UNIVERSIDAD CARLOS III DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

INGENIERÍA DE TELECOMUNICACIÓN

PROYECTO FIN DE CARRERA

CONTROL DE TASA VBR EN DOS PASADAS PARA CODIFICACIÓN DE VÍDEO H.264/AVC

AUTOR: MIGUEL ÁNGEL FUENTE RODRÍGUEZ
TUTOR: MANUEL DE FRUTOS LÓPEZ

24 de junio de 2010

Título: *CONTROL DE TASA VBR EN DOS PASADAS PARA CODIFICACIÓN DE VÍDEO H.264/AVC*

Autor: *MIGUEL ÁNGEL FUENTE RODRÍGUEZ*

Tutor: *MANUEL DE FRUTOS LÓPEZ*

La defensa del presente Proyecto Fin de Carrera se realizó el día 24 de junio de 2010, siendo calificada por el siguiente tribunal:

Presidente: *Jesús Cid Sueiro*

Secretario: *Sergio Sanz Rodríguez-Escalona*

Vocal: *Francisco Valera Pintor*

Habiendo obtenido la siguiente calificación:

Calificación:

Presidente **Secretario** **Vocal**

Agradecimientos

Lo bueno de tirarse catorce años para terminar la carrera es que en ellos ha pasado mucha gente por tu vida, has aprendido de ellos y tienes muchísimo que agradecer (aquí el que no se consuela es porque no quiere...).
Es por esto que quizá estés leyendo estas páginas y te identifiques, pero no te veas mencionado. Si es así, perdoname. Te aseguro que ha sido un simple descuido.

No puedo empezar de otra manera que agradeciendo y dedicando este Proyecto Fin de Carrera a quienes más debo, y no son otros que mis padres, Conrado y Ángela, ya que sin ellos no estaría aquí. Gracias por vuestro apoyo y comprensión, por vuestra firmeza y paciencia (que vaya si habéis tenido...). Gracias por haber aguantado mis malos momentos, mis cabreos y todas esas veces en las que creía que había perdido el norte. Gracias por haber estado ahí siempre y por haber vivido mis progresos como vuestros.
Dicho esto, creo que es el momento de decir: *¡Papá, ya estoy arriba!*.

También quiero agradecer enormemente a mis hermanas, Chús y Montse, y a mis cuñados, Alfredo y Pedro, por su interés, su insistencia y su cariño y apoyo. Vosotros habeis sido en gran parte ese soporte que necesitaba para poder terminar lo que he llegado a creer que no tendría fin.
Un agradecimiento muy especial va para una persona que aunque todavía no sabe hablar ha hecho renacer en mi una ilusión de la que ya ni me acordaba. Es mi sobrino Rodrigo y, a su manera, me ha hecho sonreir incluso cuando no tenía ni motivos ni ganas para hacerlo. La verdad es que no se como agradecértelo, pero esto también es posible gracias a ti.

No me quiero olvidar del resto de mi familia, mis tías y tíos Nanchy, Chús, Agustín, Pili, Venancio y María Ángeles. Gracias por vuestro cariño y por la atención que me habéis prestado. Supone una felicidad muy grande saber que puedo contar con vosotros.
No puedo evitar dedicar y dar las gracias a tres personas que supusieron mucho para mí: mi abuela Narcisa, a mi tio Justo y a la Tata Pili. Os hizo mucha ilusión verme entrar en la universidad, pero no habeis podido llegar a este día. Dondequiera que estéis, gracias.

Gracias muy especiales a mi tutor, Manuel de Frutos, que ha sabido aconsejarme siempre que me he quedado atascado con algún problema, recibiéndome siempre que me pasaba por su

despacho (la mayoría de las veces sin avisar, como un *malqueda*) aunque estuviera hasta arriba de trabajo. Gracias por tu paciencia, por tus ánimos para acabar y por tu minuciosidad que me ha hecho aprender un montón de cosas.

Quiero dar las gracias también a mis amigos de Aranjuez, Alberto, Ana, Carlos, María, Mario, Cesar, Raúl, Javi, Jorge, Estanis y Segis. Sin vosotros esto no habría podido ser posible. Con vosotros he compartido casi toda mi vida; habéis aguantado lo que no está escrito, mis malos rollos y mis momentos de bajón, los buenos momentos y los buenísimos. Me siento un privilegiado por teneros a mi lado.
Por otro lado, también quiero dar las gracias a la gente del Ópera y del Cue (Alfonso, Paco, Isa, Monika, Polu, Luisma, Alba, Emilio, Toni, Jose, Fisu, Sergio, David, etc.) con la que he compartido tantas y tantas noches de trabajo y de fiesta. ¡Eso sí que ha sido desconectar!
También quiero recordar a la gente que, en cierto modo, ha vivido partes de mi carrera estudiando conmigo en la biblioteca: Carlos M., Kiko, Raquel, Edu, Justo, Cámara y Toni, gracias por todos esos momentos vividos en el Farnesio.

Y por supuesto, gracias a toda la gente de la universidad, Carlos, Edu, Emilio, Gele, Dani, Furfur, Nacho, Sara P., Molo, Raquel, Elisa, Gabi, Isaac, Guillermo, Sara G., Blanca, Cano, Ramón, Pablo, Iván, Teju, Patri, Sonia, Carlos G., Sacha, Sergi, Sergut, Pay y Jose. Habéis sido, con diferencia, lo mejor de la universidad y lo mejor que me llevo de aquí. Gracias por compartir esta etapa de vuestra vida conmigo, tantos buenos momentos, comidas, cenas, confidencias y por darme el mejor regalo: vuestra amistad. Aunque los kilómetros y el trabajo no nos han dejado vernos todo lo que me hubiera gustado, siempre os llevo conmigo.

Y finalmente, una dedicatoria muy muy muy especial a toda la gente que creía que no sería capaz de terminar esto y para los que creían que terminaría tirando la toalla (que, aunque pocos, los ha habido). Gracias por equivocaros. Llegar a este punto supone una satisfacción enorme, pero gracias a vosotros esta satisfacción es mucho mayor.

El trabajo que nunca se empieza es el que tarda más en finalizarse.

J. R. R. Tolkien

El hombre superior es el que siempre es fiel a la esperanza;
no perseverar es de cobardes.

Eurípides

Resumen

En este Proyecto Fin de Carrera se busca definir un algoritmo de control de tasa para H.264/AVC que permita la obtención de una secuencia de vídeo con una calidad lo más estable posible. Para ello se propone un mecanismo VBR de dos pasadas. En la primera pasada se calcularán las complejidades de cada cuadro, que representarán de alguna menera sus necesidades de tasa. Igualmente se hará una detección de cambios de escena mediante diferencia segunda de histograma.
En la segunda pasada se realizará un reparto de bits en cuatro niveles: escena, GOP, miniGOP y cuadro, utilizando las complejidades calculadas previamente. Estos bits se utilizarán en el modelo R-QP que nos dará el parámetro de cuantificación a utilizar en cada cuadro. Este modelo será cuadrático en los cuadros P y B y exponencial en los cuadros I, permitiendo éste último al mismo tiempo una estimación correcta de la QP inicial a emplear al inicio de la secuencia y de cada escena.

Los experimentos realizados muestran que se obtiene una calidad con variaciones suaves a corto plazo, lo que proporciona una calidad percibida satisfactoria. Al mismo tiempo, el volumen total de información generada se mantiene en torno al valor objetivo establecido.

Abstract

In this Master's Thesis we are looking for a rate control algorithm for H.264/AVC which allows the obtaining of an enhanced visual quality video sequence. In order to do this, we propose a two-pass VBR mechanism. In the first pass the complexities of each frame, which represent in some way their rate requirements, are calculated. Likewise, a shot detection will be performed using a second-difference of histogram based method.
In the second pass, a four level bit allocation (scene, GOP, miniGOP and frame) will be made using the frame complexity previously calculated. These bits will be used in the R-QP model, which will give the quantization parameter to use in each frame. This model will be quadratic for P and B frames and exponential in the case of I frames, making this one possible a correct estimation of the initial QP at the beggining of the sequence or scene.

The experimental results show that the quality achieved presents soft variations in the short term, which provide a satisfactory perceived quality. At the same time, the amount of information generated is near the specified target.

Índice general

Lista de Figuras

Lista de Tablas

Capítulo 1

Introducción

En la sociedad actual es innegable la importancia que tiene el vídeo como soporte de información, tanto en ámbitos empresariales (videoconferencias, seguridad, etc.) como de ocio (cine, televisión, etc.). Es destacable que desde la invención del cinematógrafo por los hermanos Lumière, hace más de un siglo, ha habido grandes esfuerzos dirigidos a la búsqueda de la mayor calidad posible en el registro y posterior representación de las imágenes. Una importantísima mejora relativamente reciente (hace menos de 30 años[1]) fue el paso del vídeo analógico al vídeo digital. Gracias a ella, la información de vídeo ha ganado en versatilidad, permitiendo cosas que antes ni siquiera eran imaginables.

Sin embargo, es obvio que una señal de vídeo transporta muchísima información, que debe ser tratada adecuadamente. Una gran cantidad de información se traduce en un gran ancho de banda requerido. Según la recomendación ITU-R BT.601 (o CCIR 601), un vídeo de definición estándar, PAL ó NTSC[2], digitalizado usando un muestreo 4:2:2 y 8 bits por muestra (lo que viene a ser una calidad normal para televisión) requiere un bit rate de 216 Mbps. Este ancho de banda es, hoy en día, inabordable a efectos prácticos. Basta con observar que las ofertas de los distintos proveedores españoles de conexión a Internet terminan en 50 Mbps (aunque hay pruebas de 100 Mbps). Además, las conexiones de área local inalámbricas no dan tasas tan elevadas. Todo esto con el agravante de que sólo estamos teniendo en cuenta el vídeo. No estamos incluyendo el sonido, subtítulos ni ninguna otra información adicional.

Es aquí donde entra otra gran innovación: la compresión de vídeo. Se basa en dos principios:

- No toda la información presente en una secuencia de vídeo es perceptible por el ser humano
- Una secuencia de vídeo tiene muchísima información redundante

Esto se traduce en que existe la posibilidad de reducir esos *monstruosos* 216 Mbps sin perder

[1] Se considera que el vídeo digital irrumpió en 1983 con el formato D-1 de Sony

[2] PAL: 625 líneas (área activa: 720x576), 25 Hz, entrelazado; NTSC: 525 líneas (área activa: 720x480), 30 Hz, entrelazado

calidad, al menos aparentemente.
Para ello, se buscan las redundancias en la información presente en el vídeo, que serán de tres tipos: espacial, temporal y estadística.
Explotando estas propiedades podemos reducir drásticamente la tasa binaria requerida para la transmisión y/o almacenamiento de secuencias de vídeo digital. Las diferentes estrategias aplicadas para la reducción del ancho de banda requerido para vídeo conforman los diferentes estándares de compresión de vídeo que han ido apareciendo a lo largo del tiempo, cada vez proporcionando más calidad de imagen y/o mayores tasas de compresión.

Desde el punto de vista de las aplicaciones, estos avances conllevan la aparición de numerosas aplicaciones y/o dispositivos que hacen uso de estas técnicas de codificación de vídeo. Una de ellas es el almacenamiento de vídeo. Éste se ha visto impulsado por la aparición de dispositivos (de captura, reproducción y almacenamiento) cada vez más pequeños y de mayor calidad, además de más baratos. Además, los medios de almacenamiento han experimentado un gran desarrollo, que ha hecho que tengan mayor capacidad, sean más rápidos, más pequeños y tengan un menor consumo. Esto hace que el vídeo esté cada vez más presente en la realidad cotidiana, reforzándose así la necesidad de mejorar las técnicas existentes.

Por otro lado, el vídeo también se transmite. El desarrollo de la Web 2.0 está haciendo que el vídeo sea protagonista en la Red. Servicios como YouTube y redes sociales como Facebook, o MySpace están haciendo cada vez un uso mayor del vídeo en su oferta. Los sistemas de difusión también están haciendo un uso intensivo del vídeo digital, como pueden ser la televisión a través de Internet (o IPTV), la difusión por satélite y la TDT. En los todos los casos es necesario controlar el consumo de ancho de banda, pero sobre todo es necesario en estos dos últimos. En un satélite el ancho de banda suele estar limitadísimo (además de que es muy caro) y en TDT el espectro es escaso y hay que optimizarlo lo más posible. Además, aunque despacio, se están introduciendo servicios de videoconferencia, bien a través de Internet o a través de telefonía móvil. En este último caso, el control de la calidad y del ancho de banda resultan críticos por la limitadísima tasa disponible en un canal radio móvil.

Por último, hay que resaltar un último avance en cuanto al vídeo se refiere, que es la aparición en escena de la alta definición (o Vídeo HD). Este servicio está cada vez más demandado y tiene consumos de ancho de banda enormes. Por ello es necesario una compresión de vídeo lo más eficiente posible, ya que es difícil meter un vídeo de estas características en un canal de comunicaciones o en un soporte físico como Blu-Ray.

A todo lo anteriormente expuesto hay que añadir que el usuario final cada vez es más exigente, por lo que esta carrera tecnológica parece no tener fin. Todo lo presente es mejorable y la evolución en este campo, por lo tanto, es continua.

De todos los sistemas de compresión que se han desarrollado hasta la fecha, el más reciente y de mayor eficiencia es el denominado H.264 / MPEG-4 AVC [1], algoritmo propuesto por los grupos MPEG, de la ISO, y VCEG, de la ITU.

En este Proyecto Fin de Carrera se trabajará sobre el problema del bit rate y de la calidad en el estándar H.264, cuyo uso está cada vez más extendido en una multitud de aplicaciones.

El objetivo de este PFC será diseñar y validar un mecanismo de control de tasa que maximice la calidad percibida. Esta percepción estará basada en la estabilidad de la calidad, esto es, que la calidad sea homogénea, sin saltos evidentes, a lo largo de la secuencia. La calidad individual de cada cuadro será algo secundario, siempre y cuando ésta sea similar en todos los cuadros. Esto quiere decir que un observador valorará mejor una secuencia cuyos cuadros tengan una calidad pobre, pero siempre la misma, que una escena con cuadros de calidad muy variable, unas veces muy alta y otras veces baja. Al mismo tiempo de esto, se persigue que la secuencia comprimida obtenida cumpla la condición de que tenga un tamaño lo más próximo posible a un tamaño objetivo.

Una aplicación tipo necesitada de un enfoque así es el almacenamiento de vídeo. Se necesita que la experiencia de visualización sea lo más satisfactoria posible para el espectador; esto obliga a una estabilidad en la calidad. Por otro lado, la secuencia de vídeo se almacena en un soporte; esto obliga a que el volumen total de información que salga del codificador debe estar controlado para ajustarse al espacio disponible en dicho soporte. Este control debe ser estricto en cuanto que la secuencia no puede ocupar más espacio del especificado (no cabría en el medio de almacenamiento, lo desbordaría) ni menos (si se genera menos información se puede presuponer que la calidad será menor).

En cuanto al contenido de este Proyecto Fin de Carrera, se comenzará con una introducción en la que se explicarán conceptos generales (como el tratamiento del color, formatos y proceso de digitalización y medidas de calidad), los conceptos en los que se basa la compresión de vídeo y las herramientas más comúnmente utilizadas para su consecución y una introducción general al estándar en torno al cual gira este Proyecto: H.264/MPEG 4 AVC.

Seguiremos con un estudio en detalle del concepto de control de tasa, los tipos existentes y las particularidades que presenta el control de tasa en H.264/AVC. Se expondrán las técnicas más empleadas en los algoritmos existentes y varias propuestas que se han formulado para realizar cada una de ellas. Asimismo se comentarán otras técnicas adicionales que sin ser de control de tasa, favorecen una implementación más efectiva del mismo. Además, mencionaremos algunos algoritmos destacados por sus prestaciones y su enfoque diferente al resto.

A continuación describiremos el algoritmo de Rate Control propuesto en este Proyecto, su escenario de aplicación, su funcionamiento y las decisiones de diseño. Después, se expondrán las pruebas realizadas al mismo y sus resultados, así como comentarios a los mismos.

Finalmente se expondrán las conclusiones de este trabajo, así como los trabajos futuros para corregir fallos y debilidades descubiertos, así como mejorar su funcionamiento.

Por último se adjuntarán un presupuesto para este Proyecto Fin de Carrera y un anexo con los experimentos realizados de una manera más detallada.

Capítulo 2

El vídeo digital

Un flujo binario que representa una secuencia de vídeo comprimida es la culminación de un proceso largo y complejo en el que entran en juego multitud de conceptos, teorias y tareas.

En este capítulo se explicarán todas las herramientas necesarias para comprender en qué consiste el proceso de digitalización y compresión de una secuencia de vídeo. Finalmente se particularizará todo esto para el caso del algoritmo H.264/MPEG-4 AVC, con el objetivo de tener una base sólida sobre la cual poder explicar el proceso de control de tasa (en el capítulo 3) y el algoritmo propuesto en este Proyecto Fin de Carrera (en el capítulo 4).

2.1. Conceptos básicos

2.1.1. Espacios de color

Un espacio de color define un modelo de representación del color. La gama cromática percibida por el ser humano puede ser representada de múltiples maneras. En el sistema visual humano (HVS, *Human Visual System*) la percepción completa de la luz se basa en tres tipos de células sensibles[1], llamadas conos. Cada tipo de cono es sensible a una determinada banda del espectro electromagnético, presentando su respuesta picos a diferentes longitudes de onda. A estos conos se les llama L (sensibles a longitudes de onda largas, región del amarillo), M (sensibles a longitudes de onda medias, región del verde) y S (sensibles a longitudes de onda cortas, región del azul-violeta). En la figura 2.1 podemos ver la respuesta de todos los tipos de células sensibles a la luz.

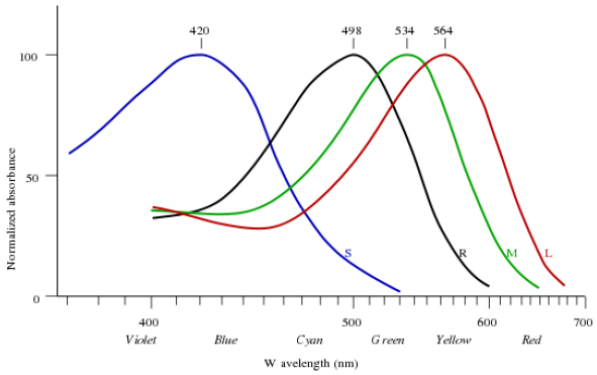


Figura 2.1: Respuesta de los tipos de células sensitivas de la retina. En negro (R), la respuesta de los bastones.

Partiendo de esto, resulta fácil considerar la representación de los colores como un sistema tridimensional en el que se parte de tres parámetros[2] que formarían base y que combinándolos pudiéramos obtener el resto de los colores.

Existen múltiples espacios de color, cada uno utilizado en un entorno o aplicación diferente. Entre los existentes podemos destacar:

2.1.1.1. Espacio HSV

Espacio cilíndrico, normalmente asociado a un cono. Consta de tres coordenadas:

- **Tonalidad (*Hue*):** Referido a la frecuencia dominante del color dentro del espectro visible. Es el color que se percibe (verde, rojo, azul...)

[1] En realidad hay cuatro tipos de células, 3 conos sensibles al color y otras células, llamadas bastones, sensibles al brillo en condiciones de baja iluminación. Éstas no son relevantes para el caso que nos ocupa

[2] La base de los colores no tiene por qué ser tres colores básicos, sino que puede ser de otra naturaleza

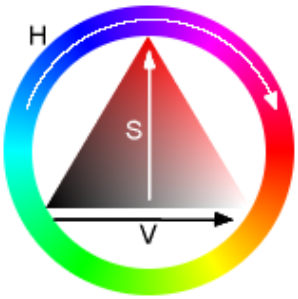


Figura 2.2: Representación de las componentes HSV

- **Saturación (*Saturation*):** Indica la cantidad del color, su viveza
- **Valor (*Value*):** Intensidad de la luz de un color (cantidad de blanco o de negro que posee un color)

2.1.1.2. Espacio CMYK

Basado en la absorción de la luz, trabaja con colores secundarios. Los colores primarios se obtienen sumando los secundarios. Es el sistema utilizado en las impresoras. La base de colores secundarios está formada por los colores Cian, Magenta y Amarillo. Se trata de un modelo de color sustractivo, ya que la suma de todos los colores da un color negro (en realidad es un negro turbio), y la ausencia de colores resulta en un color blanco. Adicionalmente se le añade un cuarto canal, llamado *Key*, que es el color negro, de manera que el contraste se ve mejorado y el negro obtenido es mejor que el resultante de sumar los tres colores CMY.

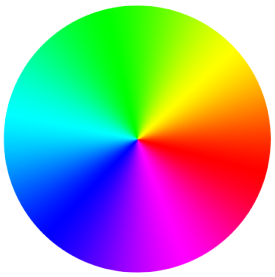


Figura 2.3: Representación de las componentes CMY

2.1.1.3. Espacio RGB

Se trata del espacio de color más sencillo conceptualmente y uno de los más empleados. Es un espacio de color aditivo, en el que partimos de una base de tres colores primarios, a saber, rojo (*Red*), verde (*Green*) y azul (*Blue*). El hecho de que sea aditivo hace que la suma de los tres colores de lugar al color blanco, mientras que la ausencia de ellos resulta en el color negro.

El hecho de que los colores elegidos para confeccionar la base sean los citados es debido a la respuesta de los tres tipos de células de tipo cono presentes en la retina del ser humano. Así podremos representar la imagen de una forma más acorde al funcionamiento del ojo humano.

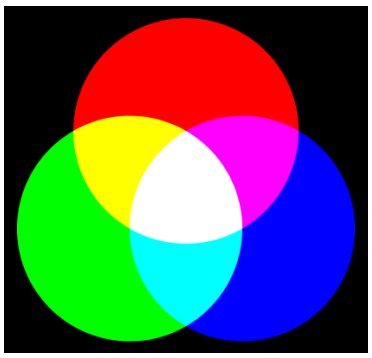
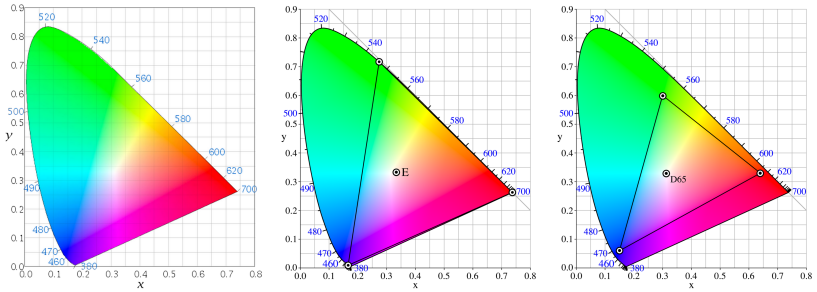


Figura 2.4: Espacio de color aditivo RGB

Al ser un espacio tan utilizado han surgido varias versiones para adaptarlo a diferentes necesidades, como sRGB, empleado en impresoras, monitores y en Internet, Adobe RGB, RGBA (RGB + Canal Alpha o de transparencia), etc. La representación de los espacios RGB y sRGB la podemos ver en la figura 2.5.



(a) CIE 1931 XYZ (b) CIE 1931 XYZ RGB (c) CIE 1931 XYZ sRGB

Figura 2.5: Representación del espacio de color CIE 1931 XYZ. En (b) y (c) e muestran las posiciones de los tres colores básicos y la posición del punto correspondiente al color blanco. Los colores fuera del triángulo no pueden ser mostrados.

Es un espacio de color ampliamente usado, pues muy adecuado para su uso en dispositivos de captura y representación de imágenes, puesto que podemos capturar cada color con un sensor por separado, con la única necesidad de filtrar previamente los colores. En los dispositivos de representación como pantallas LCD o tubos de rayos catódicos (o CRT) podemos descomponer cada píxel en tres *sub-pixels*, cada uno de un color, de manera que a una determinada distancia parece que los colores se mezclan, dando la sensación de un color verdadero. Un ejemplo de descomposición de una imagen en sus componentes RGB lo podemos ver en la figura 2.6

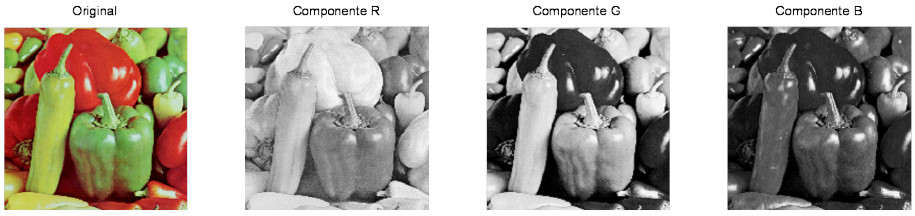


Figura 2.6: Descomposición en componentes RGB

2.1.1.4. Espacio YCbCr

Este espacio de color surge por dos motivaciones:

- Una vez aparecida la televisión en color es necesario mantener la compatibilidad con los antiguos televisores en blanco y negro
- El Sistema Visual Humano es menos sensible a la información de color que a la información de brillo

Este último motivo hace que con este espacio sea posible representar las imágenes en color de manera más eficiente que con otros esquemas como RGB, ya que podemos representar la información de brillo (o Luminancia, o *Luma*) con mayor resolución que la información de color. Esta información de color se representa como la la diferencia de color, llamada Crominancia o *Croma*, e indica la diferencia existente entre cada color y la Luminancia. Las componentes se pueden relacionar con RGB de la siguiente manera:

$$\begin{aligned} Y &= k_r \cdot R + k_g \cdot G + k_b \cdot B \qquad (k_r + k_g + k_b = 1) \\ Cb &= B - Y \\ Cr &= R - Y \\ Cg &= G - Y \end{aligned}$$

Observamos que pasamos de tres parámetros a cuatro, por lo que uno es redundante. Ya que la suma Cb + Cr + Cg es una constante, se ha optado por prescindir de la variable Cg, ya que puede ser obtenida a partir de las otras tres variables. Por lo tanto solo se consideran 3 variables: Y, Cb y Cr, como podemos ver en la figura 2.7. Si tenemos en cuenta que, como se ha mencionado anteriormente, el sistema visual humano es menos sensible a la información de color que a la de brillo, podemos representar la información correspondiente a la Crominancia a una resolución inferior, de manera que aparentemente la calidad no se ve afectada. De este modo se puede ahorrar mucho espacio de almacenamiento y/o ancho de banda a la hora de transmitir el vídeo. Esto no es posible utilizando el espacio RGB, ya que en éste las tres componentes tienen la misma importancia relativa, no hay una más relevante que otra, ni a efectos prácticos ni perceptuales. Esta propiedad hace que el espacio de color YCrCb sea muy adecuado para este tipo de tareas.

Sin embargo, el esquema RGB es muy utilizado, ya que es idóneo para la captura y representación de imágenes, por lo que se debe poder hacer una conversión entre ambos espacios de color. Ésta es la siguiente:

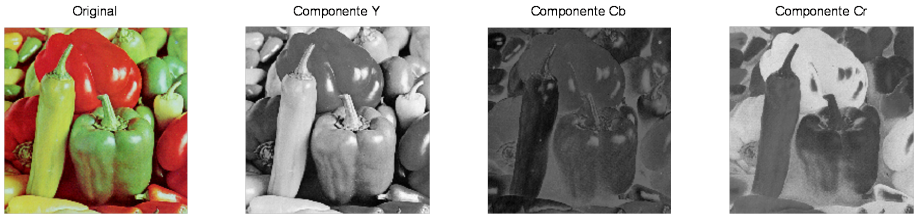


Figura 2.7: Descomposición en componentes YCbCr

$$\begin{aligned} Y &= k_r \cdot R + (1 - k_b - k_r) \cdot G + k_b \cdot B \\ Cb &= \frac{0{,}5}{1 - k_b} \cdot (B - Y) \\ Cr &= \frac{0{,}5}{1 - k_r} \cdot (R - Y) \end{aligned}$$

$$\begin{aligned} R &= Y + \frac{1 - k_r}{0{,}5} \cdot Cb \\ G &= Y - \frac{2k_b(1 - k_b)}{1 - k_b - k_r} \cdot Cb - \frac{2k_r(1 - k_r)}{1 - k_b - k_r} \cdot Cr \\ B &= Y + \frac{1 - k_b}{0{,}5} \cdot Cb \end{aligned}$$

Según la recomendación BT.601 de la ITU-R, las constantes k_b y k_r se definen como $k_b = 0{,}114$ y $k_r = 0{,}299$, por lo que las anteriores expresiones quedan simplificadas:

$$\begin{aligned} Y &= 0{,}299 \cdot R + 0{,}587 \cdot G + 0{,}114 \cdot B \\ Cb &= 0{,}564 \cdot (B - Y) \\ Cr &= 0{,}713 \cdot (R - Y) \end{aligned}$$

$$\begin{aligned} R &= Y + 1{,}402 \cdot Cr \\ G &= Y - 0{,}344 \cdot Cb - 0{,}714 \cdot Cr \\ B &= Y + 1{,}772 \cdot Cb \end{aligned}$$

Un aspecto importante que hay que recalcar es que existe un espacio de color denominado YPrPb que no hay que confundir con YCrCb. Son análogos. La diferencia estriba en que el espacio YPrPb está referido al mundo analógico, mientras que el YCrCb está referido a un entorno digitalizado. Añadir también otra fuente común de errores, que es identificar los espacios YCrCb y YUV de manera indistinta. Aclarar que el espacio YUV es un sistema analógico similar, pero con factores de escala distintos a los de YCbCr, por lo que no son la misma cosa.

2.1.2. Muestreo

Un primer paso imprescindible en el proceso de captura de una señal (de cualquier naturaleza) es el muestreo de la misma. La información presente en un vídeo presenta variaciones espaciales y temporales, que habrá que extraer con su correspondiente muestreo.

2.1.2.1. Muestreo temporal

Consiste en la toma de instantáneas de manera periódica, con una frecuencia determinada por la aplicación. A esta frecuencia de muestreo se le denomina frame rate. Cuanto mayor sea, más suaves serán los movimientos registrados en la secuencia de vídeo, pero también habrá que manejar una mayor cantidad de información. Según el valor del frame rate podemos clasificar:

- **Frame rate $<$ 10 fps[3]:** Movimiento poco o nada natural. Se percibe como si fuera "a trompicones". Útil para comunicaciones de muy bajo ancho de banda.
- **Frame rate entre 10 y 20 fps:** Tasa válida para comunicaciones con un ancho de banda limitado. Los movimientos se perciben más suaves, salvo en escenas con movimientos rápidos, en los que se puede percibir brusquedades en dichos movimientos.
- **Frame rate a 25 ó 30 fps:** Es el estándar usado en televisión, sólo que en este caso la sensación de suavidad en el movimiento se ve incrementada por el uso de la técnica de entrelazado, que se explicará más adelante.
- **Frame rate a 50 ó 60 fps:** Los movimientos son percibidos suavemente, a costa de un gran incremento en la cantidad de datos generados.

2.1.2.2. Muestreo espacial

Una vez tomada una instantánea de la escena, deberemos muestrearla espacialmente para poder establecer un flujo de información. La forma de muestrear una imagen (ya fija) se puede asumir al hecho de superponer una rejilla regular sobre dicha imagen, de manera que las muestras se toman en las intersecciones de dicha rejilla. Cuanto más fina sea la rejilla, mayor número de muestras se tomarán y, por lo tanto, mayor resolución tendrá la imagen capturada. A cada una de estas muestras se le denomina *Píxel*. En la figura 2.8, la imagen tiene superpuestas dos rejillas. La rejilla gris (más fina) generará una imagen de mayor resolución que la rejilla negra (más gruesa) por tener más puntos de muestreo.

Formatos de Muestreo YCbCr

El espacio de color YCrCb, utilizado para la creación del flujo de información, almacenamiento y transmisión, es susceptible de varios tipos de muestreo, en virtud de la menor importancia que

[3] Cuadros por segundo, *frames per second*

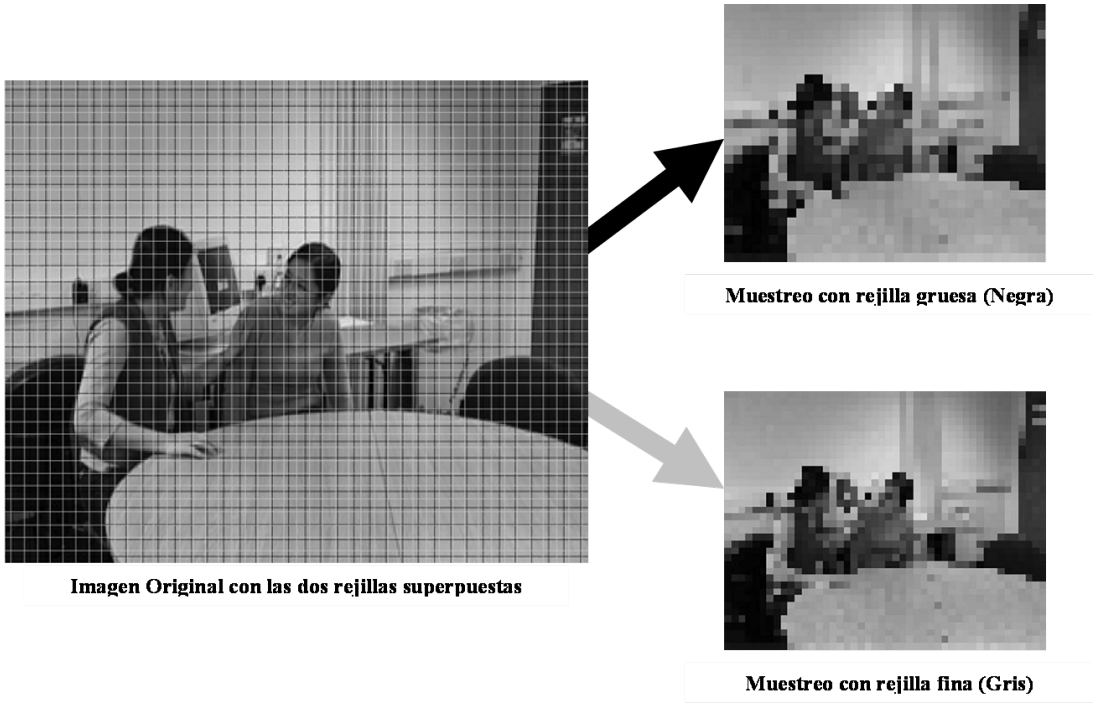


Figura 2.8: Rejilla de muestreo espacial

tiene, a nivel perceptual, la información del color (Crominancia o *Croma*, componentes "Cb" y "Cr") frente a la información del brillo (Luminancia o *Luma*, componente "Y"). En un píxel existen tres muestras, una por cada componente. La diferencia entre los distintos tipos de muestreo radica en el submuestreo que se realiza a la componente de Croma de cada píxel. La componente de Luma no se submuestrea. Los diferentes tipos de muestreo YCrCb, mostrados en la figura 2.9 son:

- **4:4:4** → Las tres componentes tienen la misma resolución. Cada píxel tiene una muestra de las tres componentes. No hay submuestreo de la componente de color.
- **4:2:2 (ó *YUY2*)** → Las componentes de crominancia tienen la misma resolución vertical, pero la mitad horizontal que la luminancia. Este esquema se usa para la reproducción de color de alta calidad.
- **4:2:0 (ó *YV12*)** → Las componentes de crominancia tienen la mitad de resolución horizontal y vertical que la luminancia. De este modo, un vídeo 4:2:0 requiere la mitad de muestras que si estuviera en formato 4:4:4. A pesar de la confusa notación, no hay que pensar que se esté eliminando una de las dos componentes. Este esquema es el usado en aplicaciones de consumo.

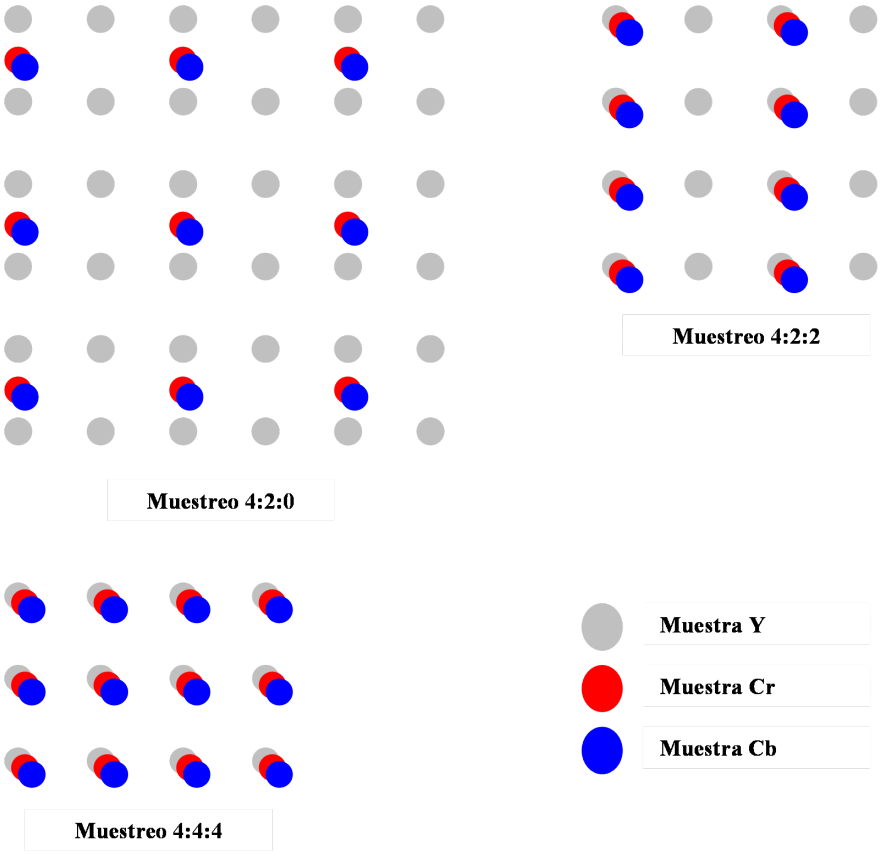


Figura 2.9: Formatos de muestreo YCbCr

2.1.3. Formatos de vídeo

En vídeo digital hay una gran variedad de formatos de cuadro, en función de su resolución y relación de aspecto. Dependiendo de la aplicación, se usará una familia de formatos u otra. En este apartado destacaremos tres familias principales: CIF, ITU-R Rec. BT.601 y el conjunto de resoluciones HD.

- CIF (*Common Interface Format*: Conjunto de formatos empleados para estandarizar las resoluciones de vídeo en aplicaciones de videoconferencia. Podemos ver las características de cada subformato en la tabla 2.1, así como una comparación de sus tamaños en la figura 2.10.

- ITU-R Rec. BT.601: Regula el formato empleado en la difusión de vídeo para televisión. Fija una frecuencia de muestreo de 13.5 MHz para la Luminancia y 6.75 MHz para la Crominancia para producir una señal muestreada en formato 4:2:2. Temporalmente la señal se muestrea a 25 Hz (en sistemas PAL / SECAM) o a 30 Hz (sistema NTSC). Este incremento de frecuencia temporal se compensa en una disminución de la resolución espacial, de manera que en ambos esquemas el bit rate total es de 216 Mbps.

Formato	Resolucion de Luminancia	Píxeles por cuadro	Bits por cuadro (asumiendo muestreo 4:2:0)
Sub-QCIF	128×96	12.288	147.456
QCIF	176×144	2.5344	304.128
CIF	352×288	101.376	1.216.512
4CIF	704×576	405.504	4.866.048
16CIF	1408×1152	1.622.016	19.464.192

Tabla 2.1: Características de los formatos CIF

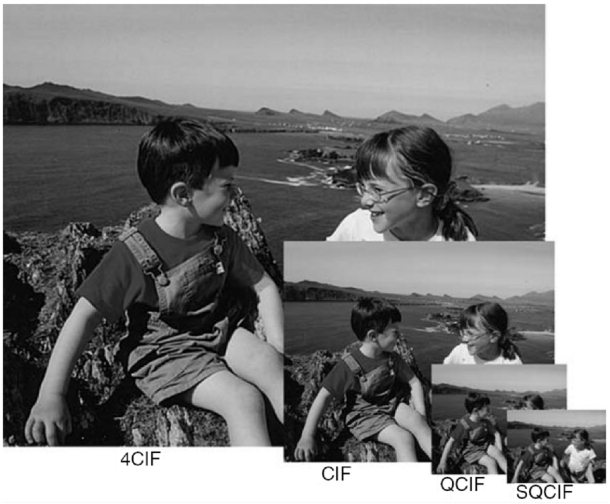


Figura 2.10: Comparacion de los tamaños de los diversos formatos CIF

- Formatos de vídeo HD: En la actualidad se está extendiendo el uso del vídeo de alta definición o HD. Éste conlleva una mayor resolución y el uso de diferentes relaciones de aspecto que el vídeo convencional CIF / BT.601. Además del sector doméstico (figura 2.11), la industria del cine y de producción audiovisual en general utiliza ampliamente estos formatos, tanto para el proceso de filmación como para la post-producción (figura 2.12). Estos formatos hacen uso, en su gran mayoría del vídeo progresivo, quedándose el vídeo entrelazado en un segundo plano.

[4]El área activa es el área de imagen realmente visible. Es igual a la total menos los intervalos de retorno horizontal y vertical que existen fuera de los límites del cuadro

[5]El DVD no es un formato de alta definición, pero se añade a esta tabla a fin de tener una referencia conocida para poder comparar los demás formatos

	Frame rate = 30 Hz (NTSC)	Frame rate = 25 Hz (PAL/SECAM)
Campos por segundo	60	50
Líneas por cuadro completo	525	625
Muestras de Luminancia por línea	858	864
Muestras de Crominancia por línea	429	432
Bits por muestra	8	8
Bit Rate total	216 Mbps	216 Mbps
Líneas activas[4] por cuadro	480	576
Muestras activas por línea (Cb, Cr)	360	360

Tabla 2.2: Características de los formatos PAL/NTSC

Formato	Resolución	Relación de Aspecto	Píxeles por cuadro
DVD[5]	720×576 (PAL) 720×480 (NTSC)	4:3 ó 16:9 (con píxeles rectangulares)	414.720 (PAL) 345.600 (NTSC)
720p	1280×720	16:9	921.600
1080i/1080p	1920×1080	16:9	2.073.600
Full Aperture Native 2K	2048×1556	1.32:1	3.145.728
Academy 2K	1828×1332	1.37:1	2.434.896
Digital Cinema 2K	2048×858 1998×1080	2.39:1 1.85:1	1.757.184 2.157.840
Academy 4K	3656×2664	1.37:1	9.739.584
Full Aperture 4K	4096×3112	4:3	12.746.752

Tabla 2.3: Características de los formatos HD

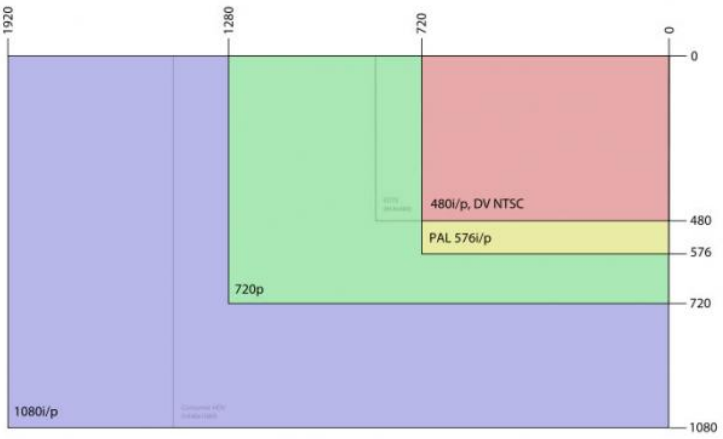


Figura 2.11: Comparación de formatos HD domésticos

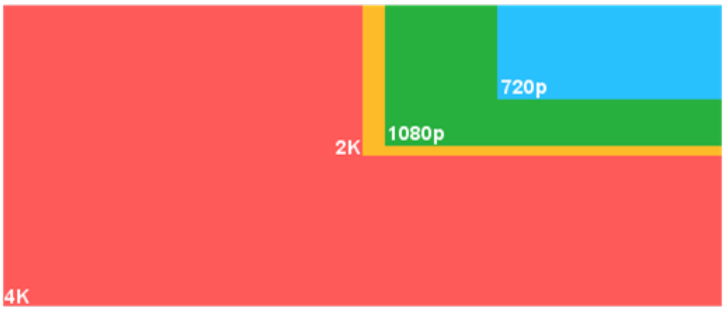


Figura 2.12: Comparación de formatos HD profesionales

2.1.4. Medida de la calidad

La percepción de la imagen es un proceso que involucra tanto al ojo como al cerebro. Se trata de un proceso complejo en el que toman parte un sinfín de factores, muchos de ellos incontrolables por el desarrollador. Principalmente hay que destacar que la calidad visual es una cuestión inherentemente subjetiva. La calidad se puede ver como la unión de fidelidad espacial (cuántos y qué detalles somos capaces de ver) y fidelidad temporal (cuánto de suaves y naturales son los movimientos de la secuencia). Sin embargo, como se ha dicho antes, hay muchos factores a tener en cuenta. El primero de todos es la valoración de calidad. Dos personas tienen valoraciones diferentes por su experiencia, por sus gustos o por el interés que despierte en él la secuencia de vídeo. Además hay otros factores como el ambiental, el cansancio del espectador, estado físico, anímico, etc. Aparte de éstos, hay tres factores muy importantes a la hora de hacer un test de calidad visual. El primero de ellos es que el espectador no ve una imagen completa, sino que se fija en puntos concretos de la misma. El segundo es la memoria a corto plazo, que implica que la opinión sobre la calidad de una escena se ve fuertemente condicionada por la calidad percibida en la(s) escena(s) anterior(es). Y finalmente, el tercer factor es la experiencia del espectador. Conforme la persona se va acostumbrando o entrenando a ver secuencias de vídeo de diferentes calidades desarrollará un sentido crítico cada vez más agudo, sabrá cómo y dónde mirar para ver fallos o defectos. Las opiniones vertidas por esta persona para la primera secuencia que visualizó y para la última no son para nada comparables. Ésta es una pega muy grande porque obliga a cambiar de observador periódicamente y cada observador tiene un estado físico y anímico distinto, factores ambientales distintos, etc.

