Codificación de Vídeo Escalable usando Motion Compensated JPEG2000 con Control de Bit-Rate

Carmelo Maturana-Espinosa, Joaquín García-Sobrino, J.J. Sánchez-Hernández y Vicente González-Ruiz

Dpto. de Informática. Grupo SAL. Universidad de Almería, Campus de Excelencia Internacional Agroalimentario, ceiA3. [1]

Palabras clave: Control del Bit-Rate, JPEG2000, Filtrado Temporal con Compensación Movimiento.

Hoy en día los dispositivos de codificación de vídeo necesitan realizar transmisión de datos hacia una gran variedad de terminales, posiblemente con diferentes resoluciones de pantalla, distinta capacidad de computación y a través de redes con ancho de banda variable. La codificación de vídeo escalable es una alternativa para soportar de forma óptima estas aplicaciones. Por otra parte, el control de bit-rate y la calidad de la imagen se han convertido en uno de los retos más importantes en la transmisión de vídeo ya que su adecuado tratamiento permite maximizar la calidad de los vídeos reconstruidos. En este trabajo estudiamos el problema de control de bit-rate para secuencias de imágenes comprimidas usando Motion JPEG2000 y estimación de movimiento. Proponemos una técnica iterativa y evaluamos su eficacia en comparación con el control de bit-rate que realiza el codec H.264/SVC, obteniendo resultados que avalan nuestra propuesta.

I. Introducción

Los avances tecnológicos y los estándares de codificación de vídeo, unidos al rápido desarrollo y mejoras de las infraestructuras de red, capacidad de cómputo y almacenamiento, han permitido un rápido incremento del número de aplicaciones que manejan vídeo. Esto, unido a proliferación de servicios multimedia en Internet y al auge que en los últimos años han experimentado las redes inalámbricas de banda ancha, han hecho que la comunicación y transmisión de vídeo a través de la red se haya convertido en un foco de interés tanto para la industria como para el mundo académico.

La transmisión de vídeo en tiempo real requiere una calidad de servicio que generalmente esté condicionada por: el ancho de banda, el retraso en la entrega y los errores de transmisión. La transmisión de vídeo presenta, por norma general, una serie de requisitos mínimos en cuanto a ancho de banda para conseguir una calidad aceptable de presentación. Por otro lado, la transmisión de vídeo requiere estrictos requisitos en cuanto al retraso en la entrega. Si los paquetes que componen el vídeo no llegan de la forma adecuada, la reproducción sufrirá pausas no deseadas. Por último, las aplicaciones de vídeo normalmente imponen límites en cuanto al número de errores permitidos ya que demasiados errores degradarían seriamente la calidad de reproducción. Además, en los casos de multidifusión de vídeo, la heterogeneidad de los receptores hacen difícil conseguir la eficiencia y flexibilidad requerida. Por lo tanto, es deseable contar con esquemas de codificación de vídeo escalable que permitan adaptarse a la situación concreta de las condiciones en las que se realiza la transmisión / reproducción.

Un esquema de codificación de vídeo escalable consiste básicamente en comprimir una secuencia vídeo en varios sub-streams. Uno de estos sub-streams se considera el sub-stream base que puede ser independientemente descodificado y proporciona generalmente una calidad visual de reproducción básica. Los demás sub-streams son mejoras del sub-stream base y pueden ser solamente descodificados junto con el sub-stream base para proporcionar una mejor calidad visual del vídeo. La descodificación del sub-stream base junto con determinadas combinaciones de los demás sub-streams produce secuencias de vídeo con mayor o menor pérdida de calidad, distinto tamaño de las imágenes del vídeo y diferente frame-rate. Esto es lo que se conoce como escalabilidad en calidad, espacial y temporal, respectivamente.

La codificación de vídeo escalable puede ayudar al BRC (Bit-Rate Control) debido a la capacidad de reconstrucción a distintas resoluciones y/o calidades a partir de bit-streams incompletos. Esta capacidad permite una fácil y rápida adaptación a las capacidades concretas de cada red y/o terminal. Se pueden conseguir diferentes formas de escalabilidad según el estándar de codificación usado. En nuestro caso, nos centramos en el estándar JPEG2000 porque nos permite conseguirlas todas.

