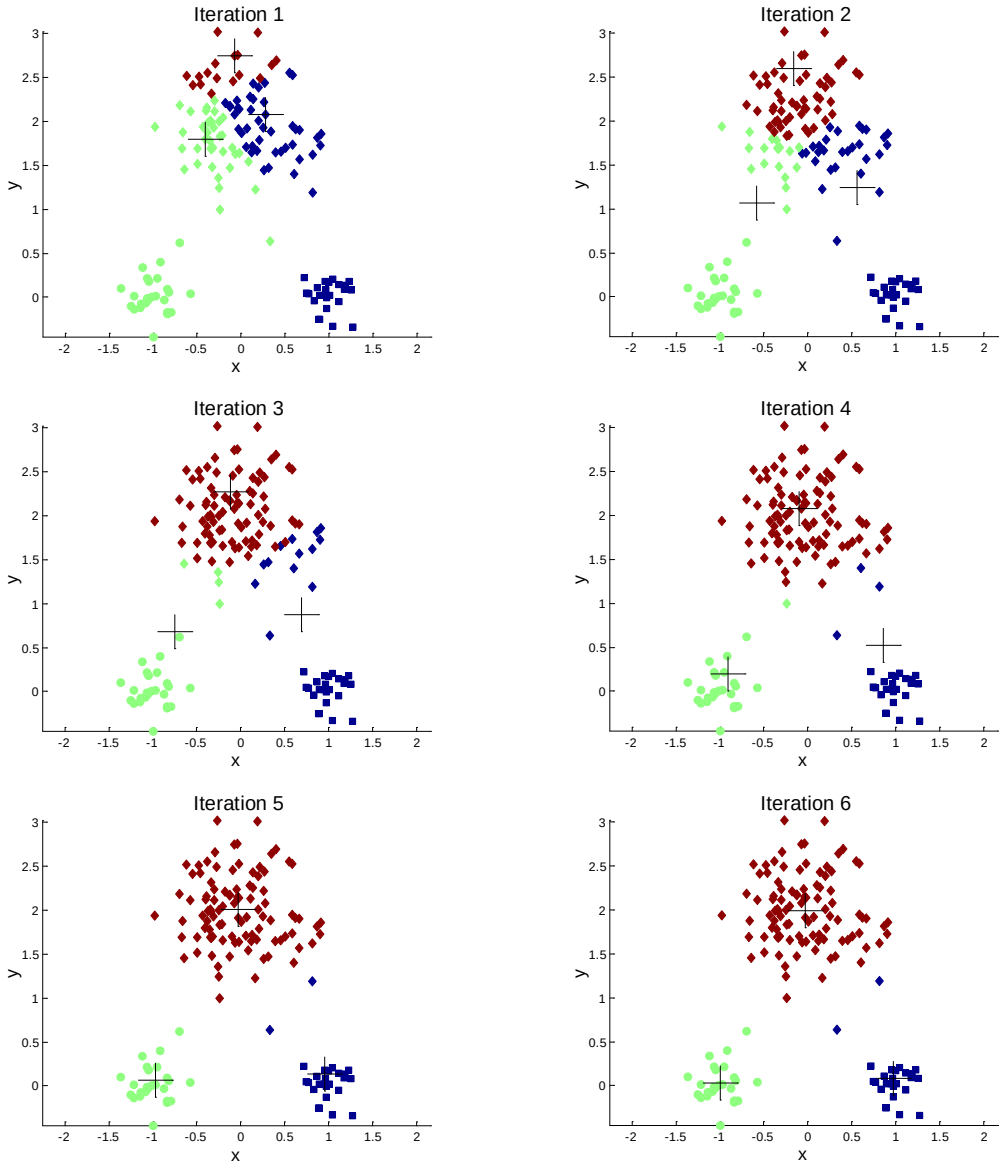


Figura 3.8 Ejemplo de agrupamiento de puntos bidimensionales con el algoritmo K-Means

A pesar de que el algoritmo de agrupamiento K-Means es un algoritmo sencillo, computacionalmente no muy pesado y que con él se obtienen resultados de agrupamiento óptimos para un gran número de escenarios diferentes, posee la desventaja de no asegurar que los centroides finales sean los óptimos, entendiendo como centroides óptimos aquellos con los que la suma total de las distancias de cada punto a su centroide es la mínima. Esto es así debido al carácter aleatorio de la inicialización de los centroides y a que el algoritmo, debido a su sencillez, no está provisto de herramientas para evitar acabar en un óptimo local. Por este motivo, al

aplicar el algoritmo K-Means varias veces al mismo conjunto de puntos, es posible que no se obtengan siempre los mismos resultados, obteniendo en unas ocasiones la solución óptima global y en otras una solución óptima local. En la figura 3.9 se muestra el resultado de aplicar de nuevo el algoritmo K-Means al mismo conjunto de puntos que en el del ejemplo de la figura 3.8, pero obteniendo esta vez un óptimo local.

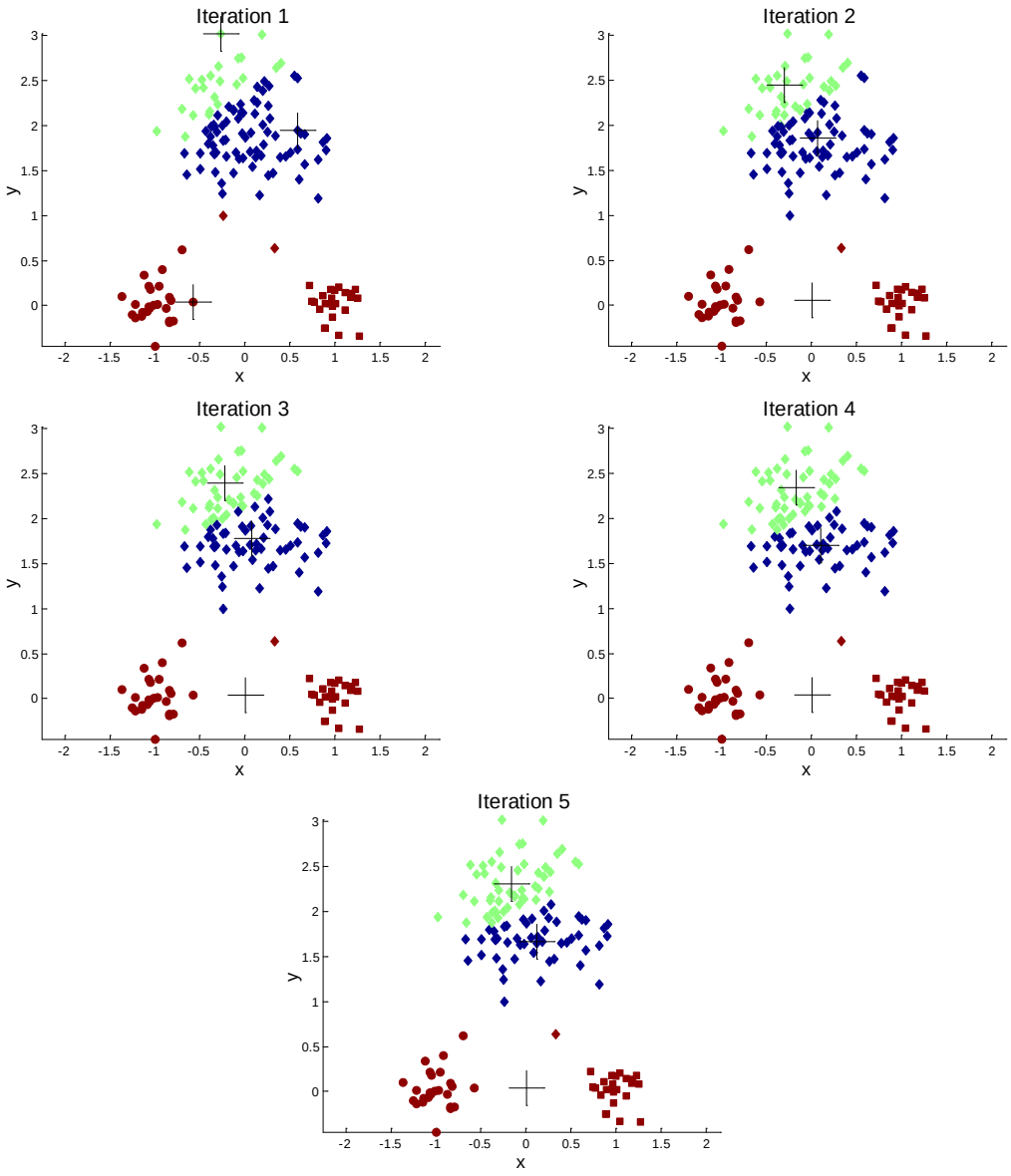


Figura 3.9 Ejemplo de obtención de óptimo local con el algoritmo K-Means.

Para evitar obtener un óptimo local, se repite el algoritmo varias veces y se evalúa en cada una de las repeticiones, la suma total de distancias de los puntos a sus centroides una vez que el algoritmo finaliza. La solución óptima global será aquella en la que dicha suma sea menor. En este proyecto, siempre que se aplica el algoritmo K-Means sobre el conjunto de vectores para obtener un número determinado de grupos, se repite 20 veces.

La distancia de los puntos a los centroides se puede calcular utilizando cualquier métrica, aunque las más comunes son la distancia Manhattan (también conocida como city block) y la distancia euclídea.

La distancia euclídea entre dos puntos sería la longitud de la recta que los une en un espacio euclídeo (es la idea "intuitiva" de distancia). Se trata por tanto de una generalización del teorema de Pitágoras. En general, la distancia euclídea entre los puntos $P = (p_1, p_2, \dots, p_n)$ y $Q = (q_1, q_2, \dots, q_n)$ de un espacio euclídeo n-dimensional se define como:

$$d_2(P,Q) = \left(\sum_{i=1}^{n}(q_i - p_i)^2\right)^{1/2} = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2} \quad (3.1)$$

La distancia euclídea coincide con el operador Norma-2 de un espacio vectorial:

$$d_2(P,Q) = \left\|\overrightarrow{PQ}\right\|_2 \quad (3.2)$$

La distancia city block, también denominada Manhattan o taxicab, es la distancia que un coche recorrería de un punto a otro en una ciudad cuyas calles estuvieran dispuestas en forma de rejilla (como ocurre con la mayoría de las calles de la isla de Manhattan). La distancia city block entre dos puntos $P = (p_1, p_2, \dots, p_n)$ y $Q = (q_1, q_2, \dots, q_n)$ de un espacio euclídeo n-dimensional se define como:

$$d_1(P,Q) = \sum_{i=1}^{n}|q_i - p_i| = |q_1 - p_1| + |q_2 - p_2| + \cdots + |q_n - p_n| \quad (3.3)$$

La distancia city block coincide con el operador Norma-1 de un espacio vectorial:

$$d_1(P,Q) = \left\|\overrightarrow{PQ}\right\|_1 \quad (3.4)$$

En la figura 3.10 se puede apreciar gráficamente la diferencia entre la distancia euclídea y la distancia city block o Manhattan en un espacio de dos dimensiones.

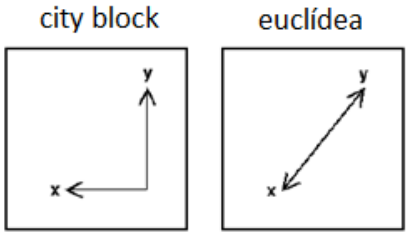


Figura 3.10 Distancias city block y euclídea entre dos puntos.

3.3.4 CÁLCULO DE LAS MATRICES KLT A PARTIR DE LOS GRUPOS OBTENIDOS

Una vez que se ha dividido el conjunto de bloques de la imagen según sus características espaciales, en un número grupos igual al número de KLTs que se desea utilizar, hay que proceder al cálculo de las matrices KLT especializadas para cada uno de los grupos.

Tal y como se explica en el punto 2.4.1.3, el núcleo de transformación de la KLT es dependiente de los datos, es decir, que los propios datos que se quieren transformar son utilizados para calcular la matriz de transformación, lo que provoca que la matriz resulte especializada para las estadísticas de dichos datos. Además, la matriz de la KLT por lo general no es separable, por lo que la KLT se aplica de forma unidimensional a vectores y no a bloques. Por estos motivos, para calcular la matriz de transformación KLT a partir de un grupo de bloques, previamente los bloques del grupo son vectorizados. La forma de vectorizar un bloque de píxeles es concatenando sus filas una detrás de otra obteniendo un vector columna. En la figura 3.11 se muestra un ejemplo para un bloque de 8×8 píxeles.

$$\begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} & p_{15} & p_{16} & p_{17} & p_{18} \\ p_{21} & p_{22} & p_{23} & p_{24} & p_{25} & p_{26} & p_{27} & p_{28} \\ p_{31} & p_{32} & p_{33} & p_{34} & p_{35} & p_{36} & p_{37} & p_{38} \\ p_{41} & p_{42} & p_{43} & p_{44} & p_{45} & p_{46} & p_{47} & p_{48} \\ p_{51} & p_{52} & p_{53} & p_{54} & p_{55} & p_{56} & p_{57} & p_{58} \\ p_{61} & p_{62} & p_{63} & p_{64} & p_{65} & p_{66} & p_{67} & p_{68} \\ p_{71} & p_{72} & p_{73} & p_{74} & p_{75} & p_{76} & p_{77} & p_{78} \\ p_{81} & p_{82} & p_{83} & p_{84} & p_{85} & p_{86} & p_{87} & p_{88} \end{bmatrix} \rightarrow \begin{bmatrix} p_{11} \\ p_{12} \\ \vdots \\ p_{18} \\ p_{21} \\ \vdots \\ p_{28} \\ \vdots \\ \vdots \\ p_{81} \\ \vdots \\ p_{88} \end{bmatrix} \equiv \begin{bmatrix} p_{1} \\ p_{2} \\ \vdots \\ p_{8} \\ p_{9} \\ \vdots \\ p_{16} \\ \vdots \\ \vdots \\ p_{57} \\ \vdots \\ p_{64} \end{bmatrix}$$

8×8 $\qquad\qquad$ 64×1

Figura 3.11 Vectorización de un bloque de píxeles de tamaño 8×8.

Tal y como se explica en el punto 2.4.1.3, las filas de la matriz de transformación de la KLT se corresponden con los autovectores asociados a los autovalores de la matriz de covarianza de los datos a transformar, ordenados de forma descendente según el valor de los autovalores.

La KLT lo que hace es obtener variables incorreladas a partir de variables correladas. En nuestro caso, un bloque de tamaño 8×8 está formado por 64 píxeles y cada píxel lo vamos a asociar a una variable distinta, considerando así que cada bloque

de una imagen está formado por 64 muestras de estas 64 variables (una muestra por cada variable). La matriz de covarianza a partir de la cual se obtendrá la matriz de transformación KLT, es la matriz de covarianza de estas 64 variables. Para calcular dicha matriz lo que hacemos es, una vez vectorizados todos los bloques pertenecientes a un grupo, unirlos formando una matriz de tamaño $64 \times k$ (3.5), siendo k el número de bloques que forman dicho grupo, en la que cada columna se corresponde con un bloque vectorizado del grupo y cada fila contiene las k muestras de cada variable aleatoria que estamos considerando.

$$\begin{bmatrix} p_1^{(1)} & p_1^{(2)} & p_1^{(3)} & \dots & p_1^{(k)} \\ p_2^{(1)} & p_2^{(2)} & p_2^{(3)} & \dots & p_2^{(k)} \\ p_3^{(1)} & p_3^{(2)} & p_3^{(3)} & \dots & p_3^{(k)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ p_{64}^{(1)} & p_{64}^{(2)} & p_{64}^{(3)} & \dots & p_{64}^{(k)} \end{bmatrix} \tag{3.5}$$

La covarianza muestral entre dos variables p_1 y p_2 viene dada por:

$$V_{p_1 p_2} = \frac{1}{k} \sum_{i=1}^{k} (p_1^{(i)} - \overline{p_1})(p_2^{(i)} - \overline{p_2}) \tag{3.6}$$

donde $\overline{p_1}$ y $\overline{p_2}$ denotan respectivamente a la media muestral de las variables p_1 y p_2.

Finalmente, la matriz de covarianza para un grupo de bloques resulta:

$$V = \begin{bmatrix} V_{p_1 p_1} & V_{p_1 p_2} & V_{p_1 p_3} & \dots & V_{p_1 p_{64}} \\ V_{p_2 p_1} & V_{p_2 p_2} & V_{p_2 p_3} & \dots & V_{p_2 p_{64}} \\ V_{p_3 p_1} & V_{p_3 p_2} & V_{p_3 p_3} & \dots & V_{p_3 p_{64}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ V_{p_{64} p_1} & V_{p_{64} p_2} & V_{p_{64} p_3} & \dots & V_{p_{64} p_{64}} \end{bmatrix} \tag{3.7}$$

Por último, la matriz de transformación KLT con la resultarán variables transformadas incorreladas a partir de las variables correladas que estamos considerando, se obtiene mediante la diagonalización de la matriz de covarianza (3.7).

Se dice que una matriz cuadrada V de tamaño $N \times N$ es diagonalizable si existe una matriz invertible P tal que:

$$P^{-1} \cdot V \cdot P = D \tag{3.8}$$

Tal y como se demuestra en [Lay, 1999], para diagonalizar V se han de hallar N autovectores linealmente independientes ($\overrightarrow{x_1}, \overrightarrow{x_2}, \dots, \overrightarrow{x_N}$) y formar con ellos la matriz de paso P usándolos como sus vectores columna; entonces $P^{-1} \cdot V \cdot P$ será una matriz diagonal D con $\lambda_1, \lambda_2, \dots, \lambda_N$ como sus elementos sucesivos en la diagonal, en donde λ_i es el autovalor correspondiente al autovector $\overrightarrow{x_i}$ con $i = 1, 2, \dots, N$.

3.4 COMPARACIÓN ENTRE CONJUNTOS DE TRANSFORMADAS

En este punto se desarrolla el método que se ha usado para obtener una medida que refleje de forma aproximada, el comportamiento que tendría el codificador multitransformada usando un juego de transformadas concreto. La finalidad de esta medida es poder comparar distintos juegos de transformadas y de esta forma poder evaluar el juego de KLTs obtenido tras la Etapa I. Además, gracias a esta medida, se podrán introducir modificaciones, tanto en el proceso de extracción de características como en el de agrupamiento, en pos de obtener un juego de KLTs mejor y se podrán comparar con los resultados de las matrices iniciales.

3.4.1 PARÁMETROS A EVALUAR

Hasta aquí, los principales procesos descritos que intervienen en la ETAPA I de nuestro codificador (extracción de características de los bloques y algoritmo de agrupamiento) están diseñados para obtener unas matrices KLT lo más especializadas posibles a los bloques que van a transformar y de esta forma, conseguir que en la mayoría de los bloques transformados la compactación sea óptima, es decir, que un porcentaje muy alto del total de la energía de un bloque se concentre en un número muy reducido de sus coeficientes transformados, resultando así en una proporción elevada el número de coeficientes energéticamente poco significativos. Luego, parece lógico pensar que para comparar distintas transformadas o conjuntos de transformadas debería bastar con medir el grado de compactación de la energía en los bloques transformados, pero realmente esto no es así ya que el grado de compactación de la energía por sí solo no es un indicador directo de la eficiencia final del codificador, aunque sí que afecta en gran medida.

Como se comenta en el punto 3.3.1, la eficiencia de un codificador de imágenes con pérdidas mediante transformada depende esencialmente, de la compresión resultante en la imagen codificada y de la distorsión presente en la imagen decodificada: a mayor compresión con menor distorsión, la eficiencia del codificador aumentará.

La compresión se consigue gracias al uso conjunto de cuantificación más codificación entrópica: el proceso de cuantificación anula los coeficientes transformados energéticamente poco significativos y reduce el rango de los coeficientes con mayor energía, hecho que aprovecha el codificador entrópico reduciendo el número de bits necesarios para codificar los coeficientes cuantificados. Pero el proceso de cuantificación es un proceso irreversible, ya que elimina parte de la energía de los coeficientes que ya no podrá recuperarse en fases posteriores, lo que provocará que el proceso de cuantificación inversa no pueda obtener los coeficientes transformados originales (sino una aproximación de los mismos) y por lo tanto, aparezca distorsión en la imagen decodificada.

Por su parte, cuanto mayor sea el grado de compactación de la energía, mayor número de coeficientes resultarán con muy poca energía o sin energía, por lo que podrán ser anulados en la cuantificación sin suponer una pérdida de energía significativa, aumentando así el grado de compresión de la imagen codificada y presentando la imagen decodificada un nivel bajo de distorsión.

Pero los procesos de cuantificación y codificación usados en nuestro codificador no son procesos adaptativos ya que actúan igual para todos los bloques, independientemente de la distribución energética que presenten sus coeficientes. Por ello, la tabla de cuantificación que usa el cuantificador se diseña en función de la compactación de la energía que se espera que presenten los bloques transformados y la tabla de códigos de longitud variable que usa el codificador, se obtiene a partir de las estadísticas que presentan los coeficientes cuantificados con dichas tablas, intentando de esta forma maximizar la compresión y minimizar la distorsión en los bloques de forma global. Por lo tanto, aunque dos bloques posean un grado de compactación en su energía similar, si la posición de los coeficientes en donde se concentra la energía no es la misma, el grado de compresión y el nivel de distorsión finales de ambos bloques será distinto. Por este motivo, el grado de compactación de la energía que provoca una transformada sobre un bloque, no es un parámetro suficiente para evaluar la eficiencia del codificador con dicha transformada y por lo tanto, para nuestro propósito de comparar transformadas o conjuntos de transformadas entre sí, será necesario utilizar parámetros que sean indicativos directos de la compresión con la que resultará la imagen codificada y de la distorsión que presentará la imagen decodificada.

A pesar de lo anterior, para evaluar los conjuntos de transformadas, no se han utilizado como parámetros el grado de compresión ni el nivel de distorsión finales y la razón es que para obtener dichos parámetros es necesario aplicar los procesos de cuantificación y codificación finales, lo cual es conveniente evitar ya que al poderse configurar la cuantificación y la codificación mediante la especificación de sus respectivas tablas, el grado de compresión y el nivel de distorsión finales estarían condicionados directamente por las tablas especificadas y realmente sería deseable poder comparar conjuntos de transformadas independientemente de las tablas usadas.

Finalmente, para evaluar los conjuntos de transformadas sin aplicar una cuantificación y una codificación específica, se ha diseñado un método que sustituye dichos procesos, pero que tiene en cuenta sus fundamentos, para de esta forma poder utilizar sus resultados como aproximaciones de los resultados reales. En lo que se basa el método es en intentar aplicar a cada bloque transformado una compresión similar independientemente de la transformada usada, y así considerar directamente el nivel de distorsión que presenta cada bloque decodificado, como un parámetro suficiente para evaluar la transformada aplicada.

El método utilizado directamente sustituye el proceso de cuantificación por un truncado de coeficientes, el cual se ha diseñado siguiendo los fundamentos de las tablas de cuantificación, es decir, conservar la energía de los coeficientes del bloque en los que

se espera que la transformada usada la compacte y anular los coeficientes que se espera que no contengan energía. Además, el número de coeficientes truncados para cualquier transformada va a ser el mismo, lo cual nos permite aproximar que el número de coeficientes nulos totales que poseerá cada bloque tras el truncado va a ser similar, puesto que los coeficientes que conserva el truncado son aquellos en los que típicamente la transformada compacta la energía y por lo tanto no se espera que sean nulos. Por esta razón, y dado que el grado de compresión que el codificador entrópico aplica a un bloque depende en gran medida del número de coeficientes nulos consecutivos que preceden a cada coeficiente no nulo, se ha considerado que el grado de compresión que se le aplica a cada bloque con este truncado es similar, lo cual nos permite utilizar como parámetro suficiente para comparar una transformada con otra o un conjunto de transformadas con otro, el nivel de PSNR resultante en la imagen reconstruida a partir del bloque de coeficientes truncado.

Aclarar que el nivel final de compresión con el que resulta cada bloque después de las fases de Cuantificación y Codificación Entrópica depende directamente de las características del bloque y por ello, asumir que con este truncado se va a aplicar el mismo grado de compresión a todos los bloques no es del todo correcto, pero sí que se puede tomar como una aproximación puesto que los coeficientes que resultan nulos tras el truncado, es muy probable que también resulten nulos usando cuantificación.

A continuación se identificarán los coeficientes dentro de cada transformada (KLT y DCT) en los que se espera que se produzca la compactación de la energía y se fijará el número de coeficientes que no se harán nulos tras el truncado.

3.4.2 Compactación De La Energía Producida Por La KLT

Debido a las características intrínsecas de la KLT, los coeficientes obtenidos tras la transformación siguen un orden decreciente dentro del vector transformado, según la cantidad de información que supone cada uno en la formación de la imagen. Por este motivo, es en los coeficientes situados en las primeras posiciones del vector en los que se compacta la gran mayoría de la energía del bloque.

Para explicar esto último revisaremos la interpretación geométrica que se le puede dar a la transformación lineal, retomando para ello el ejemplo que se utiliza en el apartado 2.1 con la imagen ‘lenna.bmp’, en el que se muestra gráficamente la alta correlación existente entre dos píxeles adyacentes de una imagen. Para ello, en vez de dividir la imagen en bloques de $N \times N$ píxeles, se divide en vectores de tamaño 2×1 píxeles, en los que la primera componente del vector es asociada al valor que toma una variable x y la segunda componente es asociada al valor que toma una variable y, resultando cada vector extraído de la imagen una pareja de valores (x, y). Por último, para examinar gráficamente la correlación existente entre las variables x e y, todos los vectores son considerados como puntos en un espacio de dos dimensiones, y son representados gráficamente en un sistema de coordenadas bidimensional (figura 3.12).

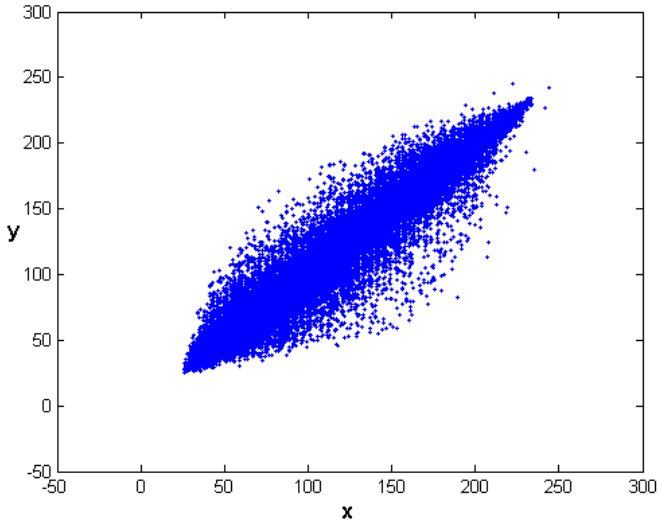


Figura 3.12 Representación en un espacio bidimensional de los puntos de 'lenna.bmp'.

Como se puede observar en la figura 3.12, las variables x e y poseen una alta correlación ya que todos los puntos forman una nube alrededor de la recta $y = x$. Ahora, mediante una transformación lineal, se intentará obtener unas variables transformadas x' e y' cuya correlación sea menor que la de las variables originales.

Como se indica en el punto 2.2.1, aplicar una transformación lineal a un vector es equivalente a aplicar una rotación al punto que simboliza. Para el caso de las variables x e y de nuestro ejemplo, se ha aplicado la transformación lineal definida en (3.9), la cual equivale a aplicar una rotación de 45º en el sentido de las agujas del reloj a los puntos de nuestro ejemplo. Por último, los nuevos puntos obtenidos tras la transformación se han representado en el mismo sistema de coordenadas (figura 3.13).

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} cos\,45° & sin\,45° \\ -sin\,45° & cos\,45° \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{3.9}$$

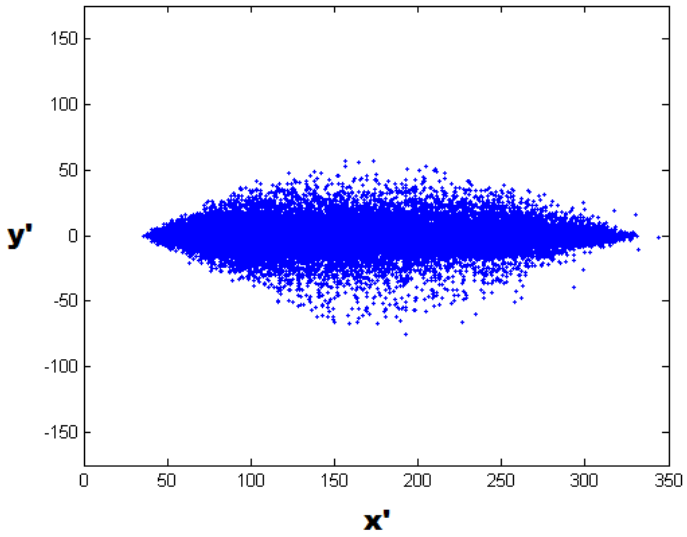


Figura 3.13 Representación en un espacio bidimensional de los puntos transformados de 'lenna.bmp'.

Evaluando la representación de los puntos transformados (figura 3.13) se puede observar que uno de los efectos de la transformación lineal aplicada es que la correlación entre las variables transformadas x' e y' es mucho menor que la correlación existente entre las variables x e y originales.

Otro de los efectos es la modificación en las varianzas de las variables transformadas con respecto a las de las variables originales, ya que para el caso de x e y, sus varianzas son similares, pero para el caso de las variables transformadas, la varianza de x' es mucho mayor que la de y'. Sin embargo, puesto que la matriz de transformación que se ha aplicado es ortonormal, la suma de las varianzas de las variables transformadas es igual a la de las variables sin transformar. Para observar gráficamente la modificación que han sufrido las varianzas, se han representado los histogramas de las variables sin transformar y de las variables transformadas en las figuras 3.14 y 3.15 respectivamente.

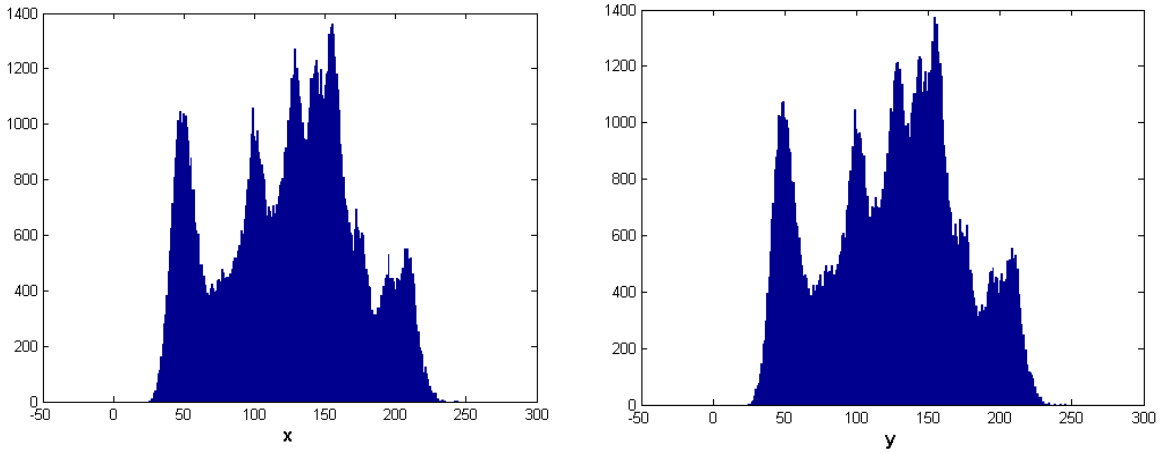


Figura 3.14 Histogramas de las variables sin transformar x e y.

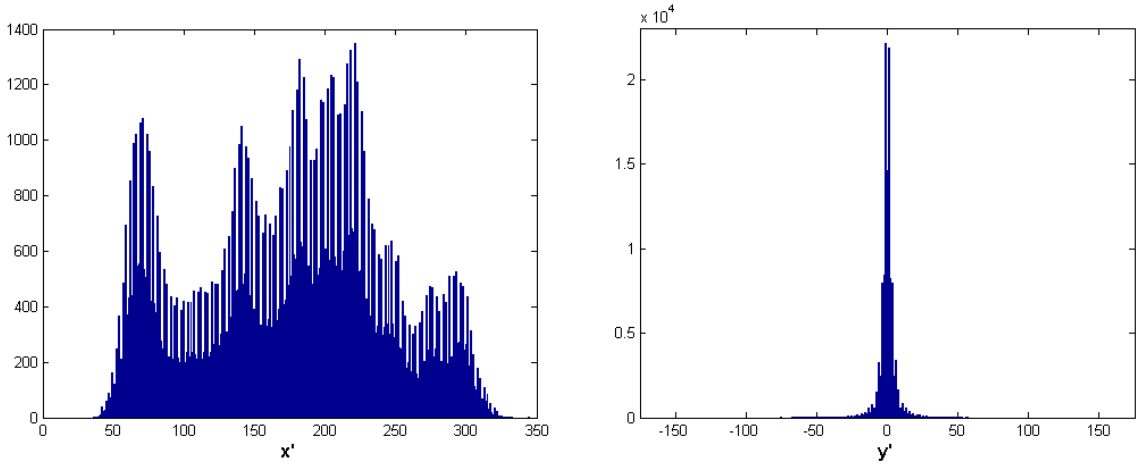


Figura 3.15 Histogramas de las variables transformadas x' e y'.

Inspeccionando la figura 3.14 se puede verificar que los histogramas de las variables antes de la transformación son prácticamente iguales. Además, el rango de valores en los que varía x es muy similar al rango en el que varía y.

Sin embargo, no ocurre lo mismo con los histogramas de las variables transformadas (figura 3.15), puesto que el rango de valores de x' es mucho mayor que el de y', debido a que el rango de la variable x ha sufrido una expansión, mientras que el de la variable y ha sufrido una compresión.

Desde el punto de vista energético, los valores de las muestras de la variable x' son altos, por lo que la variable x' posee gran cantidad de energía, mientras que las muestras de la variable y' contienen mucha menos energía puesto que sus valores han pasado a ser cero o cercanos a cero. En definitiva, lo que ha conseguido la transformación es compactar la mayor parte de la energía en una de las dos variables, otorgándole así mayor importancia a dicha variable puesto que contiene la mayor parte de la información.

Si aplicamos ahora la KLT a los mismos puntos bidimensionales del ejemplo anterior, dichos puntos también sufrirán una rotación, pero dicha rotación será la exacta para eliminar totalmente la correlación existente entre las variables. Consecuentemente, el rango de valores de cada variable también se verá modificado, y con ello las varianzas de las variables transformadas, pero como el núcleo de transformación de la KLT también es ortonormal, la suma total de las varianzas de las variables sin transformar seguirá siendo igual a la de las variables transformadas.

La ventaja de la KLT frente a la transformación del ejemplo es que, además de obtener variables transformadas totalmente decorreladas, dichas variables se encuentran ordenadas de forma decreciente según sus varianzas. Esto es así por las características intrínsecas de la KLT, puesto que las filas de su matriz de transformación se corresponden con los autovectores de la matriz de covarianza de los datos, pero ordenados en orden decreciente según los autovalores de la misma matriz, los cuales a su vez se corresponden con las varianzas de las variables transformadas.

En el caso del proyecto que nos atañe, la forma de eliminar la correlación entre píxeles es similar a la del ejemplo anterior, con la principal diferencia de que en vez de extraer de la imagen bloques de 2×1 píxeles, se extraen bloques de tamaño 8×8 píxeles, con la intención de eliminar la correlación entre un mayor número de píxeles vecinos. Cada bloque es vectorizado según muestra la figura 3.11 y cada una de sus componentes es asociada al valor que toma una variable determinada, por lo que cada vector contendría los respectivos valores de las 64 variables que se estarían considerando. Del mismo modo, cada uno de estos vectores se puede interpretar como un punto en un espacio, pero esta vez de 64 dimensiones. Este espacio es imposible de representar gráficamente, pero las mismas operaciones que se pueden realizar en un espacio de dos dimensiones se pueden extender a espacios de más dimensiones, por lo que aplicar la KLT a estos puntos de 64 dimensiones también equivale a aplicarles una

rotación (aunque en un espacio de 64 dimensiones) y a que los rangos de valores de cada una de las 64 variables también se vean comprimidos o expandidos.

Lo relevante de usar la KLT es que las nuevas 64 variables transformadas se van a encontrar ordenadas de forma decreciente según sus varianzas, es decir, en la primera componente de cada vector transformado se van a encontrar los valores de la variable transformada con mayor varianza, en la segunda componente de cada vector transformado se van a encontrar los valores de la variable transformada con segunda mayor varianza y así sucesivamente. Gracias a este orden, rápidamente se pueden identificar las variables transformadas que aportan más información en la reconstrucción de las variables originales, asociadas a las componentes iniciales de los vectores transformados y las que aportan menos información debido a que poseen menor varianza, asociadas a las componentes finales de los vectores.

Debido a esta propiedad, si se transforma un vector de datos correlados con una KLT especializada para ese tipo de datos, la mayor parte de la energía se concentrará en los primeros coeficientes del vector transformado. Sin embargo, si se utiliza la misma KLT para transformar un vector de datos también correlados, pero con distintas estadísticas que el vector anterior, la concentración de la energía en los primeros coeficientes será mucho menor.

Por todo ello podemos concluir que los coeficientes transformados en los que se espera que una KLT especializada compacte la energía son los coeficientes situados en las primeras posiciones del vector transformado.

3.4.3 Compactación De La Energía Producida Por La DCT

El núcleo de transformación de la DCT, al contrario que el de la KLT, no depende de los datos que se desean transformar sino todo lo contrario, permanece invariable. Esto hace que la distribución energética que presentan los coeficientes de un bloque transformado con la DCT, dependa directamente de las estadísticas que relacionan los píxeles del bloque, por lo que a priori, no resulta tan fácil como en la KLT identificar los coeficientes en los que se espera que se compacte la energía. Sin embargo, sabemos que para bloques de tamaño 8×8 píxeles extraídos de imágenes, la distribución energética resultante al transformarlos con la DCT suele ser parecida. Es esta cualidad de la DCT en imágenes, la que se aprovechará para identificar los coeficientes en los que típicamente la DCT compacta la energía.

Sabemos que la DCT al transformar un bloque lo que hace es descomponerlo en un conjunto discreto de frecuencias espaciales. En la figura 3.4 se representan gráficamente el conjunto de 64 frecuencias espaciales para la DCT bidimensional de tamaño 8×8. Cada una de estas frecuencias espaciales está asociada a un coeficiente transformado en concreto, de forma que se puede obtener la imagen original mediante combinación lineal de dichas frecuencias espaciales, utilizando como peso para cada

frecuencia espacial el valor de su coeficiente asociado. Por este motivo, la energía de los coeficientes transformados dependerá del grado de presencia de cada una de estas frecuencias espaciales en el bloque original, por ello, si se transforma con la DCT bloques en los que predominen las bajas frecuencias, se obtendrán bloques transformados en los que los coeficientes más cercanos al DC concentrarán la mayor parte de la energía (ya que son los asociados a las bajas frecuencias), mientras que al transformar bloques en los que predominen las altas frecuencias espaciales la energía se concentrará en los coeficientes más alejados al DC.

Como ya se ha comentado anteriormente, por lo general en los bloques de una imagen predominan las bajas frecuencias. Esto se debe a que, normalmente, dentro de los bloques de una imagen el valor de los píxeles desde un punto a otro varía de forma lenta. Por este motivo, al transformar un bloque de una imagen mediante la DCT, la energía suele compactarse en los coeficientes más cercanos al DC, mientras que según nos alejamos del DC la energía de los coeficientes disminuye drásticamente, llegando a ser nulos los coeficientes más alejados, que son los asociados a las frecuencias más altas. Retomando la figura 3.5 del punto 3.3.2, en la que se muestra la distribución energética normalizada media de los bloques DCT de todas las imágenes del set utilizado para las pruebas de este proyecto, se puede apreciar que la gran parte de la energía se concentra en el coeficiente de DC y el resto de energía se reparte entre los coeficientes de AC, pero de forma gradual, conteniendo mucha más energía los coeficientes más cercanos al DC y disminuyendo progresivamente la cantidad de energía según nos aproximamos a los coeficientes asociados a las altas frecuencias.

Para observar con más claridad cómo van disminuyendo los pesos asociados a las frecuencias espaciales de la DCT según aumenta la frecuencia, en vez de representar la distribución energética media de todos los bloques transformados pertenecientes a las imágenes usadas en este proyecto, con los mismos bloques transformados, se han calculado los pesos medios asociados a dichas frecuencias espaciales y se han representado en la figura 3.16. Para ello, todos los bloques se han transformado con la DCT bidimensional y puesto que para esta representación únicamente nos interesa el peso de los coeficientes, se les ha aplicado el valor absoluto a todos los coeficientes transformados prescindiendo así del signo. Seguidamente, en cada bloque, los pesos se han normalizado con la suma total de pesos contenidos en dicho bloque. Una vez normalizados los pesos de todos los bloques, se ha hallado el bloque medio y se ha representado gráficamente, eliminando antes el peso asociado a la frecuencia nula (coeficiente DC) ya que son los pesos asociados a las frecuencias no nulas los que nos interesa representar gráficamente.

En la figura 3.16 se puede observar claramente cómo los pesos asociados a las frecuencias bajas (cercanos al coeficiente DC) poseen los valores más altos, mientras que según nos vamos desplazando hacia los pesos asociados a las altas frecuencias, los valores van disminuyendo progresivamente, llegando a ser nulos o prácticamente nulos por debajo de la diagonal formada por los pesos (8,1) y (1,8). Tras esto podemos concluir que los coeficientes en los que se espera que la DCT compacte la energía al

transformar los bloques de una imagen son el coeficiente de DC y los coeficientes más próximos a él.

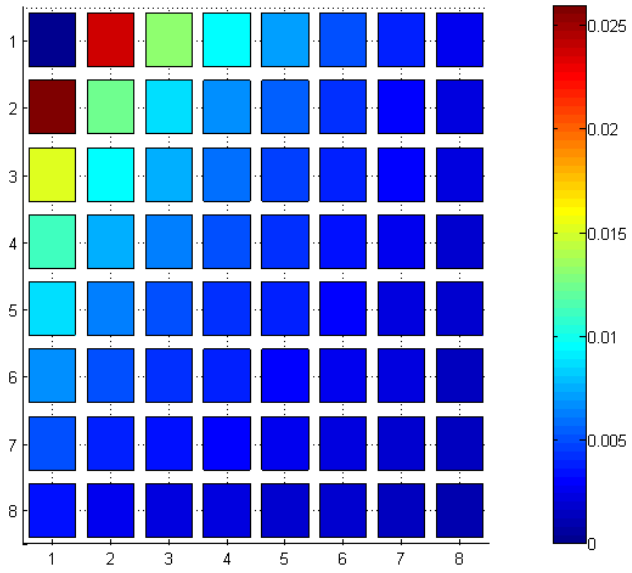


Figura 3.16 Bloque medio de los pesos normalizados DCT.

Una vez identificadas las posiciones de los coeficientes en los que se espera que se compacte la energía tanto por parte de la KLT como por parte de la DCT, quedaría por fijar el número de coeficientes que se conservarán tras aplicar truncado.

3.4.4 TRUNCADO DE COEFICIENTES

Tras identificar las posiciones de los coeficientes en los que se espera que una KLT especializada compacte la energía (posiciones iniciales del vector transformado) y también la posición de los coeficientes en los que lo hará la DCT (coeficientes cercanos al DC), la forma de comparar una transformada con la otra será truncando el mismo número de coeficientes conservando siempre los más significativos y comparando la distorsión que presenten las imágenes reconstruidas.

Pero el truncado de coeficientes puede beneficiar más a la KLT que a la DCT, sobre todo en aquellos bloques que presenten altas frecuencias espaciales, ya que en estos bloques la DCT no compacta la energía cerca del DC sino que aparece repartida en los coeficientes de alta frecuencia, que son los susceptibles de ser truncados. Sin embargo esto no ocurre con la KLT ya que las matrices KLT obtenidas tras el proceso de agrupamiento, en general, tendrán un cierto grado de especialización por lo que la energía en los vectores transformados se compactará en los coeficientes situados en las primeras posiciones, que son los susceptibles de no ser truncados.

Para intentar que el truncado no perjudique a la DCT, se ha de aplicar un truncado tal que la distorsión que presenten las imágenes codificadas con la DCT, sea similar a la que presentarían si se hubieran codificado con JPEG, en el cual se utiliza cuantificación escalar en vez de truncado. Además, este truncado también debe afectar por igual a las mismas frecuencias espaciales verticales que horizontales, y así no beneficiar ni perjudicar a las imágenes en las que predomine uno de estos tipos de frecuencias.

Teniendo en cuenta lo anterior, junto con la distribución de los pesos asociados a las frecuencias espaciales, representada en la figura 3.16, se ha elegido que tras el truncado se conserven los 28 coeficientes representados en la figura 3.17.

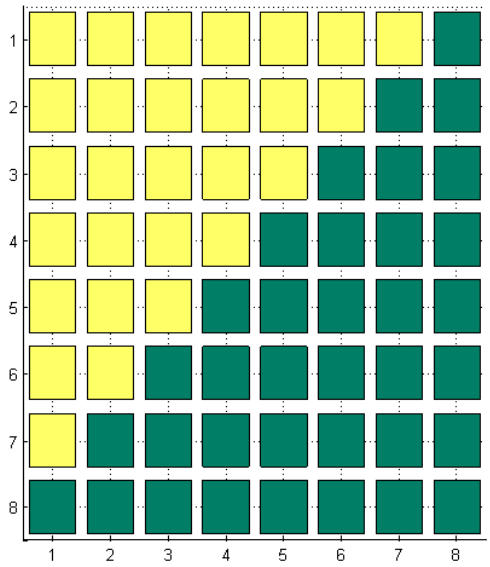


Figura 3.17 Los 28 coeficientes DCT que se conservan tras aplicar truncado.

Al conservar los coeficientes representados en amarillo se están conservando aquellos en los que típicamente la DCT compacta la energía y también se están incluyendo algunos coeficientes de media-alta frecuencia para que aquellas imágenes que contengan bloques compuestos mayoritariamente por altas frecuencias, no resulten con un alto grado de distorsión al aplicar el truncado. Además, este truncado se ha realizado de forma simétrica con respecto a la diagonal formada por el coeficiente de DC y el coeficiente de más alta frecuencia vertical y horizontal (8,8) viéndose de esta forma igualmente afectadas las frecuencias horizontales y las verticales.

En la tabla 3.1 se muestran los respectivos niveles de distorsión que presentan las imágenes obtenidas tras dividir cada imagen original del set de prueba que se utiliza en este proyecto, en bloques de tamaño 8×8 píxeles, transformarlos todos con la DCT bidimensional, hacer nulos los coeficientes según el truncado descrito y aplicarles la DCT inversa bidimensional.

IMAGEN	MSE	PSNR (dB)
lenna.bmp	9,5903	38,3125
baboon.bmp	123,953	27,1982
barbara.bmp	29,2411	33,4709
kodim01.png	50,0602	31,1359
kodim02.png	13,0962	36,9594
kodim03.png	11,1519	37,6573
kodim04.png	12,5972	37,128
kodim05.png	64,538	30,0326
kodim06.png	40,7274	32,0319
kodim07.png	12,0049	37,3372
kodim08.png	76,7571	29,2796
kodim09.png	12,1501	37,285
kodim10.png	15,7615	36,1548
kodim11.png	31,4464	33,1551
kodim12.png	10,9331	37,7434
kodim13.png	130,5531	26,9729
kodim14.png	33,9268	32,8254
kodim15.png	18,4593	35,4686
kodim16.png	17,8092	35,6243
kodim17.png	19,328	35,2689
kodim18.png	48,9941	31,2294
kodim19.png	26,8746	33,8374
kodim20.png	19,4638	35,2385
kodim21.png	36,4192	32,5175
kodim22.png	22,6036	34,589
kodim23.png	7,9042	39,1522
kodim24.png	62,8695	30,1464

Tabla 3.1 Distorsión en las imágenes con la DCT aplicando truncado de coeficientes diseñado.

