

Anexo A

Acrónimos

- **Cb:** Componente chroma blue-diferencia.
- **Cr:** Componente chroma red-diferencia.
- **DWT:** Discrete Wavelet Transformation.
- **FPS:** Frames Por Segundo.
- **JPEG:** Joint Photographic Experts Group.
- **MPEG:** Motion Picture Experts Group.
- **PSNR:** Peak Signal to Noise Ratio.
- **RLE:** Run Length Encoding
- **SPIHT:** Set Partitioning In Hierarchical Trees.
- **Y:** Componente luma.

Anexo B

MPEG-4

Una idea básica de compresión de vídeo consiste en emplear cualquier método para la compresión de imágenes, que se denominan INTRAFRAMES, aplicado a cada uno de los cuadros o frames. Sin embargo, los niveles de compresión que se pueden alcanzar de esta forma son limitados por el tratamiento diferenciado de cada cuadro.

Los métodos de compresión INTERFRAMES tratan de aprovechar las características de redundancia temporal debida a la similitud entre cuadros vecinos, junto a redundancias espaciales, espectrales y psicovisuales para obtener porcentajes de compresión mayores.

MPEG-4 trata más de agrupar los estándares ya existentes que de crear nuevos formatos (formato de formatos). Se trata de un metaformato que engloba tanto la posibilidad de almacenar imágenes y sonidos comprimidos, como la de añadir objetos sintéticos, interpretar y representar una composición de escena a partir de múltiples objetos.

Para que el ojo humano perciba la sensación de movimiento, las imágenes estáticas deben sucederse a una velocidad de entre 15 y 30 fps, que elimina cualquier tipo de “saltos” en la imagen. MPEG se aprovecha de las deficiencias de la vista, para eliminar información que el ojo humano no percibe, cuando la velocidad de movimiento es muy grande.

Para ello utiliza un método de compresión llamado interframe. Este no almacena las imágenes estáticas, sino los cambios que se producen entre una imagen y la siguiente. Para llevar a cabo esto, MPEG trabaja con tres tipos de cuadros o imágenes: I o Intraframes, es decir, imágenes completas, P o imágenes predecibles, que son las imágenes que muestran los cambios producidos en una imagen anterior, y B o imagen Bidireccional, que contiene referencias a las anterior y a la próxima imagen de la secuencia de vídeo.

B.1 Codificación Diferencial y Compensación de Movimiento en MPEG-4

Para obtener factores de compresión adecuados a la transmisión en tiempo real es necesario aprovechar la redundancia temporal existente entre las sucesivas imágenes. Téngase en cuenta que, dada la naturaleza de la mayor parte de escenas, gran parte de los objetos y elementos permanecen en la misma posición y con las mismas características de un fotograma a otro. Esto indica que es posible obtener una excelente aproximación de la información contenida en un fotograma aprovechando la información que disponemos del fotograma anterior.



Figura B.1: *Ejemplo de redundancia temporal.*

En la figura B.1 se muestran dos fotogramas consecutivos de una secuencia de vídeo. En este ejemplo, las diferencias sólo son apreciables cuando se realiza la resta entre las dos imágenes. En la figura B.2 se muestra el resultado de realizar esta diferencia. Sobre la imagen diferencia se ha calculado el valor absoluto y para hacer más evidentes la diferencia entre niveles se ha multiplicado por un factor 3, finalmente se ha representado en vídeo inverso.

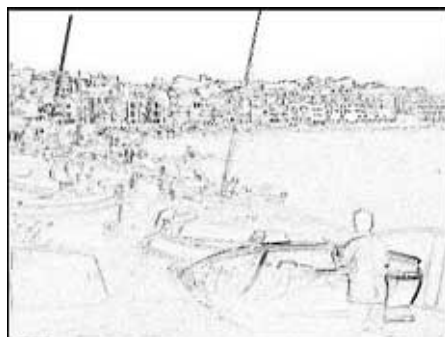


Figura B.2: *Diferencia entre imágenes consecutivas.*

La imagen resultante indica de forma clara el resultado se esperaba, ya que la mayor parte de los elementos están próximos al cero, por lo que se ha reducido la entropía de

forma notable, pudiendo codificar la imagen de forma eficientemente. Existen algunas zonas de la imagen en las que se observa que las diferencias aumentan de nivel y que se corresponden, aproximadamente, con los contornos de los objetos. Esto se debe al movimiento de la cámara, que está realizando un ligero movimiento horizontal y al de los propios objetos (el barquero está dando la vuelta a la barca).

La compensación de movimiento es una técnica de predicción temporal que intenta optimizar los resultados obtenidos por la codificación diferencial que acabamos de ver, analizando el movimiento que realizan los objetos dentro de la imagen y compensándolos.

Considerando las imágenes del ejemplo anterior, se observa que la mayor parte de diferencias entre las dos imágenes se deben al propio movimiento de la cámara, por lo que podrían reducirse con un ligero desplazamiento relativo entre ellas. En general, la codificación diferencial presenta problemas cuando se producen desplazamientos de los objetos dentro de la imagen o aparecen nuevos motivos que pueden ser debidos al movimiento de la cámara, el cambio de ángulo de visión (zoom) el desplazamiento real de los objetos o los cambios de plano. Para evitar estos errores se usa la técnica de compensación de movimiento.

En la figura B.3 se muestra un ejemplo de un desplazamiento de cámara en una imagen simplificada.

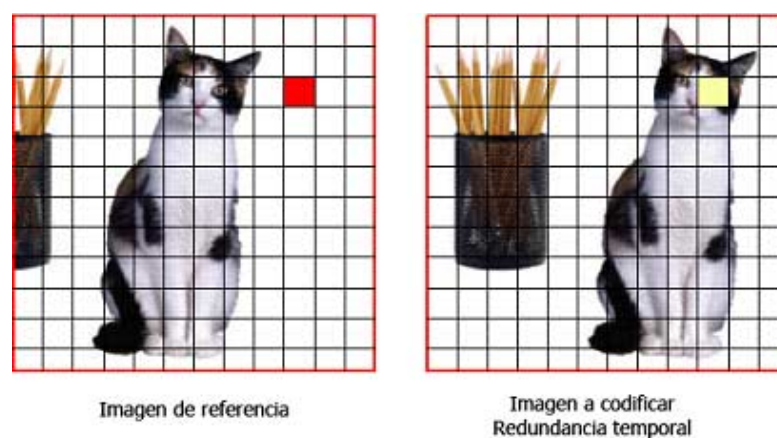


Figura B.3: *Ejemplo de desplazamiento de cámara en una imagen.*

La figura muestra cuál sería la zona de la imagen utilizada para predecir el bloque de interés mediante una codificación diferencial. Obviamente, el error de predicción es en este caso igual a la imagen que se desea predecir, por lo que la ganancia del codificador es nula.

La figura B.4 muestra el bloque que se utilizaría para la predicción si se usaran técnicas de compensación de movimiento. Es evidente, que, en este ejemplo, el error de predicción puede llegar a ser exactamente nulo, por lo que no resultaría necesario volver a codificar el bloque, únicamente se tendría que transmitir el vector de desplazamiento que se debe utilizar en la imagen de referencia para construir este fragmento de imagen.

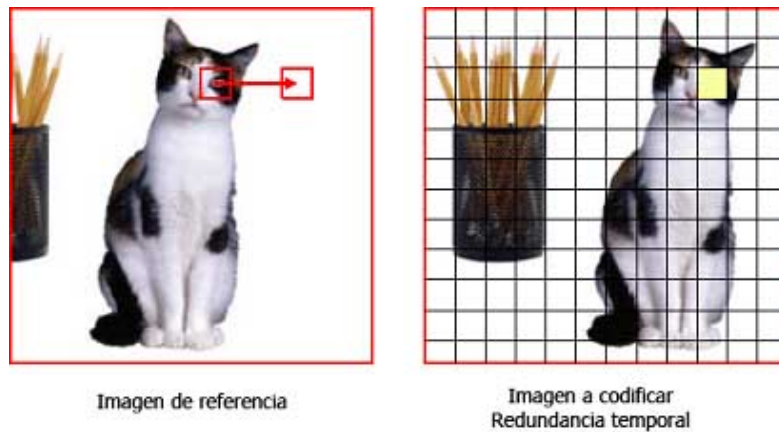


Figura B.4: *Ejemplo de desplazamiento de cámara en una imagen.*

En síntesis, el proceso para codificar un bloque de una imagen mediante la compensación de movimiento consiste en determinar qué zona o bloque de la imagen de referencia puede utilizarse de forma más eficiente para realizar la predicción del bloque actual. Una vez determinada esta región, se debe transmitir el vector de desplazamiento asociado (únicamente las dos componentes de desplazamiento horizontal y vertical) y el error que se produce al utilizar esta estimación de la imagen en sustitución de la imagen original. El decodificador, que se supone que de algún modo ya ha decodificado la imagen que utilizamos como referencia, puede recuperar la información del bloque actual utilizando el bloque desplazado de la imagen de referencia y añadiendo el error que se ha transmitido.

Anexo C

Set Partitioning In Hierarchical Trees (SPIHT)

El algoritmo SPIHT fue introducido por primera vez por Said y Pearlman [7] para la compresión de imágenes. Este algoritmo se puede ver como una extensión del algoritmo Embedded Zero-tree Wavelet (EZW) diseñado por J. M. Shapiro [6]. El algoritmo EZW supuso un importante avance en la codificación de imágenes dado que rompía con la tónica general de incremento en la eficiencia obtenida gracias a un incremento en la complejidad. Esta técnica no solamente era competitiva en cuanto a prestaciones de compresión comparada con la técnicas más complejas de compresión de imagen sino que además, era extremadamente rápida en cuanto a su ejecución y producía un flujo de bits embebido. Esto es muy importante dado que gracias a esta propiedad la codificación se podía parar en cualquier instante, controlándose de forma precisa la cantidad de información que se utilizaba en la codificación de la imagen. Siguiendo este trabajo, A. Said y W. A. Pearlman introdujeron el algoritmo SPIHT para la compresión de imágenes. El algoritmo SPIHT se aplica directamente a los coeficientes obtenidos realizando la transformada wavelet a la imagen.

Los principios de SPIHT son el ordenamiento parcial de los coeficientes transformados por magnitud con un algoritmo de ordenamiento basado en la partición en conjuntos, transmisión ordenada del plano de bits y explotación de la autosimilitud entre los diferentes niveles de la transformada. Siguiendo estos principios, el codificador siempre envía los bits más significativos al decodificador.

C.1 Transformada Wavelet

La aplicación de la transformada wavelet permite pasar las muestras en el dominio del tiempo a un dominio transformado donde resulta más eficiente su codificación. De esta forma se consigue una gran eficiencia en cuanto a operaciones y tiempo de ejecución.

Las Wavelets, funciones bases de la Transformada Wavelet, son generadas a partir

de una función Wavelet madre, mediante traslaciones y dilataciones. Estas funciones permiten reconstruir la señal original a través de la Transformada Wavelet inversa. La Transformada Wavelet no es solamente local en tiempo, sino también en frecuencia. Dentro de los usos de esta poderosa herramienta podemos nombrar, además del análisis local de señales no estacionarias, el análisis de señales electrocardiográficas, sísmicas, de sonido, de radar, así como también es utilizada para la compresión y procesamiento de imágenes y reconocimiento de patrones.