Para resolver estos problemas se han desarrollado una serie de algoritmos y recomendaciones, como las recomendaciones ITU-R BT.500 e ITU-T P.910. Aunque lo realmente deseable sería encontrar un método de medida de calidad que fuese independiente de las personas, que lo pudiera hacer una máquina. De este modo nos evitaríamos todos los problemas mencionados anteriormente, y además ganaríamos en disponibilidad para poder hacer pruebas y medidas en cualquier momento. Desgraciadamente, éstas medidas *objetivas* tienen una validez limitada, por lo que nuevas propuestas están actualmente en estudio por el grupo VQEG (*Video Quality Experts Group*). Esperando nuevas metodologías, la medida más usada por sus buenos resultados y por la facilidad de cálculo es la PSNR.

La PSNR (*Peak Signal to Noise Ratio*) es el método más utilizado para la medida de calidad

de vídeo. Constituye una medida de las llamadas *objetivas* (calculadas mediante un sistema automático, sin intervención de personas) y puede dar unos resultados rápida y fácilmente con una fiabilidad destacable, sobre todo para comparar diferentes algoritmos, más que para una medida de calidad absoluta. Su cálculo se reduce a la siguiente expresión:

$$PSNR_{dB} = 10 \cdot \log_{10} \frac{(2^n - 1)^2}{MSE}$$

donde MSE es el error cuadrático medio entre la imagen original y la imagen a evaluar y n es el número de bits por cada muestra de la imagen.

Sin embargo, la PSNR no se correla bien con las medidas de calidad subjetiva recogidas por otros procedimientos como Rec, ITU-R BT.500, además de que un valor mayor de PSNR no tiene por qué corresponderse con una mayor calidad subjetiva. Esto se ve bien en el ejemplo de la figura 2.13:

(a) Original (b) PSNR=30.6 dB (c) PSNR=28.3 dB

(d) PSNR=27.7 dB (Fondo borroso)

Figura 2.13: Comparación de una misma imagen con diferentes valores de PSNR

La imagen 2.13(d) sólo tiene distorsionado el fondo, que se ha puesto borroso. La mayor parte de los observadores puntuarían esta imagen mejor que la imagen 2.13(c), a pesar de que ésta

tiene mayor PSNR. Esto es debido a que un espectador no se fija en la imagen completa, sino en puntos de interés, y en la imagen 2.13(d) la cara está más definida, más nítida que en la imagen 2.13(c).

2.2. Compresión de vídeo: Fundamentos y herramientas

Como ya se mencionó en la introducción, la compresión de vídeo supone un gran avance en la continua carrera por evolucionar las prestaciones y versatilidad del vídeo digital. El vídeo no comprimido tiene unas dimensiones en ancho de banda y espacio de almacenamiento requeridos que no son asumibles en prácticamente ningún escenario. Como se comentó, si codificamos según la recomendación ITU-R BT.601, el vídeo resultante sin comprimir requiere una tasa de 216 Mbps, excesivamente alta para su utilización práctica, ya sea en el sector doméstico como en el profesional. Y estamos hablando de un vídeo de resolución moderada-baja en formato 4:2:2. Si trabajamos con vídeo de alta definición los requerimientos del sistema aumentarían de forma exponencial. Además, el sector doméstico no necesita de un muestreo 4:2:2, sino que con un 4:2:0 es suficiente. Por otro lado, el sector profesional tiene otros requerimientos en muestreo (es habitual que trabajen en formatos 4:2:2 ó 4:4:4), además del vídeo progresivo, característica que va apareciendo también en las aplicaciones domésticas. Si hablamos de sistemas de transmisión el problema es más dramático, ya que generalmente el ancho de banda está muy limitado, por lo que el formato de vídeo deberá estar lo más optimizado posible (y lo más comprimido posible, dentro de unos niveles de calidad).

Todo lo expuesto anteriormente hace indispensable la aparición y uso de formatos de compresión de vídeo. Estos formatos se presentan como CODECs. Un CODEC es una pareja de aplicaciones, codificador o compresor y decodificador o descompresor (COdificador / DECodificador, en inglés *enCOder / DECoder*), y son las herramientas utilizadas para resolver el problema de los requisitos del vídeo.

Los CODECs se pueden clasificar principalmente en dos categorías: con pérdidas (*lossy*) y sin pérdidas (*lossless*). En los formatos sin pérdida la imagen reconstruida por el decodificador es exactamente igual a la original, el proceso no ha introducido ninguna distorsión ni pérdida de información sobre los datos iniciales. Estos formatos son, obviamente, los más fieles, pero las tasas de compresión son modestas, limitándose a unos ratios de 3:1 ó 4:1. Sin embargo, es en los formatos de compresión con pérdidas donde está lo interesante de la compresión de vídeo. En éstos, la imagen reconstruida no es exactamente igual que la imagen original. Esto es debido a que en estos sistemas se introduce distorsión en los datos originales porque se elimina parte de la información original. Gracias a esto, las tasas de compresión obtenidas pueden llegar a ser muy elevadas. El procesado que se realiza está orientado a la eliminación de redundancias en los datos, esto es, eliminar la información perceptualmente poco o nada relevante. Para el estudio y procesado de esta información podemos dividir el problema de las redundancias en tres partes o modelos, como se explicará en los apartados 2.2.1, 2.2.2 y 2.2.3: Modelado temporal, modelado espacial y modelado estadístico o cuantificación entrópica.

Con todo esto, un esquema simplificado de un compresor sería como el que muestra la figura 2.14.

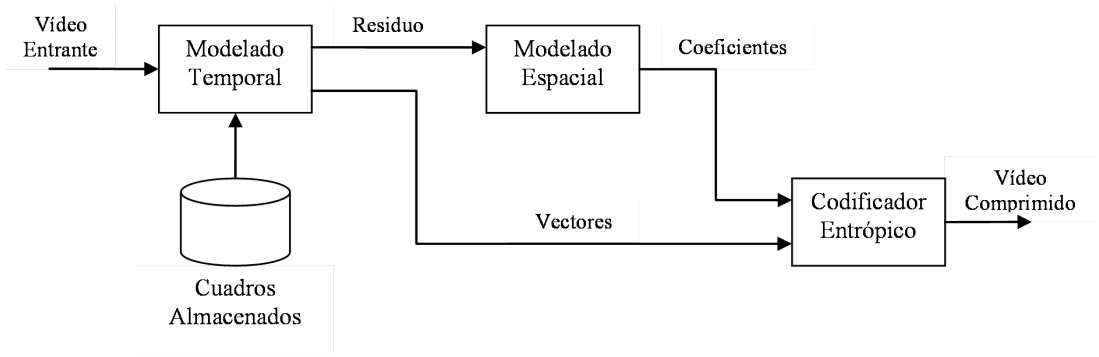


Figura 2.14: Esquema simplificado de un compresor

2.2.1. Modelado temporal

El modeloado temporal parte de la base de que la imagen va variando con el tiempo. Dos imágenes consecutivas serán, salvo casos particulares[6], muy similares, de manera que podremos *predecir* una imagen a partir de la imagen anterior.

En esta fase, el movimiento presente en la secuencia de vídeo es caracterizado y modelado, de manera que a la salida obtenemos una serie de parámetros descriptores del movimiento denominados *vectores de movimiento* y una imagen residual o *residuo*, que se corresponde con la diferencia entre la imagen real y la predicha mediante estos vectores de movimiento.

El movimiento de la imagen puede deberse a múltiples causas (no todas ellas se deben a un movimiento propiamente dicho). Entre éstas podemos citar, además de los propios movimientos de los objetos presentes en la escena, los movimientos de la cámara, los cambios de iluminación y la oclusión y aparición de elementos en la imagen. Viendo esto es fácil suponer que podemos realizar un "mapa" de los movimientos, algo así como marcar las trayectorias de cada píxel, de manera que con una imagen, que llamaremos de referencia, podríamos estimar la imagen siguiente con sólo aplicar estas trayectorias. Sin embargo, esto sería contraproducente porque el volumen de información generado sería demasiado grande. Para evitar este inconveniente los movimientos no se describen a nivel de píxel, sino a nivel de bloque, de manera que la imagen se dividirá en bloques de M $\times$ N píxeles.

El proceso completo de modelado temporal consta básicamente de dos fases:

- **Estimación del movimiento:** Búsqueda en una imagen de referencia (ya codificada, pero puede ser pasada o futura en el orden de reproducción[7]) de un bloque que se corresponda con el bloque a estudio.
- **Compensación del movimiento:** Consiste en determinar el *offset* entre el bloque estudiado y su correspondiente en la imagen de referencia, realizar una predicción con los vectores obtenidos y obtener un residuo. Si la predicción ha sido buena, el residuo resultante será una imagen con un margen dinámico pequeño. Una vez calculado el residuo se

[6] Como podrían ser los cambios de plano
[7] El orden de codificación no tiene por qué ser el mismo que el orden de reproducción de los fotogramas

le añade a la imagen predicha y se almacena como imagen de referencia. Este último paso es de gran importancia, ya que de este modo se garantiza que las imágenes de referencia que se usan en codificador y decodificador son iguales.

A efectos prácticos, los bloques utilizados son, en la mayoría de los estándares, de 16×16 píxeles. A estos bloques se les denomina *Macrobloques*. En una imagen muestreada según un patrón 4:2:0, cada macrobloque estará formado por un bloque de 16×16 píxeles, correspondiente a la luminancia, y por un bloque de 8×8 píxeles para cada componente de crominancia, como se muestra en la figura 2.15.

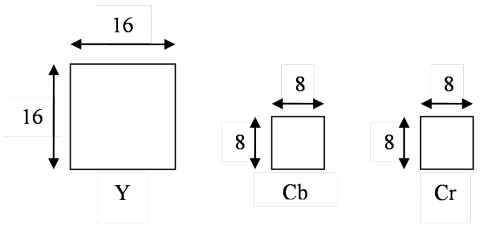


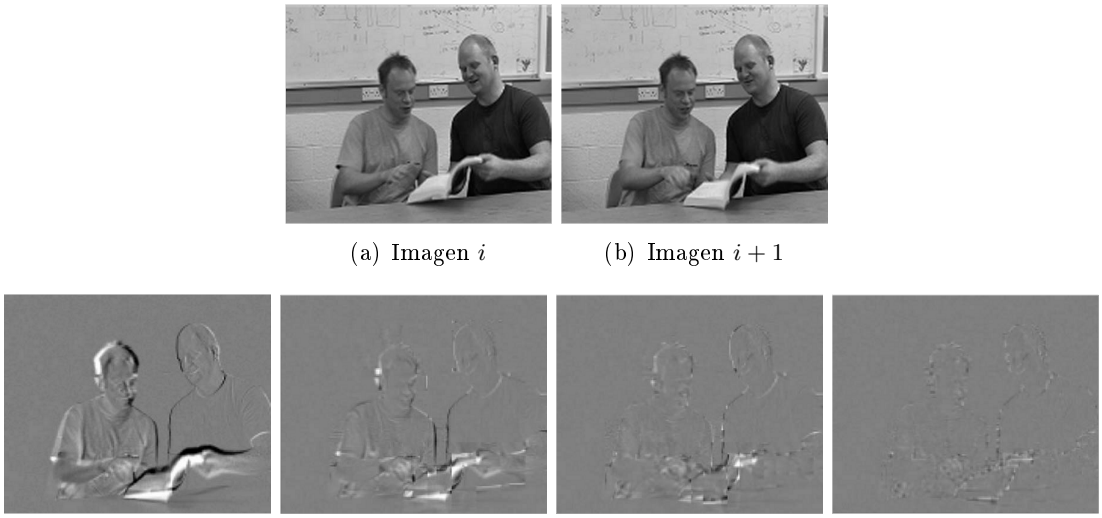
Figura 2.15: Tamaño de los macrobloques de luminancia y de crominancia

Una vez determinados los Macrobloques (o MBs), existen tres tipos de predicciones que se pueden realizar para estos macrobloques: Inter, Intra y Bidireccional:

- **Inter:** El MB se codifica usando compensación de movimiento
- **Intra:** El MB se codifica sin compensación de movimiento.
- **Bidireccional:** Se codifica como en el caso Inter, solo que esta vez se utilizan dos imágenes de referencia, una pasada y otra futura (en el orden de exhibición).

Hemos visto que los MBs tienen un tamaño de 16×16 píxeles, pero este tamaño no siempre es el más adecuado para la codificación. Es por esto por lo que utilizamos Bloques, que no son sino subdivisiones de los Macrobloques. El uso de un tamaño bloque u otro condiciona la energía del residuo, esto es, la cantidad de bits que deberemos emplear en codificarlo, como se puede comprobar en la figura 2.16. Usar bloques pequeños en la compensación de movimiento hace que la predicción pueda ser más precisa y que el residuo tenga poca energía ocupando, por lo tanto, pocos bits. Sin embargo, muchos bloques conllevan muchos vectores de movimiento, lo que implica gastar muchos bits en codificarlos. En el caso contrario, si utilizamos tamaños de bloque grandes, habrá menos vectores de movimiento, pero la predicción será poco precisa, con lo que el residuo tendrá una energía mayor y serán necesarios más bits para codificarlo. Como se puede ver, se trata de una situación de compromiso, en la que hay que compensar el ahorro de bits en una faceta con el gasto extra en la otra. Todo esto sin contar que si hay más bloques la carga computacional se incrementa, ya que la compensación de movimiento es un proceso costoso en este aspecto.

Por último hay que destacar otra técnica ampliamente utilizada, como es la compensación sub-píxel. Ésta se basa en que los movimientos del mundo real son continuos y a una velocidad concreta no especificada, por lo que entre un cuadro y el siguiente no se corresponden caprichosamente a un movimiento de un número entero de píxeles. Dicho de otro modo, un objeto se puede

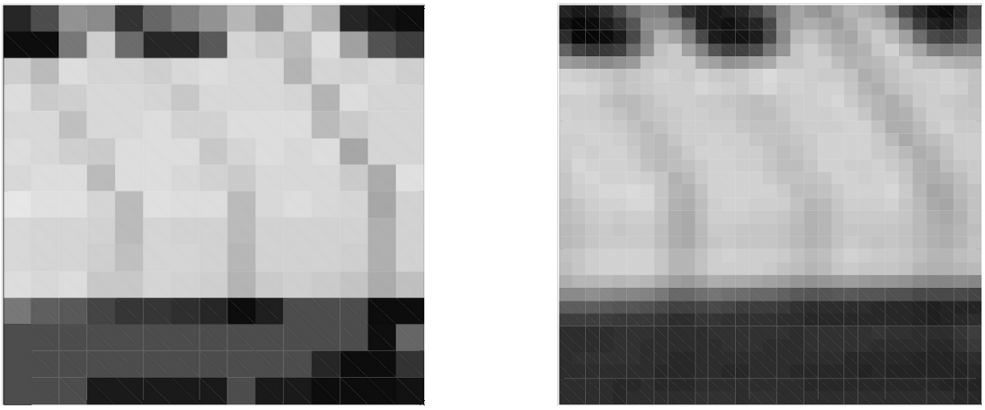


(a) Imagen i (b) Imagen $i+1$

(c) Sin compensación (d) Bloques 16×16 (e) Bloques 8×8 (f) Bloques 4×4

Figura 2.16: Resíduos generados para cada tamaño de bloque

mover sólo medio píxel. Es por esto que se utilizan técnicas de interpolación sobre la imagen de referencia (figura 2.17) a fin de conseguir un vector de movimiento más preciso (figura 2.18). Ésta técnica, de un notable coste computacional, reduce la energía del residuo, pero no hay que olvidar que el hecho de que los vectores sean más precisos obliga a que éstos tengan más bits. En este caso seguimos en una situación de compromiso.



(a) MB Original (b) MB Interpolado 4x

Figura 2.17: Interpolación de los macrobloques en compensación sub-píxel

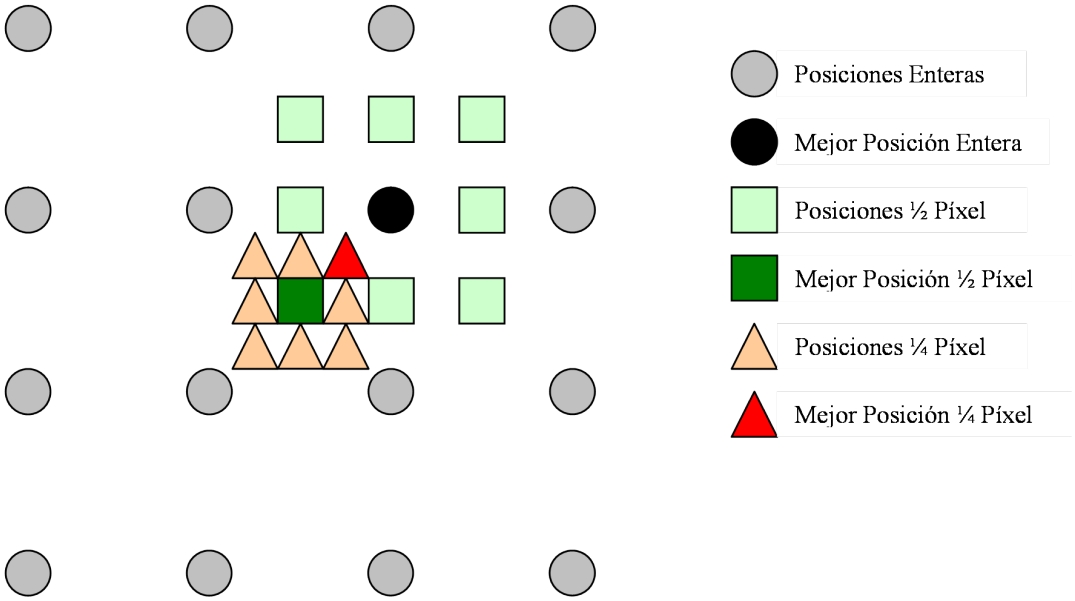


Figura 2.18: Posiciones posibles del vector de movimiento según el grado de interpolación

2.2.2. Modelado espacial

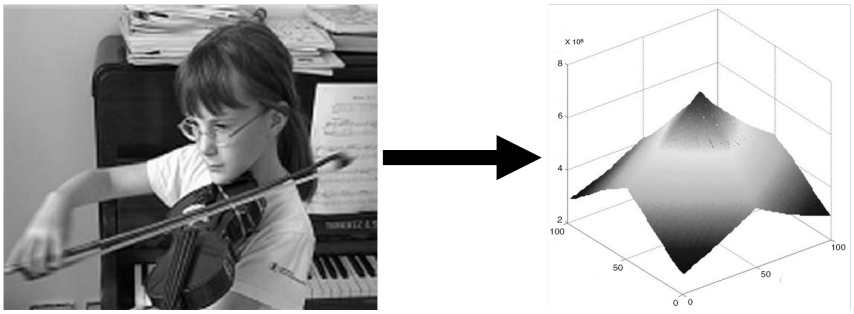
Este modelado se basa en la asunción de que dos píxeles vecinos (o próximos) serán muy similares.

Como hemos visto anteriormente, el resultado del proceso de estimación + compensación de movimiento es un conjunto de vectores de movimiento y un residuo. En el modelado espacial se trabajará con dicho residuo con el fin de reducir su tamaño. Sin embargo, hay que destacar que los modelos temporal y espacial, aunque operan en conjunto son independientes entre sí.

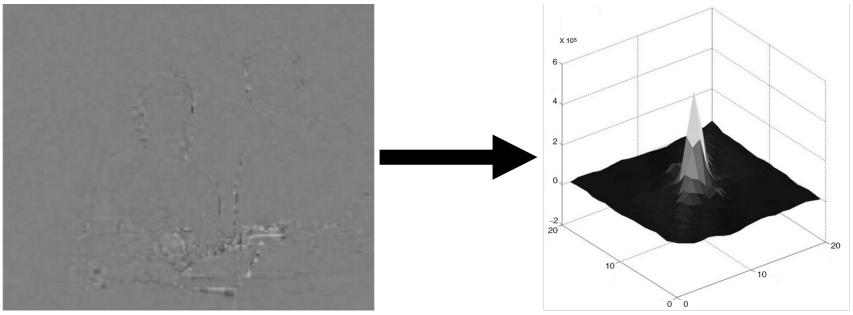
Para hacernos una idea sobre la compresibilidad de una imagen se recurre a la autocorrelación de la misma. Cuanto más rápido decrezca, cuanto más se parezca a una *delta*, δ, esto es, cuanto menos correlada esté, más se podrá comprimir. En el caso de los residuos producidos por el modelado temporal, éstos ya están muy incorrelados, como se puede comprobar en la figura 2.19.

Al igual que en el modelado temporal, en este caso también se puede realizar una predicción de las muestras de una imagen a partir de muestras vecinas previamente codificadas. A este esquema se le llama DPCM (*Differential Pulse Code Modulation*).

Sin embargo, la operación más significativa en este modelado espacial es la codificación transformada. En ella la imagen o residuo son transformados a otro dominio, de manera que sus muestras estén decorreladas (o más decorreladas en el caso del residuo), a fin de poder conseguir una mayor tasa de compresión. Existen múltiples transformaciones posibles, que podemos clasificar en dos grupos:



(a)



(b)

Figura 2.19: Autocorrelaciones de (a) una imagen y (b) un residuo

- **Basadas en bloque:** Operan en bloques o pequeñas fracciones de la imagen. Bajos requerimientos de memoria y muy adecuadas para su uso conjunto con técnicas de estimación de movimiento por bloques. Como contrapartida, son propensas a la aparición de artefactos en los bordes de la imagen. Ejemplos de este tipo son la Transformada Discreta del Coseno (DCT), la Transformada Karhunen-Loeve (KLT) o la Descomposición en Valores Singulares (SVD).

- **Basadas en imagen:** Operan en la imagen completa y mejoran los resultados obtenidos por las transformaciones basadas en bloque. En su contra, tienen grandes requerimientos de memoria. El ejemplo más común de este tipo es la Transformada Wavelet Discreta (también llamada "DWT" ó simplemente "Wavelet").

A continuación expondremos los conceptos generales de las transformadas DCT y DWT.

2.2.2.1. Transformada DCT

La Transformada Discreta del Coseno o DCT actúa sobre bloques de N×N muestras. Es una transformación reversible, por lo que existe la transformada inversa, o IDCT. Las expresiones matemáticas de estas transformaciones son las siguientes:

$$\begin{aligned} \text{DCT} &: Y = AXA^T \\ \text{IDCT} &: X = A^TYA \end{aligned}$$

donde

A es la matriz N×N de la transformación:

$$A_{ij} = C_i \cdot \cos\frac{(2j+1)i\pi}{2N}, \text{ donde } C_i = \begin{cases} \sqrt{\frac{1}{N}}, & i = 0 \\ \sqrt{\frac{2}{N}}, & i > 0 \end{cases}$$

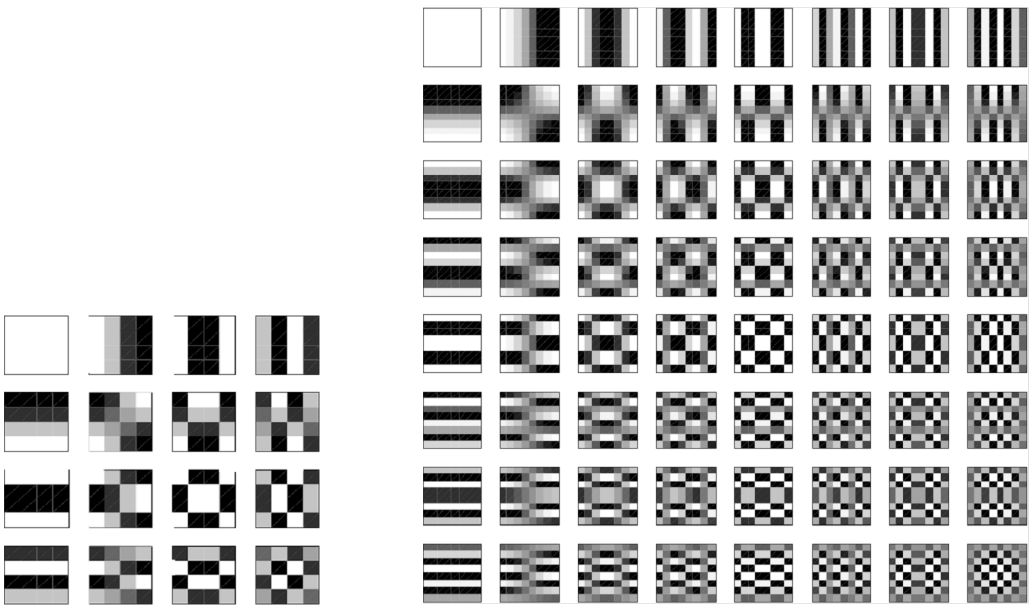
Y es la matriz N×N de coeficientes resultante de la transformación:

$$Y_{xy} = C_x C_y \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} X_{ij} \cdot \cos\frac{(2j+1)y\pi}{2N} \cos\frac{(2i+1)x\pi}{2N}$$

X es una matriz con los valores de las N×N muestras del bloque:

$$X_{ij} = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} C_x C_y Y_{xy} \cdot \cos\frac{(2j+1)y\pi}{2N} \cos\frac{(2i+1)x\pi}{2N}$$

La utilidad de esta transformada es que obtenemos una serie de coeficientes (una matriz N×N) que son los pesos de una base de N×N funciones. Si reconstruimos la imagen con los N^2 coeficientes, obtendríamos la imagen original. Sin embargo, de todos los coeficientes obtenidos, sólo unos pocos tienen valores significativos, esto es, la mayoría de los coeficientes tienen muy poca energía. Si aproximamos a cero estos últimos habremos reducido en gran medida el volumen de información, mientras que la distorsión introducida habrá sido pequeña, ya que esos coeficientes eran próximos a cero. La base de funciones a la que nos estamos refiriendo se corresponde a la de la figura 2.20.



(a) Base 4×4 (b) Base 8×8

Figura 2.20: Bases de funciones DCT

2.2.2.2. Transformada wavelet

La Transformada Wavelet es una transformación que actúa sobre toda la imagen de una vez. Consiste en una serie de filtros paso alto y paso bajo que se aplican sobre las filas y columnas de la imagen a tratar. Posteriormente la imagen filtrada es diezmada en un factor 2. Este proceso se realiza iterativamente, de manera que cuantas más iteraciones, más sub bandas son creadas (ver ejemplo en la figura 2.21). Todas ellas se organizan en un árbol, cuya profundidad se corresponde con el número de iteraciones. La utilidad de esta transformación radica en que las imágenes filtradas (o sub-bandas) son más cercanas al negro cuanto mayor sea su profundidad en el árbol. Esto hace que estas sub-bandas se puedan comprimir. Para la transformación inversa se realiza el proceso contrario: interpolación, filtrado y suma de las imágenes obtenidas.

Este tipo de transformación resulta muy adecuada para sistemas de codificación de vídeo escalar o por capas, en los que dependiendo del caso (ancho de banda, sobreprecio que paga el cliente, etc.) se puede obtener intencionadamente una secuencia de vídeo con más o menos calidad.

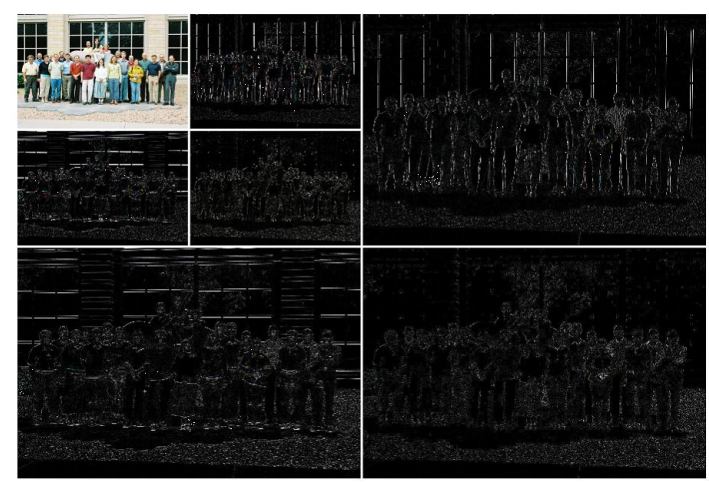


Figura 2.21: Descomposición wavelet de dos etapas

2.2.2.3. Cuantificación

Cuantificación Escalar

La cuantificación es el proceso con pérdidas y no reversible que hace que la información se reduzca en mayor o menor medida. Existen varios tipos de cuantificadores, aunque nos centraremos en el más sencillo, el cuantificador uniforme. En este caso, la función de transferencia es la siguiente:

$$FQ = \text{round}\left(\frac{X}{QP}\right)$$

$$Y = FQ \cdot QP$$

donde

X es el valor de entrada al cuantificador
QP es el escalón de cuantificación
FQ es el valor cuantificado
Y es el valor reconstruido

Esta función de transferencia se corresponde con la de la figura 2.22

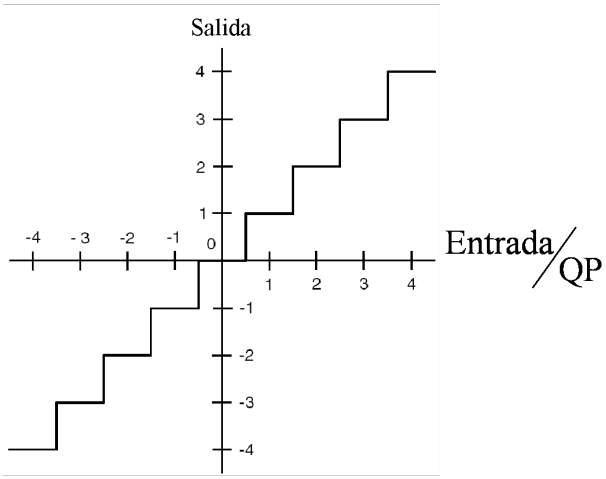


Figura 2.22: Cuantificador uniforme

El parámetro más importante del cuantificador es el escalón de cuantificación (distancia entre dos valores cuantificados consecutivos), ya que es el que determina la cantidad de elementos de entrada que van a *redondearse* a cero, provocando así una mayor o menor compresión. Debido a esto, es el parámetro que determina las pérdidas de información, y por lo tanto, la distorsión introducida en la imagen.

Cuantificación Vectorial

La cuantificación vectorial asocia un conjunto de datos de entrada a un vector, que es la mejor aproximación de los datos de entrada de entre las presentes en una base de datos en el codificador. Este vector tiene un código dentro de la base de datos, que es lo que se transmitirá. En el otro extremo, el receptor tiene la misma base de datos almacenada, de manera que con sólo recibir el código será capaz de ofrecer a su salida un conjunto de datos muy similar al original. Su funcionamiento esquemático es el indicado en la figura 2.23.

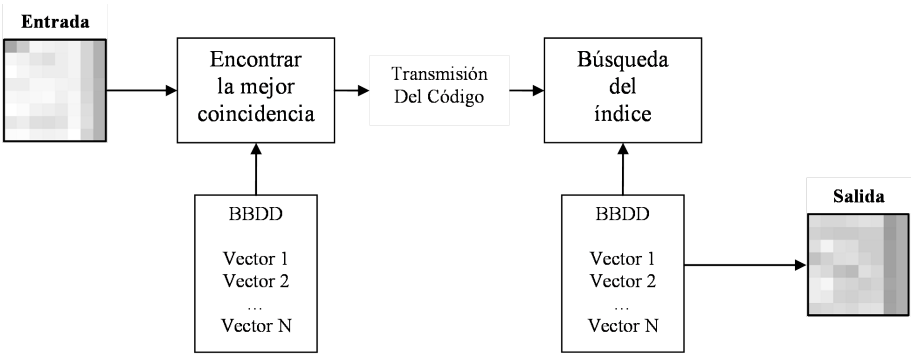


Figura 2.23: Esquema de un cuantificador vectorial

2.2.2.4. Reordenamiento y codificación de ceros

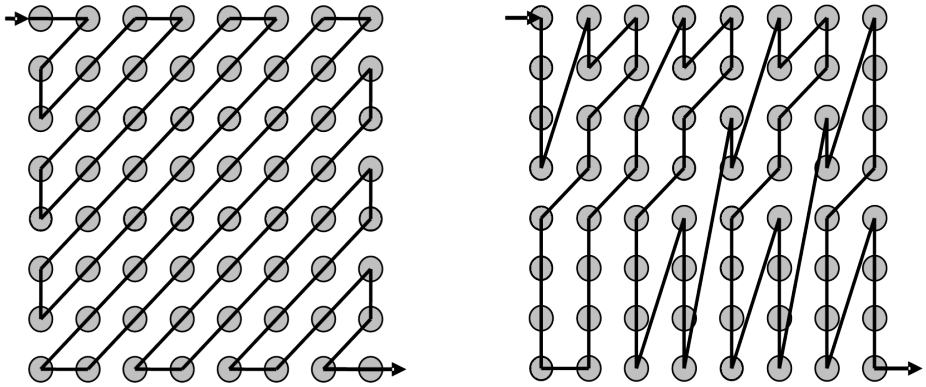
La salida del cuantificador es un conjunto de valores, muchos de los cuales son cero. Es necesario encontrar una forma eficiente de ordenar y codificar estos valores. Si podemos juntar la mayor cantidad de ceros podremos representarlos de manera más eficiente y compacta en el flujo de datos.

Para ello, lo primero será investigar cómo están distribuidos los coeficientes. Para ello deberemos distinguir entre las diferentes transformaciones realizadas. En este caso diferenciaremos DCT y Wavelet.

DCT

Podemos observar que los mayores valores (coeficientes con mayor energía) están ubicados en los coeficientes correspondientes a las bajas frecuencias, especialmente en el coeficiente DC (0,0). Es por esto que primero reordenaremos los coeficientes para que primero estén juntos todos los correspondientes a las bajas frecuencias, y después los de altas frecuencias, la mayoría de ellos, cero. El resultado de este reordenamiento es un escaneo en *zig-zag* del macrobloque (ver figura 2.24). Este escaneo variará dependiendo de si el vídeo es progresivo o entrelazado, ya que en el caso entrelazado la distribución de los coeficientes de un campo varía en su componente vertical.

Una vez reordenados los datos, tenemos secuencias largas de ceros, que deberemos compactar. Para ello se codificará la cadena de valores mediante pares (*run*, *level*), de manera que *run* indica el número de ceros que preceden a un coeficiente no nulo y *level* indica el valor de ese coeficiente no nulo. Para el caso del último coeficiente no nulo, se usará un código especial llamado *last*, que indicará que la información del macrobloque ha terminado. Otro modo es usar ternas (*run*, *level*, *last*) en vez de pares, de manera que si $last = 1$, se trata del último coeficiente.



(a) Vídeo Progresivo (b) Vídeo Entrelazado

Figura 2.24: Modos de escaneo en zig-zag de un macrobloque

Wavelet

En el caso de la Transformada Wavelet, cuanto mayor sea la sub-banda (más cercana a la esquina inferior derecha, mayor número de coeficientes serán cero. Los coeficientes distintos de cero se suelen corresponder con objetos o estructuras en la imagen. Además, si en una sub-banda hay coeficientes no nulos, los correspondientes en sub-bandas superiores muy probablemente tampoco serán nulos. Del mismo modo, cuando un coeficiente es cero, muy probablemente los coeficientes correspondientes a sub-bandas superiores serán cero. Podemos relacionarlos en una especie de árbol de coeficientes no nulos, denominado *zerotree*, mostrado en la figura 2.25.

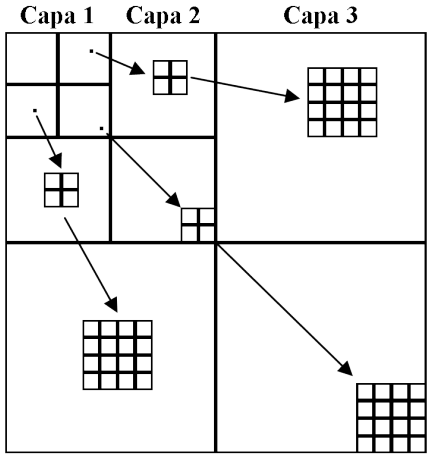


Figura 2.25: Esquema de codificación *zerotree*

2.2.3. Codificación entrópica

Tras realizar los procesos descritos anteriormente, el resultado obtenido consiste en un conjunto de datos que presentan una serie de patrones estadísticos que podremos procesar para comprimirlo aún más (por ejemplo, los vectores de movimiento que se repiten se codifican con menos bits o sucesiones largas de ceros o de unos, que se pueden compactar fácilmente). Para ello se usará un codificador de entropía o entrópico.

Pero antes de comprimir la información es vital ser capaz de generarla de la manera más eficiente posible. Para ello, en el momento de la generación se utiliza una codificación predictiva. Ésta consiste en explotar las correlaciones locales presentes en el vídeo. En el caso de los vectores de movimiento, es fácil suponer que el movimiento presente en dos bloques vecinos se parecerá mucho, por lo que se realizará una predicción del movimiento en base a vectores previamente calculados. Posteriormente se calculará el vector de movimiento real y se calculará la diferencia, MVD (*Motion Vector Difference*), que será lo que se transmita. Del mismo modo, el parámetro de cuantificación o QP variará poco en un cuadro, y menos en una vecindad, por lo que es preferible señalizar la diferencia de QPs, si la hubiera, que utilizar el valor completo de QP, ya que ocuparía más bits.

Una vez obtenidos todos los datos de la manera más eficiente posible se acometerá la codificación entrópica. Para realzarla tenemos, básicamente, dos tipos de códigos: codificación de longitud variable y codificación aritmética.

2.2.3.1. Codificación de longitud variable

Los códigos de longitud variable, o VLCs, asignan palabras clave a símbolos de entrada, de manera que los símbolos más frecuentes se asocian con las palabras clave (o *codewords*) más cortas, mientras que los símbolos de entrada menos frecuentes se asocian a las palabras clave más largas. La codificación más habitual de esta familia es la codificación Huffman.

En la codificación Huffman se asigna un VLC a cada símbolo en base a su probabilidad de ocurrencia. Si las probabilidades son estimadas de manera precisa la codificación Huffman es capaz de compactar significativamente el conjunto de datos de entrada. Como punto en contra, este algoritmo obliga a construir una tabla de ocurrencias de cada símbolo en la secuencia, que sólo se podrá realizar cuando toda la secuencia se haya codificado. Esto conlleva introducir un retardo generalmente inaceptable (sobre todo cuando la secuencia es más o menos larga). Además, al ser estas tablas generadas *ad-hoc* para cada vídeo, es necesaria su transmisión al extremo receptor para que éste sea capaz de decodificarlas, lo que implica introducir un overhead que puede resultar excesivo.

Como solución a todos estos problemas, los sistemas de compresión de vídeo suelen adoptar un conjunto de símbolos precalculados a partir de vídeos genéricos, que aunque no sean tan precisos como la construcción *ad-hoc* de las tablas sí funcionan bastante bien.

Además de lo hablado, los VLCs sufren de otro inconveniente, que es su fragilidad a errores

de transmisión o pérdidas. Si un error de este tipo ocurre la información se pierde, pero además los errores se propagan, pudiendo provocar grandes errores en la decodificación de la secuencia. Para tratar de mejorar este inconveniente se han desarrollado los VLCs reversibles (RVLCs), que son similares a los VLCs tradicionales, con la ventaja añadida de que son decodificables en ambas direcciones (hacia delante o hacia atrás), mejorando su comportamiento ante situaciones de pérdidas o errores en la transmisión.

2.2.3.2. Codificación aritmética

Como se ha visto anteriormente, los códigos VLC asignan un número entero de bits a cada símbolo. Esta estrategia es sub-óptima, ya que según su ocurrencia, cada símbolo necesitará más o menos bits, siendo esta cantidad en realidad un número fraccionario. Esto se puede ver claramente en el caso de un símbolo con probabilidad mayor que 0.5: sólo podríamos darle 1 bit (en realidad habría que darle menos de 1 bit). En este caso la compresión alcanzada sería muy poco eficiente. Para solucionar este problema surge la codificación aritmética. Ésta convierte una secuencia de símbolos en un número fraccionario, de manera que se puede acercar al número óptimo (fraccionario) de bits por símbolo para optimizar la eficiencia de la compresión. Este número fraccionario no está limitado a un número entero de bits para cada símbolo transmitido.

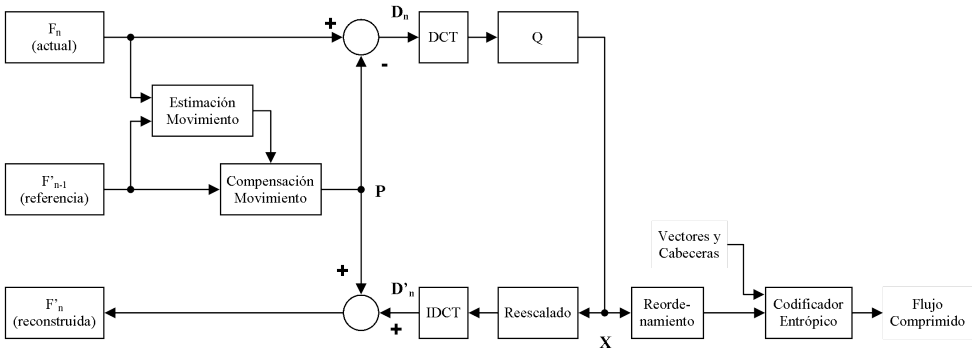
Existe un esquema de codificación aritmética denominado CAE (*Context-based Arithmetic Encoding*), que utiliza las características espacio-temporales locales para realizar una estimación de las probabilidades de los símbolos. Este esquema es utilizado en el Main Profile de H.264 bajo el nombre CABAC (*Context-based Adaptive Binary Arithmetic Coding*).

2.2.4. Esquema final del CODEC

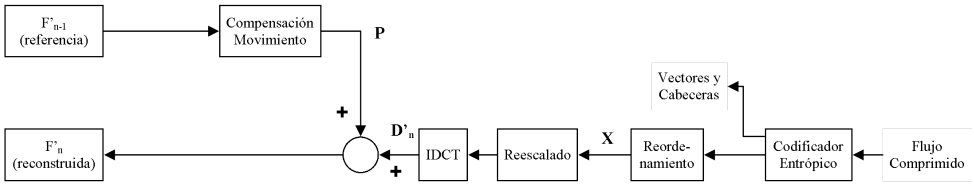
Una vez vistas las herramientas utilizadas en un esquema general de compresión de vídeo, expondremos una visión general de lo que hemos llamado CODEC. El modelo de CODEC más utilizado es el denominado DPCM/DCT (DPCM por implementar un mecanismo de estimación y compensación de movimiento, DCT por utilizar la transformada discreta del coseno). El objetivo del CODEC es, a partir del cuadro n-ésimo, F_n, y de unas imágenes de referencia, F_{n-1}, generar un flujo de datos comprimidos, que al llegar al decodificador son interpretados para dar lugar a un cuadro reconstruido $F_n{}'$ que será diferente al cuadro original debido a las pérdidas introducidas por la compresión en el paso de cuantificación.

El esquema general de un CODEC DPCM / DCT está descrito en la figura 2.26.

En este esquema se puede ver que el codificador implementa un lazo de decodificación completo. El motivo de esto es que los cuadros de referencia de que dispone no sean cuadros originales, sino decodificados, con el fin de que el codificador y el decodificador dispongan exactamente de las mismas referencias.



(a) Codificador



(b) Decodificador

Figura 2.26: Esquema genérico de un CODEC DPCM/DCT

2.3. El estándar H.264/MPEG-4 AVC

La norma H.264/AVC [1] es un estándar de codificación de vídeo de alta compresibilidad, cuyo objetivo inicial era obtener una mayor eficiencia de codificación sin un gran incremento computacional. Se debía obtener una buena calidad de imagen al mismo tiempo que una notable reducción de la tasa binaria utilizada en comparación con los estándares existentes (principalmente MPEG 2, H.263 y MPEG 4 Parte 2).

El desarrollo de este sistema comenzó en 1998, cuando se convocó el concurso de ideas[8] para el desarrollo de H.26L, a la postre el embrión de H.264, por parte del grupo VCEG[9].
En 1999 concluye el primer *test model* de H.26L, TM1.
Por otro lado, en 2000 el grupo MPEG[10] inicia el concurso de ideas para la creación de herramientas avanzadas de compresión de vídeo para su estándar MPEG-4.
En 2001, el grupo MPEG adopta H.26L como base para la creación de la Parte 10 de MPEG-4. Ya que las líneas de trabajo pasan a ser comunes, los grupos VCEG y MPEG forman una *joint venture* denominada JVT, *Joint Video Team*, que se encargará del desarrollo del futuro formato de vídeo.
En Marzo de 2003 las especificaciones finales del nuevo formato fueron publicadas. Este formato se denominó con el nombre híbrido H.264/MPEG-4 AVC (ITU-T H.264 y MPEG-4 Part 10 o MPEG-4 Advanced Video Coding (AVC)) debido a los dos organismos que participaron en su elaboración. En algunos casos se puede ver bajo el nombre JVT, por ser desarrollado por el *Joint Video Team*.
En 2004 se publicaron las *Fidelity Range Extensions*, FREXt, que son una serie de extensiones y añadidos al estándar original que responden a una serie de demandas, principalmente por parte del sector profesional, para hacerlo más versátil y útil.
Desde 2005 se viene trabajando en el anexo SVC, *Scalable Video Coding*, cuya motivación es hacer un CODEC H.264 escalable. No hablaremos de este campo, ya que se sale del objetivo de este Proyecto Fin de Carrera.