El resto del Paper está organizado como sigue. En el Apartado II se presenta una visión general de JPEG2000, Motion JPEG2000 y H.264/SVC. Éste último se ha usado para compararlo a nivel de eficiencia con nuestra propuesta. Una breve descripción del codec MCTF+JPEG2000 básico (MCJ2K) además del algoritmo propuesto de un eficiente BRC para MCJ2K, se muestra en la Sección III. En la Sección

[1]Este trabajo ha sido financiado por subvenciones del Ministerio de Ciencia e Innovación de España (TIN2008-01117), la Junta de Andalucía (P08-TIC-3518 y P10-TIC-6548), y en parte financiado por el Fondo Europeo de Desarrollo Regional (ERDF).

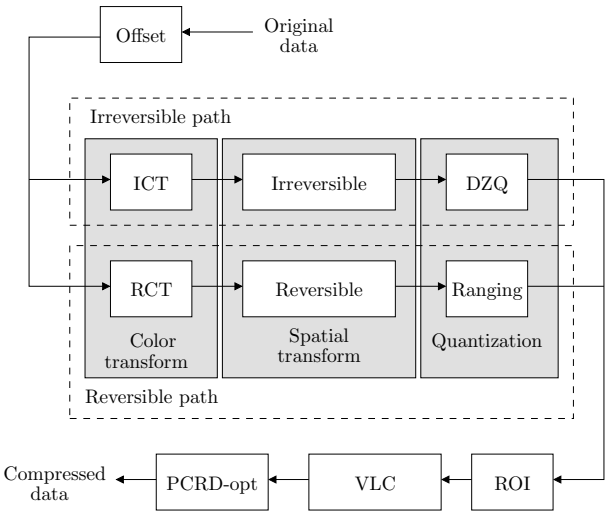

Fig. 1. Diagrama de bloques del compresor JPEG2000.

IV presentan los resultados experimentales. Por último, en la Sección V se muestran las conclusiones a las que llegamos en el estudio.

II. Trabajo relacionado

JPEG2000 [1], [2] es el estándar ISO/ITU más reciente de compresión de imágenes. El objetivo de este apartado es sólo proporcionar una breve descripción a muy alto nivel del codec JPEG2000 para el posterior entendimiento los siguientes apartados de este trabajo.

En la Figura 1 se muestra una versión simplificada del diagrama de bloques del codificador JPEG2000. En primer lugar, según se desee aplicar el codec con o sin pérdida se toma uno de los caminos representados. El primer paso, llamado Offset normaliza las muestras de cada bloque entre [-0.5, 0.5[provocando una pérdida irreversible en los datos, o teniendo encuenta el nº de bits/componente para permitir una decodificación reversible. A continuación, cada una de estas particiones se somete de forma opcional a una transformación de componentes con el objetivo de conseguir descorrelacionar las componentes de color. Dicha transformación de color puede ser irreversible (ICT), donde las componentes se representan en formato YCbCr. En caso de hacerse de forma reversible (RCT) la transformación se hace en un dominio de enteros, en el que se puede invertir las operación sin perdida de precisión. En cualquier caso a las componentes resultantes se les aplica la Transformada Wavelet (llamada transformada espacial en la Figura 1) y después son cuantificadas, eliminando generalmete ruido de alta frecuencia.

La siguiente etapa, consiste en la definición de la(s) region(es) de interés (ROI). Si las hubiera, los bloques comprendidos en dicha región serian extraídos en mayor medida que el resto, aportando una mayor calidad de ellos, haciendo uso de la codificación de longitud variable (VLC) que hace efectiva la compresión de los datos. Esto es, cada sub-banda se divide en codeblocks rectangulares, que se comprimen de forma independiente usando un codificador bitplane que hace tres pasadas sobre cada bitplane de cada codeblock consiguiendo como resultado un bitsream truncable de cada codeblock. Por último la etapa PCRD-opt (Post Compression Rate-Distortion Optimization), organiza los datos en el code-stream con el objeto de conseguir las diferentes formas de escalabilidad. Así durante la descompresión, para la escalabilidad en calidad, por ejemplo, las sub-bandas deben ser decodificadas usando un orden LRCP (capas de calidad, resolución, componentes y precintos) ya que es el orden de paquetes que guarda mejor relación entre el tamaño del code-stream y calidad de la imagen descomoprimida usando dichos paquetes.

El comité JPEG amplió el estándar JPEG2000 para la codificación de vídeo. El resultado se conoce como Motion JPEG2000 (MJ2K). MJ2K no incluye compensación de movimiento (cada frame se comprime de forma individual) y está orientado a generar vídeo comprimido altamente escalable que puede ser fácilmente editado.