En una imagen cualquiera, existe una relación directa entre los colores que esta presenta en cada pixel y la distribución del espectro en frecuencia. De esta manera, se sabe que las variaciones suaves en el color se corresponden con variaciones de baja frecuencia y las variaciones fuertes de color se corresponden con variaciones de alta frecuencia. La transformación del espacio de intensidades de color al espacio frecuencial se puede llevar a cabo de muchos modos. Uno de los posibles es la DCT, otra de las posibles técnicas es la descomposición de una imagen por medio de la Transformada Discreta Wavelet o Discrete Wavelet Transform (DWT).

La forma general de una transformada wavelet unidimensional (1-D) se puede apreciar en la Figura C.1

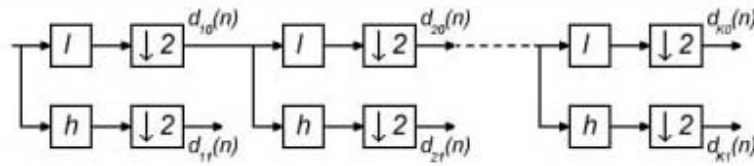


Figura C.1: Descomposición de una wavelet de 1-D, de K niveles.

Aquí una señal pasa a través de un filtro pasa bajos y de otro pasa altos, denotados por l y h respectivamente, y luego se submuestra en un factor de dos - esto causa un proceso con pérdida-, logrando una transformada de un nivel. Se pueden realizar múltiples niveles repitiendo los procesos de filtrado y eliminación (submuestreo), únicamente sobre la rama del filtro pasa bajos. Este proceso se lleva a cabo para un número K finito de niveles, obteniendo los coeficientes wavelet: $d_{10}(n)$, $i \dots 1, \dots, K$ y $d_{K0}(n)$. Cuando no se necesita tener conocimiento de la escala o de la frecuencia, el conjunto entero de coeficientes wavelet se expresa como $w(n)$. La transformada wavelet de 1-D puede extenderse a una transformada wavelet bidimensional (2-D) utilizando filtros wavelet separables. Con los filtros separables, la transformada 2-D puede calcularse aplicando una transformada 1-D a cada fila de los datos de entrada, y luego aplicándola a cada columna de los datos de entrada.

Así, como la transformada wavelet se utiliza para separar una imagen en varias clases de importancia, la transformada inversa o antitransformada se utiliza para recomponer las distintas clases de datos, y de este modo poder reconstruir la imagen. Aquí, también se utiliza un filtro pasa bajos y otro pasa altos, pero el filtrado se realiza en forma

opuesta. Es decir, se comienza desde el nivel más alto, aplicando primero los filtros por columna y luego por fila, hasta llegar al primer nivel.

La DWT aplicada a imágenes proporciona una matriz de coeficientes, conocidos como coeficientes wavelet. Si a una imagen le aplicamos la DWT obtenemos cuatro tipos de coeficientes (ver Figura C.2): aproximaciones (LL), detalles horizontales (HL), detalles verticales (LH) y detalles diagonales (HH). La aproximación (LL) contiene la mayor parte de la energía de la imagen, es decir, la información más importante, mientras que los detalles tienen valores próximos a cero.

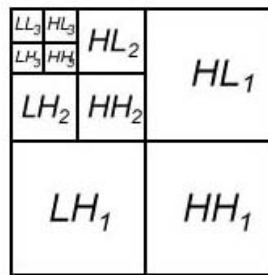


Figura C.2: Esquema de la organización de los coeficientes wavelet.

La elección de las wavelets analizadoras juega un papel muy importante en los resultados finales. Entre las características más importantes a tener en cuenta se encuentran: soporte compacto, simetría, etc. Las wavelets biortogonales son las más eficientes para un posterior proceso de compresión, y en particular, aquellas con pocos coeficientes, ya que el coste de obtención de los coeficientes se incrementa con su número.

Generalmente, la energía de las imágenes se concentra en las frecuencias bajas. Una imagen tiene un espectro que se reduce con el incremento de las frecuencias. Estas propiedades de las imágenes quedan reflejadas en la transformada wavelet discreta de la imagen. Los niveles más bajos de compresión se corresponden con las bandas de alta frecuencia. En particular, el primer nivel representa la banda de más alta frecuencia y el nivel más fino de resolución. A la inversa, el último nivel (n) de descomposición corresponde con la banda de frecuencia más baja y la resolución más tosca. Así, al desplazarse de los niveles más altos a los más bajos, o sea, de baja resolución a alta resolución, se observa una disminución de la energía contenida en las subbandas recorridas.

Si los coeficientes wavelet obtenidos por medio de la transformada wavelet discreta (DWT) para un nivel concreto poseen pequeñas magnitudes (valores próximos a cero), se espera que esos coeficientes wavelet estén en los primeros niveles de descomposición. El aumento del nivel de descomposición wavelet produce unos coeficientes con mayores magnitudes. Adicionalmente, se puede comprobar cómo existen similitudes espaciales a través de las subbandas.

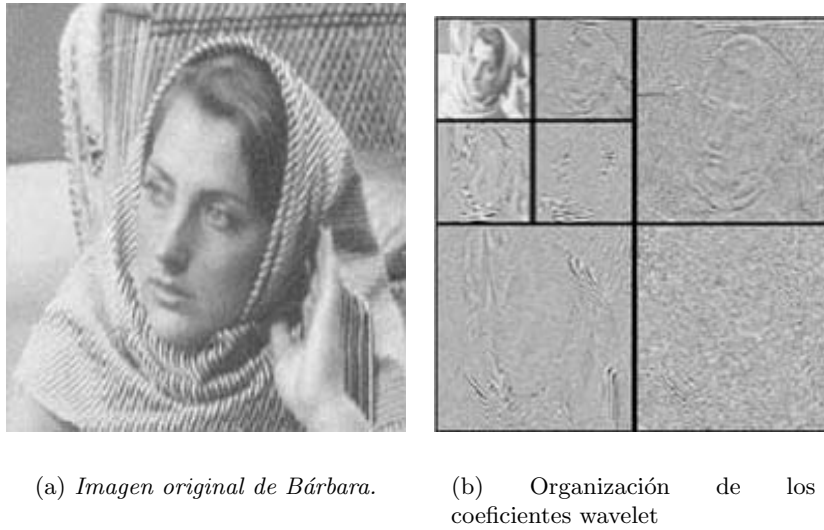


Figura C.3: *Ejemplo de la transformada wavelet.*

En la Figura C.3 se puede observar los contornos de Bárbara (base de imágenes para investigación científica) en los distintos niveles y cómo son más bastos en el primer nivel de descomposición, además de cierta similitud entre los distintos niveles. El cálculo de la transformada wavelet para todas las posibles escalas supone una gran cantidad de información. Escoger solo aquellas escalas y posiciones que resulten interesantes para ciertos estudios es una tarea difícil. Si se escogen aquellas escalas y posiciones basadas en potencias de dos, los resultados serán más eficaces. Este análisis se denomina DWT.

Para muchas señales la información más importante se encuentra en las frecuencias bajas, mientras que en las altas frecuencias se encuentran los detalles o matices de la señal. Por ejemplo, en el caso de la voz humana, si eliminamos los componentes con altas frecuencias, la voz suena diferente pero se sigue entendiendo su mensaje. En cambio, si lo que se elimina son las componentes de bajas frecuencias, el mensaje se vuelve irreconocible. Por eso el análisis wavelet permite descomponer la señal en aproximaciones y detalles, a éste proceso se le conoce con el nombre de análisis. Este filtrado nos proporciona el doble de datos de los que son necesarios, este problema se soluciona con la operación de downsampling. La transformada wavelet discreta (DWT) se emplea para obtener una nueva representación de la imagen, más apropiada para el proceso de compresión.

C.2 Algoritmo de ordenamiento de partición de grupos

Las mismas reglas de partición se definen tanto en el codificador como en el decodificador. Se dice que un conjunto de coeficientes subbanda c_i en el subconjunto T es significativo para una profundidad de bit n si $\max_{i \in T} \{|c_i|\} \geq 2^n$, en caso contrario se dice que es no significativo. Si el subconjunto es no significativo, se manda un cero

al decodificador. Si es significativo, se manda un uno al decodificador y después el subconjunto es dividido de acuerdo con el árbol de ordenamiento (ver Figura C.4) hasta que todos los grupos significativos son únicamente puntos significativos. En esta fase de la comunicación, llamada etapa de ordenamiento, los índices de los coeficientes son distribuidos dentro de tres listas, la lista de puntos no significativos (LIP), la lista de grupos no significativos (LIS) y la lista de puntos significativos (LSP). En esta etapa, sólo los bits relacionados con las entradas en LSP y las salidas binarias de los tests de magnitud son transmitidos al decodificador.

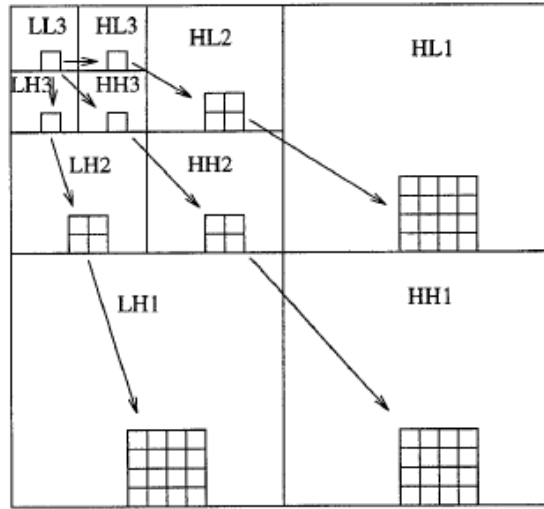


Figura C.4: Coeficientes wavelet organizados en árboles jerárquicos.

C.3 Etapa de refinado

Después de la etapa de ordenamiento, se toman los coeficientes significativos para el umbral 2^n y se envía al decodificador el n -ésimo bit mas significativo de cada uno de los coeficientes que se encontraron relevantes con un umbral superior. Transmitiendo el flujo de bits con este ordenamiento en el plano de bits, siempre se transmiten los bits más significativos que quedan por transmitir. El boceto del proceso completo de codificación sería el siguiente:

1. Inicialización: LSP está vacía. Establecimiento de las raíces de los árboles de nivel superior en LIP y en LIS. Establecimiento del umbral 2^n con $n = \lfloor \log_2 (\max_i |c_i|) \rfloor$.
2. Etapa de ordenamiento: Usando el algoritmo de ordenamiento de partición en grupos, distribuir los índices de los coeficientes en las listas LIP, LIS y LSP de forma apropiada.
3. Etapa de refinado: Para cada entrada en LSP que haya sido significativa para umbrales mayores, mandar el n -ésimo bit más significativo al decodificador.

4. Decrementar n en uno y volver al paso 2 hasta que el objetivo que se desee cumplir (un determinado error o una tasa de transmisión determinada) se alcance.

C.4 Ejemplo del funcionamiento del algoritmo

Se va a considerar un ejemplo simple para mostrar y así entender mejor el funcionamiento del algoritmo. Se va a aplicar un paso del algoritmo a la descomposición wavelet en 3 niveles de una imagen de 8x8 muestras produce 64 coeficientes distribuidos que se muestra en la Figura C.5. Todo el procedimiento de codificación se muestra en la Tabla C.1. A continuación se exponen algunas definiciones y explicaciones importantes para interpretar la Tabla C.1.