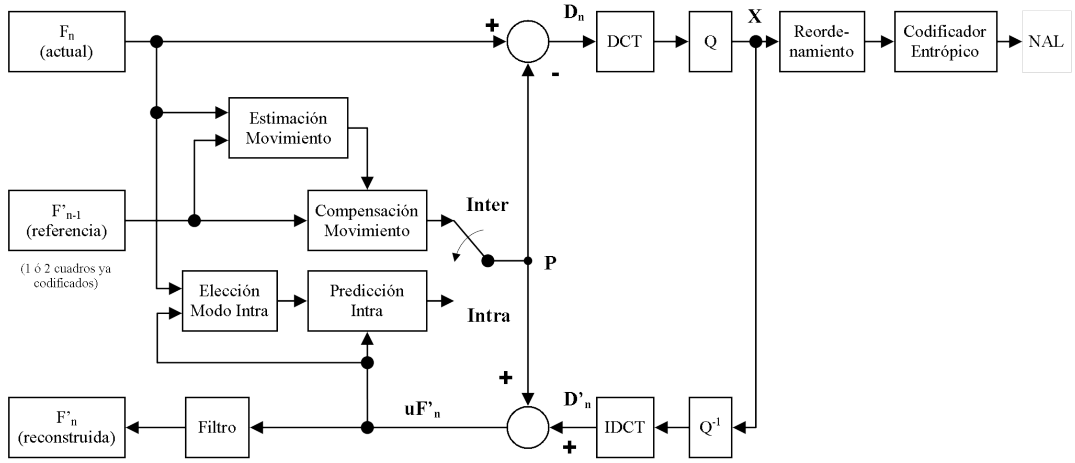
2.3.1. Características del CODEC

Los diferentes estándares de codificación de vídeo no especifican un codificador, sino una sintaxis (qué códigos se usan) y una semántica (qué significa cada cosa), esto es, indican cómo se codifica. Además, se indica el procedimiento para leer e interpretar el flujo binario para finalmente conseguir una serie de imágenes. Con todo esto, el estándar lo que especifica es cómo es un flujo binario conforme al estándar. Un codificador que cumpla con el estándar debe generar un flujo como el especificado. Finalmente, para poder comprobar su validez y a fin de mejorar la interoperabilidad se especifica un decodificador hipotético de referencia o HRD[11] (*Hypothetical Referente Decoder*). Un flujo que no puede ser interpretado por el HRD no es conforme al estándar. El motivo por el que el proceso de codificación (que no el algoritmo) no está especificado

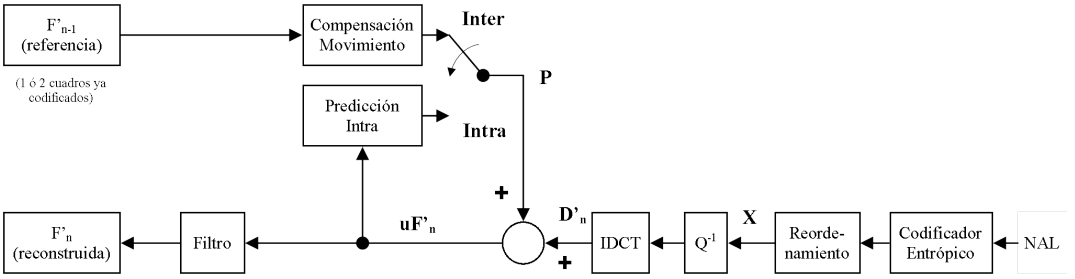
[8] En la literatura se le denomina *Call for Proposals*
[9] *Video Coding Experts Group*, grupo de trabajo de la ITU-T
[10] *Moving Picture Experts Group*, grupo de trabajo de ISO/IEC (nombre oficial: ISO/IEC JTC1/SC29/WG11)
[11] El HRD está conceptualmente conectado a la salida del codificador y consiste en un buffer de entrada, llamado CPB (*Coded Picture Buffer*), un decodificador y un dispositivo de visualización

es dar mayor flexibilidad para su implementación, al mismo tiempo que estimular la evolución del estándar en temas como calidad visual conseguida o eficiencia en la codificación.

En cuanto al CODEC en sí, sigue la estructura DPCM/DCT comentada anteriormente (con estimación y compensación de movimiento y codificación mediante transformada DCT), con alguna modificación propia. El esquema general que sigue se puede ver en la figura 2.27.



(a) Codificador H.264



(b) Decodificador

Figura 2.27: Esquema general del CODEC H.264

En el bloque codificador podemos observar, dos ramas claramente diferenciadas: una rama de codificación (parte superior) y otra rama de decodificación (parte inferior).

La presencia de la rama de decodificación responde a la necesidad expuesta en la explicación del modelo DPCM/DCT, y es que ambos, codificador y decodificador, trabajen exactamente con las mismas imágenes de referencia. Como novedad podemos ver la incorporación de un nuevo bloque "*Filtro*", que se encargará de reducir los artefactos introducidos en la imagen, fruto de la codificación por bloques. En la rama de codificación podemos observar un esquema similar al del modelo general DPCM/DCT, con la aparición de un nuevo método de predicción denominado *Intra* (sin referencia a otros cuadros, sino sólo a sí mismo), de manera que en cada macrobloque se decidirá entre la predicción Intra e Inter, y dentro de cada una se elegirá un modo concreto de predicción. Asimismo podemos observar que para las imágenes de referencia podemos utilizar

más de una imagen previamente codificada.

El esquema del decodificador es funcionalmente igual que la rama de realimentación del codificador, ya que su función es similar (decodificar la secuencia de vídeo), aunque con distinto fin (en el codificador es la reproducción de la secuencia, mientras que en el codificador es la generación de imágenes de referencia).

Como dato a tener en cuenta, observamos que la salida del codificador (y la entrada del decodificador) se corresponden a la capa NAL[12], que es una entidad de abstracción que dota a H.264 de una gran versatilidad a la hora de su utilización en diferentes entornos de red o de almacenamiento físico.

2.3.2. Formato de vídeo. Tiras y macrobloques

En sus orígenes, H.264 admitía la codificación de vídeo muestreado con un patrón 4:2:0, tanto en modo progresivo como entrelazado, con una precisión en cada muestra de 8 bits. Actualmente el estándar también soporta la codificación de vídeo con muestreos 4:2:2 y 4:4:4 y precisiones de hasta 14 bits gracias a la introducción de las *Fidelity Range Extensions*, FREext, para su uso en aplicaciones profesionales y de edición.

Como se ha comentado, H.264 utiliza un sistema de codificación de bloques denominados macrobloques (MBs). Estos macrobloques están formados por conjuntos de 16×16 muestras de Luminancia, o *Luma*, y sus muestras de color asociadas, de forma que podemos tener bloques de Crominancia o *Croma* de 8×8, 16×8 ó 16×16 muestras, dependiendo del tipo de muestreo empleado, 4:2:0, 4:2:2 ó 4:4:4, respectivamente.

Existen tres tipos de macrobloques:

- Macrobloques "I": Se predicen utilizando predicción Intra a partir de muestras previamente decodificadas del cuadro actual (figura 2.28). La predicción se realiza para el MB completo (16×16 muestras), existiendo 4 modos (direcciones) posibles, o para cada sub bloque de 4×4 muestras del MB, con 9 modos especificados. Adicionalmente, las extensiones FREext [5] permiten bloques de 8×8 muestras, pudiendo emplear las mismas direcciones empleadas por los bloques 4×4.

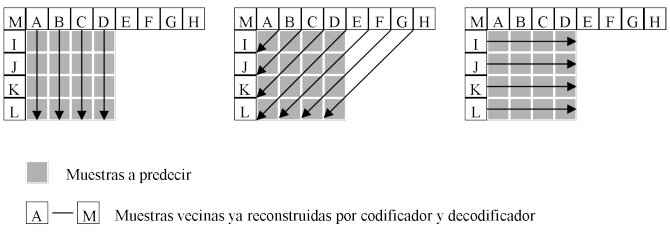


Figura 2.28: Mecanismo de predicción Intra

[12] *Network Abstraction Layer*, Capa de Abstracción de Red

- Macrobloques "P": Se predicen mediante predicción Inter a partir de una o más imágenes de referencia (ubicadas en una o dos listas, «lista 0» y «lista 1»). En este caso cada MB puede ser dividido en «particiones de macrobloque», esto es, sub-bloques de tamaño 16×16, 16×8, 8×16 u 8×8 muestras de luminancia. Adicionalmente, si se utilizan particiones 8×8, éstas pueden ser adicionalmente divididas en sub-particiones de macrobloque, de tamaño 8×8, 8×4, 4×8 ó 4×4 muestras de luminancia (y sus correspondientes muestras de crominancia). Estas divisiones se muestran en la figura 2.29. Cada partición de MB se predice a partir de una imagen de la lista 0, mientras que cada sub partición de MB (en el caso de que se utilice) se predice a partir de la misma imagen de la lista 0.

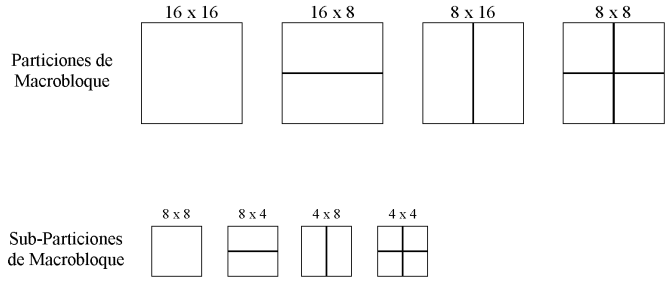


Figura 2.29: Particiones y sub-particiones de macrobloque

- Macrobloques "B": Se predicen mediante predicción Inter empleando mecanismos similares a los de los Macrobloques P a partir de una o dos imágenes de referencia, una de la lista 0 y/u otra de la lista 1 (ver figura 2.30). Si existieran, todas las sub particiones de macrobloque se predicen a partir de la(s) misma(s) imágen(es) de referencia, una de la lista 0 y/u otra de la lista 1.

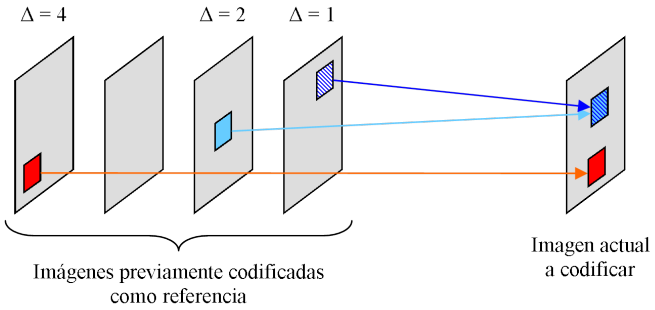


Figura 2.30: Predicción de macrobloques "B"

En un escalón superior nos encontramos con las Tiras o *Slices* (figura 2.31). Una tira es un conjunto de macrobloques escaneados mediante un sistema de barrido, o *raster scan* aunque no tienen por qué ser contiguos). Las tiras pueden ser de tres tipos, dependiendo del formato de macrobloques que contengan. Así, tenemos tiras I, que sólo pueden contener macrobloques "I", tiras P que pueden contener macrobloques "I" y "P" y tiras B, que pueden albergar macrobloques de cualquier tipo ("I", "P" ó "B"). Adicionalmente existen otros dos tipos de tira, denominados "SI" y "SP", que sirven para una conmutación eficiente entre diferentes flujos de vídeo.

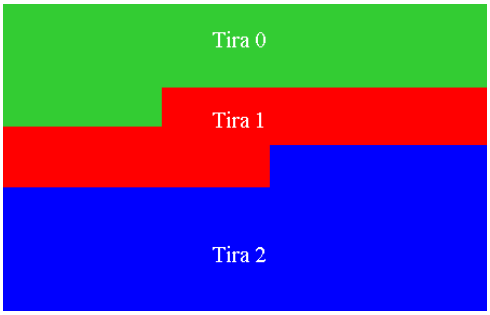


Figura 2.31: Cuadro compuesto por tres tiras

Por encima del nivel de tira está el nivel de imagen. En este nivel tenemos dos opciones, en función de la naturaleza del vídeo: si el vídeo es progresivo tendremos cuadros (o *frames*), mientras que si el vídeo es entrelazado tendremos campos (o *fields*). La forma de codificar la imagen dependerá en gran medida del tipo de imagen que sea.

Finalmente, el vídeo se presenta en forma de una secuencia de imágenes. Esta secuencia está organizada en GOPs (*Group Of Pictures*), que son grupos de imágenes de diferentes tipos ("I", "P", "B") de longitud fija[13] y estructura regular. Estos grupos van encabezados por una imagen de tipo I, ya que no se presupone ninguna referencia anterior. Posteriormente a la imagen I se suceden imágenes P y B (si son empleadas) siguiendo un patrón regular, de manera que la imagen I sirve como referencia para la predicción de las imágenes P y B, y la imagen P sirve para predecir otras imágenes P y B, mientras que las imágenes B no sirven para predecir ninguna otra imagen[14]. Podemos ver un ejemplo de GOP en la figura 2.32.

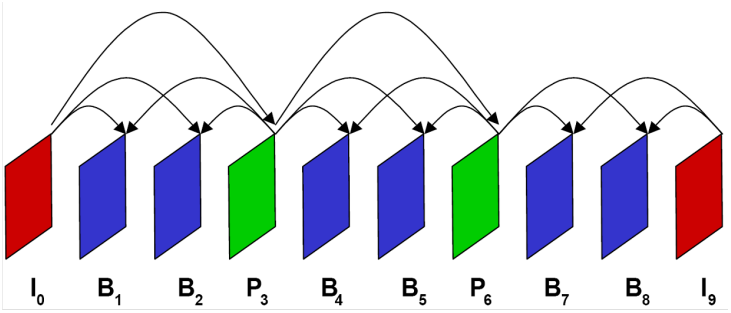


Figura 2.32: GOP de longitud 9 con patrón IP2B

El uso de estas estructuras viene motivado por tener un mayor control de la reproducción, así como para evitar la propagación indefinida de errores que pudieran ocurrir en el proceso de codificación/transmisión o almacenamiento/decodificación.

[13] Pueden no tener longitud fija si se consideran algoritmos de cambio de escena que modifiquen dinámicamente la estructura de los GOPs

[14] Existen casos en los que es interesante el uso de cuadros B como referencia de otros cuadros B

2.3.3. Selección de modo

Una característica novedosa introducida por H.264 es el conjunto de modos posibles de predicción. En el estándar se contemplan hasta siete modos de codificación[15], a saber, INTRA 4×4, INTRA 16×16, SKIP, INTER 16×16, INTER 16×8, INTER 8×16 e INTER 8×8. A todos estos modos hay que añadirles las subparticiones de macrobloque, subdivisiones adicionales en presentes en el modo INTER 8×8: INTER 8×8, INTER 8×4, INTER 4×8 e INTER 4×4.
Para realizar una codificación eficiente hay que ser capaces elegir el modo (y submodo) adecuado en cada momento y para cada macrobloque.

Esta selección de modo se realiza mediante el proceso de RDO, basado en un proceso de optimización que utiliza la formulación de Lagrange. En dicho proceso se define una función de coste que habrá que evaluar en todos los bloques de la secuencia y para todos los modos posibles, eligiendo como modo más adecuado para un bloque el que proporcione un menor coste [6, 7]. La función de Lagrange a minimizar es la siguiente:

$$J_{MODE} = (S_k, I_k \mid Q, \lambda_{MODE}) = D_{REC}\,(S_k, I_k \mid Q) + \lambda_{MODE} \cdot R_{REC}\,(S_k, I_k \mid Q)$$

donde

J_{MODE} es el coste a minimizar
S_k es el macrobloque que está siendo procesado
I_k es el modo de codificación contemplado ($\mathbf{I}$ es el conjunto de modos posibles)
Q es el valor del cuantificador
λ_{MODE} es el multiplicador de Lagrange
D_{REC} es la distorsión
R_{REC} es la tasa obtenida

La forma de evaluar la función de coste dependerá del tipo de modo contemplado (Intra, Inter o Skip):

- **Modos INTRA:** La distorsión $D_{REC}\,(S_k, I_k \mid Q)$ se mide como la SSD (*Sum of Squared Differences*, Suma de Diferencias Cuadradas) de los píxeles del macrobloque reconstruido (s') y del original (s):
$$SSD = \sum_{(x,y)\in A} \left| s\,[x,y,t] - s'\,[x,y,t] \right|^2$$
donde A es el macrobloque procesado.
La tasa $R_{REC}\,(S_k, INTRA \mid Q)$ será la tasa a la salida del codificador entrópico.

- **Modo SKIP:** La tasa $R_{REC}\,(S_k, SKIP \mid Q)$ y la distorsión $D_{REC}\,(S_k, SKIP \mid Q)$ no dependen del valor actual de QP. La distorsión se determina por la SSD entre el macrobloque actual y el obtenido por predicción. La tasa será, aproximadamente, un bit por cada macrobloque.

- **Modos INTER:** En los modos Inter el coste es más complejo de calcular, ya que en estos modos hay que incluir el paso de estimación de movimiento (elección del vector de

[15] Si consideramos las extensiones FREXt, también existe un modo INTRA 8×8

movimiento más adecuado en cada caso).
Para realizar esta tarea también se empleará un proceso de optimización de Lagrange, en el que se evaluará una función de coste para todos los posibles vectores de movimiento[16]. Esta función de coste es

$$m_i = \underset{m \in M}{\arg\min} \left\{ D_{DFD}\left(S_i, m\right) + \lambda_{MOTION} \cdot R_{MOTION}\left(S_i, m\right) \right\}$$

donde

m_i es el vector de movimiento resultante, el que da menor coste
D_{DFD} es el término de distorsión, determinado por

$$D_{DFD}\left(S_i, m\right) = \sum_{(x,y) \in A} \left| s\left[x, y, t\right] - s'\left[x - m_x, y - m_y, t - m_t\right] \right|^p$$

con p=1 para que la distorsión sea SAD y p=2 para ser SSD
S_i es el macrobloque que está siendo procesado
m es el vector de movimiento que está siendo evaluado
M es el conjunto de vectores de movimiento a estudio (típicamente ±32 posiciones de píxel enteras)
Q es el valor del cuantificador
λ_{MOTION} es el multiplicador de Lagrange
R_{MOTION} es el número de bits necesarios para transmitir las componentes del vector de movimiento y los índices de las imágenes de referencia utilizadas.

Una vez determinado el mejor vector para el modo, se calcula la distorsión, D_{REC} como la SSD entre los píxeles originales y los reconstruidos del macrobloque. La tasa R_{REC} será la suma de los bits empleados para la información del modo, los vectores de movimiento y los coeficientes de la transformada DCT. Para el caso de H.264 el multiplicador de Lagrange para el proceso de elección de modo mediante RDO es

$$\lambda_{MODE} = 0{,}85 \cdot 2^{(Q-12)/3}$$

y la relación entre λ_{MODE} y λ_{MOTION} es

$\lambda_{MOTION} = \sqrt{\lambda_{MODE}}$, si en el cálculo de D_{DFD} se ha empleado $p = 1$ (SAD)

$\lambda_{MOTION} = \lambda_{MODE}$, si en el cálculo de D_{DFD} se ha empleado $p = 2$ (SSD)

2.3.4. Perfiles y niveles

Las diferentes funcionalidades presentes en el estándar están especificadas por medio de perfiles. Éstos no son más que subconjuntos de elementos de sintaxis y herramientas disponibles para cumplir con una determinada parte del estándar. De este modo, dos secuencias sujetas a diferentes perfiles pueden tener grados de calidad y/o versatilidad muy diferentes, lo que haría a cada una adecuada para una aplicación diferente.

[16] La cantidad de vectores a analizar podrá estar limitada para reducir la carga computacional

Inicialmente, en la publicación del estándar en 2003 se establecieron tres perfiles: *Baseline*, *Main* y *Extended*. Posteriormente con la adición al estándar de las extensiones FREXt [5] el número de perfiles aumentó considerablemente para satisfacer las demandas de todos los sectores interesados. En H.264 actualmente [1] se definen 11 perfiles diferentes, con diferentes características y orientados a diferentes tipos de aplicaciones:

- **Baseline Profile (BP):** Orientado para aplicaciones de bajo coste. Ampliamente usado en videoconferencia y aplicaciones móviles.
- **Main Profile (MP):** Originalmente ideado como el perfil principal para aplicaciones de consumo. Cada vez menos usado a favor del High Profile.
- **Extended Profile (XP):** Perfil usado para aplicaciones de streaming. Posee mayor capacidad de compresión y más recursos para el tratamiento de errores y conmutación entre diferentes flujos.
- **High Profile (HiP):** Orientado para la difusión y almacenamiento de vídeos, especialmente de alta definición. Este perfil es el utilizado en el formato Blu-ray.
- **High 10 Profile (Hi10P):** Superior al High Profile, proporciona una precisión de 10 bits por muestra.
- **High 4:2:2 Profile (Hi422P):** Pensado para aplicaciones profesionales que utilizan vídeo entrelazado. Emplea muestreo 4:2:2 y precisión de 10 bits por muestra.
- **High 4:4:4: Predictive Profile:** Soporta muestreo hasta 4:4:4, hasta 14 bits por muestra, codificación de regiones sin pérdidas y la codificación de cada imagen como tres planos de color separados.

Los 4 perfiles que faltan son subconjuntos de los perfiles High que sólo utilizan codificación Intra. Son los denominados *all-Intra Profiles*:

- **High 10 Intra Profile:** Perfil High 10 utilizando solo codificación Intra
- **High 4:2:2 Intra Profile:** Perfil High 4:2:2 utilizando solo codificación Intra
- **High 4:4:4 Intra Profile:** Perfil High 4:4:4 utilizando solo codificación Intra
- **CAVLC 4:4:4 Intra Profile:** Perfil High 4:4:4 utilizando solo codificación Intra y codificación entrópica CAVLC (no usa CABAC)

Por otro lado, los niveles expresan los límites de rendimiento para cada perfil, poniendo límites a la tasa de muestreo, tamaño de las imágenes, bit rate generado y requisitos de memoria, a fin de regular la capacidad de procesamiento necesaria para cada aplicación. En H.264 se especifican un total de 16 niveles, cuyas características principales pueden consultarse en la sección A.3 de [1].

2.3.5. Novedades e innovaciones

El formato H.264/AVC incorpora un gran número de novedades e innovaciones respecto a anteriores estándares de compresión de vídeo como H.263 ó MPEG-2 [8]. Estas contribuciones están repartidas en todos los pasos del modelo DPCM/DCT empleado.

La primera novedad a destacar es la separación del proceso de codificación en dos capas: VCL (*Video Coding Layer*) y NAL (*Network Abstraction Layer*). La capa VCL se encargará del procesado del contenido visual, su análisis y compresión, mientras que la capa NAL se encargará de dar formato a los datos generados por la capa VCL, aplicar las cabeceras correctas y adaptar esta información a una variedad de capas de transporte para su adecuada transmisión o almacenamiento, proporcionando al mismo tiempo herramientas que hagan esta transmisión más robusta.

Además de las capas VCL y NAL, las principales novedades introducidas son las siguientes:

- Sistema de estimación y compensación de movimiento:
  - Tamaño de bloque para compensación de movimiento variable. Mayor flexibilidad y posibilidad de bloques de luma de hasta 4×4 muestras.
  - Precisión de vectores de un cuarto de muestra en lugar de, como mucho, media muestra, como se usaba en MPEG-4.
  - Posibilidad de que los vectores apunten más allá de los bordes de la imagen.
  - Compensación de movimiento con hasta 16 imágenes de referencia.
  - Independencia entre el orden de visualización de las imágenes y el orden para su referencia.
  - Posibilidad de utilizar imágenes tipo B (obtenidas mediante bi predicción) como referencia.
  - Predicción ponderada. Mejora la eficiencia en los *fades* o fundidos entre imágenes.
  - Modos de predicción: "*skipped*" y "*direct*" mejorados introduciendo predicción de movimiento en ellos.
- Codificación Intra mediante predicción de dirección espacial.
- Filtro en el bucle de decodificación para reducir los efectos de bloque y mejorar la calidad objetiva y subjetiva.
- Selección de modo de codificación mediante optimización R-D y función de coste de Lagrange.
- Bloque de transformación:
  - Transformada de pequeño tamaño. Uso de transformación de bloque de tamaño 4×4 (posibilidad de usar transformada 8×8 en perfiles FREXt).
  - Transformación jerárquica. Dependiendo del contenido se podrán utilizar diferentes tamaños de bloque para la transformación.
  - Aritmética necesaria de sólo 16 bit.

  - Transformación inversa exacta. Todos los decodificadores reconstruyen igualmente una secuencia.

- Codificación entrópica
  - Introducción de codificación aritmética: CABAC (*Context-Adaptive Binary Arithmetic Coding*).
  - Uso de codificación CAVLC (*Context-Adaptive Variable-Length Coding*).

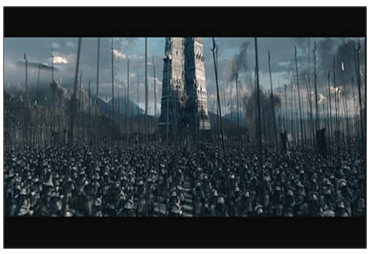
- Métodos para mejorar la robustez a la hora de transmitir
  - Nueva estructura de cabeceras más robusta y eficiente.
  - Introducción de la capa NAL.
  - Flexibilidad en el tamaño de las tiras.
  - Flexibilidad en el orden de los macrobloques (FMO, *Flexible Macroblock Ordering*). Se permite dividir la imagen en regiones llamadas "grupos de tiras", de manera que se pueden codificar y decodificar de manera independiente, consiguiéndose así una robustez adicional ante posibles errores.
  - Orden arbitrario de las tiras (ASO, *Arbitrary Slice Ordering*). Las tiras que constituyen una imagen se pueden enviar en cualquier orden, de manera que se puede mejorar el retardo extremo a extremo.
  - Imágenes redundantes. Se pueden enviar representaciones (generalmente con menos calidad) de determinadas imágenes o regiones de imágenes, de manera que un fallo en la imagen principal pueda ser parcialmente subsanado con la copia redundante.
  - Partición de los datos. Se pueden clasificar los datos hasta en tres categorías (A, B ó C) según su importancia o criticidad, de manera que cada categoría se envía por separado (en unidades NAL distintas), permitiendo un mejor comportamiento en presencia de errores. La partición A contiene las cabeceras (la información más sensible a los errores), la partición B contiene los residuos de las tiras I y SI y la partición C transporta los residuos de las tiras P y B.
  - Imágenes SI / SP. Introducción de dos nuevos tipos de imágenes que permiten la sincronización de múltiples decodificadores con un único flujo de información. Además, estos tipos de imágenes especiales permiten una conmutación eficiente entre diferentes flujos de vídeo.

Capítulo 3

Control de tasa

El control de tasa o *Rate Control* constituye un proceso crítico en la calidad, versatilidad y utilidad del codificador de vídeo que se esté contemplando. La naturaleza de la información de vídeo es inherentemente variable. El contenido de las imágenes es variante en el tiempo y en el espacio. La cantidad y naturaleza del movimiento capturado en la secuencia determina su complejidad temporal. Respecto a la información espacial, es obvio que el contenido de dos imágenes puede ser muy diferente y que en función de éste la información contenida en las imágenes variará, por lo que su complejidad también lo hará. Cuanto mayor sean estas complejidades, mayor será el ancho de banda requerido para almacenar las imágenes. Podemos ver unos ejemplos de esto en la figura 3.1:

(a) Poca complejidad espacial



(b) Gran complejidad espacial

(c) Poca complejidad temporal

(d) Gran complejidad temporal

Figura 3.1: Ejemplos de complejidades

Estas variaciones presentes en toda secuencia de vídeo deben ser controladas, a fin de que dicha secuencia pueda ser empleada en una aplicación concreta, con sus limitaciones e imposiciones. Estas limitaciones suelen venir impuestas por el hecho de que el ancho de banda disponible para *canalizar* la secuencia una vez comprimida puede estar limitado. Además, este límite puede variar con el tiempo, en función, por ejemplo, del estado de la red de tránsito. Igualmente, el retardo es un factor importante a la hora de elegir una determinada política de control de tasa. Cuando decimos *canalizar* no nos referimos sólo al hecho de transmitir por una red, sino también al uso de la secuencia en un conjunto aplicaciones donde no hay transmisión propiamente dicha, como puede ser el almacenamiento de vídeo en un soporte, donde existen limitaciones en cuanto al estado de los buffers de los dispositivos implicados y de los buses internos por donde discurre la información generada.

En los estándares de compresión de vídeo recientes, la parte relativa al control de tasa es informativa, es decir, no viene impuesta por la organización creadora del estándar sino que se deja abierta para estimular una evolución y mejora de las prestaciones por parte de los fabricantes de equipos.

Con el fin de conseguir una determinada tasa objetivo, un algoritmo de rate control ajusta dinámicamente determinados parámetros de la codificación, siempre con la condición de que el flujo a la salida sea conforme al estándar. Entre los parámetros susceptibles de ser ajustados se encuentran el parámetro de cuantificación o QP (el más importante), el frame rate, el multiplicador de Lagrange λ [9], la asignación de modos *skip* y *direct*, etc. El valor de esta tasa objetivo dependerá, primeramente, del requerimiento que tenga la aplicación, que podrá ser:

- **Requerimiento en forma de tasa.** La aplicación impone una serie de restricciones a la tasa generada por el codificador (tasa constante, máxima, mínima, retardo, tamaño y gestión del buffer, etc).
- **Requerimiento de calidad.** La aplicación impondrá un requisito de calidad constante o de mínima variación.

De todos los parámetros a tener en cuenta a la hora de aplicar un mecanismo de rate control, uno especialmente importante es el buffer. Para ver su importancia basta con tener en cuenta que el codificador funciona en base a un bitrate, esto es, tendrá una tasa objetivo que cumplir influenciada por las condiciones de la red o del mismo buffer que estamos tratando. Por el contrario, el decodificador funciona en base a un frame rate, o sea, tiene que extraer de la red (realmente del buffer de recepción) una cantidad de cuadros por segundo, independientemente de lo que ocupen. Por lo tanto, la generación de información y la extracción de la misma no siguen criterios (ni restricciones) iguales. Debido a esto, el buffer tiene una importancia crítica en una implementación práctica del codificador de vídeo, ya que se encargará de absorber las variaciones existentes entre la tasa generada y la tasa consumida, aparte de las variaciones que pudiera tener la red en un determinado momento.

Visto esto, en un algoritmo de rate control hay varios asuntos a los que hay que prestar una especial atención, especialmente los siguientes:

- **Distorsión.** Como se ha visto anteriormente, en un proceso de compresión de vídeo existe una fase de cuantificación, en la que en función de un parámetro QP[1] se puede regular la cantidad de detalle que se conservará en la secuencia de vídeo una vez restaurada. Si este QP tiene un valor bajo, se mantendrá una gran parte del detalle presente en la secuencia original, a cambio de resultar un bit rate elevado. Por el contrario, si QP tiene un valor alto el nivel de detalle se verá reducido y el bit rate generado disminuirá. Queda evidente que, debido al proceso de cuantificación, la secuencia reconstruida no es exactamente igual a la original, ya que se ha introducido una distorsión no salvable. El objetivo de todo algoritmo de rate control es conseguir la máxima calidad (o la mínima distorsión) cumpliendo con una limitación impuesta en el bit rate. La calidad típicamente se mide mediante la PSNR, mientras que la distorsión puede ser medida de diferentes maneras, dependiendo de la propuesta de algoritmo contemplada.

- **Complejidad computacional.** Cada aplicación tiene una capacidad computacional concreta, por lo que sus requisitos en este aspecto deben ser muy tenidos en cuenta. Si se busca un rendimiento aceptable, el algoritmo de rate control debe ser lo suficientemente sencillo. En caso de buscar una solución óptima, el algoritmo deberá tener una complejidad computacional asumible y razonable.

- **Limitaciones propias de la aplicación.** Como se ha mencionado anteriormente, existen restricciones impuestas por la propia aplicación, como pueden ser el retardo, el jitter, limitaciones en los buffers, la tasa o el espacio disponible si estamos en una aplicación de almacenamiento de vídeo.

Como se ha mencionado anteriormente, existe una relación entre la tasa y la distorsión. Esta relación se basa en la teoría de tasa-distorsión, más conocida como teoría *Rate-Distortion*, o simplemente R-D ó RD. Esta teoría establece que una reducción en la distorsión conlleva un aumento en la tasa. Se define una función R-D para describir el límite inferior de la tasa para una determinada distorsión. Dicho de otro modo, la función R-D determina la tasa mínima a emplear para obtener una distorsión objetivo. Sin embargo, no hay garantías de que este límite inferior sea alcanzable en condiciones prácticas. Esta función, según la teoría mencionada, es continua. Esto no es aplicable a la compresión de vídeo con pérdidas debido a que en ésta sólo son posibles unos determinados pares R-D, ya que en el proceso de rate control lo que se toca es, básicamente, el parámetro QP, que es de naturaleza discreta (en H.264/AVC sólo puede tomar 52 valores enteros, entre 0 y 51). Es por esto por lo que se utiliza la teoría ORD (*Operational Rate Distortion*) [11]. La función ORD presenta la curva convexa característica del mecanismo de compresión utilizado, pero a diferencia de la función R-D, se pueden obtener los valores óptimos de QP que proporcionan una mínima distorsión a una determinada tasa. Este método, aunque proporciona valores óptimos para QP no es eficiente en situaciones prácticas, especialmente en codificación de vídeo en tiempo real, debido a que la generación de la *curva* ORD con pares R-D reales (figura 3.2) tiene un coste computacional enorme, muchas veces inasumible que, además, introduce un retardo inaceptable. Es por esto que en situaciones prácticas se utilizan modelos para la función R-D (figura 3.2), que aunque no den la solución óptima sí darán una solución razonablemente buena con un coste computacional muy inferior. Estos modelos se pueden generar

[1] El parámetro puede ser el Parámetro de Cuantificación, QP, o directamente el escalón de cuantificación utilizado (Q_{step}). En cualquier caso, existe una relación biunívoca entre QP y Q_{step}.

explotando las propiedades estadísticas de la secuencia de vídeo o mediante un procedimiento empírico y un proceso de regresión posterior.

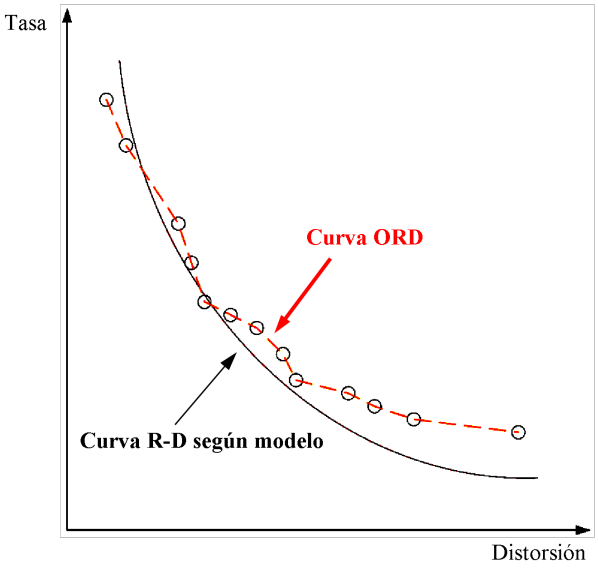


Figura 3.2: Comparación de curva ORD y curva R-D según un modelo. El error introducido puede ser importante.

Generalmente, el proceso de rate control se divide en dos subprocesos: Bit Allocation y Bit Rate Control. En el subproceso denominado Bit Allocation se determina un cupo de bits a utilizar para codificar la unidad básica que se esté contemplando (GOP, cuadro, tira o macrobloque), mientras que en el Bit Rate Control se realiza un control para utilizar el cupo asignado en la primera fase lo más completamente posible, sin sobrepasarlo ni que sobren bits.

3.1. Tipos de control de tasa

El mecanismo de control de tasa puede ser clasificado en función de numerosos criterios, como puede ser el grado de optimización, el número de pasadas realizadas, si se realiza o no en tiempo real o si es de bit rate constante o variable.

3.1.1. Control de tasa óptimo vs. no óptimo

El control de tasa óptimo [11] consiste en minimizar la distorsión obtenida cumpliendo un requisito impuesto en la tasa. Los métodos de control óptimo de la tasa se centran en la distorsión obtenida, por lo que pueden ser vistos como una solución para el requisito de calidad constante. El control óptimo hace un uso intensivo de las características *Rate-Distortion*. Tomaremos que para un cuadro genérico n, el bit rate requerido es R_n y la distorsión obtenida es D_n, medida según un determinado criterio. El vídeo digital utiliza compensación de movimiento, por lo que hay que tener en cuenta una importante interdependencia entre cuadros vecinos, de manera que

una decisión tomada para uno de estos cuadros puede afectar en mayor o menor medida a sus cuadros vecinos. De este modo, para determinar el parámetro de cuantificación del cuadro n, QP_n, deberemos considerar también los a cuadros precedentes y los b cuadros sucesivos. Todo esto suponiendo que estemos trabajando a nivel de cuadro. Si estamos trabajando a nivel de macrobloque, por ejemplo, deberemos considerar todos los macrobloques vecinos (según una ventana que hayamos definido previamente) de todos los cuadros vecinos, o sea, muchísimos más elementos a considerar. Esto nos sirve para hacernos una idea de la enorme capacidad computacional que requieren estos algoritmos.

Estos algoritmos se centran en manejar la distorsión, según una política concreta. Estas políticas pueden ser el obtener la mínima distorsión media (MINAVE), problema que responde a la siguiente expresión:

$$\mathbf{Q}^* = (Q_1^*, \ldots, Q_N^*) = \underset{(Q_1,\ldots,Q_N)}{\arg\min} \left(\sum_{n=1}^{N} D_n \left(Q_{n-a}, \ldots, Q_{n+b} \right) \right)$$

Del mismo modo, otra política a seguir es minimizar la máxima distorsión alcanzada (MINMAX):

$$\mathbf{Q}^* = (Q_1^*, \ldots, Q_N^*) = \underset{(Q_1,\ldots,Q_N)}{\arg\min} \left(\underset{n\in\{1,\ldots,N\}}{\max} \left\{ D_n \left(Q_{n-a}, \ldots, Q_{n+b} \right) \right\} \right)$$

Finalmente tenemos la estrategia de minimizar la variación de la distorsión obtenida (MINVAR):

$$\mathbf{Q}^* = (Q_1^*, \ldots, Q_N^*) = \underset{(Q_1,\ldots,Q_N)}{\arg\min} \left(\sum_{n=2}^{N} \left| D_n \left(Q_{n-a}, \ldots, Q_{n+b} \right) - D_{n-1} \left(Q_{n-a-1}, \ldots, Q_{n+b-1} \right) \right| \right)$$

A estas estrategias se le pueden añadir limitaciones, como una limitación en el nivel del buffer ($0 \leq B_n \leq B_{max}$) o una limitación de espacio máximo, igual a R_{max}, en el caso de una aplicación de almacenamiento de vídeo $\left(\sum_{n=1}^{N} R_n \left(Q_{n-a}, \ldots, Q_{n+b} \right) \leq R_{max} \right)$. Viendo esto, queda patente la importancia y criticidad del proceso de bit allocation. Para su realización se ha propuesto la utilización de la optimización de Lagrange, que consiste en el uso de la teoría de multiplicadores de Lagrange para obtener el cupo de bits requerido en cada caso.

Los algoritmos no óptimos aparecen en respuesta a la grandísima carga computacional requerida por los algoritmos óptimos. Estos algoritmos pueden no garantizar el mejor resultado posible, pero sí uno razonablemente bueno. Se basan en aproximaciones y modelos realizados previamente de las funciones R-D y de la generación de datos mediante herramientas estadísticas y/o el estudio previo de la secuencia de vídeo que se va a codificar. Gracias a ellos el proceso es computacionalmente asumible (en algunos casos hasta ligero) y los resultados son satisfactorios.

3.1.2. Control de tasa en tiempo real vs. no tiempo real

Como ya se ha comentado, el control de tasa es un mecanismo cuya elección está fuertemente relacionada con la naturaleza y propósito de la aplicación en cuestión. En cuanto a los requisitos temporales de la codificación, existen dos tipos de aplicaciones de compresión: compresión en

tiempo real (o *Real Time*, *RT*) y compresión no en tiempo real o compresión en tiempo diferido (o *Non Real Time*, *NRT*) [12]. Por aplicaciones de tiempo real entendemos aplicaciones en las que la captura y la posterior compresión deben realizarse en un tiempo lo más pequeño posible. Esto es debido a unas exigencias en retardo y jitter muy elevadas, no sólo desde el punto de vista de la red, sino también desde el punto de vista de la experiencia del usuario final de la aplicación. Por lo tanto, el algoritmo de control de tasa debe ser *movido* ágilmente por el sistema. Entre las aplicaciones que hacen uso de estas condiciones estrictas de tiempo real podemos reseñar las aplicaciones de videoconferencia. En ellas, el retardo es muy importante, ya que condiciona la fluidez de la comunicación, y el jitter debe estar lo más reducido posible, ya que de lo contrario la comunicación resulta molesta al percibirse que va *a tirones*.

Por otro lado tenemos las aplicaciones en tiempo diferido, o NRT. En ellas el tiempo de codificación y procesado no constituye ninguna limitación. En ellas podremos utilizar algoritmos de rate control más precisos y computacionalmente más costosos. Entre éstas podemos indicar las aplicaciones de almacenamiento de vídeo, como puede ser Blu ray, en la que el vídeo almacenado previamente sin comprimir es comprimido para posteriormente ser grabado en discos. Es evidente que el retardo introducido no constituye ningún obstáculo y el jitter tampoco, ya que se entiende que los sistemas en los que se reproducirán las secuencias de vídeo serán capaces de hacerlo en condiciones (por estar el estándar limitado a una serie de perfiles y niveles que cumplirán codificador y decodificador).

Entre ambos extremos tenemos otras aplicaciones, que podríamos introducir entre las aplicaciones en tiempo real, pero con requisitos algo más relajados. Como representante de estas aplicaciones podemos incluir la difusión en directo de vídeo por televisión, como puede ser la televisión por satélite o la televisión digital terrestre. En éstas, el vídeo debe ser codificado lo más rápido posible, pero teniendo en cuenta que introducir un pequeño retardo adicional no supone un perjuicio demasiado grande, siempre que gracias a él obtengamos una mayor calidad de imagen. Por poner un ejemplo de este caso, un espectador no se va a dar cuenta de si un informativo le llega con unos segundos de retardo o justo en el momento en el que el presentador está informando de la noticia.

3.1.3. Control de tasa en una pasada vs. varias pasadas

Otra posible clasificación de los algoritmos de rate control es la realizada en base al número de pasadas necesarias para su realización. Por *pasada* entenderemos un escaneo de la secuencia de vídeo para un determinado fin. Un algoritmo de una pasada será aquel en el que en ningún momento hay dos o más cuadros en el codificador pendientes de ser procesados. Esto es, solo procesaremos un cuadro cuando el anterior ya ha sido comprimido. Esto quiere decir que un cuadro se codificará con la información que seamos capaces de extraer de él y de la información que hayamos podido obtener de los cuadros anteriores, pero en ningún caso de la información contenida en cuadros futuros. La compresión en una pasada tiene el inconveniente de que, al no conocer nada sobre los cuadros futuros, hay que tomar decisiones y realizar tareas de forma *ciega*, sin saber si los cuadros venideros son más complejos (y que requieren más bits) o menos complejos (requiriendo menos bits), pudiendo provocar problemas en el bit allocation.Todo esto pone de manifiesto una gran dificultad por parte del esquema en una pasada de proporcionar

vídeo con una calidad realmente estable, que es la base de la calidad visual percibida. Este esquema de rate control es utilizado en aplicaciones en tiempo real y, típicamente, en sistemas con una complejidad moderada o pequeña.

En el otro extremo nos encontramos las aplicaciones de compresión de vídeo en varias pasadas (generalmente, dos pasadas). En la primera pasada lo que se hace es recorrer la secuencia de vídeo completa para extraer sus características. Evidentemente esta tarea no se puede realizar en tiempo real, ya que necesitamos tener almacenada la secuencia completa de antemano. La codificación en dos pasadas proporciona una mayor calidad que la realizada en una sola pasada, ya que permite un mejor ajuste de los parámetros y una mayor uniformidad de la calidad, debido a que las decisiones tomadas en la codificación ya no se realizan a ciegas, como en el caso anterior, sino con el conocimiento de las imágenes venideras. Es evidente que también requiere un mayor esfuerzo computacional, ya que la codificación se puede realizar en más de una ocasión. En determinados escenarios, puede realizarse una tercera pasada, pero en este caso la revisión de calidad de la tercera pasada no la hace una máquina, sino una persona para poder realizar una valoración subjetiva de la calidad[2].

3.1.4. Control de tasa CBR vs. VBR

Como ya se ha comentado, el control de tasa es un mecanismo que se encarga de ajustar el volumen de datos generado por el codificador de vídeo a una serie de requisitos impuestos por la aplicación. Entre estos requisitos están las imposiciones sobre el tipo de tráfico generado, bit rate constante (CBR) o bit rate variable (VBR) [12]. Esta distinción resulta fácil de entender si asumimos que después de codificar una secuencia de vídeo hay que hacer «algo» con dicho vídeo codificado. Este «algo» puede ser varias cosas, pero principalmente se reduce a dos: transmitir el vídeo y/o almacenarlo en algún medio.

3.1.4.1. Rate control CBR

Este esquema de control de tasa es el que impone una tasa constante a la salida del codificador. Su formulación sería la siguiente:

$$\text{mín}\, D : R = R_{max} = R_{CTE}$$

donde D es la distorsión obtenida en la imagen recuperada
R es la tasa instantánea generada
R_{max} es la tasa constante requerida por la aplicación (R_{CTE})

En este esquema el primer objetivo a cumplir es ajustarse a la tasa (constante) impuesta por el sistema o aplicación, antes de cumplir unos requisitos de calidad. En todo caso, la calidad debe ser aceptable, pero es algo secundario.

[2] No hay que olvidar que la calidad es un concepto inherentemente subjetivo, no medible por máquinas. La llamada calidad objetiva no es sino una correspondencia más o menos precisa entre unas mediciones hechas y los resultados obtenidos en estudios a partir de observadores humanos.

Una vez conseguida la estabilidad en la tasa, el gran reto de este sistema de rate control reside en ofrecer un nivel de calidad lo suficientemente elevado. Es sabido que el factor más importante sobre la calidad percibida no responde a la fidelidad que tiene la imagen reconstruida respecto a la imagen original sin comprimir, como se podría pensar en un principio. El factor más importante responde a las variaciones que presenta la calidad a lo largo del tiempo. Un observador valorará más positivamente una secuencia con una mayor distorsión respecto del original (medida con el criterio que sea, MSE, PSNR, etc.) siempre que ese nivel de distorsión se mantenga estable y acotado; por el contrario, el mismo observador puntuará negativamente una secuencia con un nivel de distorsión inestable, con variaciones importantes en el tiempo, aunque en algunos momentos llegue a presentar tasas de fidelidad respecto al original muy elevadas. De hecho, aquí es donde está el reto de estos sistemas. Como se ha comentado al inicio de este capítulo, el vídeo es un medio cuyo contenido, temporal y espacialmente, es altamente fluctuante y esto se refleja en la cantidad de datos generada por un codificador. Un vídeo con una tasa 100 % constante es, en cualquier caso, una quimera. Si se quiere obtener una tasa constante se debe sacrificar la calidad, ya que ésta variará inversamente con la complejidad de la secuencia que se esté codificando. Para tratar de suavizar este problema se ha contemplado el uso de buffers en codificador (ver figura 3.3) y decodificador, de manera que sean capaces de absorber pequeñas variaciones en el flujo de datos generado.