Tal y como muestra la tabla 3.1, la mayoría de las imágenes recuperadas sólo con 28 coeficientes DCT poseen un nivel de PSNR igual o superior a 30 dB. Esto es un buen indicador de que el truncado elegido es válido para nuestro objetivo ya que, para la mayoría de las imágenes, la distorsión resultante es la típica que presentan las imágenes codificadas usando compresión con pérdidas. Para comprobarlo, en las figuras 3.18 y 3.19 se muestran respectivamente la imagen original 'kodim05.png' y su reconstrucción con sólo 28 coeficientes DCT y de igual forma en las figuras 3.20 y 3.21 se muestran la imagen original 'kodim24.png' y su reconstrucción respectivamente. En ambas reconstrucciones el nivel de PSNR es de 30 dB, y se puede comprobar comparándolas con sus originales que la distorsión apenas es perceptible, siendo muy similar a la que se hubiera obtenido si se hubieran codificado con JPEG. Sólo las imágenes 'baboon.bmp', kodim08.png', y 'kodim13.png' están por debajo de 30 dB, lo cual es lógico ya que las tres poseen zonas en las que claramente predominan las altas frecuencias espaciales, pero puesto que los niveles de PSNR de estas imágenes recuperadas no son muy bajos y dado que las tablas de cuantificación recomendadas en JPEG también penalizan más a

las altas frecuencias, se ha considerado que dichos niveles también se aproximan a los que se obtendrían si las imágenes se hubieran codificado con JPEG.

Figura 3.18 'kodim05.png'

Figura 3.19 'kodim05.png' recuperada con 28 coeficientes DCT tras aplicar truncado

Figura 3.20 'kodim24.png'

Figura 3.21 'kodim24.png' recuperada con 28 coeficientes DCT tras aplicar truncado

Por lo tanto, tras todo lo anterior, se asume que la elección de este truncado en concreto es válida para nuestro propósito de comparar la DCT y la KLT.

3.4.5 Evaluación De Las Matrices KLT Obtenidas Tras La Etapa I

Como se ha descrito en los puntos anteriores, para comparar el funcionamiento de un conjunto de transformadas frente a otro en una imagen, lo que se hace es comparar los niveles de PSNR que presentan las imágenes reconstruidas, tras haber transformado los bloques de la imagen original con ambos conjuntos de transformadas y habiendo aplicado truncado de coeficientes.

En concreto, cada imagen reconstruida se obtiene tras haber seguido tres pasos: el primero es la transformación de cada bloque de la imagen original con su respectiva transformada asignada del conjunto disponible; el segundo es el truncado de los coeficientes del vector o del bloque transformado obtenido (vector para el caso de la KLT o bloque para el de la DCT) y el tercero es la aplicación de la transformada inversa correspondiente a cada bloque/vector truncado.

A la hora de aplicar el truncado de coeficientes, si se está realizando sobre un bloque transformado con la DCT, el truncado que se aplica es el descrito en el apartado anterior. Sin embargo, si el bloque se ha transformado con una KLT, en vez de un bloque transformado se habrá obtenido un vector de coeficientes, por lo que el truncado sólo conservará los primeros 28 coeficientes (el mismo número que en la DCT).

Este mismo método que se ha definido para comparar el funcionamiento de dos conjuntos de transformadas sobre una imagen, es el que se ha usado para verificar que las técnicas utilizadas en la Etapa I de nuestro codificador (extracción de características y agrupamiento) consiguen agrupar los bloques de cada imagen según sus características espaciales y obtener a partir de los grupos, matrices KLT especializadas. Para ello, se ha realizado un experimento en el que, para cada imagen del conjunto utilizado, se compara el funcionamiento de las KLTs obtenidas a partir de la Etapa I, frente al funcionamiento de KLTs obtenidas mediante agrupamiento aleatorio de los bloques; esta comparación se ha hecho con distintas cantidades de KLTs, comenzando con 1 y acabando con 7. En las tablas A.1 y A.2 del anexo A, se recogen los resultados de este experimento, en el que para cada imagen y para cada cantidad de KLTs, se muestran los resultados obtenidos con los dos tipos de KLTs (las obtenidas mediante el algoritmo de agrupamiento de la Etapa I y las obtenidas mediantes agrupamiento aleatorio); en la tabla A.1 se recogen los resultados de MSE y en la tabla A.2 se recogen los resultados de PSNR. Para mejorar la visualización de los resultados, respectivamente en la tabla A.1 y en la tabla A.2, se han resaltado en color rojo los valores mínimos de MSE y los valores máximos de PSNR.

Analizando los resultados individualmente de cada imagen podemos verificar que, para la gran mayoría de las imágenes, el nivel de PSNR es superior cuando se utilizan las matrices KLT obtenidas tras la Etapa I del codificador, que cuando se utilizan las KLTs resultantes de agrupar los bloques aleatoriamente. La mejora que suponen las matrices KLT de la Etapa I frente a las matrices KLT del agrupamiento aleatorio, se hace más notable según aumenta el número de grupos, sobre todo cuando

se utilizan 3 o más. Para el caso de 2 KLTs y en menor medida para el de 3, en algunas imágenes, los dos niveles de PSNR están muy próximos entre sí, incluso en las imágenes kodim12.png, kodim18.png y kodim23.png, el nivel de PSNR obtenido con las KLTs de la Etapa I es ligeramente inferior al obtenido con las KLTs del agrupamiento aleatorio. Esto es así porque cuando se divide el conjunto de bloques en un número pequeño de conjuntos, aunque se agrupen según sus características espaciales, los conjuntos de bloques aún resultarán bastante heterogéneos y dependiendo del número de características espaciales predominantes en la imagen, los grupos obtenidos pueden resultar bastante similares a los del agrupamiento aleatorio.

Para obtener un análisis global, se ha calculado de forma conjunta el MSE de todas las imágenes para los diferentes cantidades de KLTs desde 1 hasta 7, calculando a partir de ellos los niveles de PSNR globales para el conjunto de imágenes usado. La tabla 3.2 recoge los valores de MSE y de PSNR globales obtenidos con las KLTs resultantes del agrupamiento aleatorio y la tabla 3.3 recoge los correspondientes valores obtenidos con las KLTs de la Etapa I del codificador.

	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
MSE_{ALE}	29,0891	28,3203	27,6292	26,9916	26,3792	25,7810	25,1996
$PSNR_{ALE}$ [dB]	33,4935	33,6098	33,7171	33,8185	33,9182	34,0178	34,1169

Tabla 3.2 MSE y PSNR globales con KLTs a partir del agrupamiento aleatorio.

	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
MSE_{ENER}	29,0891	26,9005	25,3445	24,3722	23,6453	23,0481	22,4463
$PSNR_{ENER}$ [dB]	33,4935	33,8332	34,0920	34,2619	34,3934	34,5045	34,6194

Tabla 3.3 MSE y PSNR globales con KLTs de la Etapa I (energía normalizada de las bandas).

Al analizar los niveles de PSNR globales recogidos en las tablas 3.2 y 3.3 se puede verificar que para cualquier número de KLTs, los niveles de PSNR globales que se obtienen tras utilizar las KLTs de la Etapa I del codificador son considerablemente superiores a los que se obtienen tras usar las KLTs resultantes del agrupamiento aleatorio, por lo que podemos concluir que el diseño de los procesos utilizados en la Etapa I del codificador (extracción de características y algoritmo de agrupamiento) son válidos para obtener matrices de transformación KLT especializadas a las características espaciales predominantes en una imagen.

Por último, para visualizar de forma gráfica la mejora en el nivel de especialización que experimentan las matrices KLT al utilizar los procesos de la Etapa I frente al agrupamiento aleatorio, se ha tomado como referencia el nivel de PSNR global

que se obtiene al utilizar una única KLT (ya que es el mismo para ambos casos) y se han calculado los aumentos en el nivel de PSNR global para los distintos casos de número de KLTs, recogiéndose en la tabla 3.4 y representándose gráficamente en la figura 3.22.

	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
$PSNR_{ALE} - PSNR_{1KLT}$ [dB]	0,0000	0,1163	0,2236	0,3250	0,4247	0,5243	0,6234
$PSNR_{ENER} - PSNR_{1KLT}$ [dB]	0,0000	0,3397	0,5985	0,7683	0,8998	1,0109	1,1258
$PSNR_{ENER} - PSNR_{ALE}$ [dB]	0,0000	0,2234	0,3748	0,4433	0,4752	0,4866	0,5025

Tabla 3.4 Niveles globales de PSNR referenciados al nivel obtenido con 1 KLT.

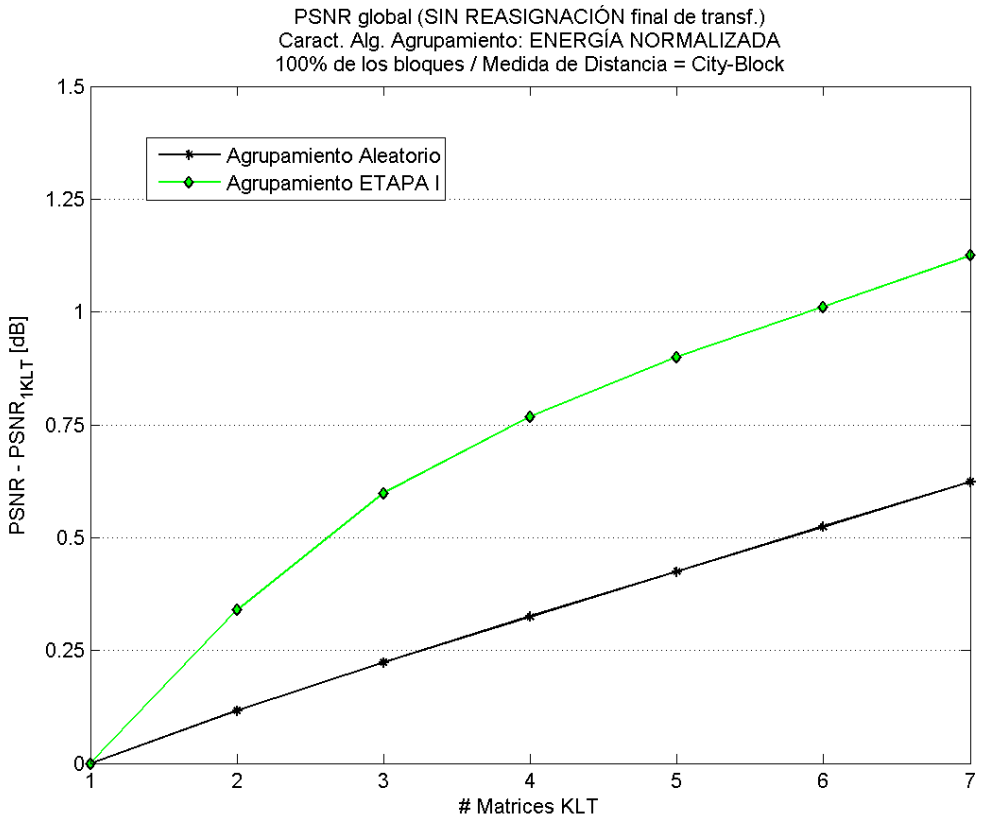


Figura 3.22 Evolución de los niveles globales de PSNR con el aumento del número de KLTs.

Como se puede observar en la figura 3.22, la especialización en las matrices KLT formadas a partir del agrupamiento aleatorio, aumenta de forma prácticamente lineal con el aumento del número de KLTs, esto es así ya que cuanto mayor es el tamaño de un grupo heterogéneo de bloques, menor es la especialización que posee la KLT resultante de dicho grupo. Por este motivo, al aumentar el número de KLTs y reducir así el tamaño de los grupos, la especialización de las KLTs resultantes de dichos

grupos aumenta y se obtienen niveles de PSNR mayores. Por otro lado, la especialización en las KLTs de la Etapa I también aumenta con el número de KLTs, pero sin embargo no lo hace linealmente desde el principio, sino que hasta 4 KLTs aumenta más rápidamente y a partir de 4 aumenta prácticamente de forma lineal hasta 7. Esto es debido a que al utilizar un número muy reducido de KLTs, aunque los bloques se agrupen según sus características espaciales, inevitablemente los grupos resultarán más heterogéneos que con un número de KLTs superior, por ello, en cuanto se aumenta el número de grupos crece rápidamente la especialización de las matrices KLTs.

Reasignación final de transformada

Los resultados de MSE y PSNR del experimento anterior demuestran que los procesos de la Etapa I son capaces de agrupar bloques con características espaciales similares y obtener a partir de cada uno de ellos una matriz KLT especializada a las características particulares que predominan en dicho grupo. Pero estos resultados no reflejan la reasignación final de transformada que sufrirán los bloques, ya que dichos resultados se han obtenido transformando cada bloque con la KLT obtenida a partir del grupo al cual pertenece dicho bloque y no con la transformada que finalmente se le aplicará, que puede ser la misma o no. Esta reasignación se efectúa en la fase de Selección de la Etapa II del codificador y su objetivo es asegurar que a cada bloque se le aplicará la transformada óptima del conjunto de transformadas candidatas.

A priori, sería lógico pensar que la KLT calculada a partir de un grupo de bloques, debería compactar mejor la energía en los bloques que pertenecen a ese grupo que la DCT y, sobre todo, que el resto de KLTs obtenidas a partir de los otros grupos de bloques, pero realmente no podemos tener la certeza de que ocurra así. El motivo es que el algoritmo de agrupamiento utilizado no puede asegurar que cada grupo obtenido posea un alto grado de homogeneidad, ya que cada bloque que participa en el algoritmo es asignado al grupo cuyo centroide se encuentra a menor distancia, lo cual no tiene por qué significar que dicho bloque se encuentre muy cercano al centroide. Por este motivo y dependiendo de cada imagen y del número de grupos que se deseen obtener, podrían resultar por ejemplo grupos de bloques muy compactos en el que la mayoría de sus bloques se encuentren muy próximos al centroide del grupo, lo que significaría que la mayoría de los bloques de ese grupo poseen las mismas características espaciales; o también podrían resultar grupos de bloques más dispersos y alejados del centroide del grupo, lo que significaría que los bloques cercanos al centroide poseerían características espaciales similares, pero los más alejados únicamente compartirían alguna característica espacial con ellos.

Normalmente, en una imagen suele haber zonas amplias que contienen un alto número de bloques con características espaciales similares, por lo que tras aplicar nuestro algoritmo de agrupamiento, se suelen obtener grupos en los que la mayoría de los bloques se encontrarán más o menos próximos al centroide y una minoría estará más alejada. En este caso, la matriz KLT obtenida a partir de uno de estos grupos, estará

mucho más especializada para los bloques próximos al centroide que para los más dispersos, lo que se traduce en que la compactación de la energía para los bloques más próximos al centroide será óptima, pero no para los más alejados, impidiéndonos asegurar que dicha transformada sea la mejor opción para estos bloques. Para los bloques periféricos, se podría dar el caso de que la DCT, o incluso una KLT obtenida a partir de alguno de los grupos vecinos, compactaran mejor su energía que la KLT del grupo al que pertenece. Es por ello que en la fase de Selección de la Etapa II, a cada bloque se le asignará aquella transformada del conjunto de candidatas (DCT y KLTs de la Etapa I) con la que se obtenga el bloque reconstruido con menor nivel de distorsión, después de haber aplicado la cuantificación de los coeficientes.

Por lo tanto, para reflejar en los resultados de distorsión obtenidos con nuestro método dicha reasignación, se ha repetido el último experimento con todas las imágenes, pero añadiendo una reasignación final de transformada, en función de la distorsión que presentan los bloques reconstruidos tras transformar cada bloque con todas las transformadas candidatas y aplicar el truncado de coeficientes definido en nuestro método. Los resultados para cada imagen de este nuevo experimento, se muestran en las tablas B.1 y B.2 del anexo B; en la tabla B.1 se recogen todos los valores de MSE y en la tabla B.2 se recogen los niveles de PSNR asociados a dichos valores de MSE. De igual forma que en el experimento sin reasignación de transformada, para cada imagen se presentan dos filas de resultados: en la fila superior se recogen los resultados obtenidos al usar las KLTs formadas a partir del agrupamiento aleatorio de bloques y en la fila inferior se recogen los resultados al utilizar las KLTs obtenidas de la Etapa I del codificador.

Analizando de forma individual los resultados de MSE recogidos en la tabla B.1 podemos comprobar que, al igual que ocurría con los resultados sin realizar reasignación final de transformada (tabla A.1), en la gran mayoría de los casos el MSE en las imágenes reconstruidas tras haber realizado truncado de coeficientes, es considerablemente menor al utilizar las KLTs de la Etapa I junto con la DCT, que al utilizar las KLTs del agrupamiento aleatorio junto con la DCT, lo cual vuelve a corroborar que los procesos diseñados en la Etapa I del codificador son válidos para obtener matrices KLT especializadas a las características espaciales presentes en cada imagen.

De forma global, se han calculado los valores de MSE del conjunto de todas las imágenes y sus correspondientes valores de PSNR, para los distintos casos de número de grupos utilizados y para ambos juegos de transformadas. En la tabla 3.5 se muestran los valores de MSE y PSNR resultantes de utilizar el juego de transformadas compuesto por la DCT más las KLTs de la Etapa I y en la tabla 3.6 se muestran los valores resultantes de utilizar la DCT más las KLTs formadas a partir de agrupamiento aleatorio.

	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
MSE_{ALE_R}	26,9273	25,4466	24,3282	23,4662	22,7233	22,0582	21,4475
$PSNR_{ALE\_R}$ [dB]	33,8289	34,0745	34,2697	34,4264	34,5661	34,6951	34,8170

Tabla 3.5 MSE y PSNR globales con KLTs a partir del agrupamiento aleatorio (con reasignación de transf.).

	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
MSE_{ENER_R}	26,9273	24,1793	22,7006	21,6844	20,8810	20,2835	19,7207
$PSNR_{ENER\_R}$ [dB]	33,8289	34,2964	34,5704	34,7693	34,9333	35,0594	35,1816

Tabla 3.6 MSE y PSNR globales con KLTs de la Etapa I obtenidas con la energía normalizada (con reasignación de transf.).

Comparando ahora los niveles de PSNR de las tablas 3.5 y 3.6, se verifica que para todos los casos de número de grupos desde 1 hasta 7, el nivel de PSNR cuando se utilizan las KLTs de la Etapa I es superior que cuando se utilizan las KLTs obtenidas a partir de agrupamiento aleatorio.

Por último, al igual que para el experimento sin reasignación de transformada, se ha tomado como referencia el nivel de PSNR global obtenido con una única KLT (utilizando reasignación junto con la DCT) y con los niveles globales de PSNR obtenidos con reasignación de transformada, se han calculado los incrementos para los distintos casos de número de KLTs desde 1 hasta 7, recogiéndose dichos resultados en la tabla 3.7 y representándose gráficamente en la figura 3.23.

	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
$PSNR_{ALE\_R} - PSNR_{1KLT\_R}$ [dB]	0,0000	0,2456	0,4408	0,5975	0,7372	0,8662	0,9882
$PSNR_{ENER\_R} - PSNR_{1KLT\_R}$ [dB]	0,0000	0,4675	0,7416	0,9405	1,1044	1,2305	1,3527
$PSNR_{ENER\_R} - PSNR_{ALE\_R}$ [dB]	0,0000	0,2219	0,3007	0,3430	0,3672	0,3643	0,3645

Tabla 3.7 Niveles globales de PSNR referenciados al nivel obtenido con 1 KLT (con reasignación de transf.).

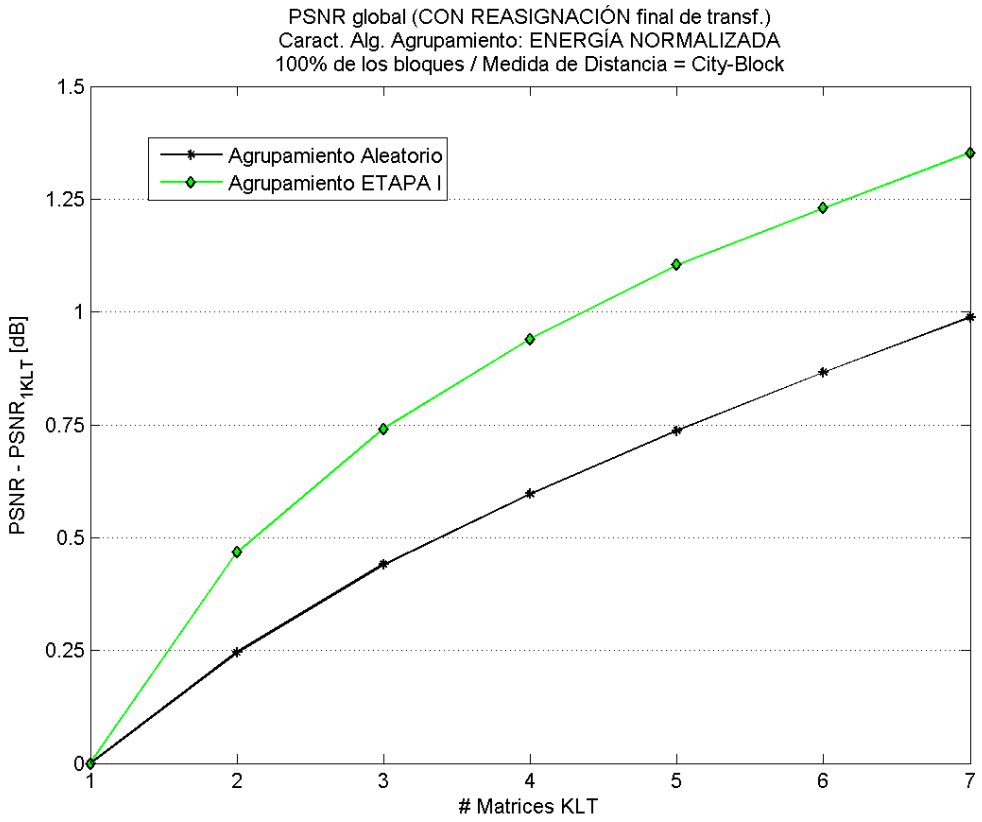


Figura 3.23 Evolución de los niveles globales de PSNR con el aumento del número de KLTs utilizando reasignación de transformada.

3.5 MAYOR ESPECIALIZACIÓN DE LAS MATRICES KLT Y OPTIMIZACIÓN DEL ALGORITMO DE AGRUPAMIENTO

En este apartado se desarrollan las diferentes modificaciones realizadas en la Etapa I del codificador con los objetivos de obtener unas matrices KLT con una mayor especialización que las obtenidas hasta ahora y una reducción de la carga computacional del algoritmo de agrupamiento.

Las modificaciones introducidas han sido tres: la primera ha sido extraer de las bandas de frecuencia de los bloques otra característica distinta a la energía normalizada para utilizar en el algoritmo de agrupamiento, la segunda ha sido reducir el conjunto de bloques que participan en el agrupamiento excluyendo previamente aquellos que previsiblemente se transformarán con la DCT además de los outliers y la tercera ha sido modificar el tipo de medida de distancia usada en el algoritmo de agrupamiento.

3.5.1 AGRUPAMIENTO CON OTRA CARACTERÍSTICA: SUMA NORMALIZADA DE PESOS

Hasta aquí, las características que se han extraído de cada bloque para utilizar en el algoritmo de agrupamiento, han sido los valores de energía (normalizados) contenidos en las nueve bandas de frecuencia definidas anteriormente. La energía en

una banda se ha obtenido calculando la suma de los pesos elevados al cuadrado de los coeficientes DCT contenidos en dicha banda y el valor obtenido se ha normalizado por la energía total contenida en todas las bandas del bloque y excluyendo al DC.

Los resultados de PSNR globales obtenidos tras aplicar truncado de coeficientes, han demostrado que utilizar el valor normalizado de la energía contenida en cada banda de frecuencia, permite agrupar los bloques según las características espaciales presentes en ellos y obtener así a partir de dichos grupos, matrices KLT especializadas para las características espaciales predominantes en cada grupo. Pero, aunque los resultados han sido satisfactorios, se ha probado a extraer otra característica de los bloques para utilizar en el algoritmo de agrupamiento, con la intención de verificar si esta nueva característica permite agrupar con mayor precisión los bloques que comparten propiedades espaciales y así, obtener a partir de los nuevos grupos matrices KLT más especializadas que compacten mejor la energía en los primeros coeficientes y se obtengan así mejores resultados de PSNR tras aplicar truncado. La nueva característica que se ha extraído de cada banda es la suma de los coeficientes DCT en valor absoluto, o pesos, contenidos en cada una, normalizada por la suma total de los coeficientes en valor absoluto de todas las bandas (excluyendo al DC).

Una vez modificada la Etapa I del codificador para que el algoritmo de agrupamiento utilice esta nueva característica, para comparar los nuevos juegos de transformadas obtenidos, se ha usado el mismo método que venimos utilizando consistente en aplicar truncado de coeficientes a los bloques transformados y evaluar los niveles de PSNR que presentan las imágenes reconstruidas. En este caso, directamente se ha utilizado reasignación de transformada, es decir, del conjunto de transformadas candidatas formado por las KLTs obtenidas de la Etapa I más la DCT, cada bloque se ha transformado finalmente con aquella que provoca un mayor nivel de PSNR en el bloque reconstruido. Al igual que en los casos anteriores, para cada imagen se ha repetido el proceso variando el número de KLTs utilizado desde 1 hasta 7.

Los resultados de MSE y PSNR individuales de cada imagen se recogen respectivamente en la tabla C.1 y en la tabla C.2 mostradas en el anexo C. En estas tablas podemos verificar a simple vista que, al igual que pasaba cuando se utilizaba como característica la energía normalizada de las bandas, para la gran mayoría de casos el nivel de PSNR obtenido es superior cuando se utilizan las KLTs resultantes de la Etapa I que cuando se utilizan las KLTs formadas a partir de agrupamiento aleatorio, por lo que a priori podríamos confirmar que esta característica también es válida para nuestro objetivo de agrupar bloques con características espaciales similares. Pero para verificar si es más beneficioso utilizar como característica de cada banda la suma normalizada de pesos que la energía normalizada, analizaremos los resultados de forma global, comparando los resultados obtenidos con los resultados globales del agrupamiento aleatorio y con los resultados globales del agrupamiento utilizando la energía normalizada. Los resultados de MSE y PSNR globales que se han obtenido utilizando como característica para el agrupamiento la suma normalizada de pesos se recogen en la tabla 3.8.

	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
MSE_{PESOS_R}	26,9273	23,5072	22,1110	21,2388	20,5207	19,9443	19,4061
$PSNR_{PESOS\_R}$ [dB]	33,8289	34,4188	34,6847	34,8595	35,0089	35,1326	35,2514

Tabla 3.8 MSE y PSNR globales con KLTs de la Etapa I obtenidas con la suma normalizada de pesos (con reasignación de transf.).

Comparando los resultados de la tabla 3.8 (suma normalizada de pesos) con los de la tabla 3.5 (energía normalizada) se puede verificar que independientemente del número de KLTs que se esté utilizando (entre 2 y 7) siempre se obtiene un mayor nivel de PSNR cuando se utiliza como característica para el agrupamiento la suma normalizada de pesos en lugar de la energía normalizada. La tabla 3.9 recoge los incrementos producidos en los niveles de PSNR y se puede comprobar que el aumento es mayor cuando se utilizan pocas KLTs (2-4) que cuando se utilizan más.

	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
$PSNR_{PESOS\_R} - PSNR_{ENER\_R}$ [dB]	0,0000	0,1224	0,1143	0,0902	0,0756	0,0732	0,0699

Tabla 3.9 Incremento en el nivel de PSNR al usar la suma normalizada de pesos como característica en lugar de la energía normalizada (con reasignación de transf.).

Para representar gráficamente el incremento en el nivel de PSNR que se produce al aumentar el número de KLTs cuando se utiliza suma de pesos normalizada y compararlo con el que se produce al utilizar la energía normalizada y también con el del agrupamiento aleatorio, en la tabla 3.10 se recogen, por un lado, los niveles de PSNR obtenidos con la suma normalizada de pesos a los que se les ha restado el nivel de PSNR que se obtiene con una única KLT (utilizando reasignación de transformada) y por otro lado, el incremento de PSNR con respecto al agrupamiento aleatorio que supone usar las KLTs obtenidas utilizando para el agrupamiento la suma normalizada de pesos. En la figura 3.24 se representa el incremento con respecto a una KLT para los casos del agrupamiento aleatorio, agrupamiento con la energía normalizada y agrupamiento con la suma normalizada de pesos.

	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
$PSNR_{PESOS\_R} - PSNR_{1KLT\_R}$ [dB]	0,0000	0,5899	0,8559	1,0306	1,1800	1,3037	1,4226
$PSNR_{PESOS\_R} - PSNR_{ALE\_R}$ [dB]	0,0000	0,3443	0,4150	0,4331	0,4428	0,4375	0,4344

Tabla 3.10 Niveles globales de PSNR obtenidos con la suma normalizada de pesos, referenciados al nivel obtenido con 1 KLT (con reasignación de transf.).

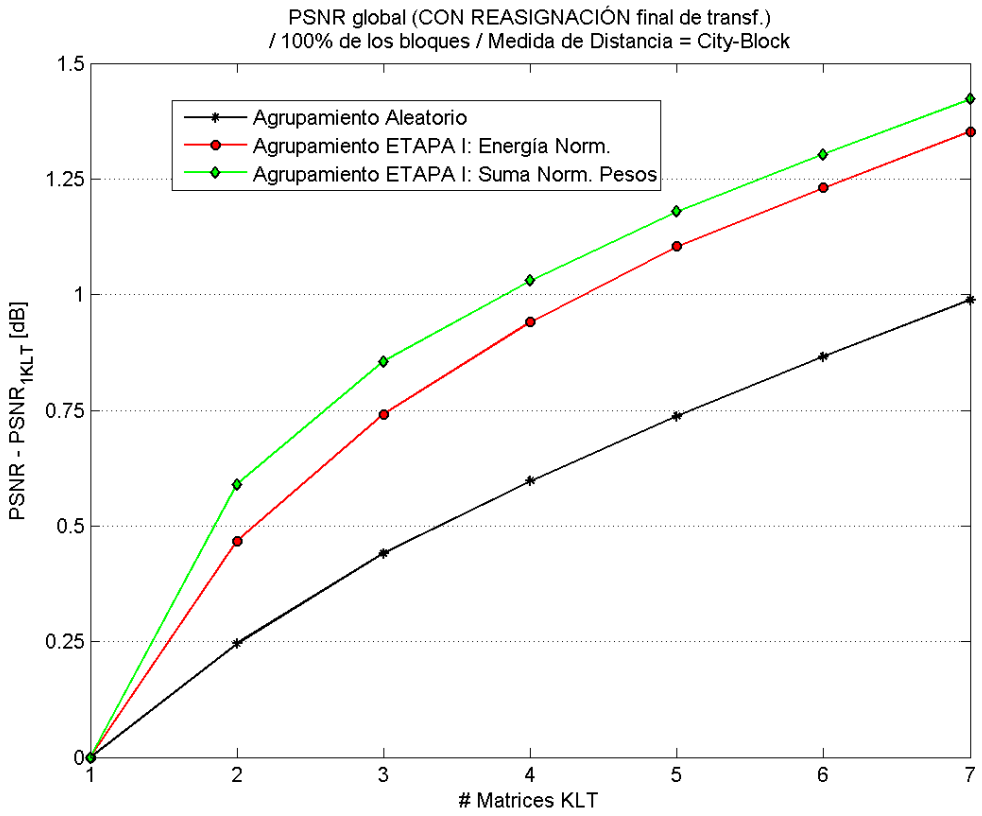


Figura 3.24 Evolución de los niveles globales de PSNR con el aumento del número de KLTs utilizando reasignación de transformada y como característica la suma normalizada de los pesos de las bandas.

Observando la figura 3.24 se puede verificar gráficamente el incremento en el nivel de PSNR que se produce al utilizar para el agrupamiento la suma normalizada de pesos. Dado que se obtiene una clara mejora en los niveles de PSNR, para la Etapa I del codificador se decide utilizar como característica de agrupamiento la suma normalizada de pesos en detrimento de la energía normalizada.

3.5.2 Reducción Del Conjunto de Bloques Que Participan En El Algoritmo De Agrupamiento

Los procesos utilizados en la Etapa I del codificador para la creación de las matrices KLT requieren de una carga computacional muy alta, especialmente el algoritmo de agrupamiento, lo que puede limitar el uso de este codificador en determinados escenarios, como por ejemplo en aplicaciones de tiempo real o en dispositivos móviles con capacidad computacional limitada. Por este motivo sería conveniente optimizar de algún modo los procesos que intervienen en dicha Etapa para reducir la carga computacional necesaria para obtener las matrices KLT.

La solución adoptada ha sido reducir el porcentaje de bloques que participan en la formación de las matrices KLT. De esta forma, la fase de Extracción de Características reducirá el tiempo de cómputo total ya que esta fase se aplica

individualmente a cada uno de los bloques que participan en el algoritmo de agrupamiento; por su lado, la fase de Agrupamiento reducirá la carga computacional y el número de iteraciones necesarias para que el algoritmo converja; por último, la fase de Cálculo de KLTs requerirá menor capacidad de cómputo para obtener las matrices KLT debido a que los grupos de bloques resultantes serán de menor tamaño.

La reducción del conjunto de bloques que participan en la formación de las matrices KLT, se ha hecho reduciendo lo máximo posible el número de bloques que participan en el algoritmo de agrupamiento, pero evitando en todo momento que dicha reducción repercuta negativamente en los niveles PSNR que se obtienen cuando se utiliza el 100 % de los bloques. Para ello, en cada imagen, se han identificado los bloques con mayor probabilidad de acabar transformándose con la DCT y no con alguna de las KLTs resultantes de la Etapa I y se han excluido del proceso de formación de las matrices KLT ya que, incluso habiendo participado en la formación de las KLTs, finalmente en la reasignación de transformada que se lleva a cabo en la Etapa II del codificador se acabarán transformando con la DCT. Además, intentando mejorar los niveles de PSNR, se han identificado los outliers de cada grupo resultante del algoritmo de agrupamiento y se ha probado a excluirlos también del proceso de formación de las matrices KLT. Ambas técnicas se explican con detalle en los puntos 3.5.2.1 y 3.5.2.2.

3.5.2.1 Preselección de los bloques candidatos a transformarse con la DCT

Hasta aquí, en la formación de las matrices KLT participan todos los bloques de la imagen, pero dado que la DCT está especializada de forma nativa para los bloques en los que predominan las bajas frecuencias, evitar que dichos bloques participen en la formación de las KLTs podría hacer que la especialización de las mismas aumentara y al mismo tiempo se redujera el cómputo necesario para obtenerlas.

Para identificar los bloques en los que predominan las bajas frecuencias, se ha calculado el MSE que presentan las reconstrucciones de los bloques, después de haberlos transformado con la DCT y haberles aplicado el truncado de coeficientes diseñado (figura 3.17). Aquellos bloques formados mayoritariamente por frecuencias bajas obtendrán un MSE muy bajo, mientras que los bloques formados mayoritariamente por altas frecuencias obtendrán un MSE muy alto. A partir de los MSE de los bloques, se ha hallado un MSE_{umbral} de forma que, si el MSE de un bloque iguala o supera dicho umbral, el bloque participará en la creación de las KLTs.

$$MSE_{bloques\ KLT} \geq MSE_{umbral} \tag{3.10}$$

Para calcular el MSE_{umbral}, además de los bloques formados mayoritariamente por bajas frecuencias, también se ha tenido en cuenta que habrá bloques que aun participando en la creación de las KLTs, se acaben transformando finalmente con la DCT, debido a que el grado de especialización global con el que resulten las matrices no

sea muy alto; esto ocurrirá principalmente cuando el número de KLTs con el que se esté trabajando sea muy reducido, ya que los grupos de bloques resultarán más heterogéneos que con números de KLTs mayores. Luego, dado que el número de bloques que finalmente se transformarán con la DCT depende del grado de especialización con el que resulten las KLTs y éste a su vez depende en gran medida del número de KLTs con el que se esté trabajando, el umbral de MSE deberá variar con el número de KLTs.

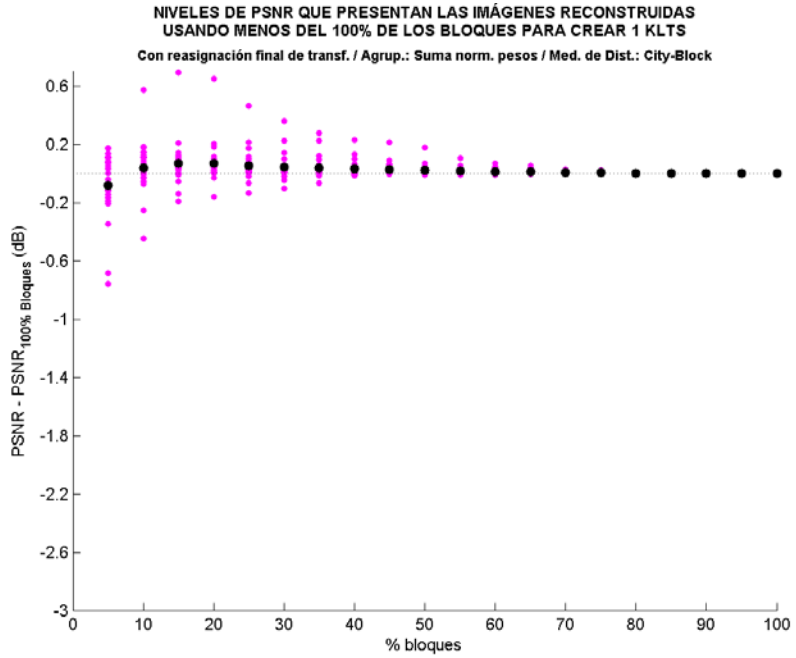
Para determinar dicho umbral, se ha realizado el siguiente experimento: en cada imagen se ha hallado el MSE de todos sus bloques tras transformarlos con la DCT y aplicarles el truncado de coeficientes y se han ordenado de mayor a menor MSE, siendo los de menor MSE los que tienen mayor probabilidad de transformarse con la DCT. Seguidamente, para cada número de KLTs entre 1 y 7, se ha ejecutado el proceso de creación de matrices, pero en vez de ejecutarlo una única vez usando la totalidad de los bloques de la imagen para crear las matrices, se ha ejecutado varias veces utilizando en cada una de ellas un porcentaje de bloques distinto, comenzando con el 5 % y aumentando de forma gradual de 5 en 5 % hasta el 100 % de los bloques (20 ejecuciones), pero siempre seleccionando para participar en la creación de las matrices los bloques con mayor MSE.

Tras la obtención de las KLTs al finalizar cada ejecución, los bloques de cada imagen son transformados utilizando reasignación de transformada mediante truncado, con dichas KLTs más la DCT. Seguidamente se les aplica truncado de coeficientes y tras ello, la transformada inversa correspondiente, obteniendo una reconstrucción de la imagen original. Finalmente, se obtiene el nivel de MSE con respecto a la imagen original y el nivel de PSNR asociado.

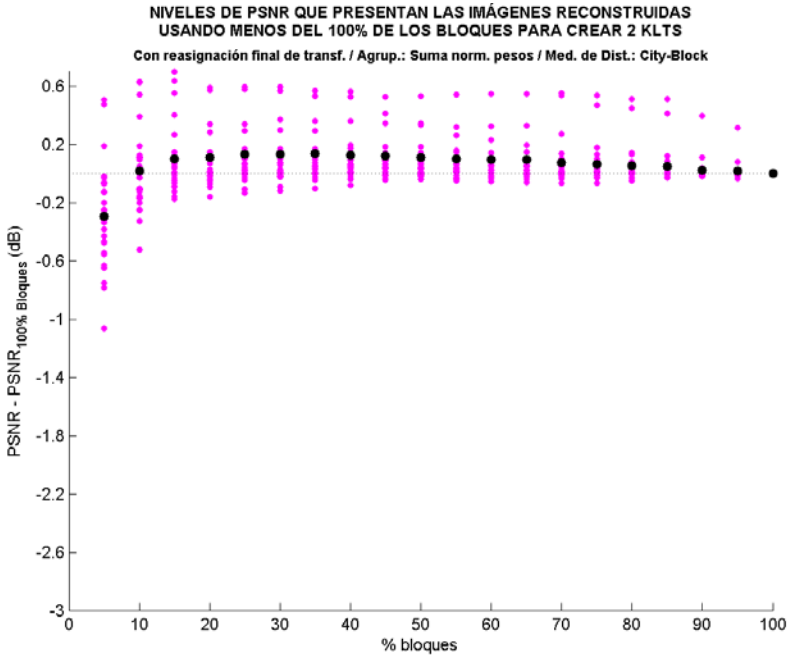
En la figura 3.25 se representan en siete gráficas (una para cada número de KLTs entre 1 y 7) los niveles de PSNR obtenidos tras variar el porcentaje de bloques con mayor MSE usado para la creación de las matrices KLT, respecto del nivel de PSNR obtenido cuando se emplean el 100 % de los bloques.

En cada gráfica, los puntos de menor tamaño y color magenta corresponden a los niveles de PSNR que han presentado las imágenes y por otro lado los puntos de color negro y mayor tamaño corresponden al valor medio de dichos niveles. Al observar las gráficas podemos verificar que, para los distintos porcentajes de bloques usados, los niveles de PSNR referenciados que presentan todas las imágenes se parecen, resultando valores muy próximos entre sí para porcentajes medios-altos y más dispersos para los porcentajes más bajos. Consecuentemente, la evolución del nivel de PSNR referenciado con el aumento del porcentaje de bloques, también es similar para todas las imágenes y para cada número de KLTs. Estas características permiten que podamos calcular una curva promedio para cada gráfica que se ajuste a los valores de PSNR mostrados. Dichas curvas ajustadas serán las que se utilizarán para verificar si los niveles máximos de PSNR se dan cuando se usa el 100 % de los bloques para crear las KLTs, o por el contrario dichos máximos se pueden alcanzar usando porcentajes menores de bloques, reduciendo así la carga computacional necesaria para la obtención de las matrices KLT.

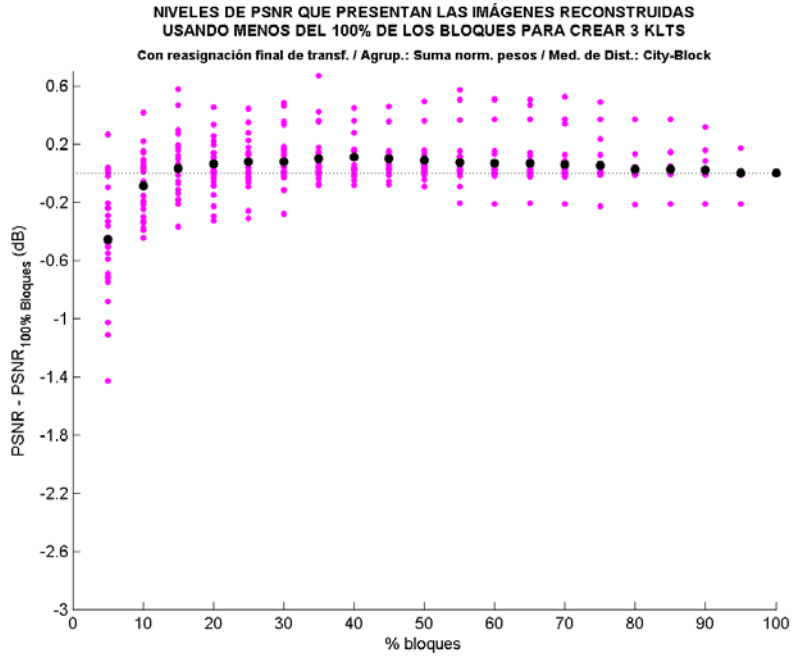
a)



b)



c)



d)

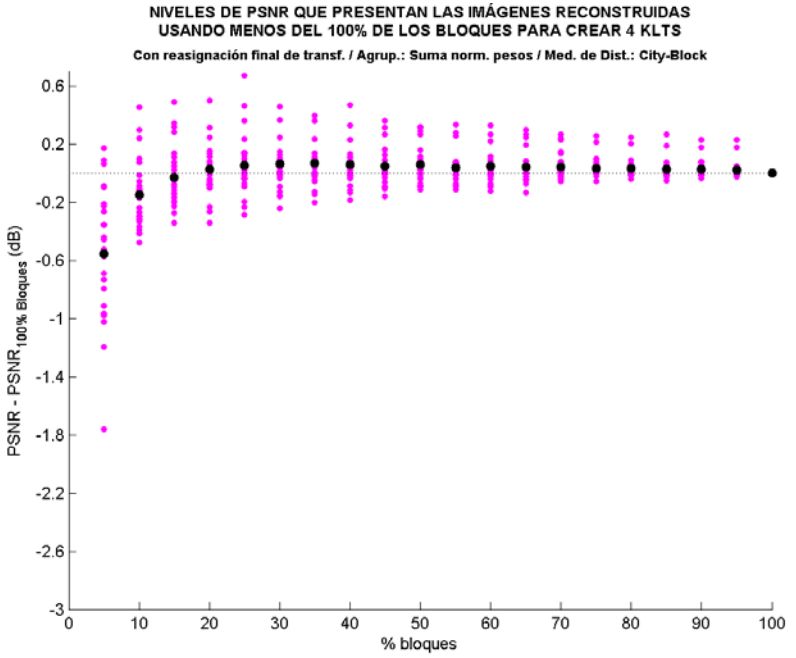
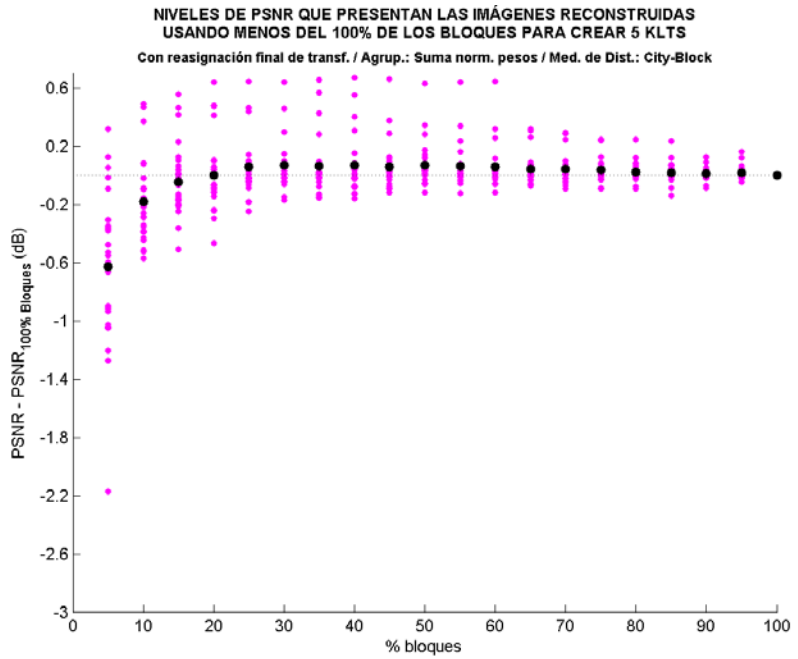
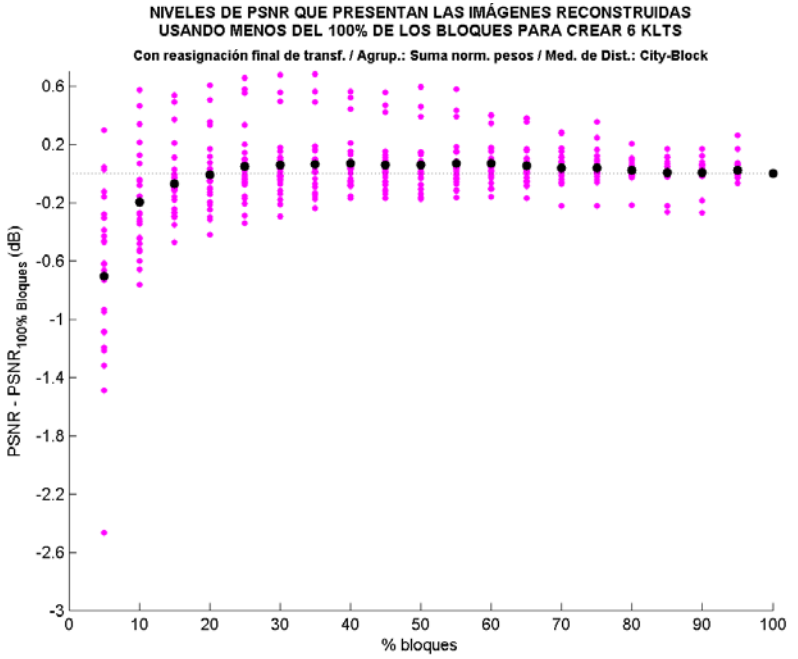


Figura 3.25 Incremento en el nivel de PSNR de las imágenes al usar porcentajes menores del 100 % de los bloques para crear las KLTs (desde 1 KLT hasta 7).

e)



f)



g)

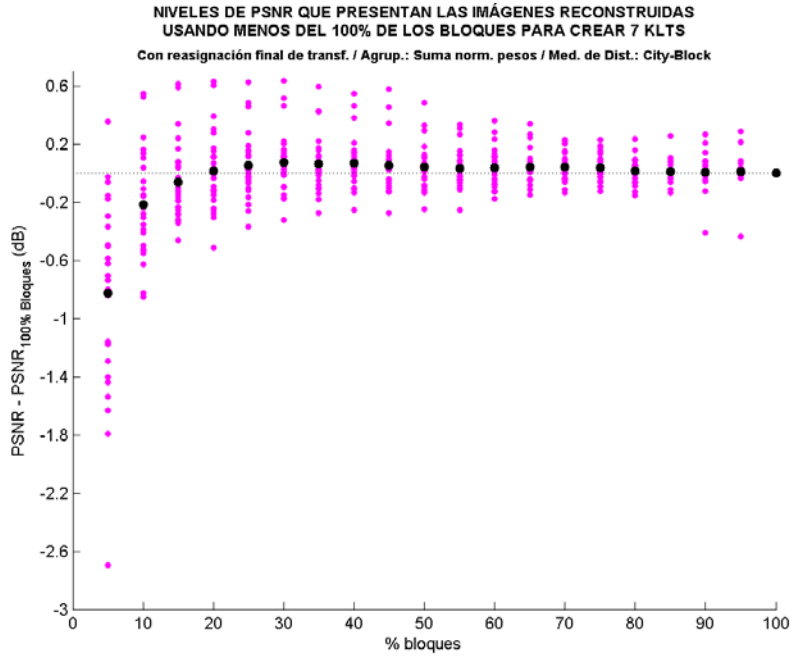


Figura 3.25 Incremento en el nivel de PSNR de las imágenes al usar porcentajes menores del 100 % de los bloques para crear las KLTs (desde 1 KLT hasta 7).