	0	1	2	3	4	5	6	7
0	63	-34	49	10	7	13	-12	7
1	-31	23	14	-13	3	4	6	-1
2	15	14	3	-12	5	-7	3	9
3	-9	-7	-14	8	4	-2	3	2
4	-5	9	-1	47	4	6	-2	2
5	3	0	-3	2	3	-2	0	4
6	2	-3	6	-4	3	6	3	6
7	5	11	5	6	0	3	-4	4

Figura C.5: Coeficientes wavelet de una imagen utilizada como ejemplo. Los números fuera de la caja indican las coordenadas usadas.

LIS Contiene grupos de coeficientes de la transformada wavelet que son definidos por las estructuras de árbol, y que se ha encontrado que tienen una magnitud más pequeña que un umbral (son no significativos). Los grupos son designados por el coeficiente raíz pero lo excluyen. Estos árboles tienen al menos dos elementos.

LIP Contiene coeficientes individuales cuya magnitud es más pequeña que el umbral.

LSP Contiene los coeficientes con magnitud mayor que el umbral (los coeficientes significativos).

O(i,j) En las estructuras de árbol, es el conjunto de descendientes directos de un nodo definido por la localización (i,j).

D(i,j) Conjunto de descendientes del nodo definido por la localización (i,j).

L(i,j) Conjunto definido por $L(i,j) = D(i,j) - O(i,j)$.

Las siguientes referencias se refieren a las entradas en la Tabla C.1 :

- (1) Tenemos las condiciones iniciales de SPIHT. El coeficiente de mayor magnitud es 63, así que el umbral se establece en 32. La notación (i,j)A o (i,j)B indica que la entrada a LIS es de tipo 'A' o 'B' respectivamente. Notar la duplicación de los coeficientes en las listas. El coeficiente (0,0) no es considerado raíz.
- (2) SPIHT empieza codificando la importancia de los píxeles individuales de LIP (1 cuando es significativo y 0 cuando no lo es). Cuando un bit es significativo se mueve a LSP y su signo es también codificado. Se usa la notación 1+ y 1- para indicar cuando un bit significativo está seguido de un bit signo.
- (3) Después de comprobar los píxeles se empieza con los conjuntos, siguiendo las entradas en LIS (las entradas activas están en negrita). En este ejemplo D(0,1) es el conjunto de 20 coeficientes $\{(0,2), (0,3), (1,2), (1,3), (0,4), (0,5), (0,6), (0,7), (1,4), (1,5), (1,6), (1,7), (2,4), (2,5), (2,6), (2,7), (3,4), (3,5), (3,6), (3,7)\}$. Al ser D(0,1) significativo, los siguientes en comprobarse son sus cuatro hijos $\{(0,2), (0,3), (1,2), (1,3)\}$.
- (4) Después de que todos los hijos sean comprobados, (0,1) es movido al final de LIS y el tipo cambia de 'A' a 'B', es decir, que la nueva entrada de LIS cambia de D(0,1) a L(0,1).
- (5) Se aplica el mismo procedimiento que en (3) y (4) al conjunto D(1,0). Notar que aunque el hijo de (0,1) es no significante, D(0,1) es significativo porque L(0,1) es significativo.
- (6) Desde que D(1,1) es significativo, no se necesita tomar ninguna acción. El algoritmo mueve el próximo elemento a LIS.
- (7) El próximo elemento de LIS, (0,1), es de tipo 'B', y por eso se comprueba L(0,1). Notar que la coordenada (0,1) se mueve del principio de LIS en este paso. Esta es ahora comprobada otra vez, pero con otra interpretación del algoritmo.
- (8) Lo mismo que arriba, pero L(0,1) es significativa, así que el conjunto es dividido en D(2, 0), D(2, 1), D(3, 0), y D(3, 1), y las correspondientes entradas son añadidas a LIS. Al mismo tiempo, la entrada (1,0)B es movida de LIS.
- (9) El algoritmo permanece evaluando los conjuntos de entradas tal como aparecen en LIS.
- (10) Cada nueva entrada es tratada como en los casos anteriores. En este caso se comprueba el hijo de (2,1).

Paso	Pixel o Grupo probado	Bit de Salida	Acción	Lista de Control
(1)				LIS = $\{(0,1)A, (1,0)A, (1,1)A\}$ LIP = $\{(0,0), (0,1), (1,0), (1,1)\}$ LSP = 0.
(2)	(0, 0)	1+	(0,0) to LSP	LIP = $\{(0,1), (1,0), (1,1)\}$ LSP = $\{(0,0)\}$
	(0, 1)	1-	(0,1) to LSP	LIP = $\{(1,0), (1,1)\}$ LSP = $\{(0,0), (0,1)\}$
	(1, 0)	0	none	
	(1, 1)	0	none	
(3)	D(0, 1)	1	test offspring	LIS = $\{(\mathbf{0},1)A, (1,0)A, (1,1)A\}$
	(0, 2)	1+	(0,2) to LSP	LSP = $\{(0,0), (0,1), (0,2)\}$
	(0, 3)	0	(0,3) to LIP	LIP = $\{(1,0), (1,1), (0,3)\}$
	(1, 2)	0	(1,2) to LIP	LIP = $\{(1,0), (1,1), (0,3), (1,2)\}$
	(1, 3)	0	(1,3) to LIP	LIP = $\{(1,0), (1,1), (0,3), (1,2), (1,3)\}$
(4)			type changes	LIS = $\{(1,0)A, (1,1)A, (0,1)B\}$
(5)	D(1, 0)	1	test offspring	LIS = $\{(\mathbf{1},0)A, (1,1)A, (0,1)B\}$
	(2, 0)	0	(2,0) to LIP	LIP = $\{(1,0), (1,1), (0,3), (1,2), (1,3), (2,0)\}$
	(2, 1)	0	(2,1) to LIP	LIP = $\{(1,0), (1,1), (0,3), (1,2), (1,3), (2,0), (2,1)\}$
	(3, 0)	0	(3,0) to LIP	LIP = $\{(1,0), (1,1), (0,3), (1,2), (1,3), (2,0), (2,1), (3,0)\}$
	(3, 1)	0	(3,1) to LIP	LIP = $\{(1,0), (1,1), (0,3), (1,2), (1,3), (2,0), (2,1), (3,0), (3,1)\}$
			type changes	LIS = $\{(1,1)A, (0,1)B, (1,0)B\}$
(6)	D(1, 1)	0	none	LIS = $\{(\mathbf{1},1)A, (0,1)B, (1,0)B\}$
(7)	L(0, 1)	0	none	LIS = $\{(1,1)A, (\mathbf{0},1)B, (1,0)B\}$
(8)	L(1, 0)	1	add new sets	LIS = $\{(1,1)A, (0,1)B, (2,0)A, (2,1)A, (3,0)A, (3,1)A\}$
(9)	D(2, 0)	0	none	LIS = $\{(1,1)A, (0,1)B, (\mathbf{2},0)A, (2,1)A, (3,0)A, (3,1)A\}$
(10)	D(2, 1)	1	test offspring	LIS = $\{(1,1)A, (0,1)B, (2,0)A, (\mathbf{2},1)A, (3,0)A, (3,1)A\}$
	(4, 2)	0	(4,2) to LIP	LIP = $\{(1,0), (1,1), (0,3), (1,2), (1,3), (2,0), (2,1), (3,0), (3,1), (4,2)\}$
	(4, 3)	1+	(4,3) to LSP	LSP = $\{(0,0), (0,1), (0,2), (4,3)\}$
	(5, 2)	0	(5,2) to LIP	LIP = $\{(1,0), (1,1), (0,3), (1,2), (1,3), (2,0), (2,1), (3,0), (3,1), (4,2), (5,2)\}$
	(5, 3)	0	(5,3) to LIP	LIP = $\{(1,0), (1,1), (0,3), (1,2), (1,3), (2,0), (2,1), (3,0), (3,1), (4,2), (5,2), (5,3)\}$
(11)	(2,1)		removed	LIS = $\{(1,1)A, (0,1)B, (2,0)A, (3,0)A, (3,1)A\}$
(12)	D(3, 0)	0	none	LIS = $\{(1,1)A, (0,1)B, (2,0)A, (3,0)A, (3,1)A\}$
	D(3, 1)	0	none	LIS = $\{(1,1)A, (0,1)B, (2,0)A, (\mathbf{3},0)A, (3,1)A\}$
(13)				LIS = $\{(1,1)A, (0,1)B, (2,0)A, (3,0)A, (\mathbf{3},1)A\}$ LIP = $\{(1,0), (1,1), (0,3), (1,2), (1,3), (2,0), (2,1), (3,0), (3,1), (4,2), (5,2), (5,3)\}$ LSP = $\{(0,0), (0,1), (0,2), (4,3)\}$

Tabla C.1: Ejemplo de codificación SPIHT.

- (11) En este caso, porque $L(2,1) = 0$ (sólo tiene descendientes hijos), la entrada (2,1)A es movida de LIS.
- (12) Finalmente, las dos últimas entradas de LIS corresponden a un conjunto insignificante, y no se toma ninguna acción. La tapa de ordenamiento termina después de que la última entrada de LIS sea comprobada.
- (13) La lista final en etapa de ordenamiento forma la lista inicial en el siguiente etapa de ordenamiento, con un umbral de 16.

Sin utilizar ningún otro forma de codificación, SPIHT usa 29 bits en este primer paso.

Anexo D

JPEG2000

El estándar JPEG2000 es un sistema de compresión y descompresión de imágenes (de tonos de gris y de color) digitales, creado en el año 2000 por el JPEG (Joint Photographic Experts Group). Este estándar está publicado como un ISO/IEC estándar y consta de varias partes, empezando por la Parte 1, que define los requerimientos mínimos necesarios para la compresión/descompresión de las imágenes (codificación, sintaxis del codestream, . . .). La primera parte sería lo mínimo necesario para cualquier implementación. Las partes sucesivas van añadiendo funcionalidad extra. En el momento de la creación del presente documento, las partes existentes son las que aparecen en la tabla D.1.

D.1 Características

JPEG2000 presenta las siguientes características:

- **Compresión de imágenes continuous-tone y bi-level:** El estándar es capaz de comprimir imágenes de tono continuo (continuous-tone) y binarias (bi-levels). Permite además una compresión de imágenes con un rango dinámico de profundidad de bits para cada componente, desde 1 bit, con una calidad comparable

Parte	Descripción
1	Sistema base
2	Extensión. Añade funcionalidad al sistema base
3	JPEG 2000 para secuencias de vídeo
4	Conformance
5	Software de referencia (implementaciones en C y Java)
6	Formato de archivo compuesto
8	JPSEC, para habilitar opciones de seguridad
9	JPIP, protocolo para la transmisión progresiva de imágenes
10	JP3D, para imágenes tridimensionales
11	JPWL, para aplicaciones wireless
12	ISO Base Media File Format (common with MPEG-4)

Tabla D.1: Partes del estándar JPEG2000.

al estándar ITU-T G4, hasta los 32 bits.

