FSVC: Un Nuevo Codificador Escalable de Vídeo

M. F. López, V. G. Ruiz, S. G. Rodríguez, J. M. Dana, J. P. Ortiz y I. García
Departamento de Arquitectura de Computadores y Electrónica
Universidad de Almería

Resumen

FSVC (Fully Scalable Video Codec) es un sistema de compresión escalable de vídeo capaz de generar *streams* de vídeos comprimidos con escalabilidad de calidad, temporal y espacial. FSVC es adecuado para aplicaciones *VoD* (Video-on-Demand) en las que un servidor suele proveer vídeo a muchos clientes con distintos requisitos de características de visualización y ancho de banda. FSVC produce un stream comprimido capaz de satisfacer estos requisitos gracias a sus propiedades de escalabilidad. Los resultados experimentales obtenidos muestran que la eficiencia de codificación de FSVC es similar o superior a la de otros codificadores de vídeo escalable.

1. Motivación

La codificación escalable de vídeo es una técnica que permite que un stream de vídeo comprimido sea descodificado de distintas formas. Los clientes de vídeo reciben partes del stream del vídeo comprimido según sus requerimientos: frame-rate, resolución espacial, calidad de imagen y/o bit-rate. El frame-rate se obtiene por medio de la escalabilidad temporal; la escalabilidad espacial proporciona un conjunto de resoluciones espaciales de las imágenes; la mejora progresiva de la calidad de la imagen se obtiene por la escalabilidad de calidad. Estos tipos de escalabilidad pueden combinarse y determinar un bit-rate que también puede estar limitado por la capacidad de cómputo del descodificador o por el ancho de banda del canal de transmisión. Por lo tanto, se puede generalizar la idea de escalabilidad al concepto de escalabilidad de bit-rate.

La codificación escalable es una característica fundamental para los sistemas de almacenamiento y transmisión de vídeo. Por ejemplo, en las aplicaciones VoD, de vídeo por demanda, el computador servidor envía un stream de vídeo a un conjunto de clientes a través de enlaces de transmisión de datos digitales. En muchas situaciones la calidad, resolución y frame-rate de las visualizaciones que requiere el cliente deben adaptarse también a las características del descodificador y al ancho de banda de transmisión disponible en cada enlace. En este contexto, con codificación convencional de vídeo, los requerimientos computacionales necesarios en los servidores son proporcionales al número y tipos de clientes puesto que o bien (1) se crea una copia específica del vídeo comprimido cumpliendo los requerimientos de cada tipo de cliente, o bien (2) se utilizan técnicas de *transcoding* para procesar el vídeo comprimido. La técnica (1) eleva las necesidades de almacenamiento; la solución (2) requiere altas capacidades de CPU y memoria que permitan el procesado en tiempo real de los streams de vídeo comprimido. La codificación escalable de vídeo resuelve ambos problemas puesto que sólo es necesario mantener una copia del stream de vídeo almacenada en el servidor y el preprocesamiento o transcoding se simplifica. El preprocesamiento consistiría sólo en una reordenación en la forma de transmitir los datos comprimidos. Es una tarea tan sencilla que puede ser realizada por el propio cliente, que seleccionaría qué porciones del stream han de enviarse.

En este trabajo presentamos el esquema de un nuevo codec de vídeo escalable que hemos denominado *FSVC* (Fully Scalable Video Codec; Codificador de Vídeo Totalmente Esca-

lable) y mostramos la evaluación de su rendimiento como compresor de secuencias de vídeo digital.

2. Descripción de FSVC

El compresor de vídeo que proponemos es un sistema de codificación diferencial. Como cualquier otro codec de vídeo, disminuye la redundancia temporal del vídeo y la redundancia espacial de cada frame de la secuencia de vídeo. FSVC produce un stream comprimido que es altamente escalable. El descodificador FSVC recupera el vídeo exactamente idéntico al original cuando recibe completamente los datos comprimidos. En la Figura 1 se puede ver el esquema de bloques funcionales de FSVC.

La secuencia de imágenes de entrada primero se divide en *GOFs* (Group of Frames). Cada GOF consta de varias imágenes consecutivas y es codificado de forma independiente al resto de GOFs. Esto permite que el descodificador (1) pueda descodificar las imágenes de cada GOF sin procesar el resto de GOFs y (2) limite en el tiempo el error de drift en las situaciones en las que el descodificador no recibe totalmente el vídeo comprimido.