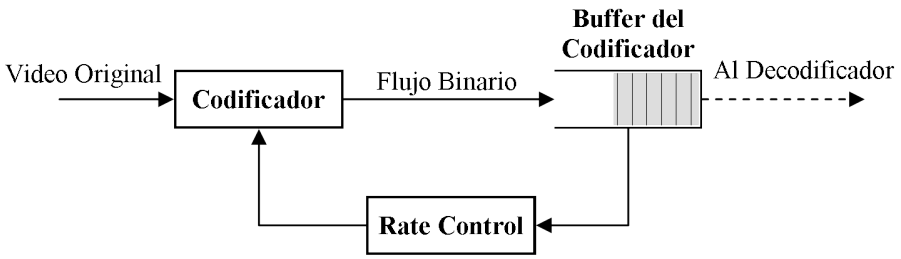


Figura 3.3: Mecanismo de Rate Control con buffer asociado

Sin embargo, el uso de buffers aporta un inconveniente, que es el aumento del retardo en el caso de una aplicación de transmisión de la señal de vídeo. En una aplicación interactiva, como puede ser una videoconferencia, este retardo debe ser lo menor posible. Esto, además del problema de la calidad, ha impulsado el desarrollo de mecanismos de codificación y transmisión de vídeo de bit rate variable, o VBR.

3.1.4.2. Rate control VBR

El rate control de bit rate variable, o VBR [13], surge principalmente de la necesidad de obtener secuencias de vídeo con una calidad más estable que la obtenida con CBR.

Básicamente, un rate control VBR se puede caracterizar mediante un conjunto de parámetros, como las tasas medias y máxima y la cantidad de datos generados y el límite impuesto a estos últimos, medido sobre los generados a la tasa media. Estos conceptos los podemos en la figura 3.4:

Una codificación VBR requiere un proceso de bit allocation variable, de manera que se asignen

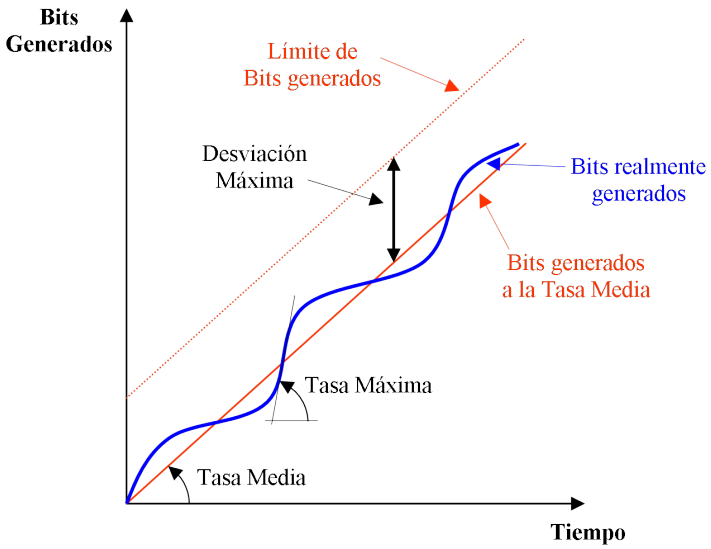


Figura 3.4: Magnitudes involucradas en en Rate Control VBR

bits a cada cuadro en función de su complejidad. De este modo se ahorran bits en los cuadros menos complejos para posteriormente invertirlos en los cuadros más complejos, que requerirán un bit rate mayor. El principal problema para conseguir este objetivo es el desconocimiento del contenido de los cuadros futuros, por lo que la estrategia de codificación VBR para calidad constante sólo será posible a efectos prácticos en aplicaciones que no requieran codificación en tiempo real. De lo contrario habría que introducir un retardo en la codificación inaceptable para la mayoría de aplicaciones para poder examinar el contenido de varios cuadros. Aun así, el objetivo de calidad constante no podría garantizarse a toda la secuencia, sino al conjunto de cuadros almacenados en el buffer para su análisis. El enfoque VBR puede ser adoptado, por ejemplo, para aplicaciones de almacenamiento de vídeo, como puede ser la autoría de DVD o Blu-ray. En estos casos, el requisito impuesto, aparte de la mencionada calidad constante, será no exceder el espacio disponible para el almacenamiento.

Esta estrategia a menudo va unida a una codificación en múltiples pasadas (generalmente dos), de manera que con la primera podamos determinar qué cuadros o fragmentos de la secuencia requieren un bit rate mayor y qué cuadros o fragmentos requieren un bit rate menor, así como una estimación de la calidad que se obtendrá finalmente. En la segunda pasada se utilizará la información recogida en la primera para ajustar los parámetros del codificador y poder realizar la codificación con la mínima variación de calidad posible.

Además de la capacidad para proporcionar calidad constante, la codificación VBR tiene la ventaja de ser más adecuada para ser transmitida a través de las redes de comunicaciones utilizadas hoy en día, basadas en conmutación de paquetes o celdas. Aparte de esto, ciertas redes son capaces de proporcionar garantías de calidad de servicio, *QoS*, mientras que otras, como sucede en Internet, proporcionan un servicio puramente *best-effort*. Podemos concluir que las redes utilizadas hoy en día tienen generalmente un comportamiento VBR[3]. De cualquier modo, la calidad total obtenida en una aplicación de transmisión de vídeo a través de una de

[3] Algunas redes también pueden ser configuradas como CBR, como el caso de la categoría CBR de ATM

estas redes depende de ambos factores: los parámetros de QoS de la red y los parámetros de codificación empleados. Esto se puede ver con un ejemplo sencillo: la calidad percibida será mayor si transmitimos vídeo de baja calidad para así tener menos pérdidas en la red, mientras que la calidad percibida será menor si transmitimos vídeo de alta calidad pero perdemos paquetes en la transmisión, ocasionando cortes y degradaciones bruscas de calidad. Si el sistema permite que el codificador tenga información sobre la red, éste podrá adaptarse a eventualidades que puedan surgir, mejorándose la calidad obtenida (o degradándose lo menos posible).

Si utilizamos un esquema de codificación VBR junto con una red de transmisión también VBR veremos que VBR tiene una serie de ventajas frente a CBR:

- Mayor calidad de imagen, ya que el bit rate puede ser distribuido entre diferentes cuadros según su complejidad
- Menor retardo, debido a un requisito de buffering menor
- La capacidad efectiva del canal se ve incrementada, ya que la unión de vídeo VBR y red VBR permite emplear múltiplexación estadística. Esto quiere decir que en una red será capaz de transportar más flujos VBR que flujos CBR

Estas ventajas conllevan la aparición de una situación de compromiso, ya que no pueden ser maximizadas simultáneamente.

3.1.4.2.1. Modos de Operación en VBR

Como se ha mencionado anteriormente, en un sistema compuesto por una codificación de vídeo VBR y una transmisión también VBR, la calidad final percibida depende de los parámetros de *ambos* subsistemas. El nivel de conocimiento que tenga el codificador sobre el estado del subsistema de transmisión puede ser determinante a la hora de conseguir una calidad satisfactoria, sobre todo ante eventualidades que pudieran surgir, de tal forma que podría adaptar la estrategia de codificación al estado actual de la red. El grado de interrelación entre el subsistema de codificación y la red puede ser más o menos extenso. Dependiendo de esto tenemos los denominados *modos de operación VBR* [13], que se corresponde con los niveles de relación codificador-red. Podemos distinguir cuatro modos diferentes, ateniendo a los diferentes componentes que existen en sistema: U-VBR, S-VBR, C-VBR y F-VBR, mostrados en la figura 3.5.

- **U-VBR (Unconstrained VBR):** En este modo el codificador trabaja de manera independiente a la red y a la UNI (*User to Network Interface*, Interfaz Usuario-Red). A esta configuración también se la conoce como «Codificador en Lazo Abierto». La tasa de salida del codificador es elegida ateniendo únicamente a los requisitos de calidad establecida, sin considerar ninguna característica, requisito ni limitación por parte de la red o de su interfaz. Desde el punto de vista de rendimiento, este modo se puede asumir que representa el rendimiento obtenible por el sistema en el escenario más optimista posible. En este modo, la tasa de salida es, en teoría, fuertemente cambiante, en función de las características de los cuadros codificados, como pueden ser su complejidad o su tipo (por ejemplo, un cuadro B generará mucho menos tráfico que un cuadro I), lo que lo convierte en un modo poco

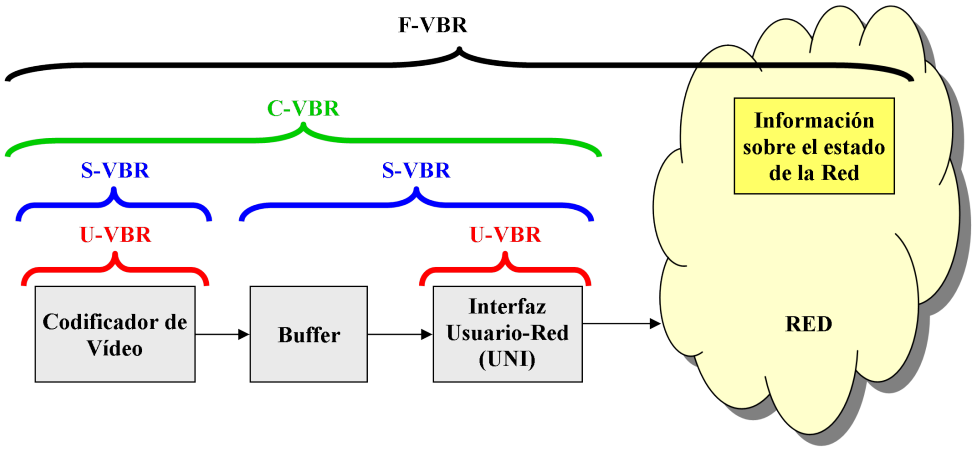


Figura 3.5: Modos de operación VBR

adecuado o, cuanto menos, de aplicación muy compleja, para entornos reales de transmisión a través de una red de datos, ya que la gestión y dimensionamiento de la misma serían muy difíciles. Como ejemplo de este modo se puede poner un codificador operando con una QP constante, de manera que el bit rate generado dependerá, básicamente, de la complejidad de la secuencia. Este modo es, sobre todo, utilizado a la hora de realizar estudios de tráfico para redes, así como modelados de generación de datos por los codificadores de vídeo.

- **S-VBR (Shaped VBR):** Este modo es una evolución del anterior, en el que se ha añadido un buffer a la salida del codificador. Este buffer realizará un conformado del tráfico que reciba del codificador, de manera que a su salida el flujo de datos será más suave (más o menos en función de su tamaño y configuración), y por lo tanto más manejable por la red, a cambio de introducir un retardo adicional en la transmisión (nuevamente en función de la configuración del buffer). El formato más sencillo del buffer conformador es emplear una política *Leaky Bucket*, de manera que se puede controlar tanto el retardo como la tasa sostenida o la ráfaga máxima de salida a la red, haciendo más sencilla la tarea de supervisión de la red para verificar el cumplimiento del contrato de tráfico. Es importante reseñar que el buffer no interacciona con el codificador. Éste sigue funcionando de la misma manera que en el modo U-VBR, por lo que el flujo binario saliente del sistema es igual al caso U-VBR. Un ejemplo de este modo concreto se puede encontrar en [14].

- **C-VBR (Constrained VBR):** En este escenario el codificador es consciente del estado del buffer de conformado, así como del estado y de las limitaciones impuestas por la UNI. De este modo, el codificador puede modular su tasa de salida, adaptándola a la información de estado del buffer, así como a las imposiciones de la UNI, como puede ser la satisfacción del contrato de tráfico con la red. Así se consigue maximizar la calidad de la secuencia de vídeo, modificando determinados parámetros de la codificación (generalmente el parámetro QP) y controlando la tasa de salida (tanto sostenida como de pico), además del retardo, al mismo tiempo que se evitan problemas que puedan surgir en el buffer de conformado, como pueden ser su vaciado (o *underflow*) o su desbordamiento (u *overflow*). Resulta claro que en este modo de funcionamiento, al contrario que en los anteriores, el flujo binario sí se ve alterado en función del estado del buffer y/o la UNI. Se puede demostrar que la calidad final conseguida empleando un sistema C-VBR es superior a la obtenida mediante U-VBR

y S-VBR, empleando la misma tasa media inyectada a la red[4].

- **F-VBR (Feedback VBR):** En este modo, el codificador tiene conocimiento global de toda la transmisión. Es consciente, además del estado del buffer y de las restricciones de la UNI, del estado de la red en todo momento. Gracias a esto, el codificador puede adaptarse a situaciones eventuales que pudieran surgir en cualquier momento, como pueden ser situaciones de congestión en la red. Esto se consigue gracias a una realimentación hacia el codificador por parte de la propia red. Un ejemplo de red que soporta este tipo de transmisión con realimentación acerca del estado es el servicio ABR (*Available Bit Rate*) de las redes ATM.

3.2. Control de tasa en H.264/AVC

Desde la publicación del estándar H.264 en 2003 se han sucedido una gran cantidad de propuestas destinadas a la mejora de las prestaciones de dicho estándar mediante la optimización de las técnicas de control de tasa. Cabe destacar que el desarrollo de estos algoritmos ha partido prácticamente desde cero, ya que la estructura interna del CODEC H.264 supuso una gran evolución respecto a los CODECs existentes. Técnicas como la optimización rate distortion mediante funciones de coste de Lagrange, la gran cantidad de modos de codificación existentes, la predicción/compensación de movimiento utilizando bloques de tamaño variable, etc., inexistentes en estándares anteriores como MPEG-2, han obligado al desarrollo de nuevos algoritmos.

Gracias a la introcucción de novedosas herramientas de codificación y al uso de subsistemas de control de tasa, el estándar H.264/AVC logra conseguir una mejora de hasta el 50 % en eficiencia de codificación respecto a MPEG-2 y hasta un 47 % respecto a H.263 Baseline [15]. Todo esto redunda en una mayor sofisticación del sistema completo de compresión. Como en los estándares de compresión de vídeo recientes, el control de tasa en H.264 constituye una parte informativa dentro de dicho estándar. Esto quiere decir que su metodología no está fijada estrictamente por el estándar (en ese caso se diría que es una parte *normativa*), sino que se deja libertad de desarrollo por parte de los investigadores y desarrolladores para mejorar su rendimiento. Lo único que H.264 explica al respecto es una serie de tratamientos internos que pueden afectar al binomio tasa-calidad, como puede ser el mecanismo de selección de modo, etc.

El mecanismo de control de tasa regula la cantidad de bits producidos en la codificación modificando, entre otros, el parámetro de cuantificación, QP, de manera que dependiendo de los valores que tome éste la imagen reconstruida preserve más o menos detalles respecto de la imagen original capturada (y sin comprimir). Por lo tanto, la clave en este mecanismo será el evaluar la relación entre QP, tasa y complejidad presente en la imagen, de manera que el propósito del control de tasa será resolver el compromiso tasa $\leftrightarrow$ calidad.

Ya que se procede a modificar el parámetro QP, es importante preguntarse a qué estructuras afecta la variación de este parámetro y en qué medida afecta esto a la calidad final. Si repasamos el esquema general del codificador H.264 podemos determinar que el parámetro de cuantificación

[4]En el caso de C-VBR y S-VBR esta tasa es la de vaciado del Leaky Bucket, mientras que en U-VBR es la tasa media generada por el codificador.

sólo tiene influencia en la información contenida en los residuos una vez transformados mediante la DCT. Por lo tanto, QP no tiene ningún efecto sobre la información relacionada con el *overhead* (o cabeceras), los datos obtenidos por predicción o los vectores de movimiento[5]. En estudios posteriores, datos como el número de bits asignados mediante bit allocation o las complejidades medidas deberán ser referidas únicamente al residuo.

El mecanismo de rate control incluye una función R-D, de manera que se pueda relacionar tasa, QP y complejidad. La función elegida en H.264 ha sido la misma que se ha utilizado en MPEG-4, y es un modelo cuadrático que se corresponde con la ecuación

$$R = \frac{C_1 \times MAD}{QP} + \frac{C_2 \times MAD}{QP^2} + \hat{h}$$

donde

$\hat{h}$ es el número de bits para la cabecera y vectores de movimiento (recordar que QP no afecta a estos elementos, de ahí que aparezca restando). Es un valor estimado mediante un determinado algoritmo, aunque es común utilizar la aproximación $\hat{h} = h^{prev}$ (aproximar el número de bits de cabecera del cuadro actual por el del cuadro anterior), suponiendo una correlación temporal con el cuadro anterior.

MAD (*Mean Absolute Difference*) es la medida de la complejidad del residuo y se calcula de la siguiente manera:

$$MAD = \sum_{i,j} |\text{residuo}_{i,j}|$$

QP es el parámetro de cuantificación que queremos determinar.

C_1, C_2 son parámetros del modelo que son actualizados mediante regresión lineal a partir de parámetros previos.

Como se ha indicado anteriormente, en H.264 existe una gran variedad de modos de codificación, en función de si estamos ante un mecanismo de predicción Intra o Inter. De entre todos los modos (y submodos) posibles habrá que elegir el que se considere más adecuado para realizar la codificación del macrobloque, elección que se realizará en base a un determinado criterio.
A este proceso de selección del modo a emplear se le denomina *Optimización de Rate Distortion*, o RDO. Como se habló en el apartado 2.3.3, en H.264 se ha adoptado un mecanismo de optimización basado en multiplicadores de Lagrange, consistente en minimizar una función de coste, de manera que se encuentre una solución de compromiso entre la tasa y la distorsión. Este proceso de RDO constituye un paso crítico en la compresión de la secuencia, así como una de las fases de gran coste computacional. Los pasos del algoritmo RDO son los siguientes:

1. Calcular la tasa y la distorsión de todos los modos y submodos posibles.
2. Evaluar la función de coste R-D de cada modo.
3. Seleccionar el modo con menor coste.

[5] Ciñéndonos al cuadro actual. El hecho de introducir más o menos distorsión en el cuadro actual podría afectar a cuadros posteriores que utilicen al actual como referencia

El problema de este método es que el mecanismo de rate control necesita conocer el valor de QP para poder realizar el proceso RDO, ya que éste depende de un parámetro λ (el multiplicador de Lagrange), que a su vez depende del valor de QP ($\lambda_{MODE} = 0{,}85 \cdot 2^{(Q-12)/3}$). Al mismo tiempo, es necesario conocer el modo de codificación para poder determinar el valor de la complejidad, medida mediante la MAD de la imagen recompuesta. Además de esto, para poder determinar el valor de QP es necesario conocer cuánto vale MAD para poder introducirlo en el modelo cuadrático antes expuesto. Estamos frente al «dilema del huevo y la gallina»[6]. Todos los parámetros están acoplados, dependen fuertemente entre sí, como se muestra en la figura 3.6.

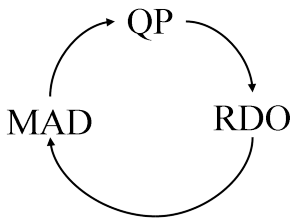


Figura 3.6: Dilema del huevo y la gallina: Dependencias entre parámetros.

Para deshacer este bucle se ha optado por una estrategia consistente en predecir la MAD mediante regresión lineal a partir de la MAD del cuadro anterior:

$$\widehat{MAD}_{actual} = a_1 \cdot MAD_{anterior} + a_2$$

donde

$\widehat{MAD}_{actual}$ es la MAD estimada para el cuadro actual.
$MAD_{anterior}$ es la MAD real obtenida (y medida) en el cuadro actual.
a_1, a_2 son parámetros del modelo. Son actualizados mediante regresión lineal.

Esta estrategia adoptada se basa en la asunción de que esta complejidad variará de forma gradual y suave a lo largo de la secuencia, por lo que no es extraño suponer que la MAD del cuadro actual será similar a la del cuadro anterior. El gran problema de este método es que falla en los cambios de escena, donde el cuadro actual no tiene nada que ver con el cuadro anterior.

Por otro lado, en H.264 el mecanismo de rate control se estructura en tres niveles:

- **Nivel GOP:** Realiza el bit allocation para un GOP completo y calcula el número de bits restante para codificar los cuadros pendientes de codificar de dicho GOP. En esta fase se definen los QPs para el cuadro I y el primer cuadro P del GOP. Al iniciar un GOP, también se encarga de inicializar el QP para el cuadro IDR[7], en caso de que sea necesario.
- **Nivel Cuadro:** Para un cuadro dentro de un GOP, se calcula el QP necesario para su codificación a partir de la tasa asignada para dicho cuadro y de la MAD predicha a partir del cuadro anterior. Después de determinar la QP se utiliza ésta para realizar el proceso de optimización de *rate distortion* (RDO) sobre el cuadro. Para realizar esto se utiliza el

[6] En la literatura es frecuente verlo expresado como *Chicken and Egg Dilemma*
[7] IDR: *Instantaneous Decoding Refresh*, imagen de tipo I o SI que borra el contenido del buffer de imágenes de referencia.

modelo R-D cuadrático. Generalmente la asignación de QP a una imagen viene condicionada por el valor de la QP de la imagen anterior, no permitiéndose cambios bruscos que pudieran afectar a la calidad percibida. Típicamente, esta variación de QP queda limitada a no más de ± 2 unidades.

- **Nivel Unidad Básica:** Este nivel puede no estar presente. Si en el codificador se requiere un control más fino sobre el bit rate generado y la calidad obtenida se puede hacer un estudio a más bajo nivel que una imagen, a nivel de unidad básica. Dicha unidad básica puede ser en este caso una tira, una fila de macrobloques, un conjunto de macrobloques o macrobloques tomados individualmente. En este caso se parte de una asignación de bits para esa unidad básica y se utiliza un modelo lineal para predecir la MAD de dicha unidad básica. Con esto, se utiliza el modelo R-D cuadrático para calcular la QP con la que aplicar el proceso RDO a la unidad básica.

3.3. Metodología y algoritmos destacados

3.3.1. Esquema típico de un algoritmo de rate control

El mecanismo de control de tasa se puede esquematizar en una serie de tareas fundamentales, involucrando cada una una serie de medidas y unos algoritmos concretos:

- **Estimación de complejidad:** La estimación y cálculo de las complejidades de las unidades básicas (cuadro, macrobloque, etc.) constituye un paso clave en la consecución de un mecanismo de rate control efectivo, ya que es una magnitud que permite establecer un criterio para redistribuir los bits disponibles para realizar la codificación en función de las necesidades propias de cada unidad básica, en función de su contenido o actividad. Este paso es previo al mecanismo de bit allocation y se puede contemplar como una parte del mismo.

  Es deseable que la magnitud contemplada para medir la complejidad sea lo más *universal* posible, esto es, que sea capaz de representar de manera fiel y rigurosa la complejidad de la unidad básica en diferentes condiciones de codificación, como pueden ser escenas muy diferentes, tasas objetivo muy distintas, patrones de GOP diferentes, etc. Del mismo modo, es deseable que sea una magnitud fácilmente calculable y de manejo sencillo, a fin de garantizar una baja carga computacional por parte de todo el mecanismo de control de tasa.

  Dependiendo de si el algoritmo contemplado es de una o más pasadas la medida de complejidad variará, sobre todo en la forma de calcularla.

- **Cálculo del presupuesto de bits o *Bit Allocation*:** Una vez calculadas las complejidades de los cuadros de la secuencia, es necesario determinar un presupuesto de bits para cada uno de dichos cuadros (o GOPs o unidades básicas, dependiendo del nivel al que se haga el control de tasa). Mediante estas técnicas se distribuirán los bits disponibles a lo largo de la secuencia, de manera que se cumplan los objetivos de tasa, estado de buffer, calidad, etc. impuestos.

Podemos considerar dos escenarios:

- Transmisión en redes: En este caso hay que tener en cuenta el estado del buffer para evitar situaciones de desbordamiento (*overflow*) y de vaciado (*underflow*). Además, hay que considerar la tasa instantánea (para ajustarnos a las capacidades de la red de tránsito) más que el volumen total de información de la secuencia. Es necesario un modelo de tráfico en el que puede haber parámetros variables en función del estado de la red.
- Almacenamiento: En este caso hay que considerar la tasa final, esto es, el volumen total que ocupa la secuencia (para que quepa en el espacio de almacenamiento asignado). Las políticas de buffer pueden ser más laxas, ya que el objetivo es almacenar más que transmitir.

El hecho de que el algoritmo sea de una o más pasadas tiene una importancia crucial, ya que en un esquema de dos o más pasadas se tiene conocimiento de *toda* la secuencia, de manera que se pueden tener en cuenta las complejidades de cuadros futuros a la hora de calcular el presupuesto del cuadro actual. De este modo, si en un futuro las complejidades son muy elevadas, el algoritmo lo tendría en cuenta asignando menos bits a los cuadros actuales, generando así una *reserva* de bits disponibles para dichos cuadros de mayor complejidad.

- **Mecanismo de asignación de QP o *Bit Rate Control*:** El último gran bloque presente en los algoritmos de rate control es el encargado de la selección del parámetro de cuantificación que satisfaga la condición impuesta por el bloque de bit allocation, esto es, escoger unas QPs de manera que la tasa de salida se ajuste al presupuesto de bits asignado para el GOP, cuadro o unidad básica sobre la que se esté trabajando. Para realizar esta tarea, el bit rate control debe, primero resolver el dilema del huevo y la gallina expuesto en el apartado 3.2 (pág. 56). Para ello, la técnica más habitual consiste en realizar una predicción de la distorsión (generalmente calculada como la MAD) a partir de los datos de cuadros o unidades básicas anteriores. Una vez resuelto este problema, se recurrirá a un modelo R-Q ó D-Q que relacione la tasa objetivo y/o la distorsión calculada con el parámetro de cuantificación, de manera que se pueda elegir este último satisfaciendo las restricciones de tasa y distorsión.

Del mismo modo, existen otros procedimientos auxiliares, que si bien no son imprescindibles para el codificador, sí mejoran su rendimiento. Estos procedimientos son:

- **Detección de cambios de escena:** Los cambios de escena constituyen uno de los sucesos más frecuentes en una secuencia de vídeo. Consisten en un cambio, generalmente brusco, en el contenido de las imágenes. Dentro de una escena o *shot*, las propiedades espaciales y temporales, así como las complejidades, se pueden considerar similares. Ejemplos de éstos pueden ser los cambios de encuadre o punto de vista de la cámara. La aparición de un cambio de escena en una secuencia puede ocurrir en cualquier momento y, por lo tanto, es muy frecuente que ocurra en medio de un GOP. Este suceso afecta muy de lleno en los algoritmos de rate control y en la calidad obtenida en una secuencia, ya que el primer cuadro del nuevo plano tendrá una grandísima proporción de macrobloques de tipo I, que son, con mucha diferencia, los que más bit rate requieren. Si el proceso de bit allocation se ha realizado a nivel de GOP sin tener en cuenta estos sucesos la calidad en ese GOP será

muy baja, ya que habría que codificar dos cuadros I[8], en vez de uno, quedando entonces una tasa disponible para los demás cuadros (P y B) claramente insuficiente para mantener el nivel de calidad. Si se dispone de un mecanismo de detección de estos cambios de escena se podría utilizar para detectarlos en una primera pasada y, posteriormente, adaptar el control de tasa para que tenga en cuenta estos sucesos, mediante cambios en la estructura de GOP, variaciones en las ventanas para la actualización de coeficientes, etc.

- **Selección de QP inicial:** Por QP inicial entendemos no sólo la QP del primer cuadro de la secuencia, sino también la QP del primer cuadro de cada escena, ya que no podemos suponer ninguna correlación entre los cuadros de dos escenas diferentes.
  En algoritmos de una pasada esta selección se realiza a ciegas, ya que no conocemos nada de la escena que comienza, por lo que es muy fácil cometer errores importantes. En algoritmos de dos pasadas, por el contrario, sí que conocemos la complejidad del cuadro y de la escena completa, por lo que se puede estimar esta QP de una forma más fiable. Sin embargo, en ambos casos esta selección no será ni mucho menos óptima, ya que los modelos R-Q empleados no están entrenados y sus coeficientes necesitan de varios cuadros para adaptarse a la escena.

Un esquema de las relaciones entre todos estos bloques se muestra en la figura 3.7.

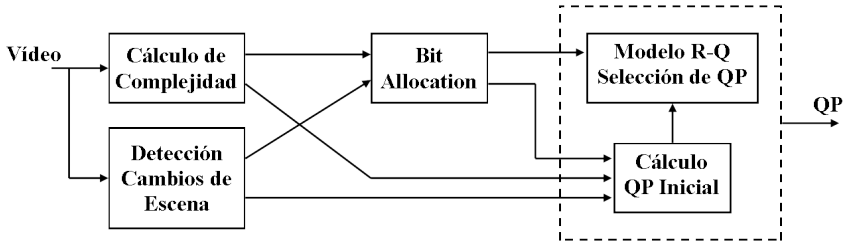


Figura 3.7: Bloques involucrados en el control del tasa

3.3.2. Algoritmos destacados

3.3.2.1. Algoritmo de Yokoyama

La propuesta de Yokoyama [17] está realizada en base a un codificador MPEG-2 de una pasada. Comienza con un método sencillo pero poco preciso de medida de complejidad, utilizado ampliamente en diferentes algoritmos de control de tasa. Este método es herencia de la denominada *Medida Global de Complejidad* del algoritmo TM5, mecanismo de control de tasa presente en MPEG-2.

Básicamente, Yokoyama calcula la complejidad como el producto del escalón de cuantificación por el ancho de banda consumido. En el algoritmo que propone utiliza dos medidas de comple-

[8] En realidad sólo se codificaría un cuadro de tipo I. Lo que ocurre es que en el cuadro de tipo P donde ocurre el cambio de escena habría una proporción enorme de macrobloques I, por lo que a efectos prácticos ese cuadro se podría decir que es de tipo I.

jidad, una representa la complejidad media de la parte de secuencia ya codificada, mientras que la otra representa la complejidad media del último GOP codificado:

$$X_{ave} = \overline{Q_{ave}} \times B_{ave}$$

$$X_{GOP} = \overline{Q_{GOP}} \times B_{GOP}$$

Este cálculo se realiza así porque el algoritmo es de una sola pasada, por lo que no tenemos información acerca de los cuadros futuros. El inconveniente es que estamos extrapolando el comportamiento de la secuencia en el pasado al comportamiento que vaya a tener en el futuro. Dicho de otro modo, estamos suponiendo una correlación temporal que no tiene por qué existir, por lo que esta medida, tal y como está definida no será adecuada en escenas muy cambiantes y en cambios de escena.

Con ellas y con el bit rate medio objetivo, R_{ave}, se calcula el escalón de cuantificación del GOP actual según la expresión

$$Q_{GOP} = \frac{\text{mín}(X_{ave}, X_{GOP})}{R_{ave}}$$

de manera que si la escena actual tiene una complejidad por encima de la media ($X_{GOP} > X_{ave}$) el parámetro de cuantificación asignado será menor y por lo tanto, el bit rate empleado aumentará. Finalmente, para ajustarse a los límites absolutos de tasa, considera dos parámetros de cuantificación adicionales, $Q_{MAXrate}$, correspondiente al escalón que genera el máximo bit rate admisible y Q_{min}, empleado para reducir el bit rate de una escena cuando éste es excesivo. Finalmente, el escalón determinado mediante este procedimiento es

$$Q_j = \text{máx}(Q_{GOP}, Q_{MAXrate}, Q_{min})$$

3.3.2.2. Algoritmo de Li

Podemos indicar que el algoritmo de una pasada presentado por Li, formalmente estándar JVT-G012 [18], constituye un mecanismo complejo y muy completo, puesto que considera múltiples aspectos del funcionamiento del codificador H.264, como puede ser la presencia del buffer, estudio de complejidades, posibilidad de bit allocation a dos o tres niveles (con la opción de utilizar el nivel de unidad básica), presencia o no de cuadros B, etc. Debido a ser el estándar vigente, este algoritmo es utilizado frecuentemente como referencia a la hora de probar otros algoritmos.

Este modelo opera a tres niveles: GOP, cuadro y unidad básica. Primeramente define un modelo de tráfico que muestra la dinámica del buffer para así poder asignar un número adecuado de bits en cada caso:

$$\begin{cases} B_c(n_{i,j+1}) = \text{mín}\left\{\text{máx}\left\{0, B_c(n_{i,j}) + A(n_{i,j}) - \frac{u(n_{i,j})}{F_r}\right\}, B_s\right\} \\ B_c(n_{1,0}) = \frac{B_s}{8} \\ B_c(n_{i+1,0}) = B_c(n_{i,N_{GOP}}) \end{cases}$$

donde

N_{GOP} es el número total de cuadros en un GOP
$n_{i,j}$ $(i = 1, 2, \ldots; j = 1, 2, \ldots, N_{GOP})$ es el cuadro j del GOP i
$B_c(n_{i,j})$ es la ocupación del buffer virtual después de codificar el cuadro $n_{i,j}$
B_s es el tamaño del buffer. Notar que inicialmente la ocupación del buffer es $B_s/8$ para evitar situaciones de *underflow*
$A(n_{i,j})$ es el número de bits generados por el cuadro j del GOP i
$u(n_{i,j})$ es el ancho de banda disponible en el canal (puede ser constante o variable en el tiempo)
F_r es la tasa de cuadros o *frame rate*

Una vez definido el modelo de tráfico, el algoritmo comienza por asignar bits a un GOP. Cada GOP tiene que ceñirse a su presupuesto, teniendo que haberlo gastado todo al terminar. En este cálculo se tiene en cuenta que el ancho de banda del canal puede variar.

Posteriormente se calcula el presupuesto para cada cuadro utilizando un control *macroscópico* en el que se determina el nivel objetivo que tiene que tener el buffer, *Tbl, (Targel Buffer Level)*, seguido de un control *microscópico* en el que se refina el cálculo del presupuesto, quedando el número de bits asignados a cada cuadro según la siguiente expresión:

$$f(n_{i,j}) = \beta \cdot \hat{f}(n_{i,j}) + (1 - \beta) \cdot \bar{f}(n_{i,j})$$

donde

β es una constante con valor $\beta = \begin{cases} 0{,}5 & \text{, si no hay cuadros B} \\ 0{,}9 & \text{, otro caso} \end{cases}$
$\hat{f}(n_{i,j})$ es el número de bits pendientes de gastar en el GOP
$\bar{f}(n_{i,j})$ se corresponde con la expresión

$$\hat{f}(n_{i,j}) = \frac{u(n_{i,j})}{F_r} + \gamma \cdot (Tbl(n_{i,j}) - B_c(n_{i,j}))$$

donde γ es una constante con valor $\gamma = \begin{cases} 0{,}75 & \text{, si no hay cuadros B} \\ 0{,}25 & \text{, otro caso} \end{cases}$

Adicionalmente puede haber un nivel de unidad básica (en caso de que la unidad básica seleccionada no fuera un cuadro). En éste, el bit allocation sólo se realizaría para cuadros P, ya que todas las unidades básicas de un cuadro I o B se codifican con una misma QP[9]. En este nivel se calcula la información de textura que va a tener cada unidad básica haciendo un reparto uniforme de los bits disponibles entre las unidades básicas del cuadro pendientes de codificar. A este nivel, el consumo de bits por parte de las cabeceras es muy significativo frente a la información de texturas, por lo que se le dedica una atención especial al cálculo las cabeceras. Finalmente, el número de bits destinados para información de textura será

$$R_{t,l} = \frac{f_{rb}}{N_{ub}} - m_{hdr}$$

[9]La QP de los cuadros I se calcula en el nivel de GOP, mientras que la de los cuadros B se calcula en el nivel de cuadro

Una vez calculados los presupuestos para cada nivel se procederá a calcular la QP. Al igual que en el bit allocation, se tendrá en cuenta el tipo de cuadro y se realizará a distintos niveles: GOP, cuadro y, si se utiliza, unidad básica.

Primeramente, a nivel GOP se determina la QP del cuadro I y del primer cuadro P de cada GOP. Además, en este nivel se determina la QP inicial de la secuencia, QP_0, a partir de la resolución de cada cuadro y de la medida de los bits por pixel. Para los sucesivos GOPs (salvo el primero de la secuencia), la QP se calcula mediante la expresión

$$QP_{st} = \frac{Sum_{PQP}}{N_P} - \frac{8 \cdot T_r(n_{i-1,N_{GOP}})}{T_r(n_{i,0})} - \text{mín}\left\{\frac{N_{GOP}}{15}, 2\right\}$$

en la que el primer término es la media de las QPs de todos los cuadros P del GOP anterior, el segundo término permite adaptarse al ancho de banda disponible en el canal y el tercer término es un factor de corrección en función del tamaño del GOP (mayor o menor de 15 cuadros).

A continuación se opera a nivel de cuadro, donde se determinarán las QPs de los cuadros P (salvo el primero del GOP) y B.

El parámetro de cuantificación de los cuadros B se calcula mediante interpolación lineal de los QPs de los dos cuadros P vecinos, QP_1 y QP_2. En cualquier caso, para que la calidad sea lo más uniforme posible, la variación de QP entre dos cuadros vecinos no deberá ser mayor de 2 unidades y el valor final obtenido tiene que estar dentro de los admitidos por H.264 (entre 1 y 51).

En el caso de los cuadros P, el parámetro de cuantificación se calcula tras el bit allocation mediante un modelo cuadrático:

$$R = \frac{C_1 \times \widehat{MAD}}{\widehat{QP}_C} + \frac{C_2 \times \widehat{MAD}}{\widehat{QP}_C^2} + \hat{h}$$

donde

R es la tasa objetivo calculada en el proceso de bit allocation, $f(n_{i,j})$

$\widehat{QP}_C$ es el parámetro de cuantificación que estamos calculando

C_1 **y** C_2 son parámetros del modelo. Son actualizados cada vez que se codifica un cuadro mediante el esquema propuesto en [19, 20]

$\widehat{MAD}$ es la predicción de la MAD del cuadro codificado. Como se mencionó al inicio de la sección, el modelo de optimización rate-distortion con función de coste de Lagrange utilizado en H.264 presenta un problema de interdependencia entre el proceso RDO, y el cálculo de QP y MAD. Para solucionar este *dilema del huevo y la gallina* se ha optado por predecir el valor de la MAD a partir del valor real obtenido en la codificación del cuadro anterior, mediante la expresión

$$\widehat{MAD}_{actual} = a_1 \cdot MAD_{anterior} + a_2$$

donde a_1 y a_2 son parámetros del modelo. Son actualizados mediante regresión lineal.

Una vez determinado el valor de $\widehat{QP}_C$ correspondiente a la tasa objetivo sólo queda hacerle un par de restricciones, que son limitar su variación entre cuadros consecutivos a no más de ± 2

unidades

$$\widetilde{QP}_C = \text{mín}\left\{QP_P + 2, \text{máx}\left\{QP_P - 2, \widehat{QP}_C\right\}\right\}$$

y limitar su rango de valores al admitido por H.264

$$QP_C = \text{mín}\left\{\text{máx}\left\{\widetilde{QP}_C, 1\right\}, 51\right\}$$

Finalmente, en un hipotético nivel de unidad básica[10] se realizará una predicción lineal de la MAD de la unidad básica, como se ha descrito anteriormente, y a continuación se utilizará el modelo cuadrático aplicado a esa unidad básica, seguido todo ello de un procedimiento para reducir artefactos y de un control de variación (para evitar saltos bruscos de calidad) y de un *clipping* o limitación del valor de QP a los límites impuestos por el estándar (entre 0 y 51).

Mejora del algoritmo de Li: Técnica de Shuguang

Esta técnica [21], supone un refinamiento a este esquema de Li, ya que el autor considera dos debilidades en el procedimiento realizado por Li en las fases de predicción de la MAD y en el bit allocation a nivel de unidad básica:

- En la asignación de bits a una unidad básica se utiliza una expresión de la forma $R_i = F_{rb}/N_{ub}$. Sin embargo, este método no considera que a las unidades básicas con mayor MAD se les debería asignar un mayor número de bits, en lugar de un reparto uniforme de los bits disponibles entre las unidades básicas pendientes de ser procesadas.
- El mecanismo de predicción lineal de la MAD de la unidad básica empleado ($\widehat{MAD}_{actual} = a_1 \cdot MAD_{anterior} + a_2$), en el que $MAD_{anterior}$ es la MAD real de la unidad básica ubicada en el mismo lugar, pero en el cuadro anterior) no es adecuado para situaciones como cambios de plano o movimientos rápidos.

Para solucionar estas debilidades, el autor propone unas modificaciones en el cálculo de los bits a asignar a cada unidad básica y de la predicción de la MAD.

Para ajustar la predicción de la MAD se introduce un parámetro η, denominado *precisión de la predicción*, de manera que la predicción corregida de la MAD será

$$\widehat{MAD}_i^{corr} = \widehat{MAD}_{actual} \cdot \frac{1}{1 + \eta_{i-1}}$$

De modo similar se corrige la asignación de bits a cada unidad básica. En este caso, para cada unidad básica i se definirá un parámetro μ_i (utilizando $\widehat{MAD}_i^{corr}$), denominado *complejidad relativa*, de manera que la asignación de bits a la unidad básica i queda modificada:

$$R_i^{corr} = R_i \cdot (1 + \mu_i)$$

[10] El nivel de Unidad Básica no siempre está presente

3.3.2.3. Algoritmo de Que

Que [22] proporciona un algoritmo de control de tasa VBR para H.264 en dos pasadas. La primera pasada consiste en una codificación con un algoritmo de control de tasa CBR T264. En ésta se recogen las estadísticas de la secuencia a codificar, como los bits generados, la distribución de macrobloques Intra y la QP empleada en cada cuadro.

Al inicio de la segunda pasada se calcula la complejidad de cada cuadro mediante el esquema de *Medida Global de Complejidad*, igual que el empleado en TM5 y utilizado por Yokoyama (ap. 3.3.2.1), solo que esta vez en un algoritmo de dos pasadas:

$$C_j = B_j \times Q_{step_j}$$

donde

C_j es la complejidad del cuadro j
B_j es la cantidad de bits generada por el cuadro j en la primera pasada
Q_{step_j} es el escalón de cuantificación empleado en la primera pasada en el cuadro j

Una vez calculadas todas las complejidades, el reparto de bits se realizará de manera proporcional a éstas. Según este mecanismo, el número de bits asignados al cuadro n en la segunda pasada es

$$B_{2,n} = B_{2,total} \cdot \frac{C_n}{C_{total}}$$

Después de este cálculo, realiza una detección de cambios de escena. Ésta se realizará mediante un mecanismo sencillo en el que el criterio de detección es el procentaje de macrobloques Intra presentes en los cuadros P, considerando el hecho de que existen modos Intra 4×4 y 16×16. Si este porcentaje es lo suficientemente elevado se decidirá que ha ocurrido un cambio de escena. Como lo más probable es que estos cambios ocurran dentro de un GOP, el cuadro P correspondiente pasará a ser de tipo I, dividiéndose el GOP en dos partes: La primera parte se fusionará con el GOP anterior, mientras que la segunda parte se fusionará con el GOP siguiente. En estos casos, el cuadro I del GOP siguiente se cambiará a P, como se puede ver en la figura 3.8.

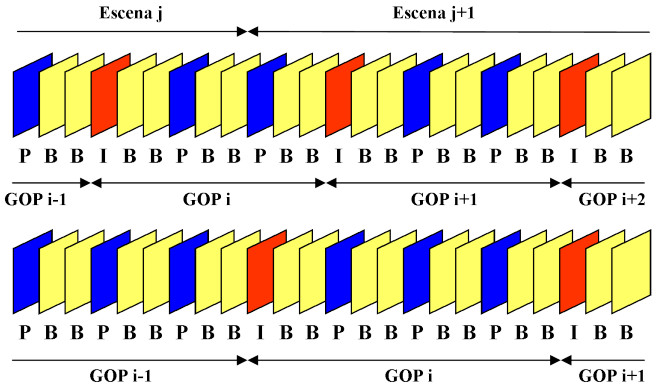


Figura 3.8: Algoritmo de Que: Recomposición de los GOPs en un cambio de escena

Ya que las diferentes escenas pueden tener comportamientos muy diferentes, posteriormente se refina estas asignaciones mediante el cálculo de un factor de PSNR para cada escena, ω_m, que refleja las variaciones de calidad respecto a la PSNR media:

$$\omega_m = 1 - \frac{\lambda \cdot (PSNR_{ave,m} - PSNR_{ave})}{PSNR_{ave}}$$

A continuación realiza dos pasadas por todas las escenas y utiliza el factor ω_m calculado anteriormente para realizar correcciones en las asignaciones de bits para cada cuadro. Finalmente calcula un parámetro $f_{n,i}$ para cada cuadro en base a una pequeña banda de guarda ($\mathrm{BF_{guard}}$) para evitar situaciones de *underflow*. Finalmente, nos quedamos con el mínimo de $f_{n,i}$ de cada GOP, f_n, de manera que el presupuesto de cada cuadro queda

$$B_{2,n,i} = \begin{cases} B_{2,n,i} \cdot f_n & ,\mathrm{f_n} < 1 \\ B_{2,n,i} & ,\text{otro caso} \end{cases}$$

Con este reparto de bits y la complejidad obtenida previamente, el escalón de cuantificación, $Qstep_{2,n}$ se calculará en dos pasos:

$$Qstep_{2,n} = \frac{C_n}{B_{2,n}}$$

$$Qstep_{2,n} = \frac{Qstep_{2,n}}{1 - \beta \cdot \delta_n}$$

donde

β es un factor de escala, definido como $\beta = \dfrac{1}{2 \cdot BF_{max}}$ (BF_{max} es el tamaño del buffer virtual).