- **Transmisión progresiva de imágenes:** Ofrece una progresión en calidad, en resolución, en posición espacial o en componente de imagen, lo cual es muy apropiado para la transmisión sobre enlaces de comunicación lentos. Estas progresiones, que se consiguen ordenando adecuadamente el codestream, pueden coexistir entremezcladas en una misma imagen. En el caso del presente proyecto, será necesario tener en cuenta esto, pero no será empleado.
- **Posibilidad de compresión con o sin pérdida:** Gracias a la inclusión de una transformada wavelet reversible, el estándar JPEG2000 ofrece esta posibilidad.
- **Acceso aleatorio al codestream de la imagen así como posibilidad de procesamiento del mismo sin necesidad de descompresión:** Los codestreams ofrecen diversos mecanismos para soportar el acceso aleatorio en función de una región de interés. Además, permiten la rotación, la traslación y el escalado, entre otros, de la imagen que definen, sin necesidad de descompresión.
- **Robustez ante errores en los datos:** JPEG2000 define una serie de mecanismos (marcadores de sincronización, etc.) para permitir la detección de errores de los datos. Por ejemplo, las aplicaciones que hacen uso de canales de comunicación wireless harían uso de esta característica.
- **Arquitectura abierta:** La arquitectura del estándar es abierta, de forma que se define la base, que sería la Parte 1, a la cual se le va añadiendo la funcionalidad que se requiera para aplicaciones específicas.
- **Mejor rendimiento a bajos bit-rates:** JPEG2000 ofrece un rendimiento superior a bajos bit-rates con respecto al resto de los estándares. Las aplicaciones de transmisión de imágenes a través de la red se benefician de esta característica.
- **Posibilidad de definir regiones de interés (ROI):** A menudo existen partes de una imagen que son más importantes que otras. El estándar permite definir ciertas ROIs en la imagen con el fin de que sean codificadas y transmitidas con mayor prioridad, mejor calidad y menor distorsión que el resto.

D.2 Características técnicas

El objetivo de JPEG 2000 es no sólo mejorar el rendimiento de compresión JPEG, sino también añadir (o mejorar) las características tales como la escalabilidad y la capacidad de edición. Tanto las tasas de compresión muy baja y muy alta son compatibles con JPEG 2000. De hecho, la capacidad del diseño para manejar una gama muy grande de las tasas de bits es uno de los puntos fuertes de JPEG 2000. Por ejemplo, para reducir el número de bits para una imagen debajo de cierta cantidad, lo recomendable en el primer

estándar JPEG es reducir la resolución de la imagen de entrada antes de codificarlo. Eso es innecesario cuando se utiliza JPEG 2000, porque ya lo hace automáticamente a través de su estructura de descomposición multiresolución. En las secciones siguientes se describe el algoritmo de JPEG 2000.

- **Transformación de color.** Inicialmente, las imágenes tienen que ser transformadas a partir del RGB del espacio de color a otro espacio de color, dando lugar a tres componentes que se manejan por separado. Hay dos opciones posibles:
 - Transformación de color irreversible (TIC) utiliza el conocido YCBCR espacio de color. Se llama “irreversible” porque tiene que ser aplicado en el punto flotante o en un punto fijo y provoca errores de redondeo.
 - Transformación de color reversible (ECA) utiliza una versión modificada del espacio de color YUV que no introduce errores de cuantificación, por lo que es totalmente reversible.
- **Teja.** Después de la transformación del color, la imagen se divide en los llamados cuadros, regiones rectangulares de la imagen que se transforman y se codifican por separado. Pueden ser de cualquier tamaño, y también es posible considerar la imagen completa como una sola baldosa. Una vez que el tamaño que se eligen todas las fichas que tienen el mismo tamaño (salvo, opcionalmente, los de las fronteras derecho e inferior). Al dividir la imagen en azulejos tiene la ventaja de que el decodificador necesitará menos memoria para decodificar la imagen y se puede optar por decodificar sólo cuadros seleccionados para lograr una decodificación parcial de la imagen. La desventaja de este enfoque es que la calidad de la imagen disminuye. El uso de muchos cuadros puede crear un efecto de bloqueo similar al viejo JPEG 1992 estándar.
- **Transformada wavelet.** JPEG 2000 utiliza dos transformaciones wavelet diferentes:
 - Irreversible: la transformada wavelet CDF 9/7. Se dice que es “irreversible” porque introduce ruido de cuantificación que depende de la precisión del decodificador.
 - Reversible: una versión redondeada de la transformada wavelet biorthogonal CDF 5/3. Utiliza solamente coeficientes enteros, por lo que la salida no requiere redondeo (cuantificación) y para que no se genere ningún ruido de cuantificación. Se utiliza en la codificación sin pérdidas. La transformada Wavelet es ejecutada por el régimen de elevación o por convolución.
- **Cuantización.** Después de la transformada wavelet, los coeficientes son escalares cuantificados para reducir la cantidad de bits para que los represente, a expensas de una pérdida de calidad. El resultado es un conjunto de números enteros que tienen

que ser codificados bit a bit. El parámetro que se puede cambiar para establecer la calidad final es el paso de cuantificación: cuanto mayor es el paso, mayor es la compresión y la pérdida de calidad. Con un paso de cuantificación que es igual a 1, no se realiza la cuantización (se utiliza en la compresión sin pérdida).

- **Codificación.** El resultado del proceso anterior es una colección de sub-bandas que representan a diversas escalas de aproximación. Una sub-banda es un conjunto de coeficientes, números reales que representan los aspectos de la imagen asociada a un rango de frecuencias determinadas, así como un ámbito espacial de la imagen. Las sub-bandas cuantizadas se dividen aún más en precintos, regiones rectangulares en el dominio wavelet. A su vez, los precintos se dividen aún más en los bloques de código. Los bloques de código se encuentran en un subgrupo de una sola banda con tamaños iguales, excepto los situados en los bordes de la imagen. El codificador tiene que codificar los bits de todos los coeficientes cuantizados de un bloque de código, empezando por los bits más significativos y progresando a bits menos significativos de un proceso de régimen de EBCOT (codificación en bloque con truncamiento óptimo). En este proceso de codificación, cada plano de bits del bloque de código se codifican en tres pases de codificación, en primer lugar los bits de codificación (y signos) de los coeficientes insignificantes con los vecinos importantes, entonces se pasa al refinamiento de bits con coeficientes significativos y, por último, sin coeficientes de vecinos importantes. + Los bits de codificación seleccionados por estos pases, a continuación, se codifican con un codificador aritmético binario, (el codificador binario MQ). El contexto de un coeficiente está formado por el estado de sus nueve vecinos en el bloque de código. El resultado es un flujo de bits que se divide en paquetes. Los que contienen bits menos significativos se pueden descartar para lograr menores tasas de bits y una mayor distorsión. Los paquetes de todas las sub-bandas se recogen en las llamadas capas de modo. El estándar JPEG 2000 no define cuántos paquetes contendrá una capa, pero en general un codec trata de construir capas de tal forma que la imagen calidad aumente monótonamente con cada capa, y la distorsión de la imagen se reduzca de una capa a otra. Así, las capas definen la progresión de la calidad de imagen dentro de la secuencia de código. El problema ahora es encontrar la longitud de paquete óptimo para todos los bloques de código que minimizen la distorsión global de tal manera que la tasa de bits de destino generada es igual a la tasa de bits demandada. Mientras que el estándar no define un procedimiento sobre la forma de optimizar la tasa de distorsión, el esquema general se da en uno de sus apéndices: Por cada bit codificado por el codificador EBCOT, la mejora de la calidad de imagen, definida como error cuadrático medio, puede ser aplicado por un algoritmo de búsqueda fácil de mesa. Esto forma para cada bloque de código un gráfico en el plano, dando una calidad de imagen en toda su longitud de bits. La selección óptima de los puntos de truncamiento, vienen dados por la definición

de las pendientes críticas de estas curvas. Este método puede ser visto como una aplicación especial del método de multiplicadores de Lagrange que se utiliza para problemas de optimización con restricciones. Los paquetes pueden ser reordenados casi arbitrariamente en el formato JPEG 2000. Las imágenes ya codificadas pueden ser enviadas a través de redes con velocidades arbitrarias mediante una orden de codificación de capa progresiva. Por otra parte, los componentes del color se pueden mover de nuevo en el flujo de bits: resoluciones más bajas (correspondientes a los sub-bandas de baja frecuencia) pueden ser enviadas en primer lugar para la imagen de vista previa. Por último, la navegación espacial de las imágenes grandes es posible a través de la teja apropiada y/o selección de la partición. Todas estas operaciones no requieren volver a codificar, excepto las operaciones de copia de bytes.

- **Rendimiento.** Comparado con el anterior estándar JPEG, JPEG 2000 ofrece una ganancia de compresión típica en el rango de 20%, dependiendo de las características de la imagen. Imágenes de mayor resolución tienden a beneficiarse más de su proceso de compresión. Imágenes buenas para aplicaciones JPEG 2000 las de gran tamaño y/o con bordes de bajo contraste, por ejemplo, las imágenes médicas.

Anexo E

Selección Wavelet madre para los modos de barrido

Para la obtención de unos mejores resultados en la compresión mediante SPIHT, es necesario seleccionar adecuadamente la wavelet madre utilizada. Por lo tanto, en este Anexo, se va a estudiar para las imágenes que se van a comprimir en esta aplicación, cuál es la wavelet madre más adecuada. También se estudiarán los distintos niveles de compresión, que en el caso estudiado, está limitado por 5, ya que la anchura de franja que tenemos es de 32 píxeles.

Primero, se va a seleccionar una wavelet madre. Para ello, se ha comprimido un vídeo formado por los distintos modos de barrido para distintas ratios de transmisión y se ha calculado el PSNR medio (ver Figura E.1). Las wavelet que se han utilizado son la biortogonal 1.1 y la 7.9, la Coiflet de 12 coeficientes y la Daubechies de 20 coeficientes. El nivel de compresión utilizado es 3, ya que éste es el mínimo aconsejable.

Como se puede ver en la Figura E.1, la wavelet madre para la que se obtienen mejores resultados es para la Coiflet de 12 coeficientes. Además, ésta es menos compleja al tener menos coeficientes que la de 20, pero si utilizamos una Coiflet de menos coeficientes, los resultados obtenidos son peores.

Ahora, una vez seleccionada la wavelet madre, se debe seleccionar el nivel de compresión. Para la elección, interesa seleccionar el menor nivel posible, ya que el nivel de complejidad es menor al disminuir el nivel. En la Figura E.2 se representa el PSNR medio en función del ratio de transmisión para diferentes niveles de compresión.

El nivel de compresión seleccionado es de 4, ya que nos garantiza prácticamente el mismo PSNR que con un nivel de 5, e independientemente del ecocardiografo utilizado se podrá garantizar este nivel por ser la anchura mínima de franja de 32 píxeles.