Para calcular dichas curvas promedio (una para cada número distinto de KLTs) se han utilizado los niveles medios de PSNR obtenidos para cada porcentaje (representados como puntos más grandes de color negro en la figura 3.25). Así mismo se ha hecho pruebas con dos tipos de curva: exponencial (3.11) y polinómica (3.12).

$$y = ae^{bx} - ce^{dx} \tag{3.11}$$

$$y = p_1x^9 + p_2x^8 + p_3x^7 + p_4x^6 + p_5x^5 + p_6x^4 + p_7x^3 + p_8x^2 + p_9x^1 + p_{10} \tag{3.12}$$

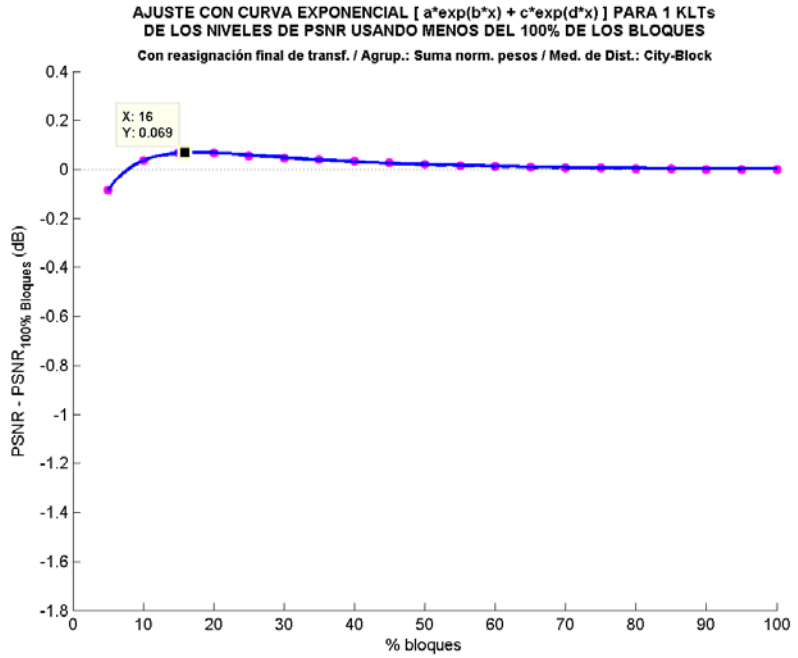
En la figura 3.26 se representan gráficamente las curvas exponenciales ajustadas a los respectivos valores promedio, para cada uno de los distintos casos de número de KLTs desde 1 hasta 7. Dichas curvas se han hallado con la herramienta "Curve Fitting" de Matlab y antes de su representación se ha reducido la precisión y se han redondeado sus valores conservando únicamente tres decimales. Así mismo, junto con las curvas exponenciales (línea azul) también se han representado en cada gráfica los puntos promedio de PSNR para cada porcentaje calculados anteriormente (puntos gruesos en color magenta), para de esta forma poder valorar gráficamente la bondad en el ajuste de la curva. Tal y como se puede apreciar en las gráficas de la figura 3.26, en la mayoría de los casos, las curvas se ajustan bastante a los respectivos valores promedio, a excepción del caso de 3 KLTs © donde el ajuste no es tan bueno y de algunos tramos finales donde no se ajusta muy bien este tipo de curva. Estas imprecisiones dan una idea de que el ajuste con curva exponencial puede que no sea el idóneo para nuestro caso. Para verificarlo, se han inspeccionado las curvas y se ha señalado en cada una el punto en el que se alcanza el máximo nivel de PSNR con el menor porcentaje de bloques y se han recogido dichos puntos en la tabla 3.11.

	$PSNR_{max}$ [dB] , Bloques [%]
1 KLT	0'069 , 16
2 KLT	0'149 , 25
3 KLT	0'101 , 26
4 KLT	0'061 , 29
5 KLT	0'074 , 34
6 KLT	0'066 , 33
7 KLT	0'068 , 29

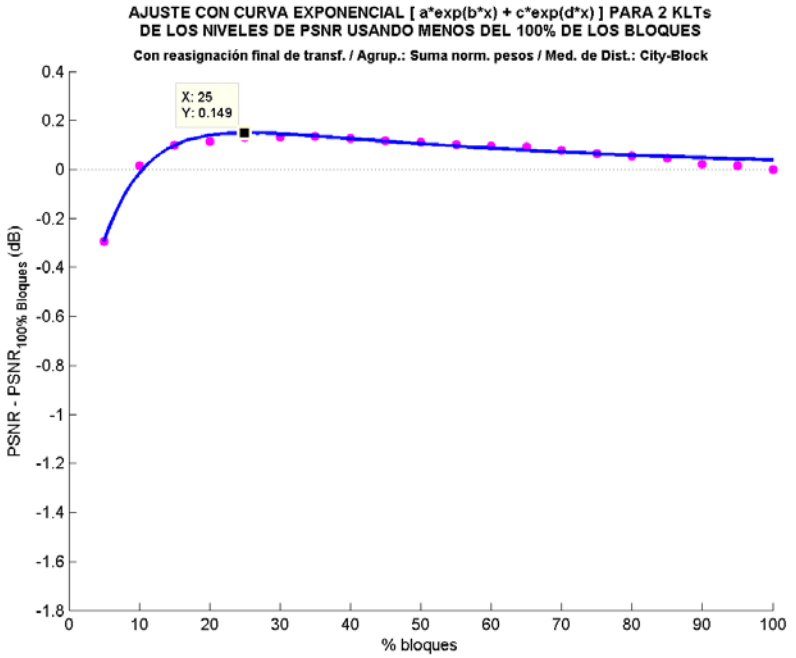
Tabla 3.11 Selección de bloques con mayor MSE para la formación de las KLTs (ajuste con curva exp.).

Según el ajuste realizado con curva exponencial, para todos los casos de números de KLT se puede conseguir un nivel de PSNR mayor al que se consigue con el 100 % de los bloques, usando porcentajes de bloques menores al 100 %, alcanzándose los máximos valores de PSNR para los porcentajes recogidos en la tabla 3.11.

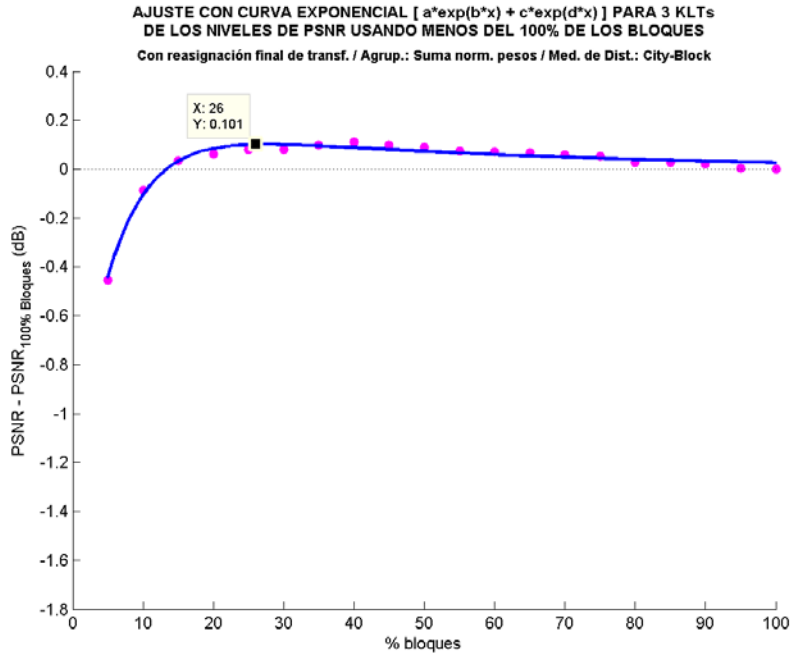
a)



b)



c)



d)

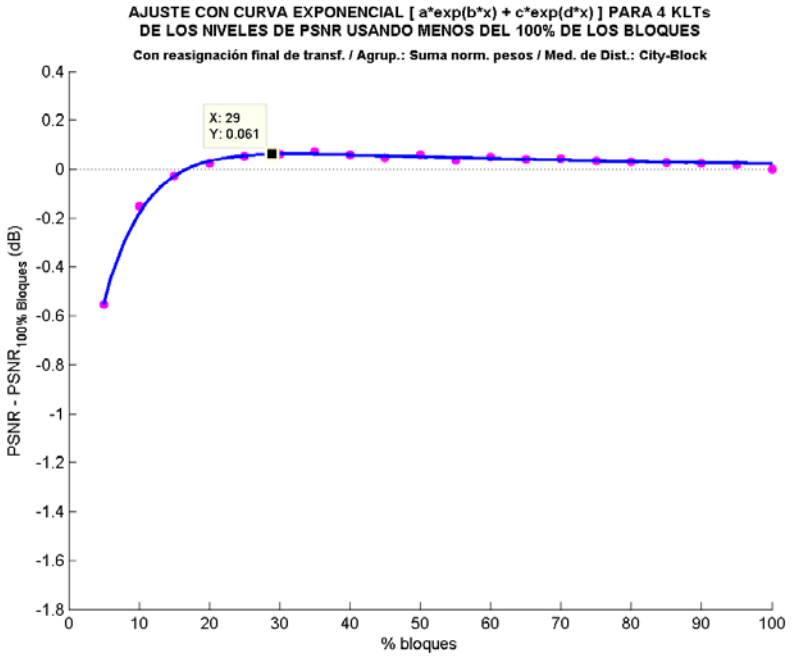
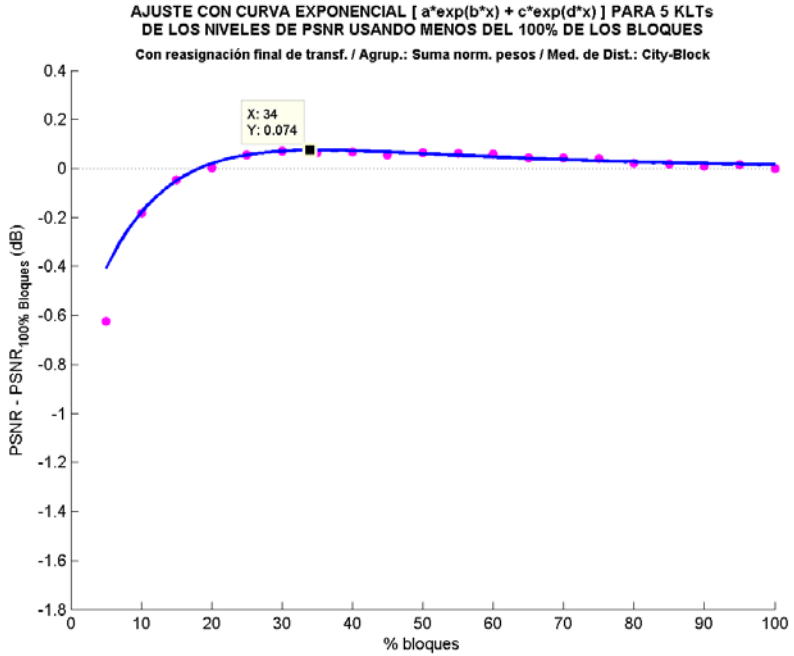
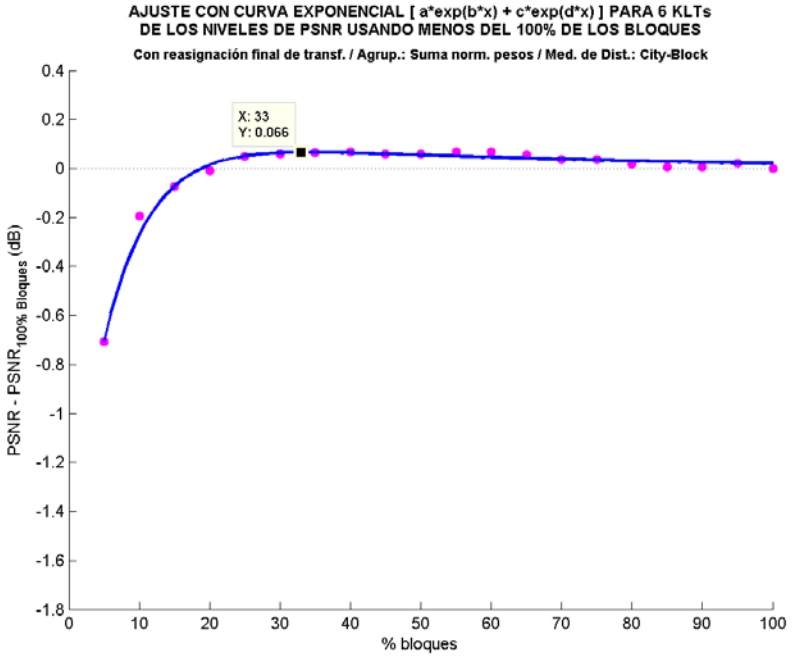


Figura 3.26 Ajuste con curva exponencial de los incrementos en el nivel de PSNR al usar porcentajes menores al 100 % de los bloques para crear las KLTs.

e)



f)



g)

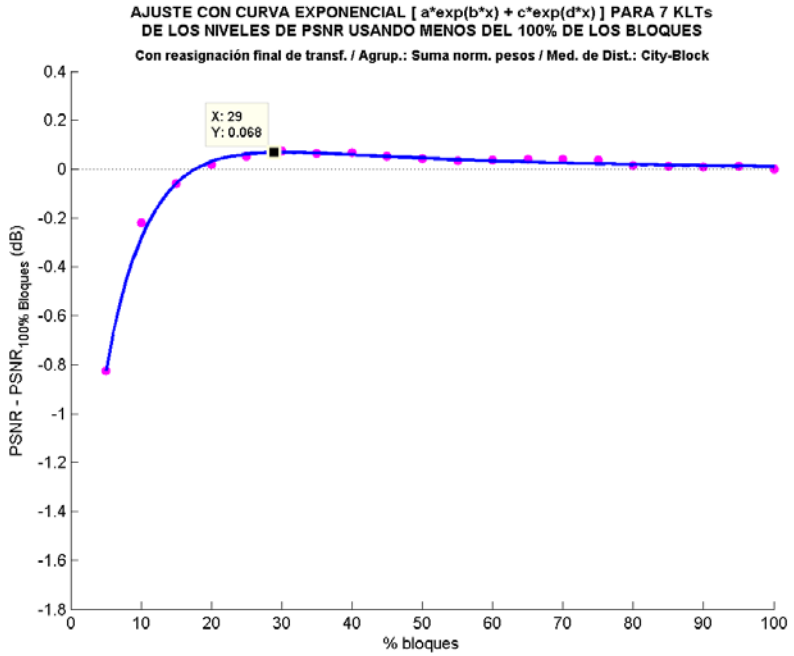


Figura 3.26 Ajuste con curva exponencial de los incrementos en el nivel de PSNR al usar porcentajes menores al 100 % de los bloques para crear las KLTs.

Para verificar que al utilizar dichos porcentajes de bloques para la obtención de las matrices KLT, se supera el nivel de PSNR que se obtiene al usar el 100 % de los bloques de la imagen, se ha realizado con las imágenes el mismo experimento con truncado de coeficientes que se viene realizando y se han hallado los niveles de PSNR globales del conjunto, pero en este caso, para cada número distinto de KLTs se han utilizado respectivamente los porcentajes de bloques recogidos en la tabla 3.11. En este experimento directamente se usa como característica de agrupamiento la suma normalizada de los pesos contenidos en las bandas de frecuencia y también se ha usado reasignación final de transformada. En la figura 3.27 se representan los valores de PSNR globales de este último experimento junto con los valores que se obtuvieron al utilizar el 100 % de los bloques.

Observando la figura 3.27, podemos comprobar que los niveles globales de PSNR que se obtienen al utilizar los porcentajes de bloques indicados por los ajustes con curva exponencial, están por encima de los niveles que se obtienen con el 100 % de los bloques, para todos los casos de número de KLTs de 1 a 7.

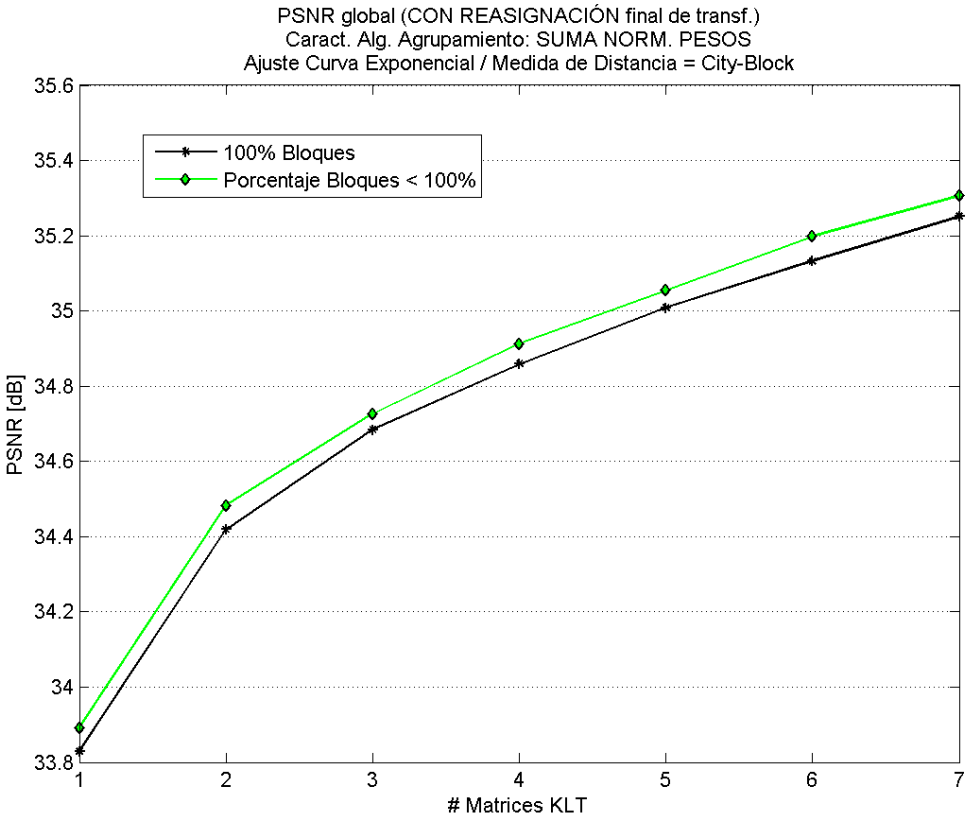


Figura 3.27 Niveles de PSNR con los porcentajes de bloques del ajuste con curva exponencial.

En la tabla 3.12 se recogen los niveles obtenidos con el 100 % de los bloques y los niveles obtenidos con los porcentajes que marcan las curvas exponenciales (tabla 3.11), para los distintos números de KLT de 1 a 7, así como sus diferencias.

	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
$PSNR_{100\_PESOS\_R}$ [dB]	33,8289	34,4188	34,6847	34,8595	35,0089	35,1326	35,2514
$PSNR_{EXP\_PESOS\_R}$ [dB]	33,8925	34,4831	34,7264	34,9131	35,0554	35,1989	35,3069
$PSNR_{EXP\_PESOS\_R} - PSNR_{100\_PESOS\_R}$ [dB]	0,0637	0,0643	0,0417	0,0536	0,0466	0,0663	0,0555

Tabla 3.12 Niveles de PSNR con los porcentajes de bloques del ajuste con curva exponencial.

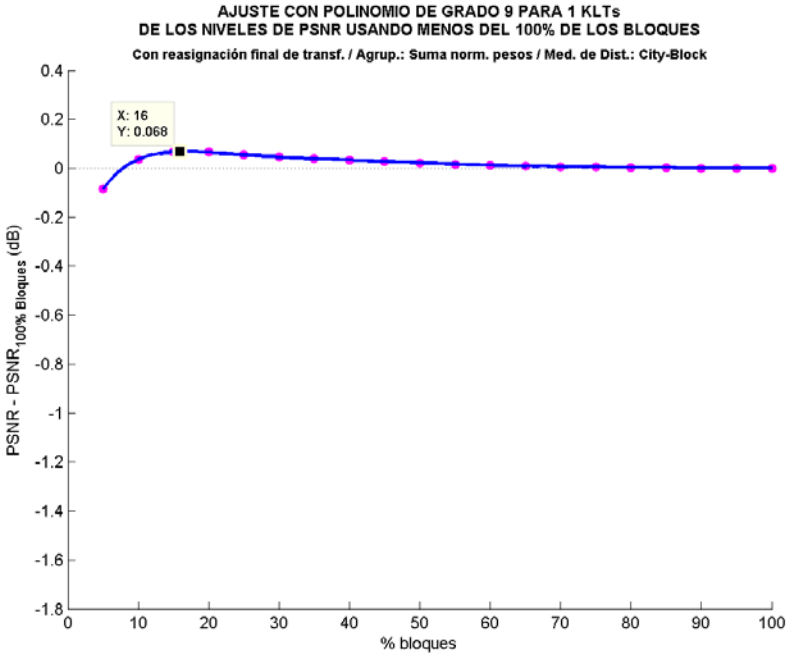
Aunque los resultados obtenidos a partir de las curvas exponenciales son satisfactorios, dado que existen tramos en los que dichas curvas no llegan a ajustarse a la evolución real de los niveles de PSNR para los distintos casos de KLT, se decide utilizar otro tipo de curva que se ajuste más. Para ello, se ha elegido como tipo de curva un polinomio de grado nueve, cuya expresión se muestra en (3.12), y se ha vuelto a utilizar la herramienta "Curve Fitting" de Matlab para calcular las distintas curvas. Aquí también se ha reducido la precisión y se han redondeado los valores de las curvas obtenidas preservando únicamente tres decimales. Las siete curvas se han representado respectivamente en las 7 gráficas recogidas en la figura 3.28, en las que también se han representado los niveles de PSNR promedio para los distintos porcentajes.

Al inspeccionar las curvas obtenidas con el ajuste polinómico (figura 3.28) se puede verificar que en este caso la bondad del ajuste es mayor que para el caso exponencial, ya que prácticamente todos los puntos están dentro de las curvas. En estas gráficas, también se han señalado los puntos en los que se alcanza el máximo nivel de PSNR, verificando que para todos los casos de número de KLTs desde 1 hasta 7, estos niveles máximos se obtienen usando porcentajes de bloques menores al 100 % para la obtención de las matrices KLT. En la tabla 3.13 se recogen dichos niveles con sus respectivos porcentajes asociados.

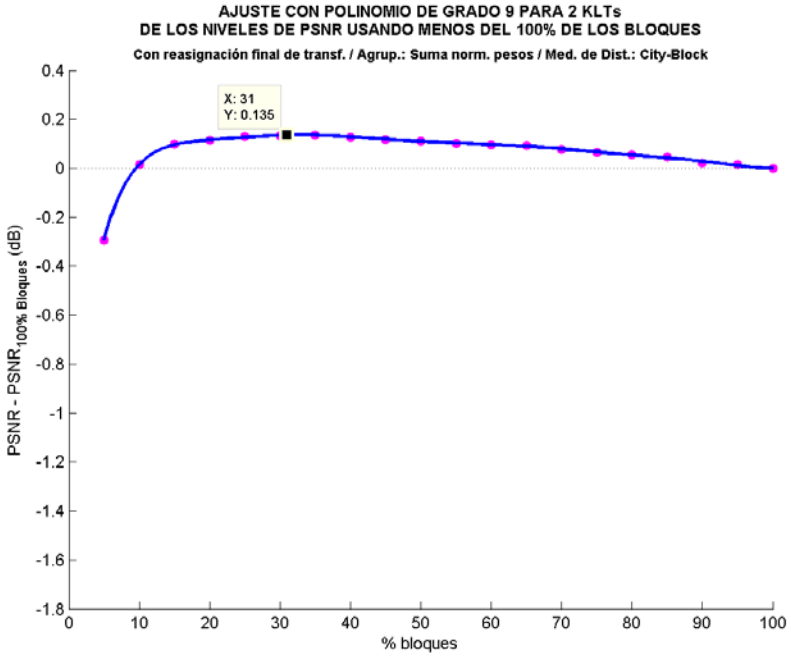
	$PSNR_{max}$ [dB] , Bloques [%]
1 KLT	0'068 , 16
2 KLT	0'135 , 31
3 KLT	0'102 , 38
4 KLT	0'068 , 33
5 KLT	0'072 , 34
6 KLT	0'072 , 34
7 KLT	0'073 , 33

Tabla 3.13 Selección de bloques con mayor MSE para la formación de las KLTs (ajuste con curva pol.).

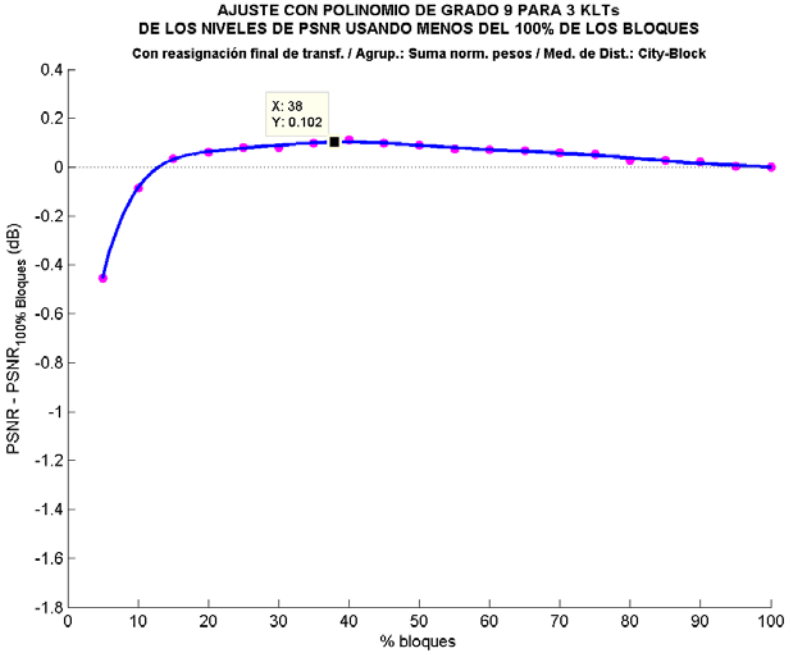
a)



b)



c)



d)

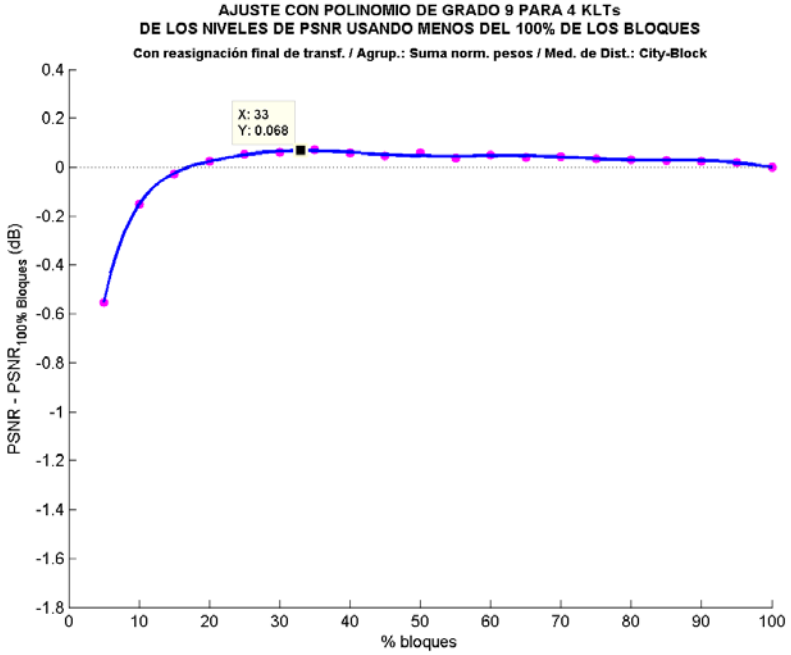
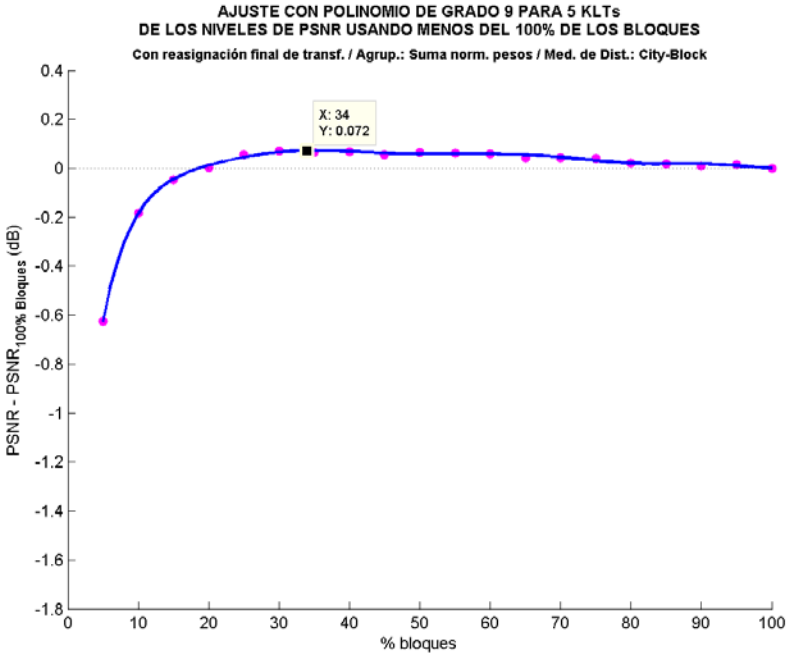
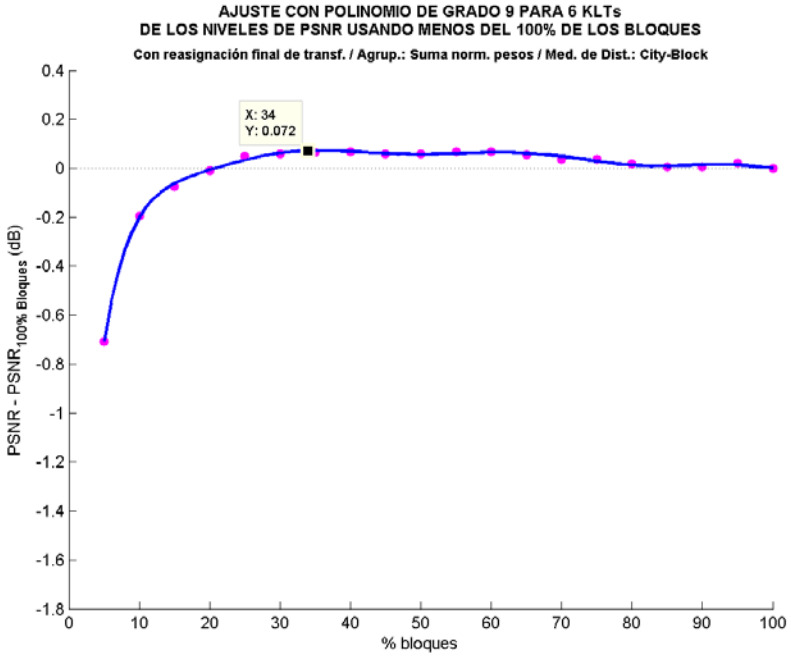


Figura 3.28 Ajuste con curva polinómica de los incrementos en el nivel de PSNR al usar porcentajes menores al 100 % de los bloques para crear las KLTs.

e)



f)



g)

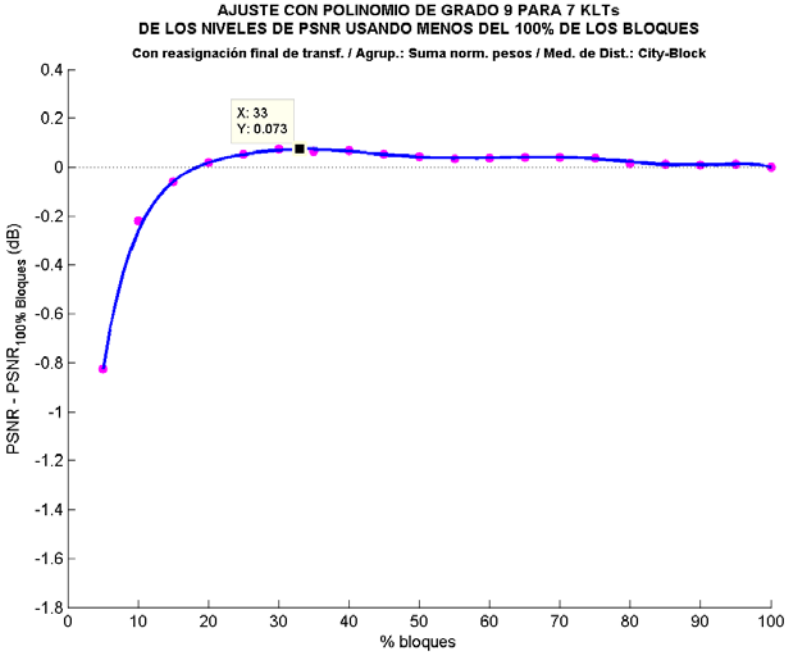


Figura 3.28 Ajuste con curva polinómica de los incrementos en el nivel de PSNR al usar porcentajes menores al 100 % de los bloques para crear las KLTs.

De nuevo, para verificar la validez de los datos recogidos en la tabla 3.13 y comprobar que se obtienen niveles de PSNR superiores al utilizar dichos porcentajes de bloques para crear las matrices KLT que al utilizar la totalidad de los mismos, se ha vuelto a repetir el experimento de truncado de coeficientes, utilizando los porcentajes de la tabla 3.13 para la obtención de las matrices KLTs y representando gráficamente en la figura 3.29 los resultados globales de PSNR que se han obtenido, junto con los niveles resultantes al utilizar el 100 % de los bloques.

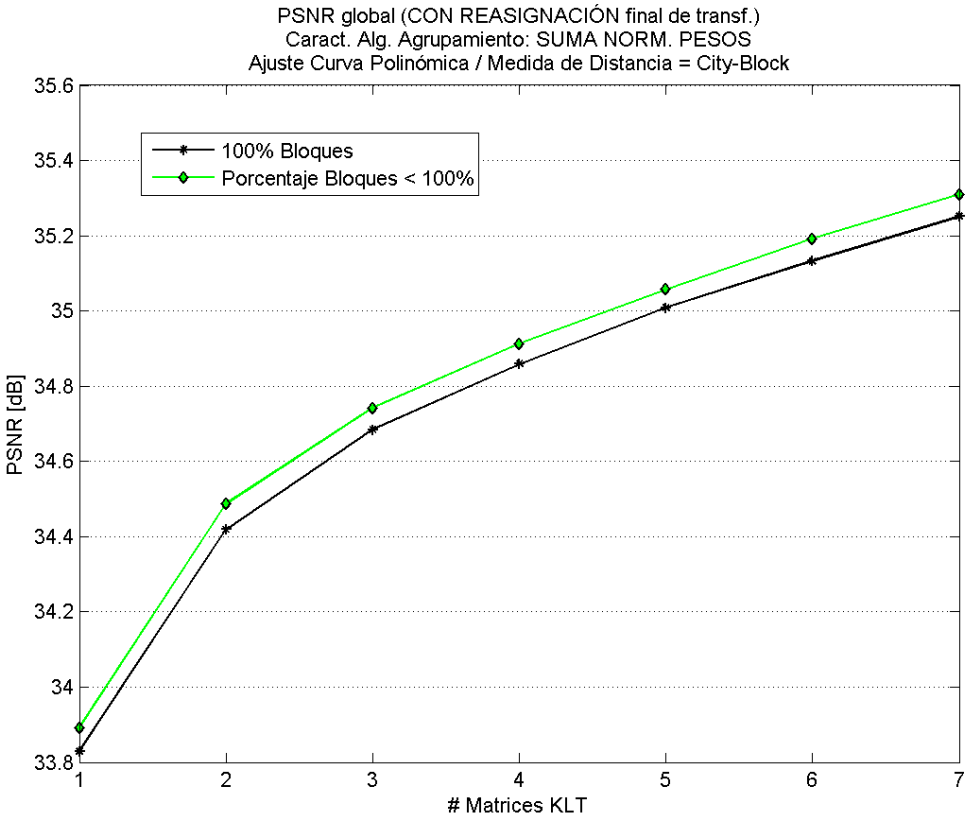


Figura 3.29 Niveles de PSNR con los porcentajes de bloques del ajuste con curva polinómica.

En la tabla 3.14 se recogen los niveles obtenidos con el 100 % de los bloques y los niveles obtenidos con los porcentajes que marcan las curvas polinómicas (tabla 3.13), para los distintos números de KLT de 1 a 7, así como sus diferencias.

	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
$PSNR_{100\_PESOS\_R}$ [dB]	33,8289	34,4188	34,6847	34,8595	35,0089	35,1326	35,2514
$PSNR_{POL\_PESOS\_R}$ [dB]	33,8925	34,4871	34,7426	34,9134	35,0574	35,1917	35,3104
$PSNR_{100\_PESOS\_R} - PSNR_{POL\_PESOS\_R}$ [dB]	0,0637	0,0683	0,0579	0,0539	0,0485	0,0591	0,0590

Tabla 3.14 Niveles de PSNR con los porcentajes de bloques del ajuste con curva polinómica.

Tal y como se puede comprobar en la figura 3.29, los niveles de PSNR globales que resultan al utilizar los porcentajes obtenidos a partir del ajuste con el polinomio de grado 9, superan los niveles que se obtienen usando la totalidad de los bloques para todos los casos de número de KLTs de 1 a 7. Además, inspeccionando los valores recogidos en la tabla 3.14, podemos verificar que para la mayoría de los casos de número de KLT se produce un incremento ligeramente superior con estos porcentajes que con los porcentajes obtenidos a partir del ajuste con curva exponencial (tabla 3.11), por lo que finalmente serán los porcentajes obtenidos con el ajuste con curva polinómica los que se usarán en nuestro codificador.

3.5.2.2 Eliminación de los Outliers

En un grupo de bloques, cuanto mayor sea el porcentaje de bloques que comparten características espaciales similares, la KLT formada a partir de dicho grupo resultará con un mayor grado de especialización para dichas características. Pero si dentro del mismo grupo, existen bloques con unas características espaciales muy diferentes a las de la mayoría, al participar también en la formación de la KLT, pueden provocar que la especialización disminuya con respecto al caso anterior. Por ello, una vez formados los grupos, sería conveniente identificar dentro de cada uno, aquellos bloques que se diferencien en demasía de la mayoría, para retirarlos del grupo y evitar que participen en la formación de la matriz KLT.

Sin embargo, los bloques que participan en el algoritmo de agrupamiento, son los bloques que mayor MSE han obtenido tras aplicar truncado de coeficientes DCT, es decir, la DCT no está especializada para estos bloques. Por ello, sería deseable que estos bloques participaran en la formación de alguna KLT para que así, dicha KLT resultara con una mayor especialización a sus características espaciales que la DCT y compactara más energía en los primeros coeficientes. Por lo tanto, si una vez formados los grupos y antes de obtener las KLTs, se retiran los bloques de cada grupo que menos se asemejan a la mayoría, se estaría evitando que las KLTs se especializaran para dichos bloques y probablemente ninguna de las transformadas disponibles compactaría la energía de estos bloques de forma óptima, por lo que el nivel de PSNR de la imagen reconstruida sería ligeramente menor que si hubieran participado en la formación de las KLTs.

Para dilucidar si los niveles de PSNR de las imágenes reconstruidas se ven beneficiados o perjudicados por excluir los bloques outlier de cada grupo antes de la formación de las KLTs, se ha vuelto a realizar el experimento aplicando truncado de coeficientes, pero identificado dichos bloques y excluyéndolos de la formación de las matrices KLT, comparando finalmente los niveles de PSNR globales que se obtienen al tratar los outliers, con los que se obtienen al no tratarlos.

Dado que K-Means es un algoritmo de agrupamiento basado en distancias, la identificación de los bloques outlier también se ha basado en distancias. Para explicar el método usado se retomará lo explicado referente al algoritmo de agrupamiento.

Cada bloque que participa en el algoritmo de agrupamiento es representado por un vector de 9 componentes, correspondientes con las características extraidas. El algoritmo trata cada vector como un punto en un espacio euclídeo de 9 dimensiones, pudiendo calcular la distancia de dicho punto hasta cualquier otro del mismo espacio. Finalizado el algoritmo, se obtienen los grupos de puntos con sus respectivos centroides finales, que como su nombre indica, el centroide de un grupo es el punto central de dicho grupo. Un punto pertenece a un grupo y no a otro porque la distancia existente desde su posición hasta el centroide de su grupo, es menor que cualquiera de las distancias a los centroides del resto de grupos.

Cuanto mayor es la concentración de puntos en las cercanías del centroide más características espaciales compartirán los bloques asociados a dichos puntos, ocurriendo lo contrario cuanto más dispersos se encuentren los puntos del grupo. Dependiendo de las características espaciales predominantes en los bloques de la imagen y del número de grupos en el que se esté dividiendo los bloques, los grupos de puntos resultarán más compactos o más dispersos, por lo que la identificación de los outliers deberá depender de este grado de compactación de cada grupo. Retomemos el resultado final del ejemplo de aplicación del algoritmo K-Means utilizado en el punto 3.3.3, en el que se aplicaba dicho algoritmo con la intención de agrupar un conjunto de puntos de dos dimensiones en tres grupos (figura 3.30). Si nos fijamos en el grupo de puntos de color azul, podemos verificar que el grado de compactación es bastante alto, mientras que si nos fijamos en el grupo de puntos de color rojo, el grado de compactación es bastante menor ya que los puntos están más dispersos entres sí. A la hora de fijar un umbral en cada grupo a partir del cual podamos considerar que los puntos son outliers, para el caso del grupo de puntos azules dicho umbral deberá ser muy conservador ya que las distancias hasta el centroide de la gran mayoría de los puntos se concentran en un rango muy pequeño. Sin embargo, en el caso del grupo de puntos de color rojo, las distancias de los puntos hasta el centroide se mueven en un rango mayor, por lo que en este caso habría que utilizar un umbral mucho menos conservador.

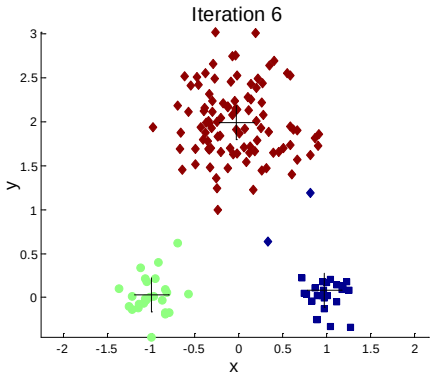


Figura 3.30 Resultado final de agrupamiento de puntos bidimensionales con el algoritmo K-Means.

Para identificar los outliers en un grupo, lo que se ha hecho ha sido calcular una distancia límite para dicho grupo teniendo en cuenta el grado de compactación del mismo, de forma que todo punto del grupo cuya distancia hasta el centroide supere dicha distancia límite se considerará un outlier. El grado de compactación presente en cada grupo nos lo va a proporcionar el rango intercuartil que presenta el conjunto de las distancias hasta el centroide de todos los puntos del grupo y que se define como la diferencia del tercer cuartil menos el primer cuartil.

Los cuartiles de una variable son los 3 valores que dividen al conjunto de datos ordenados, en 4 partes iguales conteniendo así cada una el 25 % de las muestras. Para explicarlo visualmente se utilizará un ejemplo real con una de las imágenes utilizadas: pongámonos en el caso de que se desee obtener una única KLT a partir de los bloques de la imagen 'lenna.bmp'. El porcentaje de bloques que nuestro codificador usa para crear una única KLT es del 16 % del total de la imagen (siempre eligiendo los bloques con mayor MSE), que para esta imagen supone un total de 656 bloques. En este caso no se ejecuta el algoritmo de agrupamiento porque se desea obtener una única KLT con los bloques, pero sí que se ha obtenido el centroide del conjunto y se han calculado las distancias desde los puntos hasta dicho centroide. Finalmente los 534 valores de distancia se han ordenado de menor a mayor y se ha representado su distribución en el histograma mostrado en la figura 3.31, en el que también se han indicado los 3 cuartiles de la distribución mediante barras rojas.