δ_n se define como $\delta_n = \begin{cases} 0 & ,\text{si n} = 0 \\ \delta_{n-1} + B\_real_{2,n-1} - B_{2,n-1} & ,\text{si n} > 0 \end{cases}$

$B\_real_{2,n-1}$ es el número real de bits generados por el cuadro n-1 en la segunda pasada.

Finalmente se realiza la conversión de escalón de cuantificación a parámetro de cuantificación, la limitación de variación de QP y de su rango de valores a los admitidos por H.264:

$$QP_{2,n} = \mathrm{round}\,(6 \cdot \log_2 Qstep_{2,n} + 4)$$

$$QP_{2,n} = \mathrm{mín}\,\{QP_{2,n-1} + 2, \mathrm{máx}\,\{QP_{2,n-1} - 2, QP_{2,n}\}\}$$

$$QP_{2,n} = \mathrm{mín}\,\{51, \mathrm{máx}\,\{QP_{2,n}, 1\}\}$$

3.3.2.4. Algoritmo de Zhang

El algoritmo presentado por Zhang [23, 24] consiste en un método de dos pasadas para el codec H.264. Al igual que en casos anteriores, la primera pasada consiste en una codificación completa para recabar las estadísticas de la secuencia. Esta codificación se realizará usando una QP constante, cuyo valor se recomienda que esté comprendido entre 16 y 30. Los demás parámetros de codificación, como pueden ser el tamaño del GOP y el número de cuadros de

referencia, deben ser los mismos que los que se usarán posteriormente para la segunda pasada. Las estadísticas recogidas en esta fase serán el bit rate y la distorsión de cada cuadro, siendo esta última medida como el MSE[11]. En este caso, el modelado de complejidad es

$$C_i = D_i \cdot R_i$$

donde

C_i es la complejidad del cuadro i
D_i es la distorsión MSE del cuadro i en la primera pasada
R_i es el bit rate del cuadro i obtenido en la primera pasada

Tras esta primera pasada se realiza una detección de cambios de escena. En ella se utiliza como criterio la complejidad medida en la primera pasada. Al haber tres tipos de cuadros diferentes, esta complejidad fluctúa significativamente ($C_I > C_P > C_B$), pero dentro de una misma escena y de un mismo tipo de cuadro los valores son similares, como se puede ver en la figura 3.9

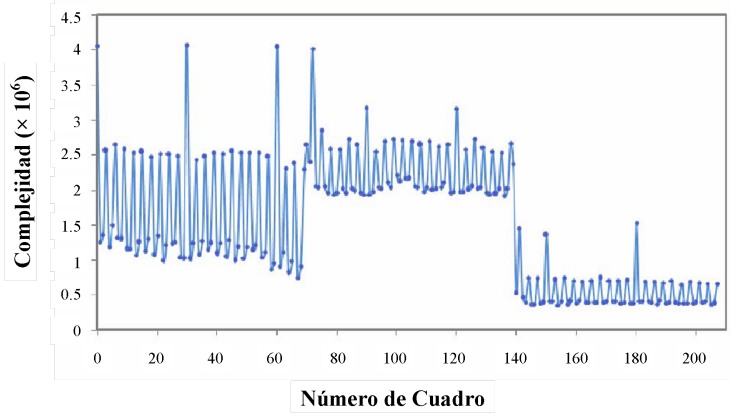


Figura 3.9: Variaciones de complejidad en una secuencia

El autor compara las diferencias de complejidad entre dos cuadros consecutivos del mismo tipo. Si la diferencia es mayor que un umbral se determina que ha habido un cambio de escena. En este caso se reinicia la ventana utilizada para actualizar el modelo D-Q utilizado para adaptarse a la nueva escena. Cabe destacar que no se consideran relevantes para el propósito del rate control los cambios ocurridos en cuadros I.

A continuación, ya en la segunda pasada, el bit allocation se realiza a nivel de GOP, dejando el nivel de cuadro para el mecanismo de selección de QP o *bit rate control*.

El reparto de bits se realizará calculando la desviación de la complejidad total del GOP actual respecto a la complejidad media de todos los GOPs de la secuencia. Además, se introduce un término de corrección que se encarga de compensar el hecho de que una vez codificado un GOP se hayan gastado bits de más o de menos:

$$T_j = \frac{C_j^{GOP}}{C_{aver}^{GOP}} \cdot T_a^j - \sum_{k=0}^{j-1} \frac{B_k}{M-k-1}$$

[11] MSE: *Mean Square Error*, Error Cuadrático Medio. Definición: $\mathrm{MSE}(\hat{X}) = E\left\{(X-\hat{X})^2\right\}$

donde

C_j^{GOP} es la complejidad total del j-ésimo GOP (el actual)
C_{aver}^{GOP} es la complejidad media de todos los GOPs de la secuencia
T_a^j es el número medio de bits objetivo de cada GOP aplicando asignación uniforme (misma cantidad de bits para todos los GOPs)
M es el número total del GOPs que hay en la secuencia

La segunda parte de la expresión se encarga de mostrar el número total de bits que hay que compensar en el j-ésimo GOP. En ella, el parámetro B_k es la diferencia entre los bits realmente generados al codificar el GOP k-ésimo y los bits que tenía asignados. Podemos ver que este exceso o defecto de bits no se carga al GOP siguiente, sino que se reparte equitativamente entre todos los GOPs restantes.

Finalmente, si tomamos como referencia los bits realmente generados por el (j-1)-ésimo GOP, T_{j-1}^c, la asignación de bits al j-ésimo GOP puede definirse como

$$\tilde{T_j} = (1-\alpha) \cdot \frac{C_j^{GOP}}{C_{j-1}^{GOP}} \cdot \frac{N_j - N_{j-1}}{N_{j-1} - N_{j-2}} \cdot T_{j-1}^c + \alpha \cdot T_j$$

donde

α es una constante cuyo valor típico es 0.8
N_j esel número de cuadros procesados después de codificar el j-ésimo GOP

Para el cálculo del parámetro de cuantificación a nivel de cuadro realizaremos una estimación de la distorsión (MSE) que se obtendrá en la segunda pasada a partir de la distorsión MSE medida en la primera pasada:

$$\sum_{i=N_{j-1}}^{N_j-1} R_i^{2^a} \cdot D_i^{2^a} = k_j \cdot \sum_{i=N_{j-1}}^{N_j-1} R_i^{1^a} \cdot D_i^{1^a} \quad , \quad j = 0, \ldots, M-1$$

donde

$R_i^{2^a}$ **y** $D_i^{2^a}$ son la tasa y distorsión del cuadro i esperados en la segunda pasada
$R_i^{1^a}$ **y** $D_i^{1^a}$ son la tasa y distorsión del cuadro i obtenidos en la primera pasada
N_j **y** N_{j-1} son el número de cuadros codificados después de codificar los GOPs j-ésimo y (j-1)-ésimo, respectivamente
k_j es un factor de escala para ajustar las diferencias que pudieran haber entre la primera y la segunda pasada

Si suponemos que la distorsión D_c^j es aproximadamente constante entre todos los cuadros del GOP j-ésimo, podemos decir que la distorsión media de un cuadro en este GOP queda:

$$D_c^j = \frac{k_j \cdot \sum_{i=N_{j-1}}^{N_j-1} R_i^{1^a} \cdot D_i^{1^a}}{\tilde{T_j}}$$

Posteriormente se hace un ajuste para evitar fuertes variaciones de la dsitorsión entre GOPs:

$$\tilde{D}_c^j = \beta \cdot D_c^j + (1-\beta) \cdot D_c^{j-1}$$

donde β es una constante de valor típico 0.6.

Una vez obtenido el valor de la distorsión esperada en un cuadro determinaremos el parámetro de cuantificación que deberá se usado para codificarlo en la segunda pasada. Para ello se empleará un modelo D-Q que nos relacione la QP con la distorsión. Este modelo podría ser el modelo cuadrático habitual. Sin embargo, como se muestra en [25], puede ser más interesante utilizar un modelo lineal, ya que, aunque el cuadrático puede ajustarse mejor, el lineal tiene la ventaja de que sus parámetros pueden ser ajustados más fácilmente y de manera más precisa, pudiendo así proporcionar mejores prestaciones que el modelo cuadrático. Es lo que se ha hecho en esta propuesta. El modelo D-Q lineal para calcular la QP de un cuadro P o B del GOP j-ésimo es

$$D_F = \tilde{D}_c^j = X_F \cdot QP_F + Y_F \quad , \quad F \in (P, B)$$

donde X_P/X_B e Y_P/Y_B son parámetros del modelo, que son actualizados tras codificar un cuadro P o B.

Finalmente se limita la variación del QP resultante a ± 2 unidades respecto de la QP del anterior cuadro del mismo tipo (P/B) que el actual.

Tratamiento especial merecen los cuadros I, ya que entre ellos hay poca correlación, debido a que están demasiado separados entre sí, para que el modelo de distorsión empleado en los cuadros P y B sea válido. Además de éstos, la situación es similar para el primer cuadro P y B de un plano, ya que no habrá referencias anteriores de otros cuadros P y B. Para estos casos se utiliza una relación obtenida experimentalmente que nos permitirá obtener el QP a emplear a partir de los datos obtenidos en la primera pasada: la distorsión predicha y la distorsión esperada:

$$QP^{2^a} = \left(\frac{\tilde{D}_c^j}{D^{1^a}} \right)^{\frac{1}{4}} \cdot (QP^{1^a} + \gamma)$$

donde γ es una constante, cuyo valor se fijará experimentalmente a 1 ó 2, salvo el caso en que el bit rate de la primera pasada sea similar al esperado en la segunda, que valdrá 0.

3.3.2.5. Otros algoritmos de control de tasa destacados

Aparte de los algoritmos comentados anteriormente, existen otros algoritmos que merecen ser destacados por su enfoque diferente. Entre ellos están los propuestos por Huang [26], De Vito [27] y el algoritmo de Shi [28].

El algoritmo de Huang es de dos pasadas y emplea como medida de distorsión directamente la PSNR. En función de un análisis de la distribución de los coeficientes de la transformada realiza un modelo R-D heurístico y un modelo PSNR-Q sencillo:

$$PSNR(R) = \alpha R + A - \frac{A-B}{1+bR}$$

$$PSNR(Q_{step}) = c \cdot \log_{10} Q_{step} + d$$

En la primera pasada extrae características, que procesará *offline* para ajustar los dos modelos anteriores. La asignación de bits y el cálculo de QP se realiza mediante un proceso iterativo.

El algoritmo de De Vito destaca por su finalidad, ya que más que perseguir una estabilidad en la calidad o un control en el bit rate generado, control en el estado de los buffers, etc., lo que implementa es un mecanismo de control de la calidad del vídeo. De este modo, ofrece dos posibilidades: obtener una calidad cuasi-constante (en cuanto a PSNR) u obtener una calidad fluctuante según un patrón establecido, de manera que podamos determinar diferentes niveles de PSNR para planos distintos (que podremos emplear para diferenciar secuencias de acción, secuencias con poca actividad, anuncios, etc.).

Aunque el autor indica que se trata de un mecanismo de una pasada, en realidad podemos asumir que se trata de un algoritmo de dos pasadas, ya que parte de un conocimiento previo de la secuencia a codificar. Para controlar las variaciones de PSNR, el algoritmo hace uso de una ventana deslizante de N cuadros, de manera que toma las decisiones sobre la QP en función de la historia de la PSNR en cuadros pasados. De este modo, si la PSNR media de los N cuadros anteriores al que se va a codificar (los pertenecientes a la ventana) no difiere mucho de la PSNR objetivo, se seguirá empleando la misma QP. Si, por el contrario, esta diferencia supera un umbral establecido, entonces la QP se modificará para tratar de reducirla:

$$\left| \frac{1}{N} \cdot \sum_{i=-1}^{-N} PSNR_{j-i} - PSNR_{objetivo} \right| < \Delta \quad \Rightarrow \quad QP_j = QP_{j-1}$$

$$\left| \frac{1}{N} \cdot \sum_{i=-1}^{-N} PSNR_{j-i} - PSNR_{objetivo} \right| > \Delta \quad \Rightarrow \quad QP_j = QP_{j-1} + \delta$$

donde

j es el índice del siguiente cuadro a codificar
Δ es la variación máxima admisible
δ es la cantidad de variación de QP, definida como

$$\delta = \pm \text{mín} \left\{ \left[\gamma \cdot \left| \frac{1}{N} \cdot \sum_{i=-1}^{-N} PSNR_{j-i} - PSNR_{objetivo} \right| \right], K \right\}$$

en la que

$[x]$ indica la parte entera de x
K es un número entero positivo que limita la máxima variación de QP
γ modela la intensidad de la dependencia entre QP y PSNR

el signo de δ es el mismo que el que tiene la diferencia entre la PSNR media y el objetivo

Este algoritmo, al no contemplar ningún mecanismo de bit allocation ni de control del bit rate, es, según la clasificación hecha en el apartado 3.1.4.2.1, un claro ejemplo de control de tasa de tipo U-VBR.

El algoritmo de Shi pertenece a una categoría especial de algoritmos de control de tasa, denominada *algoritmos de rate control basados en regiones de interés*. Estos algoritmos son un refinamiento de los algoritmos tradicionales de rate control. En ellos, el objetivo no es maximizar la PSNR de todo el cuadro, sino sólo la de las regiones de interés para un observador. Según los estudios, un observador tiende a fijarse más en la zona central de la imagen y en las zonas con movimiento, por lo que es razonable detectar estas zonas para asignarles un bit rate más elevado y así una mayor calidad, mientras que al resto se le asigna un bit rate más bajo (con la correspondiente degradación de la calidad en esas zonas).

Hay que dejar claro que en este tipo de métodos no hay procesos de segmentación. No se reconocen, identifican ni separan ningún elemento de la imagen, por lo que no tiene sentido hablar de objetos de vídeo como en MPEG-4 [4]. Lo que se hace es detectar regiones que destaquen por su movimiento y/o posición, y estas regiones no tienen por qué corresponderse con un objeto de manera exacta, y ni siquiera con el objeto entero. Esto se puede ver fácilmente en la figura 3.10.

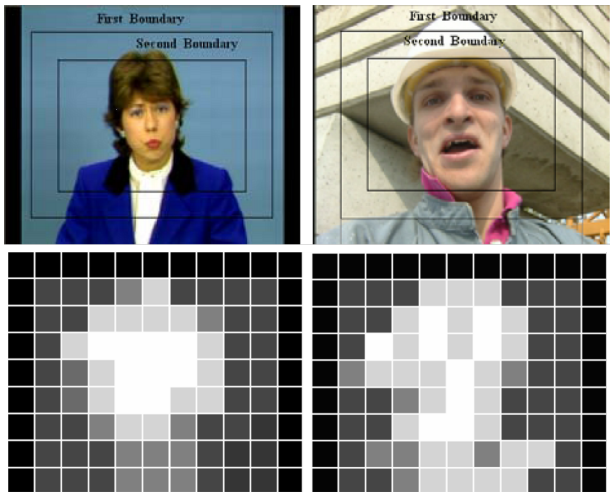


Figura 3.10: Regiones de interés en secuencias de vídeo

Este algoritmo clasifica los macrobloques de un cuadro en cinco categorías, según su importancia. Esta importancia se mide como un peso calculado como el cociente entre la MAD del macrobloque actual y la media de las MADs de todos los macrobloques del cuadro. Una vez clasificados los macrobloques, se le aplica a cada uno el control de tasa ya conocido basado en un modelo cuadrático y en la predicción lineal de la MAD. Finalmente, se ajusta la QP obtenida para cada macrobloque considerando su categoría.

Además de estos últimos algoritmos, se han formulado propuestas realmente *exóticas*. Entre éstas podemos destacar un rate control a nivel de unidad básica cuyo tamaño varía según su actividad [29] o un algoritmo en el que lo que se modifica es el multiplicador de Lagrange, λ, de manera que la tasa se controla variando el mecanismo de RDO, en lugar de la QP [9].

3.3.3. Técnicas complementarias al Control de Tasa

Sin salirnos del contexto del control de tasa, existen una serie de técnicas adicionales que, si bien no pertenecen a ningún algoritmo de control de tasa en concreto, sí pueden mejorar el rendimiento de todos los algoritmos (o un número importante de ellos).

3.3.3.1. Técnicas de detección de cambios de escena

El método planteado por Zeng [30] actúa de manera diferente para cada tipo de cuadro (I, P o B). Para los cuadros I se utilizará la diferencia de histogramas de modos Intra, basándose en la premisa de que contenidos similares se corresponden con modos de predicción Intra similares. Si esta diferencia es mayor que un umbral, se considerará que ha habido un cambio de escena. Para evitar falsas alarmas se requiere adicionalmente que esta diferencia medida en el cuadro I actual sea mayor que la diferencia en el cuadro I anterior y en el siguiente. Para los cuadros P se utiliza como criterio el número de macrobloques Intra presentes. Si este número es mayor que un umbral y sus vecinos tienen pocos o ningún macrobloque Intra se considera que ha habido cambio de escena. Con los cuadros B lo que se hace es ver en qué dirección (pasada o futura) apuntan sus vectores de referencia (si toma como referencia cuadros pasados, futuros o uno pasado y otro futuro). De forma gráfica, estos tres casos son los que aparecen en la figura 3.11. Para evitar falsas alarmas, se aplica un criterio adicional: Si dos cuadros B tienen un número de macrobloques Intra superior a un umbral determinado, la situación se considerará una falsa alarma.

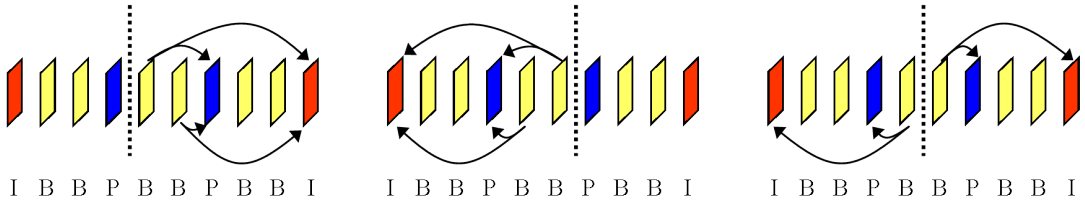


Figura 3.11: Detección de cambios de escena en cuadros B según Zeng

En otra propuesta, Dimou [31] plantea un algoritmo útil para el caso de aplicaciones en tiempo real. Utiliza como magnitud a explorar la Suma de Diferencias Absolutas, o SAD[12]. Esta SAD presenta picos en los cambios de escena, pero tambien pueden aparecer picos sin haber un cambio de escena debido a múltiples factores (movimientos rápidos, cambios repentinos de iluminación, transiciones rápidas, etc.). Para evitar esto, el autor presenta un algoritmo en el que el umbral con el que se compara la SAD es móvil. Para ello caracteriza la SAD en una ventana (intervalo [n-(N+1), n-1]), en forma de su media (m(n)) y desviación típica (σ(n)), de manera que el umbral en el instante actual, T(n), será

$$T(n) = a \cdot (SAD(n-1) - m(n)) + (b+a) \cdot m(n) + c \cdot \sigma(n)$$

a, b, c son constantes que regularán el comportamiento dinámico del umbral. Tras detectar un

[12] $SAD(n) = \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} |F_n(i,j) - F_{n-1}(i,j)|$, en la que F_n es el n-ésimo cuadro de tamaño N×M

cambio de escena, el umbral pasa a valer la SAD en ese cuadro y se le hace decaer en cuadros sucesivos para evitar falsas detecciones en los cuadros siguientes.

3.3.3.2. Técnicas de selección de la QP inicial

La selección de la QP inicial con la que comenzar la codificación de una secuencia de vídeo constituye un primer dilema que afrontar, ya que ésta es especialmente crítica con la calidad obtenida en sucesivos cuadros. El hecho de que en H.264 la variación de QP esté limitada (por defecto) a ± 2 unidades hace que el error cometido tarde varios cuadros en ser corregido, en converger a un valor de QP adecuado.

Existen varias propuestas para este problema. Una muy simple y extendida es la que fija unos valores para QP_0 en función del número de bits por pixel ($bpp = \dfrac{T_{bitrate}}{F_r \times N_{pixel}}$, donde $T_{bitrate}$ es el bit rate objetivo), agrupados en intervalos.

Este tipo de soluciones proporcionan resultados muy poco satisfactorios, ya que no tienen en cuenta el contenido de la imagen. Pueden asignar una misma QP a dos cuadros realmente diferentes y con complejidades muy distintas siempre que tengan la misma resolución y la misma tasa objetivo. Además, comparando con la curva R-QP real, el error cometido puede ser muy grande. A esto le tenemos que añadir la propiedad de monotonicidad, expuesta en [32], que dice que el coste R-D de un cuadro *Inter* crece monotónicamente con la QP de sus imágenes de referencia, incluso si se utiliza la misma QP para codificar la imagen actual y la de referencia. Esto viene a decir que los errores de cuantificación en las imágenes de referencia se propagan. Debido a esta propiedad, lo deseable s que $\mathrm{QP_I} < \mathrm{QP_P} < \mathrm{QP_B}$. Si inicialmente la $\mathrm{QP_I}$ está calculada con errores, esta relación entre las QPs puede no cumplirse, extendiéndose el error por un número importante de cuadros.

Debido a estos problemas, se han investigado nuevos métodos para calcular este parámetro QP_0.

En el método de Kwon [33] se parte de la base de que el bit rate es inversamente proporcional a la QP, de manera que cuando uno crece el otro decrece en la misma proporción. Basándonos en esto, podemos emplear un método de dos pasadas para este cuadro, de manera que en la primera pasada obtengamos los bits generados por ese cuadro, E_0, habiendo utilizado el esquema anterior para elegir QP_0. Si este E_0 resulta demasiado elevado, la QP óptima deberá ser mayor, mientras que si E_0 es demasiado baja, QP_0 deberá tener un valor mayor.

Podemos hacer una estimación de la QP_0 óptima, QP'_0, a partir de E_0 y de una relación lineal:

$$QP'_0 = \text{máx}\left\{1, \text{mín}\left\{(\alpha \times E_0 + \beta), 51\right\}\right\}$$

donde α y β son parámetros del modelo que dependen de las condiciones de la codificación, por lo que se obtienen experimentalmente. Estos parámetros cumplen que a menor resolución de la imagen, mayor pendiente (α) tiene la expresión lineal.

Mucho más sofisticada es la propuesta de Czúni [34]. En ella, se parte de un conocimiento previo de la secuencia para obtener la función R-QP necesaria con la que poder determinar la QP inicial de forma inmediata con sólo introducir la tasa objetivo para ese cuadro. Para calcular la función se utiliza una red neuronal con una capa oculta con seis neuronas, determinada experimentalmente y entrenada mediante el algoritmo de retropropagación. El modelo empleado para las curvas R-QP será logarítmico, de la forma $QP = a \cdot \ln(R) + b$, donde a y b se ajustan para minimizar el error cuadrático del modelo. Una vez determinada y entrenada esta red neuronal con un conjunto de muestras de entrenamiento, se podrá utilizar para estimar la curva R-QP de otras secuencias desconocidas para las que querramos obtener una QP inicial, QP_0, adecuadamente.

Capítulo 4

Algoritmo de control de tasa propuesto

4.1. Escenario

En este Proyecto Fin de Carrera se propone un algoritmo de Control de Tasa cuyo objetivo es la consecución de una secuencia de vídeo con la mayor calidad percibida posible, tomando como criterio su constancia, es decir, la secuencia comprimida generada debe tener la calidad lo más estable posible. Cabe decir que no se considera necesario que la calidad sea constante a lo largo de *toda* la secuencia; basta con que esta calidad varíe despacio, de manera suave, a lo largo de una escena[1].

El escenario va a ser el de una aplicación *offline*, en la que el tiempo de codificación no supone una restricción. Además, se supone que la red de transporte va a tener capacidad suficiente para poder cursar toda la información generada por el codificador, por lo que no se considera necesaria la presencia de un buffer a la salida, ya que no será necesario suavizar la variación de la tasa de salida del codificador para introducirla en la red. Por lo tanto, se trata de un Rate Control U-VBR (según la clasificación del apartado 3.1.4.2.1, pág. 53). Este escenario podría ser el almacenamiento de una secuencia de vídeo en un medio, como puede ser un disco duro o un disco óptico, como Blu-Ray. Ya que la finalidad de este algoritmo es que las variaciones de calidad a corto plazo sean suaves, para poder absorberlas se permitirán variaciones *locales* de tasa, por lo que se considerará *localmente VBR*. Por otro lado, aunque el algoritmo sea VBR, el volumen total de información generada debe ajustarse firmemente al volumen objetivo. La motivación de esto es obvia: si nos pasamos de volumen, la secuencia no cabrá en el medio de almacenamiento, mientras que si nos quedamos cortos, generaremos menos información, reduciendo por lo tanto la calidad final de la secuencia.

Para ajustarnos a esto, proponemos un algoritmo de Control de Tasa de dos pasadas, VBR

[1] Considerando *escena* como el conjunto de fotogramas o cuadros en los que el contenido y la actividad tienen una variación leve o incluso nula. En el mundo cinematográfico se conoce como *plano*, y constituye la unidad narrativa más pequeña pero significativa del hecho audiovisual.

localmente y con detección de cambios de escena, con el objetivo de obtener un flujo comprimido de salida que, una vez reconstruido, proporcione una secuencia con calidad percibida lo más constante posible.

4.2. Descripción del algoritmo

Para poder realizar un reparto adecuado y eficiente de los bits disponibles es imprescindible conocer las características de la secuencia de vídeo completa. Para ello, el algoritmo se compondrá de dos pasadas, esto es, se realizarán dos codificaciones. La primera de ellas servirá para analizar el contenido de la secuencia, las complejidades de cada cuadro, detectar los cambios de escena y obtener el número total de escenas de la secuencia. En la segunda pasada se realizará la codificación definitiva, tomando los datos generados en la primera pasada, procesándolos y haciendo correcciones conforme surjan diferencias entre los objetivos y lo finalmente generado. El esquema general del algoritmo se muestra en la figura 4.1.

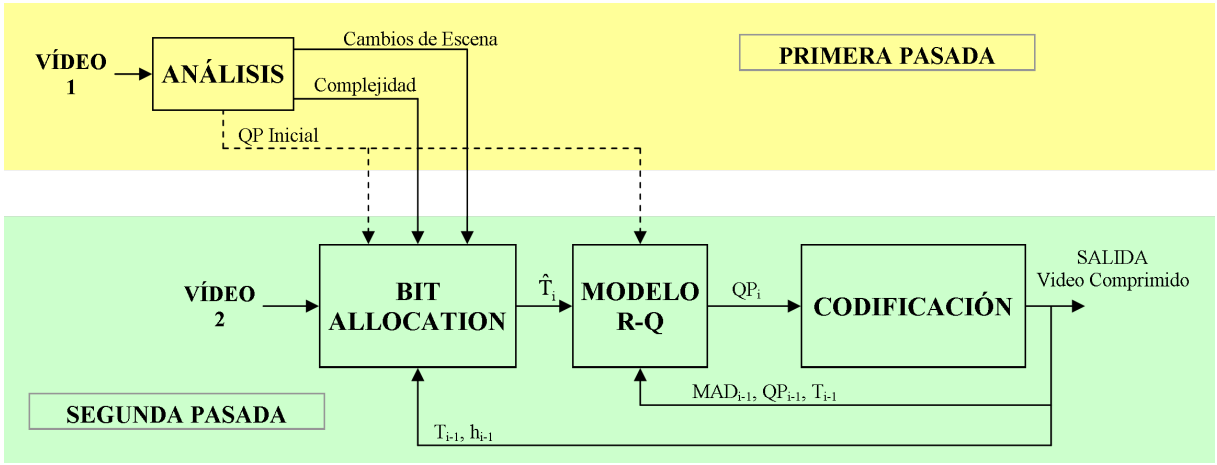


Figura 4.1: Esquema del algoritmo propuesto

4.2.1. Primera pasada

El objetivo principal de realizar una codificación previa es la de obtener un conocimiento de la secuencia de vídeo a procesar. Para ello, en esta primera codificación se calculará la complejidad de cada cuadro que compone la secuencia y se detectarán los cambios de escena presentes en la misma.

4.2.1.1. Cálculo de complejidades

En esta primera pasada se realizará un cálculo de la complejidad de cada cuadro que compone la secuencia. Esta complejidad se calculará mediante la expresión

$$C_i = QP_i \cdot Bits_i$$

donde

C es la complejidad calculada
i es el cuadro procesado
$Bits$ es la cantidad de bits empleados en codificar el residuo

La metodología a seguir ha sido codificar toda la secuencia con una QP constante, igual para todos los cuadros e independiente de las características de la secuencia (longitud, resolución, bit rate objetivo, etc.). En este proceso se obtiene una cantidad de bits para cada cuadro, correspondiente a los bits empleados en codificar su residuo de predicción. Una vez codificada la imagen, se calcula su complejidad mediante la expresión citada anteriormente y se almacena para pasársela posteriormente al codificador en la segunda pasada.
En las pruebas realizadas se ha utilizado un valor de QP igual a 30. El motivo de este valor es que está dentro de la franja intermedia de valores de QP permitidos en H.264[2] y entre los valores más recomendados en la bibliografía (de 20 a 30).

4.2.1.2. Detección de cambios de escena

Como se ha mencionado anteriormente, el objetivo del algoritmo es obtener una secuencia con una calidad perceptualmente estable, esto es, una secuencia el la que las escenas que la componen tengan una calidad estable, lo cual no quiere decir que todas tengan los mismos niveles de calidad.

Para poder tratar las escenas como entidades independientes primero habrá que detectarlas y delimitarlas. Para ello emplearemos un mecanismo de detección basado en el cálculo de la diferencia entre los histogramas de dos cuadros consecutivos.

La idea es sencilla. Un histograma representa la distribución de los diferentes valores que tienen los píxeles en un cuadro. Si hay un cambio de escena, los contenidos del último cuadro de la escena anterior y del primero de la nueva escena serán muy diferentes y, por lo tanto, sus histogramas también lo serán, haciendo que el parámetro "diferencia de histograma" presente un pico en esa transición. Por el contrario, si no hay un cambio de escena, el histograma del cuadro se mantendrá en valores similares a los histogramas de los cuadros vecinos.
Una ventaja de esta medida es que es relativamente inmune a los movimientos que puedan tener los objetos (salvo que éstos sean excesivamente rápidos), ya que el histograma sólo variará debido a las contribuciones de los descubrimientos y oclusiones producidas por los objetos desplazados.

En la propuesta de rate control calcularemos los histogramas de los cuadros I y P, y las diferencias entre ellos. Para el cálculo de estos histogramas consideraremos las tres componentes

[2] Recordar que los valores permitidos van de 0 a 51.

de la imagen (Y, Cb, Cr), teniendo en cuenta a la hora de normalizarlos las diferentes resoluciones de las componentes de luminancia (Y) y de crominancia (Cb y Cr), según el muestreo utilizado (ver apartado 2.1.2.2). Finalmente se utilizará como criterio la diferencia segunda (la diferencia de la diferencia) del histograma como magnitud a explorar para determinar la presencia o no de un cambio de escena comparándola con un umbral establecido, como se muestra en la figura 4.2. El motivo de usar la diferencia segunda en lugar de la diferencia primera es que en la diferencia segunda los picos que aparecen en un cambio de escena son más acentuados y *afilados*, lo que permite discernir mejor dónde está el cambio de escena entre los diferentes cuadros.

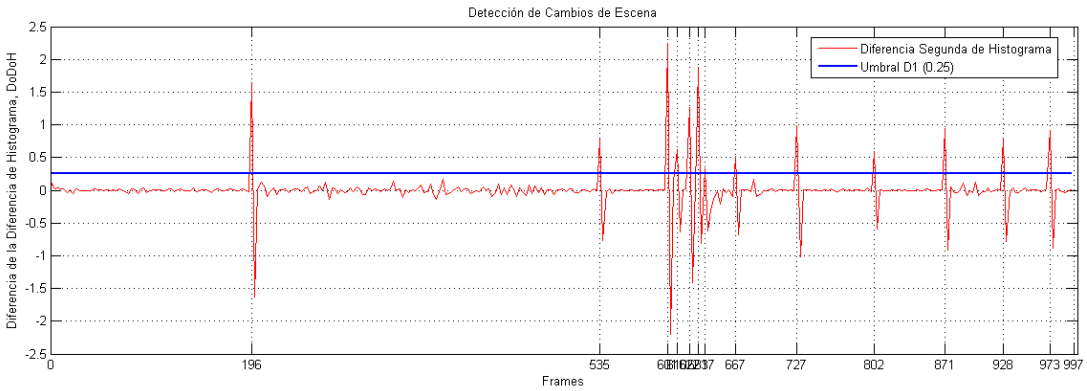


Figura 4.2: Detección de cambios de escena en *elpatriota_ 78*

Para la determinación de este umbral emplearemos las probabilidades de Falsa Alarma o Falsa Detección (P_{FA_i}), que indicará la probabilidad de datectar un cambio de escena donde realmente no lo hay, y de Pérdida (P_{M_i}), que indicará la probabilidad de no detectar un cambio de escena que realmente sí existe en ese punto. Las expresiones para calcular estas probabilidades para un determinado umbral η_i son

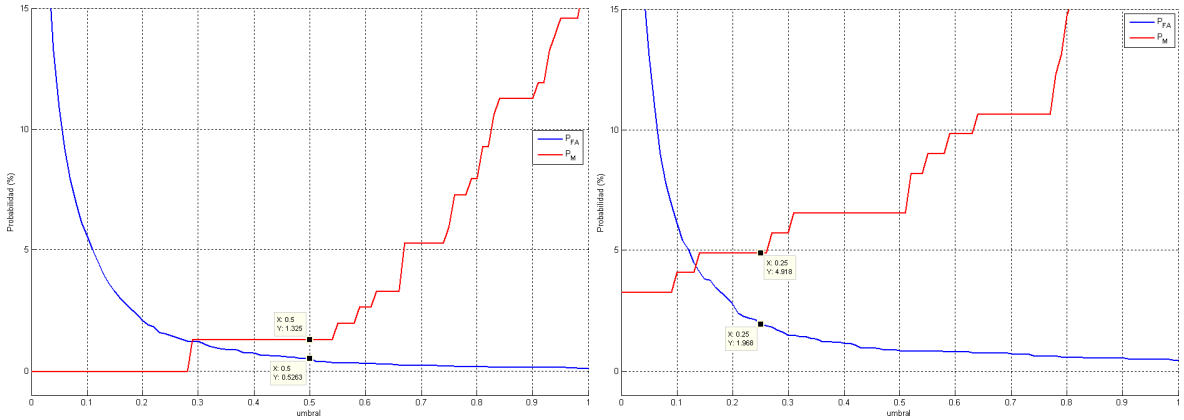
$$P_{FA_i} = \frac{\text{Total Falsas Alarmas}_i}{\text{Total No Cambios de Escena}}$$

$$P_{M_i} = \frac{\text{Total Pérdidas}_i}{\text{Total Cambios de Escena Reales}}$$

donde

Total Falsas Alarmas$_i$ es el número de falsas detecciones usando el umbral η_i
Total Perdidas$_i$ es el número de cambios de escena no detectados usando el umbral η_i
Total No Cambios de Escena es el número de cuadros en los que no ocurre un cambio de escena
Total Cambios de Escena Reales es el número de cambios de escena que realmente existen

Para elegir este umbral realizamos una serie de codificaciones de múltiples secuencias utilizando diferentes umbrales, de manera que podamos obtener las gráficas con la evolución de las probabilidades anteriores en función del umbral empleado. Se han realizado experimentos con dos resoluciones de vídeo, CIF (352 × 288) y D1 (720 × 576), obteniendose curvas diferentes. Estas curvas pueden verse en la figura 4.3. Cada resolución de vídeo a la entrada tendrá asociado un par de curvas (P_M y P_{FA}) diferente y, por lo tanto, deberá tener asociado un umbral de detección diferente. En cualquier caso podemos observar que las prestaciones de este método empeoran según se sube la resolución.



(a) CIF (b) D1

Figura 4.3: Probabilidades de falsa alarma y de pérdida

Antes de decidirnos por un valor de umbral u otro, hay que tener en cuenta que estamos en una situación de compromiso entre las probabilidades de falsa alarma y de pérdida. Al basarse nuestro algoritmo en la diferenciación de escenas, una tasa de pérdida elevada afectará negativamente a los resultados, ya que estaremos tratando de la misma manera dos escenas distintas (y con diferentes requerimientos). Si, por el contrario, la probabilidad de falsa alarma es elevada, supodrá la aparición de numerosas escenas *fantasma*. Estas escenas, como se verá posteriormente, han de ser inicializadas (hay que determinar su QP inicial) de manera independiente de las escenas anteriores. En una escena *fantasma* esta inicialización ofrecerá peores resultados que los obtenidos en el caso de que la detección hubiera sido correcta (es decir, que no se hubiera detectado ese cambio de escena *fantasma*). Con este error, la calidad sería peor durante unos cuadros hasta que se estabilizaran los parámetros. Además, con cada escena falsamente detectada se introduce un cuadro Intra, con el consiguiente gasto innecesario de bits.

En nuestro caso, considerando todos estos factores, hemos elegido los umbrales de la tabla 4.1.

Formato	Umbral	P_{FA}	P_M
CIF	0.5	0.5263 %	1.325 %
D1	0.25	1.968 %	4.918 %

Tabla 4.1: Umbrales utilizados para la detección de cambios de escena

A pesar de todo este estudio, hay que tener en cuenta que se trata de una decisión basada en un umbral fijo. Las alternativas basadas en umbrales dinámicos [31] pueden adaptarse mejor al contenido de la secuencia.

A lo largo de la primera pasada, cuando se detecta un cambio de escena se da por finalizado el GOP anterior y automáticamente se comienza un nuevo GOP, almacenando este punto para pasárselo a la segunda pasada como dato a tener en cuenta. El motivo de esto es forzar que el primer cuadro de la nueva secuencia sea de tipo Intra para así refrescar completamente su contenido. La estructura del GOP no cambia, pero el último GOP de la escena que termina sí puede ser más corto (ver figura 4.4). Esta reestructuración se realiza en la primera pasada para

que los tipos de todos los cuadros sean consistentes con los tipos de los cuadros de la segunda pasada, ya que si no se hiciera, estariamos utilizando la complejidad de un cuadro P para codificar el mismo como I, o usar la complejidad de un I para codificarlo como P, provocando un mal funcionamiento del codificador por asignar más o menos bits de los necesarios a ese cuadro.

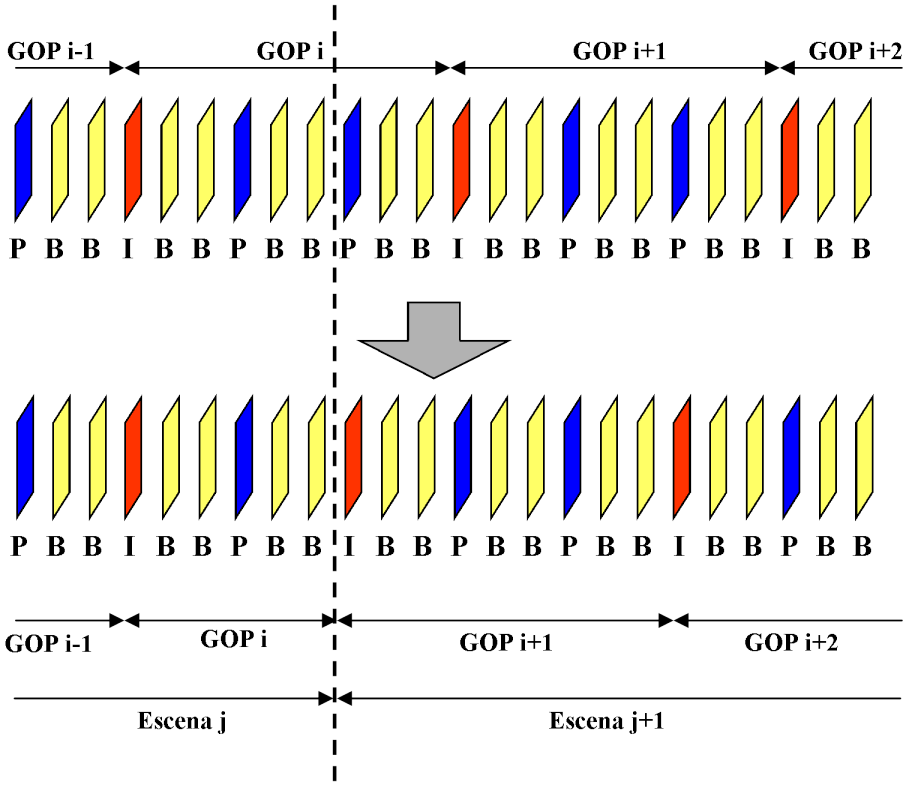


Figura 4.4: Reestructuración de los GOPs en un cambio de escena

Anteriormente se ha mencionado que en este algoritmo de Control de Tasa sólo calculamos las diferencias de histograma (y por lo tanto, los cambios de escena) en los cuadros I y P. El motivo de esto radica en el hecho de que el orden de codificación de los cuadros es distinto al orden de reproducción o visualización. Por un lado, si se consideraran los cuadros B y un cambio de escena recae en alguno de ellos, habría que convertir este cuadro B en I, pero además habría que convertir y recodificar cuadros ya codificados (el I ó P que le sirve de referencia y que se codifican antes, aunque se visualicen después), además de que se deformaría la estructura de GOP. Esto elevaría mucho la complejidad de implementación de este esquema. Además, el no considerar los cuadros B en los cambios de escena no afecta en mayor medida, ya que estos cuadros B, en caso de estar afectados por el cambio de escena tomarían como referencia el cuadro I/P vecino que mejor les corresponda (de una escena o de otra), como se puede ver en la figura 4.5. Debido a esto, al detectar una nueva escena no debemos *resetear* el buffer de imágenes de referencia[3], porque si se resetea, este cuadro B podría no tener una referencia válida y utilizaría muchos macrobloques intra, generando un gran desajuste en el presupuesto de bits y, presumiblemente, en la calidad. Para evitar este *reseteo*, hay que forzar que el primer cuadro de la nueva escena (de tipo I) no sea de tipo IDR[4]

[3]CPB: *Coded Picture Buffer*
[4]IDR: *Instantaneous Decoder Refresh*, *flag* en un cuadro I que indica que hay que borrar el buffer de referencia.

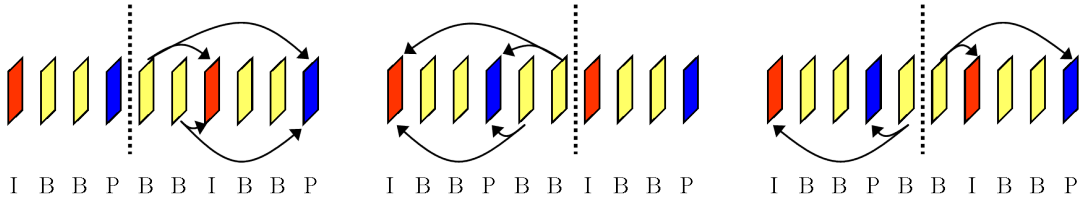


Figura 4.5: Comportamiento de los cuadros B en los cambios de escena

4.2.2. Segunda pasada

En esta segunda pasada se realizará la codificación, propiamente dicha, de la secuencia de vídeo usando el algoritmo VBR propuesto. Para ello hará uso de los datos obtenidos en la primera pasada: las complejidades de cada cuadro y la información de las escenas que componen la secuencia (número de escenas, longitud y posición del primer cuadro). Con esta información se realizará primeramente el proceso de reparto de bits, o *bit allocation*, para cada cuadro. Posteriormente se aplicará un modelo R-Q para obtener el parámetro de cuantificación, o QP, necesario para satisfacer la condición dada por el bit allocation.

4.2.2.1. Bit allocation

El mecanismo de reparto de bits se hará en cuatro niveles: Escena, GOP, miniGOP y Cuadro.

El motivo de tanta jerarquización es el de mantener lo más controlado posible el consumo de bits durante la codificación.
Se comienza por un nivel de escena porque lo primero que se persigue en este algoritmo es una calidad constante dentro de una escena. Por lo tanto, en la asignación de bits se debe tener en cuenta que los contenidos de dos escenas son diferentes y, por lo tanto, sus requerimientos en cuanto a bits/ancho de banda también lo serán.
Se continúa en un nivel de GOP, ya que se trata de una estructura que se repite a lo largo de la escena y que contiene siempre un cuadro de tipo Intra, que es, con diferencia, el que consume más bits.
Más adelante, introducimos un nuevo nivel, llamado miniGOP. Un miniGOP es un conjunto de un cuadro I o P, seguido del número de cuadros B consecutivos fijados en el codificador. Si, por ejemplo, el esquema de codificación es IP2B, un miniGOP sería una secuencia *IBB* ó *PBB*. Este nivel se introduce para tener un mayor control sobre el bit rate gastado. Cada miniGOP tendrá su propio presupuesto de bits, y un exceso o un defecto en su uso no repercutirá en el presupuesto de los siguientes miniGOPs, por lo que un fallo o error en la codificación de un cuadro no se debería propagar al siguiente miniGOP. Estos fallos, como se indicará después, se acumulan y se compensan a nivel de escena.
Finalmente tenemos un nivel de cuadro, en el que se repartirán los bits del miniGOP entre los cuadros que lo componen.

En todos los casos, el método de reparto es análogo, siendo éste un mecanismo de reparto proporcional a la complejidad total del cuadro o conjunto de cuadros que componen el nivel.