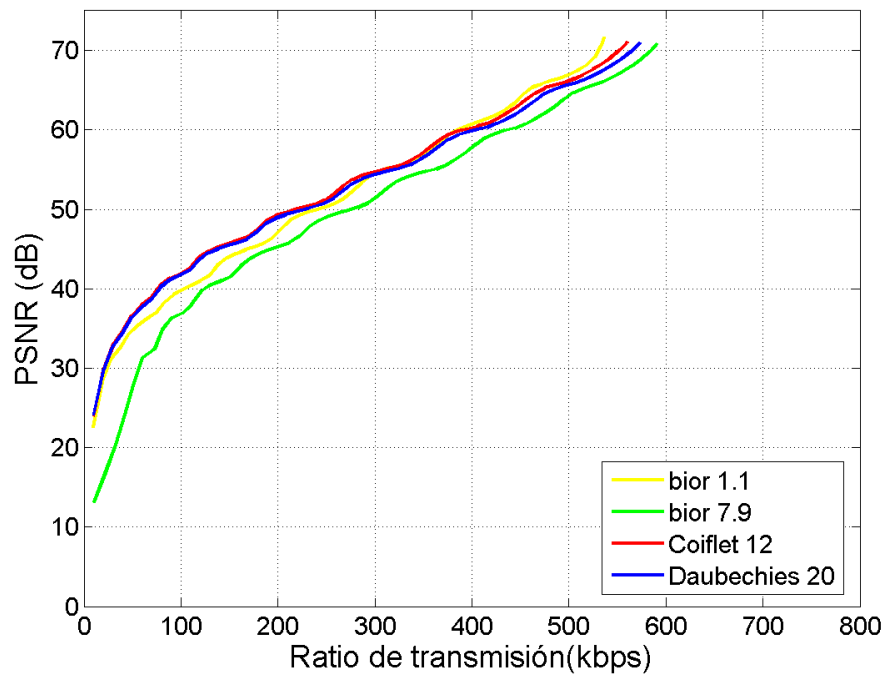


Figura E.1: *Curva ratio-distorsión para diferentes wavelets madre.*

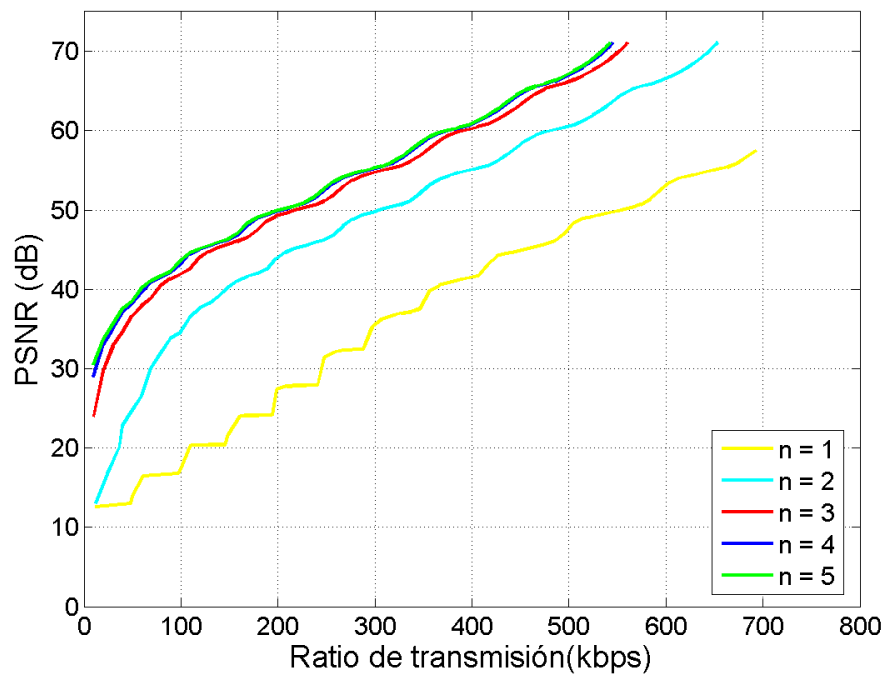


Figura E.2: *Curva ratio-distorsión para diferentes niveles.*

Anexo F

Set Partitioning In Hierarchical Trees (SPIHT) 3D

El algoritmo SPIHT 3D es una extensión del algoritmo SPIHT (explicado en el Anexo C) de dos a tres dimensiones, una temporal y dos espaciales, para la compresión de vídeo. El compresor y decompresor sigue el mismo esquema que el algoritmo SPIHT en dos dimensiones, pero aplicado a tres dimensiones como se puede ver en la Figura F.1.

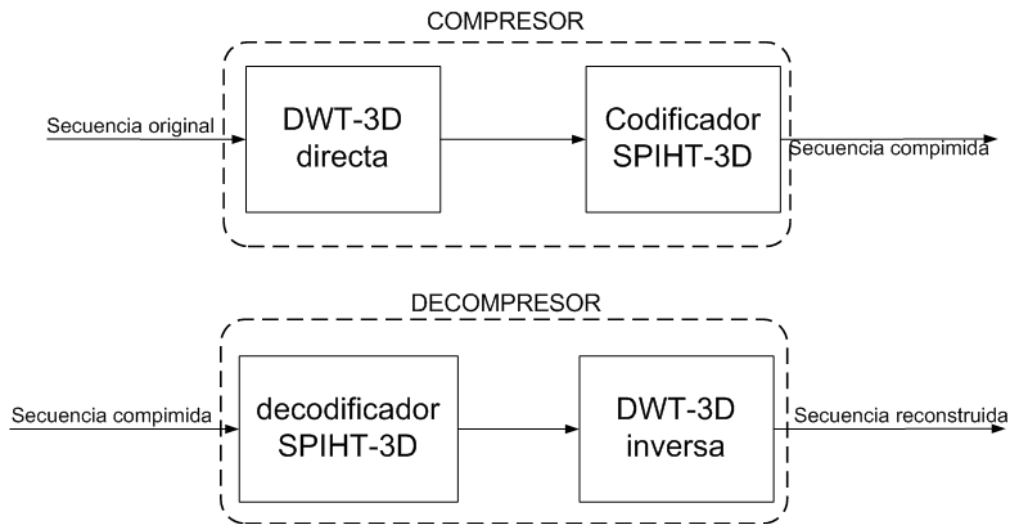


Figura F.1: *Esquema algoritmo SPIHT-3D.*

Primero se realiza la transformada wavelet de tres dimensiones. La extensión de dos a tres dimensiones se realiza utilizando filtros wavelet separables, primero en el dominio temporal y después en el espacial. El árbol de ordenación es el que se muestra en la Figura F.2. Ahora, los descendientes de un nodo (i,j,k) , excepto para el nivel más alto y

bajo, vendrán dados por la ecuación F.1.

$$O(i,j) = \{ (2i, 2j, 2k), (2i, 2j+1, 2k), (2i+1, 2j, 2k), \\ (2i+1, 2j+1, 2k), (2i, 2j, 2k+1), (2i+1, 2j, 2k+1), \\ (2i, 2j+1, 2k+1), (2i+1, 2j+1, 2k+1) \} \quad (F.1)$$

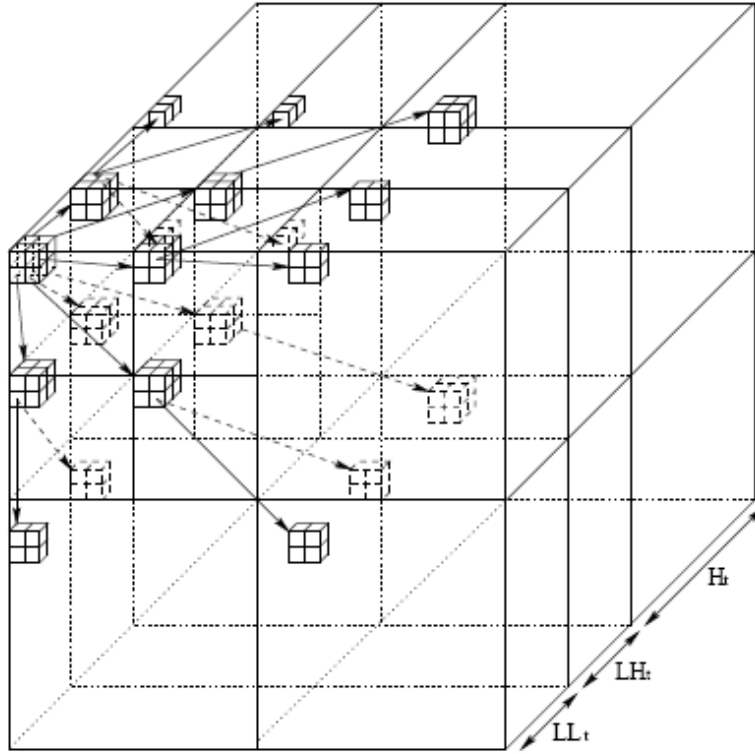


Figura F.2: *Árbol de ordenación para la wavelet de 3 dimensiones.*

El funcionamiento del decodificador SPIHT es el mismo que el de dos dimensiones, lo único que cambia es el árbol de ordenación.

Una de las consideraciones que se deben tener en cuenta, es el nivel de descomposición, que depende del número de frames que se procesan al mismo tiempo. Uno de los valores típicos seleccionados es 16 frames, para no tener un excesivo retardo en la compresión y tener una adecuada eficiencia en la compresión.

Anexo G

Run Length Encoding (RLE)

La compresión RLE (Run Length Encoding) es una forma muy simple de compresión de datos en la que secuencias de datos con el mismo valor consecutivas son almacenadas como un único valor más su recuento. Esto es más útil en datos que contienen muchas de estas "secuencias"; por ejemplo, gráficos sencillos con áreas de color plano, como iconos y logotipos.

Por ejemplo, considera una pantalla que contiene texto en negro sobre un fondo blanco. Habría muchas secuencias de este tipo con píxeles blancos en los márgenes vacíos, y otras secuencias de píxeles negros en la zona del texto. Supongamos una única línea (o scanline), con N representando las zonas en negro y B las de blanco:

BBBBBBBBBBBBBNBBBBBBBBBBBBNNBBBBBBBBBBBBBBBBBBBBBBBBBBB
NBBBBBBBBBBBBBBB

Si aplicamos la codificación RLE a esta línea, obtendríamos lo siguiente:

12B1N12B3N24B1N14B

Interpretado esto como 12 bes, 1 ene, 12 bes, 3 enes, etc. El código RLE representa el original de 67 caracteres en tan sólo 16. Esta codificación traducida a binario, cuyo principio es el mismo, se utiliza para el almacenamiento de imágenes. Incluso ficheros de datos binarios pueden ser comprimidos utilizando este método. El primer byte contiene un número que representa el número de veces que el carácter está repetido. El segundo byte contiene al propio carácter.

Algunos formatos que utilizan esta codificación incluyen Packbits, PCX e ILBM.

La codificación RLE realiza una compresión de datos sin pérdidas y es muy utilizado en imágenes de 8 bits indexadas (en un principio fue utilizado para imágenes en blanco y negro). No funciona tan bien en imágenes donde varía constantemente el color de los píxeles como fotografías, aunque JPEG lo utiliza de forma efectiva en los coeficientes que quedan después de transformar y cuantificar bloques de imágenes.

Anexo H

Resultados evaluación ecocardiogramas estándar

En este Anexo se muestra para cada uno de los ecocardiogramas estándar evaluados una tabla (Tablas H.1, H.2, H.3, H.4, H.5, H.6 y H.7). Cada tabla contiene el PSNR medio según el ancho de banda configurado (ancho de banda de línea) para cada uno de los métodos de compresión estudiados y el ancho de banda medio para el método propuesto, ya que este es menor que el ancho de banda configurado.