La reducción de la redundancia temporal se realiza con un esquema de codificación diferencial MCTF (Motion Compensated Temporal Filtering [1]) modificado. El movimiento entre las distintas imágenes de la secuencia se modeliza de forma convencional: mediante vectores que indican el movimiento de bloques cuadrados (de tamaño fijo) que forman las imágenes. Estos vectores de movimiento los calcula el módulo ME en el dominio de la imagen. Para cada frame $I[i]$ el módulo MC de compensación del movimiento genera una imagen predicción $P[i]$. MC usa los vectores de movimiento $M[i]$ obtenidos por el módulo ME para crear la mejor predicción posible que aproxima el frame actual a codificar. El proceso de codificación predictiva se lleva a cabo en el dominio transformado wavelet para (1) minimizar el desagradable efecto de blocking a bit-rates bajos cuando esta operación se realiza en el dominio de la imagen y para (2) eliminar el error de drift en la escalabilidad espacial [2]. Por esto, cada frame $I[i]$ se transforma usando la DWT-2D antes de eliminar la redundancia temporal. La predicción está formada por los bloques obtenidos compensando el movimiento en las imágenes de referencia, que también están en el dominio wavelet. En la determinación de los bloques se usa la fase correcta para evitar la no invariabilidad respecto al desplazamiento de la DWT [3]. En cada nivel de resolución se utilizan los mismos vectores de movimiento. La imagen a codificar y su predicción se restan, obteniendo así la secuencia de residuos E, que tiene menor entropía que la secuencia de imágenes originales I.

La codificación diferencial MCTF se realiza en el dominio wavelet. El frame predicción está compuesto de bloques wavelet de una imagen previa en la secuencia de vídeo (bloques-P) y por bloques de una imagen futura (bloques-N), como se muestra en la Figura 2. Cuando no es posible generar la predicción, se utiliza un bloque intra (bloques-I). La Figura 2 muestra cómo se genera la escalabilidad temporal. Por ejemplo, el frame 5 se predice con los frames 1 y 9; el frame 3 se predice con los frames 1 y 5; y así hasta que se procesan todos los frames del GOF. Cada bloque de un frame predicho, puede ser predicho hacia delante (desde un frame anterior) o hacia detrás (desde un frame futuro). La elección entre predicción hacia delante o hacia detrás se hace (1) minimizando el error cuadrático medio *MSE* (Mean Square Error) y (2) minimizando los errores de drift. Los errores de drift se acumulan a lo largo de las dependencias entre los frames predichos. Por lo tanto, para el frame predicho 2, las predicciones hacia delante tienen mayor prioridad que las predicciones hacia detrás porque, en el descodificador, el frame 1 (cuyos bloques son todos intra) será reconstruido con mayor o menor calidad pero sin errores de drift, ya que es un frame que se codifica independientemente de los demás.

Nótese que los frames de cada GOF se predicen con 2 frames intra (frames 1 y 9 de la Figura 2). Esta característica no está permitida en los conocidos estándares MPEG. Pero, para marcos de trabajo de vídeo escalable

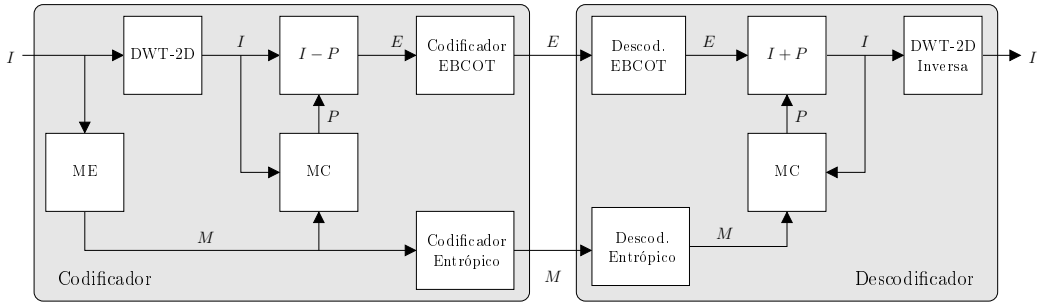

Figura 1: Diagrama de bloques de FSVC.

