UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

PROYECTO FIN DE CARRERA

SCALABLE VIDEO ANALYSIS FOR CONTENT BASED ADAPTATION

Andrés Cortés Marlia

Marzo 2009

SCALABLE VIDEO ANALYSIS FOR CONTENT BASED ADAPTATION

AUTOR: Andrés Cortés Marlia
TUTOR: Jesús Bescós Cano
TUTOR QMUL: Ebroul Izquierdo

Grupo de Tratamiento de Imágenes
Dpto. de Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Marzo de 2009

PROYECTO FIN DE CARRERA

Título: *Scalable Video Analysis for Content Based Adaptation*

Autor: D. Andrés Cortés Marlia

Tutor: D. Jesús Bescós Cano

Tutor QMUL: D. Ebroul Izquierdo

Tribunal:

Presidente: José María Martínez Sánchez

Vocal: Kostadin Koroutchev

Vocal secretario: Jesús Bescós Cano

Fecha de lectura:

Calificación:

Resumen:

La gran cantidad de nuevos terminales y redes de acceso han creado un nuevo reto para la distribución de información, y especialmente la distribución de vídeo debido a su gran demanda de tasa binaria. Surge por tanto la necesidad de adaptar el vídeo a los diversos terminales y redes de acceso, teniendo en cuenta las preferencias de usuario y las condiciones de uso.

La solución tradicional al problema de adaptación o es muy costosa (exigiendo decodificar toda la secuencia y recodificarla teniendo en cuenta las restricciones impuestas) o exige múltiples versiones adaptadas almacenadas. Por estos motivos, la codificación escalable de vídeo (Scalable Video Coding, SVC) que permite tener embebidas distintas versiones adaptadas en el *stream* de vídeo es una solución elegante.

El objetivo global de este proyecto es desarrollar e integrar un motor de adaptación (temporal y espacial) para el codec escalable (aceSVC) desarrollado por la Queen Mary University of London (QMUL) en el RC-CAT, y comparar los resultados de adaptación con el estándar MPEG.

Abstract:

The great quantity of new terminals and access networks has created a new challenge for the distribution of information, and specially that of video due to its great demand of binary rate. There arises therefore the need to adapt the video to the diverse terminals and access networks, taking into account the user's preferences and the conditions of use.

The traditional solution to the problem of adaptation either it is very costly computationally (requiring to decode the whole sequence and encoding it again taking into account the imposed restrictions) or it is required multiple adapted stored versions, for what the scalable video coding (SVC) which allows having different adapted versions embedded in the video streaming, it is an elegant solution.

The overall objective of this project is to develop and to integrate an adaptation engine (temporal and spatial) for the scalable codec (aceSVC) developed by Queen Mary University of London (QMUL) in the RC-CAT, and comparing the adaptation results with the MPEG standard results.

Palabras clave:

RC-CAT, CAT, SVC, aceSVC, MPEG, API, codificación, adaptación, *keyframes*, HVSBM, MCTF, DWT, QMUL.

Agradecimientos

Quiero agradecer en primer lugar a mis tutores, Jesús Bescós Cano y Ebroul Izquierdo, la oportunidad brindada y el apoyo para realizar mi Proyecto Fin de Carrera tanto en la Queen Mary University of London como en la Universidad Autónoma de Madrid.

Asimismo, quisiera agradecer a todos los miembros del GTI y del Media and Vision Lab. por el buen ambiente y la ayuda recibida ante los problemas y cuestiones que han surgido durante este tiempo.

A todos los profesores que han contribuido a mi formación como ingeniero.

A mis compañeros de universidad, de Erasmus, de residencia y de trabajo, en especial a Cristina Monsalve, Bárbara Valenciano, Carolina Camacho, Maribel Molina, Marcos Martínez, José Canós, Miguel Cortés, Alejandro Abejón, Ismael Mateos, Jorge Ayllón, Juan Carlos, Fernando García, Alberto Harriero, que me han ayudado a ver las cosas de otro modo y han hecho disfrutar de mi vida universitaria.

Por último gracias a mis padres, mis tíos, mis hermanos, mis abuelos, y a Mónica, que desde el primer día me han ayudado, aconsejado y apoyado en todas mis decisiones y que sin ellos esto no hubiera sido posible.

Andrés Cortés Marlia

Marzo de 2009

INDICE DE CONTENIDOS

1 INTRODUCCIÓN **1**

1.1 MOTIVACIÓN **1**

1.2 OBJETIVOS **2**

1.3 ORGANIZACIÓN DE LA MEMORIA **3**

2 ANTECEDENTES **5**

2.1 CODIFICACIÓN ESCALABLE DE VÍDEO (SVC) **5**

2.2 ACESVC **7**

2.2.1 Codificador ***9***

2.2.2 Bitstream escalable ***15***

2.2.3 Extractor ***18***

2.3 RC-CAT **20**

2.3.1 Introducción ***20***

2.3.2 Visión General de la arquitectura del RC-CAT ***21***

2.3.3 Análisis de Contenido ***22***

2.3.4 Generación de Medios ***25***

2.3.5 Control de adaptación ***27***

3 DISEÑO E IMPLEMENTACIÓN **33**

3.1 GENERACIÓN DE UN VÍDEO ACESVC **33**

3.1.1 Parámetros ***33***

3.2 CAMBIOS EN LA ARQUITECTURA DEL RC-CAT **37**

3.2.1 AceSVC Partial Decoding ***39***

3.2.2 VideoReaderSVC ***43***

3.3 ANÁLISIS DE LA LIBRERÍA MMVSVC **45**

3.3.1 Limitaciones ***47***

3.4 INTEGRACIÓN **50**

3.4.1 Análisis de Contenidos *50*

3.4.2 Adaptación de Contenidos *51*

4 PRUEBAS Y RESULTADOS **53**

4.1 CAMBIOS DE TOMA **54**

4.2 ACTIVIDAD DE MOVIMIENTO **57**

4.3 MOVIMIENTO DE CÁMARA **59**

4.4 ADAPTACIÓN ESPACIAL **62**

4.5 COSTE COMPUTACIONAL **64**

5 CONCLUSIONES Y TRABAJO FUTURO **67**

5.1 CONCLUSIONES **67**

5.2 TRABAJO FUTURO **69**

REFERENCIAS **71**

GLOSARIO **73**

ANEXOS ..... **I**

A MANUAL DE INSTALACIÓN ..... **I**

B FICHERO DE PARÁMETROS DE CODIFICACIÓN EMPLEADOS EN LOS EXPERIMENTOS ..... **V**

INDICE DE FIGURAS

FIGURA 2-1 HERRAMIENTA DEL CODEC DE VÍDEO ESCALABLE ACESVC 9

FIGURA 2-2 ESQUEMA *LIFTING* DE COMPENSACIÓN DE MOVIMIENTO ESPACIO-TEMPORAL. ILUSTRACIÓN EXTRAÍDA DE [13] 11

FIGURA 2-3: CUADROS RESULTANTES DEL PROCESO MCTF. (A) CUADRO PASO BAJO (L) Y (B) CUADRO PASO ALTO (H) 11

FIGURA 2-4 COMPENSACIÓN DE MOVIMIENTO MCTF PARA UN GOP DE 16 USANDO LA TRANSFORMADA HAAR 12

FIGURA 2-5: EJEMPLO DEL CAMPO DE VECTORES RESULTANTE DE APLICAR EL HVSBM 13

FIGURA 2-6 (A) ESQUEMA DE FILTROS DE LA TRANSFORMADA 2D DWT A PARTIR DE UN FILTRO 1D DWT. EJEMPLO DE LA DESCOMPOSICIÓN DIÁDICA EN SUBBANDAS PARA LA IMAGEN LENNA PARA UN NIVEL DE DESCOMPOSICIÓN 2D DWT (A) Y PARA DOS NIVELES (C) 14

FIGURA 2-7 ORGANIZACIÓN DEL *BITSTREAM* DE UN VÍDEO ACESVC 15

FIGURA 2-8 ESCALABILIDAD DE LA RESOLUCIÓN EN EL *BITSTREAM* 16

FIGURA 2-9 ESCALABILIDAD DE LA CALIDAD EN EL *BITSTREAM* 16

FIGURA 2-10 REPRESENTACIÓN EN 3D DEL *BITSTREAM* ESCALABLE 17

FIGURA 2-11 MÓDULOS DEL CODEC ACESVC Y ADAPTACIÓN BASADA EN ACESVC 18

FIGURA 2-12 EJEMPLO DE ADAPTACIÓN MÚLTIPLE 19

FIGURA 2-13 ARQUITECTURA DEL RC-CAT 22

FIGURA 2-14 ARQUITECTURA DEL MÓDULO DE ANÁLISIS DE CONTENIDO 23

FIGURA 2-15 ARQUITECTURA DEL MÓDULO GENERADOR DE MEDIOS 26

FIGURA 2-16 SELECCIÓN JERÁRQUICA DE KEYFRAMES 29

FIGURA 2-17 CRITERIO DE SELECCIÓN DE *KEYFRAMES* BASADO EN LA SUBDIVISIÓN LINEAL DE LA CANTIDAD DE ACTIVIDAD DE MOVIMIENTO ACUMULADA 31

FIGURA 3-1: ARQUITECTURA FINAL DEL RC-CAT CON EL CODEC ACESVC INTEGRADO 38

FIGURA 3-2 ESTRUCTURA DETALLADA DEL CUADRO COMPRIMIDO (CFRAME) 39

FIGURA 3-3 COMPOSICIÓN DEL GOP A DIFERENTES RESOLUCIONES TEMPORALES USANDO PREDICCIÓN BIDIRECCIONAL Y CUATRO CAPAS TEMPORALES 40

FIGURA 3-4 NIVELES DE LOS VECTORES DE MOVIMIENTO EN ACESVC PARA UN GOP DE 8 CUADROS. M_K^I CORRESPONDE CON LA DISTANCIA I^{TH} EN CUADROS CON RESPECTO A LA REFERENCIA DEL NIVEL DE ESCALA TEMPORAL K^{TH}. 42

FIGURA 3-5 ESTRUCTURA TÍPICA DE UN GOP DE 12 CUADROS EN MPEG 42

FIGURA 3-6 DIAGRAMA DE BLOQUES DEL PROCESO DE ADAPTACIÓN DE UNA SECUENCIA 44

FIGURA 3-7 (A) ESTRUCTURA DEL BUFFER DE ALMACENAMIENTO DE LOS VECTORES DE MOVIMIENTO (DONDE M ES EL NÚMERO DE CUADROS PASO ALTO EN EL GOP). (B) ORDEN DE LOS BLOQUES EN UN CUADRO. 45

FIGURA 3-8 VECTORES DE MOVIMIENTO DEVUELTO POR LA LIBRERÍA MMVSVC EN UN CUADRO COMPUESTO POR 16X16 BLOQUES. 47

FIGURA 4-1 SELECCIÓN DE SEGMENTOS DE SECUENCIA QUE GENERAN FALSOS POSITIVOS EN EL DETECTOR DE TOMAS. 56

FIGURA 4-2 RANGO Y ACTIVIDAD DE MOVIMIENTO PARA EL CODEC MPEG 58

FIGURA 4-3 RANGO Y ACTIVIDAD DE MOVIMIENTO PARA LA SECUENCIA TENIS USANDO DIFERENTES RESOLUCIONES TEMPORALES DEL CODEC ACESVC 58

FIGURA 4-4 (A) PATRONES DE MOVIMIENTO DE CÁMARA EXTRAÍDOS MANUALMENTE (*GROUND-TRUTH*) PARA LA SECUENCIA TENIS1.MPG. (B) RESULTADOS DE LOS PATRONES DE MOVIMIENTO OBTENIDOS DEL ALGORITMO DE MOVIMIENTO DE CÁMARA INTEGRADO EN EL RC-CAT. 60

FIGURA 4-5 (A) PATRONES DE MOVIMIENTO DE CÁMARA EXTRAÍDOS MANUALMENTE (*GROUND-TRUTH*) PARA LA SECUENCIA TENIS1.MPG. (B) RESULTADO DE LOS PATRONES OBTENIDOS EN LA SECUENCIA DE TEST USANDO EL ALGORITMO DE MOVIMIENTO DE CÁMARA PARA EL CASO DE ACESVC 61

FIGURA 4-6 CUADROS ORIGINALES DE LA SECUENCIA TENIS -(A), (E), (I), (M), (Q)- Y LOS RESULTANTES DE APLICAR EL ALGORITMO DE SEGMENTACIÓN DE MOVIMIENTO. LOS CASOS (B), (F), (J), (N) Y (R) SON EL RESULTADO OBTENIDO SOBRE MPEG, MIENTRAS QUE (C), (G), (K), (O) Y (S) ESTÁN REFERIDOS A LAS MÁSCARAS OBTENIDOS PARA ACESVC AL MISMO NIVEL TEMPORAL QUE LA SECUENCIA TEMPORAL (25 CUADROS POR SEGUNDO), Y (D), (H), (L), (P) Y (T) ESTÁN REFERIDOS A UN NIVEL TEMPORAL INFERIOR (12.5 CUADROS POR SEGUNDO). 63

FIGURA 4-7 TIEMPO MEDIO DE DECODIFICACIÓN DE UN GOP A DIFERENTES ESCALAS ESPACIALES: EL EJE IZQUIERDO INDICA EL TIEMPO MEDIO POR GOP EMPLEADO EN LA LECTURA DEL *BITSTREAM* Y LA DECODIFICACIÓN DE LOS VECTORES DE MOVIMIENTO, Y EL DERECHO INDICA EL TIEMPO PARA LA DECODIFICACIÓN EZBC. 65

FIGURA A-1 DIRECTORIO PRINCIPAL DE LA HERRAMIENTA RC-CAT I

FIGURA A-2 FICHERO DE CONFIGURACIÓN PARA ADAPTAR LAS SECUENCIAS II

FIGURA A-3 DIRECTORIO RAÍZ DEL CD III

INDICE DE TABLAS

TABLA 2-1 TIPOS DE TRANSFORMADAS *WAVELET* IMPLEMENTADAS EN EL CODEC ACESVC 10

TABLA 2-2 RELACIÓN DE LAS CARACTERÍSTICAS BÁSICAS PARA MPEG Y ACESVC..... 24

TABLA 2-3 RESUMEN DE LOS CRITERIOS EMPLEADOS PARA GUIAR LA SELECCIÓN DE *KEYFRAMES* EN EL NIVEL 3 30

TABLA 3-1: PARÁMETROS DE CONFIGURACIÓN DE UN VÍDEO ACESVC 34

TABLA 3-2: FILTROS *WAVELETS* IMPLEMENTADOS EN LA LIBRERÍA 36

TABLA 3-3: PUNTOS DE DECODIFICACIÓN (*BIT RATES* [KBPS]) PARA DIFERENTES RESOLUCIONES TEMPORALES (T) Y ESPACIALES (S) 37

TABLA 3-4 CUADROS DESCOMPRIMIDOS A LA RESOLUCIÓN TEMPORAL MÁS ALTA DE UN GOP DE 8 CUADROS PARA LAS DOS MANERAS DE CODIFICACIÓN DE LA SECUENCIA DE ENTRADA 41

TABLA 3-5 PARÁMETROS NECESARIOS PARA LA ADAPTACIÓN DE UNA SECUENCIA 44

TABLA 3-6 INFORMACIÓN PROPORCIONADA USANDO EL API DE ACESVC A DIFERENTES ESCALAS TEMPORALES USANDO UN GOP DE 8. 48

TABLA 4-1 VERSIONES DE CODEC ACESVC UTILIZADAS EN LOS EXPERIMENTOS. LAS TRES VERSIONES NECESARIAS PARA ADAPTAR LA SECUENCIA ESTÁN REPRESENTADAS POR UN *. EL RESTO DE VERSIONES, MARCADAS POR UN +, ÚNICAMENTE SE HAN EMPLEADO PARA TESTAR EL COMPORTAMIENTO DE LOS ALGORITMOS EN DISTINTAS SITUACIONES. 53

TABLA 4-2 COMPARACIÓN DE LOS RESULTADOS OBTENIDOS EN LOS EXPERIMENTOS DE LA SECUENCIA NEWS ENTRE EL ESTÁNDAR MPEG Y ACESVC 55

TABLA 4-3 RESULTADO DE LOS EXPERIMENTOS DE LA SECUENCIA NEWS 55

TABLA 4-4 UMBRALES (T1 A T4) PARA LAS DIFERENTES VERSIONES DE LA SECUENCIA TENIS CALCULADOS A PARTIR DEL DESCRIPTOR DE ACTIVIDAD DE MOVIMIENTO DE MPEG-7..... 57

TABLA 4-5 TIEMPO TOTAL DE ADAPTACIÓN PARA MPEG Y DIFERENTES VERSIONES DE ACESVC DE UN SEGMENTO DE 540 CUADROS DE LA SECUENCIA NEWS. 66

1 Introducción

Este capítulo presenta la necesidad que ha llevado a la codificación escalable al punto de mira para solucionar los problemas actuales de los codificadores de vídeo en cuanto a la adaptabilidad de las secuencias de vídeo, así como los objetivos y el enfoque sobre la estructuración y la organización de la memoria del proyecto.

1.1 Motivación

Actualmente existen muchas maneras de tener acceso al contenido multimedia, tantas como diferentes y heterogéneos terminales y redes que pueden ser usados por los distintos usuarios. La adaptación es un campo de aplicación clave para poder llevar el contenido de los servidores a los usuarios finales, cada uno de los cuales usa sus propios terminales y redes, con sus propias capacidades y restricciones (en resolución, ancho de banda, colores, etc.).

La solución tradicional al problema de adaptación consiste en decodificar el contenido y volverlo a codificar teniendo en cuenta las restricciones del usuario final, o bien consiste en almacenar las distintas versiones del mismo contenido. En el primer caso, el coste computacional es elevado; en el segundo, lo es la capacidad de almacenamiento requerida.

La codificación escalable de vídeo (*Scalable Video Coding*, SVC), es una solución elegante al problema de la adaptación. Un *stream* de vídeo escalable contiene embebidas distintas versiones de la fuente (vídeo original), con mínima redundancia, que pueden ser decodificadas en diferentes resoluciones, tasas de cuadro y calidades, simplemente seleccionando las partes adecuadas de dicho *bitstream* [1]. Así, la codificación de vídeo escalable permite una adaptación muy simple, rápida y flexible a una gran variedad de terminales y redes, con diferentes capacidades y características.

La adaptación también puede estar basada en contenido, por ejemplo eliminando los cuadros menos interesantes o reduciendo la calidad sólo en las zonas menos relevantes. En definitiva, la técnica consiste en utilizar información semántica extraída previamente del análisis del contenido para realizar una mejor adaptación [2].

Las técnicas de análisis de secuencias de vídeo son especialmente costosas. Sin embargo, el análisis directo sobre el vídeo comprimido o codificado ofrece la eficacia requerida para permitir soluciones rápidas, ya que evita la decodificación completa del contenido multimedia y permite disponer de parámetros de análisis (por ejemplo, de vectores de movimiento) presentes en el dominio comprimido.

Basándonos en las ventajas que ofrece la codificación escalable en cuanto a la adaptabilidad, lo que se presenta en este pfc es el análisis y la posterior integración del codificador escalable aceSVC dentro de la herramienta de análisis RC-CAT, la cual nos permite adaptar las secuencias de vídeo según las exigencias del usuario final.

Finalmente, se presenta una comparación de los resultados de los algoritmos presentes en el RC-CAT obtenidos entre el codec aceSVC y un codificador no escalable, en nuestro caso MPEG-1/2.

1.2 Objetivos

Para que el RC-CAT sea capaz de realizar la adaptabilidad requerida, es necesario además de conocer las exigencias del usuario (tamaño de la pantalla del terminal, *bitrate*, calidad,…), obtener una serie de características semánticas, como son por ejemplo los cambios de toma y la determinación de los cuadros clave de la secuencia. Estas características se determinan a partir del análisis de las características básicas del dominio comprimido (imágenes de baja resolución, vectores de movimiento e información de textura).

Actualmente los motores de adaptación de SVC (como el desarrollado por el *Media and Vision Lab.* de la *Queen Mary University of London* (QMUL) -aceSVC–, descrito en [3][6][7][8][9][10] y brevemente en la sección 2.2) son simples extractores de las partes requeridas del *bitstream*, con independencia del contenido de la secuencia codificada.

El objetivo global de este proyecto es desarrollar un motor de adaptación para el codec aceSVC que tenga en cuenta el contenido de la secuencia, para lo cual es necesario inferirlo a partir de técnicas de análisis eficientes. Estas técnicas se basarán en la extracción directa de parámetros del dominio comprimido.

Más concretamente, el primer objetivo consiste en especificar y diseñar un API para poder acceder de un modo transparente a las características del dominio comprimido *wavelet*-SVC (información de color y vectores de movimiento) necesarias para aplicar los algoritmos que actualmente operan en el RC-CAT (detector de tomas, actividad de movimiento, movimiento de cámara y segmentación gruesa del movimiento) y así obtener información adicional, información semántica, para la adaptación (máscaras de segmentación de objetos y cuadros clave [4]).

El segundo objetivo está centrado en el desarrollo de un motor de adaptación que tenga en cuenta los resultados del análisis de las características del dominio comprimido, es decir, información de carácter semántico, para adaptar tanto temporalmente como espacialmente las secuencia de entrada en función de las preferencias del usuario final.