En este Anexo, además, se incluyen dos figuras por cada vídeo, una para el ancho de banda medio por tramo (H.1, H.3, H.5, H.7, H.9, H.11 y H.13) y otra para el PSNR medio por tramo (H.2, H.4, H.6, H.8, H.10, H.12 y H.14). En cada figura se representa el ancho de banda medio o el PSNR medio por tramo correspondiente a un modo (modo B, Doppler o barrido) para cuatro anchos de banda configurados (100, 400, 700 y 1000 kbps) en función del tiempo. El ancho de banda medio de los modos B y Doppler Color corresponde con el ratio de transmisión. En los modos de barrido no en todos los casos el ancho de banda medio corresponde con el de transmisión, ya que aproximadamente a los 600 kbps se alcanza la compresión sin pérdidas en estos modos, que nos da el ancho de banda máximo. Los tramos en los que el ancho de banda medio es cero son en los que el vídeo está parado. Cada tramo correspondiente a un modo se le ha asignado una etiqueta en la que “B” es el modo B, “D” es el modo Doppler Color y “S” son los modos de barrido.

	Metodo propuesto		H264	Xvid
BW linea (kbps)	BW (kbps)	PSNR (dB)	PSNR (dB)	PSNR (dB)
100	75.7658	35.6935	32.362	32.1076
150	113.6482	37.7642	33.577	32.116
200	151.5293	39.4661	34.481	33.043
250	189.4109	40.7869	35.168	35.2029
300	227.2928	42.2159	35.828	36.3963
350	265.1747	43.3871	36.467	37.1342
400	302.9502	44.8262	37.054	37.8197
450	340.0423	45.9884	37.698	38.3263
500	375.7695	47.3491	38.212	38.6983
550	410.284	48.2767	38.751	39.0753
600	443.9719	49.258	39.13	39.4856
650	475.9852	49.5077	39.535	39.7737
700	507.7882	49.7079	39.851	40.0754
750	539.5913	49.8885	40.183	40.4313
800	571.3944	50.0593	40.467	40.7009
850	603.1974	50.2105	40.737	40.9275
900	635.0005	50.3598	41.006	41.1608
950	666.8036	50.5013	41.248	41.3564
1000	698.6066	50.6416	41.512	41.5799

Tabla H.1: PSNR y ancho de banda para el vídeo 1.

	Metodo propuesto		H264	Xvid
BW linea (kbps)	BW (kbps)	PSNR (dB)	PSNR (dB)	PSNR (dB)
100	64.6188	36.4977	34.808	32.5414
150	96.9287	38.7317	36.515	32.601
200	129.2019	40.8205	37.404	36.0043
250	161.4574	42.4502	38.357	37.668
300	193.7105	44.3764	39.061	38.6777
350	225.8248	46.122	39.651	39.457
400	257.6405	47.9894	40.139	40.0338
450	288.603	49.6506	40.537	40.5543
500	318.6545	51.4143	40.916	41.0099
550	347.79	52.9539	41.258	41.3583
600	373.5774	54.034	41.562	41.6339
650	397.3509	54.2599	41.831	41.9287
700	421.0698	54.4434	42.094	42.2694
750	444.7887	54.6065	42.339	42.611
800	468.5075	54.7623	42.561	42.877
850	492.2264	54.9032	42.78	43.1142
900	515.9453	55.0362	42.991	43.3523
950	539.6642	55.1643	43.178	43.543
1000	563.3831	55.2878	43.351	43.8107

Tabla H.2: PSNR y ancho de banda para el vídeo 2.

	Metodo propuesto		H264	Xvid
BW linea (kbps)	BW (kbps)	PSNR (dB)	PSNR (dB)	PSNR (dB)
100	87.9562	35.2642	35.466	32.1076
150	131.934	36.9427	36.766	32.116
200	175.9106	38.3875	37.748	33.043
250	219.8869	39.5922	38.559	35.2029
300	263.8643	40.8785	39.204	36.3963
350	307.8216	42.0505	39.76	37.1342
400	351.736	43.2998	40.227	37.8197
450	395.435	44.5146	40.611	38.3263
500	438.4291	45.6461	40.964	38.6983
550	479.9494	46.6189	41.269	39.0753
600	519.3033	47.2991	41.542	39.4856
650	557.2151	47.5798	41.802	39.7737
700	595.098	47.8272	42.028	40.0754
750	632.9809	48.0545	42.256	40.4313
800	670.8637	48.2672	42.455	40.7009
850	708.7466	48.471	42.643	40.9275
900	746.6295	48.658	42.828	41.1608
950	784.5124	48.835	42.993	41.3564
1000	822.3953	48.9982	43.158	41.5799

Tabla H.3: PSNR y ancho de banda para el vídeo 3.

	Metodo propuesto		H264	Xvid
BW linea (kbps)	BW (kbps)	PSNR (dB)	PSNR (dB)	PSNR (dB)
100	67.8436	36.1873	32.471	32.9887
150	101.7627	38.4059	34.134	33.0114
200	135.682	40.4812	35.166	34.4651
250	169.5817	42.4384	35.954	36.7517
300	203.3999	44.3937	36.62	38.0025
350	237.1149	45.9715	37.226	38.865
400	270.679	47.9009	37.738	39.5312
450	303.9992	49.4186	38.168	40.0238
500	336.741	51.23	38.673	40.4924
550	367.6791	52.6147	39.078	40.7846
600	396.559	53.8817	39.553	41.1221
650	422.371	54.1238	40.004	41.5576
700	447.9916	54.2967	40.46	41.865
750	473.6122	54.4622	40.931	42.0795
800	499.2328	54.6149	41.313	42.3754
850	524.8534	54.7498	41.619	42.6544
900	550.474	54.8765	41.923	42.9139
950	576.0946	54.9985	42.191	43.1333
1000	601.7152	55.1151	42.405	43.3573

Tabla H.4: PSNR y ancho de banda para el vídeo 4.

	Metodo propuesto		H264	Xvid
BW linea (kbps)	BW (kbps)	PSNR (dB)	PSNR (dB)	PSNR (dB)
100	69.4848	34.346	34.945	32.2505
150	104.2257	36.2936	36.392	32.2777
200	138.9655	38.0629	37.674	35.8556
250	173.7047	39.7197	38.527	37.2141
300	208.4462	41.3286	39.188	38.3986
350	243.1859	42.9598	39.729	39.1029
400	277.8622	44.6081	40.206	39.7381
450	312.1184	46.3495	40.629	40.2361
500	345.1068	47.9561	40.998	40.6268
550	375.3893	49.5258	41.296	41.0686
600	401.773	50.6015	41.608	41.4203
650	426.2201	51.0267	41.88	41.737
700	450.2815	51.3159	42.135	42.0556
750	474.3272	51.551	42.375	42.3505
800	498.373	51.77	42.598	42.5917
850	522.4187	51.9917	42.807	42.8586
900	546.4644	52.1926	43.011	43.0902
950	570.5102	52.3696	43.195	43.3089
1000	594.5559	52.5305	43.374	43.5622

Tabla H.5: PSNR y ancho de banda para el vídeo 5.

	Metodo propuesto		H264	Xvid
BW linea (kbps)	BW (kbps)	PSNR (dB)	PSNR (dB)	PSNR (dB)
100	62.2508	36.8308	36.625	32.7821
150	93.3754	39.2568	37.922	34.7989
200	124.4993	41.2153	38.916	37.2949
250	155.6238	42.7829	39.713	38.5519
300	186.7486	44.4716	40.331	39.4133
350	217.8685	45.7334	40.865	40.0541
400	248.7788	47.4389	41.291	40.5748
450	279.3953	48.7131	41.652	41.0752
500	309.2639	50.242	41.987	41.4264
550	337.7334	50.9719	42.267	41.7702
600	365.7056	51.6716	42.554	42.1106
650	393.1322	51.8958	42.799	42.4291
700	420.5531	52.1065	43.053	42.7077
750	447.974	52.311	43.264	43.0315
800	475.395	52.4988	43.465	43.2306
850	502.8159	52.6694	43.657	43.449
900	530.2368	52.8259	43.825	43.7095
950	557.6577	52.9726	43.999	43.9332
1000	585.0786	53.1131	44.158	44.1363

Tabla H.6: PSNR y ancho de banda para el vídeo 6.

	Metodo propuesto		H264	Xvid
BW linea (kbps)	BW (kbps)	PSNR (dB)	PSNR (dB)	PSNR (dB)
100	47.8828	38.0357	36.576	31.7778
150	71.8239	41.1126	38.054	32.9328
200	95.7633	43.7481	39.162	36.6373
250	119.7001	45.831	39.942	38.3793
300	143.5355	47.8342	40.532	39.4352
350	167.2456	49.9264	40.972	40.1861
400	190.7295	51.8092	41.356	40.7195
450	213.5735	53.8773	41.693	41.1384
500	234.3206	55.721	41.976	41.6066
550	252.3165	57.4256	42.257	41.9822
600	268.1818	58.1448	42.502	42.3658
650	283.2232	58.3633	42.74	42.6353
700	297.9265	58.4846	42.975	42.9089
750	312.6297	58.5983	43.202	43.2572
800	327.3329	58.7027	43.388	43.4936
850	342.0361	58.7996	43.563	43.7223
900	356.7394	58.8934	43.736	43.9525
950	371.4426	58.9855	43.895	44.1941
1000	386.1458	59.0754	44.048	44.4097

Tabla H.7: PSNR y ancho de banda para el vídeo 7.

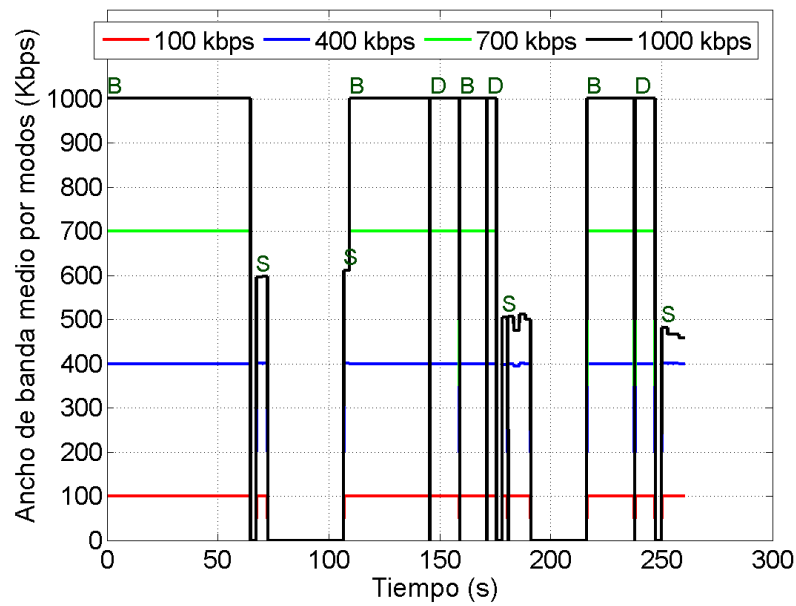


Figura H.1: Ancho de banda medio por tramo para el vídeo 1. Donde “B” es el modo B, “D” es el modo Doppler Color y “S” son los modos de barrido.