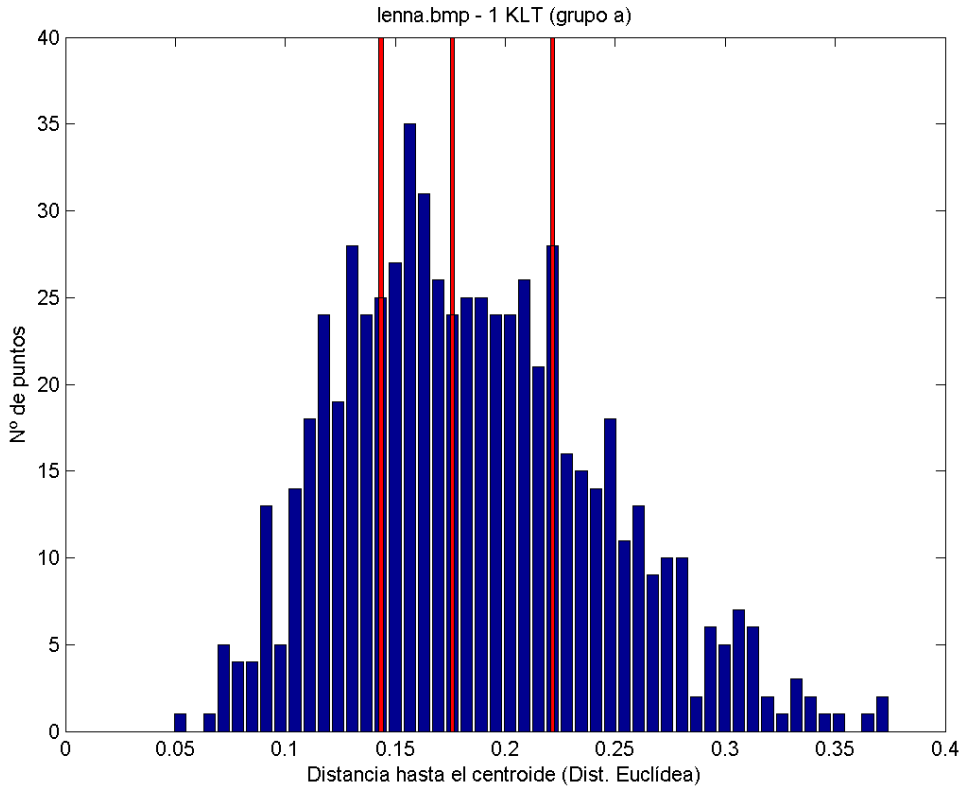


Figura 3.31 Histograma de las distancias de los puntos al centroide para 'lenna.bmp' con 1 KLT.

La primera barra roja indica la posición del primer cuartil (Q_1) y divide la distribución de forma que a la izquierda deja el 25 % de los puntos y a la derecha el 75 %. La siguiente barra roja (la central) indica la posición del segundo cuartil (Q_2) y divide la distribución en dos partes iguales (a cada lado hay el 50 % de los valores). Por último, la barra roja de la derecha marca el tercer cuartil (Q_3), el cual deja a la izquierda el 75 % de las valores y a la derecha el 25 %. El rango intercuartil (IQR) es:

$$IQR = Q_3 - Q_1 \tag{3.13}$$

En cada grupo de puntos, los valores de distancia menores al primer cuartil corresponden a los puntos más cercanos al centroide y los valores de distancia a partir del tercer cuartil pertenecen a los puntos más alejados del centroide, por lo que cuanto menor sea el rango intercuartil, mayor será el grado de compactación del grupo y viceversa, de ahí que el rango intercuartil suponga una medida directa del grado de compactación del grupo.

Por todo ello, para calcular el umbral a partir del cual los puntos se considerarán outliers, se ha definido la *distancia límite* (DL) como

$$DL = Q_3 + 1'5 \cdot IQR \tag{3.14}$$

A continuación en la figura 3.32 se representa de nuevo el histograma del ejemplo utilizado, pero señalando únicamente la posición del primer y tercer cuartil (barras rojas) y la posición de la *distancia límite* (barra verde).

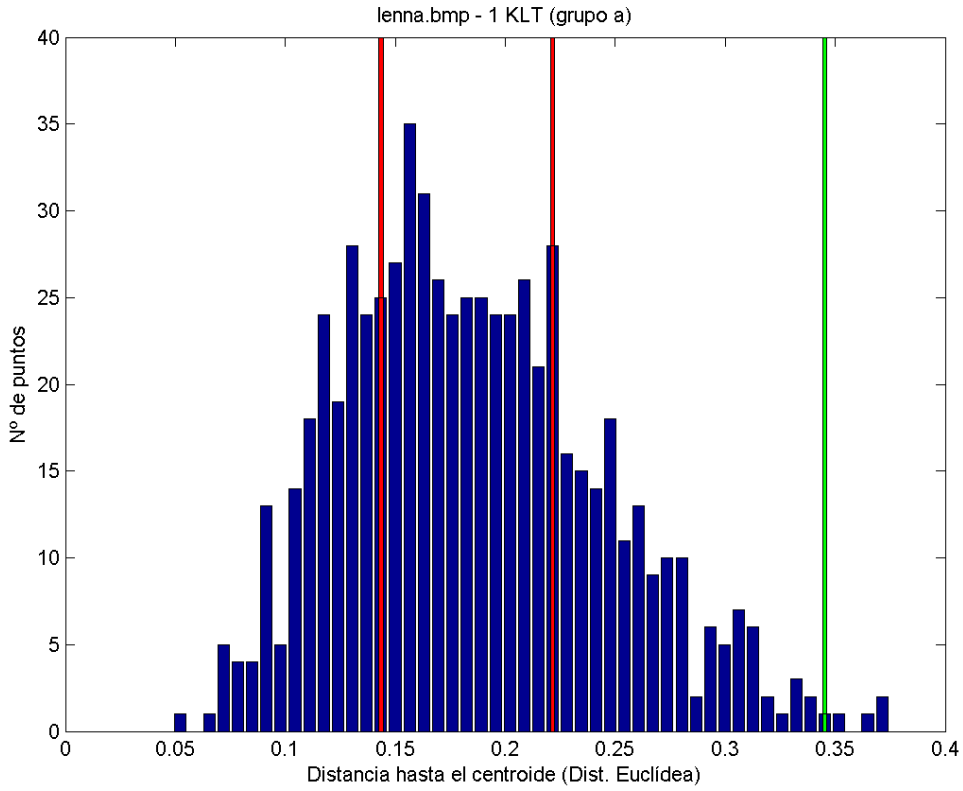


Figura 3.32 Distancia límite para 'lenna.bmp' con 1 KLT.

En la figura 3.32 podemos verificar que el rango intercuartil para este grupo es considerablemente amplio, por lo que la *distancia límite* para este caso también es un valor alto y por ello aquí sólo se han identificado 5 puntos del grupo como outliers. Para el caso en el que se crea una única KLT es de esperar que el rango intercuartil sea amplio ya que los bloques que forman el grupo no se eligen mediante el algoritmo de agrupamiento, si no que se selecciona el 16 % de los bloques de la imagen con mayor MSE tras aplicar truncado de coeficientes DCT, por lo que puede resultar un grupo de bloques muy dispares entre sí y consecuentemente conformar un grupo de puntos muy dispersos. Por el contrario, al formar los grupos mediante el algoritmo de agrupamiento, las distancias de los puntos del grupo hasta el centroide se distribuirán en un rango menor, reduciéndose la amplitud del rango intercuartil y con ello la *distancia límite*. Esto último se puede observar en la figura 3.33, en la cual se representa el histograma de las distancias de los puntos hasta el centroide, de uno de los grupos obtenidos tras aplicar nuestro algoritmo de agrupamiento a la imagen 'lenna.bmp' para obtener 4 KLTs. Para este caso el rango intercuartil es mucho más estrecho que en el anterior y consecuentemente la *distancia límite* es mucho más conservadora.

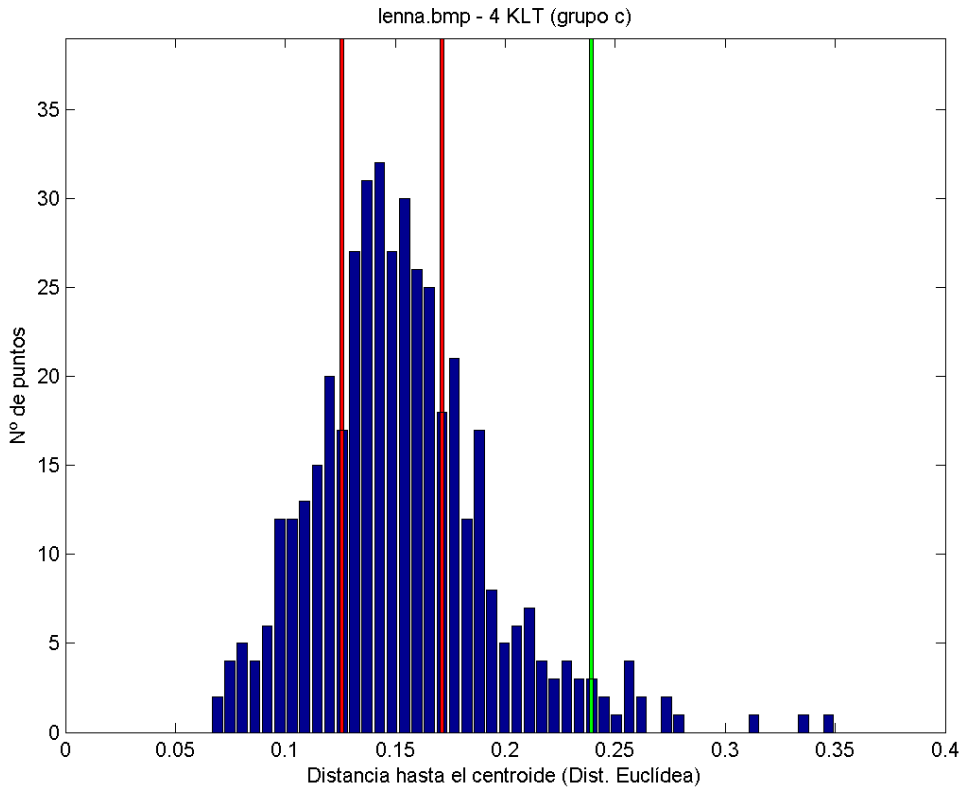


Figura 3.33 Distancia límite en el grupo c para 'lenna.bmp' con 4 KLTs.

Una vez definido el método para identificar los outliers en un grupo, se ha ejecutado el proceso de creación de matrices KLT variando el número de KLTs desde 1 hasta 7 con el juego de imágenes utilizado, pero identificando los bloques outliers de

cada grupo resultante y extrayéndolos de dichos grupos para evitar que participen en la creación de las matrices KLT. Por último en la tabla 3.15 se ha recogido la suma total de outliers identificados en los grupos para los distintos casos de número de KLTs.

Imagen	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
lenna.bmp	5	19	52	34	27	25	24
baboon.bmp	17	38	35	30	38	31	25
barbara.bmp	16	52	88	55	45	33	53
kodim01.png	22	64	69	68	77	67	63
Kodim02.png	19	42	65	66	68	53	65
kodim03.png	21	64	65	82	74	65	52
kodim04.png	47	81	81	59	51	49	49
kodim05.png	29	64	65	67	58	51	48
kodim06.png	65	136	94	74	76	72	53
kodim07.png	28	41	73	57	65	66	58
kodim08.png	21	75	46	48	62	57	51
kodim09.png	22	43	77	59	67	79	76
kodim10.png	13	38	53	44	43	31	48
kodim11.png	42	89	83	66	40	51	50
kodim12.png	0	113	141	46	44	53	45
kodim13.png	25	69	75	61	67	59	68
kodim14.png	20	88	89	47	33	43	36
kodim15.png	107	149	102	54	61	45	55
kodim16.png	62	124	111	70	54	59	64
kodim17.png	39	36	62	40	49	52	55
kodim18.png	19	53	64	57	66	53	53
kodim19.png	4	129	66	53	52	56	45
kodim20.png	26	103	57	40	50	32	29
kodim21.png	40	46	69	51	60	71	57
kodim22.png	51	109	71	65	56	57	50
kodim23.png	29	22	47	34	52	59	46
kodim24.png	12	47	71	45	49	59	58

Tabla 3.15 Total de outliers identificados para los distintos números de KLT.

Finalmente, para verificar si el tratamiento de los outliers descrito supone una mejora global, se ha vuelto a realizar el experimento de truncado de coeficientes con el juego de imágenes, pero ahora en cada grupo, previo al cálculo de su matriz KLT, se han identificado los outliers y se han retirado del grupo los bloques asociados. Los

niveles de PSNR que se han tomado como referencia, son los resultantes de utilizar para crear las matrices KLT los porcentajes de bloques recogidos en la tabla 3.13. Los niveles globales de PSNR obtenidos se representan gráficamente en la figura 3.34.

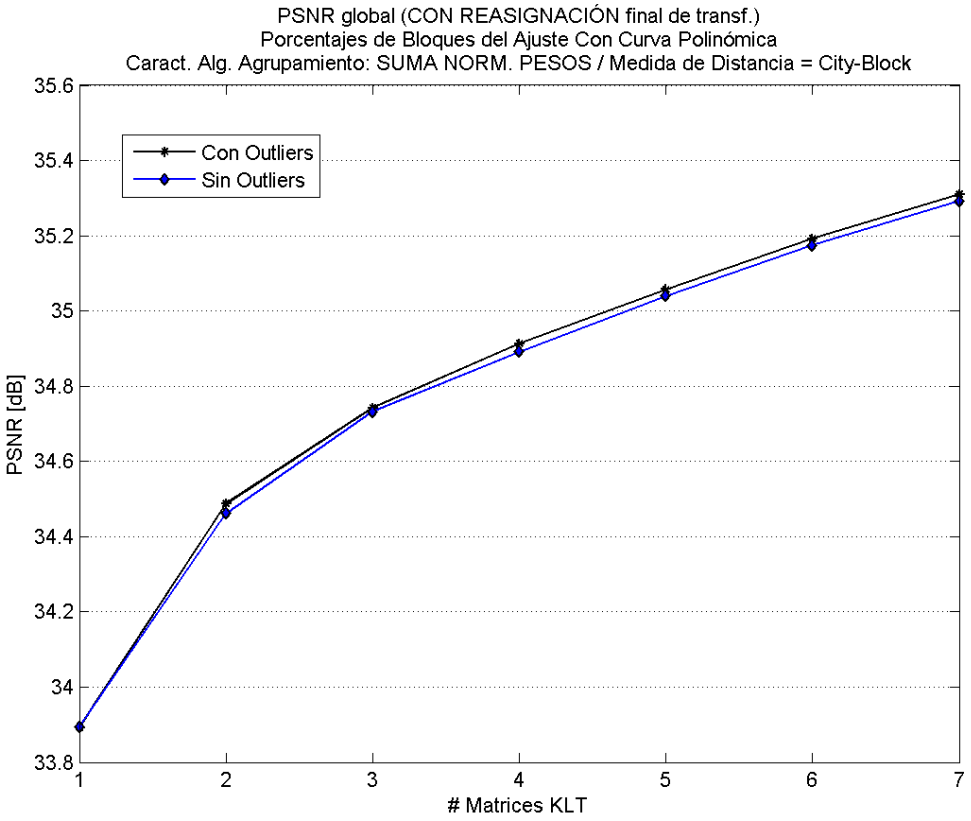


Figura 3.34 Evolución de los niveles globales de PSNR con tratamiento de outliers.

Tal y como se puede apreciar en la figura 3.34, los niveles globales de PSNR que se obtienen al tratar los outliers son ligeramente inferiores a los que se obtienen sin tratarlos, por lo que podemos concluir que la influencia negativa de estos bloques en la especialización de las matrices KLT, supone un perjuicio menor que el que conlleva excluirlos de la formación de las matrices y por consiguiente, se ha decidido que dichos bloques outliers no se tratarán, permitiendo así que participen en la formación de las matrices KLT.

3.5.3 Uso De Otra Medida De Distancia En El Algoritmo de Agrupamiento

Como se ha explicado en puntos anteriores, el algoritmo de agrupamiento en cada iteración asigna a cada bloque el centroide más cercano de todos los disponibles, evaluando las distancias desde el punto asociado al bloque hasta los diferentes centroides. Hasta aquí, en todas las pruebas que se han realizado, el tipo de medida que

se ha utilizado para calcular dichas distancias ha sido la city-block, obteniendo unos resultados satisfactorios. Sin embargo, se ha efectuado un último experimento usando otro tipo de distancia para verificar si los resultados globales de PSNR del conjunto de imágenes utilizado para las pruebas varían. La distancia elegida ha sido la euclídea, ya que al igual que la city-block, es una de las más usadas.

Los resultados de PSNR globales se han obtenido repitiendo el mismo experimento de truncado de coeficientes, pero utilizando la distancia euclídea para el algoritmo de agrupamiento, en vez de la city-block. Además, el codificador se ha configurado según los resultados de los experimentos realizados previamente: 1) se realiza reasignación final de transformada mediante truncado de coeficientes; 2) el vector de características de cada bloque se forma a partir de la suma de pesos normalizados de cada banda; 3) para formar los diferentes conjuntos de KLTs, se usan los bloques con mayor MSE tras truncar los coeficientes DCT, según los porcentajes de la tabla 3.13; 4) no se realiza ningún tratamiento con los bloques outliers de cada grupo permitiéndoles participar en la formación de las matrices. La tabla 3.16 recoge los niveles de PSNR obtenidos utilizando la distancia euclídea, junto con los que se obtienen al utilizar la distancia city-block y en la figura 3.35 se representan gráficamente dichos valores.

	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
$PSNR_{CITY\text{-}BLOCK}$ [dB]	33,8925	34,4871	34,7426	34,9134	35,0574	35,1917	35,3104
$PSNR_{EUCLÍDEA}$ [dB]	33,8925	34,4970	34,7372	34,9058	35,0545	35,1745	35,2834

Tabla 3.16 Niveles de PSNR globales obtenidos con la distancia euclídea.

Tal y como se puede observar en la figura 3.35, para el caso de 2 KLTs el nivel de PSNR global es ligeramente superior cuando se utiliza la distancia euclídea en lugar de la city-block, sin embargo, cuando el número de KLTs es superior a 2, los niveles globales obtenidos con la distancia euclídea siempre están por debajo que los que se obtienen con la distancia city-block, sobre todo para 6 y 7 KLTs. Dado que para 2 KLTs la ventaja de la distancia euclídea frente a la city-block es muy pequeña, se ha decidido que en el algoritmo de agrupamiento del codificador propuesto en este proyecto, únicamente se utilice la distancia city-block.

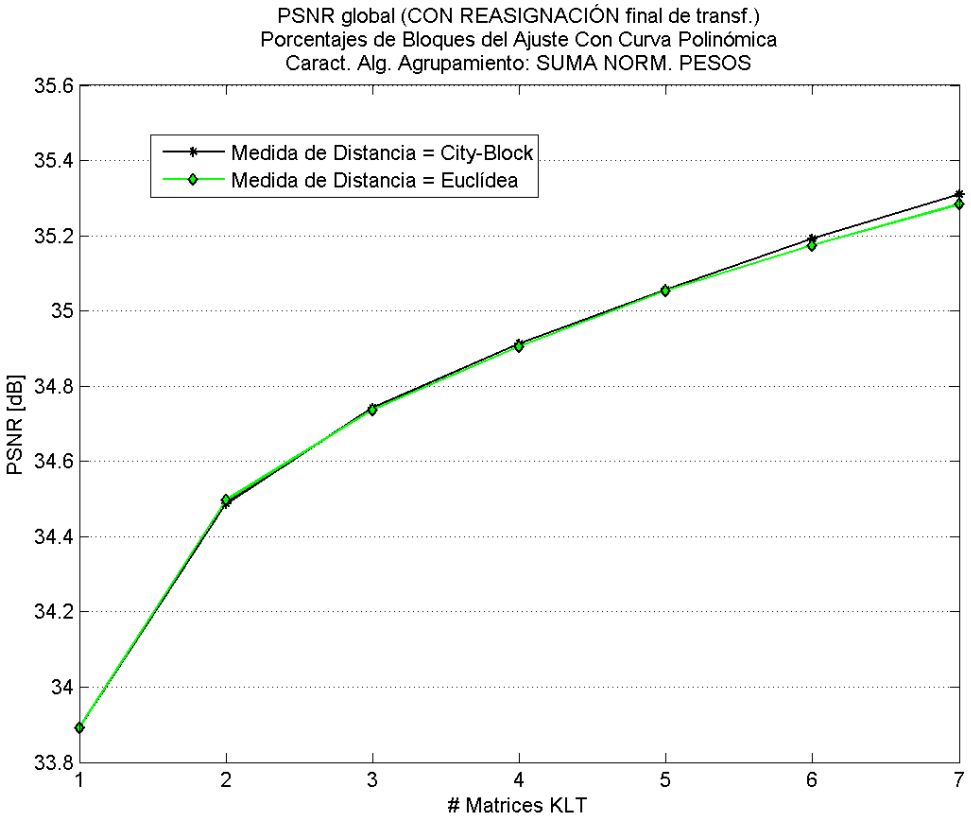


Figura 3.35 Niveles de PSNR globales con la distancia city-block y con la distancia euclídea.

3.6 CODIFICACIÓN USANDO MÁS DE UNA TRANSFORMADA

Los procesos de codificación de las imágenes en JPEG y en nuestro codificador adaptativo son muy similares ya que ambos aplican fases similares de transformación, cuantificación y codificación entrópica a cada bloque, con los mismos objetivos: aplicar una transformación lineal a los píxeles de cada bloque para compactar la energía en unos cuantos coeficientes, cuantificar los coeficientes transformados para anular los que poseen poca energía y reducir el rango de los más energéticos y por último, codificar los coeficientes cuantificados mediante codificación Huffman. Pero entre ambos procesos de codificación existe una diferencia notable ya que a la hora de transformar un bloque, en JPEG únicamente se utiliza la DCT, pero en nuestro codificador existe un conjunto de transformadas candidatas, formado por las KLTs resultantes de la Etapa I junto con la DCT. Por todo ello, las fases de transformación y cuantificación de nuestro codificador adaptativo se han diseñado teniendo en cuenta los dos tipos de transformada que se pueden aplicar a un bloque (KLT y DCT). Además, se ha añadido una nueva fase de selección cuya función es elegir para cada bloque la transformada, del conjunto de candidatas, óptima para dicho bloque. Finalmente, la fase de codificación entrópica también se ha modificado ligeramente con respecto a la fase análoga de JPEG para que, junto a la información del bloque, también se codifique la señalización de la transformada seleccionada.

3.6.1 TRANSFORMACIÓN

Cuando un bloque de píxeles llega a la fase de transformación de nuestro codificador, en vez de transformarse únicamente con la DCT como ocurre en JPEG, el bloque es transformado linealmente con todas las matrices disponibles del conjunto de transformadas candidatas (DCT más KLTs de la Etapa I), obteniendo así un número de bloques transformados distintos, igual al número de transformadas candidatas. Dado que la selección de transformada para el bloque en cuestión se efectúa tras la cuantificación, todos los bloques transformados obtenidos en esta fase son entregados a la siguiente fase de cuantificación, junto con la señalización de la transformada usada con cada bloque.

Señalar que la transformación del bloque con la DCT se hace en su forma bidimensional (exactamente igual que en JPEG) obteniendo un bloque de coeficientes transformados de tamaño 8×8, pero la transformación con una KLT se hace unidimensionalmente, por lo que el bloque es vectorizado previamente como indica la figura 3.11, resultando un vector de coeficientes transformados de tamaño 64×1. Por lo tanto, a la fase de cuantificación se le entrega el bloque transformado DCT y los vectores transformados KLT.

3.6.2 CUANTIFICACIÓN

En esta fase, los coeficientes del bloque transformado DCT y de los vectores transformados KLT, son divididos cada uno con un escalón de cuantificación determinado, conservando únicamente la parte entera del resultado. Es en la tabla de cuantificación, especificada de forma externa al codificador, en donde se indica qué escalón se aplica a cada coeficiente. Finalizada esta fase, todos los vectores KLT cuantificados y el bloque DCT cuantificado son transferidos a la fase de Selección.

Como ya se ha comentado, lo que se busca con la cuantificación es anular los coeficientes energéticamente poco significativos y disminuir la precisión de los coeficientes energéticamente significativos; de esta forma se mejora la codificación entrópica de los coeficientes transformados, reduciéndose el número de bits necesarios para codificar el conjunto de coeficientes.

La cuantificación es un proceso irreversible y por lo tanto no es posible recuperar la pérdida de información que provoca, de ahí que tras el proceso de cuantificación inversa, que consiste en multiplicar cada coeficiente cuantificado por el mismo escalón que se utilizó para dividirlo, normalmente no se recuperan los coeficientes transformados originales, pero sí valores aproximados. Dado que la proximidad o lejanía de los valores recuperados con respecto a los valores originales depende del escalón utilizado con cada coeficiente, es necesario utilizar una tabla de cuantificación que tenga en cuenta la distribución de la energía en los coeficientes del bloque, para así utilizar escalones pequeños con los coeficientes que contengan mayor

energía y escalones grandes con los coeficientes que no contengan prácticamente energía. De esta forma, se conserva en su mayoría la energía de los coeficientes con mayor peso en la formación del bloque original y se anulan los coeficientes que prácticamente no aportan información en el bloque original.

3.6.2.1 Cuantificación De Los coeficientes DCT

Tal y como se explica en el punto 3.4.3, en el que se hace un análisis de la distribución energética que poseen los bloques transformados con la DCT procedentes de bloques de 8×8 píxeles de imágenes, la DCT en el caso particular de imágenes, provoca una distribución energética típica en los bloques transformados. En dicha distribución, un porcentaje muy alto del total de la energía del bloque se concentra en el coeficiente DC y en los coeficientes AC más próximos a él, conteniendo mucha más energía el coeficiente DC que los coeficientes AC, y el resto de la energía se distribuye en los coeficientes más alejados del DC. Además, los coeficientes AC cercanos al DC son los asociados a las bajas frecuencias espaciales, las cuales poseen la información más relevante psicovisualmente hablando, mientras que los coeficientes AC más alejados del DC son los asociados a las frecuencias altas, que son las responsables de los detalles más finos del bloque y consecuentemente menos importantes. Por estas razones, la tabla de cuantificación que debe usarse para un bloque DCT debe aplicar escalones de cuantificación pequeños a los coeficientes más cercanos al DC e incrementar el tamaño de los escalones según se distancia del DC.

JPEG está diseñado para utilizar cualquier tabla de cuantificación, aunque el estándar recomienda una tabla específica, la cual se obtuvo tras la realización de un experimento subjetivo cuyos resultados se recogen en [Lohscheller, 1984] y se muestra en la figura 2.2 (luminancia). Esta tabla es la que se ha usado en todas las pruebas de este proyecto para cuantificar los coeficientes DCT, tanto en el codificador JPEG como en nuestro codificador y en la misma, se puede verificar que el tamaño de los escalones es menor para los coeficientes más cercanos al DC y que según nos alejamos el tamaño aumenta prácticamente de forma gradual.

3.6.2.2 Cuantificación De Los Coeficientes KLT

Al igual que la DCT, la KLT también provoca una distribución energética típica en los coeficientes de los vectores transformados. Como se detalla en el punto 3.4.2, cuando un vector se transforma con una KLT especializada para las estadísticas de dicho vector, un porcentaje muy alto de la energía se compacta en los primeros coeficientes del vector transformado, distribuyéndose la energía restante en los coeficientes situados en las posiciones medias y finales del vector, disminuyendo de forma gradual hasta las últimas posiciones. De forma similar a como ocurre con la DCT, el primer coeficiente transformado del vector concentra el mayor porcentaje de la energía (porcentaje similar al que concentraría el DC si se hubiera transformado el bloque con la DCT).

Por otro lado, con la KLT no se pueden utilizar tan directamente como con la DCT, criterios psicovisuales para identificar los coeficientes con mayor o menor importancia, ya que los vectores base que forman el núcleo de transformación de la KLT no son siempre los mismos, sino que varían para cada KLT debido a que son dependientes de los datos que se utilicen para crear las matrices.

Sin embargo, sí sabemos que los coeficientes de un vector transformado KLT resultan ordenados en orden decreciente según la varianza de las variables transformadas asociadas, es decir, en la primera posición de cada vector transformado KLT se encuentran los valores de la variable transformada con mayor varianza, en la segunda posición se encuentran los valores de la segunda variable transformada con mayor varianza y así sucesivamente hasta la última posición de los vectores transformados en los que se encuentran los valores de la variable transformada con menor varianza. Por otro lado, como se explica en el punto 3.4.2, las variables transformadas con mayor varianza son las que aportan mayor información en la creación del vector original y las que tienen menor varianza son las que aportan muy poca información, por lo tanto es aconsejable que el proceso de cuantificación para un vector transformado KLT utilice escalones pequeños para las primeras posiciones del vector, puesto que son los coeficientes asociados a las variables con mayor varianza, y aumentar progresivamente el tamaño de los escalones según se vaya desplazando por el vector hasta la última posición.

Por todo ello, para cuantificar los coeficientes de los vectores transformados KLT, nuestro codificador utiliza los mismos escalones que los usados para cuantificar los bloques transformados DCT (figura 2.2 luminancia), pero ordenados en orden creciente, asignando así los escalones más pequeños a los coeficientes situados en las primeras posiciones y los mayores a los coeficientes situados en las últimas posiciones.

3.6.3 SELECCIÓN

Esta fase es la encargada de encontrar para el bloque que se está codificando la transformada óptima del conjunto de transformadas candidatas, así como de entregar al codificador entrópico los coeficientes transformados y cuantificados asociados a dicha transformada, junto con la señalización de la misma.

Realmente esta fase no sería necesaria si se pudiera extraer de cada bloque las características idóneas para caracterizarlo espacialmente de forma precisa y a su vez, el algoritmo de agrupamiento fuera capaz de dividir el conjunto de bloques de una imagen en el número óptimo de grupos según las características espaciales predominantes en los bloques; de esta forma, cada grupo resultante tendría unas características espaciales totalmente diferenciadas del resto y por lo tanto, no habría ambigüedad sobre la pertenencia de un bloque a un grupo u otro. Esto es así porque cuando un grupo sólo contiene bloques con unas características espaciales determinadas, la matriz KLT que se obtiene a partir de dicho grupo está muy especializada para dichas características

espaciales. Por el contrario, si el grupo está compuesto por bloques con varias características espaciales, la especialización de la KLT para cada una de estas características será menor que en el caso anterior. Por ello, si el algoritmo de agrupamiento obtiene unos grupos muy bien diferenciados, es muy improbable que la KLT de un grupo compacte de forma eficiente la energía en un bloque de otro grupo y por lo tanto, no sería necesario seleccionar para cada bloque qué transformada del conjunto disponible de candidatas es la óptima para él, ya que directamente se podría utilizar la asignación del algoritmo de agrupamiento.

Pero realmente no podemos asegurar que esto pase en nuestro codificador, ya que aunque se estén extrayendo unas características que son capaces de caracterizar espacialmente a un bloque en gran medida, el número de grupos viene impuesto de forma externa al codificador y por lo tanto, no podemos asegurar que el número utilizado sea el idóneo como para que los grupos de bloques resultantes posean unas características espaciales totalmente diferenciadas entre ellos.

Además, también hay que tener en cuenta el efecto de la cuantificación de los coeficientes, ya que aunque una transformada compacte la energía de forma óptima en un bloque, se puede dar el caso que la tabla de cuantificación usada no sea la idónea para el tipo de compactación que provoca esa transformada y por lo tanto, la pérdida de energía tras la cuantificación puede ser considerable, pudiendo darse el caso que otra transformada del conjunto disponible, a pesar de no compactar la energía tan eficientemente, funcione mejor.

La última razón para no utilizar la asignación de transformada resultante del algoritmo de agrupamiento, es que no todos los bloques de la imagen participan en el algoritmo, es decir, que hay un grupo de bloques para el que no se está especializando ninguna KLT porque se da por hecho directamente que para estos bloques la DCT compacta la energía de forma óptima. Pero realmente, dado que la selección de dichos bloques se hace mediante la evaluación del MSE que resulta tras aplicarles truncado de coeficientes, no podemos asegurar que se sigan obteniendo los mismos resultados al aplicar cuantificación de coeficientes.

Por todo lo anterior, no se aplica a cada bloque la transformada asignada tras el algoritmo de agrupamiento, sino que se elige para cada bloque la transformada óptima del conjunto de candidatas. La forma de hacerlo es midiendo directamente la distorsión que presenta cada uno de los bloques reconstruidos, tras transformar el bloque original que se está codificando con cada una de las transformadas del conjunto y aplicarles cuantificación de coeficientes a los bloques transformados obtenidos.

Dado que de la fase de cuantificación directamente se reciben el bloque DCT cuantificado y los vectores KLT cuantificados, lo que hace la fase de selección es invertir los procesos de cuantificación y transformación para obtener las reconstrucciones del bloque original. Para ello, lo primero que hace es aplicar la cuantificación inversa, que consiste en multiplicar los coeficientes por el mismo escalón que se utilizó para cuantificarlos, obteniendo un bloque DCT transformado y unos

vectores KLT transformados aproximados a los originales. Seguidamente lo que se hace es aplicar la transformada inversa correspondiente al bloque DCT y a los vectores KLT, obteniendo así reconstrucciones del bloque original, bloque en el caso de la DCT y bloque vectorizado en el caso de la KLT, que serán aproximaciones del mismo puesto que cada uno presentará una determinada distorsión debida a la energía eliminada en el proceso de cuantificación. Después se calcula la distorsión presente en cada reconstrucción mediante el cálculo del MSE y se asigna a cada bloque la transformada que haya obtenido el menor valor de MSE. Finalmente, se transfieren a la fase de codificación entrópica, los coeficientes transformados y cuantificados asociados a dicha transformada, junto con la señalización de la misma, que no es más que el número entero que identifica a cada transformada, que será 0 cuando la transformada seleccionada sea la DCT o mayor que 0 (entre 1 y 7) cuando lo sea una de las KLTs.

Dado que el codificador entrópico trabaja con cadenas de coeficientes, si la transformada seleccionada es la DCT, el bloque de coeficientes DCT cuantificados debe ser vectorizado. La forma de vectorizar los bloques DCT en nuestro codificador es la misma que se utiliza en el estándar JPEG y es siguiendo una secuencia en zigzag, comenzando con el coeficiente DC y continuando con los coeficientes AC tal y como muestra la figura 2.18. Vectorizar siguiendo esta secuencia en zigzag permite que aparezcan cadenas de coeficientes nulos consecutivos, lo cual hace que el codificador entrópico codifique el bloque de forma más eficiente. Por el contrario, si la transformada es una de las KLTs, el vector de coeficientes cuantificados es transferido directamente a la fase de codificación entrópica.

3.6.4 CODIFICACIÓN ENTRÓPICA

Esta fase es la encargada de codificar la información de un bloque añadiendo compresión sin pérdidas mediante el uso de códigos de longitud variable. La información a codificar de un bloque está formada por la cadena de coeficientes cuantificados junto con la señalización de la transformada.

3.6.4.1 Codificación De La Cadena de Coeficientes

La codificación de la cadena de coeficientes cuantificados se realiza utilizando la misma técnica que en JPEG, en la que no se codifican explícitamente los coeficientes AC nulos, sino que son codificados de forma implícita junto con los coeficientes AC no nulos. Para ello cada coeficiente AC distinto de cero, se codifica mediante el uso de dos símbolos: el primer símbolo está formado por una pareja de valores que son el número de coeficientes nulos que preceden al coeficiente que se está codificando y el número de bits necesarios para codificar el valor del propio coeficiente; y el segundo símbolo directamente es el valor del coeficiente. La codificación del primer símbolo se realiza mediante la asignación de un código Huffman, mientras que la del segundo es

directamente en complemento a 2. Por su parte, el coeficiente DC (primer coeficiente en la cadena) se codifica por separado y diferencialmente con el coeficiente DC del bloque anterior, codificándose únicamente la diferencia también mediante dos símbolos, aunque en este caso el primer símbolo sólo está formado por el número de bits necesario para codificar la diferencia y el segundo símbolo está formado por el valor de la diferencia. De igual forma que con los coeficientes AC, el primer símbolo se codifica mediante el uso de códigos Huffman y el segundo símbolo mediante complemento a 2.

La compresión que añade el codificador entrópico radica en la codificación Huffman, ya que asigna los códigos más cortos a los símbolos más frecuentes y los más largos a los menos frecuentes. La asignación de dichos códigos para el primer símbolo de los coeficientes AC, así como para el primer símbolo de los coeficientes DC, debe indicarse en el codificador mediante sus respectivas tablas de códigos Huffman. En el estándar JPEG se recomienda el uso de dos tablas (tablas 2.4 y 2.5), que son las que se utilizan en nuestra fase de codificación tanto para los vectores DCT como para los vectores KLT.

3.6.4.2 Codificación De La Señalización De La Transformada Óptima

El decodificador, para revertir el proceso del codificador y recuperar una reconstrucción de cada bloque, debe aplicar a cada uno la transformada inversa que le corresponda, por lo que es absolutamente necesario que se señalice al codificador qué transformada del conjunto de candidatas, se usó para transformar cada bloque.

Para señalizar la transformada que se ha usado con cada bloque, cada una de las transformadas del conjunto de candidatas se ha identificado con un número entero positivo. En nuestro caso se ha asignado el ‘0’ para la DCT y para las KLTs, dado que en ningún momento se utilizan más de 7, se ha asignado a cada una un entero entre ‘1’ y ‘7’.

La codificación del identificador (ID) de cada transformada se ha hecho directamente mediante la codificación binaria del entero correspondiente. Pero dado que por cada bloque es necesario codificar su correspondiente identificador y enviarlo al decodificador, esta información extra puede influir negativamente en la compresión final de la imagen. Por este motivo, se ha intentado minimizar el impacto de codificar la identificación de la transformada para lo cual se ha utilizado siempre el menor número de bits necesarios para poder identificar todas las transformadas del conjunto de candidatas. De esta forma, si únicamente se utiliza una KLT además de la DCT sólo se utiliza un bit para codificar el identificador de las transformadas; si por el contrario, el número de KLTs utilizado está entre dos y tres se utilizan 2 bits. Y finalmente, si el número de KLTs está entre 4 y 7 se utilizan 3 bits. En la tabla 3.17 se muestran los códigos binarios asignados a cada una de las transformadas.

		Nº DE KLTs USADAS		
		1 KLT (1 bit)	2-3 KLTs (2 bits)	4-7 KLTs (3 bits)
TRANSFORMADA	***ID***	***CODIFICACIÓN EN BINARIO DEL ID***		
DCT	'0'	0	00	000
KLT_1	'1'	1	01	001
KLT_2	'2'	----------	10	010
KLT_3	'3'	----------	11	011
KLT_4	'4'	----------	----------	100
KLT_5	'5'	----------	----------	101
KLT_6	'6'	----------	----------	110
KLT_7	'7'	----------	----------	111

Tabla 3.17 Códigos binarios de los índices de las transformadas.

Dado que el número de bits utilizado para codificar los identificadores de las transformadas varía dependiendo del número de KLTs usado, es necesario indicar el decodificador previa decodificación de los bloques, cuántas KLTs se han usado para codificar los bloques.

CAPÍTULO 4 - RESULTADOS DEL CODIFICADOR MULTITRANSFORMADA

4.1 INTRODUCCIÓN

En este punto se compara el funcionamiento del codificador adaptativo multitransformada de este proyecto frente a un codificador JPEG trabajando en su modo secuencial basado únicamente en la DCT. Se pretende evaluar la eficiencia de cada codificador de forma global. Para ello, se ha utilizado un conjunto de imágenes con diferentes características espaciales y se ha codificado dicho conjunto con cada codificador varias veces, modificando en cada ocasión el grado de calidad.

Como ya se ha descrito anteriormente, la eficiencia de un codificador de imágenes mediante compresión con pérdidas depende principalmente de dos factores: de lo comprimida que resulte la imagen codificada y de la distorsión que presenta la imagen decodificada con respecto a la imagen original. A mayor compresión con menor distorsión, la eficiencia del codificador aumenta.

Para medir la compresión que un codificador aplica a una imagen, se ha calculado el porcentaje que supone el tamaño en bits de la imagen comprimida con dicho codificador, con respecto al tamaño en bits de la imagen sin compresión (8 bits por píxel). Del mismo modo, para medir la compresión que un codificador aplica de forma global sobre un conjunto de imágenes, primero se ha calculado el tamaño total en bits que ocupan las imágenes comprimidas con dicho codificador, después se ha calculado el total de bits que ocupan las imágenes codificadas sin compresión y finalmente se ha calculado el porcentaje que supone el total de bits con compresión frente al total de bits sin compresión.

Para medir la distorsión que un codificador añade a una imagen, primero se codifica la imagen con dicho codificador y seguidamente se decodifica, pasando a calcular el error cuadrático medio, o MSE (ecuación 2.37), que presenta la imagen decodificada con respecto a la imagen original para finalmente obtener, a partir del MSE, el nivel de PSNR (ecuación 2.38) el cual es un indicador directo de la calidad que presenta la imagen decodificada. Del mismo modo, para medir la distorsión que un codificador añade de forma global a un conjunto de imágenes, primero se halla de forma independiente el MSE que presenta cada imagen decodificada y una vez obtenidos todos los MSEs, se calcula con ellos el MSE medio, a partir del cual finalmente se obtendrá el nivel de PSNR global del conjunto de imágenes.

La tabla de cuantificación especificada en ambos codificadores ha sido la tabla recomendada en el estándar JPEG (tabla 2.2: Luminancia). Del mismo modo, las tablas de códigos de longitud variable que se han especificado en los codificadores entrópicos de ambos codificadores, son las recomendadas en el estándar JPEG (Tablas 2.4 y 2.5).

4.2 CURVA DISTORSIÓN–COMPRESIÓN DE UN CODIFICADOR

Dado que para un codificador de imágenes, la distorsión que presentan las imágenes decodificadas y la compresión que presentan las imágenes codificadas están fuertemente relacionadas, normalmente ambos datos suelen presentarse de forma conjunta en lo que se denomina la curva distorsión-compresión. Para formar dicha curva, en el eje de abscisas (eje x) se representan los ratios de compresión que el codificador aplica a las imágenes y en el eje de ordenadas (eje y) se representan los niveles de PSNR que presentan las imágenes decodificadas a partir de dicho codificador. Por lo tanto, cada punto de la curva estará formado por un ratio de compresión determinado junto con el nivel de PSNR asociado a dicho ratio de compresión.

Para obtener los puntos de la curva distorsión-compresión, lo que se hace es variar el ratio de compresión que el codificador aplica a la imagen original, para seguidamente codificar la imagen con dicho ratio y decodificarla, calculando finalmente el nivel de PSNR que presenta la imagen decodificada para dicho ratio de compresión.

La forma de obtener las diferentes parejas de ratio de compresión y de nivel de PSNR de cada codificador se ha hecho mediante la modificación del Factor de Calidad de los codificadores (‘QF’ de sus siglas en inglés). Este factor puede tomar valores comprendidos entre 1 y 100, correspondiendo 1 con la peor calidad y 100 con la mejor calidad. En función del valor del QF, los escalones de la tabla de cuantificación especificada son modificados de forma que, cuando dicho factor es menor de 50 los escalones aumentan su tamaño proporcionalmente sin superar el valor de 255 y si es mayor de 50, los escalones disminuyen su tamaño igualmente de forma proporcional como máximo hasta 1. Así, cuando el valor del QF es igual a 100, todos los escalones de la tabla de cuantificación serán iguales a 1, por lo que los coeficientes cuantificados serán prácticamente iguales a los coeficientes sin cuantificar y por lo tanto, la imagen decodificada se aproximará mucho a la imagen original, pero consecuentemente la compresión se verá reducida. Por el contrario, si el valor del QF es igual a 1, todos los escalones de la tabla de cuantificación serán iguales a 255, por lo que tras la cuantificación, la gran mayoría de los coeficientes serán nulos y la distorsión que presente la imagen decodificada con respecto a la original será muy alta, pero la compresión se verá incrementada notablemente. Por último indicar que si el valor del QF es igual a 50, los escalones de la tabla de cuantificación no sufren alteración ninguna. Los valores de QF utilizados para obtener los puntos de las respectivas curvas distorsión-compresión de cada codificador han sido: 10, 20, 30,40, 50, 60, 70, 80 y 90.

En la figura 4.1 se muestra la curva distorsión-compresión del codificador JPEG para el conjunto de imágenes usado. En dicha figura, se puede observar que para ratios de compresión muy bajos (por debajo del 8 %) la calidad de la imagen decodificada se degrada mucho y los niveles de PSNR no superan los 28 dB. Por esto último y dado que en rara ocasión niveles de PSNR por debajo de 28 dB son aceptables, se ha evitado comparar los codificadores por debajo de estos niveles, por ello, finalmente los QF

utilizados en el codificador JPEG van desde 30 hasta 90 y en el caso del codificador multitransformada van desde 20 hasta 90.

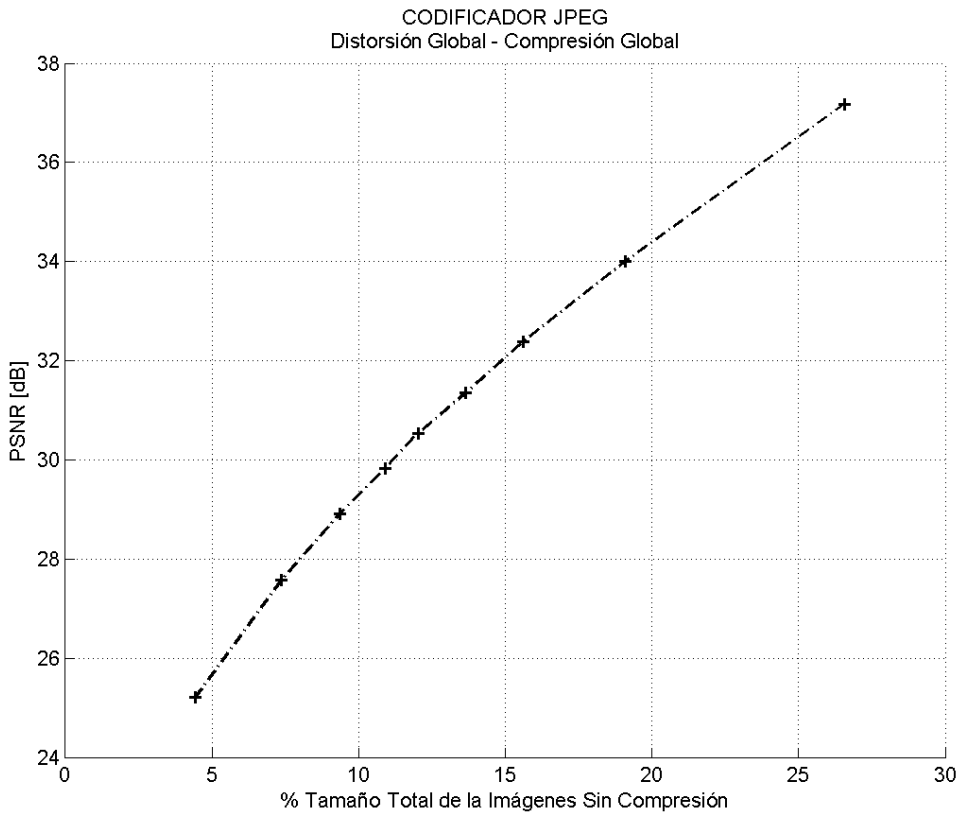


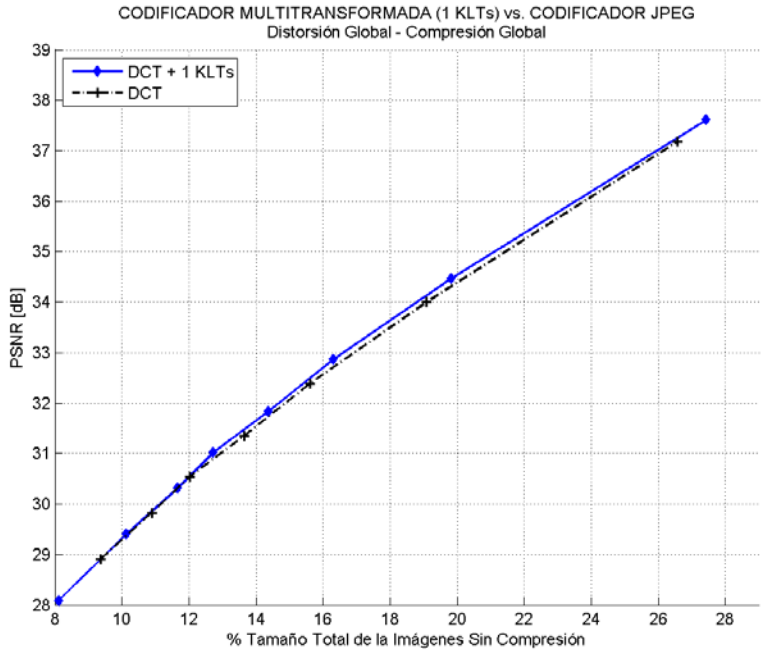
Figura 4.1 Curva distorsión-compresión del codificador JPEG para el conjunto de imágenes usado

4.3 RESULTADOS GLOBALES DEL CODIFICADOR MULTITRANSFORMADA FRENTE AL CODIFICADOR JPEG.

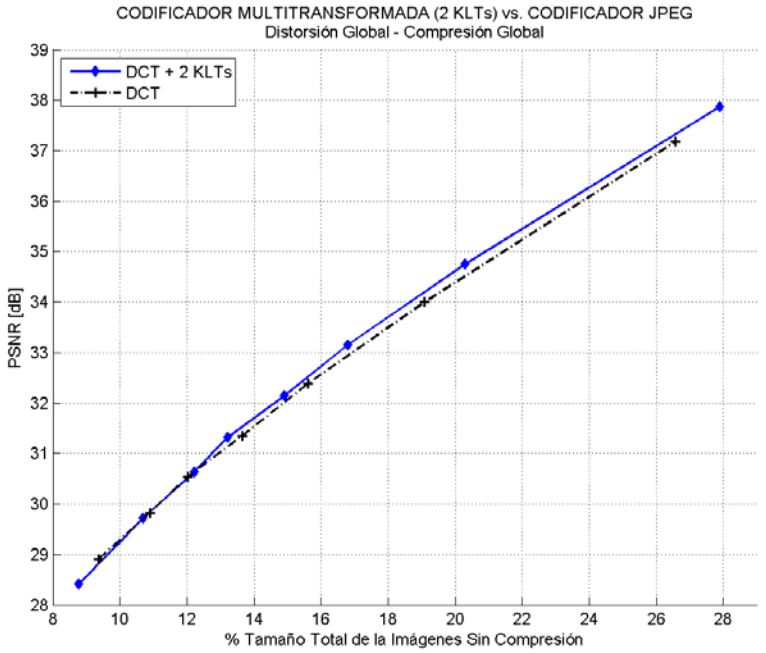
Para comparar ambos codificadores, en la Figura 4.2 se representan de forma conjunta las curvas de distorsión-compresión globales del codificador multitransformada y del codificador JPEG. Dado que el codificador multitransformada se puede configurar para trabajar con varias KLTs, en la misma figura se han representado las distintas curvas del codificador multitransformada al utilizar desde 1 hasta 7 KLTs.