H.264/AVC [3], [4] es un codificador de vídeo estándar que proporciona gran eficiencia de compresión. La versión escalable de H.264/AVC es H.264/SVC. En SVC se permite escalabilidad temporal, espacial y en calidad.

III. Propuesta

Son muchos los esfuerzos destinados al desarrollo y mejora de sistemas de codificación de vídeo eficientes. Estos estudios consiguen buenos resultados centrándose en características concretas de la "codificación ideal" de un vídeo como pueden ser un buen acceso aleatorio al contenido [5] o portabilidad en la transmisión por un canal estrecho [6]. Sin embargo, parece evidente que el mayor número de ventajas se consigue por medio de compensación temporal de movimiento, donde por ejemplo, puede llegar a conseguirse una notable reducción del bit-rate necesario para llevar a cabo la codificación de un vídeo. Aunque se trata de un aspecto importante de mejora del rendimiento de compresión, sigue siendo un proceso que no cuenta una metodología estándar. Existen numerosos estudios que aplican alguna estrategia de predicción de movimiento usando técnicas como MCTF [7] (Motion Compensated Temporal Filtering) o LIMAT [8] (Lifting-based Invertible Motion Adaptive Transform) y que muestran resultados interesantes.

Como se observa en las dependencias de la Figura 2, MCTF provee de una escalabilidad temporal y controles para minimizar el impacto de errores y pérdidas en los datos. El resultado de aplicar MCTF a una secuencia de imágenes es otra secuencia de imágenes filtradas en el dominio del tiempo, que pueden ser comprimidas eficientemente por un compresor de imágenes MJ2K.

En este trabajo se propone usar una técnica novedosa a la que hemos denominado MCJ2K (Motion Compensated JPEG2000), pretende aunar las ventajas de MCTF y Motion JPEG2000 en la com-

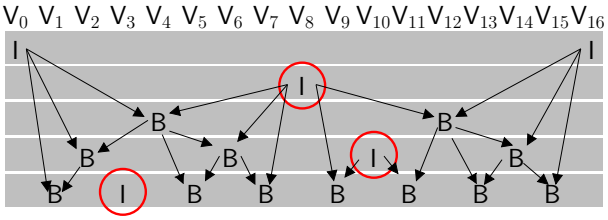

Fig. 2. MCTF sin suficiente correlación temporal.

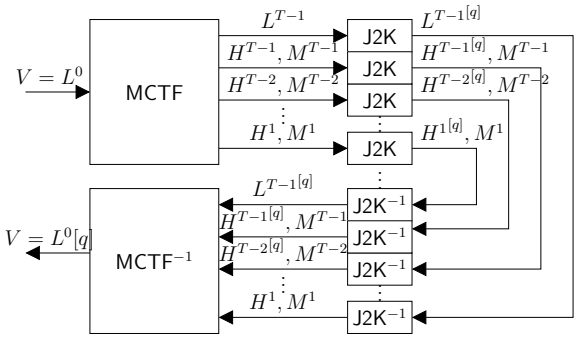

Fig. 3. Estructura básica de MCJ2K.

presión de secuencias de imágenes con compensación de movimiento.

MCJ2K proporciona compresión escalable en espacio, tiempo y calidad mediante la distribución de los datos en capas de calidad y la posterior ordenación de los mismos. Durante la descompresión para la escalabilidad en calidad, las imágenes de las sub-bandas temporales deben ser decodificadas usando un orden LRCP. Cada sub-banda temporal será un code-stream diferente, comprimido de forma independiente con MJ2K. El esqueleto de MCJ2K se puede ver en la Figura 3, donde el vídeo original es divido en sub-bandas temporales mediante el algoritmo de MCTF y cada una de las imágenes residuo (B) en cada sub-banda es comprimida usando J2K. Para la descompresión se invierte el proceso dando lugar al vídeo reconstruido, a partir de vectores de movimiento (M) comprimidos sin pérdidas (habitualmente), imágenes comprimidas con pérdidas (generalmente) intra (I) y residuos.

Es importante precisar que la implementación de MCTF debe sufrir ciertas modificaciones para adaptarse al esquema inicial (Figura 3) y siga una típica implementación recursiva "filter bank", con transformaciones hacia adelante y hacia atrás. Por ejemplo, en caso de no encontrar suficiente correlación temporal, las imágenes B son reemplazadas por I y las referencias en los vectores de movimiento se eliminan (Figura 2).