Más en detalle, las tareas identificadas para lograr los objetivos citados son:

1. Estudio del estado del arte y de las herramientas existentes
    - Estudio del estado del arte en codificación escalable y del funcionamiento del *wavelet*-SVC, así como de las técnicas básicas de análisis sobre dominios transformados o comprimidos
    - Familiarización con el marco de trabajo y con las herramientas del entorno de codificación de QMUL
    - Estudio de los algoritmos para análisis temporal

2. Desarrollo del interfaz de programación (API) con el dominio *wavelet*-SVC.
    - Análisis del actual API
    - Diseño del API necesario para extraer las características requeridas para el análisis del dominio comprimido, tanto espacialmente como temporalmente
3. Diseño y desarrollo de una aplicación del API
    - Implementación y pruebas de algunos de los algoritmos existentes para extracción de características temporales (cambios de toma o plano, actividad de movimiento, etc.…)
    - Diseño de un extractor de datos del *bitstream* que utilice la información resultante del análisis temporal y el API de acceso al dominio transformado para llevar a cabo una adaptación basada en características semánticas
    - Evaluación de los resultados

1.3 Organización de la memoria

La memoria consta de los siguientes capítulos:

- **Capítulo 1.** Introducción, objetivos y motivación del proyecto
- **Capítulo 2.** Antecedentes, breve descripción de las características de un codificador escalable, del codec aceSVC y de la herramienta RC-CAT
- **Capítulo 3.** Descripción del diseño y la implementación del sistema afectada por las limitaciones encontradas en la librería mmvsvc
- **Capítulo 4.** Resultados obtenidos con las pruebas realizadas al sistema completo que se ha descrito
- **Capítulo 5.** Conclusiones obtenidas tras el desarrollo del sistema. Relación de posibles líneas futuras de desarrollo y propuesta de mejoras del sistema
- **Anexo A.** Breve manual con instrucciones de instalación del software utilizado por el sistema
- **Anexo B.** Información del fichero de parámetros de codificación empleados en los experimentos

2 Antecedentes

Para entender las herramientas con las que se cuenta en este proyecto, este capítulo se centra en una breve revisión de los conceptos generales de los codificadores escalables, así como de las herramientas bajo estudio de este pfc, el codificador escalable aceSVC y la herramienta de análisis de contenido, el RC-CAT.

Se partirá de una introducción general sobre la codificación escalable de vídeo que permita definir las características y ventajas que se van a manejar durante el pfc y a continuación se presentará el codificador aceSVC y la herramienta de adaptación, el RC-CAT.

2.1 Codificación escalable de vídeo (SVC)

La finalidad de un algoritmo de compresión de vídeo digital es eliminar la redundancia espacial y temporal de una secuencia de vídeo de forma que se pueda representar con una distorsión visual aceptable y un ratio de compresión elevado.

En un sistema de comunicación de vídeo tradicional, el codificador comprime la señal de vídeo de entrada a un *bit-rate* que es menor que, y cercano a, la capacidad del canal; además, el decodificador reconstruye la señal de vídeo usando todos los bits recibidos por el canal. Por lo tanto, en este modelo, el codificador conoce la capacidad del canal y el decodificador es capaz de decodificar todos los bits recibidos a una velocidad suficiente como para reconstruir el vídeo.

Estas dos suposiciones no tienen por qué cumplirse en aplicaciones en tiempo real como el *streaming* de vídeo (*streaming media* si incluye sonido) [5]. El *streaming* de vídeo es una técnica para la transmisión de vídeo por Internet que permite la visualización del mismo mientras es recibido, por lo que no es necesario descargarlo completamente antes de reproducirlo. Los datos se transmiten de forma que pueden ser procesados como un flujo continuo, sin pausa; si el usuario recibe los datos con mayor velocidad que la requerida para la reproducción, necesita guardar el exceso en un buffer. Si los datos no llegan lo suficientemente rápido la visualización será de mala calidad.

Para la transmisión de vídeos sobre redes de transmisión heterogéneas, como Internet, el sistema de codificación de vídeo necesita proveer el *bitstream* al cual debe ser decodificado para reconstruir la señal de vídeo con una calidad óptima a varios *bit-rates*, resoluciones espaciales y tasas de cuadro. En este contexto, sólo los sistemas de codificación de vídeo escalable pueden cumplir estos requerimientos.

En definitiva, la escalabilidad de un vídeo se refiere a un formato de codificación que permite una única codificación del flujo de vídeo, permitiendo manipular los *bitstreams* codificados y extraer múltiples representaciones de la misma fuente por medio de la decodificación de una parte o de todo el *bitstream*.

El vídeo que se transmite en las aplicaciones en tiempo real ha sido comprimido previamente. En el momento de la codificación, normalmente no se conoce ni la capacidad del canal ni los recursos computacionales del receptor.

En general, estas aplicaciones se enfrentan a la difícil tarea de proveer vídeo a diferentes niveles de resolución temporal, resolución espacial, calidad y/o *bit-rate*. Estos parámetros dependen de la capacidad de cada receptor, de las opciones preferidas por los usuarios y de la capacidad variable del canal.

Una solución a este problema consiste en comprimir y almacenar cada secuencia de vídeo a distintos *bit-rates*. De esta forma, el servidor será capaz de entregar el vídeo en la forma requerida. Esta solución tiene dos problemas principales:

- Introduce una gran sobrecarga de almacenamiento
- No es viable para aplicaciones en tiempo real por la sobrecarga computacional del codificador

Una solución alternativa consiste en utilizar un tipo de codificación que permita al receptor seleccionar dinámicamente estos parámetros. La secuencia de vídeo se codificada una vez y luego puede ser decodificada a distintos:

- *Bit-rates*
- Cuadros por segundo
- Calidades de imagen
- Resoluciones espaciales y temporales

Ésta es una solución muy atractiva por la flexibilidad que ofrece y se denomina escalabilidad de vídeo.

Podemos clasificar los tipos de escalabilidad de vídeo en los siguientes:

- Escalabilidad de *bit-rate*. El receptor puede pedir un *bit-rate* de datos determinado, elegido de un conjunto o de un intervalo continuo
- Escalabilidad temporal. El receptor puede elegir cuántos cuadros se mostrarán por segundo
- Escalabilidad espacial. El receptor puede requerir que las imágenes estén a una resolución determinada
- Escalabilidad SNR. Para unas determinadas resoluciones espacial y temporal, el receptor puede requerir un nivel de calidad determinado, calculado como el grado de fidelidad entre la imagen original y la decodificada. La calidad se mide normalmente con la métrica SNR (Signal-to-Noise Ratio)
- Escalabilidad basada en objetos: Permite añadir, manipular, eliminar objetos durante el proceso de codificación de la secuencia de vídeo o después del haber generado el vídeo comprimido

Por lo tanto, un codificador escalable será aquel que reciba como entrada un vídeo digital y produzca como salida un vídeo comprimido que posea una o varias de las características de escalabilidad antes mencionadas.

La ventaja de poder elegir diferentes combinaciones de las características de escalabilidad es crucial para la distribución simultánea a diferentes usuarios finales. En este caso, los usuarios con bajo rendimiento tienen la oportunidad de recibir la secuencia de vídeo a una resolución espacial y/o tasa de cuadro menor sobre el canal de acceso de baja capacidad. Del mismo modo, usuarios con alto rendimiento puede acceder a resoluciones espaciales y/o tasas de cuadro y/o calidad superiores de la secuencia de vídeo.

Las características anteriormente mencionadas se centran en la escalabilidad del *bitstream* después de la codificación Aunque éste es el caso más deseable y más comúnmente aplicable, en general, el proceso de de escalabilidad puede llevarse a cabo en una o una combinación de las siguientes tres etapas:

(i) Durante el proceso de codificación por parte del codificador.

(ii) Después de la codificación por medio de un analizador de la red de transmisión.

(iii) Durante la decodificación por parte del decodificador. Este es el caso del RC-CAT.

En la primera etapa (i), el codificador necesita conocer a priori las necesidades de cada usuario final particular antes de codificar el vídeo. El codificador de vídeo puede orientar un determinado *bit-rate* y/o una resolución temporal y espacial determinada en el momento de la codificación para satisfacer las especificaciones requeridas por el decodificador.

Esta aproximación es ideal para transmisiones punto a punto (*unicast*) donde el codificador sirve a cada usuario final (decodificador) independientemente. El inconveniente es la necesidad de codificar y transmitir tantas veces como distintos decodificadores haya, lo que conlleva a la congestión de la red.

En el segundo escenario (ii), el codificador proporciona un *bitstream* escalable sin ningún conocimiento previo del usuario final. El *bitstream* constará de varias versiones de la secuencia de vídeo original, que pueden ser reordenadas o desechadas por un analizador en la red, el cual proporcionará el *bitstream* adaptado para cada una de las diversas y diferentes necesidades de los usuarios finales.

Finalmente en el tercer escenario de vídeo escalado (iii), la escalabilidad es realizada en el lado del decodificador. El decodificador recibe todo el *bitstream* pero tiene la oportunidad de decidir sólo la parte del *bitstream* que cumpla con sus requerimientos. Se trata por tanto, de una reducción de la resolución y/o *bit-rate* que pueda ser motivada por una serie de factores en el decodificador, como la limitación de la potencia de procesado de la *CPU* y la resolución de la pantalla del terminal.

2.2 aceSVC

El objetivo principal de la codificación de vídeo ha consistido tradicionalmente en la optimización de la calidad del mismo a un *bit-rate* determinado. Sin embargo, esto ha cambiado gracias a la aparición de aplicaciones de vídeo en redes de computadores tales como el *streaming* de vídeo por Internet debido a que en el momento de la codificación del vídeo normalmente no se conoce ni la capacidad del receptor ni el ancho de banda disponible para la transmisión.

El continuo aumento del ancho de banda ha llevado al contenido multimedia a la vanguardia del interés del consumidor, con el fin de conseguir enviar dicho contenido a cualquier lugar y tiempo con un alto grado de adaptabilidad. La amplia gama de sistemas de distribución de contenido de vídeo con conexiones de alta y baja velocidad y con diferentes tamaños de pantalla de dispositivos (móvil, PDA, cine,…), han puesto de manifiesto un problema de adaptabilidad.

La solución para este tipo de aplicaciones a tiempo real consiste en realizar una codificación escalable del vídeo, es decir, que el vídeo comprimido resultado de la codificación tenga propiedades de escalabilidad.

En base a ello, el *Media and Vision Lab.* de la *Queen Mary University of London* (QMUL), desarrolló un software que permitía una codificación escalable de vídeo (denominado aceSVC) basado en la transformada *wavelet*, como núcleo compresor (al contrario que MPEG-4 que se basa en la transformada DCT), útil para este tipo de aplicaciones que requieren una adaptación eficiente del vídeo comprimido.

Este codificador escalable permite tres tipos diferentes de escalabilidades, que pueden ser combinados para adaptarse al entorno deseado:

- Escalabilidad de la resolución (reduciéndola por un factor múltiplo de 2)
- Escalabilidad temporal (reduciéndola por un factor múltiplo de 2)
- Escalabilidad de la calidad / SNR

Tal como muestra la Figura 2-1 dicho software está dividido en tres partes: un codificador, un extractor que permite seleccionar la parte deseada del *bitstream* en función de la aplicación a utilizar o de las preferencias del usuario, y un decodificador.

En este caso, la secuencia de vídeo original con resolución 4CIF (704x576 píxeles) a 60 cuadros por segundo es codificada generando un *bitstream* que contiene múltiples versiones adaptadas de la secuencia original (4CIF, CIF y QCIF a distintas resoluciones temporales -60, 30, 15 y 7.5 cuadros por segundo- y calidades). Por lo tanto, seleccionando las partes adecuadas del *bitstream* se puede decodificar la versión adaptada de la secuencia de entrada que se adecue a los requisitos del dispositivo del usuario final.

Por ejemplo, si consideramos un escenario de difusión de televisión donde la red está limitada a 4 Mbps y el dispositivo de televisión presenta una pantalla de 704x576 y una tasa de cuadros de 30 fps, por medio del extractor se extraerá la versión “4CIF, 30fps, 4Mbit/s”.
Si consideramos que el usuario accede al mismo contenido usando una PDA con una pantalla de 352x288 a 30 fps a través de una red UMTS donde el máximo bitrate es de 1 Mbps, se seleccionará la versión adaptada “CIF, 30fps, 1Mbit/s”.

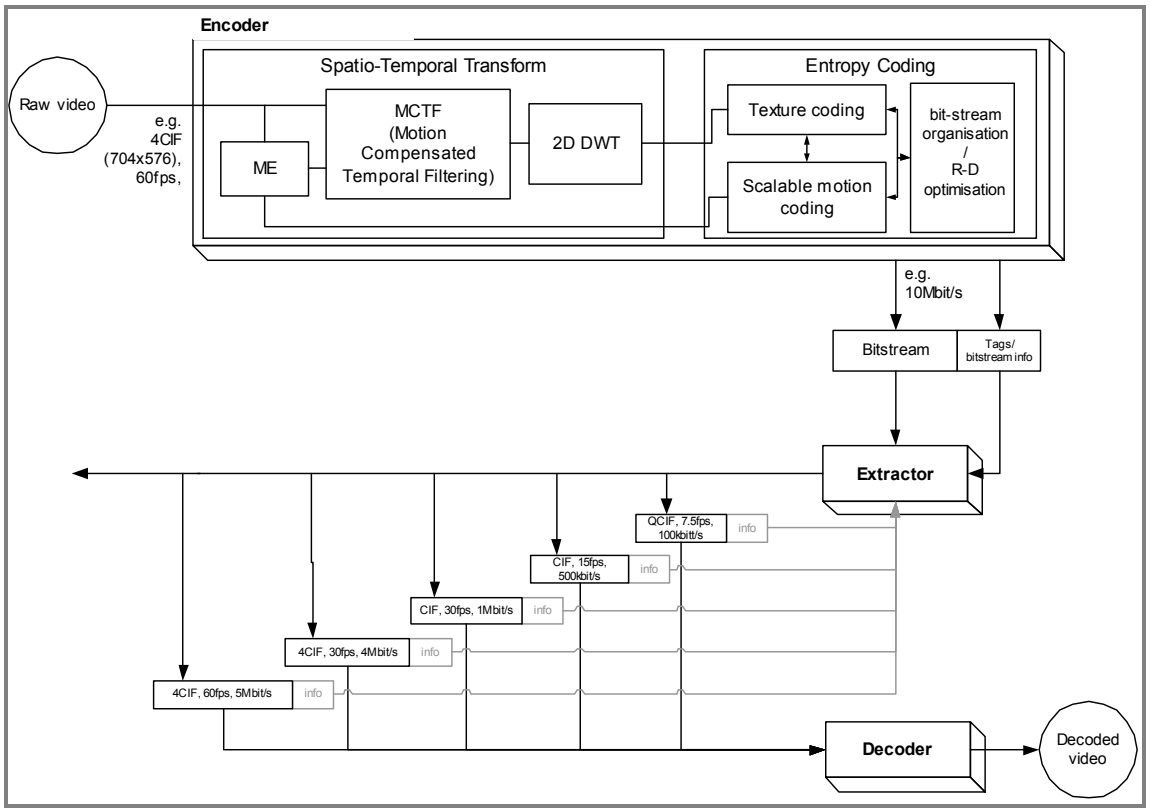

Figura 2-1 Herramienta del codec de vídeo escalable aceSVC

La información que a continuación aparece es un resumen de [6][7][8][9][10], y no pretende entrar en detalle sino ofrecer al lector una visión cualitativa del tema. Si el lector está interesado en profundizar en este campo, se le invita a acudir a las referencias previamente citadas.

2.2.1 Codificador

El codificador o *encoder* encargado de transformar la señal de entrada a una forma codificada usada para la transmisión (en nuestro caso al formato aceSVC), está compuesto por dos módulos: un primer módulo encargado de realizar la transformada espacio-temporal basándose en el esquema de codificación diferencial MCTF (*Motion Compensated Temporal Filtering*) seguida de una descomposición espacial 2D DWT (*Discrete Wavelet Transform*) (también conocido como enfoque t+2D), y un segundo módulo encargado de realizar la codificación entrópica para eliminar la redundancia que pueda haber en la señal.

2.2.1.1 Transformada espacio-temporal

La adaptabilidad en la codificación de vídeo se consigue por medio de una descomposición espacio-temporal eficiente de la secuencia de vídeo de entrada, seguida de una codificación entrópica.

Para señales en tres dimensiones (3D) como son los vídeos, la transformada espacio-temporal consiste en dos partes: una transformada espacial (DWT) y una transformada temporal (MCTF). Ambas transformaciones explotan la redundancia temporal y espacial entre píxeles presente en señales de vídeo y descomponen la señal en una estructura de subbandas jerárquica en 3D, donde los coeficientes de dicha estructura son codificados entrópicamente para alcanzar la codificación escalable.

2.2.1.1.1 Motion Compensated Temporal Filtering (MCTF)

La reducción de la redundancia temporal se realiza con un esquema de codificación diferencial MCTF (*Motion Compensated Temporal Filtering*) interpolando los cuadros a partir de sus cuadros vecinos (anterior y/o posterior).

Este módulo aplica una compensación de movimiento a un conjunto de imágenes (*Group Of Pictures* o GOP). La herramienta aceSVC únicamente emplea la descomposición temporal diádica, por lo que la reducción del *frame rate* es por un factor de potencia de 2.

La Tabla 2-1 muestra los filtros implementados en el codificador escalable. El filtro más sencillo que se puede usar en el MCTF es un filtro de Haar donde se consigue una predicción *forward*. Para obtener un mayor orden de decorrelación se utilizan filtros *wavelet* más complejos como las predicciones bidireccionales LeGall 5/3 *wavelet* con una ventana deslizante.

WAVELET	PREDICCIÓN	ACTUALIZACIÓN
Haar with delta low-pass	Uni-direccional	No
LeGall 5/3 wavelet with delta low-pass	Bi-direccional	No
LeGall 5/3 symmetric wavelet	Bi-direccional	Sí

Tabla 2-1 Tipos de transformadas *wavelet* implementadas en el codec aceSVC

Para garantizar la reversibilidad del esquema, se emplea una implementación de MCTF conocida como *lifting* de banco de filtros *wavelet* [11][12][13]. En el caso de realizar una descomposición temporal del vídeo, después de separar la señal de entrada en cuadros pares e impares se estima el movimiento entre los cuadros de entrada. El campo de vectores resultante (denominado como *v* en la Figura 2-2) se empleará para las operaciones de compensación de movimiento tanto en los pasos de predicción (P) como de actualización (U).

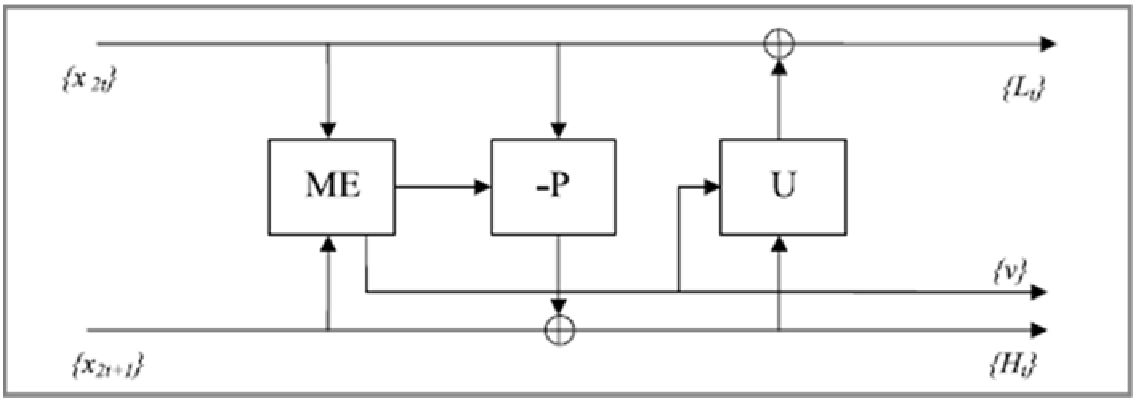

Figura 2-2 Esquema *lifting* de compensación de movimiento espacio-temporal. Ilustración extraída de [13]

El operador P genera la predicción de cada cuadro impar (cuadro actual) a partir de los cuadros pares (cuadros de referencia) mediante las etapas de estimación y compensación de movimiento. A continuación, la salida de P se resta del cuadro actual para obtener el error del cuadro residual (cuadros paso alto) denominado por H en la Figura 2-3.

El cuadro residual obtenido del proceso anterior se añade al cuadro de referencia por el operador U, generando un conjunto de cuadros paso bajo (denominados por L). Para conseguir una descomposición temporal de mayor orden, se vuelve a aplicar el MCTF a los cuadros paso bajo obtenidos. El resultado de este modulo es un conjunto de cuadros paso alto (denominados H) y paso bajo (denominados L) tal como se ilustra en la Figura 2-3.

(a) **(b)**

Figura 2-3: cuadros resultantes del proceso MCTF. (a) cuadro paso bajo (L) y (b) cuadro paso alto (H)

La Figura 2-4 muestra un ejemplo de cuatro descomposiciones temporales (GoP de 16, 8, 4 y 2 cuadros) de una secuencia a partir de la transformada Haar. En este ejemplo, los cuadros rojos son cuadros paso bajo y los amarillos se corresponden con los cuadros paso alto.