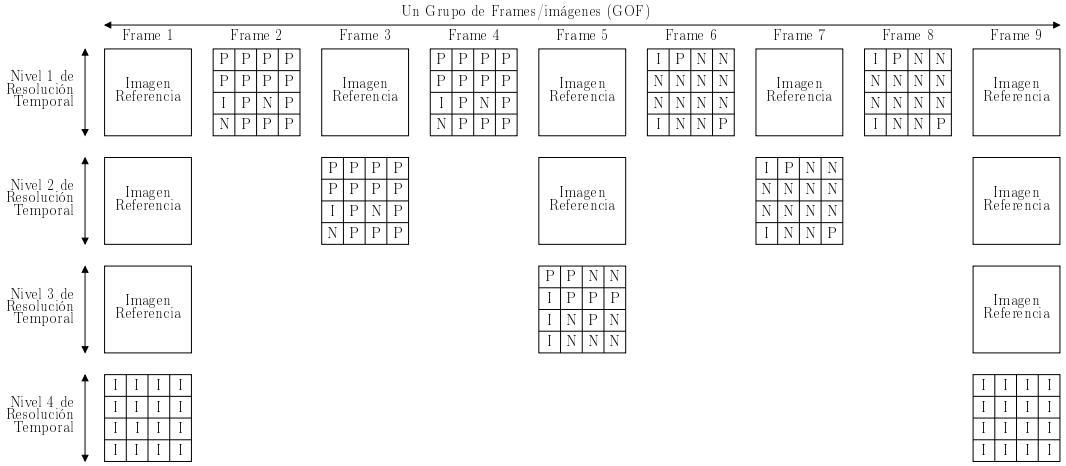

Figura 2: Ejemplo del esquema de codificación diferencial para un GOF de 8 imágenes (sólo se describe para una subbanda de la DWT). El frame 9 está compartido con el siguiente GOF.

como el que nos ocupa, esta técnica permite mejores reconstrucciones en el descodificador porque los frames intra son compartidos por 2 GOFs.

Cada residuo $E[i]$ se comprime usando el codificador entrópico EBCOT [4] del estándar de compresión de imágenes JPEG 2000. De esta forma se genera una colección de paquetes que forman parte del stream del vídeo comprimido. Cada paquete es la contribución de un precinto P (una región espacial de la imagen) a una capa de calidad L (un nivel de distorsión), para una resolución espacial R, para una componente de color C y para un GOF G. El stream de paquetes es fácilmente reordenable porque la información que permite localizar cada paquete también va incluida en el stream.

La información relativa al movimiento M se comprime entrópicamente. Se ha utilizado un codificador Huffman para eliminar la redundancia estadística de los vectores de movimiento similar al de los estándares MPEG. Estos datos han de ser descomprimidos completamente por el descodificador para poder reconstruir el vídeo.

3. FSVC en Sistemas VoD

La mayoría de los sistemas VoD se implementan usando una arquitectura cliente-servidor. El stream de vídeo comprimido se almacena en el servidor y los clientes recuperan los datos (o parte de ellos) para, descodificándolos, visualizar la secuencia de vídeo. La codificación escalable de vídeo es extremadamente útil en esta clase de aplicaciones ya que, normalmente, la transmisión completa del stream de vídeo comprimido es imposible. Gracias a las técnicas de escalabilidad, el sistema VoD seleccionará qué paquetes han de ser transmitidos en cada momento según las circunstancias: características de las peticiones de los clientes y ancho de banda disponible en los enlaces de comunicación.

En el codec FSVC, la salida generada por el compresor es una secuencia de paquetes que se colocan en el stream de vídeo comprimido en un orden determinado. El orden de transmisión de los paquetes es importante porque determina las características de las imágenes de vídeo descodificadas cuando sólo una parte del stream es recibida por el cliente. En nuestro sistema FSVC el compresor selecciona el orden (también llamado *progresión*) dependiendo de los requerimientos de visualización:

1. La **Progresión por calidad** se usa si los clientes necesitan preservar la resolución espacial y frame-rate originales pero permitiendo la variación contínua de la calidad de las imágenes que forman cada GOF en función del bit-rate (o del ancho de banda) disponible. Cualquier progresión subordinada a GL- puede ser usada con este propósito porque los paquetes se organizan en el stream primero por GOF (G) y después por capas de calidad (L).

2. La **Progresión por resolución** es proporcionada por cualquier ordenación subordinado a GR-. Usando estas progresiones es posible descomprimir la secuencia de vídeo a un nivel de resolución inferior porque los paquetes de datos son enviados por nivel de resolución dentro de cada GOF.