A. *El code-stream MCJ2K*

Al aplicar el algoritmo de compresión de vídeo de MCTF, el code-stream generado, es compuesto de múltiples conjuntos de información. Como se observa en la Figura 2 se obtienen tantas sub-bandas de información sobre texturas como número de niveles de resolución temporales (TRL) indicados para la compresión. Por otra parte, la información del movimiento se distribuye en tantos campos de movimiento como sub-bandas de altas frecuencias. El code-stream se compone de la siguiente información:

1. **Texturas:** Comprenden la secuencia de imágenes del vídeo divididas en sub-bandas. Cada sub-banda está compuesta por imágenes I y/o residuos B, que son comprimidos con un nº concreto de capas de calidad, de manera que se puede enviar sólo el nº de capas deseado de una sub-banda. Cada capa adicional proporciona información más precisa de las imágenes de la sub-banda. No hay límite en el nº de capas con la que se desee crear una sub-banda, pero un nº demasiado alto provoca ineficiencias, ya que cada capa necesita de una cabecera. Además puede que la información de la sub-banda no se pueda dividir en más partes (capas) y entonces se creen capas vacías sin información útil, sólo con cabeceras, siendo una capa que únicamente engorda el code-stream. En la Figura 3 podemos observar que existen dos tipos de sub-bandas, la L y las H, según el rango de frecuencia de las texturas que almacenan.
   (a) **L (Low):** Esta sub-banda no se relaciona con información de movimiento, siendo totalmente independiente del resto. Contiene sólo imágenes I.
   (b) **H (High):** Son generadas TRL - 1 sub-bandas, que contienen las imágenes residuos producidas por el algoritmo MCTF. Dichas imágenes sólo contienen texturas de altas frecuencias.
2. **Movimiento:** Para cada sub-banda con altas frecuencias, existe un conjunto de vectores de movimiento, comprimidos sin pérdida. Éstos indican la posición de cada uno de los bloques de cada imagen B, en cada sub-banda H. Para nuestra propuesta será considerado cada campo de movimiento como una capa de calidad.

B. *Reconstrucción de un code-stream incompleto*

Este repertorio de sub-bandas, campos de movimiento y capas hace posible un envío selectivo y óptimo del code-stream cuando es necesario truncarlo. Así en situaciones de transmisión reales, donde se produce un truncamiento del code-stream o incluso la pérdida completa de una sub-banda, el descompresor necesita poder reconstruir el vídeo usando un code-stream incompleto. En nuestra propuesta, en caso de faltar uno(s) campo(s) de movimiento, se usa la compensación del movimiento para interpolar los datos que faltan. De manera que, el descompresor interpola linealmente los frames que no son recibidos, manteniendo constante el framerate. Si los datos que faltan son una(s) sub-banda(s), las texturas no recibidas se interpretan como negro y se mezclan con las texturas de las que sí se dispone.

C. Un algoritmo de Control del Bit-Rate (BRC)

En este trabajo se evalúa el uso de una técnica de control del bit-rate que permite maximizar la calidad del vídeo descomprimido en un entorno de transmisión de ancho de banda inferior al necesario para enviar el vídeo sin truncar. La disminución de la distorsión que produce cada capa (entiéndase como un campo de movimiento o capa de calidad de una sub-banda), es aditiva [9], es decir, podemos calcular la distorsión (o calidad) de un vídeo reconstruido, como la suma de las calidades que proporciona cada capa de cada sub-banda o campo de movimiento de forma independiente, lo mismo ocurre entre las capas de calidad de cada imagen. Nuestro algoritmo BRC, se sirve de esta característica para conocer cual es el beneficio real (disminución de distorsión frente a aumento de bit-rate) de cada capa a la hora de la reconstrucción. Así, según nuestro algoritmo BRC, la selección del orden óptimo de entre todas las capas se lleva a cabo tratando de minimizar la curva Rate/Distorsión (R/D) del vídeo reconstruido, siendo R su bit-rate y D su error cuadrático medio (RMSE). Si la selección de capas es adecuada, cada capa adicional enviada al descompresor proporciona una disminución en la distorsión y un aumento en el bit-rate consumido. Es decir, añadir una capa a la reconstrucción queda representado en una gráfica R/D como un punto que se sitúa más a la derecha (más bit-rate) y más abajo (menos distorsión) que el punto anterior que carecía de dicha capa. Así la pendiente, (*slope* en el Algoritmo 1) en una curva R/D debería ser monótonamente no creciente, con lo que dibuja un triángulo rectángulo entre cada par de puntos, permitiendo calcular su inclinación. A mayor inclinación, mejor opción constituye la capa analizada. La labor de nuestro algoritmo BRC consiste en calcular las inclinaciones derivadas de añadir una capa al code-stream en cada sub-banda y campo de movimiento, para seleccionar la mejor, obteniendo así un nuevo code-stream con dicha capa adicional. Este nuevo code-stream se usa entonces para la siguiente iteración de este proceso de cálculo y selección.