Para reconstruir la secuencia, se parte del nivel temporal más bajo (k = 0) y se va subiendo nivel a nivel hasta obtener las resolución temporal deseada. A partir del cuadro LLLL, de los vectores de movimiento y del cuadro residual HLLL se reconstruye el cuadro 9 del nivel k = 1 aplicando el proceso inverso de la Figura 2-2. A este nivel se tienen dos cuadros paso bajo (1 y 9) y dos cuadros paso alto (5 y 13). Si se quiere reconstruir un nivel temporal más, los cuadros 5 y 13 se reconstruirán con sus respectivos vectores de movimiento y con el cuadro paso bajo 1 para el cuadro 5 y el 9 para el cuadro 13.

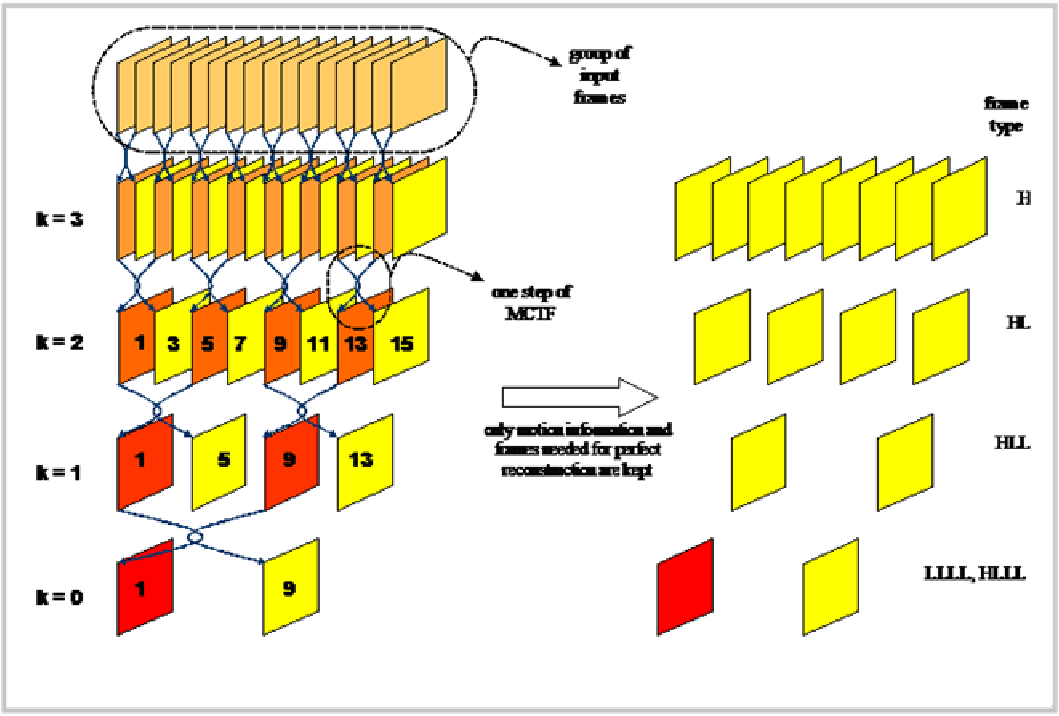

Figura 2-4 Compensación de movimiento MCTF para un GoP de 16 usando la transformada Haar

A diferencia de los estándares de vídeo MPEG-1 y MPEG-2 que utilizan un *block matching* con un tamaño de bloque fijo para estimar el movimiento, la estimación de movimiento (ME), en aceSVC al igual que en otros estándares como el AVC/H.264, está basada en la técnica *block matching* jerárquico de tamaño de bloque variable (*Hierarchical Variable Size Block Matching* o HVSBM). Este método es el encargado de buscar el campo de vectores de movimiento en el MCTF.

Esta técnica permite representar más eficientemente la información del movimiento mediante el empleo de menos vectores de movimiento en áreas grandes que presentan un movimiento uniforme. El campo de vectores del HVSBM se particiona en una estructura *quadtree* asignando un vector de movimiento a cada bloque en cada subbloque. La Figura 2-5 muestra un ejemplo de la estructura *quadtree* y los vectores de movimiento obtenidos por el HVSBM [14].

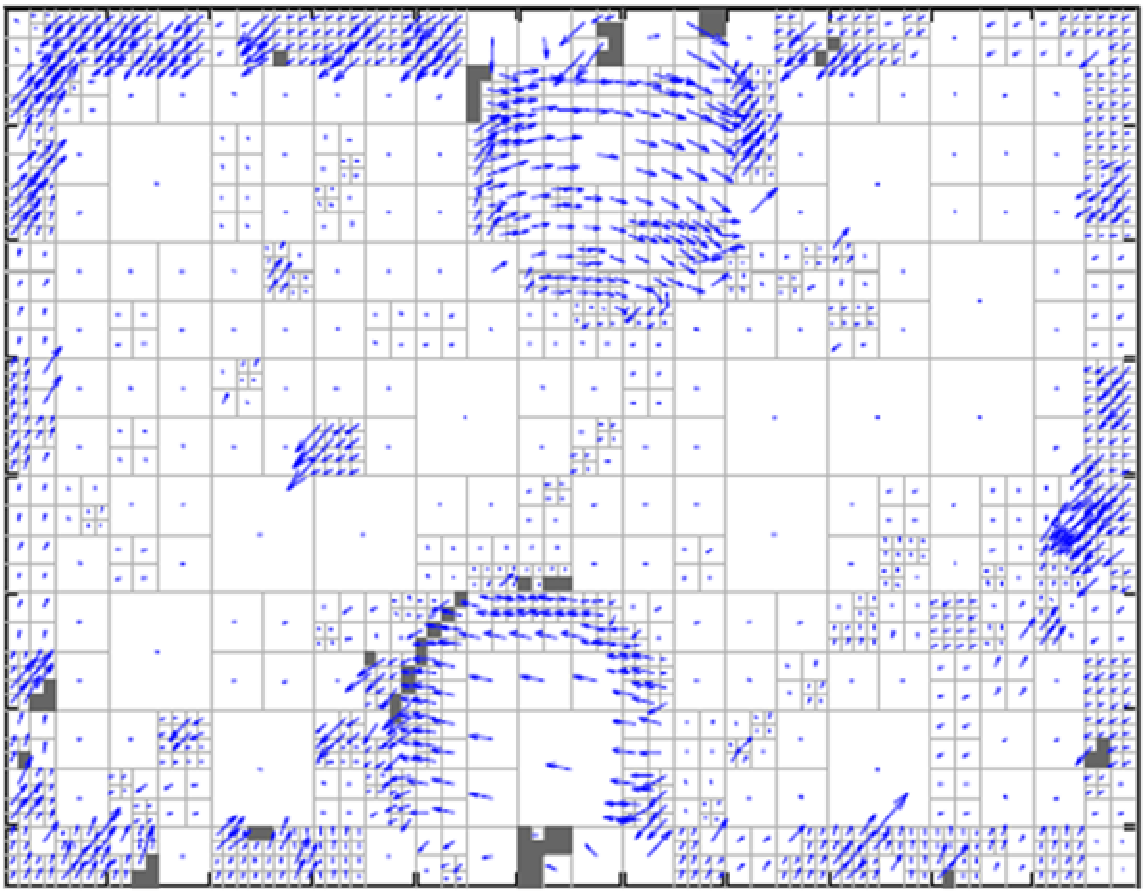

Figura 2-5: Ejemplo del campo de vectores resultante de aplicar el HVSBM

2.2.1.1.2 Descomposición espacial (2D DWT)

La transformada espacial discreta (DWT) es una transformada separable con lo que puede extenderse fácilmente a múltiples dimensiones por medio de la aplicación de la transformada unidimensional, 1D DWT, a cada una de las dimensiones.

En particular, la descomposición espacial implementada en el codec aceSVC, al tratarse de imágenes en 2D, consiste en la transformada DWT bidimensional (2D DWT), la cual se obtiene aplicando la transformada 1D DWT en cada fila y cada columna de la imagen. De ellas, son bien conocidas las transformadas de Daubechies 9-*tap*/7-*tap* (9/7) y la 5-*tap*/3-*tap* (5/3).

En aceSVC al igual que en JPEG2000, se usa la transformada discreta *wavelet* (DWT), que es una transformada diádica por lo que se obtienen descomposiciones de potencias de 2. Así, cada vez que se aplica esta transformada, el cuadro se descompone en cuatro bandas de frecuencias (ver Figura 2-6 (a)), es decir, cada cuadro se descompone en un cuadro promedio y tres imágenes detalle.

La Figura 2-6 (a) presenta el esquema de 2D DWT donde se aplica primeramente la transformada DWT unidimensional a las filas del cuadro y a continuación a las columnas. Las figuras (b) y (c) son el resultado obtenido de aplicar este tipo de transformada a la imagen Lena con una y dos descomposiciones espaciales respectivamente.

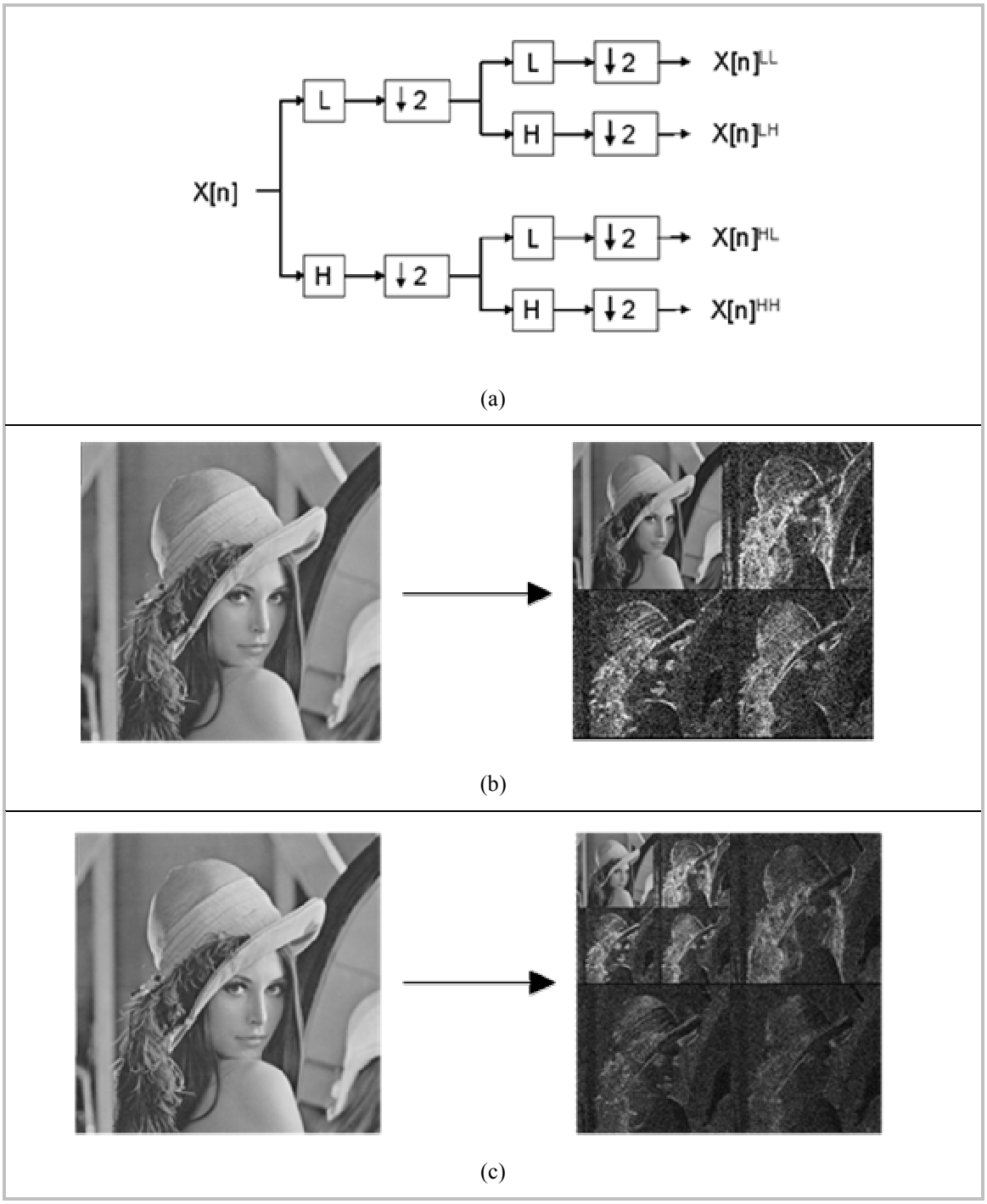

Figura 2-6 (a) Esquema de filtros de la transformada 2D DWT a partir de un filtro 1D DWT. Ejemplo de la descomposición diádica en subbandas para la imagen Lenna para un nivel de descomposición 2D DWT (a) y para dos niveles (c)

Para más información acerca de este tema, se invita al lector a acudir a las referencias [6][28][29][30].

2.2.1.1.3 Codificación entrópica

Este módulo está compuesto por tres partes:

- Una codificación de textura basada en la codificación de los cuadros del GoP, obtenidos a partir de la descomposición espacio-temporal, mediante EZBC (*Embedded Zero Block Coding*).
- Una codificación escalable de los vectores de movimiento.
- Un módulo de optimización R-D que optimiza la asignación de los bits a lo largo de los elementos del *bitstream* para un conjunto en particular de puntos de decodificación o para unas tasas/resolución de decodificación.

Para profundizar más en este tema, se aconseja al lector acudir a las referencias [6][8].

2.2.2 Bitstream escalable

Una secuencia de vídeo transformado que use una de las técnicas *wavelet* de *inter-frame* (técnica que explota la correlación temporal entre cuadros consecutivos para poder codificar con el mínimo número de bits posibles) normalmente está organizada jerárquicamente en un *bitstream*.

La Figura 2-7 muestra un ejemplo de un *bitstream* de un codec escalable. Cada GOP está formado por cuadros descompuestos temporalmente y espacialmente. La información de movimiento es enviada como una parte incorporada de cada cuadro paso alto (H, HL, HLL..., etc.), por lo que tanto el cuadro residual como la información de movimiento están asociadas a estos cuadros.

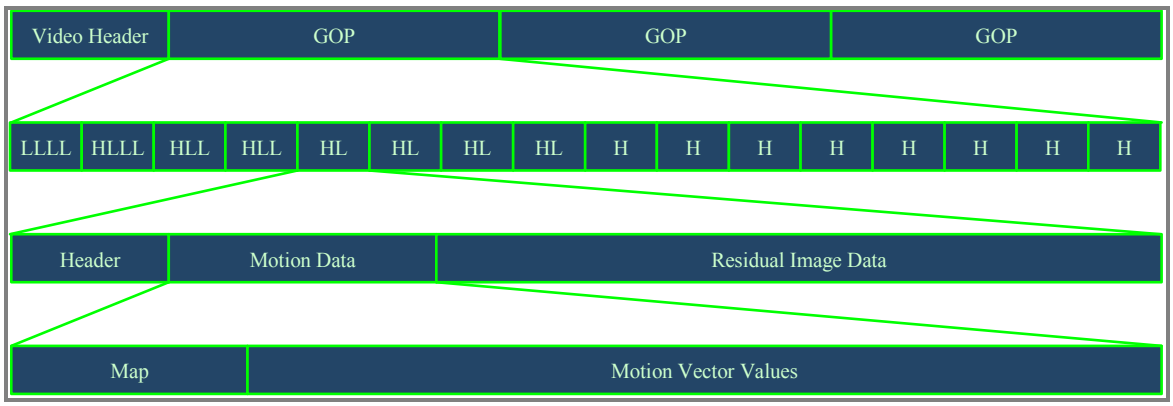

Figura 2-7 Organización del *bitstream* de un vídeo aceSVC

La transformada espacial *wavelet* (2D DWT) de cada cuadro temporal tiene como finalidad la obtención de la escalabilidad de la resolución de la secuencia para todos los niveles temporales. La Figura 2-8 muestra dicha adaptabilidad para la subbanda LLLL.

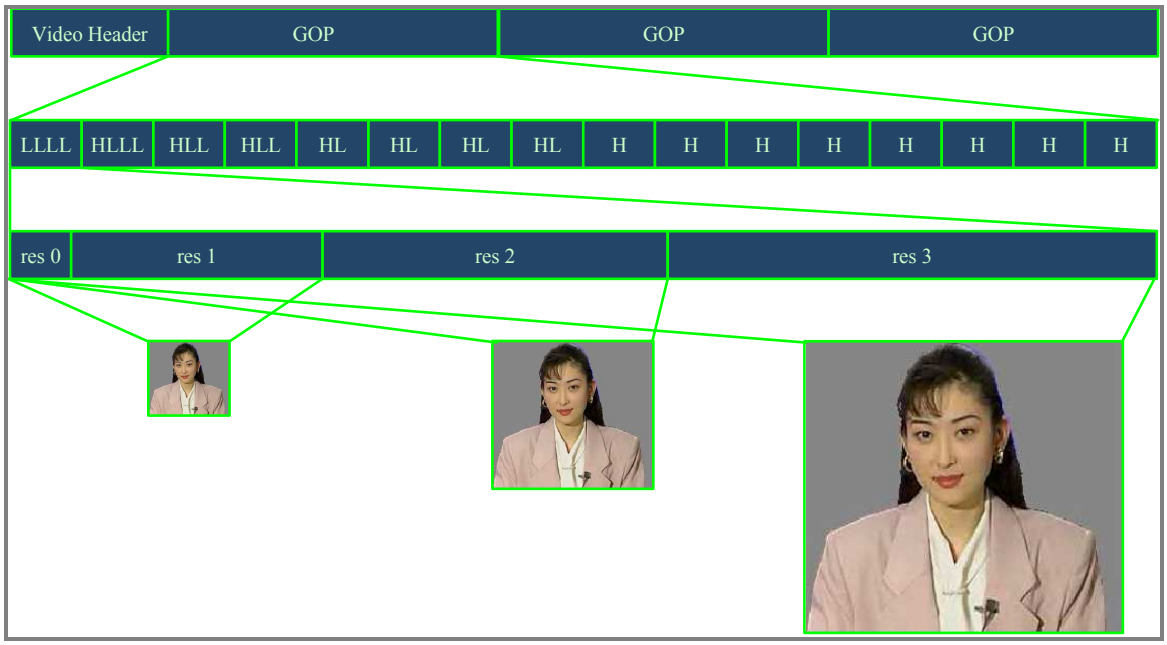

Figura 2-8 Escalabilidad de la resolución en el *bitstream*

Como la descomposición diádica se emplea tanto en las transformadas temporales como en las espaciales, la reducción en cada nivel de la escalabilidad temporal y espacial es por un factor de dos. Cada nivel de resolución espacio-temporal realiza una granulación fina, que permite la extracción con precisión de byte. La siguiente figura refleja la funcionalidad de la escalabilidad por calidad soportada.

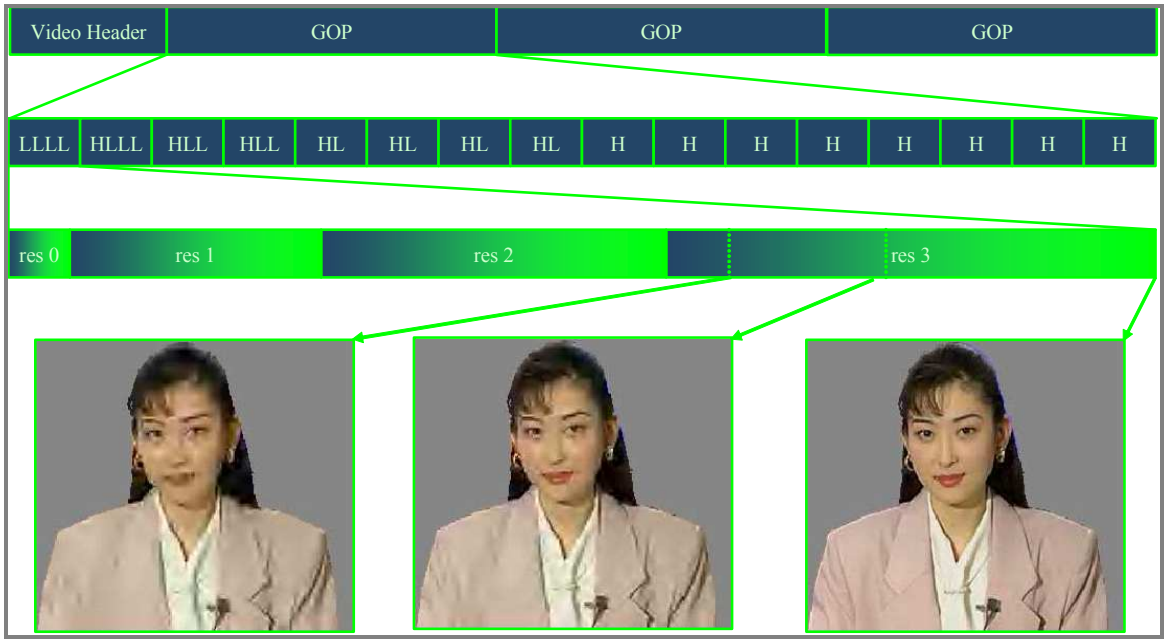

Figura 2-9 Escalabilidad de la calidad en el *bitstream*

El *bitstream* se organiza de acuerdo a una estructura de tres niveles espacial-temporal-calidad. Para una interpretación más sencilla del proceso de extracción, el *bitstream* puede representarse como una estructura 3D QTS (*Quality, Temporal resolution, Spatial resolution*), con diferentes partes del *bitstream* en diferentes coordenadas, tal como aprecia en la Figura 2-10.