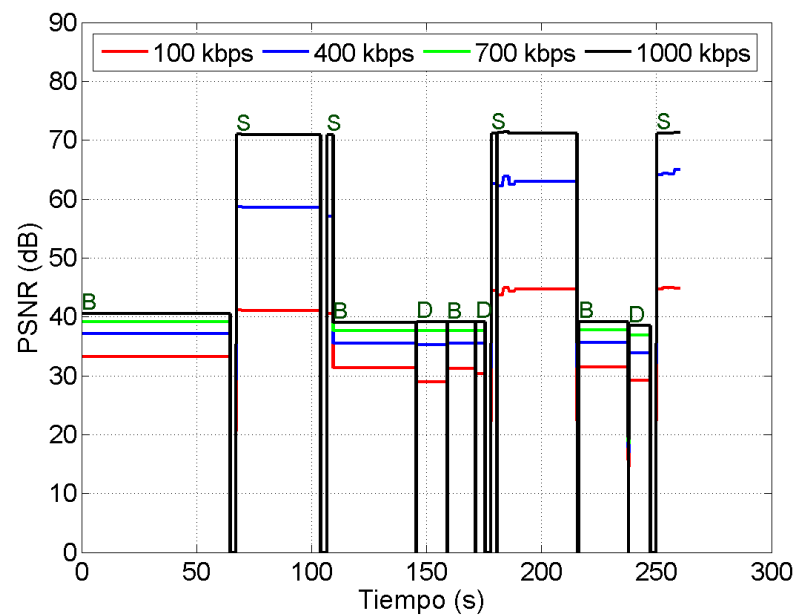


Figura H.2: PSNR medio por tramo para el vídeo 1. Donde “B” es el modo B, “D” es el modo Doppler Color y “S” son los modos de barrido.

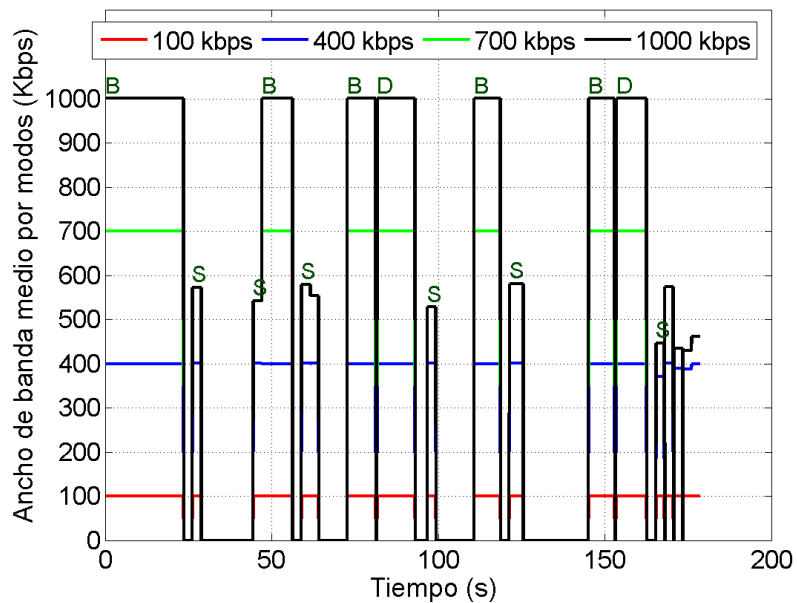


Figura H.3: Ancho de banda medio por tramo para el vídeo 2. Donde “B” es el modo B, “D” es el modo Doppler Color y “S” son los modos de barrido.

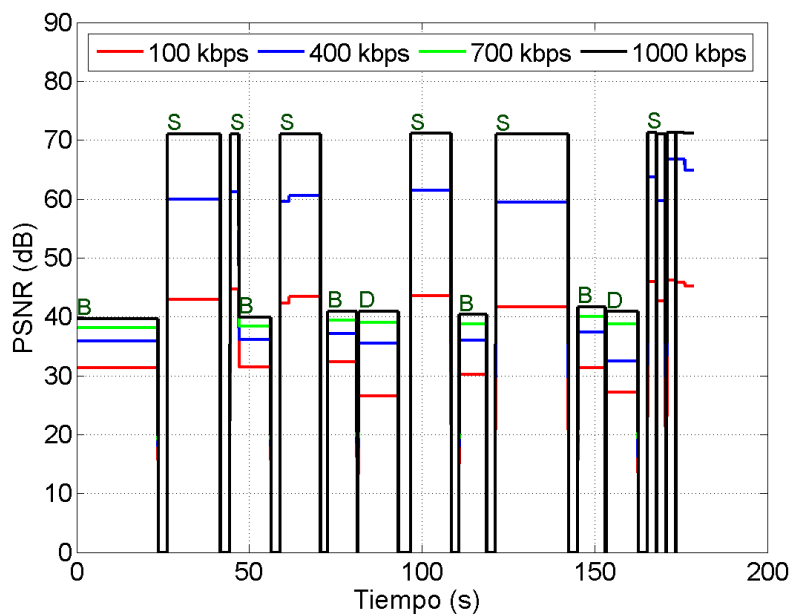


Figura H.4: PSNR medio por tramo para el vídeo 2. Donde “B” es el modo B, “D” es el modo Doppler Color y “S” son los modos de barrido.

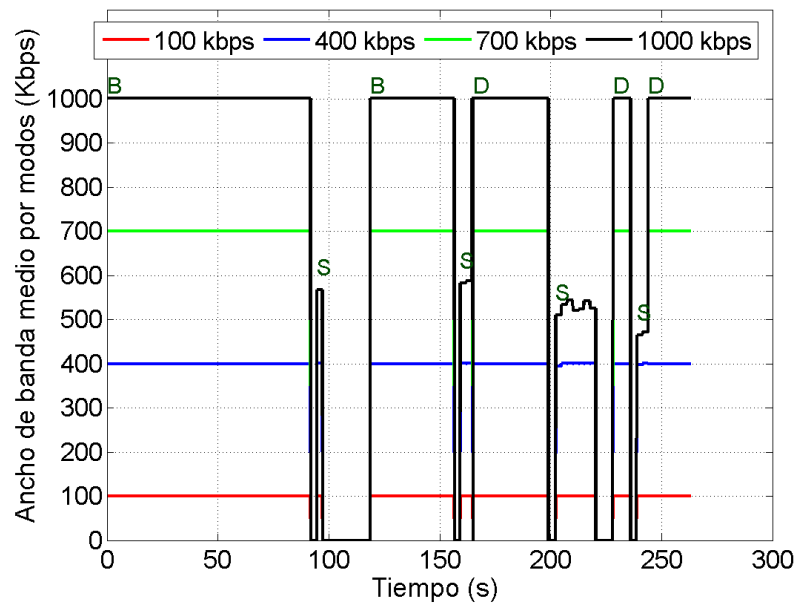


Figura H.5: Ancho de banda medio por tramo para el vídeo 3. Donde “B” es el modo B, “D” es el modo Doppler Color y “S” son los modos de barrido.

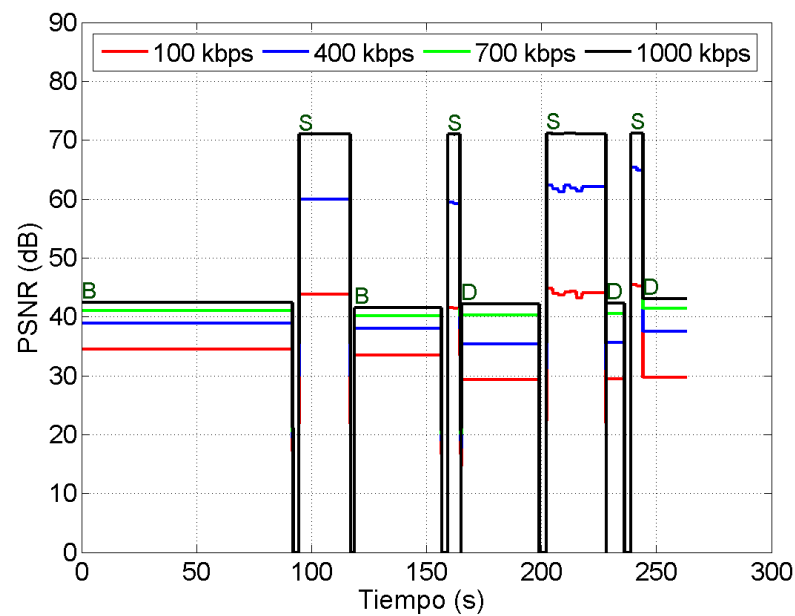


Figura H.6: PSNR medio por tramo para el vídeo 3. Donde “B” es el modo B, “D” es el modo Doppler Color y “S” son los modos de barrido.

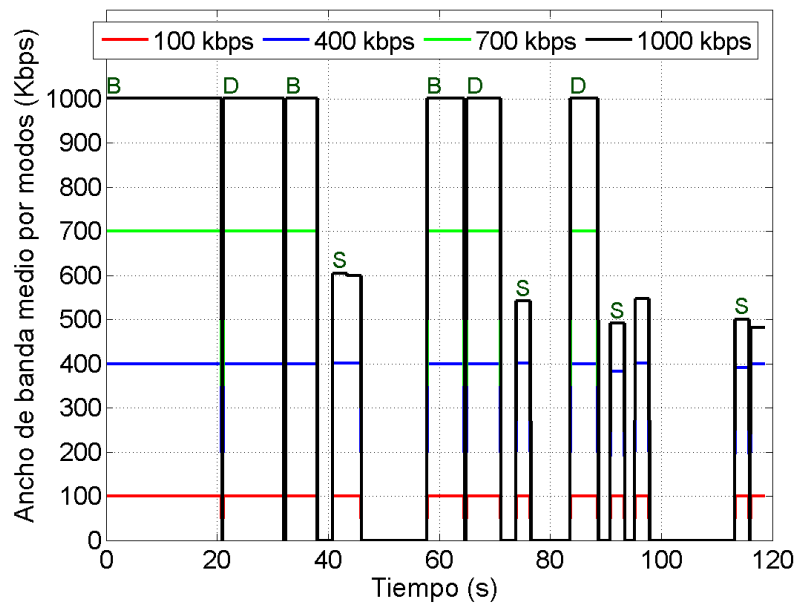


Figura H.7: Ancho de banda medio por tramo para el vídeo 4. Donde “B” es el modo B, “D” es el modo Doppler Color y “S” son los modos de barrido.

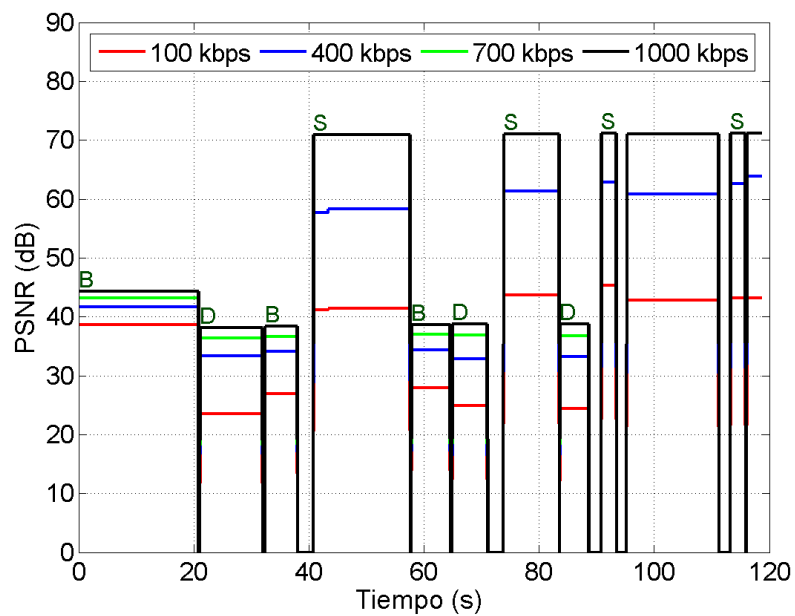


Figura H.8: PSNR por tramo para el vídeo 4. Donde “B” es el modo B, “D” es el modo Doppler Color y “S” son los modos de barrido.