En el pseudocódigo del Algoritmo 1 se describe este proceso, así como los parámetros de entrada y salida en la Tabla 1. El parámetro L^0 representa el vídeo original y t su duración, ésta es necesaria para calcular el valor real del ancho de banda consumido por un code-stream y así poder comparar con el ancho de banda disponible (B). Para la aplicación de nuestro algoritmo de BRC se debe disponer del code-stream completo (C), es decir, no truncado, generado previamente mediante la compresión del vídeo original. Esta compresión necesita de parámetros que también son necesarios para poder truncar y reconstruir dicho code-stream, éstos son, entre otros menos relevantes, el nº de TRL (T) y el nº de capas con las que se comprimió cada sub-banda y campo de movimiento, representado como ($Q[]$), este parámetro es un vector de enteros donde las primeras (T) posiciones hacen referencia a las sub-bandas de texturas, siendo éstas por ejemplo: L, H2 y H1 para un T=3. Las siguientes *T-1* posiciones hacen referencia a los campos de movimiento correspondientes a H2 y H1, en el ejemplo dado. La finalidad del algoritmo consiste en devolver un fichero (*E_list*) con el orden óptimo de envío de las capas.

En la *línea 1* del Algoritmo 1 se inicializa el parámetro *V[]*, el cual tiene la misma estructura que el parámetro *Q[]*, con la salvedad de que en este último se indican el nº de capas del code-stream sin truncar (C) y en *V[]* se indican el sub-conjunto de capas de C que constituye un code-stream truncado (*C_part*). En un principio se considera que no se está enviando ninguna capa, quedando representado por un valor igual a [0, ···, 0] en *V[]*. La estructura de datos *E[]* contiene un determinado *V[]*, que también se inicializa, y más datos calculados sobre la reconstrucción del vídeo (L^q) según las capas que dicho parámetro *V[]* indica, como son: la pendiente (S), el consumo de bit-rate (R) y la distorsión (D). En esta misma línea de pseudocódigo se representa la estructura de datos *E[V[], S, R, D]* como *E[V[]]* para indicar que el valor que se está inicializando es *V[]*. Lo mismo ocurre en la *línea 2* donde la misma estructura se representa como *E[S, R, D]* indicando que los valores asignados son sólo estos tres. Dicha inicialización consta de tres asignaciones: la pendiente, que se iguala a un valor mínimo, como podría ser 0 o un nº negativo; el bit-rate que se igualaría a 0, al no haberse transmitido aún nada, no hay ancho de banda consumido; y por último la distorsión que debe ser ahora máxima, dado que al no enviar nada la reconstrucción será peor en calidad que la que pueda generar cualquier truncado del code-stream.

En la *línea 3* se encuentra el bucle que condiciona el nº de iteraciones del algoritmo de BRC, el cual es útil iterar mientras queden capas para enviar y se disponga de ancho de banda B. Para la primera condición, se conocen el nº de capas totales en el parámetro *Q[]*, donde siempre debe existir alguna sub-banda (posición del vector) que sea mayor estricto que su correspondiente sub-banda o campo de movimiento en *V[]*.

En el bucle *for* se evalúan tantos *V[]*, entonces llamados *V_test*, como sub-bandas y campos de movimiento contenga el code-stream. Cada *V_test* consiste en la adición de una capa al code-stream representado por *V[]*. Así iterativamente de las líneas *7* a la *10* la función *extracting_layers* extrae code-streams truncados (*C_part*), que son reconstruidos (L^q) para calcular el ancho de banda usado, la distorsión y la pendiente del *V_test* evaluado.

En la *línea 12* la estructura de datos *E[]* se mantiene actualizada conteniendo el *V[]* y demás datos asociados a la mayor pendiente (S) encontrada hasta el momento. De forma que tras finalizar dicho *for* la mejor opción, entiéndase como el *V_test* que proporciona el code-stream que genera el vídeo reconstruido con una mayor pendiente en una curva R/D, es almacenada, en el fichero *E_list* (*línea 16*) para su posterior consulta durante el envío.