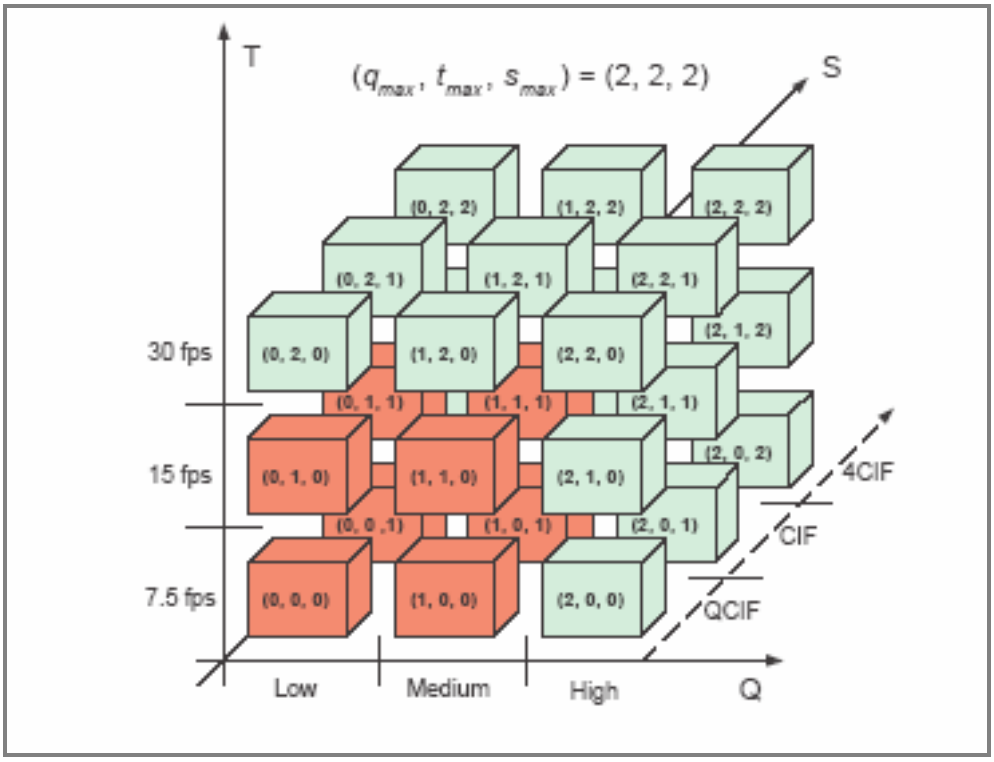

Figura 2-10 Representación en 3D del *bitstream* escalable

Las coordenadas se determinan mediante la capa de calidad y las subbandas espacio-temporales que representan una parte particular del *bitstream*. Estas partes particulares son denominadas elementos o *átomos* del *bitstream*, ya que representan la entidad más pequeña que se puede añadir o eliminar del *bitstream*.

Si denotamos la capa más baja como la 0, y la capa de calidad y las resoluciones temporales y espaciales más altas como q_{max}, t_{max} y s_{max} respectivamente, cada átomo se puede representar mediante la tupla (q,t,s) correspondiente a su posición en el espacio QTS tal como muestra la Figura 2-10. Si denotamos la resolución temporal, espacial y de calidad deseada mediante (Q_i, T_j, S_k), donde $Q_i \in \{Q_0, ..., Q_{qmax}\}$, $T_j \in \{T_0, ..., T_{tmax}\}$ y $S_k \in \{S_0, ..., S_{smax}\}$, el extractor produce un *bitstream* que se expresa como:

$$\bigcup_{q=0,t=0,s=0}^{i,j,k}(q,t,s)$$

En otras palabras, dicho *bitstream* producido por el extractor preserva todos los átomos con los índices iguales o inferiores al índice más alto deseado, descartando el resto.

2.2.3 Extractor

En función de las restricciones de salida o de una demanda particular de adaptación, el motor de adaptación realiza la extracción de los átomos del *bitstream* necesarios y forma un nuevo *bitstream* adaptado.

A su vez, si es necesario, como el *bitstream* adaptado preserva todos los átomos con los índices iguales o inferiores al índice seleccionado en la extracción, es posible obtener una nueva adaptación del *bitstream* previamente adaptado que puede ser empleada para una futura adaptación en algún nodo de la cadena de transmisión, lo que conlleva al concepto de adaptación múltiple.

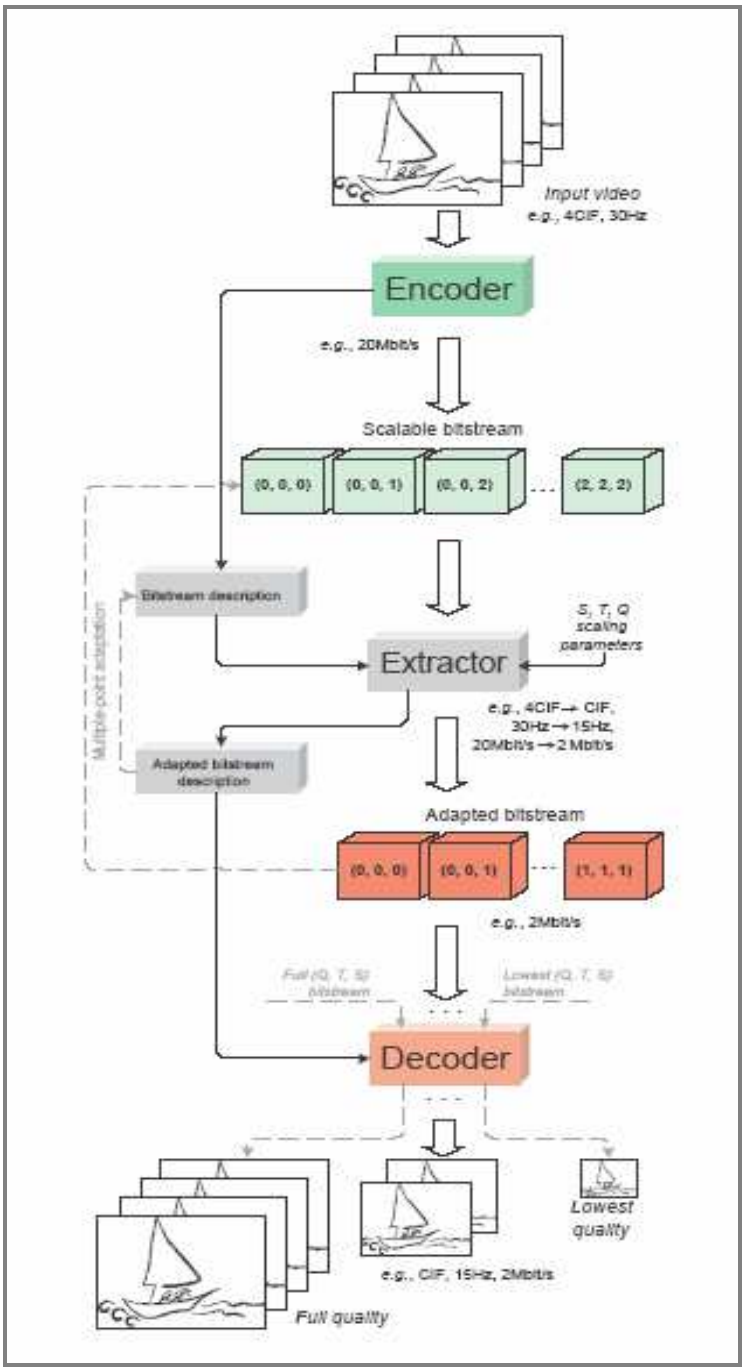

Figura 2-11 Módulos del codec aceSVC y adaptación basada en aceSVC

Así por ejemplo, si una secuencia 4CIF a 60 cuadros por segundo se codifica con tres descomposiciones espaciales, temporales y de calidad, el *bitstream* contendrá las versiones adaptadas 4CIF, CIF y QCIF a 60, 30 y 15 cuadros por segundo con tres niveles de calidad. Si se extrae la capa CIF a la máxima resolución temporal (60 cuadros por segundo) y de calidad, este nuevo *bitstream* adaptado contendrá las versiones CIF y QCIF a 60, 30 y 15 cuadros por segundo con tres niveles de calidad, de forma que se puede seguir extrayendo las versiones QTS inferiores a ésta.

La Figura 2-11 muestra el diagrama de bloques de los tres módulos de aceSVC, y cómo se realiza la adaptación de la secuencia.

La Figura 2-12 muestra varias aplicaciones del concepto de adaptación múltiple. El primer punto de adaptación se realizaría en el servidor de vídeo ya que éste contiene el vídeo comprimido en su calidad total. Si se demanda una calidad de vídeo inferior, se puede realizar una extracción improvisada y entregar dicho vídeo adaptado. Sin embargo, la adaptación también puede realizarse dentro de la red en los llamados *routers* escalables que contienen el software del extractor por lo que son capaces de realizar dicha extracción.

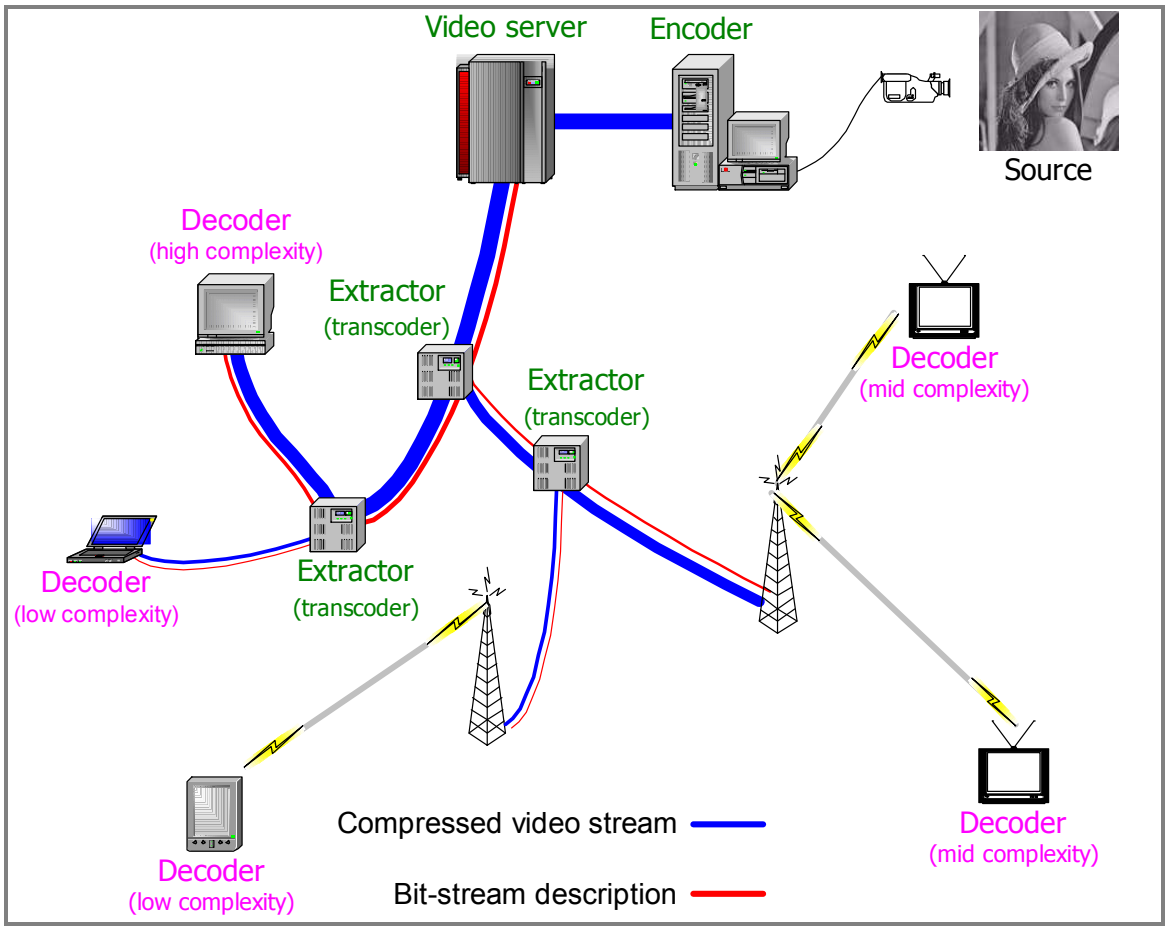

Figura 2-12 Ejemplo de adaptación múltiple

2.3 RC-CAT

Esta sección describe el componente denominado *Real-time Content-based Content Adaptation Tool* (RC-CAT), encargado de realizar la adaptación *online* de la secuencia de entrada en función de una serie de restricciones impuestas por el usuario final. Lo que se presenta a continuación es una visión global de la estructura y los principales componentes que componen la arquitectura de la herramienta, sin entrar en detalle en el funcionamiento de los mismos.

En caso de que el lector desee profundizar más en alguno de los temas que se presentan, se le invita a acudir a las referencias [15][16][17][22] de donde se ha extraído la información proporcionada.

2.3.1 Introducción

RC-CAT es una herramienta de adaptación de contenido (*Content Adaptation Tool* o CAT) cuyo principal objetivo consiste en generar un fichero multimedia de salida de acuerdo a unos parámetros de adaptación (*bitrate*, formato, etc.) definidos durante el proceso.

Dicha herramienta está desarrollada para alcanzar una adaptación *online* eficiente (si es posible a tiempo real) de vídeos, basada en la identificación y la preservación del contenido más representativo. Para ello, este CAT selecciona los cuadros clave o *keyframes* (opcionalmente resaltando los objetos de primer plano) representativos del vídeo, los cuales se emplean para generar la versión adaptada, ya sea un conjunto de imágenes estáticas o un vídeo en formato MPEG-1/2.

Para conseguir que la adaptación realizada por el RC-CAT sea a tiempo real, el análisis de contenido debe basarse en la información directamente disponible en el flujo de vídeo codificado, es decir, de aquella que se pueda extraer eficientemente del llamado *dominio comprimido*, de manera que se evite la decodificación y el análisis de todo el vídeo.

Con el fin de realizar dicha adaptación, el RC-CAT realiza una estimación inicial de la relevancia de los cuadros completos y de determinadas regiones de los cuadros del vídeo de entrada. Asimismo, si la adaptación implica la eliminación de algunos datos (de un modo temporal, como la elaboración de un resumen del vídeo, o espacial, como la eliminación o merma de calidad de los objetos del fondo de la escena), se descartarán en primer lugar los datos menos importantes para el usuario.

Las preferencias que tienen los usuarios respecto a la visualización del contenido y el contexto de uso del mismo (las características del terminal que se utiliza y de la red involucrada) determinan el tipo de adaptación a llevar a cabo en el RC-CAT y el modo en que los cuadros o las regiones de interés serán incluidas en el contenido final adaptado.

Los modos de adaptación definidos en la herramienta son los siguientes: uno referido a características temporales, otro referido a características espaciales y el último en función de una combinación de los modos anteriores.

En cuanto a la adaptación referida a características temporales, el estado actual del RC.-CAT considera tres tipos de modalidades de adaptación:

- *Video slideshows*: Este tipo de codificación conserva la duración original de la secuencia. Para ello como la adaptación se realiza sólo con las *keyframes*, cuando se codifica un cuadro que no es *keyframe*, se replica el *keyframe* anterior.
- *Video skims*: Idéntico al anterior, pero sin replicación de *keyframes*, lo que conlleva a la reducción de la longitud con respecto a la secuencia original.
- *Image storeboards*: Conversión de las *keyframes* del vídeo a imágenes

La adaptación basada en la extracción de características espaciales es gestionada por medio de las regiones de interés. Estas regiones son extraídas mediante una separación o segmentación *gruesa*, basada en criterios de movimiento, del fondo de la escena (*background*) y de los objetos de primer plano (*foreground*). El *foreground* se considera como la región de interés del cuadro.

La adaptación de la región de interés se lleva a cabo presentándola con alta calidad, mientras que el *background* puede ser sustituido por un color uniforme (generalmente negro) para reducir su coste de codificación o filtrado por un filtro paso bajo, para reducirlo también pero manteniendo parcialmente su visibilidad.

Finalmente, se debe remarcar que la adaptación en el RC-CAT fue diseñada para operar en tiempo real de una manera eficiente. Estos dos requerimientos imponen restricciones en el diseño de algoritmos de análisis y en la manera de realizar la adaptación. Las características relevantes deben de ser extraídas en tiempo real y con poco retraso para conseguir que la adaptación pueda realizarse de un modo *online*.

2.3.2 Visión General de la arquitectura del RC-CAT

Esta sección expone una visión general de la arquitectura, centrándose en los componentes relacionados con el análisis espacial y la adaptación.

La Figura 2-13 muestra dicha arquitectura, donde se distinguen tres grandes módulos: el módulo Análisis de Contenido (*Content Analysis*, CA), el módulo Control de Adaptación (*Adaptation Control*, AC) y el módulo de Generación de Medios (*Media Generation*, MG).

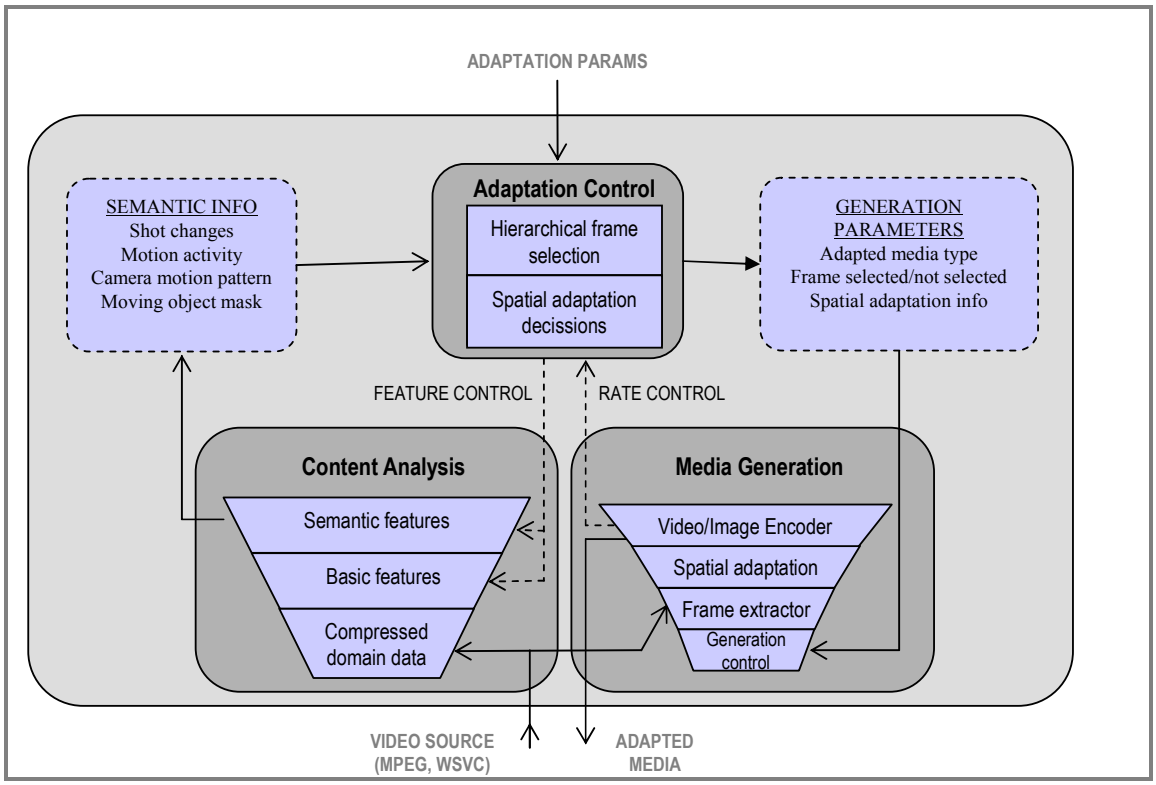

Figura 2-13 Arquitectura del RC-CAT

2.3.3 Análisis de Contenido

El módulo de Análisis de Contenido es el encargado del análisis del vídeo de entrada, es decir, de la extracción de la información semántica relevante que será usada para guiar el proceso de adaptación del contenido. El tipo y número de características que se pueden extraer depende de la decisión inicial realizada por el módulo de Control de Adaptación.

Con el fin de conseguir que el RC-CAT opere en tiempo real, la extracción de las características semánticas por parte del análisis de contenido debe realizarse por medio de técnicas que operen directamente sobre el vídeo comprimido y siguiendo un modelo abstracto que permita una adaptación transparente para vídeos basados en DCT (MPEG) y para vídeos basado en *wavelet* (aceSVC), evitando la decodificación completa de toda la información del vídeo de entrada.