Espacio Total

Primeramente se calculará el espacio total objetivo que tiene que ocupar la secuencia una vez codificada. Para ello partimos del bit rate objetivo, que tomaremos como bit rate medio. El espacio total deseado será el ocupado por un flujo con un bit rate constante igual a la tasa media indicada:

$$Espacio\_total = bitrate \cdot \frac{N_{seq}}{F_r}$$

donde

N_{seq} es el número de cuadros que forman la secuencia
F_r es el frame rate

Nivel de Escena

Una vez calculado el espacio total objetivo, el presupuesto de bits para una escena será

$$BA_{escena_i} = Espacio\_total \cdot \frac{C_{escena_i}}{C_{seq}} + BA_{escena\_acumulado} \cdot \frac{C_{escena_i}}{C_{seq\_restante}}$$

donde

C_{escena_i} es la complejidad total de la escena que comienza, i.
C_{seq} es la complejidad total de la secuencia, calculada como la suma de las complejidades de todos los cuadros que la componen, C_l, obtenida en la primera pasada:

$$C_{seq} = \sum_{l=0}^{N_{seq}-1} C_l$$

$BA_{escena\_acumulado}$ es un acumulador con el numero de bits consumidos de menos (superávit de bits, valor positivo) o de más (déficit de bits, valor negativo) en las escenas anteriores.
$C_{seq\_restante}$ es la complejidad total de la parte de secuencia que queda por codificar

En esta expresión vemos que a la escena se le asigna una fracción del presupuesto total para la secuencia correspondiente a su complejidad relativa dentro de dicha secuencia.
El segundo término de la ecuación es un factor de ajuste. Ya que en la práctica resulta imposible consumir *exactamente* el número de bits presupuestados, al terminar un cuadro se habrán consumido bits de más o de menos. Si sumamos la contribución de todos los cuadros tendremos la desviación cometida en la escena entre los bits presupuestados y los consumidos. Este error cometido hay que compensarlo pasándolo a las siguientes escenas.
Ya que las escenas no tienen por qué tener la misma duración ni tener complejidades similares, a algunas les será más fácil absorber un deficit o superávit de bits que a otras, por lo que un reparto equitativo no sería adecuado en este caso. Debido a esto, el factor que regula la proporcionalidad será la relación entre la complejidad de la escena que comienza y la complejidad total pendiente de ser codificada. De este modo, escenas con mayor complejidad (que tendrán

a priori una mayor asignación de bits) podrán absorber un mayor déficit de bits procedente de escenas anteriores[5]. Este ajuste sólo se realizará entre escenas, ya que si se realizan dentro de una escena (por ejemplo, entre GOPs, como en [23, 24]) las variaciones de calidad provocadas serían perceptibles, degradando la calidad percibida. Sin embargo, al hacerlo entre escenas, estas variaciones quedan enmascaradas, ya que dentro de la escena la calidad sería más estable; con esta corrección se aumenta o disminuye la calidad media de la escena, pero no su estabilidad, que es lo más determinante.

Nivel de GOP

En el nivel de GOP, el mecanismo de asignación de bits es análogo al empleado en el nivel de escena, solo que aquí la relación de complejidades es entre la complejidad total del GOP, C_{GOP_j}, y la de la escena a la que pertenece, C_{escena_i}, no realizándose ninguna corrección adicional:

$$BA_{GOP_j} = BA_{escena_i} \cdot \frac{C_{GOP_j}}{C_{escena_i}}$$

Nivel de miniGOP

Como se ha explicado anteriormente, en este algoritmo de control de tasa introducimos un nuevo nivel, llamado *miniGOP* para controlar el consumo de bits y evitar que se propaguen dentro del GOP consumos excesivamente grandes o excesivamente pequeños de cuadros anteriores. El reparto en este nivel solo se efectúa cuando el codificador va a codificar un cuadro I (siempre que no sea el primer cuadro I de una escena, en cuyo caso este cuadro se codifica aparte) o P. Como en niveles anteriores, el reparto se realiza mediante un cociente de complejidades usando una expresión similar a la empleada en el nivel de GOP:

$$BA_{miniGOP_k} = BA_{GOP_j} \cdot \frac{C_{miniGOP_k}}{C_{GOP_j}}$$

Nivel de Cuadro

Finalmente, el nivel más bajo de reparto de bits es el nivel de cuadro. En él se reparten los bits según el esquema habitual en este algoritmo, tomando como referencia la asignación y la complejidad del nivel de miniGOP:

$$T_l = BA_{miniGOP_k} \cdot \frac{C_l}{C_{miniGOP_k}}$$

Existe un caso especial, y es el del primer cuadro I de una escena. La asignación de bits a

[5] Tambien absorberán un mayor superávit, pero se considera sobre todo el caso del déficit, ya que si éste es muy grande, en escenas muy cortas o poco complejas el presupuesto podría quedarse negativo

este cuadro se toma referente a la complejidad de la escena que comienza:

$$T_l = BA_{escena_i} \cdot \frac{C_l}{C_{escena_i}}$$

Podemos ver la evolución del reparto de bits a lo largo de los diferentes niveles en la figura 4.6.

Una cuestión que hay que tener presente es que el mecanismo de reparto de bits se basa en complejidades relativas, esto es, cocientes de complejidades. Debido a esto, si la complejidad medida (en una escena, GOP, miniGOP o cuadro) es muy baja (incluso próxima a cero) los errores cometidos en el reparto de bits pueden ser muy grandes. Por lo tanto, los fragmentos de secuencia con complejidad baja son más propensos a errores.

4.2.2.2. Selección de QP

En esta fase de la codificación se partirá del presupuesto de bits para cada cuadro obtenido en la fase de bit allocation y se persigue obtener una QP que se ajuste a este presupuesto. Ya que, como se comentó en el apartado 3, obtener la curva ORD no es computacionalmente asumible, emplearemos modelos R-D, que en este caso serán cuadráticos y exponenciales.

Cálculo de QP en cuadros I

Para la determinación de la QP de los cuadros Intra se sigue un proceso basado en la propuesta de Zhou [35]. Este algoritmo es de una única pasada, por lo que lo adaptaremos para nuestro algoritmo de dos pasadas.

Se trata de un modelo R-D exponencial para la determinación de QP que tiene la característica de apoyarse no solo en el presupuesto de bits para ese cuadro, sino también en su contenido, usando gradientes e histogramas para su determinación. En el caso de cuadros Intra se corresponde con la siguiente expresión:

$$T_l = \alpha_I \cdot (\text{Grad} \cdot \text{SOH}) \cdot e^{(-\beta_I \cdot \widehat{QP}_l)}$$

de la que, despejando, podemos obtener la estimación de QP:

$$\widehat{QP}_l = \frac{-1}{\beta_I} \cdot \ln\left(\frac{T_l}{\alpha_I \cdot \text{Grad} \cdot \text{SOH}}\right)$$

donde

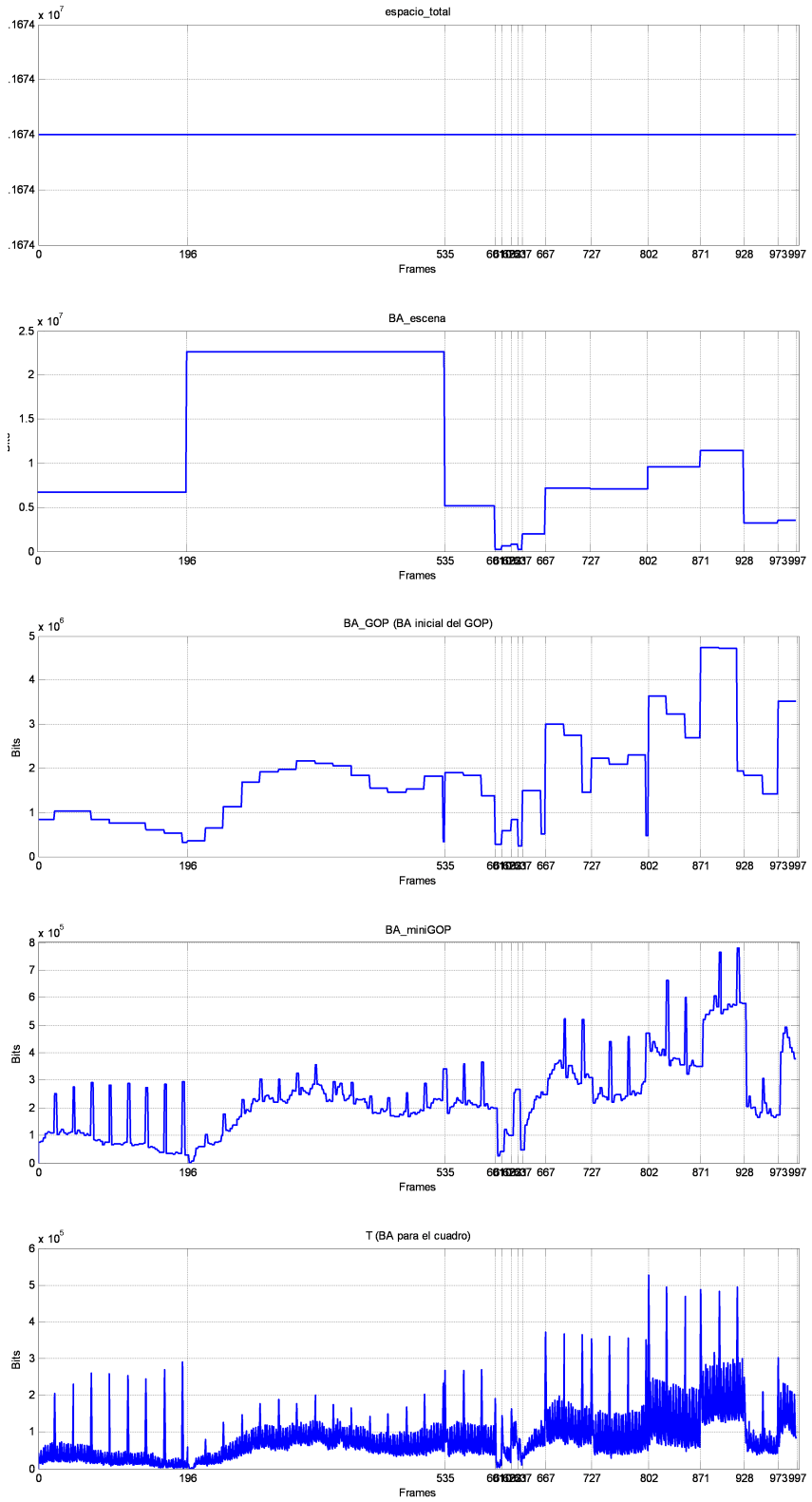


Figura 4.6: Evolución del bit allocation a lo largo de los cinco niveles (secuencia, escena, GOP, miniGOP y cuadro) en *elpatriota_78*

Grad es el gradiente del cuadro a codificar, calculado según la expresión

$$\begin{aligned}\text{Grad} \;=\; & \sum_{i=0}^{M_{\text{Y}}-2}\sum_{j=0}^{N_{\text{Y}}-2}\left(|\text{Y}_{i,j}-\text{Y}_{i,j+1}|+|\text{Y}_{i,j}-\text{Y}_{i+1,j}|\right)/(M_{\text{Y}}N_{\text{Y}})+\\ & +\sum_{i=0}^{M_{\text{Cb}}-2}\sum_{j=0}^{N_{\text{Cb}}-2}\left(|\text{Cb}_{i,j}-\text{Cb}_{i,j+1}|+|\text{Cb}_{i,j}-\text{Cb}_{i+1,j}|\right)/(M_{\text{Cb}}N_{\text{Cb}})+\\ & +\sum_{i=0}^{M_{\text{Cr}}-2}\sum_{j=0}^{N_{\text{Cr}}-2}\left(|\text{Cr}_{i,j}-\text{Cr}_{i,j+1}|+|\text{Cr}_{i,j}-\text{Cr}_{i+1,j}|\right)/(M_{\text{Cr}}N_{\text{Cr}})\end{aligned}$$

en la que

$\text{Y}_{\text{i,j}}, \text{Cb}_{\text{i,j}}, \text{Cr}_{\text{i,j}}$ son la luminancia y las crominancias del píxel de posición (i,j)
$M_{\text{Y}}, N_{\text{Y}}$ son las filas y columnas que componen la luminancia
$M_{\text{Cb}}, N_{\text{Cb}}, M_{\text{Cr}}, N_{\text{Cr}}$ son las filas y columnas que componen las crominancias

SOH es el estadístico del histograma del cuadro, calculado como

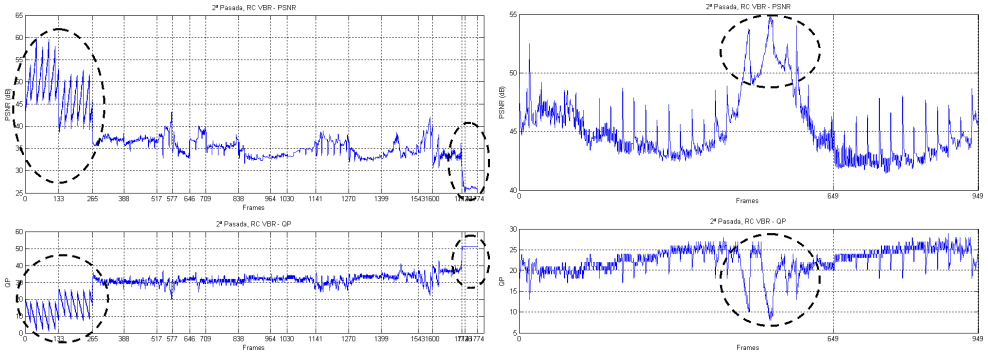
$$\text{SOH} = \sum_{k=0}^{255}\left(\log_2 \text{Hist}_{\text{Y}}[k] + \log_2 \text{Hist}_{\text{Cb}}[k] \log_2 \text{Hist}_{\text{Cr}}[k]\right)$$

en la que $\text{Hist}_{\text{Y}}, \text{Hist}_{\text{Cb}}, \text{Hist}_{\text{Cr}}$ son los histogramas de la luminancia y las crominancias

$\widehat{QP}_l$ es el parámetro de cuantificación que queremos determinar

T_l es el presupuesto de bits para el cuadro

β_I es una constante del modelo. Si bien el autor indica que su valor es 0.1, experimentalmente hemos comprobado que este valor no siempre se comporta bien, dando peores resultados numéricos e incluso inestabilidades en algunos casos en el cálculo de los cuadros Intra (ver figura 4.7).



(a) *cities* (CIF) (b) *master_22* (D1)

Figura 4.7: Fallos al usar β_I=0.10

Tras hacer varios barridos hemos llegado a la conclusión de que el mejor valor, para el que se observan los mejores resultados numéricos (unas veces evidentes y otras veces sutiles) y ninguna inestabilidad, sería 0.14, que es el que emplearemos en todas las pruebas realizadas. No obstante, en el desarrollo del algoritmo se ha observado que para secuencias de tamaño

CIF el valor 0.13 también da valores satisfactorios. Esto nos indica que en caso de realizar pruebas con otras resoluciones, como QCIF, 4CIF, 720p, 1080p, etc.[6], este parámetro debería estudiarse para determinar su valor adecuado para cada tamaño de cuadro.

α_I es un parámetro del modelo. Su valor inicial se determina al inicio de cada escena usando datos de la primera pasada y se actualiza tras codificar un cuadro Intra según la expresión

$$\alpha_I = \frac{nbits_I}{\text{Grad} \cdot \text{SOH} \cdot e^{-\beta_I \cdot QP_l}}$$

en la que

$nbits_I$ es la cantidad de bits *realmente* empleados en codificar el cuadro Intra actual

QP_l es la QP que se ha utilizado para codificar el cuadro Intra actual

Una vez determinado el valor de $\widehat{QP}_l$, se aplican unos controles de variación para limitar la variación de QP a fin de evitar saltos bruscos de calidad.
Primeramente se impondrá un límite de variación a medio plazo para la QP de ±5 unidades entre el cuadro actual y el Intra anterior. Este valor es relativamente elevado[7] para favorecer la convergencia del modelo (si los cuadros P y B se adaptan, el Intra supondría un lastre si su capacidad de variación fuera menor, p.ej. ±2). Posteriormente se introduce otro control de variación a corto plazo, ±2 unidades respecto a la QP del cuadro anterior, para garantizar una fluctuación suave de la calidad.

$$\widetilde{QP}_l = \text{mín}\left\{\text{máx}\left\{QP_{Intra\_ant} - 5, \widehat{QP}_l\right\}, QP_{Intra\_ant} + 5\right\}$$

$$\widetilde{QP}_l = \text{mín}\left\{\text{máx}\left\{QP_{l-1} - 2, \widetilde{QP}_l\right\}, QP_{l-1} + 2\right\}$$

Este esquema está adaptado para obtener la QP de un cuadro I a partir de los datos de los cuadros I anteriores (mediante la actualización del parámetro α_I). Sin embargo esta información no está disponible para el caso del primer cuadro Intra de una escena, ya que no podemos presuponer una correlación con el cuadro Intra anterior por pertenecer éste a otra escena. Debido a esto se toma la información obtenida en la primera pasada en el mismo cuadro.

Primeramente se hace el bit allocation para este primer cuadro, ya que en la fase de bit allocation no se hacía. Al ser un cuadro crítico, ya que determinará en gran medida la calidad de los cuadros sucesivos por ser la primera referencia, su presupuesto se tomará de forma proporcional a la escena completa:

$$T_l = BA_{escena_i} \cdot \frac{C_l}{C_{escena_i}}$$

La información extraida de la primera pasada servirá para calcular un valor inicial del parámetro α_I, que redundará en una QP inicial:

$$\alpha_I = \frac{nbits_I^{1pass}}{\text{Grad} \cdot \text{SOH} \cdot e^{-\beta_I \cdot QP^{1pass}}}$$

donde

[6] Las pruebas realizadas en este Proyecto se corresponden a secuencias CIF (352×288) y D1 (720×576)
[7] En la mayoría de implementaciones el límite de variación es ±2 ó ±3

$nbits_I^{1pass}$ es el número de bits que se gastaron para codificar ese mismo cuadro I en la primera pasada

QP^{1pass} es la QP (constante) empleada en la primera pasada

Una vez hayados estos valores, acudimos a la fórmula del modelo R-D:

$$\widehat{QP}_l = \frac{-1}{\beta_I} \cdot \ln\left(\frac{T_l}{\alpha_I \cdot \text{Grad} \cdot \text{SOH}}\right)$$

La QP de este primer cuadro Intra de la escena no está sujeto a los controles de variación, ya que la variación de contenido de una escena a la siguiente puede ser grande. Este valor de QP se utilizará también para el primer cuadro P y el primer B de una escena.

$$\widetilde{QP}_l = \widehat{QP}_l$$

En la ejecución del algoritmo podemos indicar dos situaciones especiales que requieren una intervención adicional:

- Podría darse que el presupuesto para una escena fuera negativo (bien porque sea una de las últimas escenas, bien porque haya un déficit de bits excesivo procedente de escenas anteriores). En este caso, la QP del primer cuadro Intra (y, por lo tanto, del primer P y del primer B) será igual a su valor máximo, 51, saltándose cualquier control de variación de QP.
- En algunos casos se ha podido comprobar que en ciertos cambios de escena el algoritmo no es capaz de estimar una QP inicial de la secuencia, provocando saltos de QP a valores extremos (hasta QP=0 ó 51). Viendo los resultados se ha concluído que estos saltos no tienen sentido y se deben a que el modelo falla. Para acotar el error cometido en estos casos basta con poner un límite de variación admisible para la QP de los cuadros Intra (en nuestro caso se ha optado por un límite de 7 unidades). De este modo, si el algoritmo intenta un salto de QP mayor de 7 unidades respecto al Intra anterior, se considerará que ha habido un error en el modelo y se descartará el valor calculado, tomando finalmente una QP igual a la QP del cuadro Intra anterior.

Por último se realiza una limitación o *clipping* para asegurarnos que los valores finales de QP estén dentro del rango aceptado por el estándar.

$$QP_l = \text{mín}\left\{\text{máx}\left\{\widetilde{QP}_l, 1\right\}, 51\right\}$$

Cálculo de QP en cuadros P y B

Para calcular la QP perteneciente a los cuadros P y B se empleará un modelo cuadrático. Dicho modelo se corresponde con la expresión

$$T_l = \frac{C_1 \times \widehat{MAD}_l}{\widehat{QP}_l} + \frac{C_2 \times \widehat{MAD}_l}{\widehat{QP}_l^2} + \hat{h}_l$$

donde

T_l son los bits presupuestados para el cuadro, calculados en el proceso de bit allocation
$\widehat{QP}_l$ es el parámetro de cuantificación que estamos calculando
C_1 y C_2 son parámetros del modelo. Son actualizados cada vez que se codifica un cuadro mediante el esquema de regresión lineal propuesto en [19, 20]
$\hat{h}_l$ es la predicción de los bits que se van a gastar en las cabeceras del cuadro. Su valor será el número de bits gastados realmente en el cuadro anterior: $\hat{h}_l = h_{l-1}$
$\widehat{MAD}_l$ es la predicción de la MAD del cuadro codificado. Como se mencionó en el apartado 3.2, esta predicción se hace con el objetivo de resolver el *dilema del huevo y la gallina*, y se realiza mediante la expresión

$$\widehat{MAD}_l = a_1 \cdot MAD_{l-1} + a_2$$

donde

a_1 y a_2 son parámetros del modelo. Son actualizados mediante regresión lineal.
MAD_{l-1} es la MAD conseguida realmente en el cuadro anterior

Este modelo se emplea en los cuadros P y B, pero de manera desacoplada. Esto quiere decir que el esquema empleado, parámetros, actualizaciones, etc. son *formalmente* iguales en ambos tipos de cuadro, pero la implementación es doble, un modelo para los cuadros P y otro para los cuadros B, formalmente igual, pero independiente del funcionamiento de los cuadros P. De este modo, para las actualizaciones de parámetros sólo se tendrán en cuenta los datos correspondientes a los cuadros anteriores del mismo tipo.
Para tener un punto de partida favorable para el mecanismo de predicción, la QP del primer cuadro P y el primer B de cada escena no se determinará mediante el modelo cuadrático descrito, sino que será directamente la misma QP del primer cuadro Intra de dicha escena.

Posterior a la obtención de $\widehat{QP}_l$ se realiza un control de variación de QP frente a la QP del cuadro inmediatamente anterior, QP_{l-1}. Esta variación máxima será de ± 2 unidades para los cuadros P y de ± 1 unidad en el caso de los cuadros B. El motivo de esta diferencia es doble: por un lado los cuadros B presentan unos valores menores de complejidad y por lo tanto el cálculo de QP en los cuadros B es más propenso a fallos. Por otro lado, el esquema empleado ha sido IP2B; si un modelo (P/B) falla el otro tiene que poder compensar este fallo (salvo que falle éste también). Como hay el doble de cuadros de tipo B que de tipo P, su variación máxima debe ser la mitad.
Esta decisión se apoya en las sucesivas pruebas que se realizaron en la fase de desarrollo y validación del algoritmo, en las que hubo casos en los que el modelo fallaba en los cuadros B y funcionaba correctamente en los cuadros P. Si la variación máxima permitida era ± 2 unidades para los dos tipos de cuadro, al cabo de un miniGOP, los dos cuadros B habian contribuido a la QP con ± 4 unidades, mientras que los cuadros P sólo contribuían ∓ 2 unidades, no pudiendo compensar el error introducido por los B y haciendo que la QP tendiera a 0 ó 51, provocando un fallo muy grande (muchas veces irreparable) en la codificación para lo que queda de secuencia, ya que los consumos de bits se alejaban mucho (en un sentido o en otro) de lo presupuestado.

En resumen, el control de variación se corresponde con las siguientes expresiones:

$$\text{Cuadros P:} \quad \widetilde{QP}_l = \text{mín}\left\{QP_{l-1} + 2, \text{máx}\left\{QP_{l-1} - 2, \widehat{QP}_l\right\}\right\}$$

$$\text{Cuadros B:} \quad \widetilde{QP}_l = \text{mín}\left\{QP_{l-1}+1, \text{máx}\left\{QP_{l-1}-1, \widehat{QP}_l\right\}\right\}$$

Es importante destacar que en el caso de que T_l sea negativo no se ejecutará el modelo (ni en cuadros P ni en B), sino que directamente se determinará la QP (para los dos tipos de cuadro) como

$$\widetilde{QP}_l = QP_{l-1} + 2$$

Por último se realiza una limitación o *clipping* para asegurarnos que los valores finales de QP estén dentro del rango aceptado por el estándar.

$$QP_l = \text{mín}\left\{\text{máx}\left\{\widetilde{QP}_l, 1\right\}, 51\right\}$$

Un factor a tener en cuenta es que la medida de la complejidad, $QP \cdot Bits$, no es lineal, por lo que el cálculo del parámetro de cuantificación estará sujeto a errores debidos a esta no linealidad de la complejidad. En este caso se parte de un presupuesto de bits para un determinado cuadro de un determinado tipo. Si este presupuesto difiere mucho de la cantidad de bits empleada en ese mismo cuadro en la primera pasada, el binomio QP-Bits de la segunda pasada será muy diferente al obtenido en la primera. De este modo, si calculamos la complejidad con los valores de la segunda pasada, ésta presumiblemente dará un valor diferente (puede que mucho).

$$QP_1 \cdot Bits_1 \neq QP_2 \cdot Bits_2$$

En este caso, la QP obtenida presumiblemente no será la óptima. Es por este motivo por el que hay que intentar que la QP de la primera pasada esté lo más próxima posible a la ideal, como se presupone en [23].

Capítulo 5

Pruebas realizadas

El algoritmo de control de tasa propuesto en el capítulo anterior se ha implementado sobre el software de referencia de la ITU versión JM 10.2 [36]. Este algoritmo está enteramente programado en lenguaje C bajo el entorno Microsoft Visual C++ 6.0.
La configuración del codificador en estas pruebas ha sido la siguiente:

- Tasa Objetivo: 512 kbps (CIF) y 2048 kbps (D1)
- Frame rate: 25 fps
- Número de cuadros de referencia: 5
- Uso de cuadros B como referencia: Sí
- Estructura de GOP: IPBB
- Tamaño de GOP (genérico[1]): 24 cuadros (1 I + 7 P + 16 B)
- Optimización R-D: Activada
- Codificación entrópica: CABAC
- Modo de estimación de movimiento: EPZS
- Rango de búsqueda en estimación de movimiento: 16 (CIF) y 32 (D1) píxeles
- Transformada Hadamard: No

Para la validación del algoritmo se han utilizado dos conjuntos de secuencias de prueba, de resoluciones CIF (352 × 288) y D1 PAL (720 × 576) y de complejidades espaciales y temporales dispares. Podemos clasificar las secuencias utilizadas en cuatro categorías, según su complejidad de textura o espacial (alta o baja) y según su movimiento (elevado o bajo). Esta clasificación se corresponde con las tablas 5.1 y 5.2.

[1] Puede verse acortado en caso de cambio de escena.

	Complejidad Textura	Cantidad Movimiento	Número Cuadros	Número Escenas	Transiciones Abruptas	Número Fundidos
airshow	Baja	Baja	1765	27	25	1
bohemia	Alta	Baja	1249	9	5	3
cities	Alta	Baja	1774	17	16	0
iceage	Alta	Baja	3295	52	48	3
starwars	Alta	Alta	2998	56	55	0
highway	Baja	Baja	1999	1	0	0
football	Alta	Alta	124	1	0	0
mobile	Alta	Alta	418	3	2	0
paris	Alta	Baja	1063	1	0	0

Tabla 5.1: Características de las secuencias CIF empleadas

	Complejidad Textura	Cantidad Movimiento	Número Cuadros	Número Escenas	Transiciones Abruptas	Número Fundidos
elpatriota_78	Alta	Alta	997	11	10	0
jamesbond_27	Baja	Alta	997	20	19	0
lotr_4 + lotr_34 + spiderman_18	Alta	Alta	2167	31	30	0
master_22	Baja	Baja	949	2	0	1
tempete	Alta	Alta	259	1	0	0
ultimosamurai_18	Alta	Baja	997	26	25	0
spiderman_18	Alta	Alta	997	38	37	0

Tabla 5.2: Características de las secuencias D1 empleadas

En estas pruebas efectuaremos una comparación entre los resultados que se han obtenido con el algoritmo propuesto y los obtenidos mediante una pasada a QP constante (la efectuada en la primera pasada, QP=30) por constituir esta una referencia válida de codificación VBR con una variación de calidad a corto plazo muy pequeña. Además, compararemos con el algoritmo CBR de una pasada implementado en el software de referencia JM 10.2 [36], por constituir éste un estándar (JVT-G012 [18]). Para ello utilizaremos secuencias con diferentes niveles de complejidad para cada resolución empleada.
Sin embargo, como se puede ver en las tablas anteriores, el número de secuencias estudiadas es demasiado numeroso para poder mostrar unos resultados fácilmente legibles, por lo que en este capítulo sólo se mostrarán los resultados más significativos e ilustrativos de cada experimento, dejando expuestos los resultados del resto de secuencias en el apéndice B.

5.1. Pruebas sobre el funcionamiento general

Primeramente presentaremos un resumen de las pruebas realizadas en forma de datos numéricos. En estos resultados se detallan la precisión en el control del gasto de bits, en forma del bit rate medio conseguido y la desviación de dicho bit rate respecto de la tasa objetivo especificada a priori. Además, se detalla el valor medio de la QP empleada en la secuencia para poder valorar la validez de la complejidad medida en la primera pasada, en cuanto a la proximidad o lejanía de los valores de QP. A continuación utilizaremos el valor medio de la PSNR de la secuencia como medida del nivel de calidad alcanzada. Finalmente utilizaremos la desviación típica de la PSNR y la mediana de la varianza local de la PSNR (con una ventana de 10 cuadros) como medidas *objetivas* de la estabilidad de la calidad final de la secuencia. Estos resultados numéricos se detallan en las tablas B.1 (para secuencias CIF) y B.2 (para secuencias D1).

Secuencia		Bitrate (kbps)	Desviación (%)	$\overline{QP}$	$\overline{PSNR}$ (dB)	σ_{PSNR}	Mediana Var.Local
bohemia	1ª Pasada	-	-	30	36.13	1.81	0.042
	2ª Pasada	514.44	0.48	23.9	40.68	2.44	0.156
	CBR	521.58	1.87	24.1	41.05	3.32	0.871
starwars	1ª Pasada	-	-	30	36.60	1.03	0.093
	2ª Pasada	512.75	0.15	27.0	38.85	1.09	0.160
	CBR	512.11	0.021	27.7	38.90	3.41	1.241
highway	1ª Pasada	-	-	30	37.40	0.40	0.045
	2ª Pasada	498.79	-2.58	24.6	40.05	0.45	0.113
	CBR	512.27	0.053	25.2	39.89	1.22	0.419
mobile	1ª Pasada	-	-	30	32.30	0.50	0.056
	2ª Pasada	509.80	-0.43	35.2	27.68	0.45	0.057
	CBR	516.69	0.92	35.9	28.62	1.78	0.891

Tabla 5.3: Resultados numéricos de las secuencias CIF (Tasa objetivo: 512 kbps)

Secuencia		Bitrate (kbps)	Desviación (%)	$\overline{QP}$	$\overline{PSNR}$ (dB)	σ_{PSNR}	Mediana Var.Local
master_22	1ª Pasada	-	-	30	40.46	2.69	0.096
	2ª Pasada	2012.0	-1.76	22.2	45.18	2.71	0.230
	CBR	2051.4	0.17	24.4	43.84	5.55	1.363
elpatriota_78	1ª Pasada	-	-	30	38.22	2.03	0.142
	2ª Pasada	2057.8	0.48	26.3	40.87	2.11	0.318
	CBR	2083.0	1.71	26.3	40.92	4.22	1.405
jamesbond_27	1ª Pasada	-	-	30	38.28	1.34	0.164
	2ª Pasada	2053.5	0.27	26.8	40.45	1.26	0.261
	CBR	2055.9	0.38	26.4	40.69	3.67	1.309
lotr_4 + lotr_34 + spiderman_18	1ª Pasada	-	-	30	37.83	1.55	0.130
	2ª Pasada	2047.1	-0.042	24.6	41.48	1.61	0.340
	CBR	2049.8	0.088	24.3	41.89	3.65	1.215

Tabla 5.4: Resultados numéricos de las secuencias D1 (Tasa objetivo: 2048 kbps)

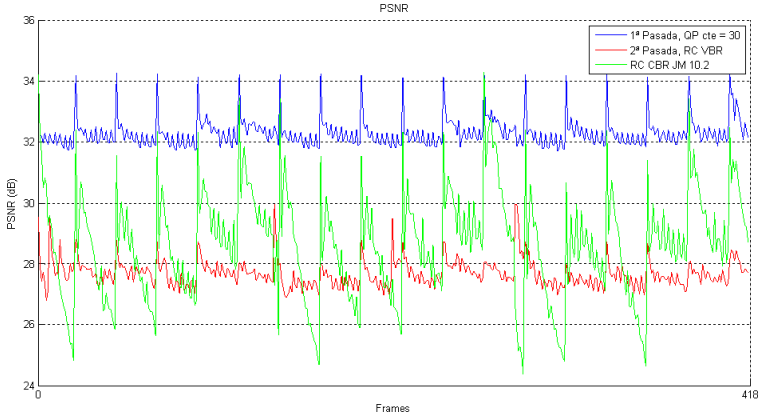
Viendo estos resultados podemos ver que el algoritmo propuesto tiene un control sobre el consumo de bits a largo plazo (visto como la tasa media) muy efectivo, a veces mejor incluso que el algoritmo CBR. Sin embargo, podemos ver dos casos significativos: *master_22* y *highway*, en los que el control no es tan bueno. El motivo de que tengan desviaciones elevadas reside en el mecanismo de ajuste del bit allocation a nivel de escena. Éste presenta los siguientes inconvenientes:

- Si la secuencia está formada por una única escena, la secuencia no tendrá posibilidad de ajuste para ceñirse al presupuesto de bits, confiando todo al buen funcionamiento del conjunto «Bit Allocation + Modelo R-Q».
- Si las escenas presentes son muy grandes es posible que se acumulen conjuntos de déficit/superávit de bits muy grandes y difíciles de compensar.
- Al final de la secuencia las escenas finales tienen que compensar todo el déficit que haya. Si éstas son muy cortas y/o están precedidas de una escena muy grande que genere mucho déficit, habrá problemas en cuanto que pueden quedarse sin presupuesto y no poder compensar el déficit heredado de escenas anteriores.

Highway está formada por una única escena (de mucha duración), mientras que *master_22* lo está por dos escenas largas. Esto hace que gran parte de la secuencia (o toda ella) se codifique sin ningún tipo de ajuste, por lo que los errores finales no se corrigen. Por el contrario, en *starwars* se han detectado 59 escenas y en *lotr_4 + lotr_34 + spiderman_18*, 47. Esto favorece un ajuste frecuente de los errores cometidos en el consumo de bits. Sin embargo, puede que no se cometan errores significativos en este consumo, como se puede ver en *mobile* (esta secuencia está compuesta por la concatenación de una misma escena tres veces, por lo que al ser estas *escenas* tan similares resulta muy difícil detectarlas) y su desviación respecto del objetivo es pequeña.

Respecto a la PSNR, los resultados son los esperables, ya que su valor es similar al alcanzado por el algoritmo CBR, pero casi siempre algo menor. Esto sucede porque se pierde valor absoluto a cambio de estabilidad, como podemos ver en los datos de la desviación típica y de la mediana de la varianza local. Respecto a estos últimos, si bien la desviación típica es un valor ampliamente utilizado en la bibliografía, esta medida puede presentar valores engañosos. Es por ello que se utiliza la mediana de la varianza local, ya que ésta analiza la estabilidad de la PSNR en un intervalo establecido (en este caso, 10 cuadros), de manera que refleja más fielmente el hecho de que un espectador valora subjetivamente las variaciones a corto plazo, más que las variaciones a lo largo de *toda* la secuencia.
Podemos ver que los valores de la mediana de la varianza local son algo mayores que en el caso de QP constante, pero la diferencia en términos absolutos tampoco es demasiado grande. Esto es esperable, ya que el caso de QP constante se acerca a la variación mínima conseguible a corto plazo. El hecho de obtener unos valores tan cercanos indica un buen funcionamiento del algoritmo propuesto en cuanto a estabilidad de la calidad se refiere. Comparando éste último con el algoritmo CBR podemos ver que las diferencias son grandísimas, tanto en valores absolutos como en relativos. Esto es debido a que en nuestro algoritmo se permite una mayor variación de tasa a corto plazo y al hecho de conocer la secuencia previamente a la codificación.

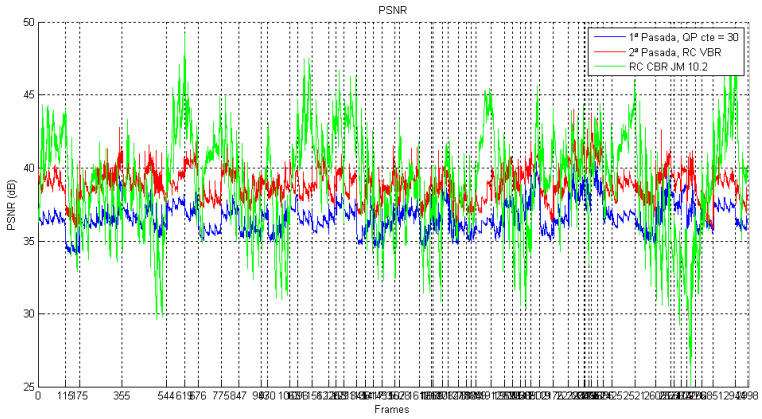
De manera gráfica, podemos ver la evolución de la PSNR y del parámetro de cuantificación, QP, a lo largo de las secuencias CIF *mobile* y *starwars* en las figuras 5.1 y 5.2, respectivamente, y de las secuencias D1 *elpatriota_78* y *jamesbond_27* en las figuras 5.3 y 5.4, respectivamente. Las líneas verticales discontinuas indican un cambio de escena detectado.



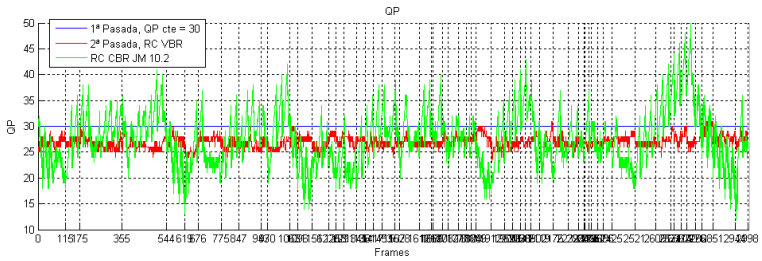
(a) PSNR

(b) QP

Figura 5.1: Evolución de PSNR y QP en *mobile* (CIF)

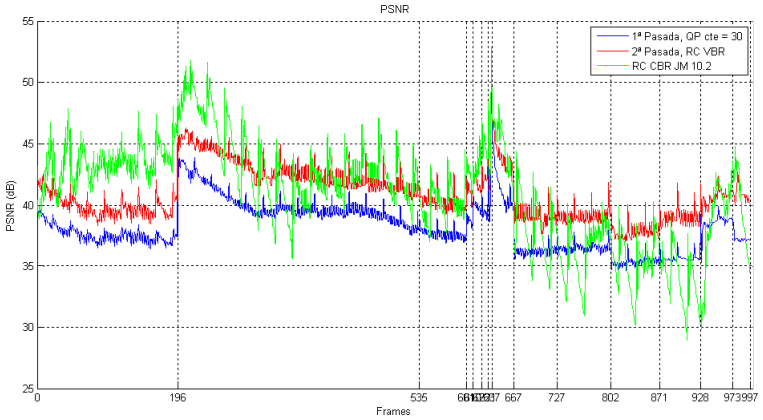


(a) PSNR

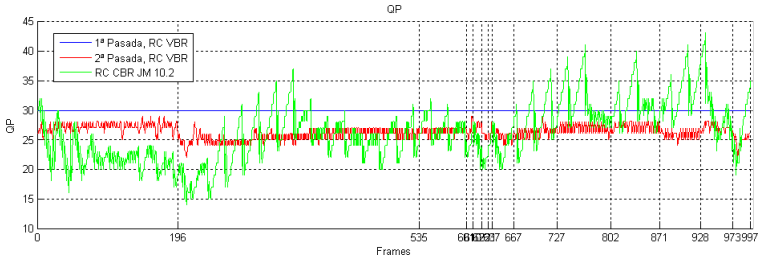


(b) QP

Figura 5.2: Evolución de PSNR y QP en *starwars* (CIF)

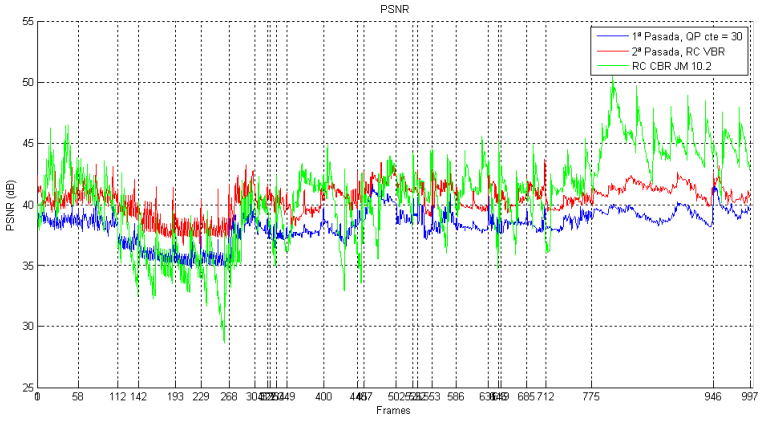


(a) PSNR

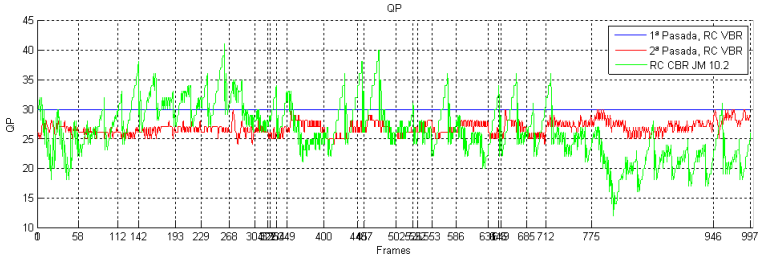


(b) QP

Figura 5.3: Evolución de PSNR y QP en *elpatriota_ 78* (D1)



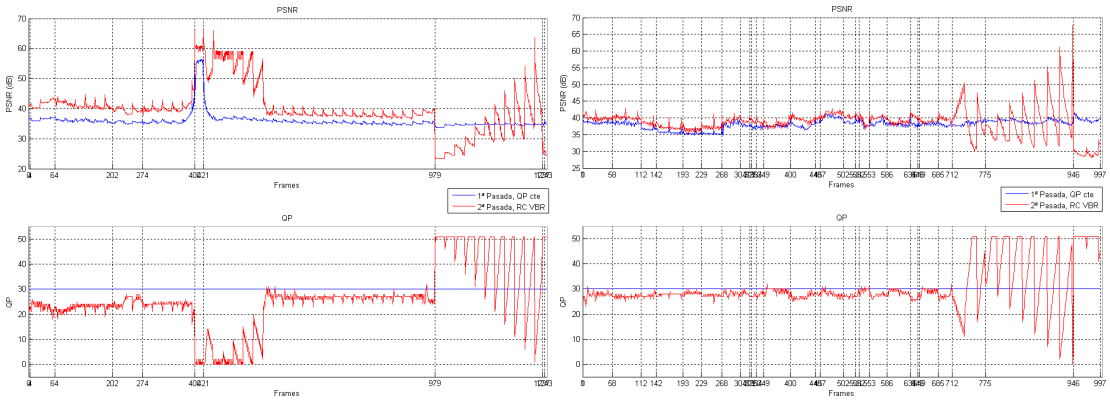
(a) PSNR



(b) QP

Figura 5.4: Evolución de PSNR y QP en *jamesbond_ 27* (D1)

Para tratar de paliar las desviaciones de la tasa media final mencionadas anteriormente se ha probado un mecanismo en el que la corrección a nivel de escena pasará a ser a dos niveles, escena y GOP, en el caso de que, al terminar de codificar una escena (o al inicio de la secuencia), falten 3 escenas o menos (con esto nos cubrimos del caso de una escena final grande) y/o falte por gastar el 33 % del espacio total (para cubrirnos en el caso de escenas finales pequeñas). Sin embargo, este método se descartó por no funcionar bien en las partes finales de la secuencia y por presentar variaciones notables de calidad cuando se hace el ajuste por GOPs, como se puede ver en la figura 5.5.



(a) QP y PSNR en *bohemia* (CIF) (b) QP y PSNR en *jamesbond_27* (D1)

Figura 5.5: Utilización del mecanismo de ajuste a niveles de escena y de GOP

5.2. Pruebas sobre la medida de complejidad

Por otro lado, cabe preguntarnos acerca de la calidad y representatividad de la medida de complejidad empleada, $QP \cdot Bits$. Esta medida es ampliamente utilizada porque es sencilla de calcular y sus prestaciones son satisfactorias. Sin embargo, una vez hechos todos los experimentos, podemos decir que se trata de una medida *débil*.

Esta debilidad se ha constatado codificando secuencias a QP constante con tres valores de QP diferentes: 10, 30 y 50. Posteriormente se han dividido las complejidades dos a dos. Si la relación entre QP y $Bits$ fuera lineal saldría, más o menos, una constante. Los resultados se reflejan en las figuras 5.6 (secuencia *starwars*, CIF) y 5.7 (secuencia *elpatriota_78*, D1).