Input	
L^0	Original video
t	Duration of L^0
B	Available bandwidth
C	Complete code-stream
T	Number of TRL of the C
$Q[]$	Total number of layers of each sub-band
$Q[_i]$	Total number of layer of a sub-band
Output	
E_list	File with the list of optimal order
Temporal Structures	
C_part	Truncated one part C
L^q	Reconstructed video from C_part
$V[]$	Number of layers in each sub-band of a C_part
$V\_test[]$	Another instance of V[]
$V[_i]$	Number of layers in a sub-band
S, R, D	Slope, Bit-Rate and Distortion of a L^q
$E[V[], S, R, D]$	Data of a particular C_part and L^q
min, max	Values for S and D

Table 1: Parámetros.

```
1  E[V[]] = V[] = create_V (T);
2  E[S, R, D] = min, 0, max;
3  while exits_layers_unsent (Q[], V[]) and B do
4  |   V[] = E[V[]];
5  |   for i=1 until i=T do
6  |   |   if Q[i] > V[i] then
7  |   |   |   V_test[] = add (V[], V[i] + 1);
8  |   |   |   C_part = extracting_layers (V_test[], C);
9  |   |   |   L^q = decompressor (C_part, T);
10 |   |   |   S, R, D = info (L^0, t, L^q, E[R, D]);
11 |   |   |   if S > E[S] then
12 |   |   |   |   E[] = V_test[], S, R, D;
13 |   |   |   end
14 |   |   end
15 |   end
16 |   E_list = save (E[]);
17 end
```

Algorithm 1: Pseudocódigo BRC.

Notar que el primer code-stream que es seleccionado para ser enviado, consta de una sola capa de la sub-banda L. Como puede verse en la Figura 2 la sub-banda L compuesta sólo por imágenes I, no depende de ninguna otra sub-banda, y por ello, la primera opción resulta ser siempre ésta.

IV. Evaluación

La evaluación del algoritmo BRC se ha llevado a cabo con code-streams generados por MCJ2K. Además para ubicar la relevancia de los resultados se han comparado las curvas R/D dadas por el codec H.264/SVC. Para proporcionar una comparación realista se han considerado todas las posibles variables que afectan a ambos codec, con el fin de igualar condiciones. Así se han usado igual nº de TRL e imágenes, tamaño de GOP, ancho de banda y escalabilidad (que ambos codec permitan truncar el code-stream el mismo nº de veces). Aunque en este último aspecto, SVC no nos ha proporcionado tanta escalabilidad como MCJ2K en resoluciones altas, como es FULL-HD, ya que produce un error de computo al intentar codificar un vídeo a más de 4 puntos de truncadoEn resoluciones pequeñas (CIF), representada en la Figura 4 Fila 1, si se ha obtenido una escalabilidad comparable entre ambos codec.

En la Figura 4 se muestran los resultados de la evaluación de nuestro algoritmo BRC. En cada gráfica se muestran 3 curvas R/D. Dos curvas del codec MCJ2K *con* y *sin* BRC, en comparación con la curva dada por H.264/SVC. A excepción de las gráficas de la Figura 4 Fila 2, donde en lugar de una curva para SVC, se muestran dos, con 3 y 4 puntos de truncado, con intención de dar una visión del codec SVC más detallada, dado que no se ha podido equiparar la escalabilidad con MCJ2K en resolución alta. Las curvas para MCJ2K *sin* BRC se han obtenido extrayendo el mismo nº de capas de todas las sub-bandas, siendo el primer punto de truncado la reconstrucción de sólo las primeras capas. Esto es equivalente a codificar y extraer el vídeo completamente con un determinado nivel de cuantificación.

Como puede verse en la Figura 4 la aplicación de nuestro algoritmo de BRC siempre mejora la eficiencia en el envío de datos del codec MCJ2K. Esta mejora es máxima, a mínimo ancho de banda y nula en el máximo ancho de banda, coincidiendo así los últimos puntos de las curvas R/D de *con* y *sin* BRC. Esto se debe a que en máximo bit-rate se envían todas las partes del code-stream, resultando la reconstrucción completa del mismo. El decremento en esta eficiencia, respecto de no usar BRC, no es lineal, ya que el aporte de cada capa depende de parámetros de entrada y de las propias características del vídeo.