Dependiendo del formato del vídeo de entrada, se seleccionará el correspondiente decodificador parcial para obtener una decodificación eficiente de las características básicas (imágenes de baja resolución, vectores de movimiento e información de textura).

Estas características básicas extraídas son analizadas por algoritmos genéricos con el fin de extraer aquellas características semánticas (cambios de toma, movimiento de cámara, actividad de movimiento y máscaras para la segmentación del *background* y el *foreground*) requeridas para realizar la adaptación deseada.

Si se requieren características espaciales para la adaptación, el módulo debe ser dirigido para obtener las máscaras de los objetos en movimiento, aunque para la extracción de dichas máscaras son necesarios los cambios de toma y los parámetros afines del movimiento de cámara. Los cambios de toma son extraídos a partir de las imágenes de baja resolución, obtenidas por las imágenes DC en el caso de MPEG o de la información de las subbandas (píxeles en formato YUV) en caso de aceSVC. Los parámetros afines del movimiento de cámara y las máscaras de objetos de movimiento son derivados a partir de los vectores de movimiento.

Las variables de entrada de este módulo son el vídeo a analizar y el tipo de adaptación requerida (temporal, espacial o ambas) usado internamente para determinar qué tipo de datos o características serán extraídas o calculadas. Las variables obtenidas como resultado de este módulo son enviadas al módulo de control de adaptación encargado de guiar el proceso de adaptación.

La siguiente figura presenta las diferentes etapas en las que está estructurado el módulo.

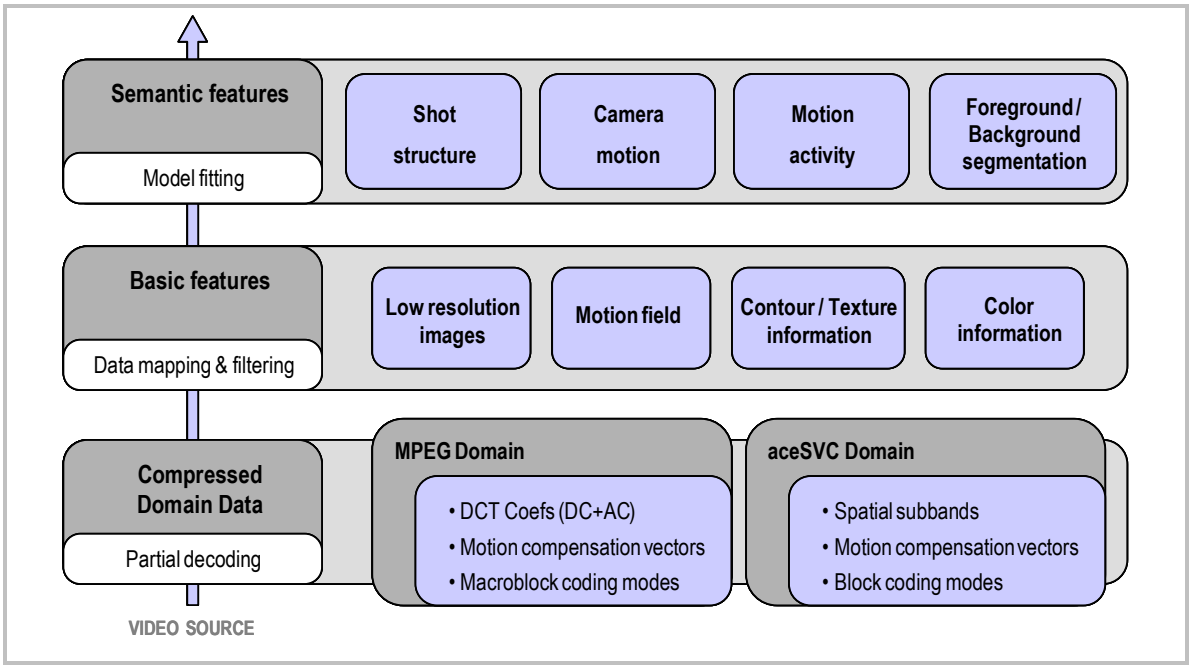

Figura 2-14 Arquitectura del módulo de análisis de contenido

2.3.3.1 Extracción de características del vídeo

El objetivo de esta etapa consiste en obtener información relevante del vídeo que será utilizada posteriormente en la etapa de análisis. La extracción de las características básicas necesarias para guiar el proceso de adaptación se realiza por medio de una decodificación parcial del flujo de vídeo de entrada, evitando así algunas etapas del proceso de análisis e incrementando la eficiencia.

Dependiendo del formato del vídeo de entrada (MPEG o aceSVC), los datos extraídos variarán en función del dominio, como muestra la Figura 2-14 y la Tabla 2-2.

2.3.3.2 Extracción de características básicas

En esta etapa se pretende obtener una representación uniforme de los datos del vídeo independientemente del dominio del que hayan sido extraídos. Esta representación permite definir las métricas y las características independientes de más alto nivel, simplificando la ampliación de funcionalidades del RC-CAT.

Dependiendo del formato del vídeo de entrada, se elegirá un decodificador parcial para extraer eficientemente las características básicas identificadas (imágenes en color de baja resolución, vectores de movimiento e información de textura). Estas características serán analizadas por una serie de algoritmos genéricos con el fin de extraer las características necesarias para realizar la adaptación requerida. La Tabla 2-2 muestra la relación entre dichas características para los dos dominios.

CARACTERÍSTICAS BÁSICAS	**MPEG**	**aceSVC**
Imágenes en color de baja resolución	Coeficientes DC	Píxeles en formato YUV
Campo de vectores	Vectores de movimiento	Vectores de movimiento
Información de contorno y textura	Coeficientes AC	Subbandas

Tabla 2-2 Relación de las características básicas para MPEG y aceSVC

2.3.3.3 Cálculo de características de nivel medio

Las características de nivel medio no pueden ser obtenidas directamente del flujo de vídeo o por medio de un mapeo rápido de algunos datos del flujo de vídeo, por eso para su obtención se aplican una serie de algoritmos de análisis.

Estas características juegan un papel importante debido a que serán usadas por el módulo de control de adaptación para guiar el proceso de adaptación. Las características definidas en esta etapa son los cambios de toma, la actividad de movimiento, el movimiento de cámara y una segmentación gruesa de las máscaras del *foreground* y del *background*.

El proceso de adaptación está dirigido por los eventos y la realimentación procedente del módulo de Generación de Medios, y controlado por el modulo de Control de Adaptación. De esta forma, la asignación de si un cuadro es o no *keyframe*, y la manera de adaptar dicho *keyframe*, es realizada por la etapa de adaptación (AC) en lugar de la etapa de análisis (CA), ya que los *keyframes* y las regiones de interés no son considerados como características de nivel medio.

- **Cambios de toma (*video shot changes*)** son considerados como la unidad básica para la segmentación temporal de vídeos debido a que agrupan cuadros de similar contenido, al resultar de una operación ininterrumpida de la cámara.

  La detección de los cambios de toma se basa en la diferencia entre cuadros consecutivos para detectar tomas, usando el algoritmo descrito en [18]. Este algoritmo se basa en una métrica de histogramas a partir de las imágenes de baja resolución obtenidas en la etapa anterior.

- **La actividad de movimiento (*motion activity*),** es la característica más usada en tareas de análisis y resúmenes de vídeos para capturar el nivel de acción en un segmento del vídeo. Para cuantificar este valor se emplea un descriptor definido en MPEG-7 [19] basado en la desviación estándar de los vectores de movimiento normalizados por unos parámetros del vídeo.

- **El movimiento de cámara (*camera motion*)** ofrece información para determinar donde tienen lugar los eventos relevantes del vídeo. No obstante, la presencia de determinados movimientos de cámara como el *panning* o el *zoom* son muy útiles para las tareas de extracción de *keyframes*.

  Para estimar el movimiento de cámara se emplea el método descrito en [20], donde el algoritmo iterativo intenta ajustar los vectores de movimiento de los cuadros P a un modelo afín, rechazando aquellos vectores que no siguen el movimiento estimado (el fondo de la imagen). La suposición subyacente de este método es que el tamaño del fondo de la imagen es más grande que el de los objetos, lo que a veces puede conducir a algunos errores en la estimación.

- La **segmentación gruesa (*coarse segmentation*)** se basa en la separación de los objetos en movimiento del primer plano (*foreground*) sobre el fondo (*background*) en resolución de macrobloque.

  Para la obtención de estas máscaras se utiliza el mismo algoritmo propuesto para el movimiento de cámara, donde los macrobloques rechazados son considerados como los objetos en movimiento (*foreground*) del vídeo. Estos macrobloques rechazados son trazados mediante un vector asociado para filtrar aquellos macrobloques que sean incoherentes con el movimiento. Para ello, se hace uso de una versión adaptada del método explicado en [21].

  La máscara final obtenida muestra aquellos macrobloques que no siguen el movimiento del background y que son consistentes con el trazado del siguiente cuadro.

2.3.4 Generación de Medios

El objetivo de este módulo consiste en generar el fichero de salida adaptado a partir del vídeo de entrada y una serie de restricciones de adaptación: el tipo de adaptación requerida y la información resultante del análisis del módulo de adaptación de contenido (*keyframes* y regiones de interés).

Estas restricciones de adaptación indicarán qué cuadros serán incluidos en el fichero de salida y la manera en la cual serán adaptados. Así, en el caso de requerirse una adaptación espacial, la región de interés de los cuadros (*foreground*) se resaltará sustituyendo el *background* por un color uniforme o filtrando éste con un filtro paso bajo para emborronarlo.

Con el fin de conservar el sincronismo en el fichero adaptado, cualquiera de las dos decisiones temporales de adaptación, incluir o no el cuadro actual, debe de comunicarse al transcodificador (*Transcoder*), tal como muestra la Figura 2-15, que en función del tipo de salida a generar (*Video Slideshow*, *Video Skimming* o *Image Story Board*) adoptará un comportamiento u otro.

Durante el proceso de adaptación, el transcodificador es alimentado con la información de las imágenes seleccionadas (*keyframes*), que serán o no incluidas en el fichero adaptado, y con la información espacial asociada a estas imágenes: las máscaras de *foreground* y *background* y el porcentaje de reducción de calidad que se aplicará cuando se codifique la región del *background*.

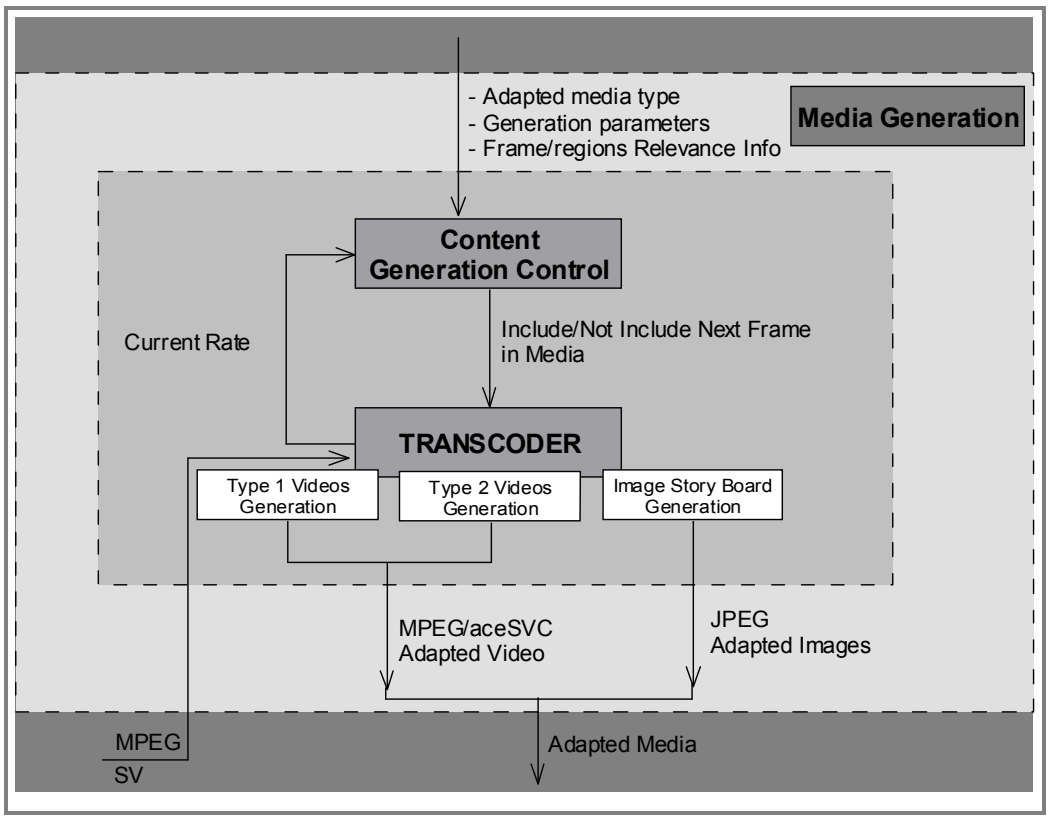

Figura 2-15 Arquitectura del módulo generador de medios

2.3.4.1 Transcodificador

El transcodificador es el encargado de generar el fichero multimedia de salida objetivo de la adaptación. Las estrategias de adaptación se han clasificado en técnicas espaciales y temporales basadas en función de las características a nivel de cuadro y de las características nivel de objeto. Según esta clasificación los modos de adaptación son los siguientes:

- Tres modos de adaptación temporal, relacionados con la manera en la cual la selección jerárquica de los cuadros clave (*keyframes*) son almacenados en el fichero multimedia de salida.

  - ***Video slideshow*** (denominado como *Type I Videos* en la Figura 2-15): el fichero de salida generado es un vídeo codificado en formato MPEG 1/2, el cual preserva la longitud del vídeo de entrada, mediante la replicación de los *keyframes*. Si el cuadro a codificar es un *keyframe* se añade al fichero de salida; si no, se replica el *keyframe* anterior. Esta situación es la ideal para poder sincronizar el audio del vídeo original.

  - ***Video skim*** (denominado como *Type II Videos* en la Figura 2-15): el fichero de salida generado es un vídeo codificado en formato MPEG-1/2 en el cual sólo los *keyframes* son almacenados, por lo que al contrario que en el modo anterior la longitud del vídeo se ve reducida.

  - ***Image story-board*****:** el fichero de salida es una colección de imágenes (sólo los *keyframes*) en formato JPEG.

- Dos modos de adaptación espacial relacionados con la manera en la que las áreas no relevantes son codificadas:

  - ***Uniform-filling*****:** las áreas que se encuentran fuera de la región de interés (*background*) se rellenan con un color uniforme, perdiendo parte de la información contextual del cuadro. Generalmente es usado el color negro para reducir el coste de codificación.

  - ***Low-pass filtering*****:** las áreas que se encuentran fuera de la región de interés son filtradas mediante un filtro paso bajo de manera que se consigue incrementar la redundancia espacial y temporal a la vez que se consigue reducir la cantidad de datos a codificar manteniendo parcialmente la información contextual del cuadro.

En el caso de requerirse una adaptación espacial se tienen en cuenta dos aspectos: la dimensión de la región de interés y la del fichero de salida. Si ambas dimensiones coinciden, simplemente se copia el contenido de la región de interés en el cuadro de salida o cuadro objetivo. Si por el contrario, las dimensiones de la región de interés son más pequeñas o grandes que las del cuadro objetivo, las dimensiones de la región se expanden o comprimen o se aplica un submuestreo proporcional a los datos.

2.3.5 Control de adaptación

El módulo de Control de Adaptación es el encargado de dirigir todo el proceso de adaptación *online*. Cuando una adaptación es requerida, los parámetros de adaptación definirán el tipo de adaptación temporal (ninguna, *video slideshow*, *video skim* o secuencia de imágenes) y la adaptación espacial (ninguna, degradación del *background* o sustitución con un color uniforme) deseada. Para cada cuadro del vídeo de entrada se realizan las siguientes acciones:

- Primero, en función de los parámetros de adaptación, el módulo de Control de Adaptación indica al de Análisis de Contenido que lea y analice un nuevo cuadro del vídeo de entrada, extrayendo las características espaciales y temporales requeridas. Estos parámetros de adaptación incluyen el tipo de adaptación deseada por el usuario (vídeo de tipo I, de tipo II y secuencia de imágenes), el *bit-rate*, el número de cuadros por segundo y las restricciones de tamaño impuestas por el terminal.

  En cuanto a la segmentación de los objetos en movimiento, otros parámetros podrían ser requeridos. En el caso de sustituir el *background* con un color uniforme, este parámetro debe ser indicado. Si la adaptación deseada implica una degradación de calidad del *background*, se tendrá que especificar la intensidad de esta degradación.

- A continuación, empezando por las máscaras de objetos extraídas y por los parámetros de adaptación, este módulo decidirá si el cuadro será incluido o no en el fichero de salida (adaptación temporal) y la manera en la que éste será incluido (adaptación espacial). El proceso de selección de estos cuadros representativos o *keyframes* se explicará en la sección 2.3.5.1

- Finalmente, el módulo control de adaptación dirige al generador de medios, especificando el conjunto de parámetros de generación para adaptar el cuadro según las especificaciones impuestas.

2.3.5.1 Proceso de selección de keyframes

Para conseguir que la adaptación sea en tiempo real, no sólo se necesita que el análisis sea eficiente y en tiempo real sino que también es necesario un esquema apropiado para evaluar los resultados del análisis y así seleccionar los cuadros que serán incluidos en el resultado de la adaptación.

En primer lugar, el proceso de selección requiere establecer un esquema que permita seleccionar un número apropiado de *keyframes* dependiendo de las preferencias de usuario y/o de las características de la red, así como algún tipo de política de prioridades para escoger los *keyframes*.

En la Figura 2-16 se observan los mecanismos establecidos para que la selección de *keyframes* se realice de forma *online*. En función de la información semántica disponible, se han identificado hasta cuatro niveles: cuanto mayor sea el nivel, mayor será el número de *keyframes* seleccionados.

Como se explicará a continuación, los tres primeros niveles basan la selección de *keyframes* en la presencia o la ausencia de eventos semánticamente relevantes; por lo tanto, el número de cuadros seleccionados en cada nivel están definidos por el número y tipo de eventos semánticos de la secuencia. Por otro lado, el cuarto nivel permite la extracción de un número variable de cuadros, ajustando el grado de submuestreo de la actividad de movimiento.

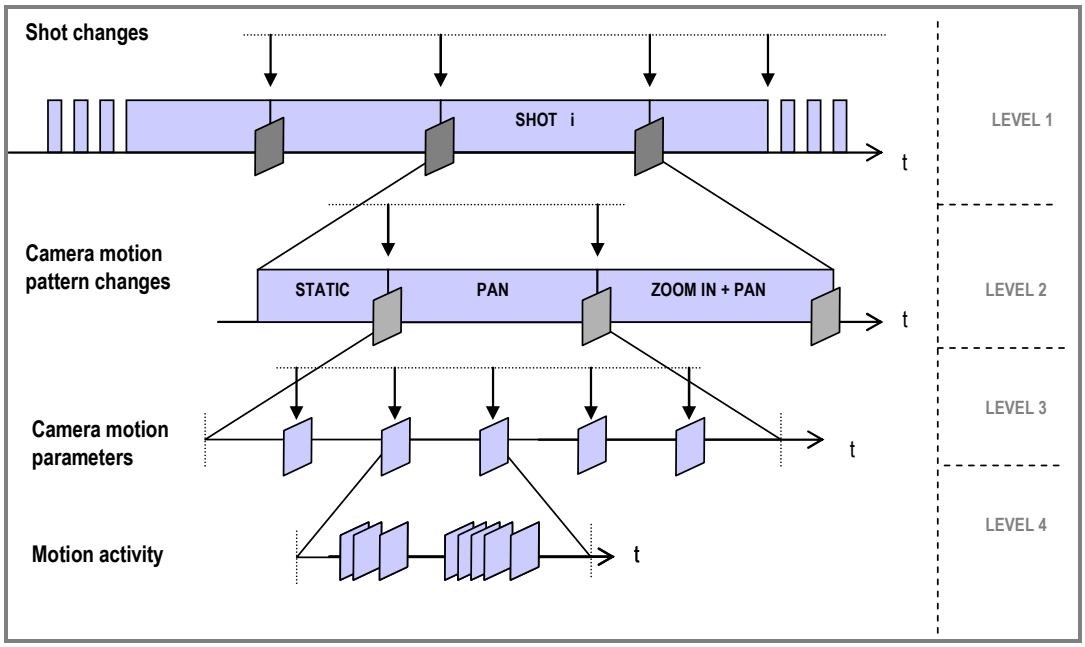

Figura 2-16 Selección jerárquica de *keyframes*

Nivel 1: Selección basada en tomas

Consideremos el caso general en el que una secuencia de vídeo contiene varios cambios de toma. Mientras el vídeo de entrada es analizado, el detector de cambios de toma marca con precisión de cuadro los límites entre tomas; siendo estos cuadros los correspondientes al primer nivel.

Teniendo en cuenta consideraciones de eficiencia, se selecciona como *keyframe* el primer cuadro después de los límites de la toma ya que puede ser extraído eficientemente (siendo el primer cuadro I para MPEG y la primera subbanda temporal paso bajo para aceSVC).