Lo buscado y esperado, es que la curva del codificador multitransformada se encuentre desplazada hacia arriba y hacia a la izquierda con respecto de la curva del codificador JPEG. Que esté desplaza hacia arriba significaría que para un mismo ratio de compresión, el codificador multitransformada consigue mayores niveles de calidad y que esté desplaza hacia la izquierda significaría que, trabajando con el mismo nivel de calidad, el codificador multitransformada aplicaría mayor compresión a los datos que el codificador JPEG.

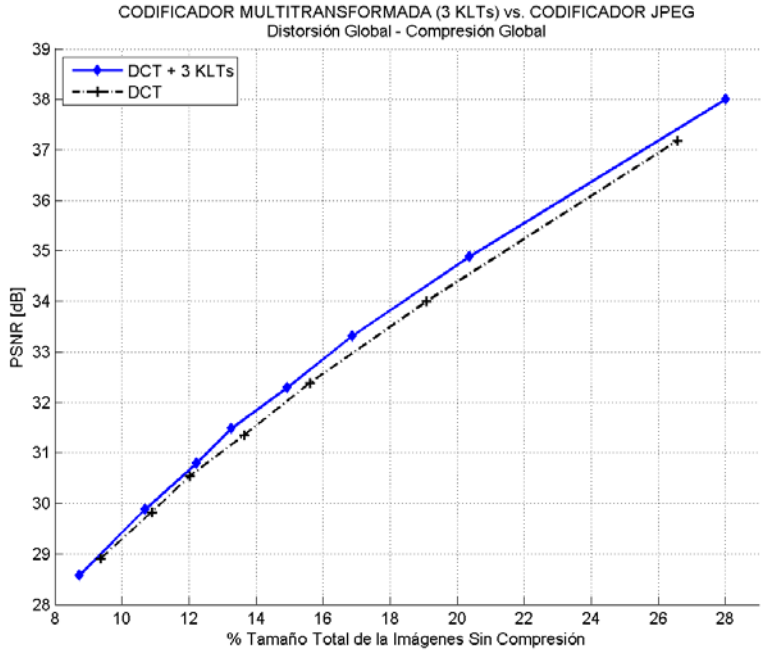
a)



b)



c)



d)

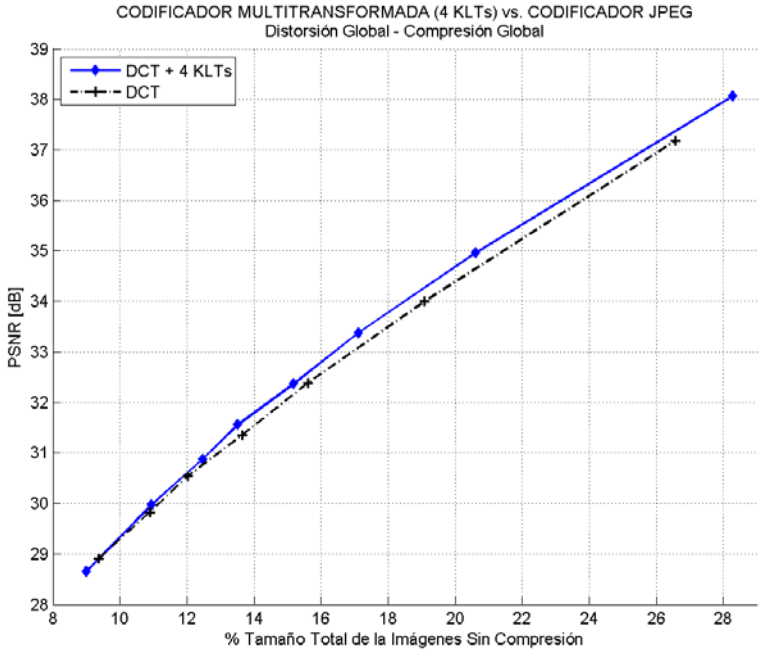
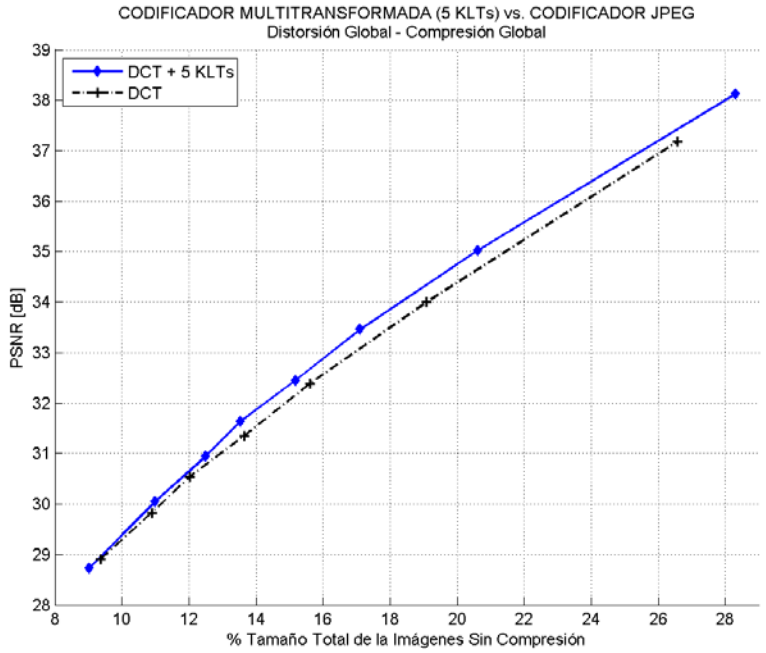
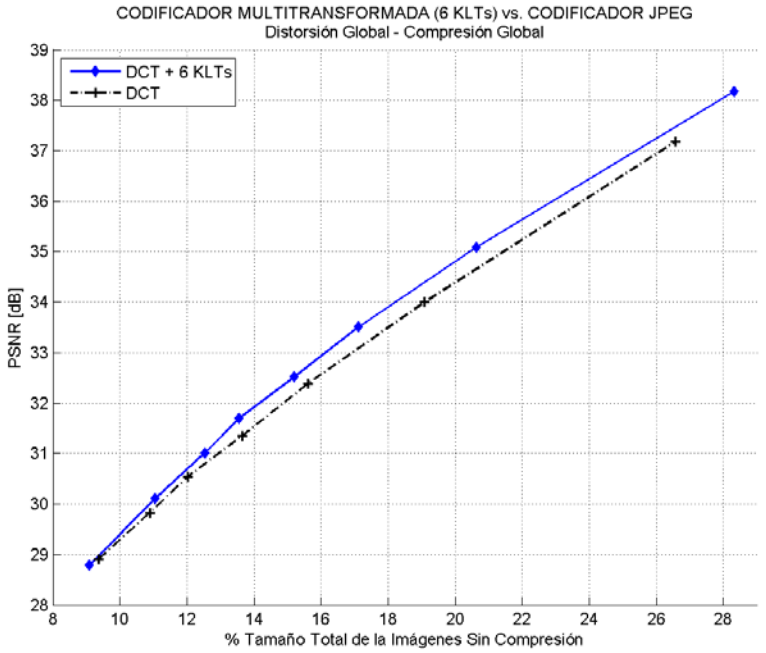


Figura 4.2 Curvas de distorsión-compresión de los codificadores multitransformada y JPEG (QF_{DCT} = 30:90 – QF_{KLT} = 20:90)

e)



f)



g)

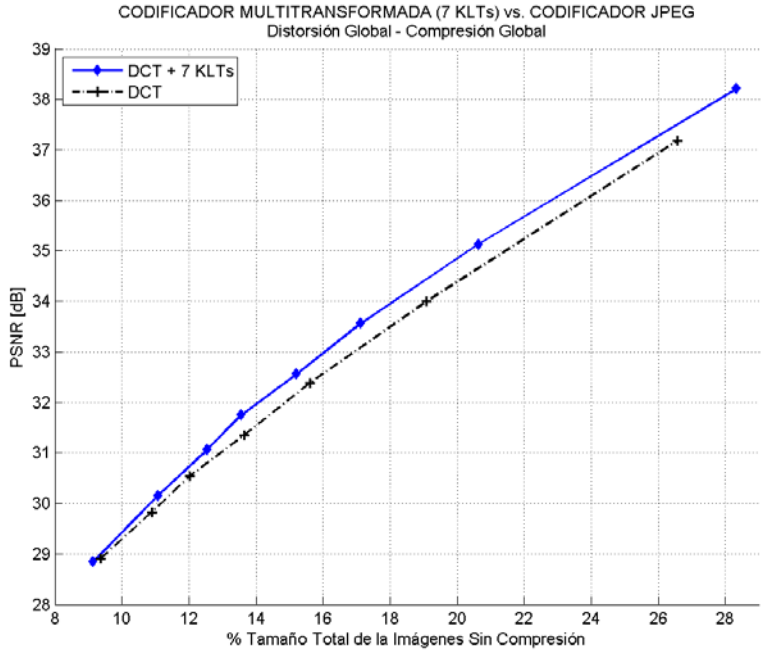


Figura 4.2 Curvas de distorsión-compresión de los codificadores multitransformada y JPEG (QF_{DCT} = 30:90 – QF_{KLT} = 20:90)

Cada curva mostrada en la figura 4.2, se ha obtenido mediante interpolación lineal a partir de los respectivos puntos obtenidos para los factores de calidad utilizados en cada codificador, por lo que dicha curva se ajusta a la realidad para los puntos asociados a los factores de calidad usados, pero para el resto de puntos la aproximación puede no ser tan buena. Para obtener unas curvas que se aproximen más al funcionamiento real de los codificadores en todo su rango de trabajo, cada una se ha ajustado mediante una función exponencial de la forma de la ecuación (3.10) utilizando la herramienta Curve Fitting de Matlab. Una vez obtenida la función de ajuste exponencial para cada curva, se han calculado los valores de PSNR correspondientes a los porcentajes comprendidos en el rango [9,27], representando finalmente todas las curvas ajustadas en la figura 4.4.

Inspeccionando la figura 4.4, se puede comprobar que la curva del codificador multitransformada prácticamente en todos los casos está por encima de la del codificador JPEG, aunque el incremento no es constante. Dicho incremento depende por un lado del número de KLTs usadas en el codificador y por otro, del ratio de compresión aplicado. Por lo tanto, se podría concluir que a primera vista, introducir matrices KLTs especializadas para las características espaciales predominantes en las imágenes, incrementa las prestaciones del codificador. Para analizar con más detalle cómo varía la ventaja del codificador multitransformada frente al codificador JPEG, para cada cantidad de KLTs, se ha calculado el incremento en el nivel de PSNR que se produce para los distintos ratios de compresión dentro del rango de trabajo y se han representado todas las curvas incremento en la figura 4.3.

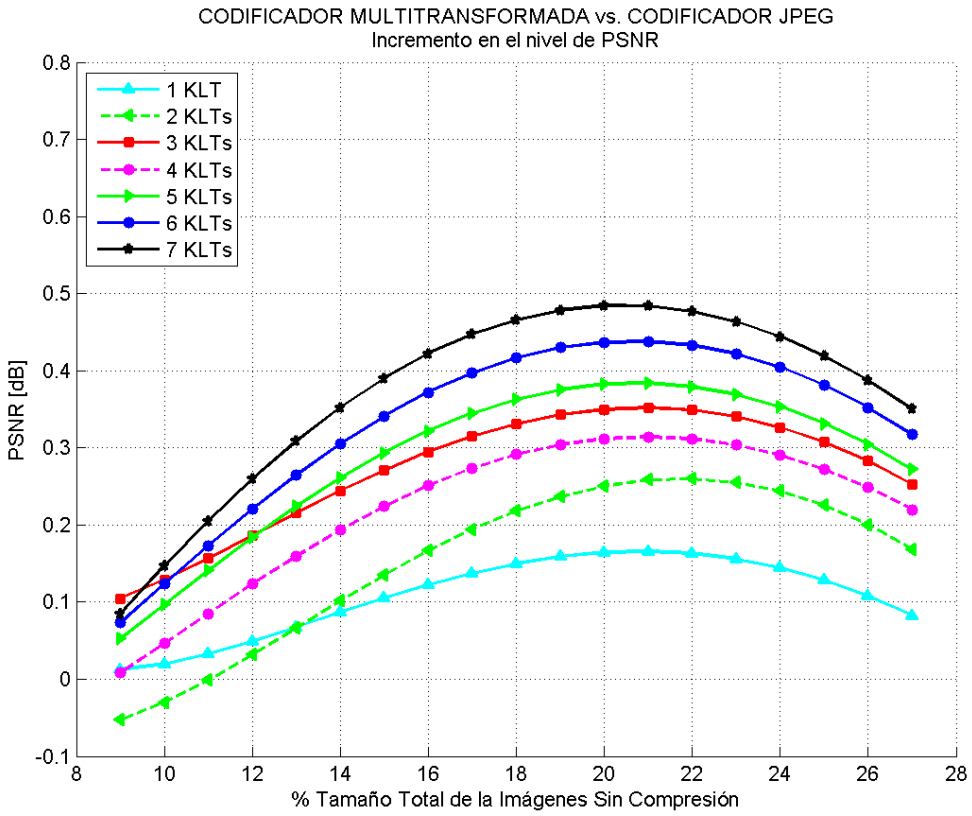
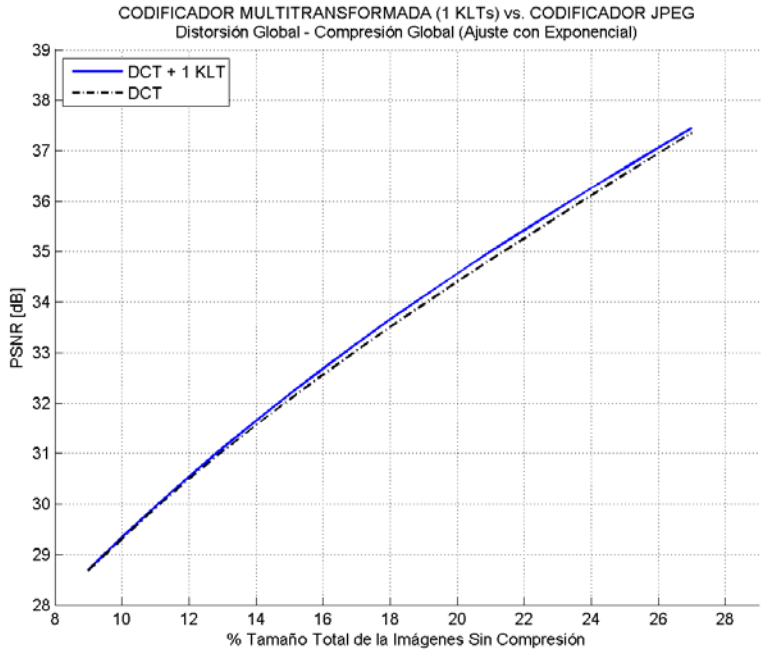
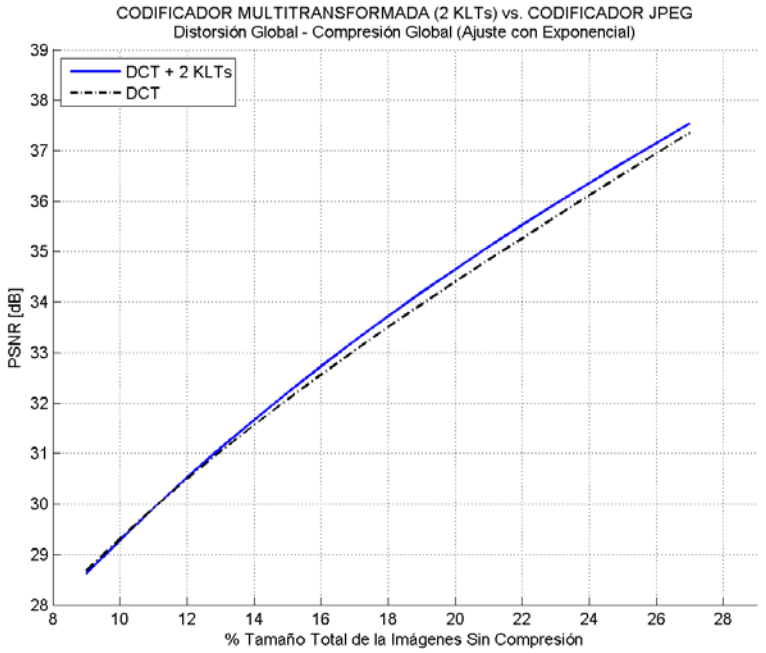


Figura 4.3 Incrementos en el nivel de PSNR del codificador multitransformada con respecto a JPEG.

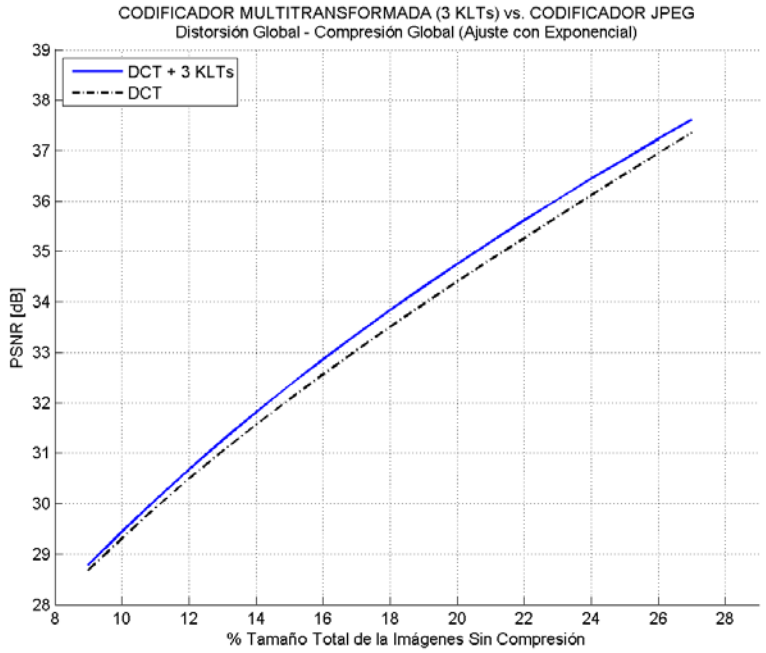
a)



b)



c)



d)

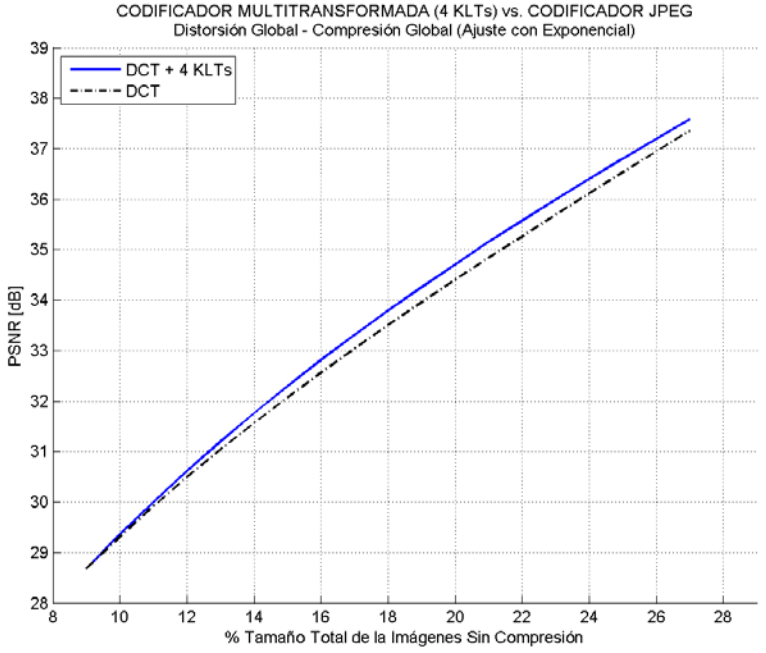


Figura 4.4 Curvas de distorsión-compresión de los codificadores multitransformada y JPEG ajustadas con función exponencial

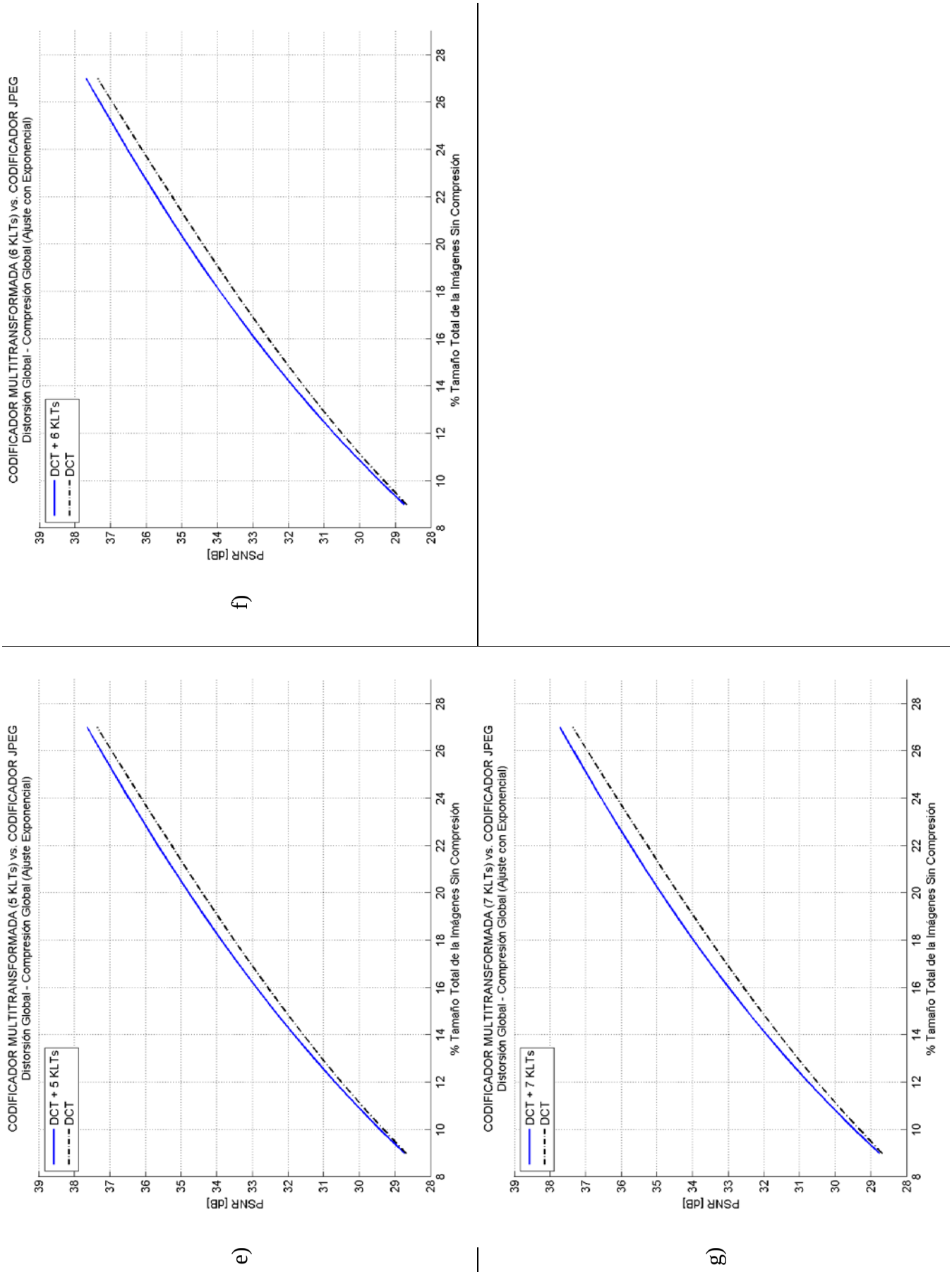


Figura 4.4 Curvas de distorsión-compresión de los codificadores multitransformada y JPEG ajustadas con función exponencial

Al comparar entre sí las curvas incremento de la figura 4.3, podemos verificar que su comportamiento es similar a lo largo del rango de trabajo en el que se mueve el ratio de compresión: para ratios de compresión muy altos (porcentajes menores de 15) se dan los menores incrementos de PSNR con respecto a JPEG, para ratios de compresión medios (porcentajes comprendidos entre 15 y 24) es donde se dan los mayores incrementos frente a JPEG y para ratios de compresión bajos (porcentajes mayores de 24) el incremento de PSNR con respecto a JPEG disminuye, aunque no tanto como para los ratios de compresión muy altos.

Por otro lado, según se ha podido demostrar anteriormente, agregar una KLT más en el codificador multitransformada va a suponer siempre una mejora en la especialización de las matrices KLT y por lo tanto, es de esperar que las prestaciones del codificador multitransformada se vean incrementadas de forma gradual según se va aumentando el número de KLTs; pero si observamos con detenimiento las curvas incremento de la figura 4.3, se puede comprobar que no está ocurriendo así, como por ejemplo al pasar de 1 a 2 KLTs o al pasar de 3 a 4 KLTs.

Esto último no concuerda con los resultados de PSNR obtenidos en las pruebas realizadas durante el diseño de las fases del codificador, ni con lo que teóricamente debería resultar ya que, para el caso de 1 KLT el codificador ni tan siquiera utiliza agrupamiento de bloques para hallar la matriz de transformación KLT, pero para el caso de 2 KLTs sí que agrupa los bloques según sus características espaciales, por lo que las matrices KLT del segundo caso están más especializadas que en el primero y deben provocar mejores resultados; sin embargo, el incremento en el nivel de PSNR para 2 KLTs es menor que para el caso de 1 KLT para porcentajes menores de 15 e incluso, para porcentajes menores de 11, el codificador multitransformada usando 2 KLTs junto con la DCT obtiene niveles de PSNR inferiores a los que obtiene el codificador JPEG únicamente con la DCT. Del mismo modo, al pasar de 3 a 4 KLTs en el codificador multitransformada, los resultados reales no concuerdan con los resultados teóricos ya que, en todo momento la curva incremento para 4 KLTs está por debajo de la curva para 3 KLTs, lo cual no debería ocurrir puesto que la especialización de las matrices de transformación en el caso de 4 KLTs es superior a las matrices en el caso de 3 KLTs.

El hecho de que el codificador multitransformada no se comporte como se espera al pasar de 1 a 2 KLTs y de 3 a 4 KLTs, es debido principalmente a la penalización en la compresión que provoca la codificación de la señalización de la transformada de cada bloque, lo cual se explica en el siguiente apartado.

4.4 EFECTO DE LA CODIFICACIÓN DEL ÍNDICE DE LA TRANSFORMADA SELECCIONADA PARA CADA BLOQUE.

El tamaño total en bits de una imagen codificada con JPEG es el resultado de sumar las longitudes de las cadenas binarias, que el codificador entrópico devuelve al codificar cada bloque de la imagen. Sin embargo, en el caso del codificador

multitransformada, además de la cadena binaria que el codificador entrópico devuelve tras codificar cada bloque, hay que sumar una cantidad extra de bits por bloque correspondiente a la codificación de la señalización de la transformada usada con dicho bloque. La cantidad de bits extra por bloque que se añade depende del número de KLTs que se esté usando (tabla 3.17): en el caso de usar una única KLT sólo se añade 1 bit por bloque; si se usan 2 o 3, la cantidad de bits por bloque que se suma es de 2 y si se usan entre 4 y 7 KLTs, la cantidad que se suma es de 3 bits por bloque.

En la tabla 4.1 se recoge el porcentaje con respecto al tamaño total de las imágenes sin comprimir, que supone dicha codificación para las distintas cantidades de KLTs usadas. Para calcular estos porcentajes, se ha sumado el total de bloques de todas las imágenes del conjunto utilizado y se ha calculado para cada número de KLTs, la cantidad total de bits que requiere la codificación de la señalización de la transformada teniendo en cuenta la tabla 3.17.

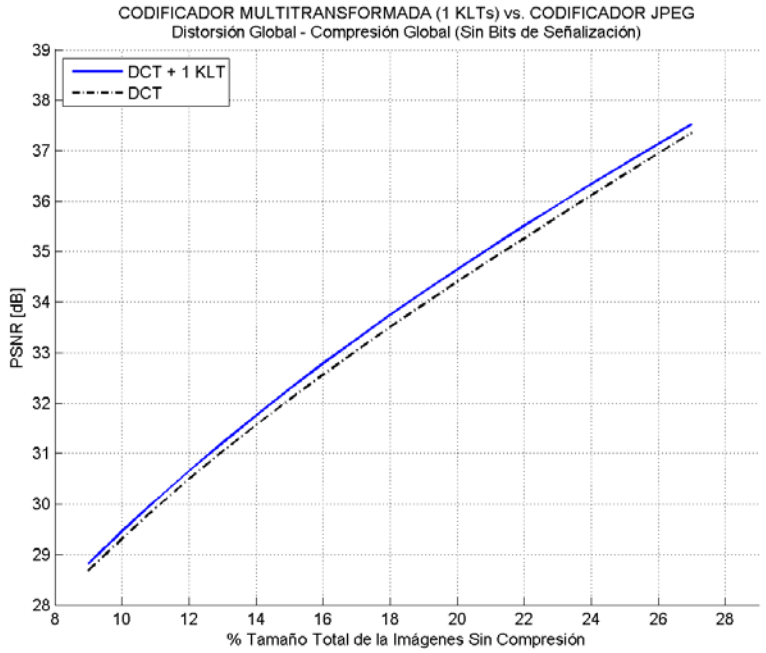
# de KLTs	Bits necesarios para codificar la transformada	% del tamaño total del conjunto de imágenes codificadas sin compresión
1 KLT	1 bit por bloque	≅ 0.2 %
2-3 KLTs	2 bits por bloque	≅ 0.4 %
4-7 KLTs	3 bits por bloque	≅ 0.6 %

Tabla 4.1 Porcentajes con respecto al tamaño total de las imágenes sin comprimir, que supone la codificación de los índices de las transformadas

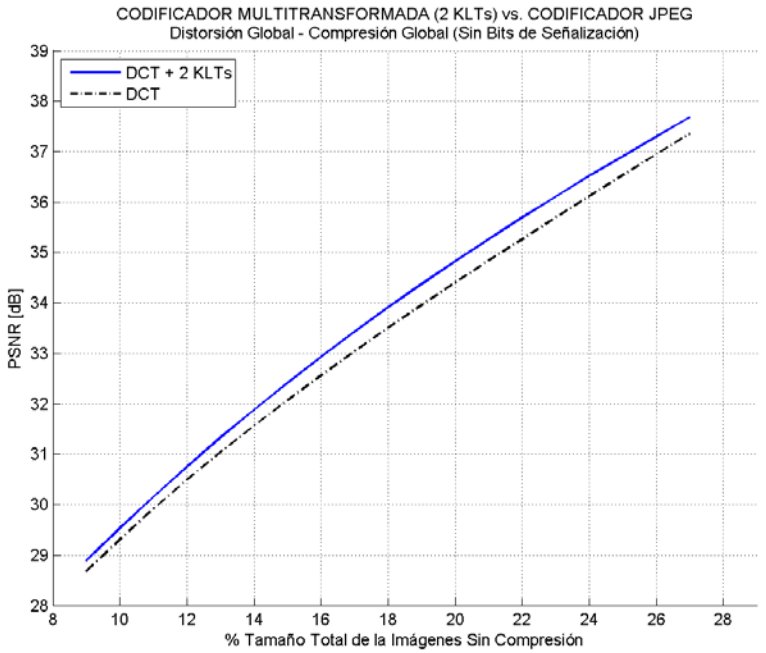
El efecto de la codificación de la señalización de la transformada es un desplazamiento hacia la derecha de la curva de distorsión-compresión del codificador multitransformada, cuya longitud dependerá del número de KLTs que esté usando el codificador (tabla 4.1).

Para comprobar visualmente el efecto de la codificación de la señalización de la transformada, se han calculado las mismas curvas de distorsión-compresión que en la figura 4.4, pero excluyendo esta vez del tamaño total en bits de las imágenes comprimidas, los bits correspondientes a la señalización de la transformada y se han representado en la figura 4.5. Del mismo modo, para este caso también se han calculado los incrementos de PSNR con respecto a JPEG para cada curva y se han representado en la figura 4.6.

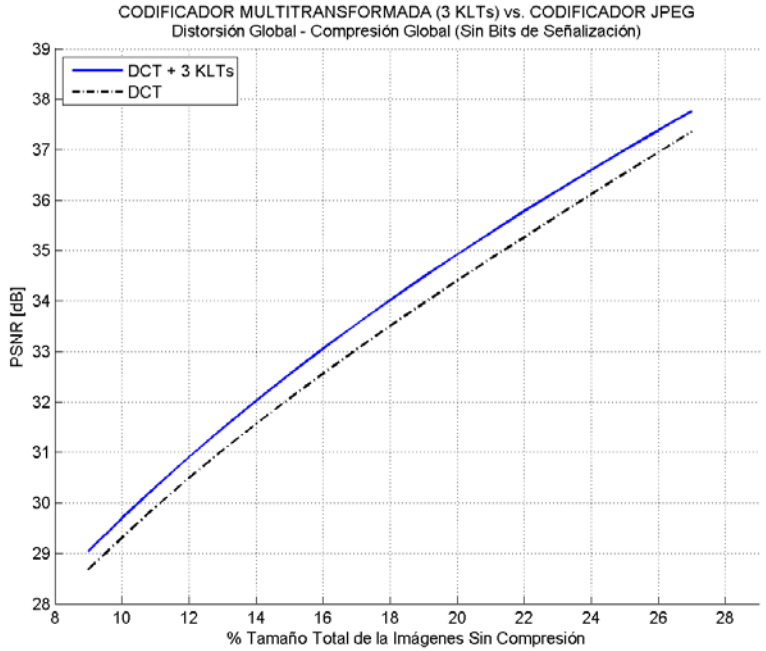
a)



b)



c)



d)

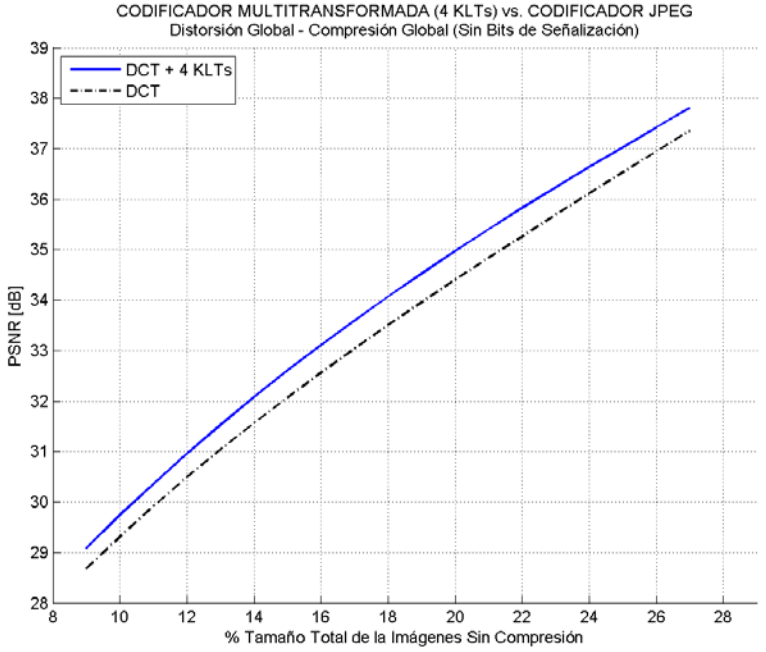
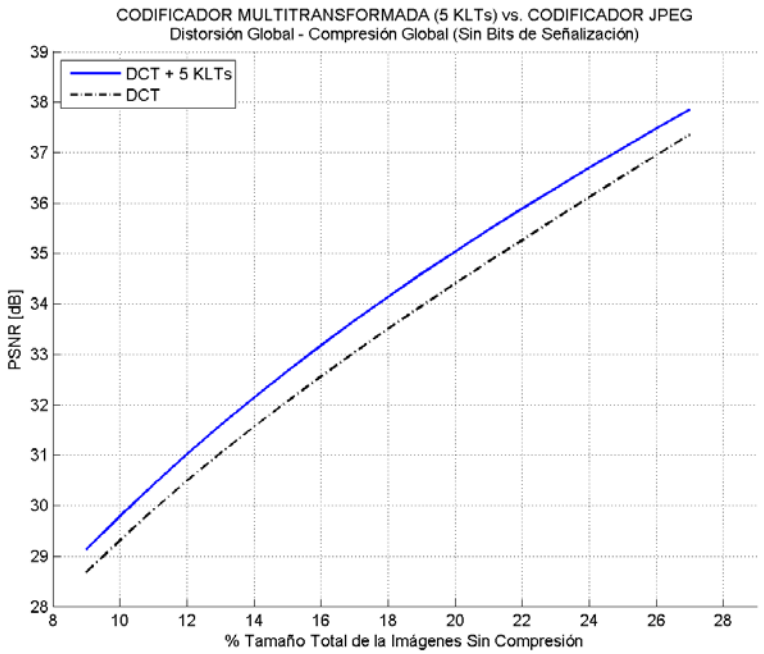
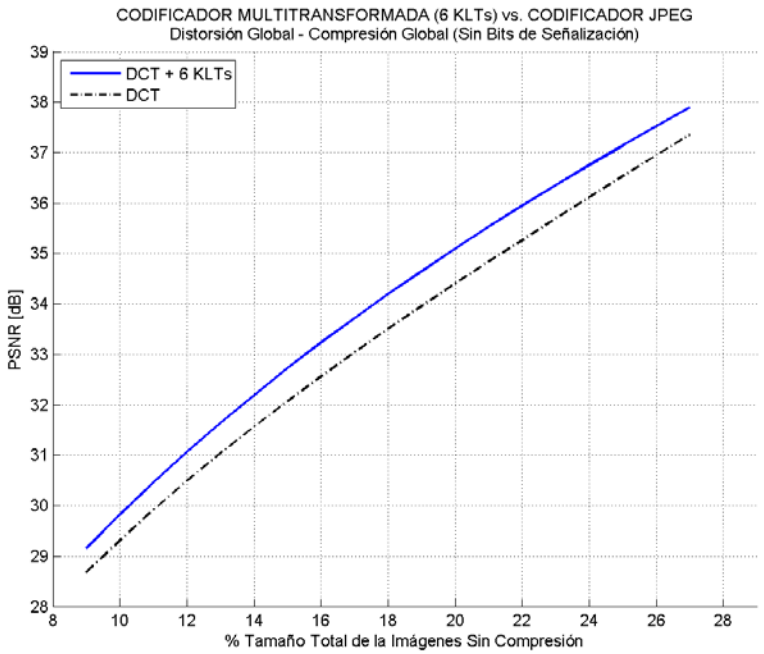


Figura 4.5 Curvas de distorsión-compresión de los codificadores multitransf. Y JPEG sin tener en cuenta la codificación de la señalización de la transformada (ajuste exp.).

e)



f)



g)

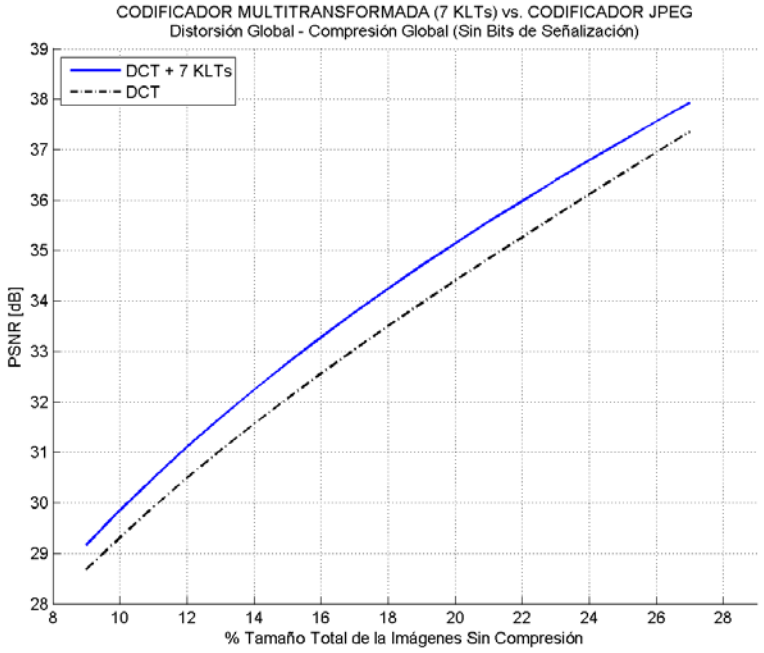


Figura 4.5 Curvas de distorsión-compresión de los codificadores multitransf. Y JPEG sin tener en cuenta la codificación de la señalización de la transformada (ajuste exp.)

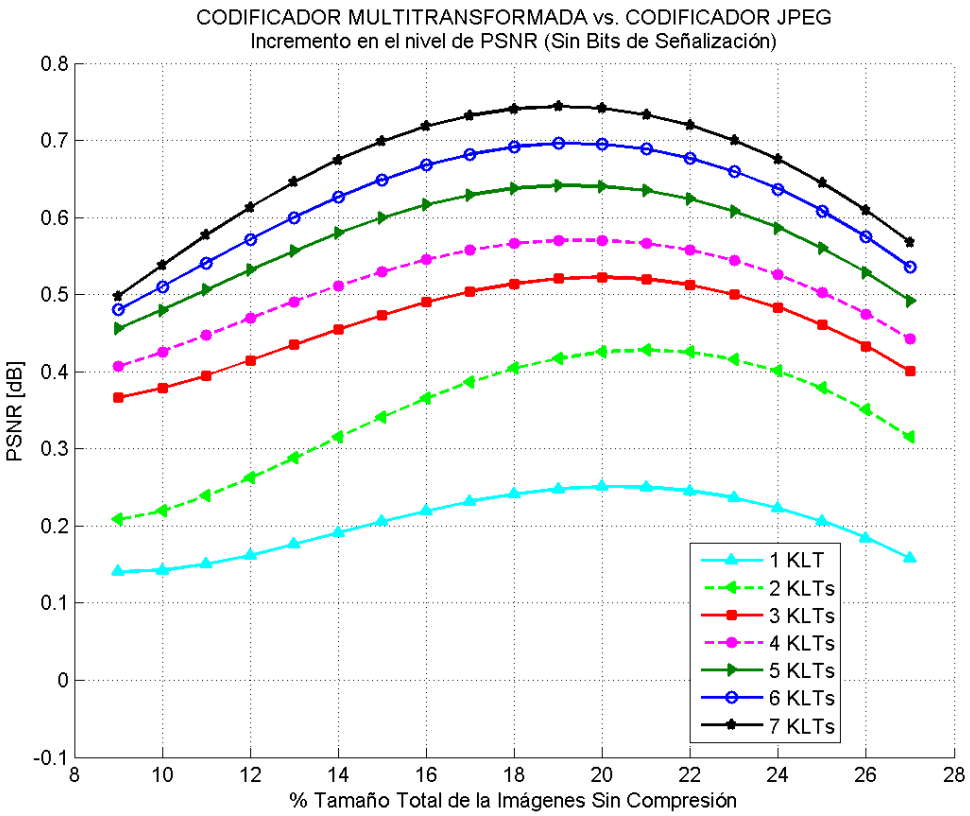


Figura 4.6 Incrementos en el nivel de PSNR del codificador multitransformada con respecto a JPEG sin tener en cuenta la codificación de la señalización de la transformada.

Comparando las curvas de distorsión-compresión de la figura 4.4 (con bits de señalización) con las curvas de la figura 4.5 (sin bits de señalización) se ve claramente la pérdida de prestaciones que supone el hecho de codificar el índice de la transformada de cada bloque.

Por otro lado, tal y como se puede comprobar en la figura 4.6, realmente al agregar una KLT nueva en el codificador multitransformada, siempre se produce un incremento en el nivel de PSNR, sin embargo, la codificación de la señalización de la transformada está provocando que la compresión se vea reducida y por lo tanto la curva de distorsión-compresión se desplace hacia la derecha, reduciéndose considerablemente el incremento en el nivel de PSNR. El desplazamiento hacia la derecha que provoca la codificación de los índices de las transformadas, dependerá del número de bits que sean necesarios para codificar todos los índices, pero dado que la codificación se hace directamente en binario, el hecho de agregar una transformada nueva en el codificador no va a suponer un nuevo desplazamiento hacia la derecha a no ser que sea necesario agregar un bit más para poder codificar todos los índices. De hecho, al utilizar desde 1 hasta 7 KLTs más la DCT, el desplazamiento sólo aumenta con respecto al caso anterior cuando el número de KLTs pasa de 1 a 2 y de 3 a 4, ya que en ambos casos es necesario agregar un bit más para poder codificar los índices de las transformadas.

Por todo lo anterior, al tener en cuenta los bits de la codificación de los índices de las transformadas, el incremento en el nivel de PSNR con respecto a JPEG se ve drásticamente reducido para los casos en los que se utilizan 2 y 4 KLTs. Para estos dos casos, la ventaja que supone sobre la especialización de las KLTs el hecho de agregar una transformada más, es inferior a la desventaja que supone utilizar un bit más para codificar los índices de las transformadas. Por ello, en el caso de 2 KLTs la curva de distorsión-compresión está por debajo de la curva de 1 KLT para porcentajes menores que 13, llegando incluso a estar por debajo de la curva del codificador JPEG para porcentajes menores que 11. Y del mismo modo, la curva correspondiente a 4 KLTs está siempre por debajo de la curva de 3 KLTs, no siendo rentable el hecho de pasar de 3 a 4 KLTs.