La cuantía de los beneficios de la optimización presentada aumentan de forma lineal al nº de TRLs y capas usado para la compresión (en nuestros ejemplos usamos 8 capas de calidad para las texturas y 1 para los vectores de movimiento), puesto que al estar compuesto de un mayor nº de partes, su ordenación y posibilidad de no envío de algunas de ellas tiene mayor relevancia. Esto queda de manifiesto en la Figura 4 Columna 1, donde a 3 TRL el ancho de banda mínimo necesario para transmitir cada vídeo (CIF y FULL-HD, respectivamente) se reduce al usar BRC de 123.264 a 68.112 Kbps y de 301.824 a 228.816 Kbps. Estas mejoras aumentan en los ejemplos a 5 TRL (véase la Figura 4 Columna 2), donde el mínimo ancho de banda necesario al usar BRC pasa de 103.538 a 20.724 kbps (vídeo CIF), y de 184.602 a 66.112 kbps (vídeo FULL-HD). Es decir, se consigue reducir a menos de la mitad el ancho de banda necesario para reconstruir los vídeos a su mínima calidad disponible, haciendo posible su transmisión bajo restricciones de bit-rate donde antes no se podría reproducir nada.

Desde el punto de vista de la calidad del vídeo, se puede valorar que todo este bit-rate ahorrado puede ser invertido en disminuir la distorsión usando un bit-rate similar. Así especialmente para 5 TRL (Figura 4 Columna 2), usando un ancho de banda similar obtenemos una disminución de la distorsión, gracias al uso del BRC, desde 28.034 a 22.684 dB y de 10.372 a 8.958 dB, para los vídeos a baja y alta resolución respectivamente.

Además el BRC presentado consigue explotar toda la escalabilidad del codec MCJ2K. Esto puede observarse en la Figura 4, comparando el mayor nº puntos en las curvas *con* que *sin* BRC. El nº de puntos de truncado para todos los code-stream de MCJ2K sin

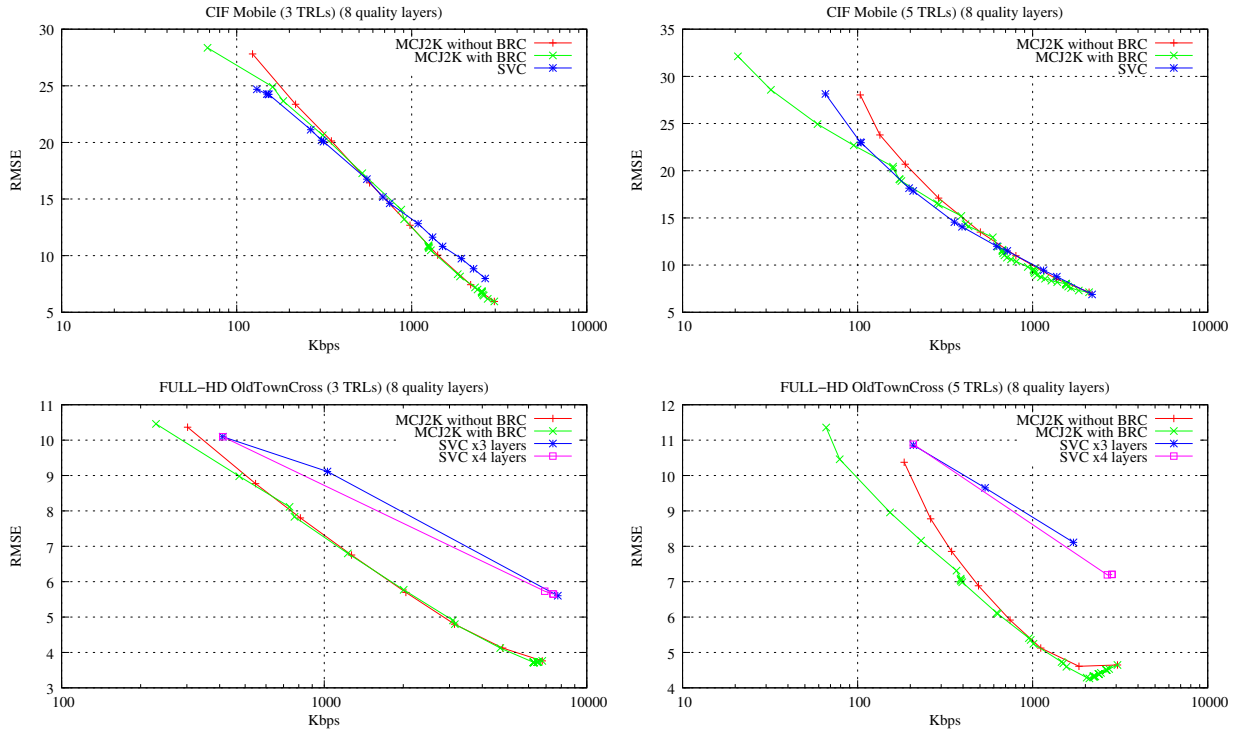

Fig. 4. Comparación entre nuestro BRC y el de H.264/SVC. Vídeos de baja y alta resolución (filas), de 3 y 5 TRL (columnas).