Nivel 2: Selección basada en el esquema de operación de la cámara

En este caso, el análisis se centra en el conjunto de imágenes correspondiente a un segmento del vídeo comprendido entre dos cambios de toma consecutivos por lo que a diferencia del nivel 1, el plano de la imagen mantiene la misma escena, es decir el último cuadro del segmento definido por un patrón de movimiento será prácticamente idéntico al primer cuadro del siguiente segmento a no ser que se trate de un cambio de toma.

En estos segmentos generalmente se muestran diferentes patrones de cámara (*static, pan, zoom,*…), siendo el último cuadro, y por tanto el *keyframe* correspondiente al segundo nivel, el que delimita cada uno de los patrones encontrados en el segmento.

Nivel 3: Selección basada en el movimiento de la cámara

Este nivel asume la disponibilidad de segmentos de secuencia homogéneos desde el punto de vista del movimiento de cámara, del esquema concreto de movimiento de cámara y de los parámetros de dicho movimiento para cada cuadro P.

Dado que el fin es la adaptación, en la identificación del esquema del movimiento de cámara no es necesario realizar una clasificación exhaustiva de todos los posibles movimientos. Por eso sólo se consideran los movimientos de cámara *pan, tilt, zoom* y *roll* y en caso de encontrar un *track*, *boom* o *dolly* se clasificarán como *pan, tilt* y *zoom* respectivamente.

En este nivel el criterio de selección se basa en la evaluación de la cantidad de contenido que entra y sale del plano de la imagen. Una vez seleccionada esta cantidad, se hace uso de los parámetros de movimiento para obtener la posición de los cambios de eventos para todo el conjunto de la secuencia.

Como se muestra en la Tabla 2-3 en caso de un movimiento de traslación constante, el *time stamp* de los eventos se incrementa linealmente, en un *roll* o en ausencia de movimiento no se asignan eventos debido a que no hay cambio de contenido, y en un *zoom* el *time stamp* de los eventos se incrementa exponencialmente.

Modelo de movimiento de cámara	$\begin{bmatrix} d_x \\ d_y \end{bmatrix} = \begin{bmatrix} a_1 & a_2 \\ b_1 & b_2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} a_0 \\ b_0 \end{bmatrix}$	d_x, d_y son las componentes horizontales y verticales respectivamente de los vectores de movimiento y x, y las coordenadas del plano de la imagen, todas ellas medidas en píxeles.
Otros parámetros	W, H son respectivamente el ancho y alto del cuadro. T_f es el intervalo de tiempo (medido en cuadros) correspondiente a los vectores de movimiento disponibles. F_n es la fracción novedosa del cuadro.	
Esquema de movimiento	*Pan, Tilt, Track, Boom*	*Zoom*
Suposiciones	$a_1 = a_2 = b_1 = b_2 = 0$, $a_0, b_0 \neq 0$	$a_1 = b_2 \neq 0$, $a_2 = b_1 = a_0 = b_0 = 0$
Time stamps de los eventos (t_i medidos en cuadros)	$t_i \in \square \Big/ i \cdot F_n \leq \left\lvert \left(\frac{a_0}{W} + \frac{b_0}{H} \right) \frac{t_i}{T_f} - a_0 b_0 \left(\frac{t_i}{T_f} \right)^2 \right\rvert$	$t_i \in \square \Big/ i \cdot F_n \leq \left\lvert 1 - (a_0 + 1)^{\frac{2t_i}{T_f}} \right\rvert$

Tabla 2-3 Resumen de los criterios empleados para guiar la selección de *keyframes* en el nivel 3

Nivel 4: Selección basada en la Actividad de Movimiento

El objetivo de este nivel está orientado a la selección de cuadros en un segmento de la secuencia delimitado por los cuadros seleccionados del nivel anterior. El criterio de selección de *keyframes* para este nivel se basa en un submuestreo lineal de la curva de actividad acumulada [23].

Primeramente, una vez alcanzado el nivel 3, el algoritmo encargado de estimar la actividad de movimiento, empieza calculando la característica definida según la siguiente formula:

$$I(n) = \frac{1}{N} \sum_{i=0}^{N-1} \left\| \vec{m}_i - \vec{c}_i \right\|^2$$

donde $\vec{c}_i$ es la componente del vector $\vec{m}_i$ debida al movimiento de cámara. De esta manera, se obtiene la curva de actividad acumulada la cual es interpolada para mostrar el valor de todos los cuadros del segmento.

Cuando dicha curva alcanza un determinado límite (umbral dinámico en función de los parámetros del Control de Adaptación), como se aprecia en la Figura 2-17 se selecciona el cuadro y el proceso empieza a acumular la actividad de nuevo.

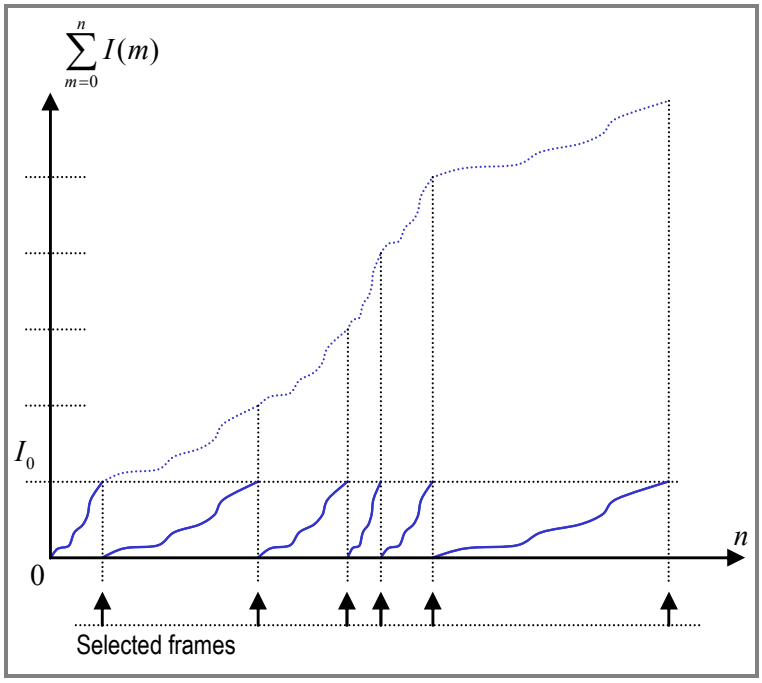

Figura 2-17 Criterio de selección de *keyframes* basado en la subdivisión lineal de la cantidad de actividad de movimiento acumulada

3 Diseño e Implementación

Este capítulo se centra en el diseño del API necesario para hacer posible la integración del codec aceSVC en la arquitectura del RC-CAT.

En primer lugar se ha llevado a cabo un estudio de la generación de un vídeo en formato aceSVC y de la librería proporcionada por la QMUL, la cual permite la extracción de las características de este domino comprimido.

Posteriormente, analizada la arquitectura del RC-CAT en la sección 2.3, se presentan los cambios que se han introducido en dicha arquitectura para conseguir integrar el nuevo codec a la herramienta.

Por último, una vez estudiada la librería de la QMUL, se presenta un enfoque alternativo para conseguir integrar el codificador escalable debido a las diversas limitaciones encontradas en la librería.

3.1 Generación de un vídeo aceSVC

Para codificar un vídeo al formato aceSVC, es necesario definir un fichero donde se especifican los parámetros de configuración, los cuales determinarán la manera en la cual el vídeo será codificado. Dicho fichero ha de tener una extensión .par, para que sea reconocible por el *encoder* de aceSVC.

A continuación, se presenta una breve descripción de los parámetros de configuración descritos en [10].

3.1.1 Parámetros

La codificación de un vídeo en aceSVC requiere la especificación de una serie de parámetros que conformarán la manera en la cual el vídeo será codificado. Dichos parámetros, que se encuentran especificados en la Tabla 3-1, se clasifican en parámetros obligatorios y opcionales.

Los parámetros que a continuación se van a presentar corresponden a aquellos que permiten definir las transformadas y los *transport stream* específicos de cada resolución en el proceso de codificación. En concreto, se van a detallar los parámetros ***MotionParameters***, ***Decomposition***, los cuales se definen por medio de otro subconjunto de parámetros, y el ***TargetDecPoints.***

El resto de parámetros de la Tabla 3-1, debido a que son auto explicativos y a que no requieren un subconjunto adicional de parámetros para su especificación, no se ha considerado necesario explicarlos en mayor detalle.

OBLIGATORIOS		**OPCIONALES**	
Nombre	***Descripción***	***Nombre***	***Descripción***
SourceVideoFile	Ruta o *path* de la secuencia de imágenes (en formato yuv) a codificar	MaxGOPSize	Tamaño máximo del GoP
FrameWidth	Ancho del vídeo	FramesToCompress	Número de *cuadros* a codificar
FrameHeight	Alto del vídeo	ThreshUpdate	Empleo de un umbral temporal de actualización en caso de usar filtros temporales con paso de actualización
FrameRate	Imágenes por segundo	OverlappedBlocks	Uso de bloques superpuestos en predicción temporal
Decomposition	Transformadas *wavelet* empleadas en la codificación	MotionParameters	Parámetros de movimiento. Son valores prefijados
TargetDecPoints	Necesarios para la adaptación de un TS específico. Ver Tabla 3-3		
CompressedVideoFile	*Path* del vídeo comprimido resultante		

Tabla 3-1: Parámetros de configuración de un vídeo aceSVC

Existen diferentes maneras de definir la forma en la cual las transformadas temporales y espaciales se llevarán a cabo, debido a que en cada descomposición temporal se especifican unos parámetros de movimiento distintos. En concreto, para la transformada temporal se define el parámetro ***MotionParameters***, compuesto por un conjunto de parámetros de movimiento que definen la descomposición a realizar. Este parámetro se especifica según la siguiente estructura:

> MotionParameters = DecompositionLabel: LambdaTemporal, MacroblockSize, MotionEstimationLev, {1st set of MotionLevParams}, {2nd set of MotionLevParams},…

De todos ellos, los parámetros obligatorios son:

- ***DecompositionLabel***: etiqueta usada para asociar a una o más descomposiciones temporales el parámetro de movimiento correspondiente.
- ***LambdaTemporal***: parámetro (número entero) para la optimización de relación tasa/distorsión entre textura e información de movimiento. Rango: $0 < LambdaTemporal < 1000$.

El resto de parámetros son sólo obligatorios para la primera descomposición temporal asociada a una resolución espacial, y no son considerados en los niveles temporales más altos de dicha resolución espacial, ya que para estos niveles temporales se derivan automáticamente de la primera descomposición temporal. Estos parámetros son:

- ***MacroblockSize***: tamaño del macrobloque. Rango: $MacroblockSize \in \{4, 8, 16, 32, 64, 128, 256\}$
- ***MotionEstimationLev***: niveles de estimación de movimiento. Rango: $MotionEstimationLev > 0$
- ***MotionLevParams***: conjunto de parámetros de estimación de movimiento para cada nivel. Por cada nivel de estimación de movimiento, se definen el siguiente subconjunto de parámetros:
  - ***MotionPrecision*:** precisión de los vectores de movimiento para el nivel estimado (0 – precisión entera, 1 - precisión de medio píxel, 2 - precisión de cuarto de píxel, 3 - precisión de 1/8 de píxel, 4 - precisión de 1/16 píxel, 5 precisión de 1/32 de píxel)
  - ***MotionSearchRange*:** ventana de búsqueda para la estimación del movimiento, definida como la distancia más lejana en píxeles (+/-) al bloque. Rango: $0 \leq MotionSearchRange \leq 64$
  - ***TreeLevel*:** hasta qué nivel de profundidad el árbol de movimiento puede crecer según el nivel de estimación de movimiento definido. El tamaño del subbloque más pequeño que se puede generar viene definido por la siguiente fórmula

$$Subbloque_{mas\_pequeño} = \frac{MacroblockSize}{2^{TreeLevel-1}}$$

    Por semejanza con MPEG donde los macrobloques son de 16x16 píxeles, si se define el valor del *MacroblockSize* igual a 128, para obtener un subbloque de 16x16, el valor de *TreeLevel* tendría que ser 4 ($\frac{128}{2^{4-1}} = 16$).

    Rango: $0 < TreeLevel \leq 8$
  - ***Growing*:** indica si se fuerza el crecimiento del árbol de vectores hasta los niveles *TreeLevel*.

El parámetro **Decomposition** define las series de transformadas *wavelet*, tanto espaciales como temporales, que guiarán el proceso de codificación. La estructura *decomposition* es una estructura recursiva de manera que las transformadas pueden realizarse sobre las subbandas resultantes paso bajo (L) y paso alto (H). Este parámetro consta de 5 elementos:

1. La subbanda afectada (paso bajo (L), paso alto (H)).
2. El tipo de transformada (espacial (S), temporal (T), buffer espacial (SB) o buffer temporal (BT)) a realizar.
3. El tipo de implementación de la transformada (*LIFTING*, *FILTERBANK* o *MDAT LIFTING*). Las transformadas temporales sólo utilizan la implementación *lifting*, mientras que las transformadas espaciales pueden emplear cualquiera de los 3 tipos de transformadas *wavelet* mencionadas.
4. El filtro de la transformada (definido en la Tabla 3-2).
5. El parámetro ***MotionParameter*** si se trata de una descomposición temporal o el número de descomposiciones en caso de ser una descomposición espacial.

TIPO	ETIQUETA
Cohen-Daubechies-Feauveau 9/7	9x7
Wavelet LeGall 5/3 simétrico	5x3
Wavelet LeGall 5/3 con delta paso bajo (wavelet 1/3 o wavelet LeGall sin paso de actualización)	1x3
Wavelet Haar 2/2	Haar
Wavelet Haar con delta paso bajo (wavelet 1/2 o Haar sin paso de actualización)	1x2
Filterbank. Implementación de un banco de filtros wavelet 3LS (3 lifting steps)	3LS

Tabla 3-2: Filtros *wavelets* implementados en la librería

Un comando típico para una transformada espacial es el siguiente:

L: S,LIFTING,9x7,3;

donde L identifica el tipo de subbanda (paso bajo), S el tipo de transformada a realizar (espacial), LIFTING implica el tipo de implementación *wavelet* a realizar, 9x7 se refiere al filtro *wavelet* a usar en este caso el de Cohen-Daubechies-Feauveau 9/7 y 3 indica el número de descomposiciones espaciales a realizar.

El siguiente comando ilustra la manera en la cual se pueden definir las transformadas temporales:

H: T,LIFTING,1x3,<T21,T22>,2;

H, T, LIFTING y 1x3 tienen el mismo significado que en la transformada espacial, mientras que <T21,T22> son etiquetas para los 2 conjuntos de parámetros de movimiento relacionados con los 2 niveles de descomposición temporal.

Por último, el parámetro **TargetDecPoints** permite definir los múltiples *bit-rates* (y por tanto calidades) asociados a un único *stream*. La Tabla 3-3 muestra los *bit-rates,* preestablecidos por la QMUL, asociados a cada tipo de resolución.

T (RESOLUCIÓN TEMPORAL)	**S (RESOLUCIÓN ESPACIAL)**			
	4CIF (704x576)	***CIF (352x288)***	***QCIF (176x144)***	***QQCIF (88x72)***
7.5	512 640 768 896 1024	192 224 256 320 384	64 80 96 112 128	26 34 42 50 58
15	768 896 1024 1280 1536	256 320 384 448 512	96 112 128 160 192	42 50 58 66 74
30	1024 1280 1536 1792 2048	384 448 512 640 768	128 160 192 224 256	58 66 74 82 90
60	1536 1780 2048 2560 3072			

Tabla 3-3: Puntos de decodificación (*bit rates* [kbps]) para diferentes resoluciones temporales (T) y espaciales (S)

3.2 Cambios en la arquitectura del RC-CAT

Inicialmente, el diseño y desarrollo del RC-CAT sólo estaba pensado para los dominios MPEG-1 y MPEG-2. No obstante, el objetivo de este PFC consiste en explorar el dominio comprimido aceSVC y ver que características similares a MPEG-1/2 se pueden extraer. De esta forma se puede establecer una comparativa de eficiencia de los algoritmos entre ambos dominios.

Para la integración de aceSVC en el RC-CAT ha sido necesario modificar la arquitectura previa de la herramienta definida en la sección 2.3 (ver Figura 2-13).

Como se comentó anteriormente, dentro del módulo análisis de contenido se encuentra integrada la extracción de las características básicas del dominio MPEG-1/2. De este modo, para integrar un nuevo codec, en nuestro caso aceSVC, se ha implementado en primer lugar un decodificador parcial que permita, al igual que con MPEG, extraer las imágenes de baja resolución y los vectores de movimiento para a continuación asociar esta información a una estructura reconocible por los algoritmos de análisis.

De esta manera, cuando se solicite al RC-CAT una operación de adaptación, la extracción de estas características para los vídeos MPEG y aceSVC se realizarán en submódulos distintos, englobando la información necesaria en una estructura única determinada para conseguir que los algoritmos de análisis extraigan información semántica independientemente del codec utilizado.

La Figura 3-1 amplía la arquitectura del RC-CAT para mostrar su comportamiento cuando se usa como parámetro de entrada un vídeo aceSVC.

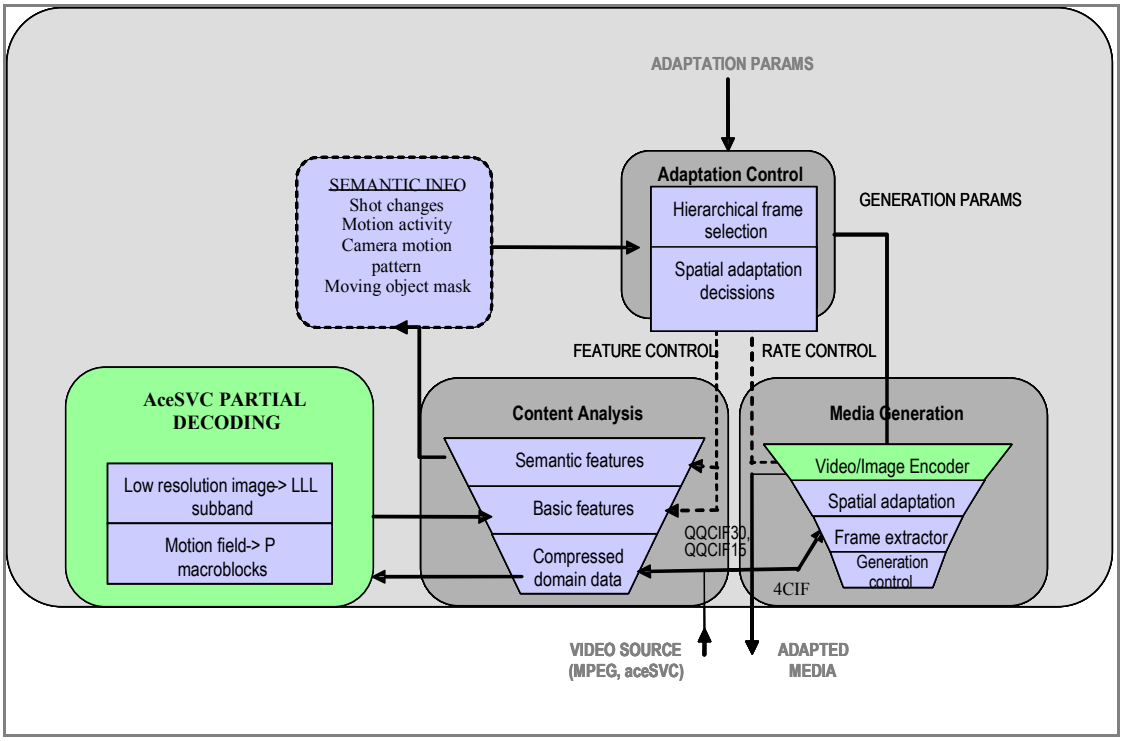

Figura 3-1: Arquitectura final del RC-CAT con el codec aceSVC integrado

En este caso, el nuevo módulo, *aceSVC Partial Decoder*, presenta un doble objetivo:

1. Obtener las características básicas del codec aceSVC tales como los cuadros de baja resolución de todas las escalas espaciales definidas anteriormente en los parámetros de configuración y el campo de vectores de todos los niveles temporales con resolución de macrobloque.

2. Englobar la información anterior en la estructura *compressed frame* empleada por los algoritmos de análisis para la obtención de las características semánticas.

En segundo lugar, para conseguir adaptar la secuencia a las necesidades del usuario final teniendo en cuenta los resultados obtenidos en los algoritmos de análisis, se ha implementado un nuevo módulo en el generador de medios (localizado dentro del *Video/Image Encoder*), denominado *VideoReaderSVC*, encargado de especificar las características de adaptación de los cuadros de la secuencia de entrada.

Este módulo se encargará de combinar la información resultante de los algoritmos de análisis, las máscaras de objetos y las *keyframes*, conjuntamente con la información obtenida del decodificador parcial (píxeles en formato YUV y vectores de movimiento) en una estructura específica, *AVFrame*, para que el generador de medios adapte cada cuadro de la secuencia a la resolución espacial y temporal deseada, al tiempo que éstos son analizados por el módulo análisis de contenido.

En las siguientes secciones se detallará cada uno de los módulos introducidos dentro de la arquitectura del RC-CAT (*aceSVC partial decoding* y *VideoReaderSVC*) para poder integrar el nuevo dominio comprimido estudiado.