4.5 RESULTADOS DEL CODIFICADOR MULTITRANSFORMADA

4.5.1 IMÁGENES EN LAS QUE PREDOMINAN LAS REGIONES FORMADAS POR ALTAS FRECUENCIAS

El codificador JPEG obtiene los peores resultados en las regiones de las imágenes formadas por altas frecuencias, ya que la DCT no decorrela de forma eficiente los píxeles de dichas regiones. El motivo de ello, es que el núcleo de transformación de la DCT está formado por un conjunto fijo de funciones base y por ello no es capaz de adaptarse a las estadísticas que presentan los píxeles. Sin embargo, la KLT sí que es capaz de adaptarse a las estadísticas de los datos, ya que las funciones base de su núcleo se obtienen a partir de los propios datos a transformar y por lo tanto, produce una decorrelación eficiente de los datos.

Por todo lo anterior, en las imágenes en las que predominan las regiones con altas frecuencias, utilizar el codificador multitransformada debe producir una mejora considerable con respecto a JPEG, incluso configurándolo con un número reducido de KLTs. Además, si en una imagen el número de regiones con altas frecuencias es mayoritario y dichas regiones poseen diferentes estadísticas, el mejor resultado global se alcanzará cuando exista una KLT especializada para cada una de estas regiones diferentes por lo que, siempre que la cantidad de KLTs sea menor que el número de dichas regiones, agregar una KLT más normalmente se traducirá en una mejora de los resultados.

A continuación, se ha seleccionado dos imágenes del conjunto utilizado en las que predominan las regiones con altas frecuencias que son ‘baboon.bmp’ (Figura 4.7) y ‘kodim13.png’ (Figura 4.8). Seguidamente, en las figuras 4.9 y 4.10, se han representado respectivamente para cada imagen la curva de distorsión-compresión del codificador multitransformada para 3 KLTs junto con la curva del codificador JPEG, y también una gráfica que muestra la diferencia en el nivel de PSNR con respecto a JPEG al usar desde 1 hasta 7 KLTs en el codificador multitransformada.

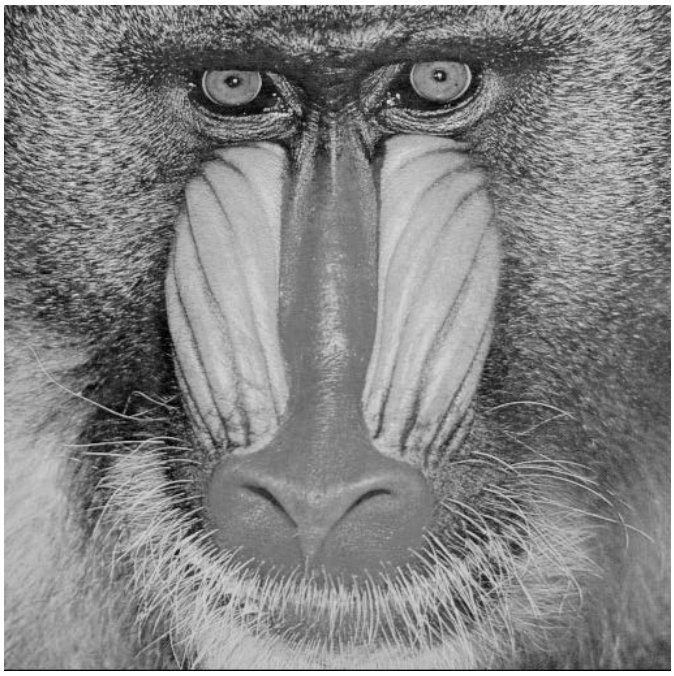


Figura 4.7 'baboon.bmp'.

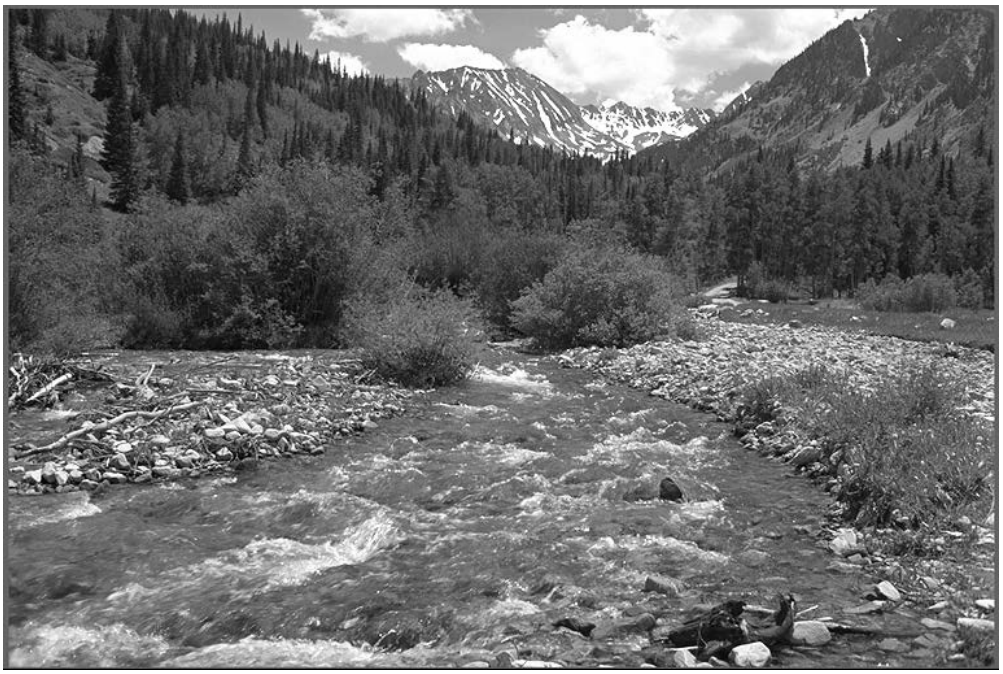
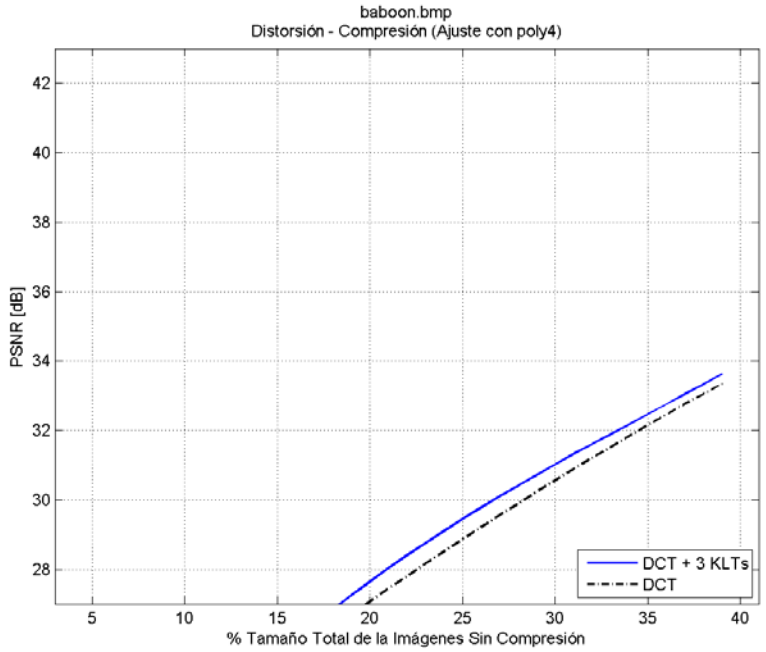


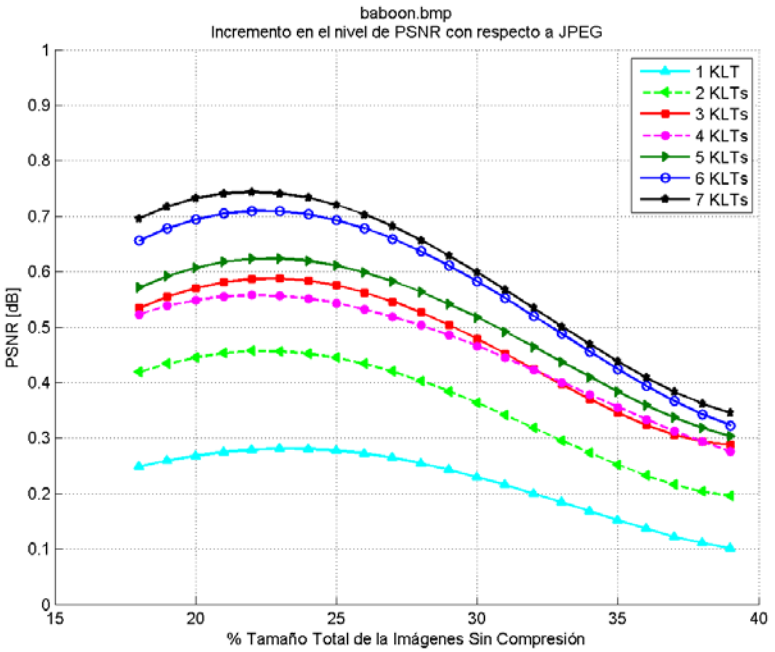
Figura 4.8 'kodim13.png'.

Con respecto a los resultados con 'baboon.bmp' (figura 4.9): En (a) se puede ver como la curva distorsión-compresión del codificador multitransformada con 3 KLTs siempre se sitúa por encima de la de JPEG, suponiendo una mejora notable el uso de tan sólo 3 KLTs. En (b) también se puede observar que según se va incrementando el número de KLTs, la mejora con respecto a JPEG aumenta, a excepción del caso de pasar de 3 a 4 KLTs, en el que se produce un retroceso. Como ya se ha indicado con los

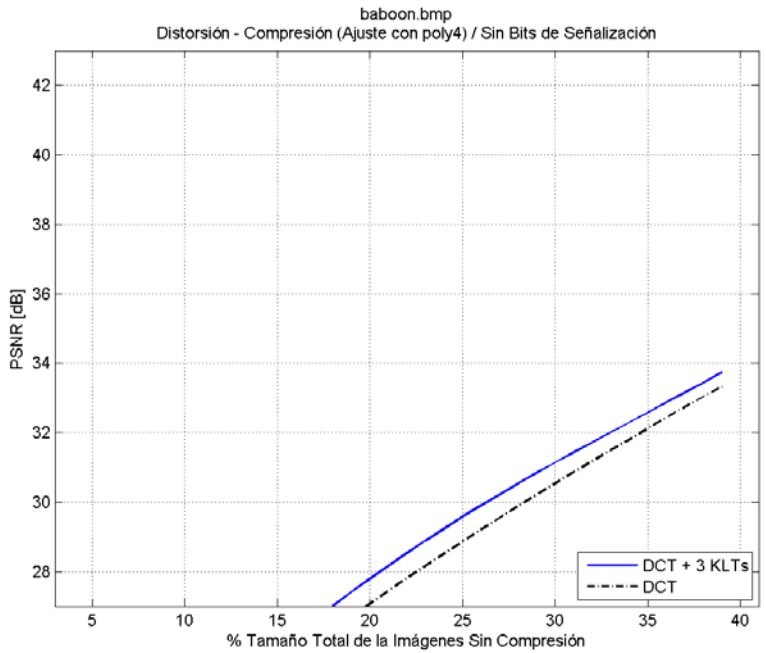
a)



b)



c)



d)

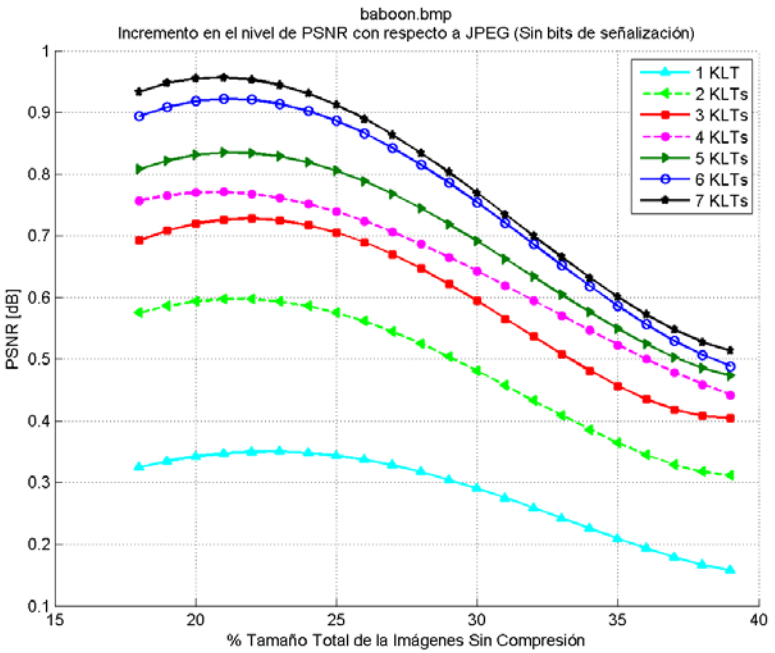
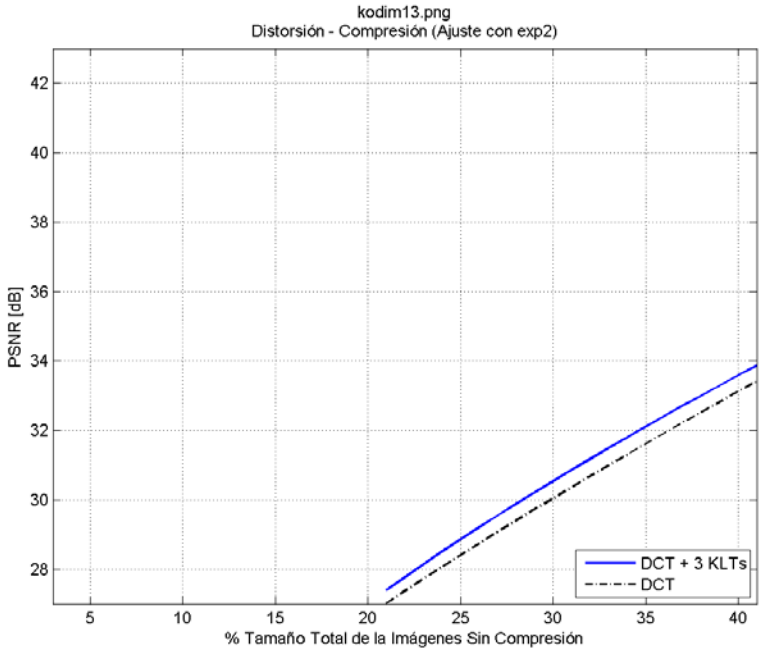
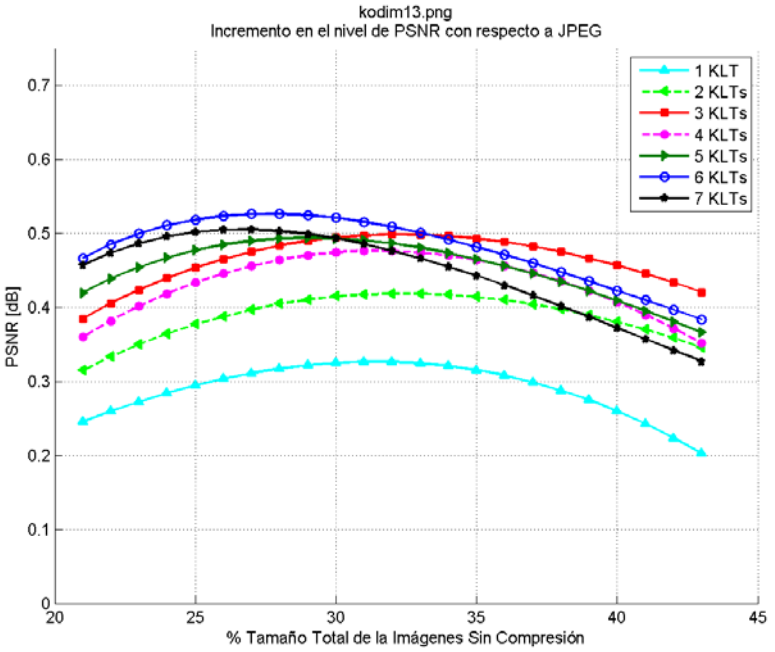


Figura 4.9 Curva Distorsión-Compresión e Incrementos de PSNR con respecto a JPEG para 'baboon.bmp', con bits de señalización (a, b) y sin bits de señalización (c, d).

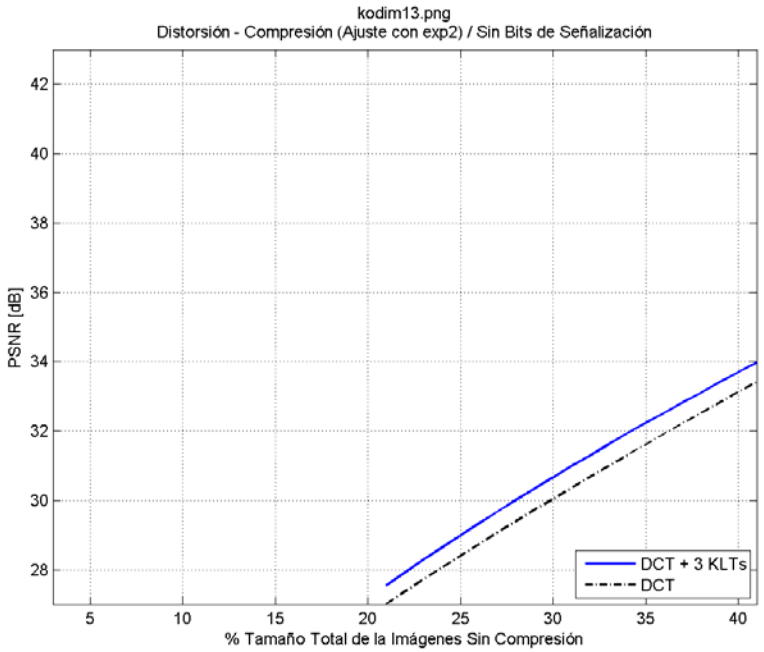
a)



b)



c)



d)

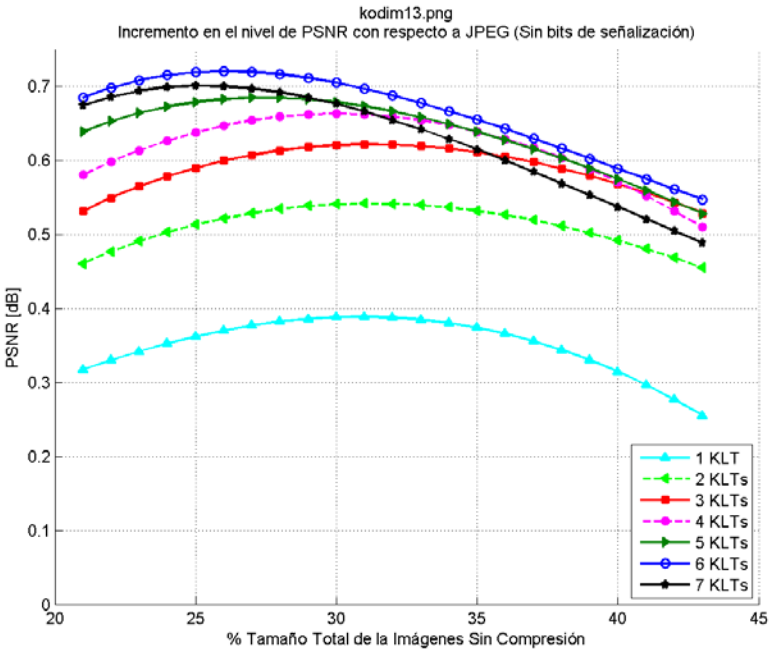


Figura 4.10 Curva Distorsión-Compresión e Incrementos de PSNR con respecto a JPEG para 'kodim13.png', con bits de señalización (a, b) y sin bits de señalización (c, d).

resultados globales, dicho empeoramiento es debido a la pérdida de compresión que provoca la codificación de la señalización de la transformada seleccionada para cada bloque, ya que al pasar de 3 a 4 KLTs, es necesario agregar un bit más por bloque para llevar a cabo dicha codificación. Por ello, en (d) se ha representado la misma gráfica que en (b) pero excluyendo del tamaño total de la imagen codificada, los bits correspondientes a la codificación de la señalización; en este caso sí que se puede comprobar que, además de obtener incrementos superiores en el nivel de PSNR con respecto a (b), ya no ocurre lo que pasaba al pasar de 3 a 4 KLTs y en todo momento, agregar una KLT más supone una mejora en los resultados.

Con respecto a 'kodim13.png' (figura 4.10): El comportamiento del codificador multitransformada en muy similar al de la imagen 'baboon.bmp', es decir, el hecho de agregar sólo 3 KLTs ya produce una mejora considerable con respecto a JPEG. Pero con esta imagen además de producirse un empeoramiento de los resultados al pasar de 3 a 4 KLTs como pasaba en 'baboon.bmp', también se produce al pasar de 6 a 7. El empeoramiento producido al pasar de 3 a 4 KLTs vuelve a ser debido a la codificación de la señalización, tal y como se puede verificar en (d), en donde se han excluido los bits de la señalización, pero sin embargo, al pasar de 6 a 7 KLTs, la codificación de la señalización no requiere agregar ningún bit más por lo que en este caso, no es la culpable; aquí, el causante del empeoramiento, es la disminución de la compresión provocada por la mejora de la especialización de las KLTs. Esta imagen está formada en su mayoría por altas frecuencias, por lo que pasar de 6 a 7 KLTs con mucha probabilidad mejorará la compactación de la energía de forma global en la imagen provocando que un mayor número de coeficientes resulten con mayor energía y por lo tanto tras la cuantificación, se reducirá el número de coeficientes nulos, así como los coeficientes no nulos resultarán con valores superiores, provocando que el codificador entrópico necesite más bits para codificarlos.

4.5.2 Imágenes En Las Que Predominan Las Regiones Formadas Por Bajas Frecuencias

Al contrario que en el caso anterior, en las regiones de las imágenes formadas por bajas frecuencias, la DCT produce una compactación eficiente de la energía, obteniendo JPEG sus mejores resultados con las imágenes en las que predominan dichas regiones.

Dado que el codificador multitransformada siempre utiliza los bloques de la imagen con mayor presencia de altas frecuencias para especializar las KLTs, en las imágenes en las que predominan las regiones con bajas frecuencias, también mejorará los resultados de JPEG, ya que se están especializando KLTs para aquellas regiones en las que JPEG ofrece peores prestaciones; Aun sí, esta mejora será más reducida que en el caso anterior, ya que para las imágenes en las que predominan las bajas frecuencias, la DCT compactará eficientemente la energía en la mayoría de los bloques, por lo tanto, la mejora dependerá de cuantas regiones con altas frecuencias presente la imagen.

Además, en estas imágenes agregar una nueva KLT puede que no siempre mejore los resultados, puesto que al contener un número reducido de regiones diferentes con altas frecuencias, es posible especializar una KLT para cada una de estas regiones y en ese caso, agregar una KLT más no supondrá una mejora considerable.

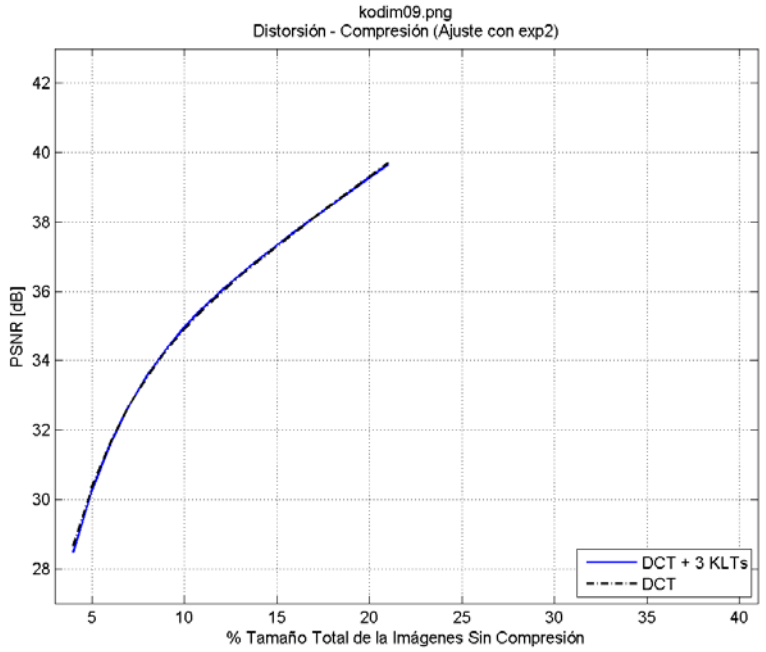
A continuación, se ha seleccionado otras dos imágenes del conjunto utilizado en las que predominan las regiones con bajas frecuencias que son 'kodim09.png' (Figura 4.11) y 'kodim20.png' (Figura 4.12) y seguidamente, en las figuras 4.13 y 4.14, se han representado respectivamente para cada imagen, la curva de distorsión-compresión del codificador multitransformada para 3 KLTs junto con la curva del codificador JPEG, y también una gráfica que muestra la diferencia en el nivel de PSNR con respecto a JPEG al usar desde 1 hasta 7 KLTs en el codificador multitransformada.

Figura 4.11 'kodim09.png'.

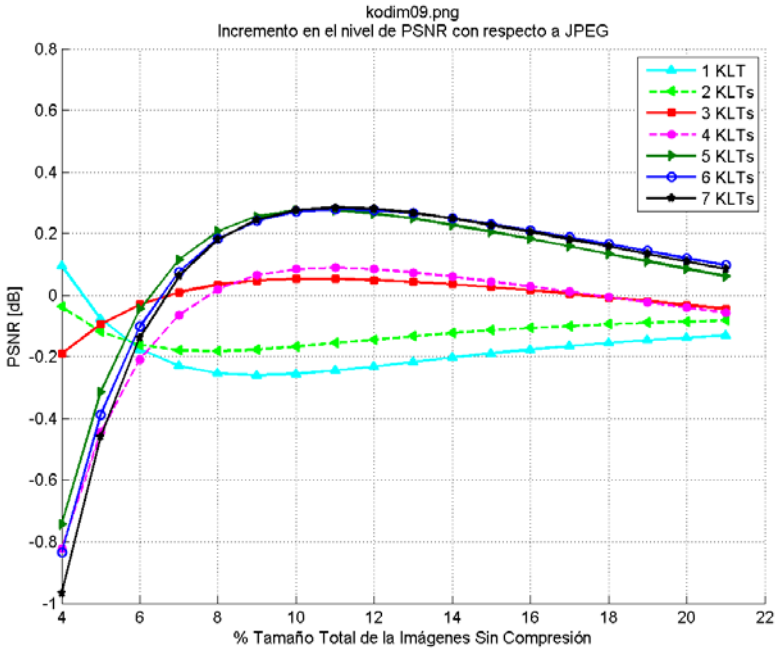
Figura 4.12 'kodim20.png'.

Con respecto a los resultados con 'kodim09.png' (figura 4.13): En (a) se puede ver como la curva distorsión-compresión del codificador multitransformada con 3 KLTs prácticamente se solapa con la de JPEG, no suponiendo ninguna mejora. Si observamos en (b) como varía la diferencia en el nivel de PSNR con respecto a JPEG al utilizar de 1 a 7 KLTs en el codificador multitransformada, cuando se usan 1 o 2 KLTs los resultados empeoran, utilizar 3 y 4 no supone ninguna mejora con respecto a JPEG y utilizar 5, 6 o 7 suponen una mejora, aunque bastante contenida para esas cantidades de KLTs. Sin embargo, analizando la imagen, se puede comprobar que aunque son predominantes las regiones con bajas frecuencias como el cielo y las velas de las embarcaciones, también existen regiones de altas frecuencias como la del agua, por lo que utilizar KLTs para estas últimas regiones debería suponer una mejora con respecto a usar sólo la DCT. Por ello, también se han plasmado en (c) y (d) los mismos resultados que en (a) y en (b), pero excluyendo del tamaño final de la imagen codificada los bits correspondientes a la codificación de la señalización de la transformada de cada bloque. Ahora en c) la curva del codificador multitransformada con 3 KLTs sí que se encuentra por encima de la del codificador JPEG, aunque el incremento es más reducido que para las imágenes en las que predominan las regiones con altas frecuencias. Y en d) también podemos comprobar que agregar KLTs mejora sustancialmente los resultados de JPEG, excepto para los casos de 1 y 2 KLTs, en los que probablemente se esté dando una pérdida de compresión con respecto a JPEG y para estos dos casos el codificador multitransformada prácticamente no suponga mejora alguna. Sin embargo, al utilizar más KLTs sí que mejora los resultados. Señalar que en esta imagen, usar 6 o 7 KLTs en el codificador multitransformada no supone ninguna mejora con respecto a usar 5, por lo que 5 KLTs serían suficientes para que exista una KLT especializada para cada una de las regiones diferentes de la imagen formadas por altas frecuencias.

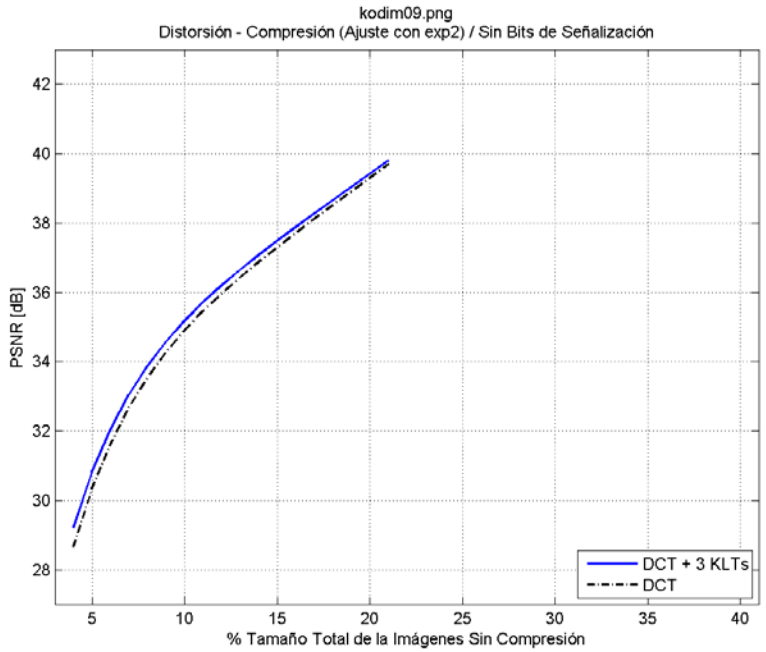
a)



b)



c)



d)

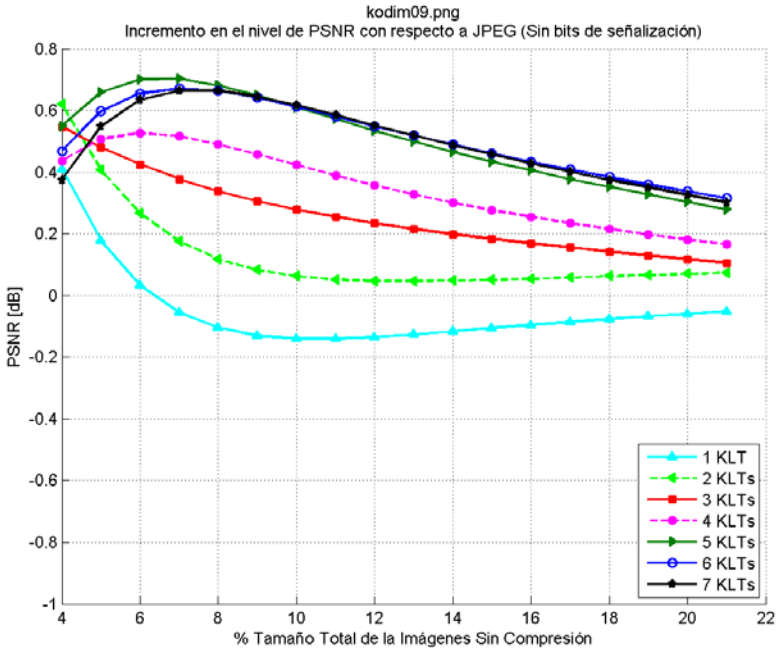
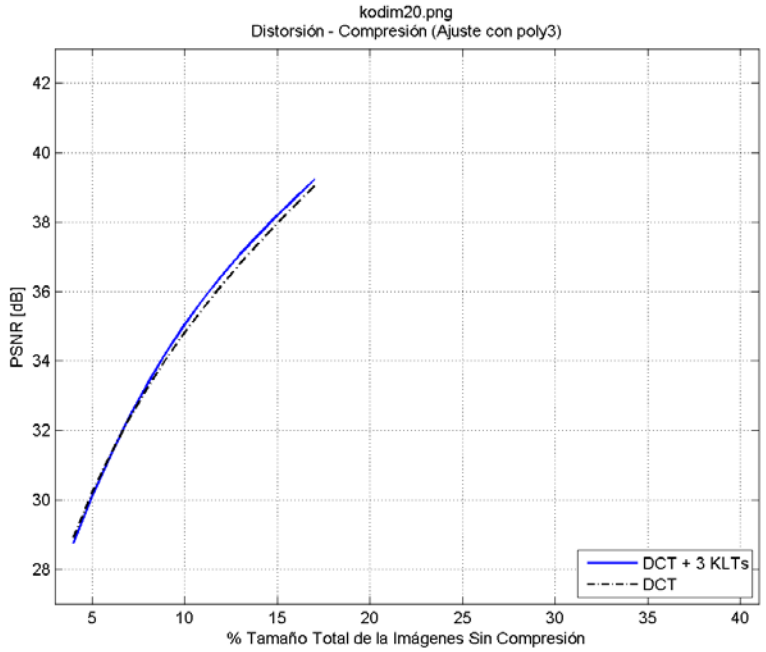
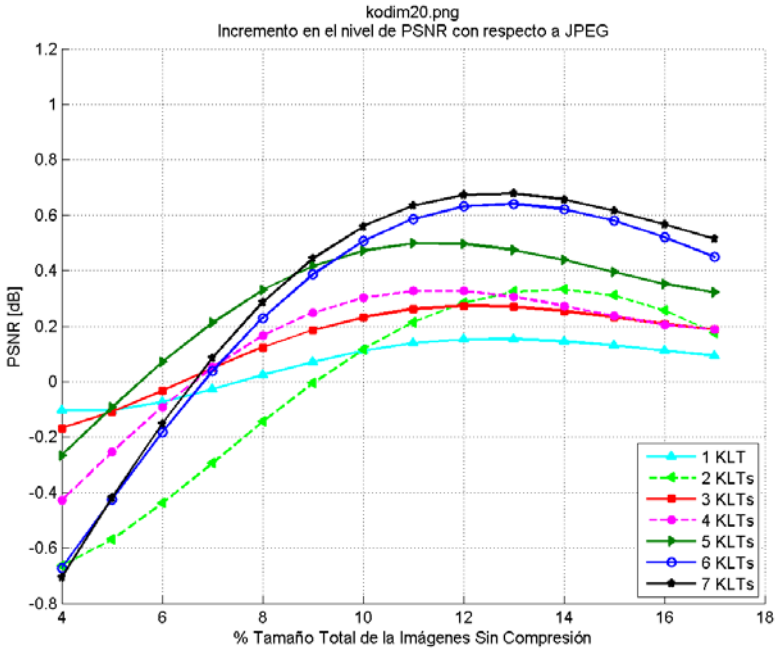


Figura 4.13 Curva Distorsión-Compresión e Incrementos de PSNR con respecto a JPEG para 'kodim09.png', con bits de señalización (a, b) y sin bits de señalización (c, d).

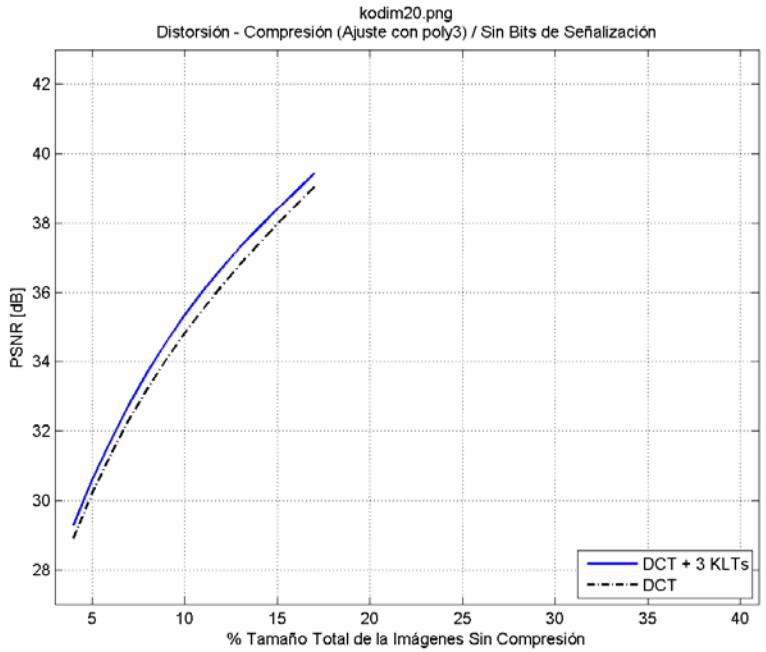
a)



b)



c)



d)

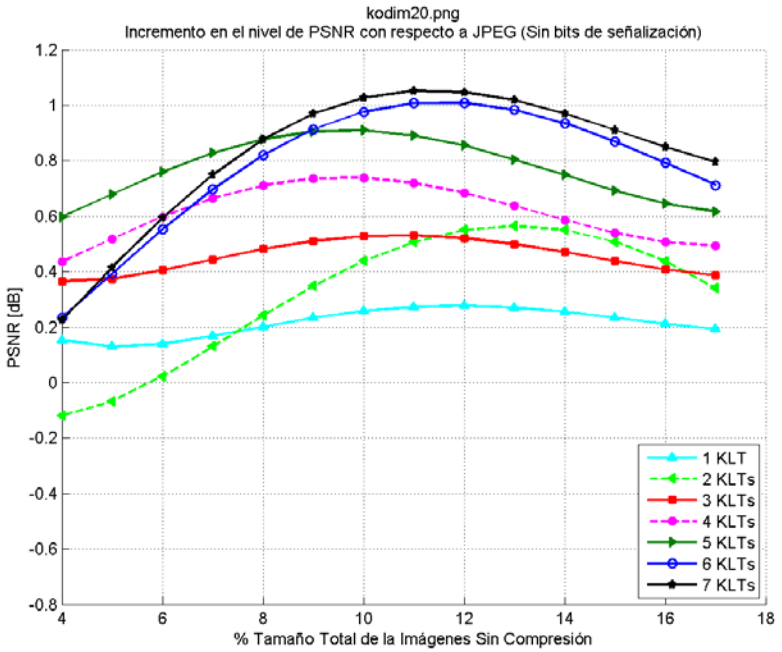


Figura 4.14 Curva Distorsión-Compresión e Incrementos de PSNR con respecto a JPEG para 'kodim20.png', con bits de señalización (a, b) y sin bits de señalización (c, d).

Con respecto a los resultados con 'kodim20.png' (figura 4.14): En (a) podemos comprobar que para esta imagen el codificador multitransformada con 3 KLTs sí que obtiene cierta ventaja frente a JPEG, aunque bastante reducida. Además, en (b) se puede comprobar que para porcentajes entre 4 y 7, el codificador multitransformada obtiene niveles de PSNR inferiores al codificador JPEG. Dada la considerable pérdida de compresión que han demostrado provocar los bits de la señalización de la transformada, en (c) y (d) se han vuelto a plasmar los resultados de esta imagen pero sin tener en cuenta dichos bits, pudiendo comprobar que la pérdida de compresión que suponen aquí, está provocando que los niveles de PSNR se vean reducidos considerablemente con respecto a JPEG ya que al excluirlos, dichos niveles se ven incrementados notablemente despareciendo casi por completo los valores negativos en (d).

4.5.3 Mejores Resultados Del Codificador Multitransformada

En este apartado, no se busca analizar las prestaciones del codificador multitransformada para un tipo de imágenes en concreto como en los dos apartados anteriores; ahora lo que se ha hecho es inspeccionar los resultados de las imágenes del conjunto utilizado, en busca de aquellas en las que el codificador multitransformada obtiene los mejores resultados frente a JPEG, para comprobar si dichas imágenes poseen características similares. Dos de las imágenes en las que el codificador multitransformada obtiene mayor ventaja frente a JPEG son 'barbara.bmp' y 'kodim05.png' (Figuras 4.15 y 4.16 respectivamente).

Figura 4.15 'barbara.bmp'.

Figura 4.16 'kodim05.png'.

Con respecto a barbara.bmp, en la figura 4.17 (a) y (b) se han mostrado gráficamente sus resultados: en (a) se ha representado la curva distorsión-compresión del codificador multitransformada para 7 KLTs junto con la curva de JPEG y en (b) se ha representado los incrementos en el nivel de PSNR con respecto a JPEG al utilizar de 1 a 7 KLTs en el codificador multitransformada. En (a) se ha representado la curva distorsión- compresión para 7 KLTs ya que con dicha cantidad se ha obtenido el mayor incremento en los niveles PSNR frente a JPEG. De hecho en (b), se puede comprobar que para este número de KLTs, el incremento de PSNR con respecto a JPEG es superior a 1'6 dB durante un tramo, lo cual es un incremento muy notable, más todavía si tenemos en cuenta que para obtener los resultados de compresión, no se han excluido los bits correspondientes a la codificación de la señalización de las transformadas. Aparte, para todas las cantidades de KLTs usadas, el incremento frente a JPEG es muy significativo y en general, agregar una nueva KLT en el codificador multitransformada provoca una mejora de sus prestaciones, sobre todo para porcentajes mayores de 15.

El buen funcionamiento del codificador multitransformada sobre 'barbara.bmp' es debido a que en la imagen predominan las regiones formadas por altas frecuencias, pero en las que sus píxeles se encuentran altamente correlados entre sí, es decir, estas regiones no poseen texturas caóticas. Por ejemplo, la región de los pantalones y del pañuelo que lleva en la cabeza está formada principalmente por altas frecuencias, pero tal y como se puede comprobar la evolución de los píxeles vecinos es muy similar existiendo una alta correlación entre sus píxeles. De igual forma ocurre con la textura del mantel, el sillón y los libros colocados en la estantería. Para estas regiones, la KLT es capaz de obtener unas imágenes base que decorrelan sus píxeles de forma más eficiente que la DCT, obteniendo de ahí su ventaja.

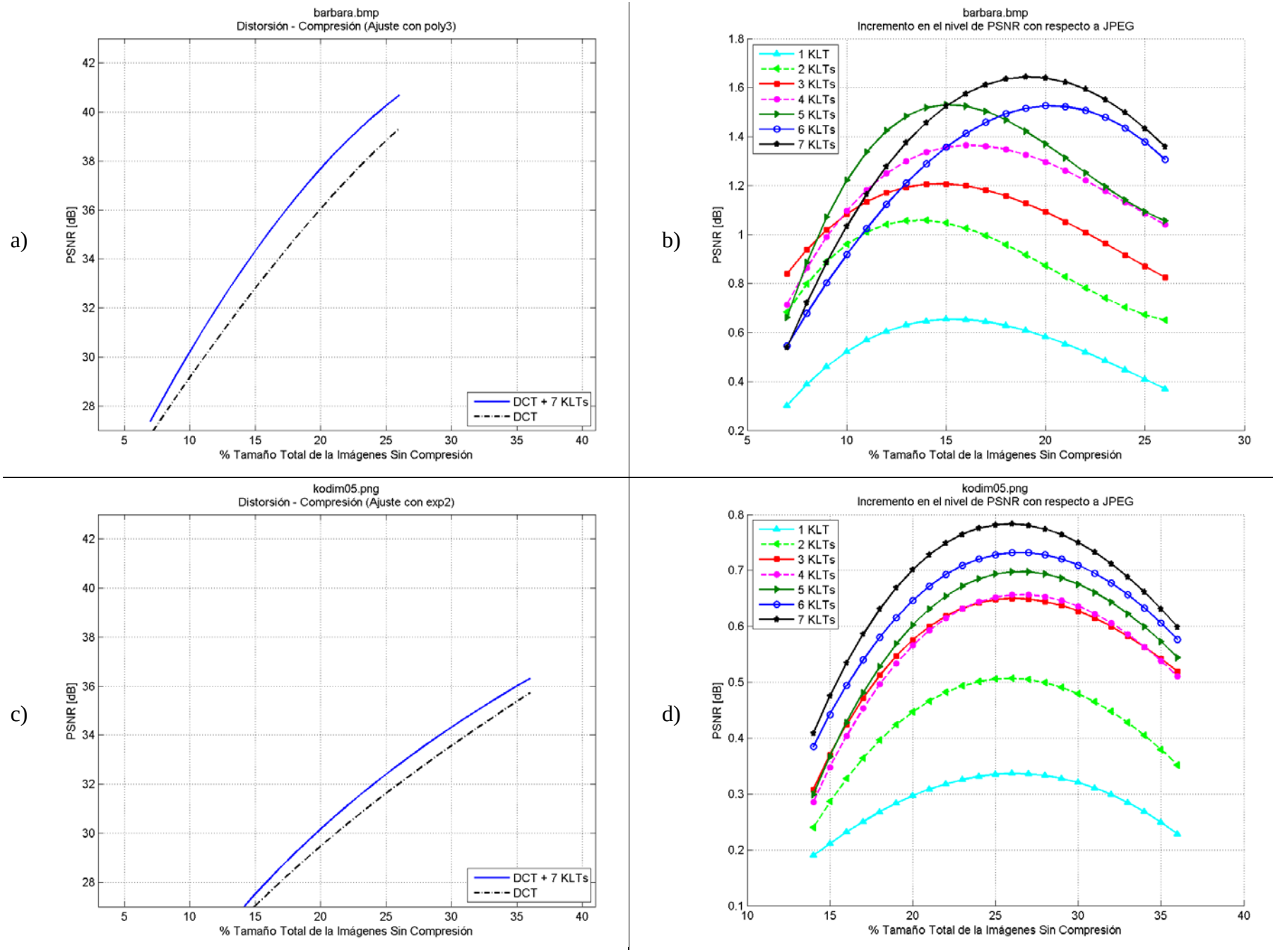


Figura 4.17 Curva Distorsión-Compresión e Incrementos de PSNR con respecto a JPEG para 'barbara.bmp' (a, b) y para 'kodim05.png' (c, d).

De la misma forma, para 'kodim05.png' en la figura 4.17 © y (d) se han representado las mismas gráficas que para 'barbara.bmp'. Para esta imagen la máxima ventaja del codificador multitransformada frente a JPEG también se da para 7 KLTs, por lo que en © se ha representado la curva distorsión-compresión para esta cantidad. En esta imagen, la ventaja no es tan grande como en el caso anterior, aunque siguen produciéndose incrementos de PSNR muy significativos con respecto a JPEG, tal y como se puede comprobar en (d) cuyos resultados, al igual que en 'barbara.bmp', son los obtenidos sin excluir del tamaño final de la imagen codificada los bits de la codificación de la señalización. Por último resaltar que, para esta imagen agregar una KLT más en el codificador multitransformada siempre mejora los resultados, a excepción del caso de pasar de 3 a 4 KLTs en los que los resultados varían muy poco, debido a la pérdida de compresión que suponen los bits de la codificación de la señalización.

En 'kodim05.png', al igual que en 'barbara.bmp', los buenos resultados del codificador multitransformada son debidos a la existencia de regiones formadas por altas frecuencias, en las que existe una alta correlación entre sus píxeles como por ejemplo, las ruedas de las motos y las cubiertas de los amortiguadores. Además, también hay regiones formadas por altas frecuencias como la tierra y la ladera de fondo en las que la correlación no es tan alta, pero en las que la KLT sigue provocando mejores resultados que la DCT gracias a su adaptación nativa a las estadísticas de los datos.

4.5.4 Peores Resultados Del Codificador Multitransformada

Finalmente, se han vuelto a inspeccionar los resultados de las imágenes del conjunto utilizado, pero ahora con el objetivo de identificar aquellas en las que el codificador multitransformada obtiene los peores resultados frente a JPEG. Dos de estas imágenes han sido 'kodim02.png' y 'kodim21.png' (Figuras 4.18 y 4.19 respectivamente).