BRC es de 8. Gracias al uso del BRC se consiguen 26 y 44 puntos de truncado para 3 y 5 TRL, respectivamente. En el caso de SVC se obtienen 16 y 12 puntos de truncado a baja resolución con 3 y 5 TRL, respectivamente. Y 4 puntos a alta resolución, sea cual sea el valor de TRL. Así, especialmente en altas resoluciones la escalabilidad de los code-stream generados por MCJ2K con BRC, mejora un 69% y 81% para resoluciones bajas y altas, respectivamente. Esto representa una mayor escalabilidad de MCJ2K con BRC respecto a SVC de: 38% y 73% en los ejemplos de la Figura 4 Fila 1. Y aumento de un 85% y 91% para la Fila 2.

V. Conclusiones

MCJ2K carecía de un control de bit-rate que le permitiera optimizar la calidad de sus reconstrucciones en code-stream escalables, como sí tienen codec como H.264/SVC, con el que comparamos nuestra propuesta de BRC, que en este trabajo, aplicamos a code-stream generados por MCJ2K. Aunque es aplicable a otros tipos code-stream escalables.

La aplicación de nuestro BRC nunca provoca un empeoramiento respecto de no usarlo. Por tanto, siempre es provechoso su uso en términos de maximizar la calidad del vídeo. Se ha conseguido poder transmitir vídeo necesitando cinco veces menos ancho de banda. Además, el uso del algoritmo de BRC posibilita explotar completamente toda la escalabilidad proporcionada por el codec MCJ2K, superando hasta en un 91% a H.264/SVC.

Sin embargo, todas estas ventajas tienen un coste en tiempo de codificación del vídeo, donde para obtener el orden óptimo se requiere de iterativas extracciones y reconstrucciones.

Como línea de trabajo futuro es interesante aplicar planteamientos que mejoren la compresión máxima, en esta línea se puede investigar la compresión escalable de los vectores de movimiento. Junto a sus resultados se obtendrá además una mayor escalabilidad del code-stream a la actual.

Referencias

[1] The Joint Photographic Experts Group, *ISO/IEC 15444-1 (JPEG2000, Part 1)*.

[2] A. Skodras C. Christopoulos and T. Ebrahimi, "The jpeg2000 still image coding system: An overview," *IEEE Transactions on Consumer Electronics*, vol. 46, pp. 1103–1127, 2000.

[3] D. Marpe H. Schwarz and T. Wiegand, "Overview of the scalable video coding extension of the h.264/avc standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, pp. 1103–1120, 2007.

[4] T. Wiegand and G.J. Sullivan, "Overview of the h.264/avc video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, pp. 560–576, 2003.

[5] R. Leung and D. Taubman, "Transform and embedded coding techniques for maximum efficiency and random accessibility in 3d scalable compression," *IEEE Transactions on Image Processing*, vol. 14, pp. 1632–1646, 2005.

[6] D. Taubman and J. Thie, "Optimal erasure protection for scalably compressed video streams with limited retransmission," *IEEE Transactions on Image Processing*, vol. 14, pp. 1006–1019, 2005.

[7] D. Gibson and M. Spann, "Multi-frame motion estimation: Application to motion compensated prediction," in *Proceedings of the 1999 IEEE International Symposium onCircuits and Systems, ISCAS 99*, July 1999, vol. 4, p. 54–57.

[8] A. Secker and D. Taubman, "Lifting-based invertible motion adaptive transform (limat) framework for highly scalable video compression," *IEEE Transactions on Image Processing*, vol. 12, pp. 1530–1542, 2003.

[9] M. Antonini T. Andre, M.o Cagnazzo and M. Barlaud, "Jpeg2000-compatible scalable scheme for wavelet-based video coding," *EURASIP Journal on Image and Video Processing*, vol. 2007, 2007.