3.2.1 AceSVC Partial Decoding

El modulo de análisis de contenido es el encargado de extraer la información básica relevante de la fuente de vídeo de entrada descrito en la sección 2.3.3. La extracción de estas características se realizan de forma *online* mediante técnicas que operan en los datos del dominio comprimido.

Cuando una adaptación es requerida, dependiendo del formato de la secuencia de entrada, se utilizará un decodificador parcial específico que permitirá la extracción de las características antes mencionadas. Estas características serán analizadas por los algoritmos de análisis con el fin de extraer los parámetros de adaptación requeridos.

Como se presenta en la Figura 3-1, para la integración del codec aceSVC se ha diseñado un decodificador parcial, *aceSVC Partial Decoder*, que permite que los algoritmos de análisis actúen de manera transparente al codec empleado.

Las características básicas obtenidas por medio del decodificador parcial se encapsulan en una estructura determinada (ver Figura 3-2), denominada *cframe* (*compressed frame*), que los algoritmos de análisis emplearán para la extracción de características temporales y espaciales para la posterior adaptación.

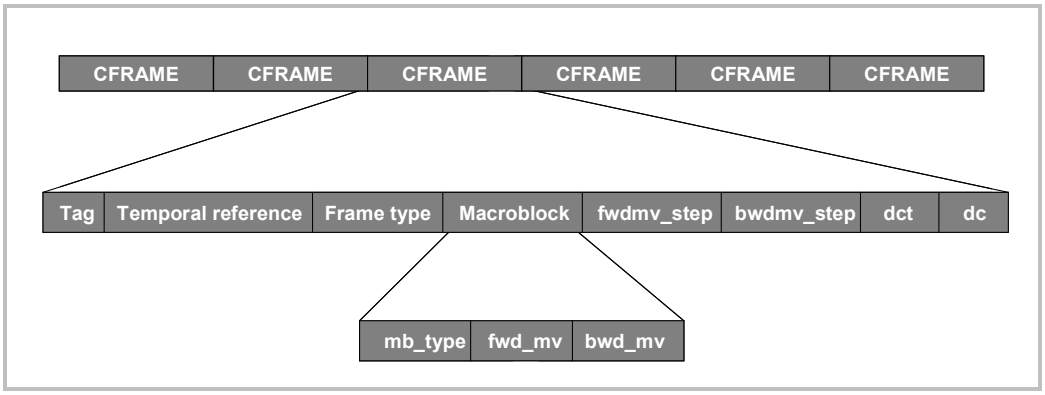

Figura 3-2 Estructura detallada del cuadro comprimido (*cframe*)

La Figura 3-3 muestra a modo de ejemplo, el resultado de codificar una secuencia aceSVC con cuatro capas temporales, utilizando predicción bidireccional, un GoP de 8 cuadros y una tasa de cuadros de 25 fps.

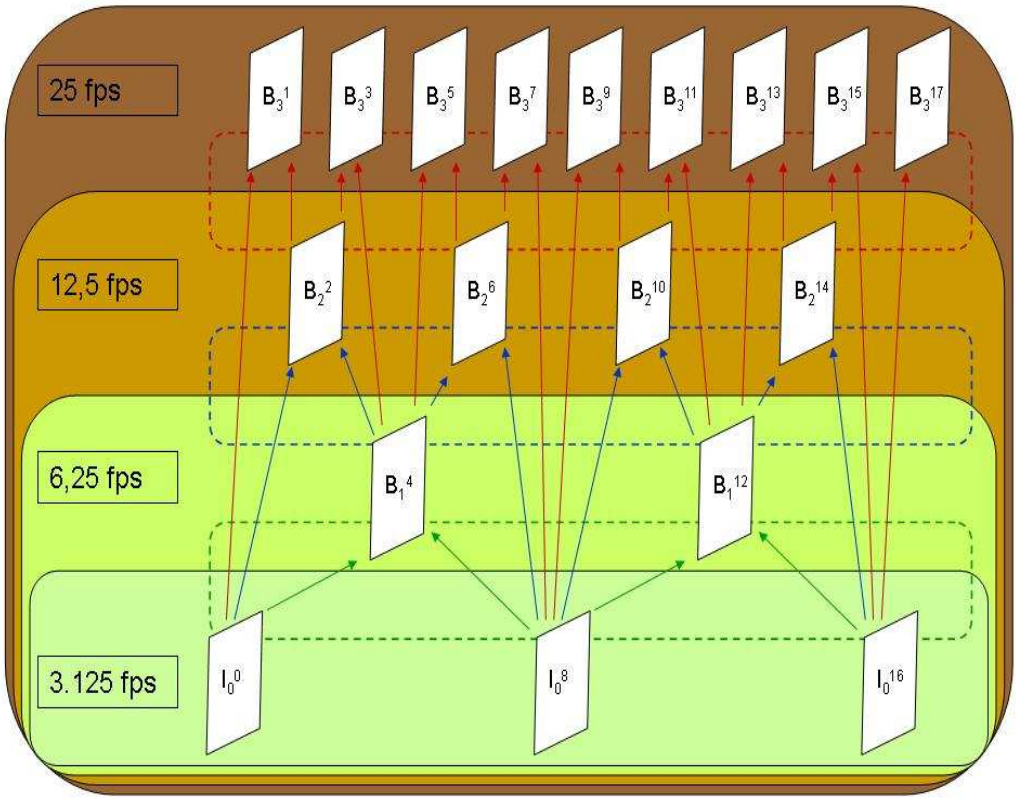

Figura 3-3 Composición del GoP a diferentes resoluciones temporales usando predicción bidireccional y cuatro capas temporales

Como puede observarse, cabe destacar que los cuadros que componen el GoP (excepto el primero) o todos presentan predicción bidireccional o todos son unidireccionales (en el caso de codificar la secuencia con predicción unidireccional, el cual no es el detallado en la figura).

A partir de este momento, y por semejanza con los codificadores híbridos se ha decidido denominar a los cuadros sin información de movimiento como cuadros I, a los cuadros con predicción bidireccional como B y a los cuadros con predicción unidireccional como cuadros P. Conviene resaltar que aunque en MPEG un cuadro B puede tener predicción unidireccional en aceSVC esto no es posible, es decir, siempre tendrá predicción bidireccional.

Una vez aclarado la notación empleada en los cuadros, en la Tabla 3-4 se detallan los cuadros descomprimidos a la resolución temporal más alta, 25 cuadros por segundo, cuando el tipo de codificación empleada en la secuencia de entrada es unidireccional o bidireccional, este último es el detallado en la Figura 3-3:

TIPO DE CODIFICACIÓN	CUADROS DESCOMPRIMIDOS
Unidireccional	$I_0^0 P_3^1 P_2^2 P_3^3 P_1^4 P_3^5 P_2^6 P_3^7$
Bidireccional	$I_0^0 B_3^1 B_2^2 B_3^3 B_1^4 B_3^5 B_2^6 B_3^7$

Tabla 3-4 Cuadros descomprimidos a la resolución temporal más alta de un GoP de 8 cuadros para las dos maneras de codificación de la secuencia de entrada

El subíndice de los cuadros descomprimidos indica el nivel de la capa temporal de la cual se extrae el cuadro, siendo 0 la capa temporal más baja, es decir, si un vídeo con resolución 4CIF a 25 cuadros por segundo se codifica con 4 niveles temporales, la capa temporal más baja corresponde a la capa cuya tasa de cuadros por segundo es de 3.125 fps; y el superíndice el orden de los cuadros en el *bitstream*.

Al contrario que MPEG, la librería aceSVC devuelve los cuadros decodificados en el mismo orden temporal de la secuencia dentro del GoP, por tanto, el campo *temporal reference* de la estructura *cframe* coincidirá con el número de cuadro de la secuencia.

El tipo de cuadro, que tomará valor 1 para cuadros tipo I, 2 para B y 3 para P, se define mediante el parámetro *frame type*. En el ejemplo ilustrado en la Tabla 3-4 será de tipo I el primer cuadro del GoP y tipo B (o P en caso de que el vídeo se haya codificado con predicción unidireccional) para el resto de cuadros. En este contexto aunque en MPEG un cuadro B puede tener predicción unidireccional no se considera como tipo 3, sino como tipo 2.

La especificación del campo de vectores se realiza por medio de cinco campos, de los cuales los 3 primeros se encuentran en un buffer que contiene la información del macrobloque (campo *macroblock*):

- *fwd_mv* y *bwd_mv*: identifican los vectores de movimiento *forward* y *backward* respectivamente.
- *mb_type*: Indica el modo de codificación del macrobloque: *intra, pattern, motion bck, motion fwd*.
- *fwdmv_step* y *bwdmv_step*: se corresponden con la distancia temporal a la referencia anterior y posterior respectivamente.

  Como puede observarse en el GoP de la Figura 3-3 y Figura 3-4 las distancias son potencia de 2, es decir, la distancia de los cuadros que aparecen en la capa 25 fps (B_3^1 B_3^3 B_3^5 B_3^7) con sus respectivas referencias (por ejemplo, I_0^0 y B_2^2 son las referencias de B_3^1, y B_2^2 y B_1^4 son las referencias de B_3^3) presentan una distancia de 1 cuadro mientras que los cuadros que aparecen en la capa 12.5 fps (B_2^2 B_2^6) con sus respectivas referencias (I_0^0 y B_1^4 son las referencias de B_2^2, y B_1^4 y I_0^8 son las referencias de B_2^6) presentan una distancia de 2 cuadros.

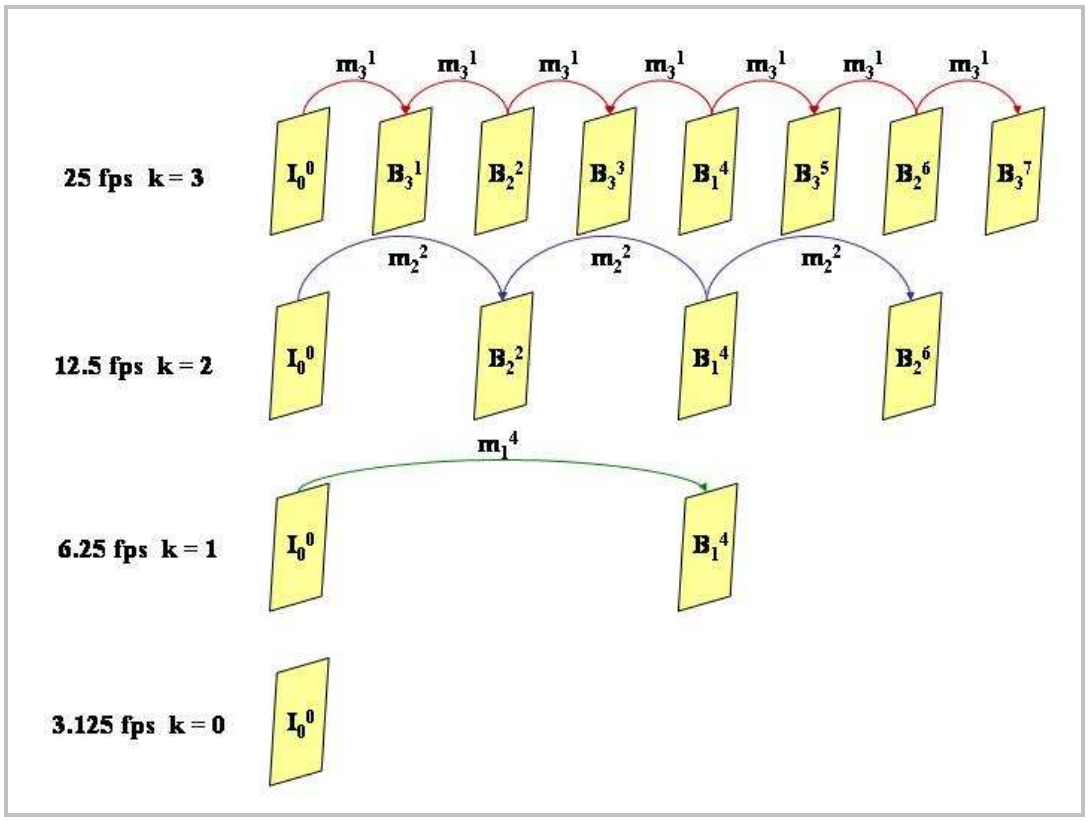

Figura 3-4 Niveles de los vectores de movimiento en aceSVC para un GoP de 8 cuadros. m_k^i corresponde con la distancia i^{th} en cuadros con respecto a la referencia del nivel de escala temporal k^{th}.

En comparación con MPEG, una secuencia típica de un GoP de 12 cuadros presenta la siguiente estructura IBBPBBPBBPBB. En este caso, los saltos varían de un cuadro a otro, tal como expone en la Figura 3-5, siendo la distancia más pequeña de 1 (m=1) y la más alta de 3.

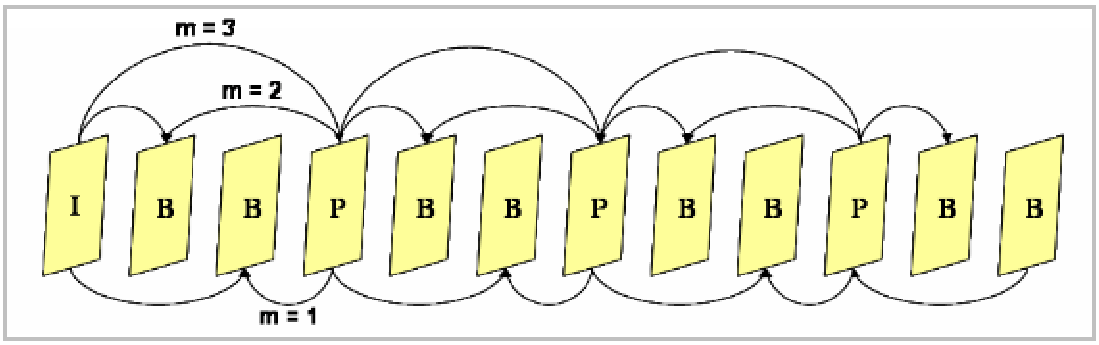

Figura 3-5 Estructura típica de un GoP de 12 cuadros en MPEG

Finalmente, el campo dct y dc se corresponde con la lista de coeficientes dct y dc respectivamente. Para este último, por analogía con MPEG (YCrCB) en el cual el tamaño de los coeficientes es 8x8 veces menor que la imagen original, en aceSVC se obtienen a partir de los píxeles YUV de la subbanda LLL.

Una vez rellena la estructura, el comportamiento del resto de módulo de análisis será independiente del dominio empleado, ya que los algoritmos de análisis sólo operan con la información almacenada en *cframe*.

3.2.2 VideoReaderSVC

Cuando una adaptación es requerida, el modulo de control de adaptación (*Adaptation Control*) se encarga de dirigir todo el proceso de adaptación.

A partir de los parámetros de entrada, el vídeo y los parámetros de adaptación que definirán el tipo de adaptación temporal y espacial a realizar, este módulo realizará una petición al módulo análisis de contenido, mientras haya cuadros por analizar, para que extraiga un cuadro del vídeo comprimido. A continuación, se analizará el cuadro y se determinarán las características semánticas definidas en la sección 2.3.3.3 necesarias para la adaptación.

Una vez obtenidas estas características, el módulo control de adaptación decidirá en función de éstas y de los parámetros de adaptación, si el cuadro extraído será incluido o no en el fichero de salida (adaptación temporal) y la manera en la cual será modificado (determinado por el tipo de adaptación espacial especificado en la sección 2.3.4.1).

Así por ejemplo, en el caso de no requerirse una adaptación temporal, todos los cuadros de la secuencia de entrada se adaptarán en función del resto de parámetros y formarán parte del fichero de salida (reduciendo su resolución, su calidad, etc.). Por el contrario, cuando se especifique la realización de una adaptación temporal, sólo los cuadros clave determinados por el módulo control de adaptación, según el método explicado en la sección 2.3.5.1, formarán parte del contenido final.

Para realizar la adaptación de los cuadros de la secuencia, el RC-CAT hace uso de la librería de código abierto *libavcodec* (librería para la codificación y decodificación de vídeo y audio). Así, para construir el *stream* de vídeo de salida es necesario especificar una serie de valores, detallados en la Tabla 3-5, dentro de la estructura *AVFrame* (estructura perteneciente a la librería *libavcodec*), para que el modulo *videoWriter* del RC-CAT genere el fichero adaptado.

Finalmente, con esta información y en función de los parámetros de generación, se codificará cada cuadro y se adaptará según las especificaciones deseadas.

La Figura 3-6 ilustra a modo de resumen el diagrama de bloques de todo el proceso de adaptación y cómo el módulo control de adaptación guía todo el proceso de adaptación. Mientras haya cuadros por analizar, este módulo indica al analizador de contenido la extracción y el posterior análisis de un cuadro de la secuencia. Del resultado del análisis y de las especificaciones del usuario, el control de adaptación decide si finalmente el cuadro analizado se adapta, por medio del generador de medios, a dichas especificaciones.

PARAMETRO	DESCRIPCIÓN
pix_fmt	Modo de codificación de los cuadros. Ha sido prefijado a PIX_FMT_YUV420P
data	Planos del cuadro (YUV). Estos plano, obtenidos en el decodificador parcial, son adaptados previamente espacialmente en caso de requerirse
pict_type	Tipo de cuadro (I, P o B)
pts	*Timestamp*
coded_picture_number	Posición del cuadro en el *bitstream*
display_picture_number	Posición de visualización del cuadro
interlaced_frame	Indica si el contenido de la imagen es entrelazado
top_field_first	Indica, si el campo es entrelazado, si el campo superior aparece primero
pan_scan	Modifica la relación de aspecto de la imagen sin deformarla
keyframe	Indica si el cuadro es un cuadro clave

Tabla 3-5 Parámetros necesarios para la adaptación de una secuencia

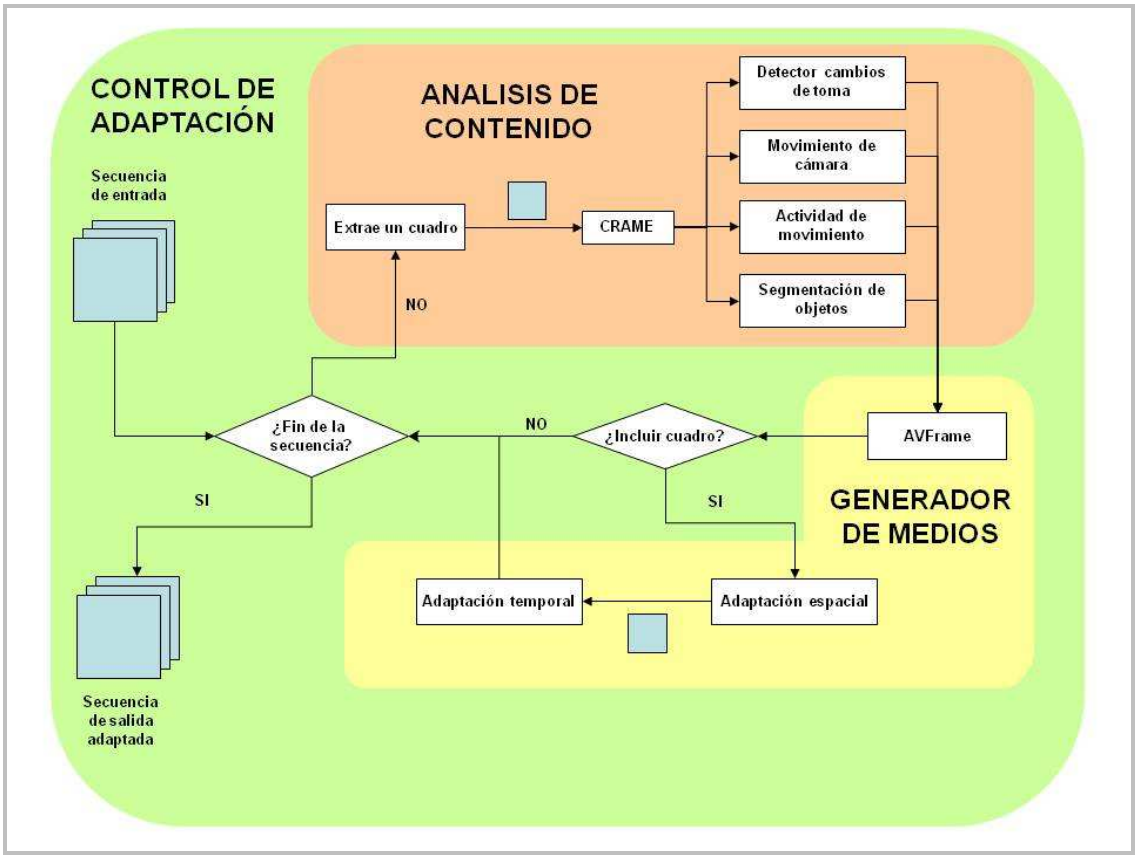

Figura 3-6 Diagrama de bloques del proceso de adaptación de una secuencia

3.3 Análisis de la librería mmvsvc

Como se ha mencionado anteriormente, la QMUL ha desarrollado una librería con el fin de hacer posible la integración del dominio aceSVC dentro del módulo de análisis del RC-CAT.