3. La **Progresión por frame-rate** se usa cuando la mayoría de los clientes requieren visualizaciones que mantengan la resolución espacial del vídeo original a alta calidad. En este caso se deberían utilizar progresiones subordinadas a GT-.

Nótese que todas las progresiones propuestas son subordinadas a G-. Por lo tanto, es sencillo acceder aleatoriamente a los GOFs de la secuencia de vídeo porque pueden ser descomprimidos independientemente.

Como ya se ha explicado anteriormente, es importante resaltar que el ordenación por defecto de los paquetes puede ser modificado en tiempo real por los clientes dependiendo de sus propios requerimientos. Por ejemplo, si el número de precintos usado es lo suficientemente grande, un cliente puede recuperar los paquetes que mejoren (con respecto al resto de la imagen) la calidad de una región (estática o en movimiento) de interés (*ROI* Region Of Interest [5]) que aparezca en la secuencia de vídeo.

Tabla 1: Resumen de FSVC y RWMH: SF = Filtros espaciales, SRL = Niveles de resolución espacial, TF = Filtros temporales, TRL = Niveles de resolución temporal, MC-PA = Precisión de píxel, EoR = Codificación de los residuos, EoMI = Codificación de los vectores de movimiento.

	FSVC	RWMH
SF	Biortog. 9/7	Biortog. 9/7
SRL	4	1
TF	Bidirecc. 1/1	IPPP...
TRL	5	-
MC	Tam. bloq. fijo	Tam. bloq. fijo
MC-PA	1/1	1/8
EoR	EBCOT	SPIHT
EoMI	Cod. Huffman	Cod. Huffman

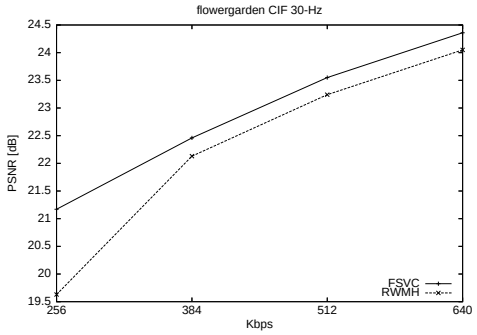

Figura 3: Valores medios de PSNRs para la secuencia *flower garden*.

4. Resultados experimentales

Se han realizado diversos experimentos para testear la eficiencia de la codificación escalable de FSVC que hemos implementado. Hemos empleado dos secuencias de vídeo ampliamente utilizadas en el campo de la compresión de vídeo: *flower garden* y *tempete*. El formato elegido ha sido CIF (352 x 288 píxeles) a 30 Hz de frame-rate (30 frames/sg). En los experimentos hemos comparado la eficiencia de codificación de FSVC con otro codificador de vídeo escalable que está siendo desarrollado en la actualidad y que se denomina RWMH (Redundant Wavelet Multi-Hypothesis Cui *et al.* [6]). Se presentan resultados de eficiencia de codificación y calidad visual en un escenario de progresión por calidad. Esta progresión es la más común en aplicaciones VoD y posee un grano de escalabilidad muy fino.

FSVC y RWMH presentan una complejidad computacional similar. La implementación de RWMH que han realizado los propios autores es software de libre distribución disponible en Internet. La Tabla 1 describe los parámetros empleados por FSVC y RWMH para comprimir las secuencias de vídeo de test.

En los experimentos realizados hemos comprimido la componente de luminancia de las secuencias de test *flower garden* y *tempete*. Las descodificaciones con ambos descompresores se han realizado siguiendo una progresión por calidad y, en concreto, en el caso de FSVC, utilizando la progresión GLTRCP. Para describir un poco mejor la configuración de FSVC añadir que cada GOF está formado por 16 frames, por lo que se dispone de 4 niveles de resolución temporal (4 TRLs). Cada frame se codifica usando 16 niveles de calidad y 4 niveles de resolución espacial (4 SRLs). Como sólo codificamos la componente de luminancia, $C = 1$ en GLTRPC. Se ha usado un precinto por nivel de resolución. Por lo tanto, cada capa de calidad está compuesta de $16 \times 4 \times 1 \times 1 = 64$ paquetes y cada GOF tiene $16 \times 64 = 1024$ paquetes. Los primeros n paquetes de cada GOF se descodificarán dependiendo del bit-rate disponible. Las descodificaciones se han efectuado a 256, 384, 512 y 640 kbps.