Con respecto a 'kodim02.png', en la figura 4.20, en (a) se ha representado la curva de distorsión-compresión del codificador multitransformada para 4 KLTs junto con la curva de JPEG y en (b) los incrementos en el nivel de PSNR con respecto a JPEG del codificador multitransformada al utilizar desde 1 hasta 7 KLTs.

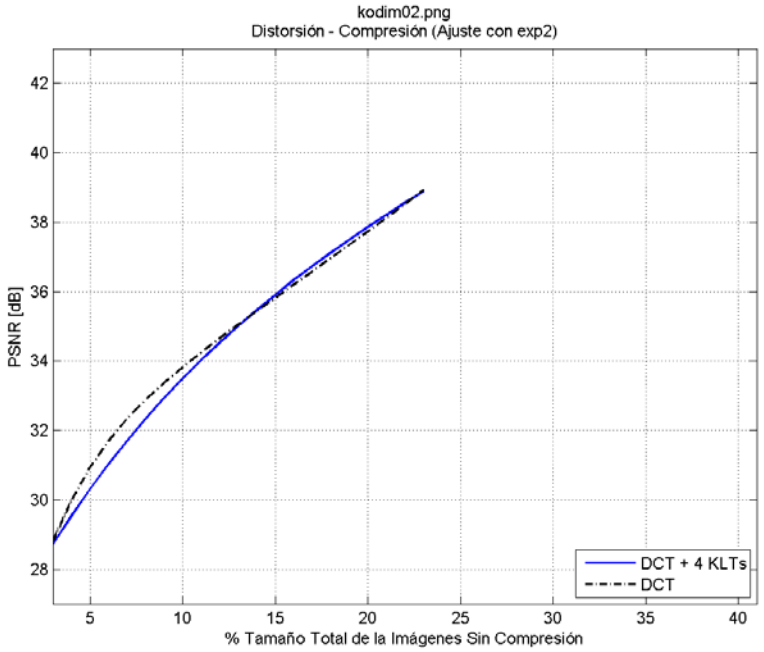
Para esta imagen, tal y como se observa en (a), el uso de 4 KLTs además de suponer una mejora muy limitada para porcentajes mayores de 13, para porcentajes menores se están obteniendo niveles de PSNR inferiores a los de JPEG, lo cual supone una clara desventaja. Además, en (b) podemos comprobar que para el resto de cantidades de KLTs usadas, los incrementos en el nivel de PSNR con respecto a JPEG son similares a los del caso de usar 4 KLTs.

Figura 4.18 'kodim02.png'.

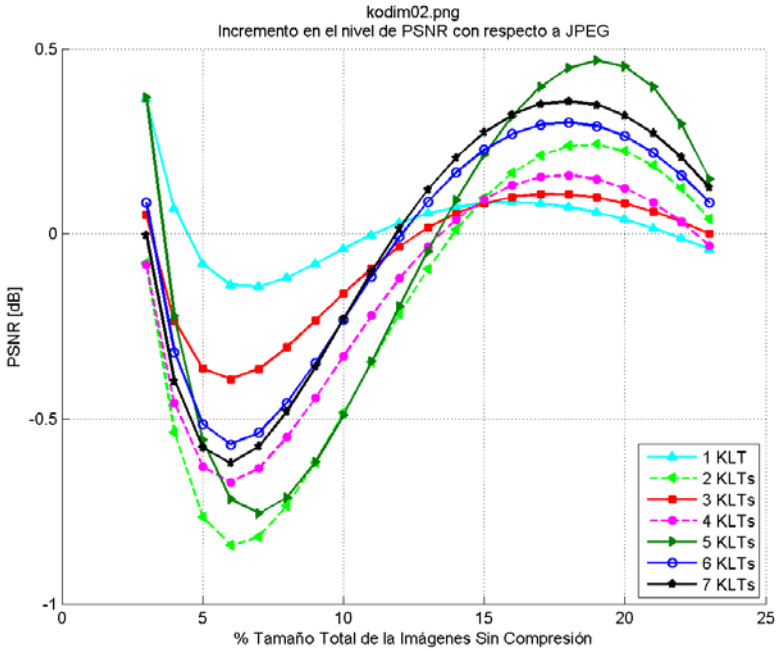
Figura 4.19 'kodim21.png'.

Con respecto a 'kodim21.png', en la figura 4.20, en © se ha representado la curva de distorsión-compresión del codificador multitransformada para 7 KLTs junto con la curva de JPEG y en (d) los incrementos en el nivel de PSNR con respecto a JPEG del codificador multitransformada al utilizar desde 1 hasta 7 KLTs.

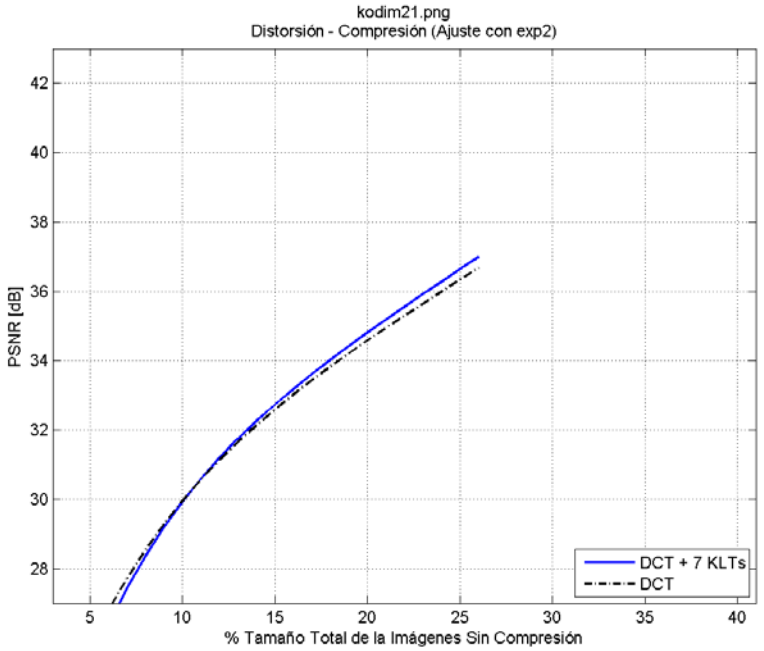
a)



b)



c)



d)

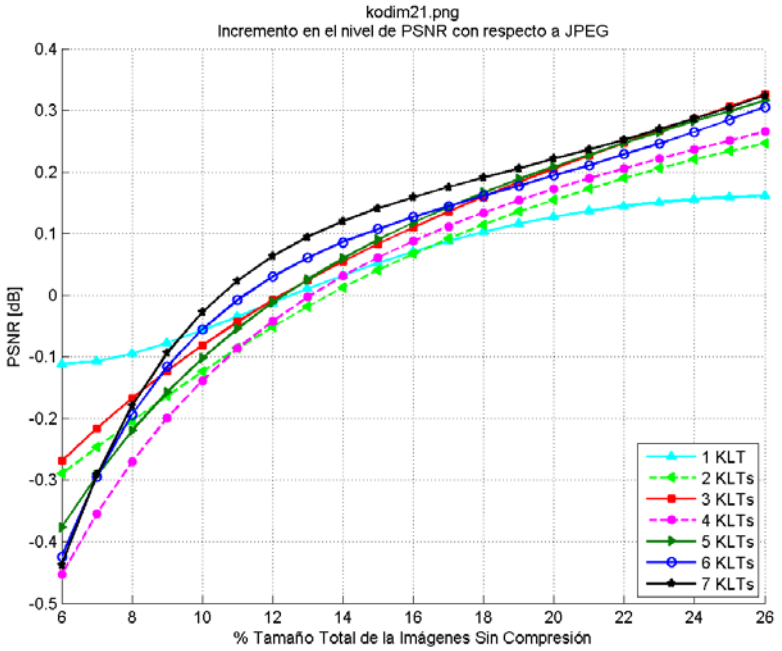


Figura 4.20 Curva Distorsión-Compresión e Incrementos de PSNR con respecto a JPEG para 'kodim02.png' (a, b) y para 'kodim21.png' (c, d).

Para ‘kodim21.png’, tras analizar las gráficas © y (d), se puede comprobar que el codificador multitransformada se comporta de forma similar a como lo hace con la imagen ‘kodim02.png’, ya que para porcentajes mayores de 12 el incremento en el nivel de PSNR con respecto a JPEG es muy limitado y para porcentajes menores, el codificador multitransformada obtiene niveles inferiores a los que obtiene JPEG. Además, en esta imagen, el hecho de incrementar el número de KLTs no produce mejoras significativas.

En estas dos imágenes, claramente el codificador multitransformada está perdiendo capacidad de compresión con respecto a JPEG, ya que el hecho de que obtenga niveles de PSNR inferiores a los de JPEG, solo puede ser debido a un desplazamiento hacia la derecha de la curva de distorsión-compresión del codificador multitransformada, provocado por un aumento del tamaño de las imágenes codificadas con respecto a JPEG. Esto es así, ya que el codificador multitransformada está diseñado de forma que selecciona para cada bloque, aquella transformada del conjunto de candidatas, que provoca mayor nivel de PSNR en la imagen decodificada, pero sin tener en cuenta el posible aumento en el tamaño de la imagen codificada.

CAPÍTULO 5 - CONCLUSIONES Y POSIBLES MEJORAS

5.1 CONCLUSIONES

El objetivo principal de este proyecto ha sido aprovechar la capacidad adaptativa de la transformada KLT, para diseñar un nuevo codificador de imágenes multitransformada, el cual mejore las prestaciones de un codificador de imágenes típico basado únicamente en la DCT. El codificador tomado como referencia ha sido el codificador JPEG en su modo secuencial y a partir de él, se ha diseñado un codificador multitransformada basado en dos etapas: por un lado, la Etapa I que es la encargada de la obtención de las matrices KLT a partir de los bloques de la imagen y por otro lado, la Etapa II que es la encargada del proceso de codificación los bloques de la imagen usando el conjunto de transformadas obtenido en la Etapa I.

Con el propósito de obtener las matrices KLT idóneas para cada imagen, la mayor parte del proyecto se ha dedicado al diseño de la Etapa I y más en concreto a sus fases de Extracción de Características y Agrupamiento. Dichas fases, finalmente han demostrado ser capaces de identificar los bloques de la imagen en los que la DCT no consigue una decorrelación eficiente de sus píxeles, caracterizar espacialmente dichos bloques mediante el uso de subbandas en el dominio de la DCT y por último, agruparlos según dicha caracterización y según el número de KLTs con el que se desee trabajar; finalmente, esto ha permitido obtener grupos de bloques con características espaciales similares y gracias a ello, a partir de cada grupo se está obteniendo una matriz KLT especializada para las estadísticas de dichos bloques.

Una vez diseñada la Etapa I, se ha procedido con el diseño de la Etapa II del codificador multitransformada. Para esta etapa, básicamente lo que se ha hecho ha sido modificar los procesos que intervienen en un codificador de imágenes por transformada típico, para poder codificar los bloques de una imagen con más de una transformada. En esta etapa también se ha incluido la fase de Selección, la cual asegura que a cada bloque de la imagen se le va a aplicar la transformada del conjunto disponible, que menos distorsión provoque en el bloque decodificado.

Con el uso de la KLT, se ha conseguido mejorar la decorrelación de los píxeles en aquellas regiones de las imágenes, en las que la DCT no consigue una decorrelación eficiente, provocando así en los bloques transformados de dichas regiones, un aumento de la compactación de la energía con respecto a la DCT. Gracias a ello, para estos bloques, se ha reducido considerablemente la energía que el proceso de cuantificación elimina de forma irreversible, consiguiendo disminuir así la distorsión en la imagen decodificada.

A la vista de los resultados globales (figura 4.3), el codificador multitransformada mejora las prestaciones del codificador JPEG ya que su curva de distorsión-compresión supera en la mayoría de los casos a la curva de JPEG. En la tabla 5.1 se recogen el incremento de PSNR con respecto a JPEG para las tasas de compresión máxima y mínima usadas, así como el incremento máximo de PSNR, para cada número de KLTs probados. Aclarar que el incremento mínimo de PSNR se obtiene con la tasa de compresión máxima usada.

# KLTs	Incremento de PSNR [dB] para la tasa de compresión al 9 % (del tamaño total sin compresión)	Máximo incremento de PSNR [dB]	Incremento de PSNR [dB] para la tasa de compresión al 27% (del tamaño total de las imágenes sin compresión)
1	0,01	0,17 (Tasa = 21 %)	0,09
2	- 0,05	0,26 (Tasa = 22 %)	0,16
3	0,1	0,35 (Tasa = 21 %)	0,25
4	0,01	0,32 (Tasa = 21 %)	0,22
5	0,05	0,39 (Tasa = 21 %)	0,27
6	0,07	0,44 (Tasa = 21 %)	0,32
7	0,09	0,49 (Tasa = 20 %)	0,35

Tabla 5.1 Incremento de PSNR del codificador multitransformada frente al codificador JPEG

La mejora del codificador multitransformada frente a JPEG es debida a que está reduciendo significativamente la distorsión en las datos decodificados pero sin embargo, a excepción de casos concretos, no está aumentado la compresión de los datos codificados, sino todo lo contrario, la compresión con respecto a JPEG en la gran mayoría de los casos se está viendo reducida. De hecho, esta pérdida en la capacidad de compresión del codificador multitransformada, está limitando en gran medida la mejora que supone usar las KLTs, llegando incluso a provocar que en determinados casos para algunas imágenes, el codificador multitransformada obtenga peores prestaciones que el codificador JPEG. La reducción en la capacidad de compresión que está sufriendo el codificador multitransformada, por un lado se debe a los bits asociados a la señalización de la transformada de cada bloque de la imagen y por otro lado se debe a que, por lo general, el codificador entrópico está devolviendo una cadena de bits de mayor longitud cuando el bloque ha sido transformado con una KLT que si se hubiera transformado con la DCT. Además de la pérdida de compresión, el uso de la KLT conlleva otros inconvenientes asociados a la no separabilidad de su núcleo de transformación, como son el cálculo y el tamaño de las matrices KLT, así como la necesaria transmisión de dichas matrices al decodificador.

Aumento de la longitud de las cadenas de bits en los bloques KLT

En el codificador multitransformada, una vez especificadas las tablas con los escalones de cuantificación y con los códigos de longitud variable respectivamente en el cuantificador y en el codificador entrópico, a cada bloque se le asigna la transformada del conjunto disponible, que menor distorsión provoca en el bloque decodificado. Cuando la fase de selección decide transformar un bloque con una KLT en vez de con la DCT, es porque la KLT está consiguiendo que se elimine menos energía de forma irreversible durante el proceso de cuantificación, gracias a que está provocando una compactación de la energía más eficiente en dicho bloque. Pero lo que suele ocurrir es que la KLT provoca un mayor número de coeficientes con una cantidad significativa de energía que la DCT, por lo que tras la cuantificación resultan menos coeficientes nulos en el caso de la KLT, provocando que el codificador entrópico necesite utilizar más bits para codificar la cadena de coeficientes cuantificados que con la DCT. Este hecho ocurre sobre todo, con aquellos bloques formados principalmente por altas frecuencias en los que la correlación entre sus píxeles no es muy alta y las imágenes base que forman el núcleo de transformación de la DCT no están adaptadas para las estadísticas de dichos píxeles; en estos casos, la KLT obtiene imágenes base más adaptadas que compactan más la energía que la DCT, pero haciéndolo en un número significativo de coeficientes.

Por otro lado, las tablas de códigos de longitud variable que se han especificado en el codificador multitransformada, han sido las mismas que en JPEG, pero dichas tablas están diseñadas según las cadenas típicas de coeficientes cuantificados que se suelen obtener en imágenes, al transformar sus bloques con la DCT, aplicarles cuantificación y finalmente vectorizarlos siguiendo un orden de zig-zag. En estas cadenas, suelen aparecer coeficientes nulos consecutivos precediendo a coeficientes no nulos, por lo que el codificador entrópico, en vez de codificar cada coeficiente de forma separada, lo que hace es codificar de forma conjunta la pareja formada por el valor de cada coeficiente no nulo, junto con el número de coeficientes nulos consecutivos que le preceden. Dependiendo de dicha pareja de valores, el codificador entrópico asigna códigos de longitud variable, asignando los más cortos a las parejas más frecuentes y los más largos a las menos frecuentes. Sin embargo, en el caso de la KLT, dado que la compactación de la energía se produce en las primeras posiciones del vector y según se avanza hacia las últimas posiciones la energía decae, es menos probable que en las cadenas de coeficientes cuantificados aparezcan coeficientes nulos consecutivos precediendo a un coeficiente no nulo, por lo que en este caso, codificar de forma conjunta el valor de cada coeficiente no nulo junto con el número de coeficientes nulos consecutivos que le preceden, no parece ser lo más adecuado.

Otro factor que también está afectando al tamaño de las cadenas de bits que el codificador entrópico devuelve al codificar los coeficientes KLT cuantificados, son los propios escalones de cuantificación usados. Los escalones de cuantificación que se están usando para los vectores KLT, son los mismos que se han especificado para los bloques DCT, pero ordenados de menor a mayor tamaño, puesto que se espera que la KLT

concentre la energía en los primeros coeficientes del vector y según se avanza hacia las últimas posiciones, la concentración de la energía disminuya. Pero dichos escalones también han sido calculados teniendo en cuenta exclusivamente la distribución típica de energía que provoca la DCT en los bloques de las imágenes, por lo que no se puede asegurar que sean los más idóneos para los coeficientes KLT.

Codificación de la señalización de la transformada

Tal y como se ha podido comprobar en los resultados mostrados en el capítulo anterior, los bits que se están utilizando para codificar el índice de la transformada que se aplica a cada bloque, están reduciendo la capacidad de compresión del codificador multitransformada. Esta reducción está siendo crítica cuando, debido al aumento del número de KLTs que se están usando en el codificador, es necesario utilizar un bit más para codificar los índices, es decir, al pasar de 1 a 2 KLTs y al pasar de 3 a 4 KLTs; en estos casos, incluso se llegan a obtener peores resultados al agregar una nueva KLT debido a que la mejora en el nivel de PSNR que provoca utilizar una KLT más, no compensa la pérdida de compresión que supone tener que utilizar un bit más por bloque para codificar los índices.

Aumento de la complejidad del cálculo

El codificador multitransformada requiere una carga computacional muy superior a la que requiere el codificador JPEG, debido en gran medida a la complejidad computacional de las fases de extracción de características y de agrupamiento de la Etapa I. Pero además, las características de la KLT hacen que, tanto para su obtención como para su aplicación a los bloques, se produzca un incremento en el cómputo con respecto a la DCT.

El núcleo de transformación de la DCT es invariable con los datos, por lo que no es necesario calcular la matriz de transformación de la DCT cada vez que se vaya a codificar una imagen. Además, su núcleo es separable y simétrico, por lo que con una única matriz de transformación del mismo tamaño que el de los bloques, se puede transformar a los mismos al dominio de la DCT.

Por otro lado, dado que los vectores base de la matriz de transformación de la DCT están compuestos por cosenos, a la hora de transformar un bloque con dicha matriz, resultan muchas multiplicaciones entre cosenos. Este hecho facilita la obtención de implementaciones “rápidas” alternativas de la DCT, ya que las simetrías de la función coseno hacen posible combinar muchos de los cálculos, reduciendo así el número de pasos necesarios para obtener los coeficientes transformados de un bloque.

Sin embargo, el núcleo de la KLT no es fijo por lo que, para cada grupo de bloques es necesario calcular la matriz de transformación correspondiente, lo cual requiere en primer lugar calcular la matriz de covarianza de los píxeles de los bloques y en segundo lugar obtener los autovectores de dicha matriz de covarianza. El número de

operaciones necesarias para calcular la matriz de covarianza de un grupo de vectores, es función del número de vectores por el tamaño de los mismos y el número de operaciones necesarias para obtener los autovectores de una matriz cuadrada, es función del orden de la matriz al cubo. Luego en el caso de la KLT, dado que los bloques de píxeles de tamaño $N \times N$ se están vectorizando en vectores de tamaño $N^2 \times 1$, la cantidad de cómputo necesario para calcular la matriz de covarianza V de un grupo de M bloques vendrá dada por $O(MN^2)$ y dado que la matriz de covarianza V tiene un tamaño $N^2 \times N^2$, la cantidad de cómputo necesario para calcular los autovectores de dicha matriz vendrá dada por $O(N^6)$.

Por otro lado, la KLT no tiene una implementación rápida directa, aunque sí se han desarrollado algunas técnicas que requieren menos cómputo pero con las que se obtienen resultados inferiores a los de la KLT directa. En el apartado 5.3.3 se da un ejemplo.

Tamaño de las matrices KLT y su envío al decodificador

Gracias a la propiedad de separabilidad y simetría del núcleo de la DCT, la transformación de un bloque de píxeles al dominio de la DCT puede llevarse a cabo mediante dos transformaciones unidimensionales, una de las columnas seguida de otra de las filas, con una única matriz de transformación del mismo tamaño que los bloques. Gracias a ello, en el caso de la DCT, no sería necesario adjuntar la matriz de transformación a la información de la imagen codificada, ya que tanto el codificador como el decodificador podrían tener almacenada una copia de esta matriz y evitar así la pérdida de compresión en las imágenes codificadas que supondría adjuntarla a las mismas.

Sin embargo, en el caso de la KLT, dado que su núcleo de transformación por lo general no es separable, la KLT no puede realizar la transformación de un bloque de píxeles (bidimensional) mediante dos transformaciones unidimensionales como ocurre con la DCT. Este hecho requiere que para transformar un bloque con la KLT, éste tenga que ser vectorizado, provocando a su vez que la matriz de transformación que se obtiene a partir de un grupo de bloques vectorizados posea un tamaño de $N^2 \times N^2$, siendo N el número de píxeles de los lados de los bloques. En nuestro caso, el codificador multitransformada se ha configurado con un tamaño de bloque de 8×8 píxeles, por lo que las matrices de transformación KLT que resultan de la Etapa I tienen un tamaño de 64×64, el cual es muy elevado, sobre todo si se tiene en cuenta que las matrices usadas en el codificador han de ser enviadas al decodificador.

El tamaño tan elevado de la matriz de transformación KLT, podría ser un factor crítico para el codificador multitransformada diseñado puesto que utiliza varias KLTs y podría darse el caso que para tamaños de imágenes pequeños, la cantidad total de información que el decodificador necesitaría para reconstruir la imagen original (imagen codificada más las matrices de transformación), se aproximara a la cantidad de información de la imagen sin codificar, por lo que no tendría sentido codificar la imagen

ya que no se estaría aplicando compresión alguna y además se estaría introduciendo distorsión.

Dado el inconveniente que supone el tamaño tan elevado de las matrices KLT, adquiere gran importancia el hecho de encontrar algún modo de reducir la información asociada a dichas matrices, antes de hacerlas llegar al decodificador. Pero tanto el tratamiento de las matrices KLT usadas en el codificador, así como la forma de hacerlas llegar al decodificador, no son objetivos de este proyecto y se proponen como trabajos futuros. Aun así, en el apartado 5.2.4 se expone una posible alternativa a la KLT convencional, que reduciría considerablemente el tamaño de su matriz de transformación, a costa de reducir la decorrelación en los datos transformados, es decir, se trataría de una transformada KLT sub-óptima.

5.2 POSIBLES MEJORAS

5.2.1 MAYOR ESPECIALIZACIÓN DE LAS MATRICES KLT

Un aumento en la especialización de las matrices KLT provocaría una decorrelación mayor en los píxeles de los bloques KLT, aumentando así la compactación de la energía en un número más reducido de coeficientes. Esto permitiría anular un mayor número de coeficientes en la cuantificación pero sin aumentar la cantidad de energía eliminada de forma irreversible, lo que se traduciría en una disminución de la distorsión y en un estancamiento o incluso en un aumento de la compresión con respecto a la DCT. Cuanto más similares sean las estadísticas de los píxeles de los bloques que forman el grupo a partir del cual se obtendrá la KLT, más especializada resultará la KLT para dichas estadísticas, luego mejorar la especialización de las KLTs pasa por mejorar los grupos de bloques que se obtienen tras el algoritmo de agrupamiento. Para ello se pueden explorar varias soluciones, como por ejemplo utilizar un algoritmo de agrupamiento diferente al K-Means, lo cual es poco recomendable ya que este algoritmo es sencillo y su carga computacional, aunque alta, no es excesiva, cosa que no ocurre con la mayoría de los algoritmos de agrupamiento, ya que suelen poseer cierta complejidad y además, suelen requerir gran carga computacional. Otra solución manteniendo el algoritmo de agrupamiento K-Means, pasaría por mejorar la caracterización de los bloques; en este sentido, caracterizar los bloques en el dominio de la DCT ha dado buenos resultados, por lo que se podría buscar una mejora de la caracterización mediante la modificación de las bandas de frecuencias en las que se ha dividido el espectro DCT o incluso extraer otra característica de las bandas.

5.2.2 CUANTIFICACIÓN Y CODIFICACIÓN ENTRÓPICA ESPECÍFICA PARA LOS BLOQUES KLT

Con respecto a la cuantificación y codificación entrópica de los bloques KLT en el codificador multitransformada, a excepción del reordenamiento de los escalones de la

tabla de cuantificación DCT, en el diseño de estos procesos no se ha tenido en cuenta la distribución típica de la energía que presentan los vectores de coeficientes KLT en imágenes. Por lo tanto, sería conveniente mediante algún experimento, intentar obtener una tabla de escalones de cuantificación adaptada para dicha distribución energética que, de forma global, provocara el máximo número posible de coeficientes nulos y redujera al máximo los valores de los coeficientes cuantificados no nulos, pero minimizando al mismo tiempo la cantidad de energía eliminada de forma irreversible. Para obtener esta tabla de cuantificación, al contrario que en la DCT, en la KLT no se podrían usar directamente consideraciones perceptuales puesto que sus funciones base por lo general no se podrán relacionar con la respuesta del sistema visual humano como ocurre con la DCT (el sistema visual humano es menos sensible a las frecuencias altas que a las bajas).

Tras obtener una tabla de cuantificación para los coeficientes KLT, sería conveniente verificar si el método utilizado en el codificador entrópico para codificar las cadenas de coeficientes cuantificados DCT, también es idóneo para las cadenas KLT; en el caso de que sí lo sea o que se decida utilizar el mismo método por simplificar el codificador, habría que calcular una nueva tabla de códigos de longitud variable a partir de vectores KLT cuantificados con los nuevos escalones. En el supuesto de que se decida modificar el método de codificación entrópica para los vectores KLT, sería necesario incluir también dicha modificación en el decodificador y obtener las correspondientes tablas de códigos de longitud variable, en función del nuevo método y de las cadenas de coeficientes cuantificados KLT. Tanto en un caso como en el otro, las nuevas tablas de escalones de cuantificación y de códigos de longitud variable, podrían definirse en el codificador multitransformada como recomendadas para los bloques KLT por lo que, al igual que las recomendadas en JPEG para los bloques DCT, éstas también podrían ir incluidas en el propio codificador/decodificador sin necesidad de agregarlas a la información codificada de cada imagen y evitar así la pérdida de compresión que eso supondría.

5.2.3 Codificación Alternativa de la Señalización de la Transformada

Una posible alternativa que redujera la cantidad de bits total que supone la señalización de las transformadas, podría ser que el codificador, en vez de transmitir al decodificador el índice de la transformada para cada bloque, que sólo lo hiciera cuando la transformada del bloque actual fuera distinta a la del bloque anterior. Lo normal es que en una imagen existan regiones formadas por grupos de bloques con características similares, por lo tanto, la probabilidad de que a dichos bloques se les acabe aplicando la misma transformada es muy alta. Por otro lado, dado que la codificación de los bloques se hace de izquierda a derecha y de arriba hacia abajo, aparecerán series de bloques consecutivos a los que se les aplique la misma transformada, por lo que sería suficiente con que el codificador transmitiera al decodificador el índice de la transformada al inicio de cada una de estas series de bloques. Para llevarlo a cabo, sería necesario crear

un carácter especial que denotase "cambio de transformada", que el codificador transmitiría al inicio de cada serie de bloques codificados, junto con el índice de la nueva transformada. A priori, este método podría ser más efectivo cuando el número de transformadas fuera reducido, ya que el cambio de transformada entre bloques consecutivos sería poco frecuente, pero cuando el número de transformadas no fuese tan reducido, el cambio sería más frecuente, por lo que se incrementaría el número de caracteres de "cambio de transformada", que el codificador tendría que transmitir al decodificador, disminuyendo así la compresión final.

5.2.4 KLT SEPARABLE SUB-ÓPTIMA

El motivo de que la matriz de transformación KLT tenga un tamaño tan grande, es porque para su cálculo se ha tenido en cuenta la correlación existente entre todos los píxeles del bloque y por lo tanto, la matriz resultante es capaz de decorrelar los píxeles en todas las direcciones del bloque; por este motivo, en nuestro caso al poseer cada bloque 64 píxeles, la matriz de transformación KLT resulta en un tamaño de 64×64. Una posible reducción del tamaño de la matriz de transformación, podría alcanzarse limitando las direcciones en las que la KLT decorrelaría los píxeles, asumiendo un empeoramiento de los resultados con respecto a la KLT original.

En [Feng, 1999], para transformar bloques de 8×8 píxeles con la KLT, proponen la sustitución de la matriz de transformación de tamaño 64×64, por el uso de una KLT separable sub-óptima que solo decorrele en la dirección vertical y horizontal. Esta KLT separable está formada por dos matrices de tamaño 8×8, en la que una de ellas decorrela los píxeles de los bloques columna por columna y la otra decorrela los píxeles fila por fila. La combinación de este par de transformadas es una transformada sub-óptima que se aproxima al rendimiento de la KLT directa, pero al sustituir la matriz de 64×64 por dos de 8×8, reduce en gran medida el cómputo necesario para la obtención de las mismas y para su aplicación a los bloques y sobre todo, reduce drásticamente la cantidad de información correspondiente a las matrices de transformación, que es necesario hacer llegar al decodificador.

La forma de aplicar esta transformación KLT separable a un bloque de píxeles P, para obtener el bloque de coeficientes transformados C, se realiza así:

$$C = B_c P B_f^T \tag{5.1}$$

en donde B_c denota la matriz de transformación KLT para las columnas y B_f denota la matriz de transformación KLT para las filas.

Dado que B_c únicamente tiene en cuenta la correlación entre píxeles dentro de las columnas, para su obtención se concatenan en una matriz todas las columnas de todos los bloques del grupo a partir del cual se quiera obtener la KLT separable y a partir de dicha matriz se calcula la matriz de covarianza, cuyos autovectores formarán la matriz

de transformación B_c. El mismo procedimiento se sigue para B_f, pero en este caso concatenando las filas de los bloques en una única matriz.

Por otro lado, en el decodificador se aplica la transformación inversa a los bloques de coeficientes del siguiente modo:

$$P = (B_c^{-1})C(B_f^{-1})^T \tag{5.2}$$

y dado que las matrices de transformación KLT son ortonormales, finalmente la transformación inversa resulta:

$$P = B_c^T C B_f \tag{5.3}$$

Por último, en [Feng, 2002] también se comenta la posibilidad de, en vez de utilizar dos matrices distintas para B_c y B_f, utilizar una única matriz de transformación B_{cf} para las filas y para las columnas, asumiendo que las estadísticas de las filas y de las columnas en los bloques de las imágenes suelen ser similares. De esta forma, se estaría reduciendo a la mitad la cantidad de información asociada a las matrices de transformación.

Por lo tanto, en este caso se concatenarían todas las columnas y todas las filas traspuestas de los bloques, en una única matriz y a partir de ella se calcularía la matriz de covarianza muestral, cuyos autovectores formarían la matriz de transformación B_{cf}.

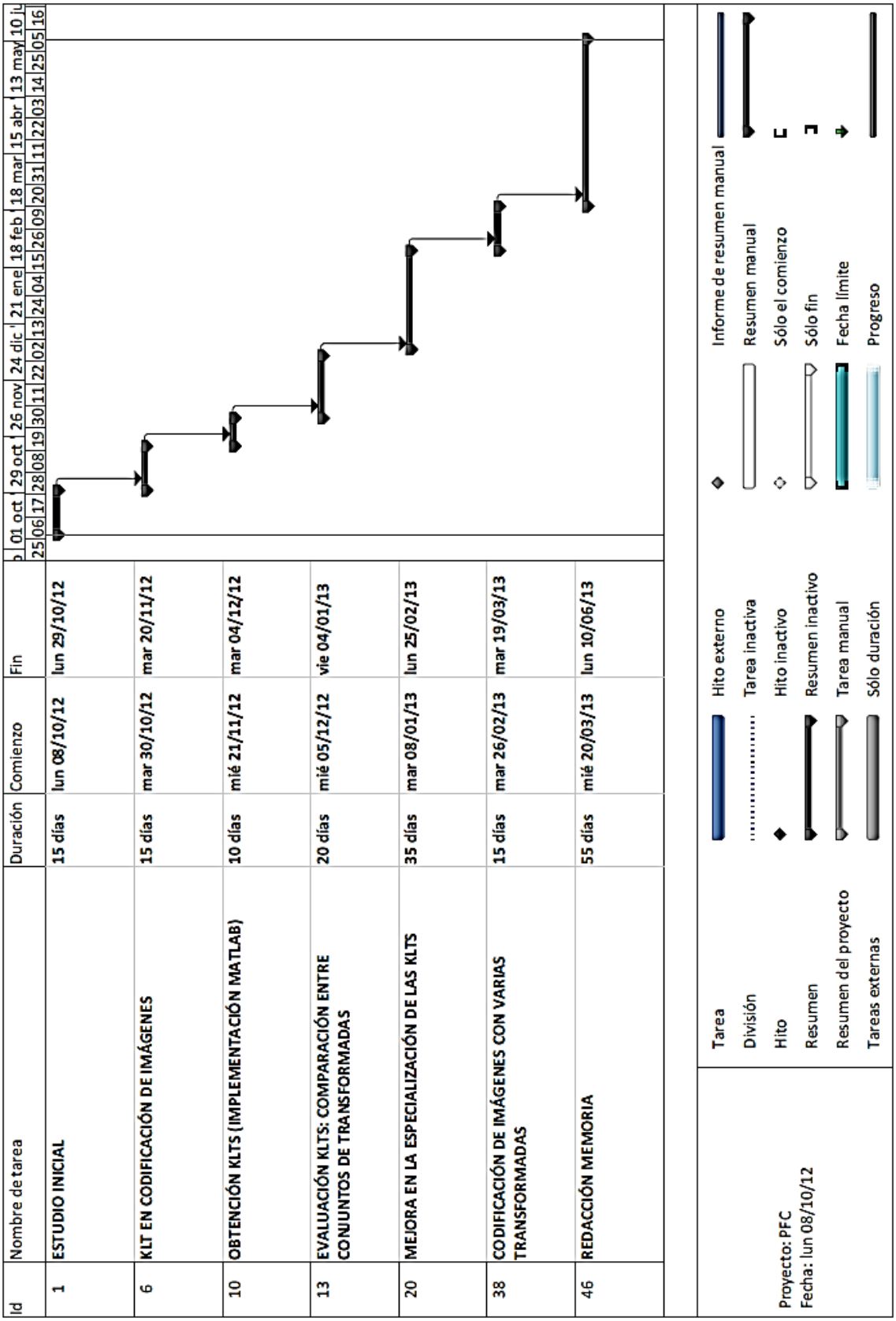
En este caso, la transformación de un bloque resultaría así:

$$C = B_{cf} P B_{cf}^T \tag{5.4}$$

y la transformación inversa así:

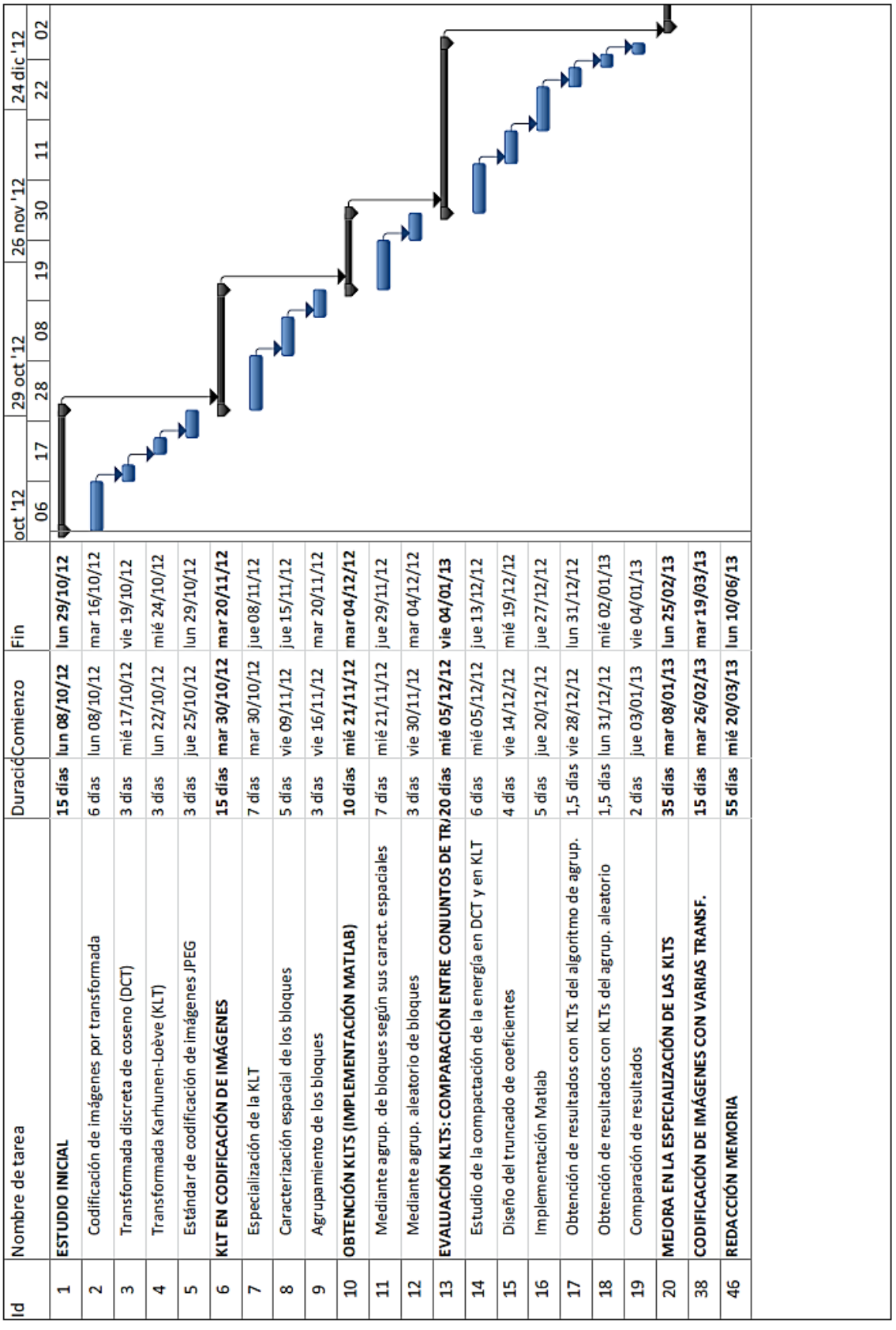
$$P = B_{cf}^T C B_{cf} \tag{5.5}$$

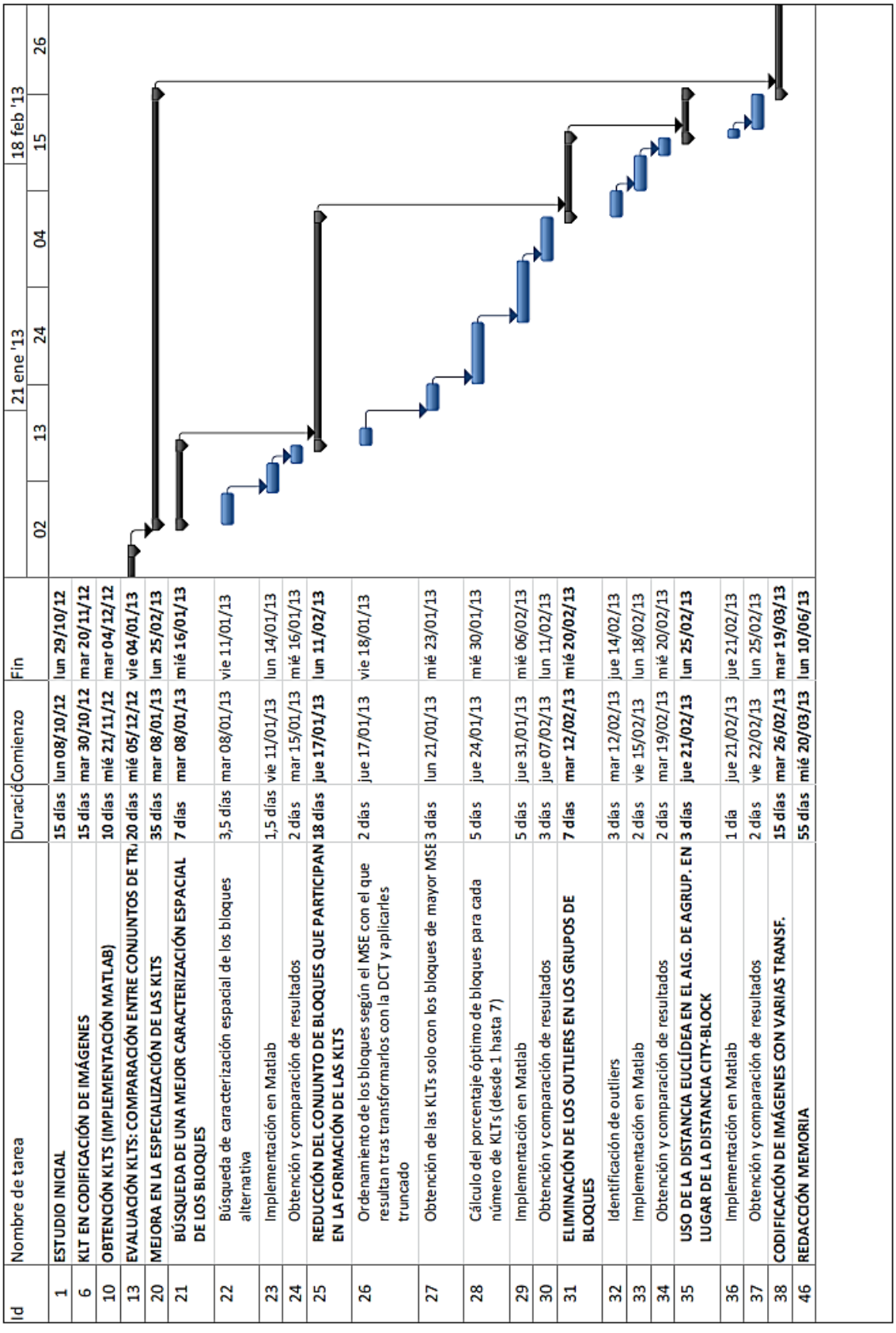
PLANIFICACIÓN



Id	Nombre de tarea	Duración	Comienzo	Fin
1	ESTUDIO INICIAL	15 días	lun 08/10/12	lun 29/10/12
6	KLT EN CODIFICACIÓN DE IMÁGENES	15 días	mar 30/10/12	mar 20/11/12
10	OBTENCIÓN KLTS (IMPLEMENTACIÓN MATLAB)	10 días	mié 21/11/12	mar 04/12/12
13	EVALUACIÓN KLTS: COMPARACIÓN ENTRE CONJUNTOS DE TRANSFORMADAS	20 días	mié 05/12/12	vie 04/01/13
20	MEJORA EN LA ESPECIALIZACIÓN DE LAS KLTS	35 días	mar 08/01/13	lun 25/02/13
38	CODIFICACIÓN DE IMÁGENES CON VARIAS TRANSFORMADAS	15 días	mar 26/02/13	mar 19/03/13
46	REDACCIÓN MEMORIA	55 días	mié 20/03/13	lun 10/06/13

Id	Nombre de tarea	Duración	Comienzo	Fin
1	**ESTUDIO INICIAL**	**15 días**	**lun 08/10/12**	**lun 29/10/12**
2	Codificación de imágenes por transformada	6 días	lun 08/10/12	mar 16/10/12
3	Transformada discreta de coseno (DCT)	3 días	mié 17/10/12	vie 19/10/12
4	Transformada Karhunen-Loève (KLT)	3 días	lun 22/10/12	mié 24/10/12
5	Estándar de codificación de imágenes JPEG	3 días	jue 25/10/12	lun 29/10/12
6	**KLT EN CODIFICACIÓN DE IMÁGENES**	**15 días**	**mar 30/10/12**	**mar 20/11/12**
7	Especialización de la KLT	7 días	mar 30/10/12	jue 08/11/12
8	Caracterización espacial de los bloques	5 días	vie 09/11/12	jue 15/11/12
9	Agrupamiento de los bloques	3 días	vie 16/11/12	mar 20/11/12
10	**OBTENCIÓN KLTS (IMPLEMENTACIÓN MATLAB)**	**10 días**	**mié 21/11/12**	**mar 04/12/12**
11	Mediante agrup. de bloques según sus caract. espaciales	7 días	mié 21/11/12	jue 29/11/12
12	Mediante agrup. aleatorio de bloques	3 días	vie 30/11/12	mar 04/12/12
13	**EVALUACIÓN KLTS: COMPARACIÓN ENTRE CONJUNTOS DE TR**	**20 días**	**mié 05/12/12**	**vie 04/01/13**
14	Estudio de la compactación de la energía en DCT y en KLT	6 días	mié 05/12/12	jue 13/12/12
15	Diseño del truncado de coeficientes	4 días	vie 14/12/12	mié 19/12/12
16	Implementación Matlab	5 días	jue 20/12/12	jue 27/12/12
17	Obtención de resultados con KLTs del algoritmo de agrup.	1,5 días	vie 28/12/12	lun 31/12/12
18	Obtención de resultados con KLTs del agrup. aleatorio	1,5 días	lun 31/12/12	mié 02/01/13
19	Comparación de resultados	2 días	jue 03/01/13	vie 04/01/13
20	**MEJORA EN LA ESPECIALIZACIÓN DE LAS KLTS**	**35 días**	**mar 08/01/13**	**lun 25/02/13**
38	**CODIFICACIÓN DE IMÁGENES CON VARIAS TRANSF.**	**15 días**	**mar 26/02/13**	**mar 19/03/13**
46	**REDACCIÓN MEMORIA**	**55 días**	**mié 20/03/13**	**lun 10/06/13**





Id	Nombre de tarea	Duració	Comienzo	Fin
1	**ESTUDIO INICIAL**	15 días	lun 08/10/12	lun 29/10/12
6	**KLT EN CODIFICACIÓN DE IMÁGENES**	15 días	mar 30/10/12	mar 20/11/12
10	**OBTENCIÓN KLTS (IMPLEMENTACIÓN MATLAB)**	10 días	mié 21/11/12	mar 04/12/12
13	**EVALUACIÓN KLTS: COMPARACIÓN ENTRE CONJUNTOS DE TR**	20 días	mié 05/12/12	vie 04/01/13
20	**MEJORA EN LA ESPECIALIZACIÓN DE LAS KLTS**	35 días	mar 08/01/13	lun 25/02/13
21	**BÚSQUEDA DE UNA MEJOR CARACTERIZACIÓN ESPACIAL DE LOS BLOQUES**	7 días	mar 08/01/13	mié 16/01/13
22	Búsqueda de caracterización espacial de los bloques alternativa	3,5 días	mar 08/01/13	vie 11/01/13
23	Implementación en Matlab	1,5 días	vie 11/01/13	lun 14/01/13
24	Obtención y comparación de resultados	2 días	mar 15/01/13	mié 16/01/13
25	**REDUCCIÓN DEL CONJUNTO DE BLOQUES QUE PARTICIPAN EN LA FORMACIÓN DE LAS KLTS**	18 días	jue 17/01/13	lun 11/02/13
26	Ordenamiento de los bloques según el MSE con el que resultan tras transformarlos con la DCT y aplicarles truncado	2 días	jue 17/01/13	vie 18/01/13
27	Obtención de las KLTs solo con los bloques de mayor MSE	3 días	lun 21/01/13	mié 23/01/13
28	Cálculo del porcentaje óptimo de bloques para cada número de KLTs (desde 1 hasta 7)	5 días	jue 24/01/13	mié 30/01/13
29	Implementación en Matlab	5 días	jue 31/01/13	mié 06/02/13
30	Obtención y comparación de resultados	3 días	jue 07/02/13	lun 11/02/13
31	**ELIMINACIÓN DE LOS OUTLIERS EN LOS GRUPOS DE BLOQUES**	7 días	mar 12/02/13	mié 20/02/13
32	Identificación de outliers	3 días	mar 12/02/13	jue 14/02/13
33	Implementación en Matlab	2 días	vie 15/02/13	lun 18/02/13
34	Obtención y comparación de resultados	2 días	mar 19/02/13	mié 20/02/13
35	**USO DE LA DISTANCIA EUCLÍDEA EN EL ALG. DE AGRUP. EN LUGAR DE LA DISTANCIA CITY-BLOCK**	3 días	jue 21/02/13	lun 25/02/13
36	Implementación en Matlab	1 día	jue 21/02/13	jue 21/02/13
37	Obtención y comparación de resultados	2 días	vie 22/02/13	lun 25/02/13
38	**CODIFICACIÓN DE IMÁGENES CON VARIAS TRANSF.**	15 días	mar 26/02/13	mar 19/03/13
46	**REDACCIÓN MEMORIA**	55 días	mié 20/03/13	lun 10/06/13

Id	Nombre de tarea	Duración	Comienzo	Fin
1	**ESTUDIO INICIAL**	**15 días**	**lun 08/10/12**	**lun 29/10/12**
6	**KLT EN CODIFICACIÓN DE IMÁGENES**	**15 días**	**mar 30/10/12**	**mar 20/11/12**
10	**OBTENCIÓN KLTS (IMPLEMENTACIÓN MATLAB)**	**10 días**	**mié 21/11/12**	**mar 04/12/12**
13	**EVALUACIÓN KLTS: COMPARACIÓN ENTRE CONJUNTOS DE TR**	**20 días**	**mié 05/12/12**	**vie 04/01/13**
20	**MEJORA EN LA ESPECIALIZACIÓN DE LAS KLTS**	**35 días**	**mar 08/01/13**	**lun 25/02/13**
38	**CODIFICACIÓN DE IMÁGENES CON VARIAS TRANSF.**	**15 días**	**mar 26/02/13**	**mar 19/03/13**
39	Modificación fase de transformación de JPEG	1 día	mar 26/02/13	mar 26/02/13
40	Modificacion de la fase de cuantificación de JPEG	1 día	mié 27/02/13	mié 27/02/13
41	Modificación de la fase de codificación entrópica de JPEG	2 días	jue 28/02/13	vie 01/03/13
42	Diseño de la fase de Selección	3 días	lun 04/03/13	mié 06/03/13
43	Implementación Matlab	4 días	jue 07/03/13	mar 12/03/13
44	Obtención resultados finales	1 día	mié 13/03/13	mié 13/03/13
45	Comparación codificador multitransformada con codificador JPEG	3 días	jue 14/03/13	mar 19/03/13
46	**REDACCIÓN MEMORIA**	**55 días**	**mié 20/03/13**	**lun 10/06/13**
47	Introducción y objetivos	3 días	mié 20/03/13	vie 22/03/13
48	Estado del arte	15 días	lun 25/03/13	mar 16/04/13
49	Codificación de imágenes con transformadas adaptativas	20 días	mié 17/04/13	jue 16/05/13
50	Evaluación del codificador multitransformada frente al codificador JPEG	10 días	vie 17/05/13	jue 30/05/13
51	Conclusiones y posibles mejoras	5 días	vie 31/05/13	jue 06/06/13
52	Planificación y presupuesto	2 días	vie 07/06/13	lun 10/06/13

PRESUPUESTO

1) Ejecución Material

- Material fungible (papel, tóner de impresión,…) ..... 200 €.
- Equipos (hardware):
  - Ordenador completo (CPU, pantalla, teclado y ratón) ..... 1200 €.
  - Impresora ..... 100 €.
- Programas (software):
  - Microsoft Windows 7 Ultimate ..... 320 €.
  - Microsoft Office 2010 Professional ..... 700 €.
  - Matlab R2009b (con Curve Fitting Toolbox) ..... 3000 €.
  - Adobe Acrobat X Pro ..... 680 €.