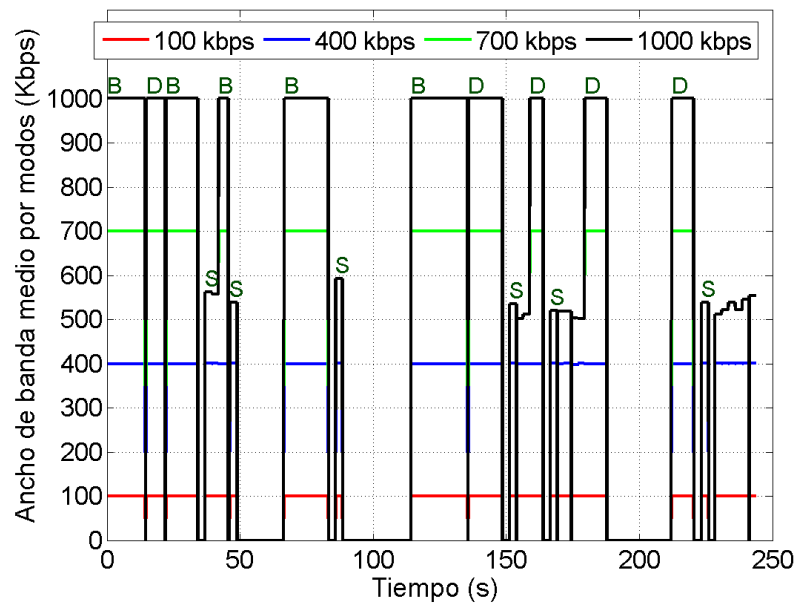


Figura H.9: Ancho de banda medio por tramo para el vídeo 5. Donde “B” es el modo B, “D” es el modo Doppler Color y “S” son los modos de barrido.

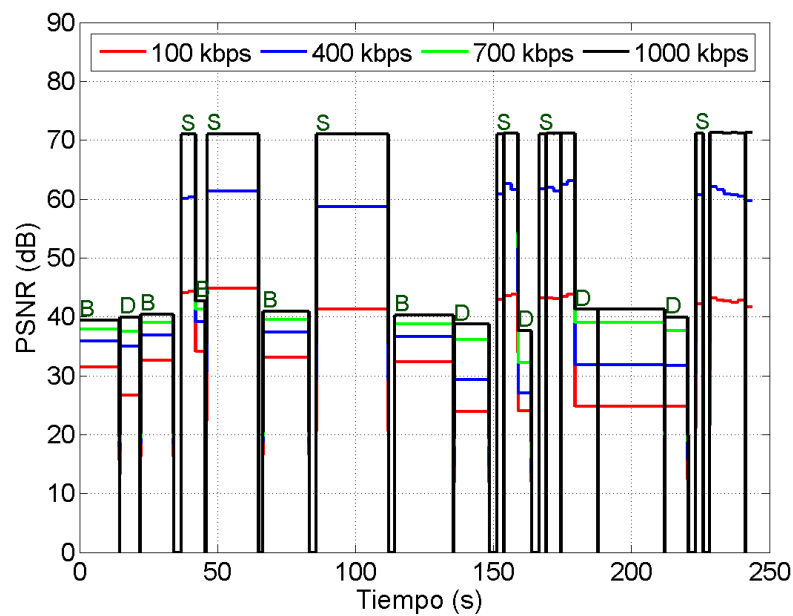


Figura H.10: PSNR por tramo para el vídeo 5. Donde “B” es el modo B, “D” es el modo Doppler Color y “S” son los modos de barrido.

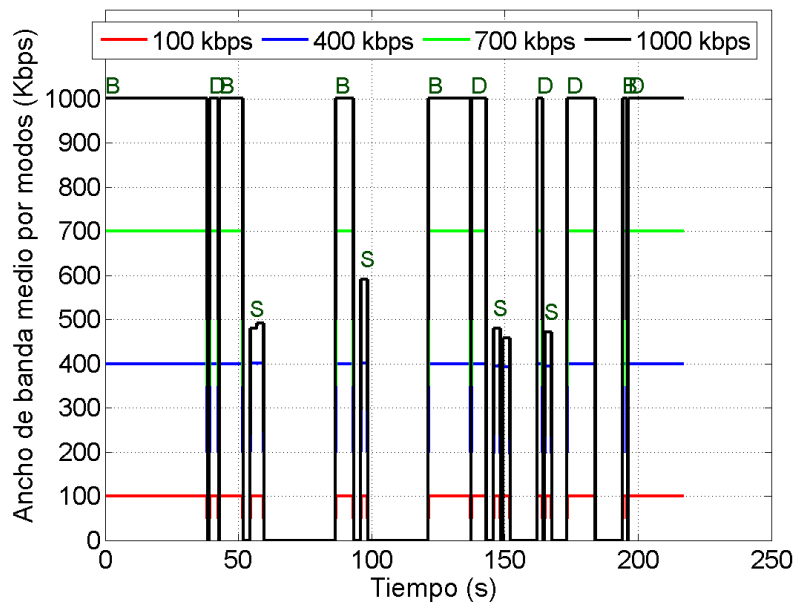


Figura H.11: *Ancho de banda medio por tramo para el vídeo 6. Donde “B” es el modo B, “D” es el modo Doppler Color y “S” son los modos de barrido.*

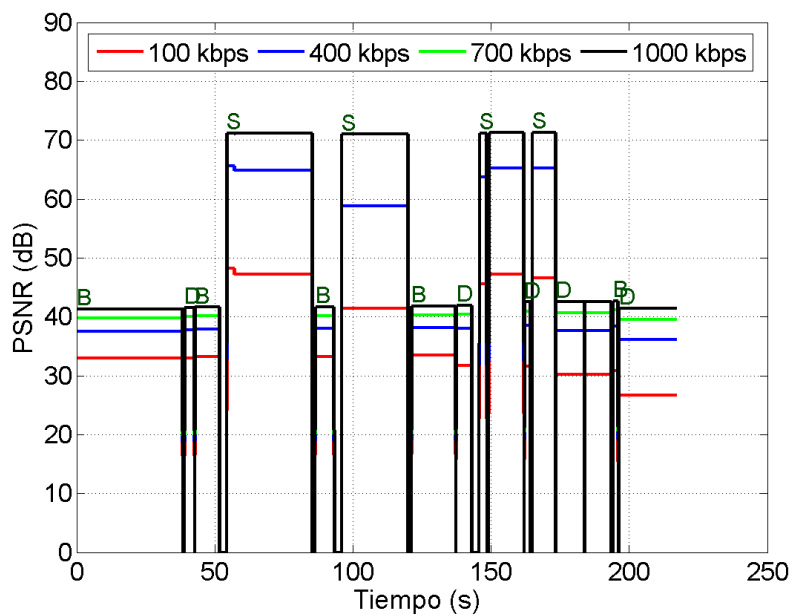


Figura H.12: *PSNR medio por tramo para el vídeo 6. Donde “B” es el modo B, “D” es el modo Doppler Color y “S” son los modos de barrido.*

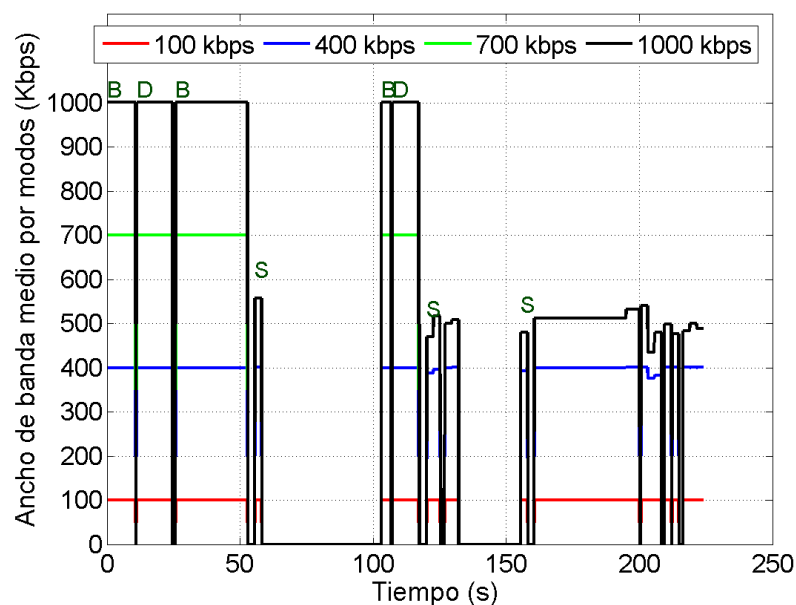


Figura H.13: Ancho de banda medio por tramo para el vídeo 7. Donde “B” es el modo B, “D” es el modo Doppler Color y “S” son los modos de barrido.

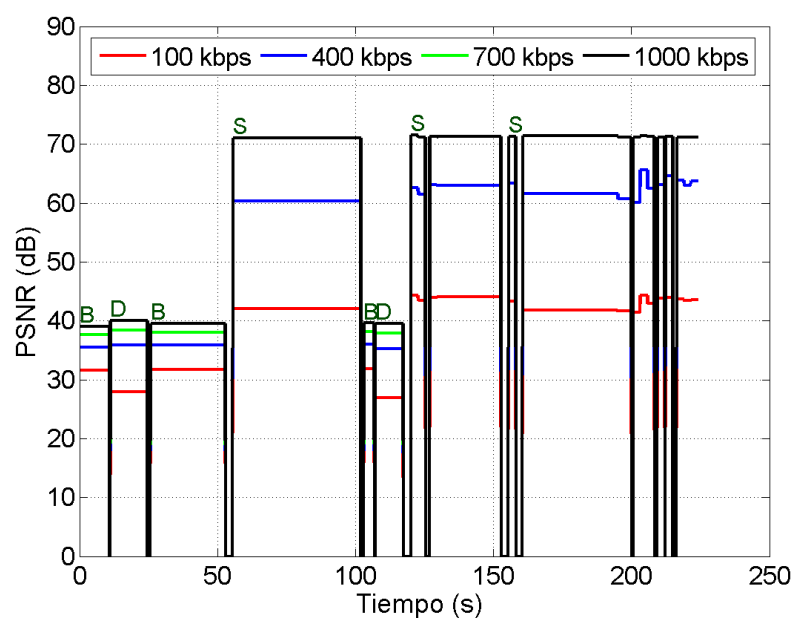


Figura H.14: PSNR medio por tramo para el vídeo 7. Donde “B” es el modo B, “D” es el modo Doppler Color y “S” son los modos de barrido.