Las imágenes reconstruidas con FSVC han sido obtenidas comprimiendo a alta calidad y descodificando al bit-rate deseado. De esta forma, estas reconstrucciones han de presentar errores de drift, aunque minimizados debido al filtrado temporal que realiza FSVC. Sin embargo, el bit-rate es un parámetro que sólo se puede seleccionar en la parte del codificador RWMH, en el momento de la compresión; no puede ser elegido por el descodificador. Esto es debido a la implementación proporcionada

FSVC.

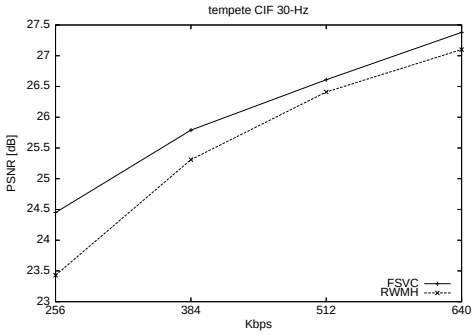

Figura 4: Valores medios de PSNRs para la secuencia *tempete*.

por el autor, no a limitaciones del esquema del codec, y causa el que las imágenes descodificadas con RWMH estén artificialmente libres de errores de drift. A pesar de ello, como se puede ver en las Figuras 3 y 4, FSVC muestra una eficiencia entre 0,5 y 1,0 dB por encima de los valores de PSNR medio obtenidos para RWMH. FSVC es especialmente superior a bajos bit-rates. El mejor rendimiento que demuestra FSVC posiblemente es debido principalmente a que (1) la eficiencia de codificación de EBCOT es superior a la de SPIHT (que es el algoritmo codificador empleado por RWMH) y a (2) el esquema de filtrado temporal que usa FSVC, en el cual cada frame intra es compartido por dos GOFs.

La Figura 5 muestra una comparación visual entre las reconstrucciones efectuadas con FSVC y RWMH. A la izquierda se pueden ver imágenes de *flower garden* y *tempete* reconstruidas con el descodificador FSVC a 256 y 512 kbps. A la derecha están ubicadas las obtenidas con RWMH. A 256 kbps FSVC es claramente superior. A 512 kbps ambos codecs parecen obtener calidad parecida, sin embargo, poniendo un poco de atención visual, algunos elementos están mejor reconstruidos con FSVC. La farola de *flower garden* y el tronco de madera vertical de *tempete* están mejor detallados en las imágenes descodificadas con

Figura 5: Comparación subjetiva visual entre FSVC y RWMH.

5. Conclusiones

FSVC está basado en el esquema de codificación escalable de vídeo "DWT-1D temporal en el dominio transformado". Se han intentado solucionar algunas deficiencias del esquema, además de añadir nuevas características deseables. FSVC utiliza elementos de JPEG 2000 y una variación de la técnica de filtrado temporal MCTF para eliminar la redundancia temporal en la señal de entrada al codificador. FSVC proporciona escalabilidad de grano fino por calidad del stream de vídeo comprimido que genera. Estas características son ideales para los sistemas de vídeo por demanda VoD y streaming de vídeo por Internet.

Además, los experimentos comparativos que hemos realizado con RWMH, uno de los codecs escalables de vídeo que está siendo desarrollado experimentalmente en la actualidad, demuestran que FSVC consigue mayor eficiencia de codificación, siendo la complejidad computacional de ambos codecs similar. Tanto el análisis objetivo como el subjetivo han mostrado la superioridad de FSVC.

Referencias

[1] S.-J. Choi and J.W. Woods, *Motion compensated 3-D subband coding of video*, IEEE Transactions of Image Processing, 1999.

[2] Y. Andreopoulos, M. Van der Schaar, A. Munteanu, J. Barbarien, P. Schelkens and J. Cornelis, *Fully-scalable wavelet video coding using in-band motion compensated temporal filtering*, Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003.

[3] M. J. Shensa, *The discrete wavelet transform: weeding the Á Trous and Mallat algorithms*, IEEE Transactions on Signal Processing, 1992.

[4] D. Taubman, *High performance scalable image compression with EBCOT*, IEEE Transactions on Image Processing, 2000.

[5] A. Skodras, C. Christopoulos and T. Ebrahimi, *The JPEG 2000 Still Image Compression Standard*, IEEE Signal Processing Magazine, 2001.

[6] S. Cui, I. Wang and J. E. Fowler, *Multihypothesis Motion Compensation in the Redundant Wavelet Domain*, Proceedings of the International Conference on Image Processing, 2003.