Como se indica en [23, 24], la magnitud $QP \cdot Bits$ es altamente fluctuante, sobre todo debido a la existencia de tres tipos de cuadros (I, P y B), que generan volúmenes de información muy diferentes. Podemos concluir que esta medida sirve para comparar los valores dentro de una misma escena y tipo de cuadro, pero dichos valores no se pueden extrapolar a otros fragmentos de la secuencia y/o otros tipos de cuadro. Dicho de otro modo, si las variaciones son pequeñas y próximas en el tiempo, la medida es fiable; en caso contrario, no, ya que la no linealidad se hace patente.
Además, esta no linealidad conlleva que el valor de QP empleado en la primera pasada influirá

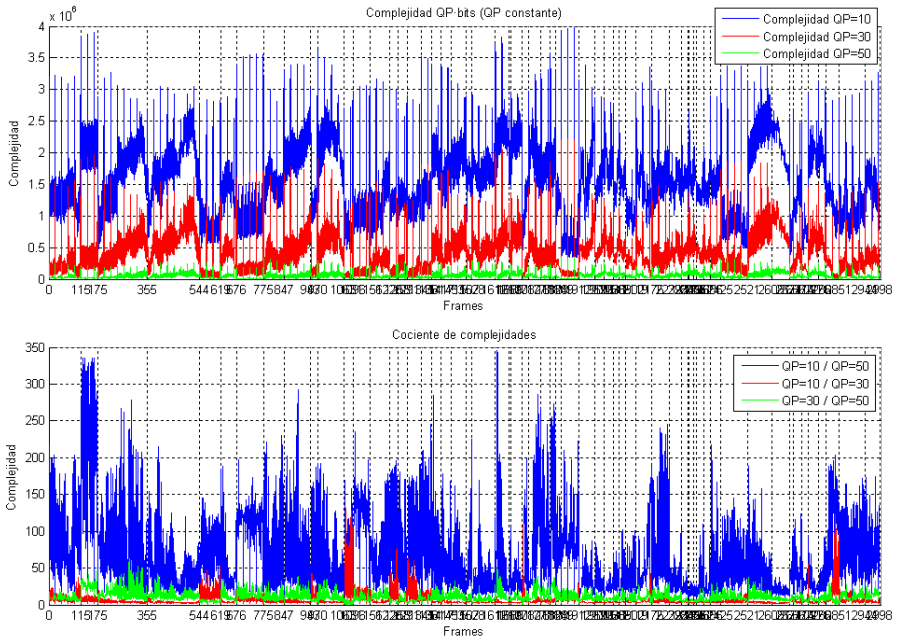


Figura 5.6: Complejidades y cocientes de complejidades en *starwars* (CIF)

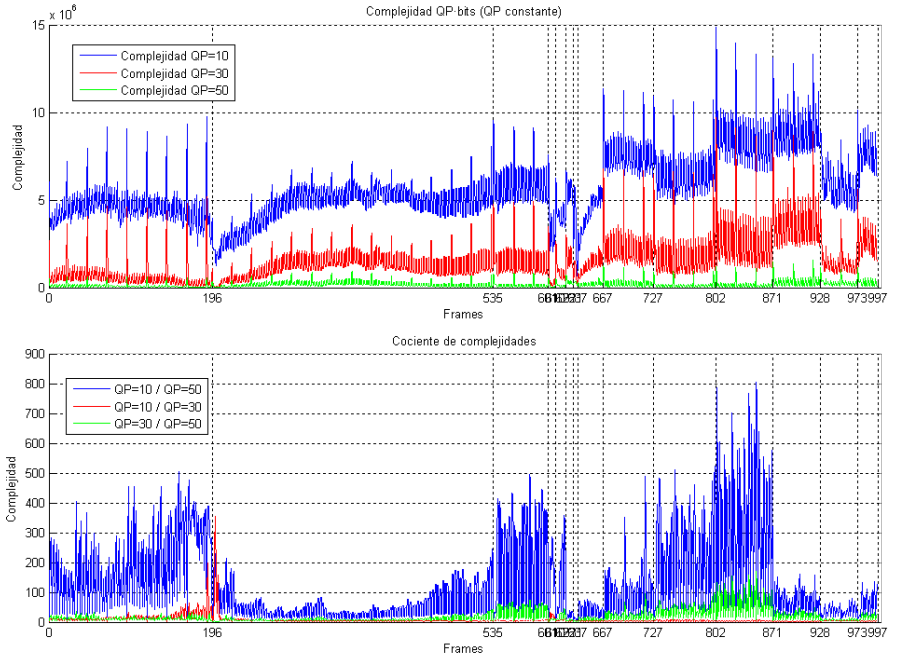


Figura 5.7: Complejidades y cocientes de complejidades en *elpatriota_ 78* (D1)

en los resultados obtenidos en la segunda pasada, pudiendo dar ésta valores muy diferentes según se haya empleado, por ejemplo, QP=10 ó QP=50 en la primera pasada. Esto se puede ver en la tablas 5.5 y 5.6. En ellas podemos observar que, generalmente, los valores de la PSNR media, $\overline{PSNR}$, no difieren en exceso; sin embargo, las diferencias notorias están en la variación de la PSNR (ya sea la desviación típica o la mediana de la varianza local) y en el ajuste de la tasa a la tasa objetivo prefijada.

Secuencia		Bitrate (kbps)	Desviación (%)	$\overline{QP}$	$\overline{PSNR}$ (dB)	σ_{PSNR}	Mediana Var.Local
bohemia	QP_1=10	530.76	3.66	23.0	40.76	2.89	0.145
	QP_1=30	514.44	0.48	23.9	40.68	2.44	0.156
	QP_1=50	515.04	0.59	23.8	40.70	2.52	0.239
starwars	QP_1=10	513.40	0.27	26.7	38.89	1.59	0.124
	QP_1=30	512.75	0.15	27.0	38.85	1.09	0.160
	QP_1=50	510.60	-0.27	27.0	38.86	1.89	0.278
highway	QP_1=10	484.37	-5.40	24.5	39.93	0.48	0.016
	QP_1=30	498.79	-2.58	24.6	40.05	0.45	0.113
	QP_1=50	425.99	-16.80	25.4	39.75	0.60	0.161
mobile	QP_1=10	657.84	28.48	33.7	29.35	1.07	0.241
	QP_1=30	509.80	-0.43	35.2	27.68	0.45	0.057
	QP_1=50	378.21	-26.13	37.7	26.68	0.96	0.182

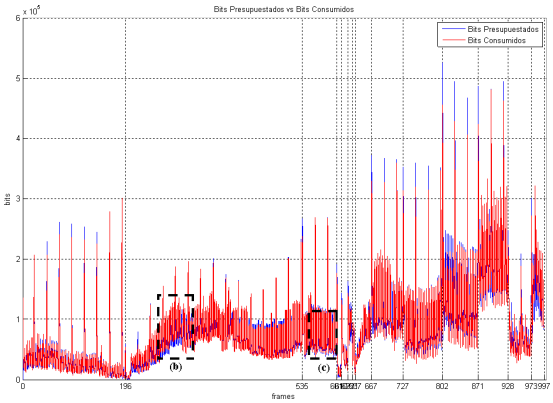
Tabla 5.5: Resultados numéricos con diferentes QPs en la primera pasada (Secuencias CIF)

Secuencia		Bitrate (kbps)	Desviación (%)	$\overline{QP}$	$\overline{PSNR}$ (dB)	σ_{PSNR}	Mediana Var.Local
master_22	QP_1=10	2011.9	-1.76	21.3	45.47	2.80	0.140
	QP_1=30	2012.0	-1.76	22.2	45.18	2.71	0.230
	QP_1=50	2240.5	9.40	28.8	41.31	9.71	0.227
elpatriota_78	QP_1=10	2071.2	1.13	25.8	40.98	2.67	0.214
	QP_1=30	2057.8	0.48	26.3	40.87	2.11	0.318
	QP_1=50	2029.0	-0.93	26.3	40.77	3.47	0.390
jamesbond_27	QP_1=10	2057.3	0.45	26.3	40.60	2.10	0.279
	QP_1=30	2053.5	0.27	26.8	40.45	1.26	0.261
	QP_1=50	2046.8	-0.06	27.0	40.36	1.77	0.418
tempete	QP_1=10	2203.7	7.60	27.3	38.20	0.42	0.066
	QP_1=30	2079.6	1.54	27.7	38.07	0.69	0.208
	QP_1=50	2121.8	3.60	27.6	38.17	0.74	0.321

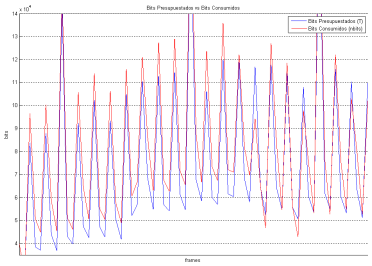
Tabla 5.6: Resultados numéricos con diferentes QPs en la primera pasada (Secuencias D1)

5.3. Pruebas sobre el modelo R-D

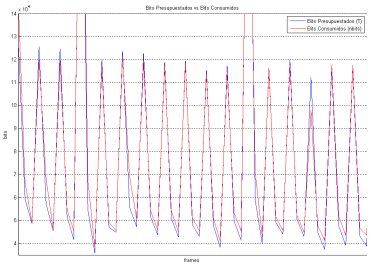
En el algoritmo propuesto se ha introducido un modelo R-D completo, compuesto por un modelado exponencial para los cuadros Intra y uno cuadrático para los P y B (ambos desacoplados). Por lo tanto, es importante confirmar el buen funcionamiento de este modelo, que aunque a vista de los resultados no se comporta mal (al menos desde un punto de vista global), sí hay que ver el comportamiento local y particular de cada tipo de cuadro, correspondiente a un modelado diferente. Para ello recurriremos a dos tipos de gráficas, en las que se mostrarán los bits asignados a cada cuadro (figura 5.8) y la desviación porcentual de cada cuadro con respecto a su presupuesto (figura 5.9).



(a) Secuencia completa. Los cuadros I tienden a gastar de menos.



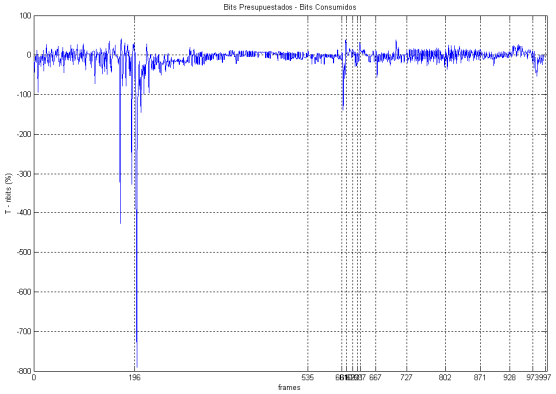
(b) Zona donde el modelo cuadrático tiende a gastar bits de más.



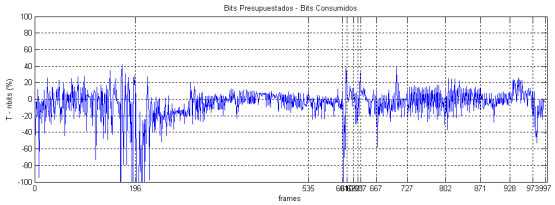
(c) Zona donde el modelo cuadrático se ajusta correctamente.

Figura 5.8: Bits presupuestados y realmente consumidos en *elpatriota_ 78*

Podemos ver que el ajuste de los bits, si bien no es perfecto, sí es muy acertado. Los valores obtenidos se ciñen bastante bien al presupuesto. Los errores cometidos están, salvo excepciones, en torno al $\pm 20\,\%$, oscilando en torno al $0\,\%$, de manera que podemos asumir que los errores cometidos se compensan, como se ve en el hecho de que al final la tasa media es muy próxima al objetivo. Particularizando a los diferentes tipos de cuadro, se puede ver que los cuadros I se ajustan bien a su tasa objetivo, con la salvedad que a veces se quedan cortos de bits. Por el contrario, los cuadros P y B se suelen ceñir bastante al presupuesto, aunque tienden en algunos momentos a consumir bits de más.



(a) Desviación completa

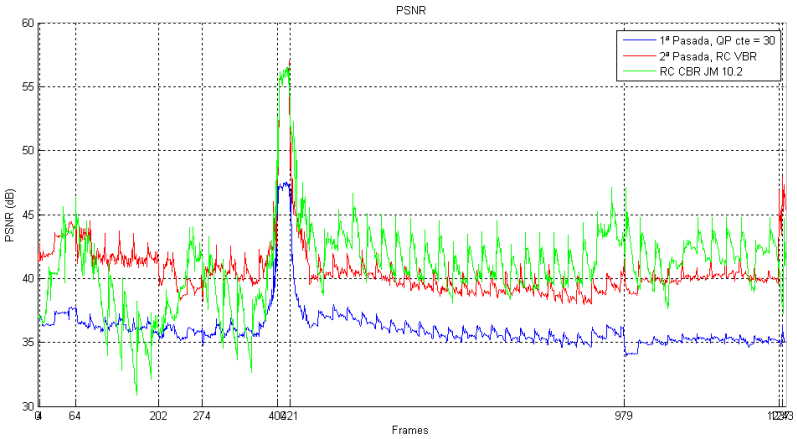


(b) Zoom entre ±100 %

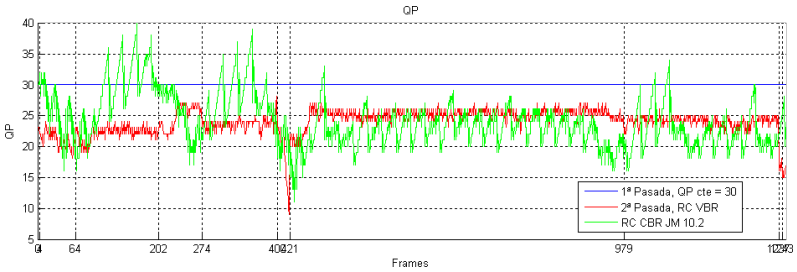
Figura 5.9: Desviación porcentual entre bits presupuestados y consumidos en *elpatriota_78*

5.4. Capacidad de adaptación ante *eventos*

Además del mantenimiento de la calidad en niveles moderadamente estables, es necesario determinar la capacidad que tiene el algoritmo de adaptarse a *eventos*. Estos *eventos* son fragmentos de escena o secuencia en los que el comportamiento difiere significativamente de los cuadros anteriores o incluso del comportamiento general de la secuencia.
De entre todas las secuencias contempladas en las pruebas de este Proyecto, podemos destacar dos con *eventos* significativos: *bohemia* (CIF) (figuras 5.10 y 5.11) y *master_22* (D1) (figuras 5.12 y 5.13). En *bohemia* se corresponde con un fundido negro, con varios cuadros a negro, y *master_22* con un fundido de larga duración. En ambos casos las complejidades en ese intervalo caen a niveles próximos a cero, provocando posibles errores en el bit allocation, como se mencionó en el apartado 4.2.2.1.
El comportamiento en ambos casos es correcto, adaptándose adecuadamente a las variaciones bruscas de complejidad.



(a) PSNR



(b) QP

Figura 5.10: Evolución de PSNR y QP en *bohemia* (CIF)

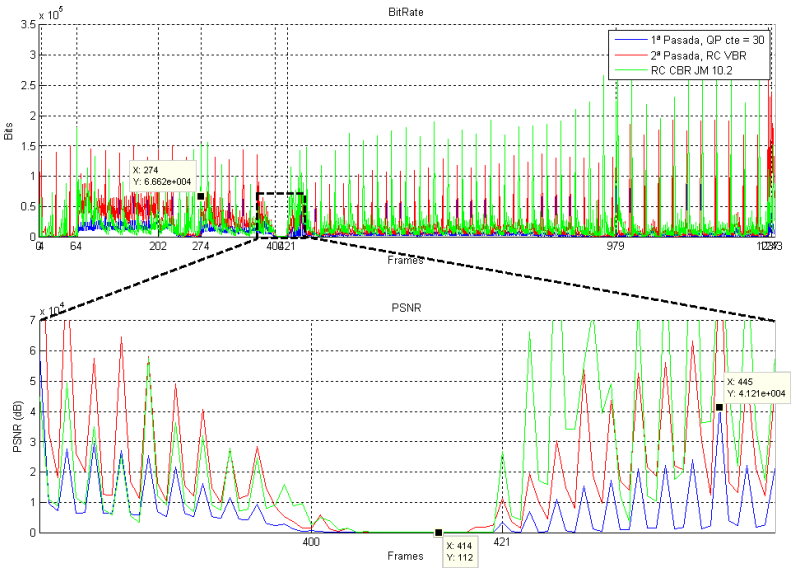
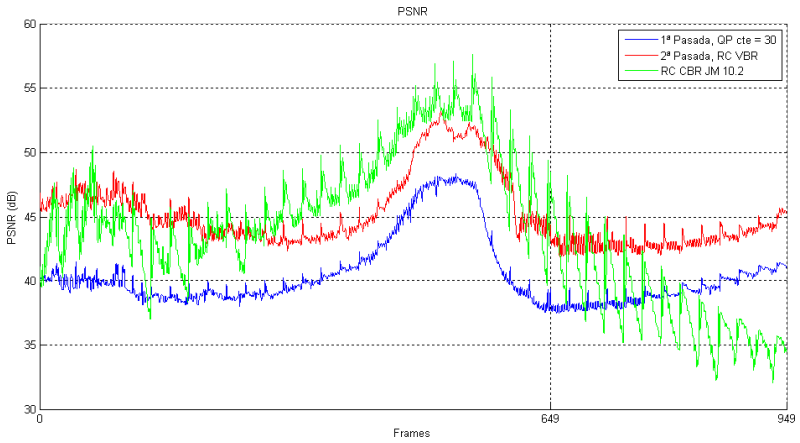
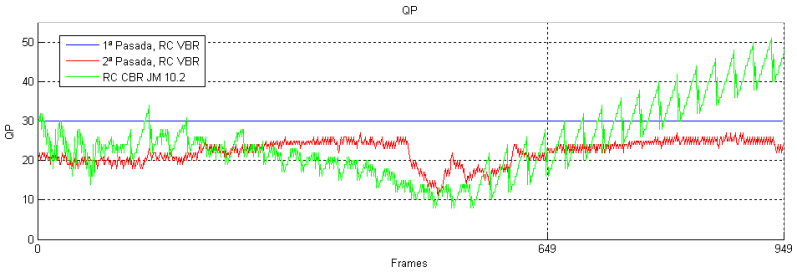


Figura 5.11: Bits empleados en *bohemia* (CIF). Notar las diferencias relativas en los valores.



(a) PSNR



(b) QP

Figura 5.12: Evolución de PSNR y QP en *master_22* (D1)

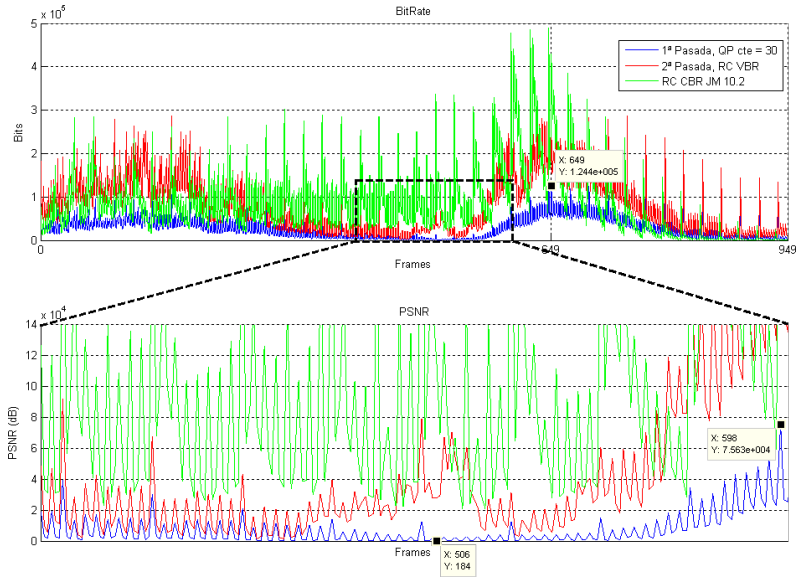
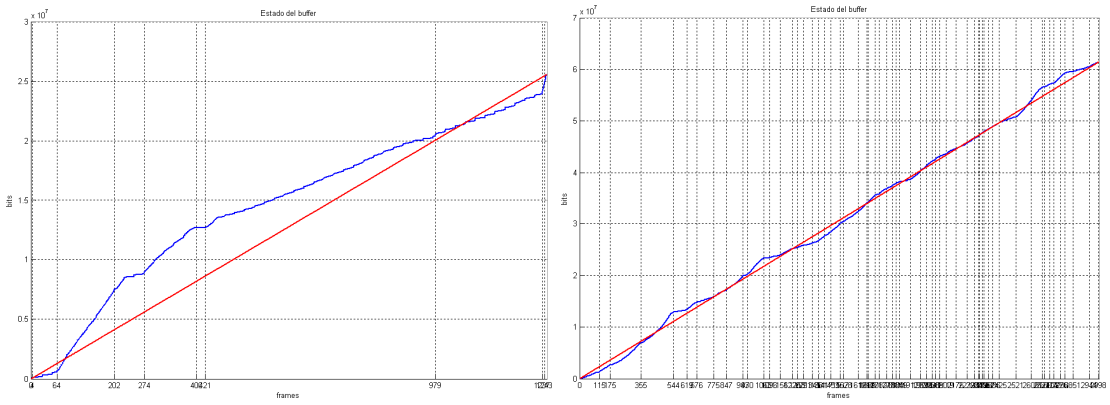


Figura 5.13: Bits empleados en *master_22* (D1). Notar las diferencias relativas en los valores.

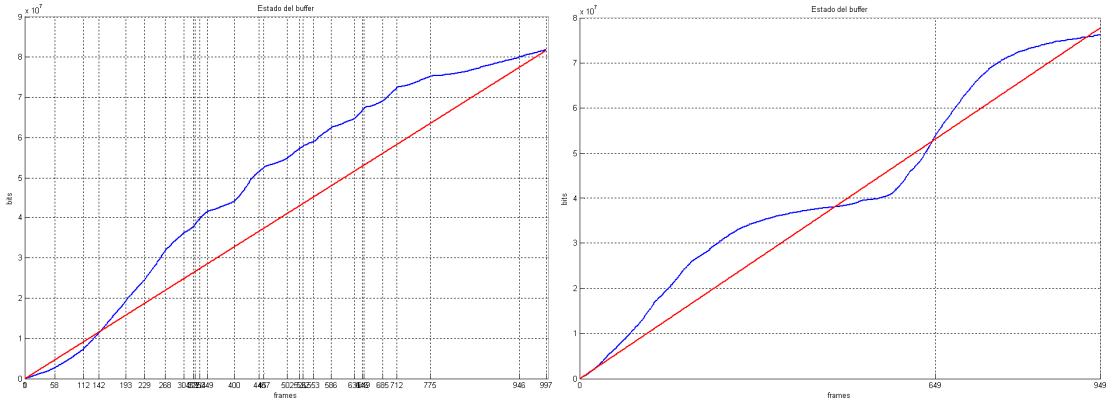
5.5. Estudio de los requerimientos de buffer

Un aspecto importante en la codificación de una secuencia es la evolución del déficit o superávit de los bits empleados frente a los presupuestados, ya que esto nos indica los requerimientos de un hipotético buffer. En el algoritmo propuesto no se ha considerado la existencia de ningún buffer, sin embargo éste es necesario para una correcta implementación, a efectos prácticos, del codificador. El dimensionamiento de este buffer para su uso en un algoritmo VBR como el que estamos tratando es una tarea difícil, ya que depende en gran medida de las características de la secuencia que estamos codificando, sobre todo de la longitud y complejidad de las escenas que la componen. Para poder determinar sus tamaño echaremos mano de unas gráficas (figuras 5.14 y 5.15) que relacionen la producción del codificador con el control de tasa propuesto y la que tendría un flujo CBR con una tasa igual a la tasa objetivo de nuestra propuesta. Con estas gráficas podremos obtener la desviación máxima por defecto y por exceso del codificador a estudio, y con éstas podremos estimar un tamaño adecuado para el buffer a emplear.



(a) *bohemia* (b) *starwars*

Figura 5.14: Evolución del teórico buffer en secuencias CIF



(a) *jamesbond_27* (b) *master_22*

Figura 5.15: Evolución del teórico buffer en secuencias D1

Una vez hecho este experimento, hemos podido ver que las exigencias de búffer de las diferentes secuencias han sido las indicadas en la tabla 5.7.

Secuencia	Máximo Exceso	Máximo Defecto
bohemia	4627840	-1385760
starwars	1852304	-1538680
highway	347384	-2444464
mobile	129368	-197464

(a) Secuencias CIF

Secuencia	Máximo Exceso	Máximo Defecto
elpatriota	336528	-15019088
jamesbond	15105016	-2116168
master_22	9588384	-5562672
tempete	98088	-1276296

(b) Secuencias D1

Tabla 5.7: Valores máximos por exceso/defecto

Vistos estos resultados, podemos ver que para secuencias CIF la máxima exigencia de buffer está en torno a 4.6 Mbit (que vienen a ser unos 0.6 MB), mientras que en secuencias D1 este buffer no debe ser inferior a 15 Mbit (unos 2 MB). Este buffer debe ser llenado antes de la reproducción, lo que dará lugar a un retardo, denominado «tiempo de *buffering*» o «*playout time*», que se situará en unos 9 segundos en secuencias CIF y 7.5 segundos en secuencias D1 (calculado en base a la tasa objetivo y siempre dependiendo del estado de la red). Estos valores son razonables para las características de estas secuencias y perfectamente asumibles por el codificador/decodificador.
Obviamente, en caso de utilizar otras resoluciones, framerates o bit rates objetivo diferentes estos valores de buffer deben ser revisados.

5.6. Comparación con otros algortimos del estado del arte

Una vez vistos el comportamiento y las prestaciones del algoritmo propuesto, resulta inevitable realizar una comparación de las prestaciones de éste con las conseguidas por otros algoritmos existentes en el estado del arte, concretamente con los algoritmos propuestos por Que [22], Huang [26] y Zhang [23, 24]..

Sin embargo, finalmente no ha sido posible realizar una comparación fiable entre estos algoritmos y el propuesto. El principal motivo de esto es que los autores anteriores no especifican completamente la configuración de los codificadores, siendo ésta muy compleja y determinante a la hora de la obtención de resultados. Se han realizado múltiples pruebas con diferentes configuraciones, pero no ha sido posible obtener una configuración fiable. Para determinar esta fiabilidad se ha usado como criterio los resultados obtenidos por los softwares de referencia (JM 9.8 en el caso de Huang y 10.1 en el de Que), suponiendo que todos den resultados similares con la misma secuencia y configuración.

El algoritmo de Zhang tiene el problema añadido de que sus experimentos son únicamente con secuencias HD, las cuales, además, desconocemos. Esto conlleva que no podamos valorar adecuadamente los resultados, ya que no conocemos las características de dichas secuencias (sus complejidades, texturas, movimiento, etc.). Por ello no se ha podido efectuar ninguna comparación cuantitativa con este algoritmo, ya que no disponemos de las secuencias empleadas por el autor. En cualquier caso, la varianza de la PSNR obtenida por el autor resulta muy baja, lo que indica una gran estabilidad en la PSNR y una gran calidad percibida.

Por el contrario, sí hemos podido realizar codificaciones "similares" a las realizadas por Que y Huang. Los resultados han sido los indicados en la tabla y la figura (para el algoritmo de Huang) y la tabla y la figura (algoritmo de Que):

Algoritmo	$\overline{PSNR}(dB)$	$var(PSNR(dB))$	Tasa obtenida (kbps)
JM 9.8	28.85	1.1991	502.74
Huang	28.28	0.0724	499.99
JM 10.2	28.79	1.7904	502.50
Propuesto	28.78	0.2494	550.75

Tabla 5.8: Algortimo de Huang. Secuencia: *Mobile*. Tasa objetivo: 500 kbps

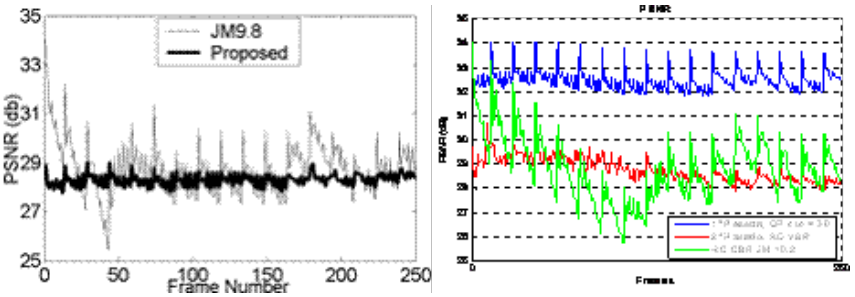


Figura 5.16: Comparación con el algoritmo de Huang

En estos resultados vemos que, si bien no son iguales, sí son comparables, por lo que la configuración del codificador empleada no debe ser muy diferente. Podemos asumir que las prestaciones de nuestro algoritmo son algo peores en estabilidad (aunque en términos absolutos la varianza sigue siendo bastante pequeña, como se puede ver en la gráfica, en la que la variación se ve pequeña), pero mejores en media. Sin embargo, la tasa dista mucho del objetivo (un 10 % más). Esto se puede explicar por el hecho de que esta secuencia está formada por una única escena, por lo que no se compensan los desajustes de tasa.
Es necesario indicar que esta secuencia *mobile* no es igual que la que hemos empleado anteriormente. Es similar, pero no está compuesta por tres escenas concatenedas, sino por una única escena más larga.

La comparación con el algoritmo de Que no resulta fiable, ya que podemos ver que con los algoritmos CBR hay diferencias muy significativas y contradictorias (con algo más de tasa, el JM 10.2 obtiene más de 2 dB en $\overline{PSNR}$ y menos de la mitad de varianza). En cualquier caso, los datos de Que son muy buenos en estabilidad. El algoritmo propuesto falla en la tasa por el mismo motivo que en el algoritmo de Huang.

Algoritmo	$\overline{PSNR(dB)}$	$var(PSNR(dB))$	Tasa obtenida (kbps)
JM 10.1	31.46	4.25	749
Que	31.42	0.17	752
JM 10.2	33.55	1.81	785.22
Propuesto	33.26	0.42	773.76

Tabla 5.9: Algortimo de Que. Secuencia: Stefan. Tasa objetivo: 750 kbps

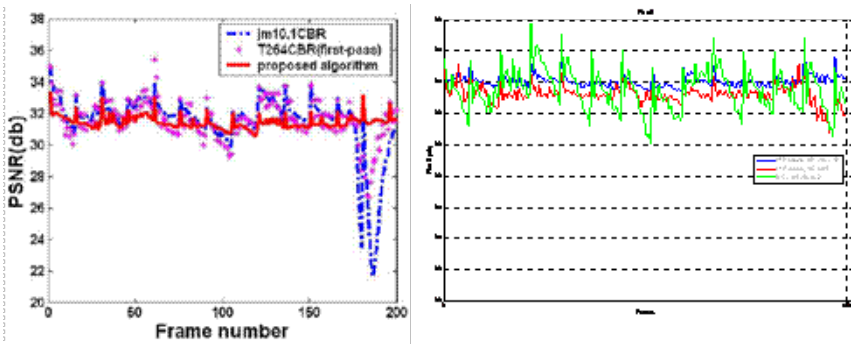


Figura 5.17: Comparación con el algoritmo de Que

Si hacemos una comparación cualitativa, podemos afirmar que el algoritmo propuesto es más completo, ya que además de tener un coste computacional bajo (sobre todo comparado con Que y Huang, que utilizan bucles), presenta detección de cambios de escena de manera fiable (por diferencia de histograma frente a diferencias de complejidades y de porcentaje de macrobloques intra) y un mecanismo de selección de QP inicial, no presente en los demás algoritmos.

Adicionalmente, en algoritmo propuesto tiene la ventaja de ser muy modular, en cuanto a que sus procedimientos (complejidades, bit allocation, modelos R-D de Ps, Bs e Is) son fácilmente separables, sustituibles y modificables, por lo que el algoritmo puede ser mejorado con la mejora de alguno de estos elementos (como las complejidades vistas en en apartado 5.2).

Por otro lado, las pruebas realizadas en estos tres algoritmos resultan escasas, en cuanto a que utilizan un número muy pequeño de secuencias y de una complejidad baja, mientras que el algoritmo propuesto en este proyecto es probado con una gran variedad de secuencias, muy largas, complejas y con un gran número de cambios de escena. Además, en este proyecto se ha empleado una medida adicional, la mediana de la varianza local, que resulta más representativa que la desviación típica o la varianza cuando lo que se quiere medir es la estabilidad de la calidad, ya que esta última puede dar lugar a resultados engañosos.

Capítulo 6

Conclusiones y trabajos futuros

6.1. Conclusiones

La codificación de vídeo es un proceso complejo que involucra numerosos bloques de procesado. Uno de ellos es el bloque de Control de Tasa. Éste constituye una parte vital del codificador, ya que se encarga de controlar el consumo de bits para que la aplicación a la que pertenece el codificador funcione correctamente. Según su naturaleza, esta aplicación impondrá unos requisitos u otros, como puede ser un *bit rate* constante (por ejemplo, una videoconferencia a través de un canal CBR), un requisito de calidad constante (caso de la difusión de vídeo por *streaming*) o una restricción en la cantidad de información generada (como ocurre en almacenamiento de vídeo)[1].

En el presente Proyecto Fin de Carrera se ha diseñado, implementado y validado un mecanismo de control de tasa para codificación de vídeo H.264/AVC que persigue obtener una calidad subjetiva elevada, manteniendo controlado el volumen total de información generada.

Ya que la calidad subjetiva es difícil de analizar, se ha optado por una medida de calidad *objetiva*, la *Peak Signal to Noise Ratio* o PSNR, que permite analizar de manera rápida y sencilla la calidad percibida de una secuencia de vídeo. El objetivo a perseguir será obtener unos valores de PSNR estables.

Debido a la naturaleza inherentemente variante de la información de vídeo, para la obtención de una calidad estable se ha optado por un mecanismo VBR de dos pasadas, que permitirá adaptar el bit rate de cada cuadro a su complejidad teniendo en cuenta las características de la secuencia completa, obtenidas en la primera pasada mediante una codificación con QP constante. Ésta complejidad se ha medido con la magnitud $QP{\cdot}Bits$, sencilla y de fácil manejo. Sin embargo, después de usarla, esta magnitud se ha revelado como una magnitud *débil* por varios motivos:

- Su valor fluctúa mucho dependiendo del tipo de cuadro (I, P, B).
- La medida es imprecisa cuando toma valores bajos.

[1] En este último caso también es deseable una calidad constante.

- La relación QP-Bits no es lineal, lo que hace que los resultados de la segunda pasada varíen con la QP empleada en la primera pasada. Sin embargo, se puede considerar *localmente* lineal en intervalos de QP pequeños.

Por otro lado, se ha tenido en cuenta el hecho de que la información en una secuencia de vídeo no es homogénea, sino que está formada por secciones o escenas, en las que se pueden presuponer unas características similares y la existencia de correlaciones temporales.
Esta consideración es clave para la consecución de una calidad estable, ya que cada escena debe ser tratada de manera independiente de las demás. De este modo no se persigue que la calidad sea estable en toda la secuencia, sino sólo dentro de la escena actual. Para poder tratar con ellas se ha implementado un mecanismo de detección de cambios de escena basado en el cálculo de la diferencia segunda de histograma entre cuadros contiguos y su posterior comparación con un umbral fijo predefinido.
Este esquema no se comporta mal, pero presenta un número de falsas alarmas que se podría reducir mediante un mecanismo de umbral dinámico. Además, este método tiene problemas a la hora de detectar transiciones graduales, ya que en estos casos tiende a, o bien no detectar el cambio de escena o bien detectar muchos cambios de escena seguidos.

Posteriormente se ha diseñado un mecanismo de reparto de bits (o *bit allocation*) multinivel, de manera que el presupuesto de bits asignado a un nivel será proporcional a la relación de complejidades entre este nivel y el nivel superior, teniendo en cuenta el presupuesto de bits de éste último. A diferencia del estándar de control de tasa JVT-G012 [18] (implementado en el software de referencia JM 10.2), en el que este reparto se hacía a nivel de GOP y posteriormente a nivel de cuadro[2], en el algoritmo propuesto se han empleado cuatro niveles: escena, GOP, miniGOP (formado por un cuadro I o P y los n cuadros B siguientes en orden de codificación[3]) y cuadro.
La validez de este mecanismo de reparto depende fundamentalmente de la validez de las complejidades medidas en la primera pasada. Como se ha podido comprobar, estas complejidades presentan inconvenientes y debilidades, por lo que el bit allocation propuesto no resulta todo lo fiable que se desearía.

La selección de QP constituye el siguiente paso del algoritmo. Para ella se han contemplado los tres tipos distintos de cuadro que forman la secuencia (I, P y B).

En cuanto a los cuadros P, se ha conservado el modelo R-D cuadrático presente en el software de referencia, ya que se ha comprobado que sus resultados eran satisfactorios.

Respecto a los cuadros B, se ha implementado el mismo modelo R-D cuadrático aunque desacoplado, esto es, formalmente igual, pero usando otro conjunto de parámetros propio que sólo se actualizarán cuando se codifique otro cuadro B. El hecho de adoptar este modelo viene originado por la similitud entre cuadros P y B (los dos se codifican, fundamentalmente, usando codificación *inter* y en menor medida *intra* y *skip*/*direct*). En este caso el modelo se comporta correctamente ya que, viendo las gráficas, se ciñe de manera bastante precisa al presupuesto de bits asignado.
Sin embargo, si se observa el conjunto de cuadros P y B se puede apreciar un pequeño rizado

[2] También lo hace a nivel de unidad básica, pero no consideramos un nivel tan bajo en este Proyecto.
[3] Donde n es el número de cuadros B consecutivos especificados en el archivo de configuración del codificador (en las pruebas realizadas, n=2).

en las gráficas de QP y de PSNR. Esto puede ser debido a la complejidad empleada, que en el caso de los cuadros B es baja y propensa a errores. Para compensar esto se ha reducido mediante *clipping* la variación del parámetro de cuantificación a ±1 entre cuadros B consecutivos.
Visto esto, podemos concluir que el modelo R-D cuadrático es un modelo adecuado para su uso en cuadros Inter (P y B).

Para los cuadros I se ha incorporado con éxito un modelo R-Q exponencial que emplea información de la tasa objetivo y del contenido del cuadro. Además de funcionar correctamente, éste modelo tiene la gran ventaja de que permite un cálculo bastante acertado de la QP inicial que hay que escoger para empezar a codificar una escena basándose además en la información recogida en la primera pasada.
Para implementar este modelo se ha partido de la propuesta de Zhou [35]. Sin embargo ha habido que modificar un parámetro, β_I, de valor recomendado 0.1, a 0.14 para evitar problemas de estabilidad en la calidad. El valor de este parámetro se ha mostrado de gran importancia y es necesario estudiar su comportamiento en caso de usar otras resoluciones que no sean las estudiadas (CIF o D1).
Para favorecer la convergencia del modelo, el *clipping* en los cuadros I debe permitir una variación mayor, fijándose en nuestro caso a ±5 unidades. Sin embargo, este método puede provocar inestabilidades impredecibles. Para prevenirlas se ha optado por imponer un límite de variación aceptable de QP de ±7 unidades. Si el modelo elige una QP con una variación mayor se considerará que ha ocurrido un error y se tomará la QP del cuadro I anterior.

El control de gasto de bits se realiza según se va avanzando en la codificación de la secuencia. Se monitoriza la diferencia entre los bits presupuestados y los bits gastados en cada cuadro, de manera que al finalizar una escena se tiene un déficit o un superávit de bits. Este desajuste se compensará entre las siguientes escenas.
La corrección a nivel de escena es acertada en cuanto a la constancia de la calidad dentro de una escena, ya que dentro de ella no se hacen correcciones, sino que se sigue siempre el mismo criterio. Sin embargo, esta opción presenta una serie de inconvenientes:

- Si la secuencia está formada por una única escena, la secuencia no tendrá posibilidad de ajuste para ceñirse al presupuesto de bits, confiando todo al buen funcionamiento del conjunto «Bit Allocation + Modelo R-Q»
- Si las escenas presentes son muy grandes es posible que se acumulen déficits/superávits de bits muy grandes y difíciles de compensar.
- Al final de la secuencia las escenas finales tienen que compensar todo el déficit que haya. Si éstas son muy cortas y/o el déficit heredado es elevado, habrá problemas en cuanto que pueden quedarse sin presupuesto (pasando forzosamente a QP=51, con la correspondiente pérdida de calidad) y/o no ser capaces de compensar el desajuste (con el correspondiente error en el volumen total de datos genreados)

Para tratar de paliar estos inconvenientes se ha probado un mecanismo en el que la corrección se realiza a nivel de escena, y en el caso de que falten 3 (o menos) escenas y/o falte por gastar un 33 % del espacio total (o menos) esta corrección pasa a relizarse a nivel de escena y de GOP. Sin embargo, este método se descartó por no funcionar bien en las partes finales de la secuencia y por presentar variaciones notables de calidad cuando se hace el ajuste por GOPs.

En cuanto al funcionamiento global del sistema, en el desarrollo y validación del algoritmo

se han encontrado muchas situaciones de inestabilidad, tanto de QP como de calidad. Estas las variaciones finalmente están limitadas de manera *dura* mediante *clippings*, pero hay que tenerlas en cuenta a la hora de efectuar cualquier modificación necesaria sobre el algoritmo (como la modificación de parámetros debido a cambios en la resolución de entrada, tasa objetivo, etc.).

Acerca de su rendimiento frente a otros algoritmos existentes, la comparación no ha podido ser realizada, al menos de manera fiable, ya que los autores de los algoritmos a comparar no especificaron de manera precisa la configuración del codificador empleada en sus pruebas. De todos modos, el algoritmo propuesto se comporta de manera notable, perdiendo la comparación de estabilidad, pero presentando mejores valores en PSNR media. De todos modos hay que considerar que en los algoritmos estudiados para su comparación las pruebas realizadas sobre ellos eran muy escasas y con secuencias de una complejidad muy baja en comparación a las empleadas en este proyecto para evaluar el algoritmo propuesto.
En cualquier caso, el presente algoritmo es más completo, ya que presenta un mecanismo de detección de cambios de escena y otro de selección de la QP inicial para la secuencia o nueva escena.

Visto todo esto, podemos concluir que se ha diseñado, implementado y validado un algoritmo de control de tasa competitivo con el estado del arte que presenta unas variaciones de calidad a corto plazo moderadas, constituyendo una calidad perceptual bastante estable a lo largo de la secuencia. Todo ello cumpliendo la condición de volumen máximo de información generada o espacio en disco empleado.
Sin embargo, como se ha mencionado, existen ciertas *lagunas* en sus prestaciones, como son los posibles escenarios en el control de gasto de bits, que pueden provocar una salida del codificador no deseada. Del mismo modo, la calidad es *relativamente* estable. Con esto nos referimos a que perceptualmente se comporta bien en cuanto que sus prestaciones son similares al caso de QP constante (variaciones a corto plazo pequeñas), pero *globalmente* (considerando fragmentos grandes de la secuencia) la calidad sí fluctúa. Por otro lado, está por ver el comportamiento de este codificador con secuencias de mayor resolución, ya que seguramente habría que cambiar de valor determinados parámetros.

Para finalizar, podemos orientar el uso de este algoritmo a aplicaciones que necesiten obtener una calidad perceptual elevada, como pueden ser codificadores domésticos, sin irnos a sectores profesionales en los que los requisitos de estabilidad global de la calidad son muy elevados, como puede ser el caso del *mastering* de una determinada secuencia de vídeo.

6.2. Trabajos futuros

A lo largo de este Proyecto Fin de Carrera, se han visto diferentes propuestas y experimentos para realizar un algortimo de Control de Tasa efectivo con el objetivo de obtener una calidad estable. A lo largo del diseño y, sobre todo, la implementación y validación del algoritmo propuesto se han observado algunos aspectos que, o bien suponen una limitación de dicho algoritmo o bien pueden ser mejorados en aras de conseguir una mejora en sus prestaciones. Estas observaciones constituirán una serie de trabajos futuros a realizar sobre el ya hecho en este Proyecto para corregirlo o mejorarlo.

El primero de estos trabajos futuros será la búsqueda y estudio de diferentes medidas de complejidad. El objetivo sería conseguir una medida de calidad más *universal*, que, idealmente, sea independiente del tipo de cuadro y del parámetro de cuantificación empleado en la primera pasada. Partiendo de esta mejor medida, se podrá realizar un bit allocation más adecuado para que la calidad del conjunto de cuadros P y B sea más estable.

Hemos visto que los cambios de escena suponen un elemento crítico en un algoritmo de control de tasa. Sin embargo, el esquema empleado en este Proyecto peca de simplicidad. Sería deseable encontrar un mecanismo de detección más robusto en el que las tasas de pérdida y falsa alarma sean menores. Podría interesar investigar un mecanismo de umbral dinámico, como el sugerido en [31]. Así mismo, en las secuencias de vídeo que se pueden encontrar en la actualidad abundan los cambios de escena mediante transiciones graduales, como pueden ser los fundidos o las cortinillas, en los que el cambio no se realiza abruptamente entre un cuadro y el siguiente. En estos casos, el algoritmo propuesto no funciona bien, pudiendo pasar por alto estas transiciones o bien detectando numerosos cambios de escena en ellas. Un método de control de tasa deseable sería el que identificara estas transiciones, como el expuesto en [37], para posteriormente dividirlas en tres tramos: escena anterior, transición y escena siguiente.

Otro problema observado es el relacionado con las escenas de larga duración. Habría que investigar algún método para compensar las desviaciones de su consumo de bits frente al presupuesto asignado, de manera que éstas no afecten de manera negativa al volumen de información generado por toda la secuencia ni a sus últimas escenas. Una posible manera de hacer esto podría ser la inclusión de un nuevo *nivel de ventana* entre el nivel de escena y el nivel de GOP, de manera que una escena de larga duración se *trocearía* en varias ventanas de longitud suficientemente grande (4-5 GOPs o mayor), las cuales tendrían su propio presupuesto y compensarían sus desajustes entre ellas. Otra posible manera de hacerlo sería que estas ventanas no fueran fijas, sino que se tratara de una ventana deslizante. En este último caso, si los desajustes fueran demasiado grandes, las variaciones de calidad necesarias serían más graduales a lo largo de la escena.

Por último, es necesario recordar que este algoritmo no tiene en cuenta ningún mecanismo de transmisión (buffer, estado de red, etc.). Por este motivo, una tarea importante sería poder adaptar el algoritmo propuesto a su transmisión en redes. Para ello habría que partir de un modelo de tráfico para así poder establecer una dinámica de variación de la tasa posible. Obviamente, este modelo debe ser a largo plazo, es decir, debe ser capaz de predecir comportamientos de la red en un plazo razonablemente largo, ya que lo deseable es que el algoritmo se pueda adaptar

escena a escena (o en un plazo estipulado por una ventana deslizante), no GOP a GOP o cuadro a cuadro, para que no haya diferencias de calidad notables.

APÉNDICES

APÉNDICE A

Presupuesto

En este apartado se detallarán los costes generados en la realización del presente Proyecto. Estos costes se dividirán en costes de material (fungible o no) y costes de personal, tanto de realización del Proyecto como de dirección del mismo.