Ejecución Material ..... 6200 €.

2) Gastos generales

- 16 % sobre Ejecución Material ..... 992 €.

3) Beneficio Industrial

- 6 % sobre Ejecución Material ..... 372 €.

4) Honorarios Proyecto

- 1240 horas a 15 € / hora ..... 18600 €.

5) Subtotal del presupuesto

- Subtotal presupuesto ..... 26164 €.

6) I.V.A. aplicable

- 21 % sobre Subtotal presupuesto ..... 5494,44 €.

7) Total presupuesto

- Total presupuesto ..... 31658,44 €.

Madrid, Octubre de 2012
El Ingeniero Jefe de Proyecto

Fdo: Valentín Cruz Rodríguez
Ingeniero Técnico de Telecomunicación.

REFERENCIAS

[Britanak, 2007] V. Britanak, Kamisetty R. Rao, Pat C. Yip, "Discrete Cosine and Sine Transforms: General Properties, Fast Algorithms and Approximations", Elsevier, 2007.

[Clarke, 1981] R. J. Clarke, "Relation Between the Karhunen-Loève and Cosine Transforms", IEE PROC., Vol. 128, Pt. F, No. 6, Noviembre 1981.

[Dufaux, 2009] Frédéric Dufaux, Gary J. Sullivan, Touradj Ebrahimi, "The JPEG XR Image Coding Standard", IEEE Signal Processing Magacine, Noviembre 2009.

[Feng, 1999] Hanying Feng, Michelle Effros, "Separable Karhunen-Loève Transforms for the Weighted Universal Transform Codign Algorithm", IEEE, 1999.

[Feng, 2002] On the Rate-Distorsion Performance and Computational Efficiency of the Karhunen-Loève Transform for Lossy Data Compression.

[Gao, 2008] Yi Gao, Jiazhong Chen, Shengsheng Yu, Jingli Zhou, Lai-Man Po, "The training of Karhunen-Loève transform matrix and its application for H.264 intra coding", Springer Science + Business Media, 2008.

[Gonzalez, 2002] Rafael C. Gonzalez, Richard E. Woods, "Digital Image Processing", 2nd Ed., Prentice Hall, 2002.

[Goyal, 2001] Vivek K. Goyal, "Theorical Foundations Of Transform Coding", IEEE Signal Processing Magazine, September 2001.

[Khayam, 2003] Syed Ali Khayam, "The Discrete Cosine Transform (DCT): Theory and Application", Tutorial, Department of Electrical & Computer Engineering, Michigan State University, 2003.

[Lay, 1999] David C. Lay "Álgebra Lineal Y Sus Aplicaciones 2ª Ed.", Addison Wesley Longman de México, 1999.

[Lim, 1990] Jae Slim, "Two-Dimensional Signal and Image Processing", Prencice Hall, 1990.

[Lohscheller,1984] H. Lohscheller "A Subjectively Adapted Image Communication

System", IEEE Trans. Commun. COM-32, December 1984.

[Rao, 2001] Kamisetty R. Rao, Pat C. Yip, "The Transform And Data Compression Handbook", Cap. 1, CRC Press, 2001.

[Ray, 1970] W.D. Ray, R. M. Driver, "Futher Descomposition of the Karhunen-Loève Series Representation of a Stationary Random Process", IEEE Trans. 1970, IT-16, pp. 845-850.

[Richardson, 2003] Iain E. G. Richardson, "H.264 and MPEG-4 Video Compression", Wiley, 2003.

[Salomon, 2007] David Salomon, "Data compression: The Complete Reference", 4ª Ed., Springer, 2007.

[Shi, 2000] Yun Q.Shi, Huifang Sun, "Image and Video Compression for Multimedia Engineering: Fundamentals, Algorithms and Standars", CRC Press, 2000.

[Skodras, 2001] Athanassios Skodras, Charilaos Christopoulos, Touradj Ebrahimi, "The JPEG 2000 Still Image Compression Standard", IEEE Signal Processing Magazine, September 2001.

[Usevitch, 2001] Bryan E. Usevitch, "A Tutorial on Modern Lossy Wavelet Image Compression: Foundations of JPEG 2000", IEEE Signal Processing Magazine, September 2001.

[Wallace, 1991] Gregory K. Wallace, "The JPEG Still Picture Compression Standard", IEEE Transactions on Consumer Electronics, December 1991.

[Wintz, 1972] Paul A. Wintz, "Transform Picture Coding", Proceedings of the IEEE, vol.60, Nº 7, 1972.

[Wu, 2006] H.R. Wu, K.R. Rao, "Digital Video Image Quality and Perceptual Coding", CRC Press, 2006.

#	imagen	Asignación	DCT	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
1	lenna.bmp	Aleatoria	9,5903	7,6259	7,3786	7,1732	6,9613	6,7413	6,5777	6,3975
		Alg. Agrup.			7,3128	6,9637	6,6730	6,5476	6,1992	6,0421
2	baboon.bmp	Aleatoria	123,9530	103,0680	100,7669	98,7148	96,5287	94,6251	92,8018	90,5169
		Alg. Agrup.			88,5094	84,3443	82,9224	81,7381	79,9977	78,5595
3	barbara.bmp	Aleatoria	29,2411	12,6261	12,3588	12,1755	11,8642	11,7054	11,3379	11,2186
		Alg. Agrup.			12,0781	10,8480	7,6281	7,2670	7,1041	6,8116
4	kodim01.png	Aleatoria	50,0602	45,1724	44,5727	43,9978	43,4370	42,9600	42,4385	41,9511
		Alg. Agrup.			41,7475	40,5413	40,1515	39,3982	38,4525	37,7035
5	Kodim02.png	Aleatoria	13,0962	11,5234	11,1513	10,8397	10,4796	10,2490	10,0179	9,8402
		Alg. Agrup.			9,7060	9,4502	9,3297	8,9354	8,8447	8,7634
6	kodim03.png	Aleatoria	11,1519	6,7658	6,5238	6,3397	6,1164	5,9274	5,7773	5,5894
		Alg. Agrup.			6,4080	5,4284	4,9845	4,7044	4,4611	4,3289
7	kodim04.png	Aleatoria	12,5972	10,3453	10,0762	9,8583	9,6383	9,4170	9,2045	9,0366
		Alg. Agrup.			9,9758	9,6037	9,0980	8,9979	8,7670	8,5937
8	kodim05.png	Aleatoria	64,5380	49,9838	49,0251	47,8802	47,3197	46,6378	45,9446	45,0943
		Alg. Agrup.			45,0474	42,0439	40,6483	39,6117	38,7590	38,0189
9	kodim06.png	Aleatoria	40,7274	31,4934	30,7100	30,0019	29,2271	28,5683	27,9632	27,3033
		Alg. Agrup.			28,9716	26,2742	25,1911	24,4027	23,9408	23,7667
10	kodim07.png	Aleatoria	12,0049	9,9180	9,5541	9,1959	8,9247	8,6344	8,3046	7,9881
		Alg. Agrup.			9,2304	8,5321	8,0526	7,3572	7,0230	6,8636
11	kodim08.png	Aleatoria	76,7571	64,9175	63,6555	62,4706	61,4700	60,6135	59,5212	58,4684
		Alg. Agrup.			58,4949	56,3544	54,2979	52,3745	51,3463	49,3024
12	kodim09.png	Aleatoria	12,1501	11,2675	10,7930	10,3561	10,0163	9,6522	9,2657	9,0529
		Alg. Agrup.			10,6207	8,4864	7,9551	7,6025	7,1717	6,8953
13	kodim10.png	Aleatoria	15,7615	10,4574	9,9766	9,6233	9,2645	8,9515	8,5783	8,3393
		Alg. Agrup.			8,4893	8,0513	7,2929	7,0467	6,6950	6,3438
14	kodim11.png	Aleatoria	31,4464	28,9290	28,0723	27,1732	26,3754	25,6231	24,9207	24,1985
		Alg. Agrup.			24,0934	23,3487	22,3280	21,7570	20,4412	20,0777
15	kodim12.png	Aleatoria	10,9331	9,1458	8,7088	8,3825	8,0473	7,7839	7,4915	7,2390
		Alg. Agrup.			8,7624	6,8502	6,6197	6,3514	6,1379	5,8670
16	kodim13.png	Aleatoria	130,5531	110,2018	108,3255	106,5566	104,7715	103,0050	101,1948	99,4696
		Alg. Agrup.			108,3082	103,6696	102,2000	99,6228	98,5883	95,2728
17	kodim14.png	Aleatoria	33,9268	28,7545	28,0197	27,5357	26,9673	26,3765	25,8801	25,2763
		Alg. Agrup.			27,7660	25,7017	25,1306	24,7911	23,9690	23,4017
18	kodim15.png	Aleatoria	18,4593	13,3682	12,9211	12,4611	12,1156	11,6645	11,3091	10,9750
		Alg. Agrup.			12,3151	11,5775	10,6069	10,1049	9,5193	9,2354
19	kodim16.png	Aleatoria	17,8092	12,0590	11,7244	11,4195	11,1568	10,8599	10,6105	10,4013
		Alg. Agrup.			11,4819	9,9790	9,8055	9,7471	9,4165	9,1696
20	kodim17.png	Aleatoria	19,3280	15,8247	14,9703	14,2315	13,5908	12,9877	12,3586	11,9002
		Alg. Agrup.			14,8351	13,7628	12,5608	11,2682	10,9270	10,6202
21	kodim18.png	Aleatoria	48,9941	42,6583	41,3048	40,2276	39,1901	38,0747	37,2277	36,2888
		Alg. Agrup.			41,3977	40,3873	38,8712	37,7358	36,9989	35,0100
22	kodim19.png	Aleatoria	26,8746	23,3726	22,8352	22,2746	21,8792	21,4566	21,0185	20,6575
		Alg. Agrup.			22,3829	19,0780	18,2105	17,0646	16,6453	16,2531
23	kodim20.png	Aleatoria	19,4638	16,4934	15,6304	14,9177	14,2179	13,5188	13,0056	12,5245
		Alg. Agrup.			15,3654	13,8138	12,6038	12,5002	11,9534	11,6820
24	kodim21.png	Aleatoria	36,4192	31,1841	30,3424	29,4989	28,8026	28,0857	27,3818	26,6173
		Alg. Agrup.			29,5926	28,7072	27,2607	26,6065	25,9404	25,1714
25	kodim22.png	Aleatoria	22,6036	20,7089	20,0418	19,5005	19,0407	18,5238	18,0475	17,6405
		Alg. Agrup.			18,7106	18,0925	17,8253	17,0923	16,7461	16,1697
26	kodim23.png	Aleatoria	7,9042	5,6902	5,2428	4,8425	4,5112	4,2418	4,0026	3,7785
		Alg. Agrup.			5,2499	4,8397	4,1589	4,0185	3,6226	3,5145
27	kodim24.png	Aleatoria	62,8695	51,8493	49,9660	48,3384	46,8600	45,3545	43,9045	42,6258
		Alg. Agrup.			49,4606	47,5726	45,6427	43,7786	42,6302	42,6115

TABLA A.1: MSE (TOTAL BLOQUES - AGRUP: ENERGÍA NORM. - DIST. CITY BLOCK - SIN REASIGNACIÓN)

#	imagen	Asignación	DCT	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
1	lenna.bmp	Aleatoria	38,3125	39,3079	39,4511	39,5737	39,7039	39,8433	39,9501	40,0707
		Alg. Agrup.			39,4900	39,7024	39,8876	39,9700	40,2075	40,3189
2	baboon.bmp	Aleatoria	27,1982	27,9996	28,0976	28,1870	28,2842	28,3707	28,4552	28,5635
		Alg. Agrup.			28,6609	28,8702	28,9441	29,0066	29,1000	29,1788
3	barbara.bmp	Aleatoria	33,4709	37,1181	37,2110	37,2759	37,3884	37,4469	37,5855	37,6314
		Alg. Agrup.			37,3108	37,7773	39,3066	39,5172	39,6157	39,7983
4	kodim01.png	Aleatoria	31,1359	31,5821	31,6401	31,6965	31,7522	31,8002	31,8532	31,9034
		Alg. Agrup.			31,9245	32,0518	32,0938	32,1760	32,2816	32,3670
5	Kodim02.png	Aleatoria	36,9594	37,5150	37,6575	37,7806	37,9274	38,0240	38,1231	38,2008
		Alg. Agrup.			38,2604	38,3764	38,4321	38,6197	38,6640	38,7041
6	kodim03.png	Aleatoria	37,6573	39,8276	39,9858	40,1101	40,2658	40,4022	40,5136	40,6572
		Alg. Agrup.			40,0636	40,7841	41,1546	41,4058	41,6364	41,7670
7	kodim04.png	Aleatoria	37,1280	37,9834	38,0978	38,1928	38,2908	38,3917	38,4908	38,5707
		Alg. Agrup.			38,1413	38,3064	38,5414	38,5894	38,7023	38,7890
8	kodim05.png	Aleatoria	30,0326	31,1425	31,2266	31,3292	31,3804	31,4434	31,5085	31,5896
		Alg. Agrup.			31,5941	31,8938	32,0404	32,1526	32,2471	32,3308
9	kodim06.png	Aleatoria	32,0319	33,1486	33,2580	33,3593	33,4729	33,5720	33,6649	33,7686
		Alg. Agrup.			33,5111	33,9355	34,1183	34,2564	34,3394	34,3711
10	kodim07.png	Aleatoria	37,3372	38,1666	38,3289	38,4949	38,6249	38,7685	38,9376	39,1064
		Alg. Agrup.			38,4786	38,8203	39,0714	39,4637	39,6656	39,7653
11	kodim08.png	Aleatoria	29,2796	30,0072	30,0924	30,1740	30,2442	30,3051	30,3841	30,4616
		Alg. Agrup.			30,4596	30,6215	30,7830	30,9396	31,0257	31,2021
12	kodim09.png	Aleatoria	37,2850	37,6125	37,7994	37,9789	38,1237	38,2846	38,4620	38,5629
		Alg. Agrup.			37,8693	38,8436	39,1243	39,3213	39,5746	39,7453
13	kodim10.png	Aleatoria	36,1548	37,9366	38,1410	38,2976	38,4626	38,6118	38,7968	38,9195
		Alg. Agrup.			38,8421	39,0722	39,5018	39,6509	39,8733	40,1073
14	kodim11.png	Aleatoria	33,1551	33,5175	33,6480	33,7894	33,9188	34,0445	34,1652	34,2929
		Alg. Agrup.			34,3118	34,4482	34,6423	34,7548	35,0257	35,1037
15	kodim12.png	Aleatoria	37,7434	38,5186	38,7312	38,8971	39,0743	39,2188	39,3851	39,5340
		Alg. Agrup.			38,7046	39,7738	39,9224	40,1021	40,2506	40,4466
16	kodim13.png	Aleatoria	26,9729	27,7089	27,7835	27,8550	27,9284	28,0022	28,0792	28,1539
		Alg. Agrup.			27,7842	27,9743	28,0363	28,1472	28,1926	28,3411
17	kodim14.png	Aleatoria	32,8254	33,5437	33,6562	33,7318	33,8224	33,9186	34,0011	34,1037
		Alg. Agrup.			33,6957	34,0312	34,1288	34,1878	34,3343	34,4383
18	kodim15.png	Aleatoria	35,4686	36,8701	37,0178	37,1752	37,2974	37,4621	37,5965	37,7268
		Alg. Agrup.			37,2264	37,4946	37,8749	38,0855	38,3447	38,4762
19	kodim16.png	Aleatoria	35,6243	37,3177	37,4399	37,5543	37,6554	37,7725	37,8735	37,9599
		Alg. Agrup.			37,5307	38,1399	38,2161	38,2420	38,3919	38,5073
20	kodim17.png	Aleatoria	35,2689	36,1374	36,3785	36,5983	36,7984	36,9955	37,2111	37,3753
		Alg. Agrup.			36,4179	36,7437	37,1406	37,6122	37,7458	37,8695
21	kodim18.png	Aleatoria	31,2294	31,8308	31,9708	32,0856	32,1990	32,3244	32,4221	32,5331
		Alg. Agrup.			31,9610	32,0684	32,2345	32,3633	32,4489	32,6889
22	kodim19.png	Aleatoria	33,8374	34,4437	34,5447	34,6527	34,7305	34,8152	34,9048	34,9800
		Alg. Agrup.			34,6316	35,3255	35,5276	35,8098	35,9179	36,0214
23	kodim20.png	Aleatoria	35,2385	35,9577	36,1911	36,3938	36,6025	36,8214	36,9895	37,1532
		Alg. Agrup.			36,2654	36,7277	37,1258	37,1616	37,3559	37,4556
24	kodim21.png	Aleatoria	32,5175	33,1915	33,3103	33,4328	33,5365	33,6460	33,7562	33,8792
		Alg. Agrup.			33,4190	33,5509	33,7754	33,8809	33,9910	34,1217
25	kodim22.png	Aleatoria	34,5890	34,9692	35,1114	35,2303	35,3340	35,4535	35,5666	35,6657
		Alg. Agrup.			35,4099	35,5558	35,6204	35,8028	35,8917	36,0438
26	kodim23.png	Aleatoria	39,1522	40,5795	40,9352	41,2801	41,5879	41,8553	42,1074	42,3576
		Alg. Agrup.			40,9293	41,2826	41,9410	42,0902	42,5406	42,6722
27	kodim24.png	Aleatoria	30,1464	30,9834	31,1441	31,2879	31,4228	31,5646	31,7057	31,8341
		Alg. Agrup.			31,1882	31,3572	31,5371	31,7182	31,8336	31,8355

TABLA A.2: PSNR [dB] (TOTAL BLOQUES - AGRUP: ENERGÍA NORM. - DIST. CITY BLOCK - SIN REASIGNACIÓN)

#	imagen	Asignación	DCT	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
1	lenna.bmp	Aleatoria	9,5903	7,2768	6,8219	6,5306	6,2686	6,0703	5,8697	5,6767
		Alg. Agrup.			6,7082	6,2843	6,0419	5,8584	5,5710	5,4447
2	baboon.bmp	Aleatoria	123,9530	95,4840	90,5578	87,1406	84,2669	82,5453	81,0699	79,1932
		Alg. Agrup.			81,3952	78,9481	76,7207	74,8358	72,9041	71,5269
3	barbara.bmp	Aleatoria	29,2411	11,0546	10,2029	9,5856	8,9473	8,7490	8,0275	7,8822
		Alg. Agrup.			9,3072	8,3795	5,9323	5,6387	5,5601	5,3276
4	kodim01.png	Aleatoria	50,0602	42,3633	40,6998	39,5307	38,6664	37,9076	37,2156	36,2880
		Alg. Agrup.			38,4298	36,8731	36,1082	35,3483	34,4720	33,6027
5	Kodim02.png	Aleatoria	13,0962	10,7067	10,0122	9,5047	9,1140	8,7353	8,4688	8,2232
		Alg. Agrup.			8,8655	8,4655	8,2401	7,8179	7,6236	7,3757
6	kodim03.png	Aleatoria	11,1519	6,0136	5,6333	5,3304	5,1044	4,8801	4,7087	4,5274
		Alg. Agrup.			5,4736	4,7255	4,3791	4,1495	3,9417	3,7932
7	kodim04.png	Aleatoria	12,5972	9,7028	9,1528	8,8057	8,5076	8,2415	7,9853	7,7407
		Alg. Agrup.			9,0460	8,5066	8,0863	7,9554	7,6952	7,4890
8	kodim05.png	Aleatoria	64,5380	45,1289	42,5656	40,4833	39,0575	38,4246	37,2707	36,2560
		Alg. Agrup.			38,3257	36,7753	35,1467	32,8601	32,0934	31,3522
9	kodim06.png	Aleatoria	40,7274	28,5078	27,2197	25,8851	24,9790	24,3341	23,7750	23,0053
		Alg. Agrup.			25,6928	23,6758	22,5806	21,6829	21,1403	20,8854
10	kodim07.png	Aleatoria	12,0049	8,9810	8,3911	7,8882	7,5975	7,2908	6,9058	6,6232
		Alg. Agrup.			8,0397	7,2958	6,9070	6,4065	6,0712	5,8832
11	kodim08.png	Aleatoria	76,7571	59,4455	56,4683	54,5119	52,5959	51,6739	50,4149	49,3236
		Alg. Agrup.			51,2364	48,9279	47,0086	44,7776	44,0629	42,5230
12	kodim09.png	Aleatoria	12,1501	10,6621	9,8836	9,1709	8,7432	8,2825	7,9068	7,6530
		Alg. Agrup.			9,7378	7,7222	7,1541	6,7889	6,3490	6,1201
13	kodim10.png	Aleatoria	15,7615	9,3821	8,6171	8,1198	7,6969	7,3640	6,9131	6,7416
		Alg. Agrup.			7,5716	7,0066	6,4788	6,2432	5,8769	5,5699
14	kodim11.png	Aleatoria	31,4464	26,6310	25,3674	23,8756	22,9336	22,0077	21,2985	20,6118
		Alg. Agrup.			22,2799	21,1834	20,1359	19,4100	18,2519	17,7524
15	kodim12.png	Aleatoria	10,9331	8,4389	7,8308	7,2821	6,9472	6,6278	6,3164	6,0339
		Alg. Agrup.			7,8803	6,1076	5,7798	5,5343	5,3523	5,1186
16	kodim13.png	Aleatoria	130,5531	104,6189	100,2524	97,1580	94,3474	91,1242	89,0401	87,5509
		Alg. Agrup.			99,3585	94,1799	92,0457	89,6577	87,9057	84,9763
17	kodim14.png	Aleatoria	33,9268	26,5635	25,3973	24,3516	23,6238	22,8447	22,2294	21,4266
		Alg. Agrup.			24,7432	22,8179	21,9093	21,3386	20,7783	20,2929
18	kodim15.png	Aleatoria	18,4593	12,4565	11,6682	10,9833	10,5737	9,9828	9,6061	9,2799
		Alg. Agrup.			11,1019	10,2952	9,3997	8,9506	8,4834	8,2229
19	kodim16.png	Aleatoria	17,8092	10,7971	10,1311	9,8334	9,5160	9,2331	8,9759	8,7508
		Alg. Agrup.			9,9938	9,2116	8,8557	8,6631	8,2894	8,0593
20	kodim17.png	Aleatoria	19,3280	14,7129	13,3689	12,2891	11,5514	10,8964	10,1673	9,6345
		Alg. Agrup.			13,1560	11,9955	10,9541	9,8914	9,4099	9,1005
21	kodim18.png	Aleatoria	48,9941	40,2413	37,6849	35,9886	34,5268	33,0929	32,0344	31,0890
		Alg. Agrup.			37,9828	36,3066	34,5148	33,3061	32,5057	30,3316
22	kodim19.png	Aleatoria	26,8746	21,0772	20,2536	19,5243	18,9506	18,3363	17,8904	17,5149
		Alg. Agrup.			19,6389	16,9422	15,7134	14,8975	14,5052	14,2383
23	kodim20.png	Aleatoria	19,4638	15,2546	13,9015	12,9370	12,1035	11,3966	10,9789	10,3904
		Alg. Agrup.			13,7694	12,1114	11,0881	11,0421	10,6217	10,3527
24	kodim21.png	Aleatoria	36,4192	29,2493	27,6065	26,2379	25,4954	24,7902	23,8691	23,1755
		Alg. Agrup.			27,2035	25,8972	24,5885	23,7970	22,9971	22,5731
25	kodim22.png	Aleatoria	22,6036	19,3029	17,9566	17,1514	16,6849	16,0151	15,6130	15,0661
		Alg. Agrup.			17,1413	16,1142	15,7441	15,0420	14,6819	14,2294
26	kodim23.png	Aleatoria	7,9042	5,2309	4,6813	4,1814	3,8482	3,5711	3,3147	3,1215
		Alg. Agrup.			4,6429	4,1645	3,6296	3,4799	3,1830	3,0820
27	kodim24.png	Aleatoria	62,8695	47,7524	44,7312	42,5799	40,9410	39,1122	37,7051	36,3034
		Alg. Agrup.			44,1206	42,0020	40,3345	38,4134	37,3283	37,2365

TABLA B.1: MSE (TOTAL BLOQUES - AGRUP: ENERGÍA NORM. - DIST. CITY BLOCK - CON REASIGNACIÓN)

#	imagen	Asignación	DCT	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
1	lenna.bmp	Aleatoria	38,3125	39,5114	39,7917	39,9813	40,1591	40,2987	40,4446	40,5898
		Alg. Agrup.			39,8647	40,1482	40,3191	40,4530	40,6715	40,7711
2	baboon.bmp	Aleatoria	27,1982	28,3315	28,5615	28,7286	28,8742	28,9639	29,0422	29,1439
		Alg. Agrup.			29,0248	29,1574	29,2817	29,3897	29,5033	29,5861
3	barbara.bmp	Aleatoria	33,4709	37,6954	38,0436	38,3146	38,6139	38,7112	39,0850	39,1643
		Alg. Agrup.			38,4426	38,8986	40,3986	40,6190	40,6800	40,8655
4	kodim01.png	Aleatoria	31,1359	31,8609	32,0349	32,1615	32,2575	32,3435	32,4236	32,5332
		Alg. Agrup.			32,2841	32,4637	32,5547	32,6471	32,7561	32,8671
5	Kodim02.png	Aleatoria	36,9594	37,8343	38,1255	38,3514	38,5337	38,7180	38,8526	38,9804
		Alg. Agrup.			38,6538	38,8543	38,9715	39,1999	39,3092	39,4528
6	kodim03.png	Aleatoria	37,6573	40,3395	40,6232	40,8632	41,0514	41,2465	41,4018	41,5724
		Alg. Agrup.			40,7481	41,3863	41,7170	41,9508	42,1740	42,3407
7	kodim04.png	Aleatoria	37,1280	38,2618	38,5153	38,6831	38,8327	38,9708	39,1079	39,2430
		Alg. Agrup.			38,5663	38,8332	39,0533	39,1242	39,2686	39,3866
8	kodim05.png	Aleatoria	30,0326	31,5863	31,8402	32,0580	32,2138	32,2847	32,4171	32,5370
		Alg. Agrup.			32,2959	32,4752	32,6720	32,9641	33,0666	33,1681
9	kodim06.png	Aleatoria	32,0319	33,5812	33,7820	34,0003	34,1551	34,2686	34,3696	34,5125
		Alg. Agrup.			34,0327	34,3877	34,5934	34,7696	34,8797	34,9324
10	kodim07.png	Aleatoria	37,3372	38,5975	38,8926	39,1610	39,3241	39,5030	39,7387	39,9201
		Alg. Agrup.			39,0784	39,5001	39,7379	40,0646	40,2981	40,4346
11	kodim08.png	Aleatoria	29,2796	30,3896	30,6128	30,7659	30,9213	30,9981	31,1052	31,2003
		Alg. Agrup.			31,0350	31,2352	31,4090	31,6202	31,6901	31,8446
12	kodim09.png	Aleatoria	37,2850	37,8524	38,1817	38,5067	38,7141	38,9492	39,1508	39,2925
		Alg. Agrup.			38,2462	39,2534	39,5852	39,8128	40,1037	40,2632
13	kodim10.png	Aleatoria	36,1548	38,4078	38,7772	39,0354	39,2676	39,4597	39,7341	39,8432
		Alg. Agrup.			39,3389	39,6757	40,0159	40,1768	40,4393	40,6723
14	kodim11.png	Aleatoria	33,1551	33,8769	34,0880	34,3513	34,5261	34,7051	34,8473	34,9896
		Alg. Agrup.			34,6517	34,8708	35,0911	35,2505	35,5177	35,6382
15	kodim12.png	Aleatoria	37,7434	38,8680	39,1927	39,5082	39,7127	39,9171	40,1261	40,3248
		Alg. Agrup.			39,1654	40,2721	40,5117	40,7002	40,8454	41,0393
16	kodim13.png	Aleatoria	26,9729	27,9347	28,1199	28,2560	28,3835	28,5345	28,6349	28,7082
		Alg. Agrup.			28,1588	28,3912	28,4908	28,6049	28,6906	28,8378
17	kodim14.png	Aleatoria	32,8254	33,8879	34,0829	34,2655	34,3973	34,5430	34,6615	34,8213
		Alg. Agrup.			34,1963	34,5480	34,7245	34,8391	34,9547	35,0574
18	kodim15.png	Aleatoria	35,4686	37,1768	37,4608	37,7235	37,8885	38,1383	38,3053	38,4554
		Alg. Agrup.			37,6768	38,0045	38,3997	38,6123	38,8451	38,9806
19	kodim16.png	Aleatoria	35,6243	37,7977	38,0742	38,2037	38,3463	38,4773	38,6000	38,7104
		Alg. Agrup.			38,1335	38,4875	38,6586	38,7541	38,9456	39,0678
20	kodim17.png	Aleatoria	35,2689	36,4538	36,8698	37,2356	37,5045	37,7580	38,0587	38,2925
		Alg. Agrup.			36,9396	37,3406	37,7350	38,1782	38,3950	38,5401
21	kodim18.png	Aleatoria	31,2294	32,0841	32,3691	32,5692	32,7492	32,9335	33,0746	33,2047
		Alg. Agrup.			32,3349	32,5309	32,7507	32,9056	33,0112	33,3118
22	kodim19.png	Aleatoria	33,8374	34,8927	35,0658	35,2250	35,3546	35,4977	35,6046	35,6967
		Alg. Agrup.			35,1996	35,8411	36,1681	36,3997	36,5156	36,5962
23	kodim20.png	Aleatoria	35,2385	36,2968	36,7002	37,0125	37,3017	37,5630	37,7252	37,9645
		Alg. Agrup.			36,7417	37,2989	37,6822	37,7003	37,8689	37,9803
24	kodim21.png	Aleatoria	32,5175	33,4696	33,7207	33,9415	34,0662	34,1880	34,3524	34,4805
		Alg. Agrup.			33,7846	33,9983	34,2235	34,3656	34,5141	34,5949
25	kodim22.png	Aleatoria	34,5890	35,2746	35,5886	35,7878	35,9076	36,0855	36,1960	36,3508
		Alg. Agrup.			35,7904	36,0587	36,1596	36,3577	36,4630	36,5989
26	kodim23.png	Aleatoria	39,1522	40,9451	41,4271	41,9175	42,2783	42,6027	42,9263	43,1872
		Alg. Agrup.			41,4629	41,9351	42,5322	42,7151	43,1024	43,2425
27	kodim24.png	Aleatoria	30,1464	31,3409	31,6247	31,8388	32,0092	32,2077	32,3668	32,5313
		Alg. Agrup.			31,6844	31,8981	32,0740	32,2860	32,4104	32,4211

TABLA B.2: PSNR [dB] (TOTAL BLOQUES - AGRUP: ENERGÍA NORM. - DIST. CITY BLOCK - CON REASIGNACIÓN)

#	imagen	Asignación	DCT	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
1	lenna.bmp	Aleatoria	9,5903	7,2768	6,8219	6,5306	6,2686	6,0703	5,8697	5,6767
		Alg. Agrup.			6,7555	6,2005	5,9064	5,7475	5,6141	5,4666
2	baboon.bmp	Aleatoria	123,9530	95,4840	90,5578	87,1406	84,2669	82,5453	81,0699	79,1932
		Alg. Agrup.			80,0722	76,8956	74,9802	72,7358	70,9066	69,3275
3	barbara.bmp	Aleatoria	29,2411	11,0546	10,2029	9,5856	8,9473	8,7490	8,0275	7,8822
		Alg. Agrup.			8,5503	7,1158	6,0631	5,4849	5,1753	4,9078
4	kodim01.png	Aleatoria	50,0602	42,3633	40,6998	39,5307	38,6664	37,9076	37,2156	36,2880
		Alg. Agrup.			38,1987	36,7238	35,9756	35,2684	34,4126	33,8957
5	Kodim02.png	Aleatoria	13,0962	10,7067	10,0122	9,5047	9,1140	8,7353	8,4688	8,2232
		Alg. Agrup.			8,7963	8,3216	8,0829	7,8711	7,6531	7,4955
6	kodim03.png	Aleatoria	11,1519	6,0136	5,6333	5,3304	5,1044	4,8801	4,7087	4,5274
		Alg. Agrup.			5,0078	4,5250	4,2046	4,1164	3,9421	3,7061
7	kodim04.png	Aleatoria	12,5972	9,7028	9,1528	8,8057	8,5076	8,2415	7,9853	7,7407
		Alg. Agrup.			8,9566	8,2976	8,0576	7,7417	7,6207	7,4377
8	kodim05.png	Aleatoria	64,5380	45,1289	42,5656	40,4833	39,0575	38,4246	37,2707	36,2560
		Alg. Agrup.			36,9368	34,0453	32,2928	31,3810	30,3977	29,5373
9	kodim06.png	Aleatoria	40,7274	28,5078	27,2197	25,8851	24,9790	24,3341	23,7750	23,0053
		Alg. Agrup.			24,3710	24,1667	23,1828	21,8782	21,5249	20,9117
10	kodim07.png	Aleatoria	12,0049	8,9810	8,3911	7,8882	7,5975	7,2908	6,9058	6,6232
		Alg. Agrup.			8,0893	7,1299	6,8198	6,7357	6,4173	6,0792
11	kodim08.png	Aleatoria	76,7571	59,4455	56,4683	54,5119	52,5959	51,6739	50,4149	49,3236
		Alg. Agrup.			49,6847	46,4124	44,5525	42,9315	41,6654	40,8549
12	kodim09.png	Aleatoria	12,1501	10,6621	9,8836	9,1709	8,7432	8,2825	7,9068	7,6530
		Alg. Agrup.			8,2680	7,4154	6,8839	6,6534	6,4399	6,1595
13	kodim10.png	Aleatoria	15,7615	9,3821	8,6171	8,1198	7,6969	7,3640	6,9131	6,7416
		Alg. Agrup.			7,5135	6,7506	6,1930	5,9276	5,6986	5,5092
14	kodim11.png	Aleatoria	31,4464	26,6310	25,3674	23,8756	22,9336	22,0077	21,2985	20,6118
		Alg. Agrup.			22,1405	20,1665	19,6771	18,8099	17,9440	17,5374
15	kodim12.png	Aleatoria	10,9331	8,4389	7,8308	7,2821	6,9472	6,6278	6,3164	6,0339
		Alg. Agrup.			6,3029	5,8027	5,5868	5,4414	5,1689	4,9938
16	kodim13.png	Aleatoria	130,5531	104,6189	100,2524	97,1580	94,3474	91,1242	89,0401	87,5509
		Alg. Agrup.			96,5888	92,5655	89,4475	87,2972	85,7259	83,7867
17	kodim14.png	Aleatoria	33,9268	26,5635	25,3973	24,3516	23,6238	22,8447	22,2294	21,4266
		Alg. Agrup.			23,5460	22,0841	21,4027	20,6616	20,2073	19,6684
18	kodim15.png	Aleatoria	18,4593	12,4565	11,6682	10,9833	10,5737	9,9828	9,6061	9,2799
		Alg. Agrup.			10,7325	9,8946	9,2602	8,9544	8,6003	8,2003
19	kodim16.png	Aleatoria	17,8092	10,7971	10,1311	9,8334	9,5160	9,2331	8,9759	8,7508
		Alg. Agrup.			9,4135	9,0299	8,7058	8,4354	8,1326	7,9883
20	kodim17.png	Aleatoria	19,3280	14,7129	13,3689	12,2891	11,5514	10,8964	10,1673	9,6345
		Alg. Agrup.			12,3841	11,5168	10,7421	10,2144	9,6425	9,4138
21	kodim18.png	Aleatoria	48,9941	40,2413	37,6849	35,9886	34,5268	33,0929	32,0344	31,0890
		Alg. Agrup.			37,7786	35,4542	34,0008	32,1935	31,5901	30,0871
22	kodim19.png	Aleatoria	26,8746	21,0772	20,2536	19,5243	18,9506	18,3363	17,8904	17,5149
		Alg. Agrup.			17,5891	16,8123	14,9347	14,6411	14,1067	13,7718
23	kodim20.png	Aleatoria	19,4638	15,2546	13,9015	12,9370	12,1035	11,3966	10,9789	10,3904
		Alg. Agrup.			14,9127	13,1068	11,6972	11,6630	10,8315	10,7844
24	kodim21.png	Aleatoria	36,4192	29,2493	27,6065	26,2379	25,4954	24,7902	23,8691	23,1755
		Alg. Agrup.			26,6775	25,3242	24,3301	23,3816	22,8920	21,8234
25	kodim22.png	Aleatoria	22,6036	19,3029	17,9566	17,1514	16,6849	16,0151	15,6130	15,0661
		Alg. Agrup.			17,0989	15,7314	15,2603	14,6323	14,1260	14,0023
26	kodim23.png	Aleatoria	7,9042	5,2309	4,6813	4,1814	3,8482	3,5711	3,3147	3,1215
		Alg. Agrup.			4,5356	3,8966	3,6219	3,3142	3,1801	3,1203
27	kodim24.png	Aleatoria	62,8695	47,7524	44,7312	42,5799	40,9410	39,1122	37,7051	36,3034
		Alg. Agrup.			43,7937	41,6101	41,5856	39,9470	38,8807	37,4969

TABLA C.1: MSE (TOTAL BLOQUES - AGRUP: SUMA PESOS NORM. - DIST. CITY BLOCK - CON REASIGNACIÓN)

#	imagen	Asignación	DCT	1 KLT	2 KLT	3 KLT	4 KLT	5 KLT	6 KLT	7 KLT
1	lenna.bmp	Aleatoria	38,3125	39,5114	39,7917	39,9813	40,1591	40,2987	40,4446	40,5898
		Alg. Agrup.			39,8342	40,2066	40,4176	40,5360	40,6380	40,7536
2	baboon.bmp	Aleatoria	27,1982	28,3315	28,5615	28,7286	28,8742	28,9639	29,0422	29,1439
		Alg. Agrup.			29,0960	29,2718	29,3813	29,5133	29,6239	29,7217
3	barbara.bmp	Aleatoria	33,4709	37,6954	38,0436	38,3146	38,6139	38,7112	39,0850	39,1643
		Alg. Agrup.			38,8110	39,6086	40,3039	40,7391	40,9914	41,2220
4	kodim01.png	Aleatoria	31,1359	31,8609	32,0349	32,1615	32,2575	32,3435	32,4236	32,5332
		Alg. Agrup.			32,3103	32,4813	32,5707	32,6570	32,7636	32,8294
5	Kodim02.png	Aleatoria	36,9594	37,8343	38,1255	38,3514	38,5337	38,7180	38,8526	38,9804
		Alg. Agrup.			38,6878	38,9287	39,0551	39,1704	39,2924	39,3828
6	kodim03.png	Aleatoria	37,6573	40,3395	40,6232	40,8632	41,0514	41,2465	41,4018	41,5724
		Alg. Agrup.			41,1344	41,5746	41,8936	41,9856	42,1735	42,4417
7	kodim04.png	Aleatoria	37,1280	38,2618	38,5153	38,6831	38,8327	38,9708	39,1079	39,2430
		Alg. Agrup.			38,6094	38,9413	39,0687	39,2424	39,3108	39,4164
8	kodim05.png	Aleatoria	30,0326	31,5863	31,8402	32,0580	32,2138	32,2847	32,4171	32,5370
		Alg. Agrup.			32,4562	32,8102	33,0397	33,1641	33,3024	33,4271
9	kodim06.png	Aleatoria	32,0319	33,5812	33,7820	34,0003	34,1551	34,2686	34,3696	34,5125
		Alg. Agrup.			34,2621	34,2986	34,4791	34,7307	34,8014	34,9269
10	kodim07.png	Aleatoria	37,3372	38,5975	38,8926	39,1610	39,3241	39,5030	39,7387	39,9201
		Alg. Agrup.			39,0517	39,5999	39,7931	39,8470	40,0573	40,2923
11	kodim08.png	Aleatoria	29,2796	30,3896	30,6128	30,7659	30,9213	30,9981	31,1052	31,2003
		Alg. Agrup.			31,1686	31,4645	31,6421	31,8030	31,9330	32,0184
12	kodim09.png	Aleatoria	37,2850	37,8524	38,1817	38,5067	38,7141	38,9492	39,1508	39,2925
		Alg. Agrup.			38,9568	39,4295	39,7524	39,9003	40,0420	40,2354
13	kodim10.png	Aleatoria	36,1548	38,4078	38,7772	39,0354	39,2676	39,4597	39,7341	39,8432
		Alg. Agrup.			39,3724	39,8374	40,2118	40,4020	40,5731	40,7200
14	kodim11.png	Aleatoria	33,1551	33,8769	34,0880	34,3513	34,5261	34,7051	34,8473	34,9896
		Alg. Agrup.			34,6789	35,0845	35,1912	35,3869	35,5916	35,6911
15	kodim12.png	Aleatoria	37,7434	38,8680	39,1927	39,5082	39,7127	39,9171	40,1261	40,3248
		Alg. Agrup.			40,1354	40,4945	40,6592	40,7737	40,9968	41,1465
16	kodim13.png	Aleatoria	26,9729	27,9347	28,1199	28,2560	28,3835	28,5345	28,6349	28,7082
		Alg. Agrup.			28,2815	28,4663	28,6151	28,7208	28,7997	28,8991
17	kodim14.png	Aleatoria	32,8254	33,8879	34,0829	34,2655	34,3973	34,5430	34,6615	34,8213
		Alg. Agrup.			34,4116	34,6900	34,8261	34,9792	35,0757	35,1931
18	kodim15.png	Aleatoria	35,4686	37,1768	37,4608	37,7235	37,8885	38,1383	38,3053	38,4554
		Alg. Agrup.			37,8238	38,1768	38,4646	38,6105	38,7857	38,9925
19	kodim16.png	Aleatoria	35,6243	37,7977	38,0742	38,2037	38,3463	38,4773	38,6000	38,7104
		Alg. Agrup.			38,3933	38,5740	38,7327	38,8697	39,0285	39,1062
20	kodim17.png	Aleatoria	35,2689	36,4538	36,8698	37,2356	37,5045	37,7580	38,0587	38,2925
		Alg. Agrup.			37,2022	37,5175	37,8199	38,0387	38,2889	38,3931
21	kodim18.png	Aleatoria	31,2294	32,0841	32,3691	32,5692	32,7492	32,9335	33,0746	33,2047
		Alg. Agrup.			32,3583	32,6341	32,8159	33,0531	33,1353	33,3470
22	kodim19.png	Aleatoria	33,8374	34,8927	35,0658	35,2250	35,3546	35,4977	35,6046	35,6967
		Alg. Agrup.			35,6784	35,8745	36,3888	36,4751	36,6365	36,7409
23	kodim20.png	Aleatoria	35,2385	36,2968	36,7002	37,0125	37,3017	37,5630	37,7252	37,9645
		Alg. Agrup.			36,3952	36,9558	37,4500	37,4627	37,7839	37,8028
24	kodim21.png	Aleatoria	32,5175	33,4696	33,7207	33,9415	34,0662	34,1880	34,3524	34,4805
		Alg. Agrup.			33,8694	34,0954	34,2694	34,4421	34,5340	34,7416
25	kodim22.png	Aleatoria	34,5890	35,2746	35,5886	35,7878	35,9076	36,0855	36,1960	36,3508
		Alg. Agrup.			35,8011	36,1631	36,2952	36,4777	36,6306	36,6688
26	kodim23.png	Aleatoria	39,1522	40,9451	41,4271	41,9175	42,2783	42,6027	42,9263	43,1872
		Alg. Agrup.			41,5644	42,2240	42,5414	42,9271	43,1064	43,1888
27	kodim24.png	Aleatoria	30,1464	31,3409	31,6247	31,8388	32,0092	32,2077	32,3668	32,5313
		Alg. Agrup.			31,7167	31,9388	31,9414	32,1160	32,2335	32,3909

TABLA C.2: PSNR [dB] (TOTAL BLOQUES - AGRUP: SUMA PESOS NORM. - DIST. CITY BLOCK - CON REASIG.)