A.1. Coste de material

Los materiales empleados en este Proyecto y su coste aproximado son los que se detallan a continuación:

- Un lugar de trabajo donde realizar el proyecto, bien sea una oficina o un local. Al coste del local habrá que añadir los costes de luz, agua, calefacción, aire acondicionado y mantenimiento general. El importe de este local se puede estimar en 1200 €/mes. Podemos suponer que este espacio estará compartido compartido por 4 proyectos, por lo que el coste se reduciría a 300 €/mes. Considerando que la duración del proyecto ha sido 12 meses dará un total de 3600 €.
- Conexión a Internet, con un coste de 60.97 €/mes[1]. Ya que esta conexión será para todo el local descrito en el anterior punto y que éste es compartido por 4 proyectos, el importe mensual asignable a este proyecto será de 15.2425 €. Durante los 12 meses de duración del proyecto, el gasto atribuíble a este concepto asciende a 182.91 €.
- Un ordenador personal. Este ordenador se empleará tanto para el aspecto de documentación y programación como para realizar las pruebas necesarias durante el desarrollo del algoritmo. Debido a que estas pruebas requieren de gran capacidad de procesado, el ordenador deberá ser de una potencia considerable. Dicho ordenador se valorará en 1200 €. El presente Proyecto ha requerido un uso de este equipo de manera exclusiva. Sin embargo, a la finalización del Proyecto, el ordenador podrá ser empleado en otros proyectos o tareas.

[1] Coste del Kit ADSL 10Mb de Movistar Empresas y su cuota de línea mensual asociada.

Debido a esto podemos asignar a este proyecto el 33 % del coste total del ordenador, por lo que éste ascenderá a 400 €.

- Un ordenador para pruebas. Como ha sido necesario realizar baterias de pruebas muy intensivas se ha necesitado ocasionalmente de un ordenador adicional similar al anterior. Este ordenador también será utilizado como repositorio de copias de seguridad del Proyecto. Debido a su uso no habitual y al hecho de poder ser compartido con varios proyectos, podemos achacar a este proyecto un 20 % de su valor, quedando en 240 €.
- Licencias de Software. En este proyecto se han empleado varios programas, cuyas licencias son las que se indican a continuación[2]:
  - Microsoft Windows 7 Professional, cuyo coste es de 309.00 €.
  - Matlab R2010a, cuya licencia asciende a 2000 €.
  - Microsoft Visual C++, como parte de la *suite* Microsoft Visual Studio 2008 Standard Edition, con un coste de 329.00 €

  El resto de software empleado ha sido gratuito.
  El importe total de las licencias de software asciende a 2638.00 €. Como este software se reutilizará en otros proyectos, podemos asignar a este proyecto un 33 % del importe total, que ascenderá a 879.34 €.
- Material de papelería. En este apartado se incluye todo el gasto referente a material de oficina, como papel, impresiones, bolígrafos, etc. El importe total se estimará en 60 €.
- Impresión y encuadernado de copias de la memoria, así como grabación del software realizado. Este oncepto podemos valorarlo en 50 €.

Concepto	**Importe**
Lugar de trabajo	3600.00 €
Conexión a Internet	182.91 €
Ordenador personal	400.00 €
Ordenador de pruebas	240.00 €
Licencias de Software	879.34 €
Material de papelería	60.00 €
Impresión y encuadernado	50.00 €
Total Coste de Material	**5412.25 €**

Tabla A.1: Coste de material

Contabilizando todos estos gastos, el importe total clasificable como gastos de material asciende a 5412.25 €.

[2]El importe de dichas licencias es el que figura en las tiendas *on-line* de las diferentes empresas de software a la hora de la realización de esta memoria.

A.2. Coste de personal

Este Proyecto ha sido realizado por dos personas: en ingeniero encargado del Proyecto y el director del mismo.

Para el cálculo del número de horas empleadas hay que señalar que el ritmo de trabajo en este Proyecto no ha sido constante, pero se puede estimar su duración real en 12 meses con una dedicación de 7 horas al día. Si contabilizamos que un mes tiene 20 días laborables esto hace un total de 1680 horas, que se han distribuido según las tareas indicadas en la tabla A.2:

Obtención de documentación y estudio	392 h
Estudio de la plataforma y el software de referencia	140 h
Elaboración del algoritmo	28 h
Implementación + Pruebas (784 h)	
Limpieza de código	126 h
Bit allocation	98 h
Modelo cuadrático para cuadros B	112 h
Modelo exponencial para cuadros I	168 h
Detección de cambios de escena	84 h
Pruebas y ajustes finales	196 h
Generación de documentación	336 h
Total	1680 h

Tabla A.2: Desglose de las diferentes tareas que componen el proyecto.

En cuanto a los honorarios, hay que considerar que hasta 2008 los colegios profesionales proporcionaban unos baremos aplicables para la aplicación de dichos honorarios. Sin embargo, según directivas europeas y a notificación del Ministerio de Economía y Hacienda, se ha suprimido esta publicación de baremos con el fin de favorecer la libertad de los mismos [38]. Si nos atenemos al último dato proporcionado por el Colegio Oficial de Ingenieros de Telecomunicación [39], de 2008, los honorarios de un ingeniero de telecomunicación ascienden a 72 €/hora. Para tratar de actualizar este dato le aplicaremos una subida correspondiente al IPC entre los meses de enero de 2008 y mayo de 2010[3], que según el Instituto Nacional de Estadística [40] ha sido un 3.7 %. Tras aplicar esta corrección, el coste por hora será de 74.66 €.
Contando que el número de horas invertidas es de 1680, los gastos por honorarios del ingeniero realizador del proyecto ascienden a 125428.80 €.

Los honorarios correspondiente a la dirección del Proyecto generalmente se calculan como un 7 % del coste total del Proyecto. Considerando que éste asciende a 130841.05 €, los honorarios correspondientes a dirección de proyecto serán 9158.87 €.

El coste total por gastos de personal asciende a 134587.67 €.

[3] Último dato disponible

Concepto	Importe
Realización de Proyecto	125428.80 €
Dirección de Proyecto	9158.87 €
Total Coste de Personal	**134587.67 €**

Tabla A.3: Coste de personal

A.3. Presupuesto total

Considerando todos los costes anteriormente detallados podemos calcular el coste total de este Proyecto:

Concepto	Importe
Coste de Material	5412.25 €
Coste de Personal	134587.67 €
Subtotal	139999.92 €
IVA (16 %)	22399.99 €
TOTAL	**162399.91 €**

Tabla A.4: Coste total del proyecto

El coste asociado al presente Proyecto asciende a CIENTO SESENTA Y DOS MIL TRESCIENTOS NOVENTA Y NUEVE EUROS CON NOVENTA Y UN CÉNTIMOS.

En Leganés, a 24 de junio de 2010

Fdo.: Miguel Ángel Fuente Rodríguez

APÉNDICE B

Pruebas restantes

En este apéndice detallaremos los experimentos realizados con el conjunto de secuencias no expuestas en el capítulo 5.

B.1. Resultados numéricos

Secuencia		Bitrate (kbps)	Desviación (%)	$\overline{QP}$	$\overline{PSNR}$ (dB)	σ_{PSNR}	Mediana Var.Local
airshow	1ª Pasada	-	-	30	39.79	2.69	0.093
	2ª Pasada	511.57	-0.084	21.2	45.76	2.89	0.195
	CBR	512.87	0.17	21.4	45.62	5.01	1.467
cities	1ª Pasada	-	-	30	36.40	1.10	0.059
	2ª Pasada	510.91	-0.21	24.1	41.02	1.43	0.198
	CBR	512.00	0.0006	24.4	41.22	3.36	0.993
football	1ª Pasada	-	-	30	32.66	0.39	0.047
	2ª Pasada	496.02	-3.12	36.4	28.24	0.45	0.048
	CBR	524.04	2.35	36.9	28.76	1.77	1.101
iceage	1ª Pasada	-	-	30	38.86	1.25	0.058
	2ª Pasada	510.98	-0.20	21.1	45.21	1.53	0.202
	CBR	514.58	0.50	22.2	45.09	3.56	1.026
paris	1ª Pasada	-	-	30	34.99	0.27	0.013
	2ª Pasada	504.85	-1.40	26.5	38.42	0.48	0.057
	CBR	515.41	0.67	26.7	38.86	1.70	0.444

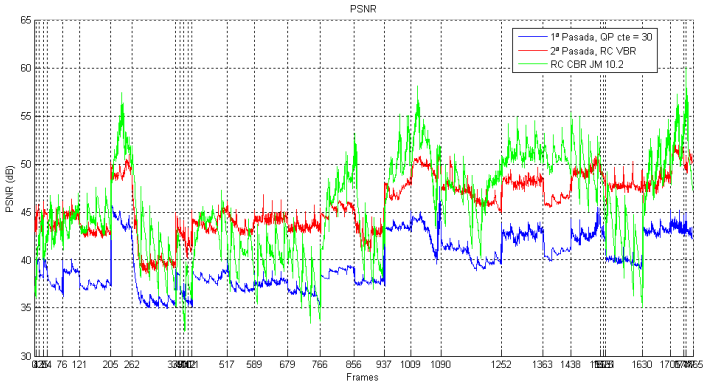
Tabla B.1: Resultados numéricos de las secuencias CIF restantes (Tasa objetivo: 512 kbps)

Secuencia		Bitrate (kbps)	Desviación (%)	$\overline{QP}$	$\overline{PSNR}$ (dB)	σ_{PSNR}	Mediana Var.Local
spiderman_18	1ª Pasada	-	-	30	36.82	1.93	0.432
	2ª Pasada	2047.0	-0.05	28.6	37.75	1.96	0.606
	CBR	2048.5	0.022	29.1	37.62	2.93	1.668
tempete	1ª Pasada	-	-	30	36.28	0.37	0.061
	2ª Pasada	2079.6	1.54	27.7	38.07	0.69	0.208
	CBR	2051.9	0.19	28.1	38.00	1.53	0.720
ultimosamurai_18	1ª Pasada	-	-	30	41.55	2.16	0.115
	2ª Pasada	2041.7	-0.31	18.7	48.11	2.23	0.348
	CBR	2072.1	1.18	19.2	47.94	2.62	1.051

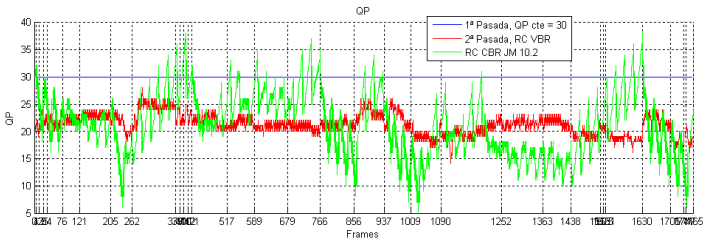
Tabla B.2: Resultados numéricos de las secuencias D1 restantes (Tasa objetivo: 2048 kbps)

B.2. Evolución de PSNR y QP

B.2.1. Secuencias CIF

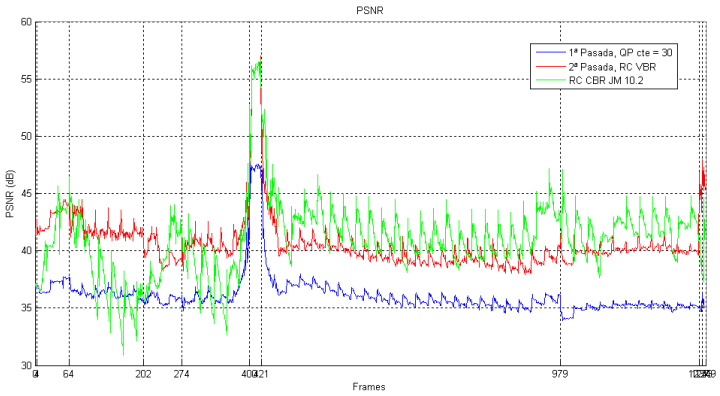


(a) PSNR



(b) QP

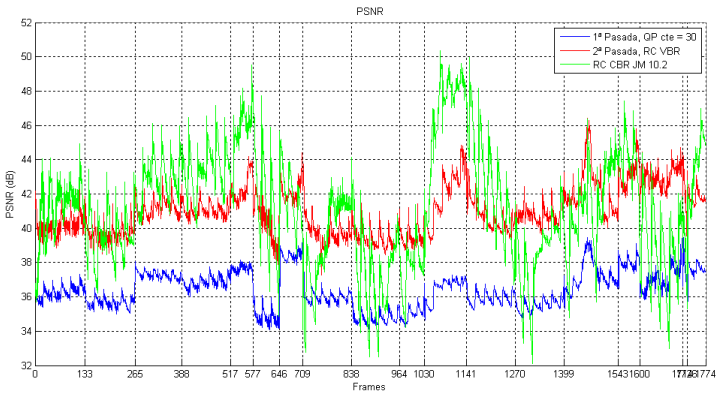
Figura B.1: Evolución de PSNR y QP en *airshow*



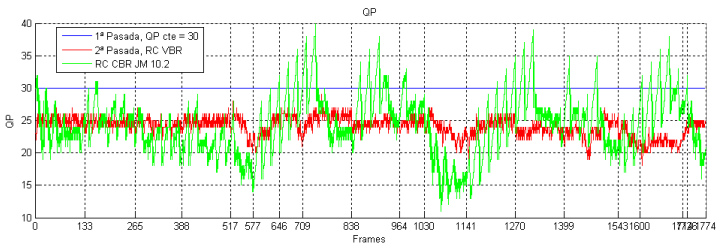
(a) PSNR

(b) QP

Figura B.2: Evolución de PSNR y QP en *bohemia*



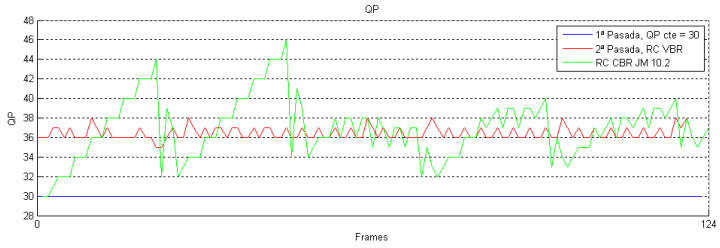
(a) PSNR



(b) QP

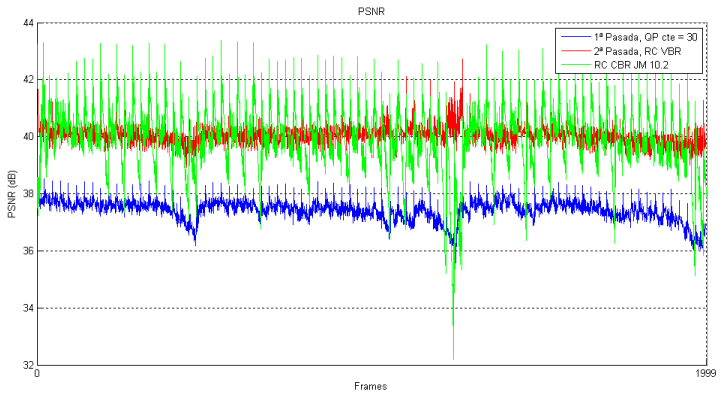
Figura B.3: Evolución de PSNR y QP en *cities*

(a) PSNR

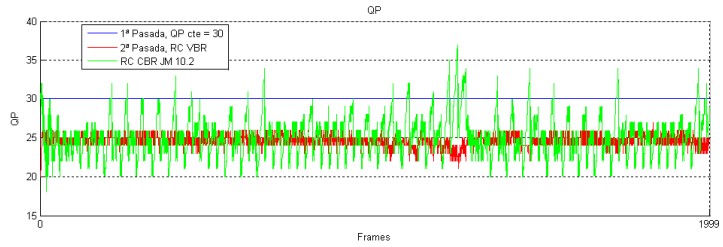


(b) QP

Figura B.4: Evolución de PSNR y QP en *football*

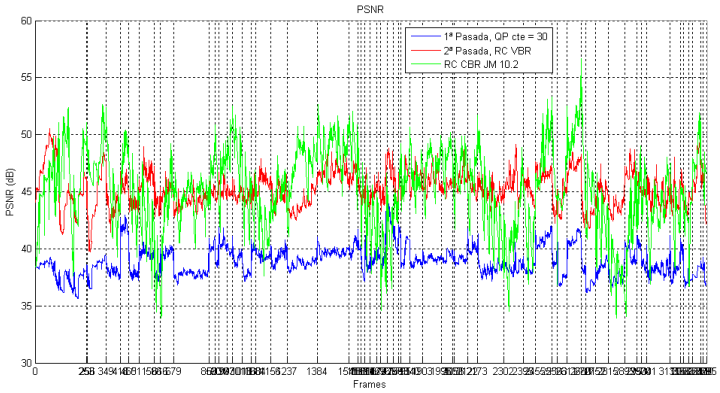


(a) PSNR

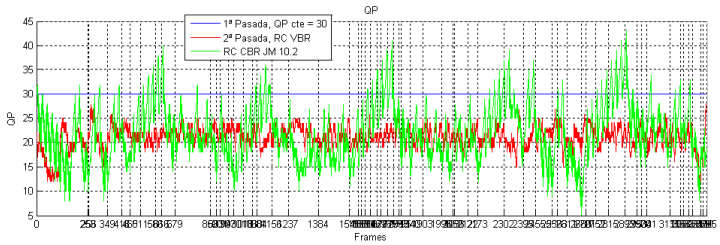


(b) QP

Figura B.5: Evolución de PSNR y QP en *highway*

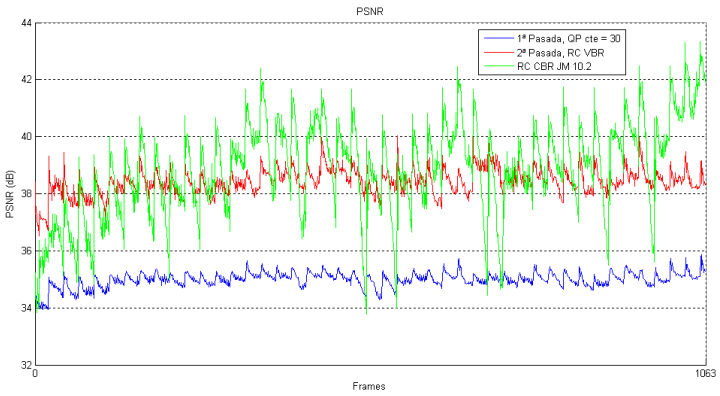


(a) PSNR

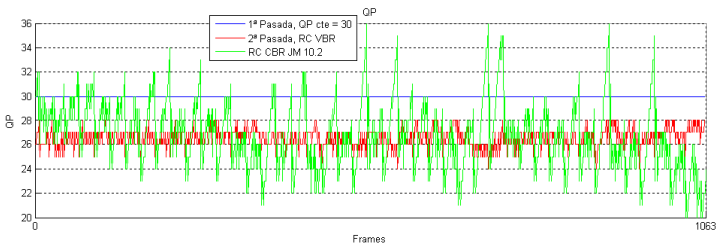


(b) QP

Figura B.6: Evolución de PSNR y QP en *IceAge*



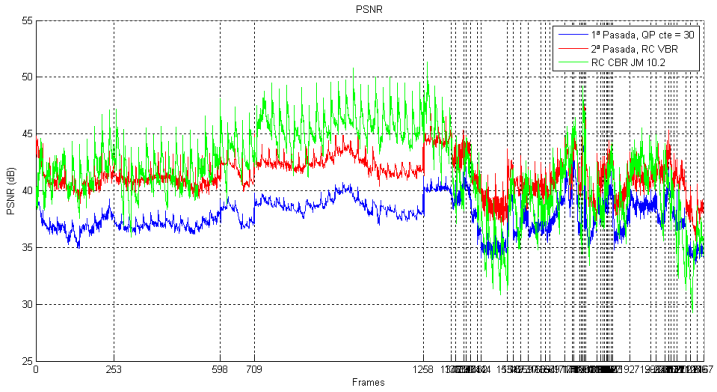
(a) PSNR



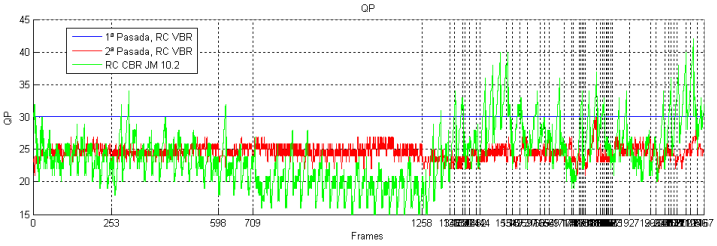
(b) QP

Figura B.7: Evolución de PSNR y QP en *paris*

B.2.2. Secuencias D1

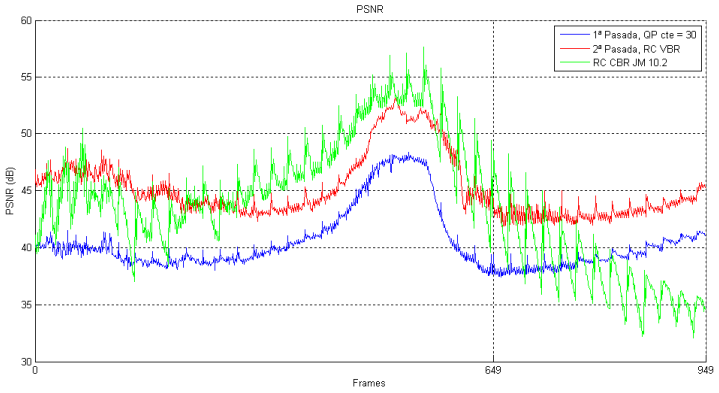


(a) PSNR

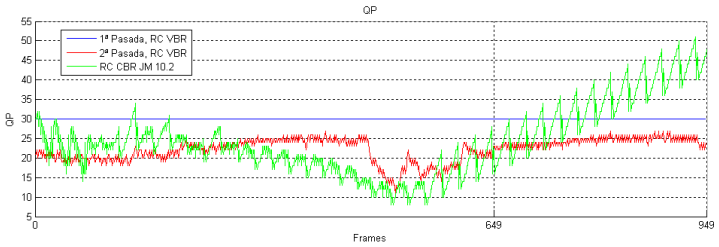


(b) QP

Figura B.8: Evolución de PSNR y QP en *lotr4 + lotr34 + spiderman_18*

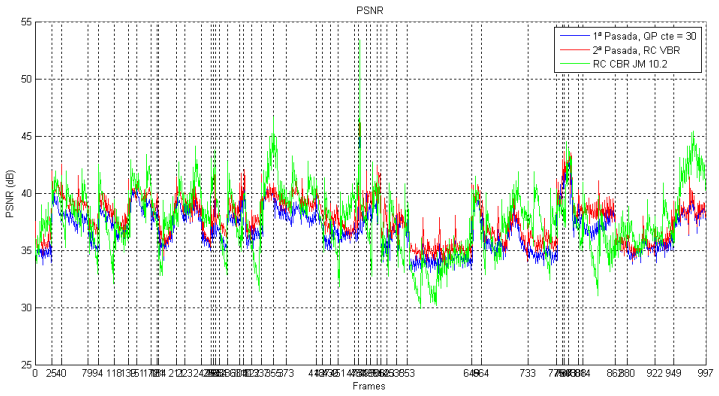


(a) PSNR

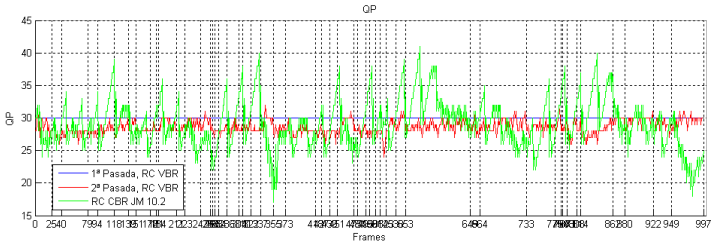


(b) QP

Figura B.9: Evolución de PSNR y QP en *master_ 22*

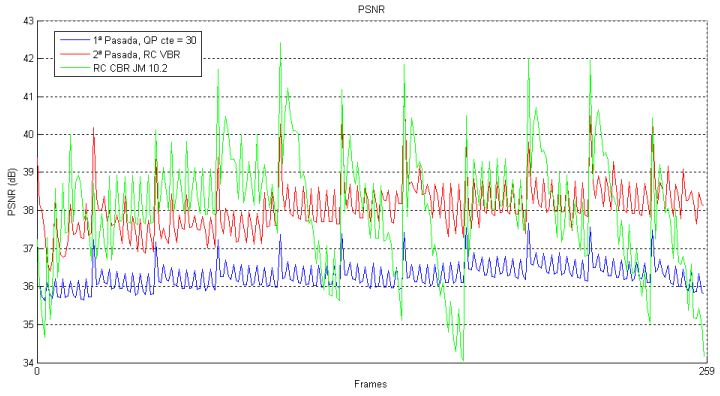


(a) PSNR

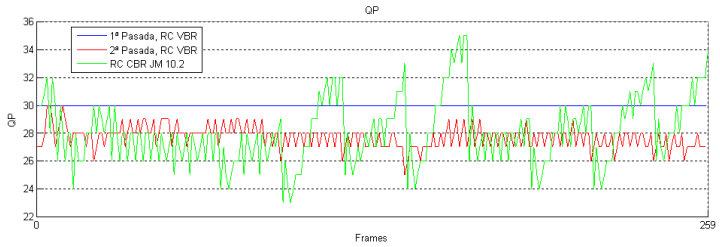


(b) QP

Figura B.10: Evolución de PSNR y QP en *spiderman_ 18*

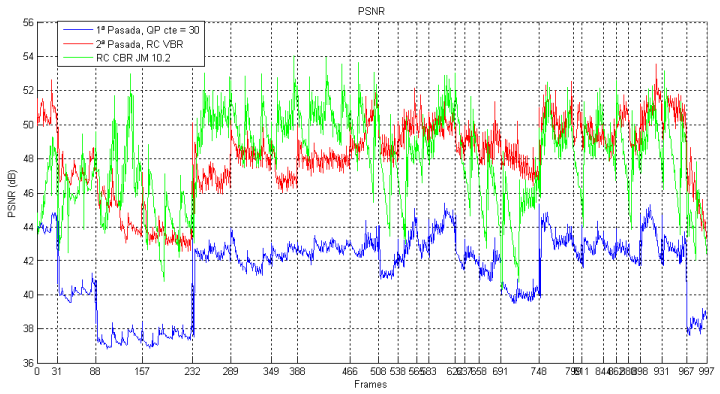


(a) PSNR

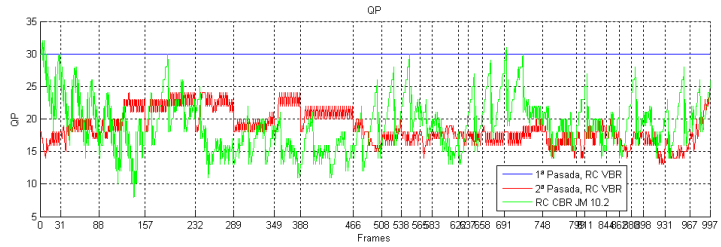


(b) QP

Figura B.11: Evolución de PSNR y QP en *tempete*



(a) PSNR



(b) QP

Figura B.12: Evolución de PSNR y QP en *ultimosamurai_18*

B.3. Necesidades de buffer

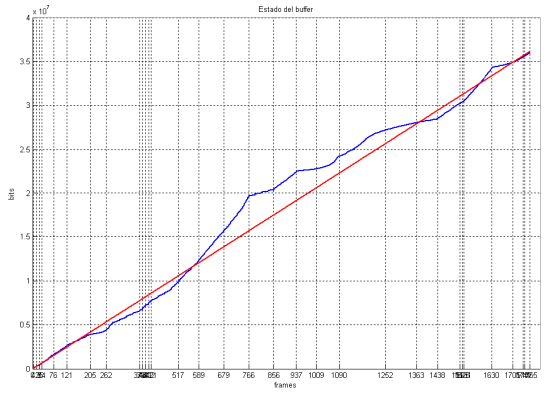
B.3.1. Secuencias CIF

Las desviaciones máximas alcanzadas entre los bits producidos y un flujo a bit rate constante igual a la tasa objetivo (en este caso, 512 kbps) han sido las indicadas en la tabla B.3.

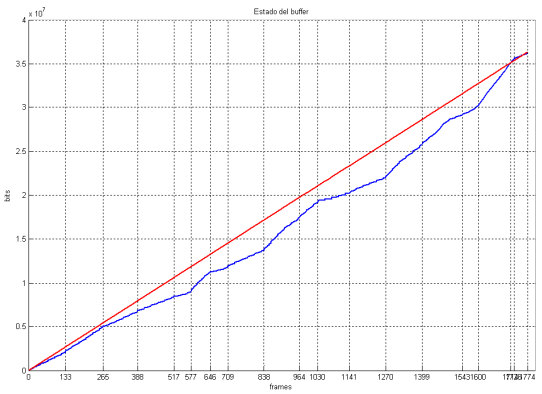
Secuencia	Máximo Exceso	Máximo Defecto
airshow	3977104	-1204232
cities	244472	-3960528
football	18672	-123648
highway	347384	-2444464
IceAge	305424	-3307864
mobile	129368	-197464
paris	893432	-449336

Tabla B.3: Valores máximos por exceso/defecto

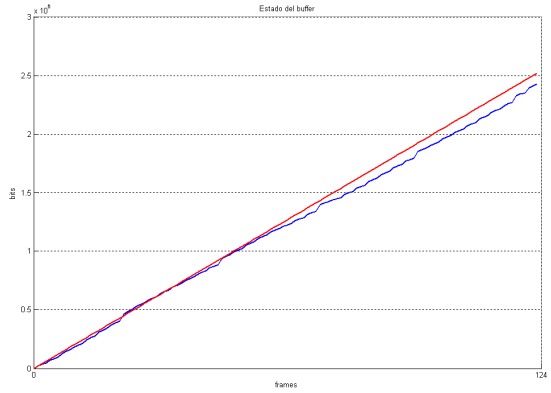
En la figura B.13 se muestran las gráficas con la evolución de la ocupación de un hipotético buffer a la salida del codificador.



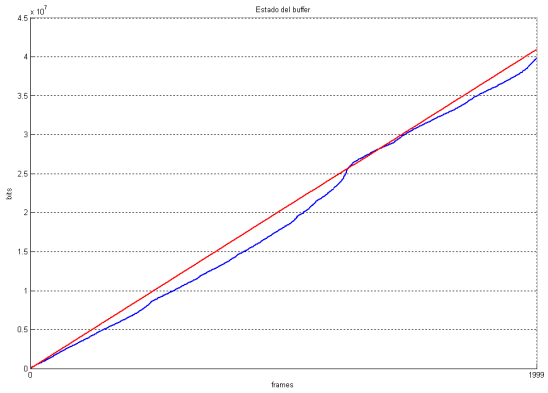
(a) *airshow*



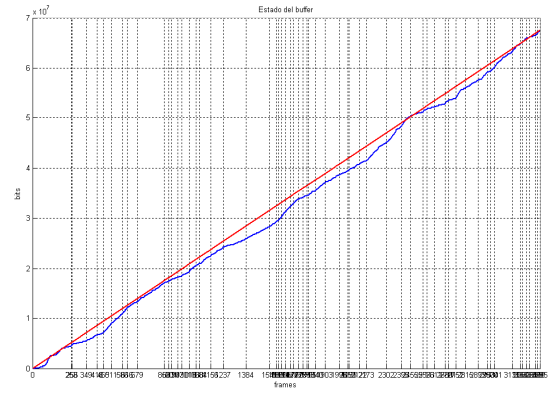
(b) *cities*



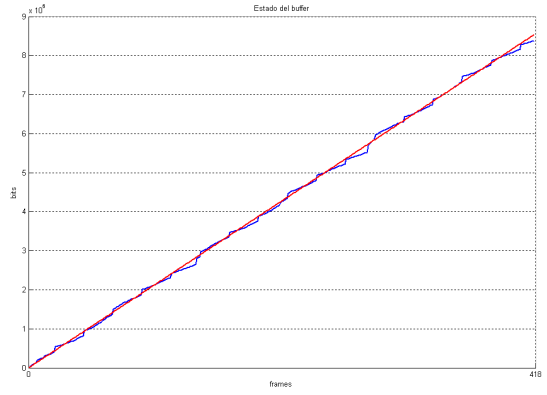
(c) *football*



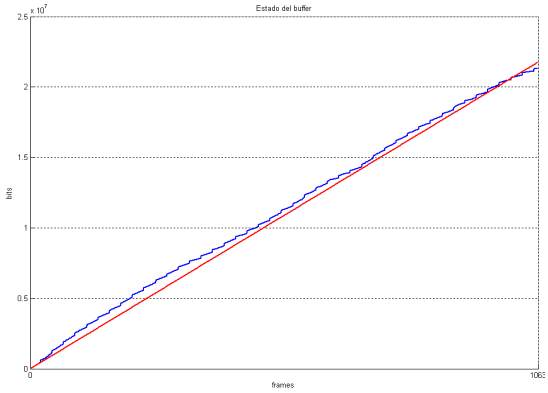
(d) *highway*



(e) *IceAge*



(f) *mobile*



(g) *paris*

Figura B.13: Evolución del teórico buffer en secuencias CIF

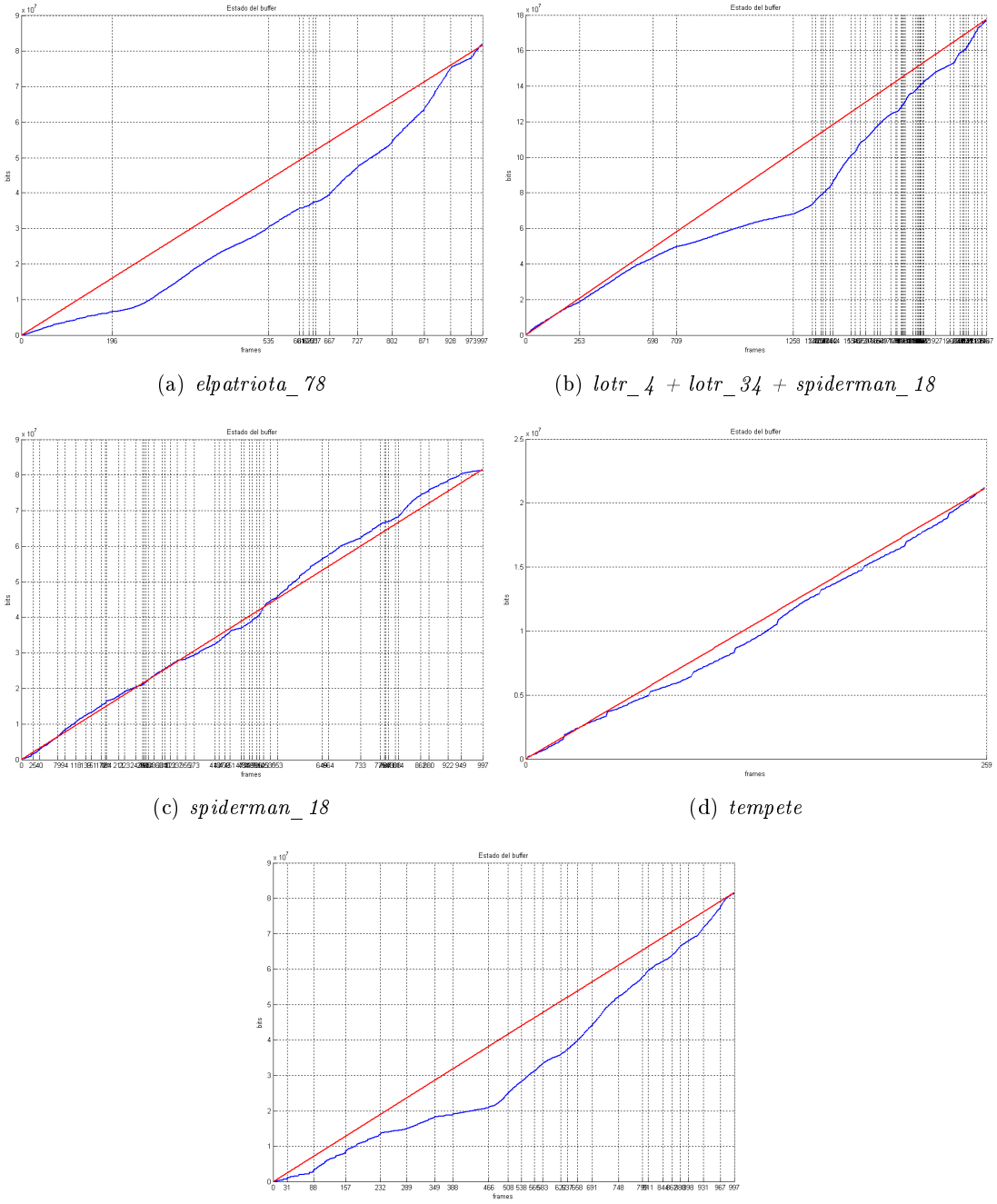
B.3.2. Secuencias D1

En el caso de las secuencias D1, las desviaciones máximas obtenidas se muestran en la tabla B.4, esta vez en base a un flujo CBR igual al bit rate objetivo (2048 kbps).

Secuencia	Máximo Exceso	Máximo Defecto
elpatriota_78	336528	-15019088
lotr_4 + lotr_34 + spiderman18	900712	-37031464
spiderman_18	3748992	-1972936
tempete	98088	-1276296
ultimosamurai_18	0	-17658904

Tabla B.4: Valores máximos por exceso/defecto

En la figura B.14 se muestra la evolución de la ocupación del hipotético buffer a la salida del codificador.



(a) *elpatriota_78*

(b) *lotr_4 + lotr_34 + spiderman_18*

(c) *spiderman_18*

(d) *tempete*

(e) *ultimosamurai_18*

Figura B.14: Evolución del teórico buffer en secuencias D1

Bibliografía

[1] "Itu-t recommendation h.264. advanced video coding for generic audiovisual services," November 2007.

[2] R. Gonzalez and R. Woods, *Digital Image Processing*. Pearson, 3rd ed., 2008.

[3] Z. Li and M. Drew, *Fundamentals of Multimedia*. Pearson, 2004.

[4] I. E. G. Richardson, "H.264 and mpeg-4 video compression," in *H.264 and MPEG-4 Video Compression*, pp. i–xxiv, Wiley, 2003.

[5] G. J. Sullivan, P. N. Topiwala, and A. Luthra, "The h.264/avc advanced video coding standard: overview and introduction to the fidelity range extensions," in *SPIE Conference on Applications of Digital Image Processing XXVII, Denver, Colorado, USA, Aug. 2–6, 2004.* (A. G. Tescher, ed.), vol. 5558, pp. 454–474, SPIE, 2004.

[6] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, and T. Wedi, "Video coding with h.264/avc: tools, performance, and complexity," *Circuits and Systems Magazine, IEEE*, vol. 4, no. 1, pp. 7–28, 2004.

[7] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, and G. Sullivan, "Rate-constrained coder control and comparison of video coding standards," *Circuits and Systems for Video Technology, IEEE Transactions on DOI - 10.1109/TCSVT.2003.815168*, vol. 13, no. 7, pp. 688–703, 2003.

[8] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h.264/avc video coding standard," *Circuits and Systems for Video Technology, IEEE Transactions on DOI - 10.1109/TCSVT.2003.815165*, vol. 13, no. 7, pp. 560–576, 2003.

[9] M. Jiang and N. Ling, "On lagrange multiplier and quantizer adjustment for h.264 frame-layer video rate control," *Circuits and Systems for Video Technology, IEEE Transactions on DOI - 10.1109/TCSVT.2006.873159*, vol. 16, no. 5, pp. 663–669, 2006.

[10] S. Yasakethu, W. Fernando, S. Adedoyin, and A. Kondoz, "A rate control technique for off line h.264/avc video coding using subjective quality of video," *Consumer Electronics, IEEE Transactions on DOI - 10.1109/TCE.2008.4637642*, vol. 54, no. 3, pp. 1465–1472, 2008.

[11] Z. Chen and K. N. Ngan, "Recent advances in rate control for video coding," *Signal Processing: Image Communication*, vol. 22, pp. 19–38, Jan. 2007.

[12] A. Ortega, M.-T. Sun, and A. Reibman, *Compressed video over networks*, ch. Variable bit-rate video coding, pp. 343–382. CRC Press, 2000.

[13] T. Lakshman, A. Ortega, and A. Reibman, "Vbr video: tradeoffs and potentials," *Proceedings of the IEEE*, vol. 86, no. 5, pp. 952–973, 1998.

[14] H. Hamdi, J. Roberts, and P. Rolin, "Rate control for vbr video coders in broad-band networks," *Selected Areas in Communications, IEEE Journal on*, vol. 15, no. 6, pp. 1040–1051, 1997.

[15] N. Kamaci and Y. Altunbasak, "Performance comparison of the emerging h.264 video coding standard with the existing standards," in *Multimedia and Expo, 2003. ICME '03. Proceedings. 2003 International Conference on DOI - 10.1109/ICME.2003.1220925*, vol. 1, pp. I–345–8 vol.1, 2003.

[16] W. Yuan, S. Lin, Y. Zhang, W. Yuan, and H. Luo, "Optimum bit allocation and rate control for h.264/avc," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 16, no. 6, pp. 705–715, 2006.

[17] Y. Yokoyama and Y. Ooi, "A scene-adaptive one-pass variable bit rate video coding method for storage media," in *Image Processing, 1999. ICIP 99. Proceedings. 1999 International Conference on* (Y. Ooi, ed.), vol. 3, pp. 827–831 vol.3, 1999.

[18] Z. Li, F. Pan, Z. Lim, G. Feng, X. Lin, and S. Rahardja, "Adaptive basic unit layer rate control for jvt, jvt-g012," in *7th Meeting, Pattaya II, Thailand*, 2003.

[19] A. Vetro, H. Sun, and Y. Wang, "Mpeg-4 rate control for multiple video objects," *Circuits and Systems for Video Technology, IEEE Transactions on DOI - 10.1109/76.744285*, vol. 9, no. 1, pp. 186–199, 1999.

[20] H.-J. Lee, T. Chiang, and Y.-Q. Zhang, "Scalable rate control for mpeg-4 video," *Circuits and Systems for Video Technology, IEEE Transactions on DOI - 10.1109/76.867926*, vol. 10, no. 6, pp. 878–894, 2000.

[21] S. Shuguang, Y. Shengsheng, and Z. Jingli, "An improved basic-unit layer rate-control scheme on h.264," in *Parallel and Distributed Computing, Applications and Technologies, 2005. PDCAT 2005. Sixth International Conference on DOI - 10.1109/PDCAT.2005.83*, pp. 815–819, 2005.

[22] C. Que, G. Chen, and J. Liu, "An efficient two-pass vbr encoding algorithm for h.264," in *Communications, Circuits and Systems Proceedings, 2006 International Conference on* (G. Chen, ed.), vol. 1, pp. 118–122, 2006.

[23] D. Zhang, Z. Chen, and K. Ngan, "Two-pass rate control for constant quality h.264/avc high definition video coding," in *Picture Coding Symposium, Lisbon, Portugal, November 2007, paper 1106.*, 2007.

[24] D. Zhang, Z. Chen, and K. N. Ngan, "Constant distortion rate control for h.264/avc high definition videos with scene change," in *Circuits and Systems, 2008. ISCAS 2008. IEEE International Symposium on DOI - 10.1109/ISCAS.2008.4542213*, pp. 3498–3501, 2008.

[25] J. Dong and N. Ling, "On model parameter estimation for h.264/avc rate control," in *Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on DOI - 10.1109/ISCAS.2007.378392*, pp. 289–292, 2007.

[26] J. Huang, J. Sun, and W. Gao, "A novel two-pass vbr coding algorithm for the h.264/avc video coder based on a new analytical r-d model," in *Proceedings of the 2007 Picture Coding Symposium*, 2007.

[27] F. De Vito and J. De Martin, "Psnr control for gop-level constant quality in h.264 video coding," in *Signal Processing and Information Technology, 2005. Proceedings of the Fifth IEEE International Symposium on* (J. De Martin, ed.), pp. 612–617, 2005.

[28] Y. Shi, S. Yue, B. Yin, and Y. Huo, "A novel roi-based rate control scheme for h.264," in *Young Computer Scientists, 2008. ICYCS 2008. The 9th International Conference for DOI - 10.1109/ICYCS.2008.174*, pp. 77–81, 2008.

[29] W. Lu, X. Gao, Q. Deng, and T. Wang, "A basic-unit size based adaptive rate control algorithm," in *Image and Graphics, 2007. ICIG 2007. Fourth International Conference on DOI - 10.1109/ICIG.2007.126*, pp. 268–273, 2007.

[30] W. Zeng and W. Gao, "Shot change detection on h.264/avc compressed video," in *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on DOI - 10.1109/ISCAS.2005.1465373*, pp. 3459–3462 Vol. 4, 2005.

[31] A. Dimou and O. Nemethova, "Scene change detection for h.264 using dynamic threshold techniques," in *Proc. of the 5th EURASIP Conference on Speech and Image Processing, Multimedia Communications and Service, Smolenice, Slovac Republic. July 2005*, 2005.

[32] G. G. Lee, H.-Y. Lin, and M.-J. Wang, "Textural complexity-based rate control algorithm," in *Multimedia and Expo, 2008 IEEE International Conference on DOI - 10.1109/ICME.2008.4607633*, pp. 1109–1112, 2008.

[33] S. Kwon, S. Lee, and D. Lee, "Improved initial qp prediction method in h.264/avc," in *Proceedings of the 3rd international conference on Mobile multimedia communications*, (Naffpaktos, Greece), pp. 1–4, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007.

[34] L. Czuni, G. Csaszar, and A. Licsar, "Estimating the optimal quantization parameter in h.264," in *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on* (G. Csaszar, ed.), vol. 4, pp. 330–333, 2006.

[35] Y. Zhou, Y. Sun, Z. Feng, and S. Sun, "New rate-distortion modeling and efficient rate control for h.264/avc video coding," *Signal Processing: Image Communication*, vol. 24, pp. 345–356, May 2009.

[36] http://iphome.hhi.de/suehring/tml/, "H.264/AVC reference software."

[37] I. Muñoz Mejías, D.; González Díaz, "Detección de cambios de escena en codificación e indexación de vídeo," Master's thesis, PFC Universidad Carlos III de Madrid, 2007.

[38] http://www.coitt.es/res/libredocs/Honorarios.pdf.

[39] http://www.coit.es.

[40] http://www.ine.es/daco/ipc.htm.