La librería básicamente consiste en una clase *SVCDecoderAPI* en la cuál se encuentra encapsulado el decodificador de aceSVC. Esta clase se encarga de controlar una serie de *buffers* antes de que cualquier módulo externo haga uso de la información extraída. Los *buffers* son los siguientes:

- *Buffer* de cuadros encargado de almacenar un cuadro del GOP en formato YUV.
- Buffer para los vectores de movimiento del GOP. Almacena los vectores de movimiento de todos los cuadros paso alto (M/2) de la capa temporal más alta del GOP.

Como se presenta en la Figura 3-7 (a), cuando una secuencia de vídeos es codificada en formato aceSVC, cada cuadro paso alto que la compone se divide en N bloques, donde N viene definido de la siguiente manera:

$$N = \frac{FrameWidth * FrameHeight}{Subbloque_{mas\_pequeño}}$$

Cada uno de estos N bloques contiene información sobre los vectores de movimiento en ambas direcciones (*forward y backward*) y están compuestos por dos componentes, una por cada dirección (MVx y MVy).

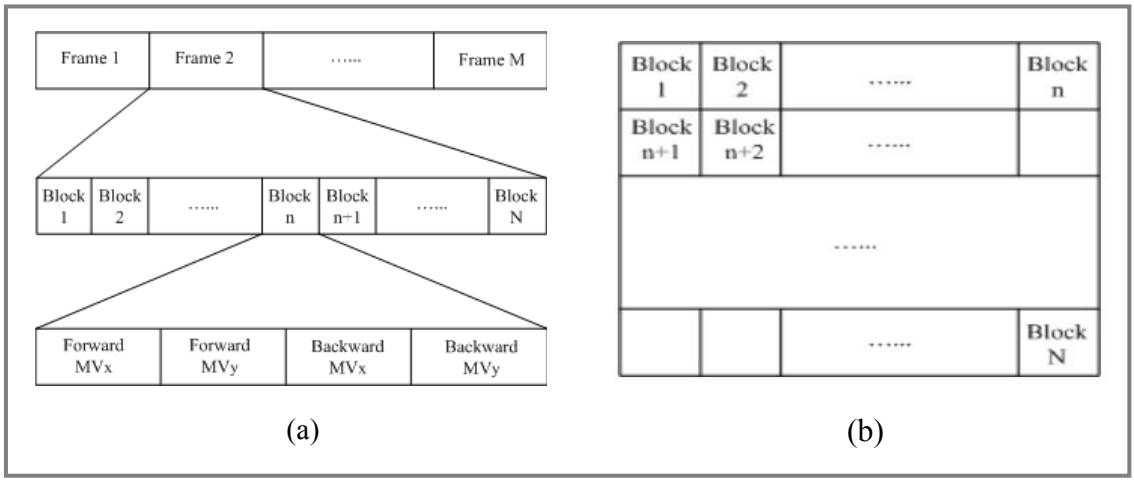

Figura 3-7 (a) Estructura del buffer de almacenamiento de los vectores de movimiento (donde M es el número de cuadros paso alto en el GoP). (b) Orden de los bloques en un cuadro.

La finalidad de esta librería, útil para aplicaciones donde se requiere interacciones con el proceso de decodificación o la extracción de datos durante la decodificación de aceSVC, consiste en la obtención de las características básicas del vídeo. Para ello, la librería proporciona los siguientes métodos:

- initDecoder(char *infileName, char *outfileName): inicializa el decodificador y los *buffers*. Toma como entrada un vídeo aceSVC (infileName) y almacena las imágenes decodificadas en un fichero de salida (outfileName) conformando un vídeo en formato YUV.
- DecodeFrame(): decodifica un cuadro del GOP.
- GetAFrame(): devuelve el puntero donde se encuentra almacenado el cuadro decodificado.
- GetFrameWidth(): devuelve el ancho de un cuadro del vídeo.
- GetFrameHeight(): devuelve el alto de un cuadro del vídeo.
- GetGOPSize(): devuelve el tamaño del GOP.
- GetMvOfGOP*(int32* frames, int32* blocksize)* : devuelve el puntero donde están almacenados los vectores de movimiento del GOP, así como el número de imágenes con vectores de movimiento (M/2) y el tamaño más pequeño del subbloque definido (*blocksize)*.
- closeDecoder*():* finaliza el decodificador liberando la memoria reservada de los *buffers*.

En definitiva, la librería proporciona:

- Información de codificación: tamaño del vídeo y tamaño del GOP
- Información de los cuadros de la capa espacial más alta del vídeo en formato YUV. Esta información se utilizará como entrada al detector de cambios de toma, primer nivel en la estimación de *keyframes* descrito en la sección 2.3.5.1.
- Información, durante el proceso de decodificación, de los vectores de movimiento de capa temporal más alta normalizados al subbloque más pequeño.

  Los vectores de movimiento se utilizarán para estimar el resto de niveles de detección de *keyframes*, es decir, como entrada a los resultados del análisis de los algoritmos de estimación del movimiento de cámara y la actividad de movimiento.

  En este sentido hay que resaltar que tal como se ilustra en la Figura 3-8 y teniendo en cuenta lo mencionado en la sección 2.2.1.1.1, la codificación de aceSVC utiliza la compensación de movimiento *MCTF* con estimación de movimiento jerárquico de tamaño variable (*HVSBM*) es decir, cada cuadro se divide en macrobloques de tamaño variable.

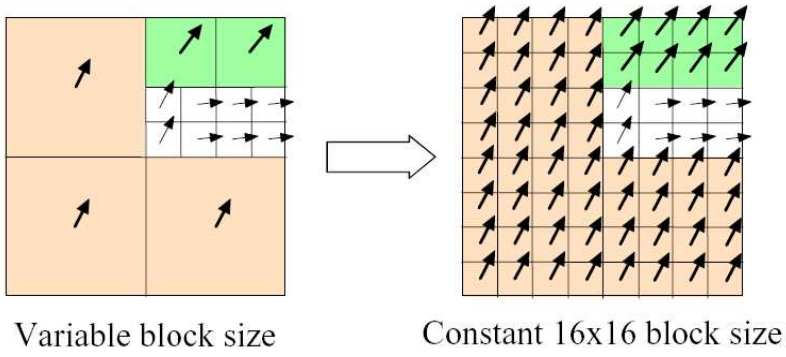

Figura 3-8 Vectores de movimiento devuelto por la librería mmvsvc en un cuadro compuesto por 16x16 bloques.

Sin embargo, la librería proporcionada, en lugar de devolver el tamaño de cada macrobloque con sus respectivos vectores de movimiento (figura de la izquierda), devuelve el campo de vectores normalizados al subbloque más pequeño definido en los parámetros de codificación como:

$$\frac{\text{MacroblockSize}}{2^{\text{TreeLevel}-1}}$$

A este respecto, si se define el tamaño de ***MacroblockSize*** igual a 64 y el parámetro ***TreeLevel*** igual a 3, los distintos macrobloques podrán tener un tamaño de 64x64 (color rosado de la figura de la izquierda), 32x32 (color verde) y 16x16 (color blanco). Como el subbloque más pequeño definido tiene un tamaño de 16x16 píxeles ($\frac{64}{2^{3-1}}$), los subbloques de 64x64 y de 32x32 son divididos en subbloques de 16x16 replicando el valor del campo de vectores en cada subbloque de 16x16, tal como se exhibe en la figura de la derecha.

3.3.1 Limitaciones

Actualmente, esta primera versión de la librería de aceSVC presenta una serie de limitaciones que han hecho que algunas de las ideas iniciales en el diseño de la integración del codec, sean irrealizables desde el punto de vista de la implementación. A continuación, se enumeran las limitaciones y cómo han influido éstas en el desarrollo del módulo de análisis.

- La librería no es capaz de extraer simultáneamente información de todas las capas espaciales, temporales y calidades de la secuencia aceSVC en las que se ha codificado sin realizar una previa extracción completa de la capa QTS deseada [16].

  A este efecto, sólo se pueden extraer los vectores de movimiento de una determinada resolución temporal, y por este motivo sólo los cuadros de la capa temporal más alta contienen información de movimiento.

Para ilustrar lo anterior, la Tabla 3-6 resume qué se puede extraer de una cierta capa temporal a partir de la estructura definida en la Figura 3-3. Para la resolución temporal más alta, 25 fps, los cuadros que componen el GoP son I_0^0 $\mathbf{B_3^1}$ B_2^2 $\mathbf{B_3^3}$ B_1^4 $\mathbf{B_3^5}$ B_2^6 $\mathbf{B_3^7}$ pero sólo los cuadros impares de este GoP, B_3^1 B_3^3 B_3^5 B_3^7, es decir, los cuadros de la capa superior (capa 25fps) de la Figura 3-3, presentan información de movimiento.

Si se quieren extraer los campos de vectores de una resolución temporal inferior, es necesario una extracción completa de la capa 12.5 fps. En este caso, el GoP resultante de la extracción estaría formado por los cuadros I_0^0 $\mathbf{B_2^2}$ B_1^4 $\mathbf{B_2^6}$ y de ellos, sólo los cuadros impares (cuadros de la capa 12.5 fps), B_2^2 B_2^6, presentarían información de movimiento.

Escala temporal	**Tasa de cuadro**	**Cuadros descomprimido**	**Vectores de movimiento extraídos**	**Distancia con respecto al cuadro de referencia**
3	25 fps	I_0^0 B_3^1 B_2^2 B_3^3 B_1^4 B_3^5 B_2^6 B_3^7	B_3^1 B_3^3 B_3^5 B_3^7	1
2	12.5 fps	I_0^0 B_2^2 B_1^4 B_2^6	B_2^2 B_2^6	2
1	6.25 fps	I_0^0 B_1^4	B_1^4	4
0	3.125 fps	I_0^0	-	-

Tabla 3-6 Información proporcionada usando el API de aceSVC a diferentes escalas temporales usando un GoP de 8.

Lo mismo sucede cuando se quiere extraer información de distintas capas espaciales. Únicamente se pueden extraer los píxeles YUV de la capa espacial más alta, de manera que si quiere extraer los cuadros YUV de una capa espacial inferior habría que realizar una extracción completa de dicha capa. Por ejemplo, si una secuencia con resolución 4CIF (704x576) se codifica con 4 capas espaciales, para extraer los píxeles YUV a la resolución QQCIF (88x72) es necesario extraer completamente la subbanda LLL para poder acceder a esta información.

Las distintas resoluciones de calidad de una determinada resolución espacio-temporal se define mediante el parámetro ***TargetDecPoints*** y al igual que en los casos anteriores, para poder acceder a una determinada capa de calidad, primeramente es necesaria una extracción completa de la capa QTS deseada.

Todo esto ha provocado que no se pueda obtener una estructura de GoP similar a la de MPEG (por ejemplo: IBBPBBPBBPBB), en la cual únicamente el primer cuadro no presenta información de movimiento.

Adicionalmente, sólo los vectores de movimiento de los cuadros impares presentan información de movimiento por lo que se pierde la funcionalidad del algoritmo de segmentación gruesa de hacer un seguimiento (*tracking*) del movimiento, obteniendo por tanto un peor funcionamiento sobre este dominio.

A su vez, como el algoritmo de detección de tomas trabaja con coeficientes dc, esta limitación también nos ha impedido obtener una resolución 8x8 veces inferior a partir de la secuencia de entrada sin necesidad de realizar una extracción previa de la subbanda LLL.

- Una solución a lo anterior, consistiría en trabajar en paralelo con varias versiones, capas QTS, de la secuencia de entrada consiguiendo así extraer al mismo tiempo información de las distintas capas a costa de obtener una disminución de la eficiencia en cuando al tiempo de decodificación.

  Sin embargo, esta solución se ha desestimado debido a la imposibilidad añadida de tener abiertas al mismo tiempo varias versiones de la secuencia, debido a las limitaciones de la librería derivadas de la compartición interna de recursos.

- Por último, la librería no es capaz de devolver los vectores de movimiento de una secuencia codificada con predicción unidireccional. En este sentido se ha decidido codificar los vídeos con predicción bidireccional.

  Como los algoritmos sólo trabajan con la información de movimiento de los cuadros P (como se detalla en la sección 2.3.3.3), a efectos de esta limitación se ha optado por definir los cuadros B como P desechando la información de los vectores de movimiento *backward.*

Estas limitaciones han provocado que la implementación se haya alejado con respecto a la idea inicial.

La solución de implementación que se detallará en la sección 3.4 consiste en dividir el análisis y la adaptación en 2 etapas, usando en cada una de ellas distintas versiones de la secuencia de entrada codificada con la arquitectura t+2D, donde se consigue que diferentes niveles espaciales de una determinada resolución temporal presenten el mismo conjunto de vectores de movimiento, al contrario que en MPEG.

Como consecuencia de este proceso de implementación, el RC-CAT pierde su principal característica, adaptación *online* de la secuencia, ya que es necesario procesar completamente tres versiones de la secuencia de entrada para conseguir la adaptación requerida.

3.4 Integración

Como se ha mencionado en la sección 3.3, la idea inicial consistía en extraer simultáneamente información de varias capas al mismo tiempo, pero debido a las limitaciones presentes en la librería aceSVC, esta idea se ha replanteado para conseguir una completa integración del aceSVC en el RC-CAT sacrificando la funcionalidad de operación *online* en dicho proceso.

El proceso completo de adaptación se realiza en 2 pasadas. En la primera de ellas, se extrae la información semántica del vídeo completo y en la segunda, a partir de la información anterior se generan los parámetros de adaptación para adaptar el contenido final. A continuación se muestran los módulos afectados y la manera en la cual se ha procedido para lograr la implementación.

3.4.1 Análisis de Contenidos

Para el módulo de análisis de contenidos, lo que se persigue es el acceso a información del dominio comprimido, en concreto los píxeles YUV y los vectores de movimiento, de manera que sean los argumentos de entrada (*inputs*) de los algoritmos de análisis.

Los algoritmos de análisis han de operar sobre ambos dominios (MPEG y aceSVC) de modo transparente: el detector de cambios de toma recibe como argumento de entrada los coeficientes dc de MPEG y el resto de algoritmos funcionan con la información de movimiento de los cuadros P.

Para establecer una analogía con los datos MPEG, en el cual los coeficientes dc tienen una resolución 8x8 veces menor que el tamaño original y la información de movimiento de los cuadros P tiene una predicción con un salto de 3 cuadros (tal como se muestra en la Figura 3-5), se han utilizado dos versiones diferentes de la secuencia original.

En el caso de que la secuencia original tenga una resolución 4CIF (704x576 píxeles) a 25 cuadros por segundo (4CIF-25), las dos versiones que se emplearán son la QQCIF (88x72 píxeles) a 25 y 12.5 cuadros por segundo (QQCIF-25 y QQCIF-12.5 respectivamente):

- Los coeficientes YUV son extraídos de la versión QQCIF-25 porque tiene una resolución 8x8 veces menor que la original (4CIF-25) y no se pierde información con respecto a ésta ya que poseen la misma resolución temporal (25 fps).

- El campo de vectores es obtenido a partir de los cuadros impares de la resolución temporal 12.5 fps para conseguir una mejor predicción de movimiento (salto de dos cuadros con respecto a la referencia) con respecto a la resolución temporal de 25 fps.

  Como las secuencias de análisis se han codificado con la arquitectura t+2D, este campo de vectores será extraído de la versión QQCIF-12.5 en lugar de la versión 4CIF-12.5 para conseguir aumentar la eficiencia en cuanto al tiempo de decodificación.

Esta solución de implementación obliga a realizar dos pasadas para obtener los parámetros de adaptación, empleando en cada una de ella una versión distinta de la secuencia de entrada.

En la primera de ellas se hace uso de los coeficientes YUV de la versión QQCIF-25 para la obtención y el posterior almacenamiento temporal de los cambios de toma de la secuencia en un fichero de texto plano auxiliar.

En la segunda pasada, empleando la información almacenada sobre los cambios de toma de la secuencia y la información de movimiento de la versión QQCIF-12.5, se obtienen y almacenan el resto de los parámetros de adaptación (cuadros clave y máscaras de objetos).

3.4.2 Adaptación de Contenidos

A partir de la idea expuesta previamente en la sección 3.2, para hacer posible la adaptación de contenidos, se ha modificado el módulo generador de medios para permitir la adaptación de los cuadros de la secuencia aceSVC (a través del *VideoReaderSVC*) de la misma manera que se realiza con vídeos MPEG.

La misma limitación que ha provocado que el análisis de la secuencia tenga que ser realizada en dos etapas distintas, está presente también en este apartado. Al no poderse extraer características temporales y espaciales de distintas capas, cuando se lee y analiza un nuevo cuadro al mismo tiempo que ésta se adapta en función de los parámetros de adaptación, ha obligado un re-diseño de la idea inicial.

Ahora, como el proceso de análisis y adaptación no pueden llevarse a cabo en una misma etapa, el proceso entero se ha dividido en dos partes. La primera de ellas, extrae la información necesaria para que el módulo de control de adaptación genere los parámetros de adaptación (mencionado en la sección 3.4.1). La segunda, utiliza la información de dichos parámetros para adaptar el contenido multimedia, es decir, cuando se llama al generador de medios para adaptar el contenido, éste antes de adaptarlo analiza la secuencia entera y a continuación adapta el contenido según el resultado del análisis.

Cualquier versión escalada del contenido puede ser empleada para la generación de la versión adaptada, aunque lo más común y por analogía a lo realizado con las secuencias MPEG, se utiliza la versión original, es decir, la versión con la máxima resolución temporal y espacial (4CIF-25).

4 Pruebas y resultados

De las múltiples versiones escaladas del contenido que se pueden incluir en un *bitstream* de aceSVC, sólo 3 de ellas marcadas con un + en la Tabla 4-1 (4CIF y QQCIF a 25 cuadros por segundo, y QQCIF a 12.5 cuadros por segundo) son utilizadas actualmente en el RC-CAT, según lo expuesto en el capítulo 3, para conseguir una adaptación completa de la secuencia de entrada.

La Tabla 4-1 muestra las diferentes versiones en las que se han codificado un vídeo en los experimentos, suponiendo que la secuencia original tiene una resolución 4CIF a 25 cuadros por segundo.

Por simplicidad se hará referencia a la resolución 4CIF como la original y a la versión QQCIF como la subbanda espacial LLL. Para los niveles temporales se usará el mismo criterio, siendo 25 cuadros por segundo la tasa de cuadros de la secuencia original, de manera que la versión con máxima resolución espacial y temporal se nombrará a partir de ahora como 4CIF-25.

TAMAÑO \ CUADROS POR SEGUNDO	25	12.5	6.25
4CIF (704x576)	*		
CIF	+		
QCIF	+		
QQCIF (88x72)	*	*	

Tabla 4-1 Versiones de codec aceSVC utilizadas en los experimentos. Las tres versiones necesarias para adaptar la secuencia están representadas por un *. El resto de versiones, marcadas por un +, únicamente se han empleado para testar el comportamiento de los algoritmos en distintas situaciones.

Como, según se ha visto con anterioridad, la librería de aceSVC no permite el acceso a la información de varias versiones simultáneamente, para extraer la información del dominio comprimido aceSVC se han empleado dos versiones diferentes de la subbanda LLL (25 y 12.5), marcadas con un * en la Tabla 4-1:

- Los píxeles en formato YUV son extraídos de la subbanda LLL a 25 cuadros por segundo debido a que tiene una resolución 8x8 veces menor que la original (4CIF-25) y a que no se pierde información con respecto a ésta ya que poseen la misma resolución temporal (25 cuadros por segundo). Así, se obtiene la misma resolución que los coeficientes DC en MPEG (8x8 veces menor que el tamaño de la secuencia de entrada).

- Como en la arquitectura t+2D diferentes niveles espaciales presentan el mismo conjunto de vectores de movimiento para una misma resolución temporal, por motivo de eficiencia, el campo de vectores se ha extraído de la subbanda LLL en lugar de la resolución original.

  Sin embargo, se ha empleado la resolución temporal inferior (12.5) a la máxima permitida (25) para conseguir que una mejor predicción del movimiento y por tanto un salto de 2 cuadros entre el cuadro estimado y el de referencia (ver Tabla 3-6 y Figura 3-3).

Para adaptar el contenido de la secuencia se puede emplear cualquier versión, aunque el caso más común y por semejanza con MPEG se ha aplicado la secuencia original (4CIF-25). Finalmente, las versiones CIF-25 y QCIF-25 únicamente han sido utilizadas para testar y comparar el funcionamiento del algoritmo de detección de tomas a diferentes escalas espaciales.

A continuación se presenta la comparativa de ambos codecs con respecto a la precisión en la detección de los cambios de toma, la actividad de movimiento, el movimiento de cámara y la adaptación espacial de una secuencia.

Las diferentes secuencias utilizadas en las pruebas han sido codificadas empleando la arquitectura t+2D con codificación bidireccional 5x3 en lugar de codificación unidireccional 1x2 con un subbloque de tamaño 16. El fichero de codificación empleado se encuentra especificado en el Anexo B.

A la vista de los expuesto en las secciones 2.3.3.3 y 3.4.1, los cuadros con información de movimiento obtenidos a partir de la librería de análisis, cuadros con predicción bidireccional (B), son tratados como cuadros con predicción unidireccional (P), es decir, se ha omitido la información de la estimación de movimiento hacia atrás (*bwd_mv*).

Los algoritmos han sido probados en un ordenador con procesador Pentium IV a 1.70 GHz y 1 GB de memoria RAM, equipo en el cual no se ha conseguido un procesado en tiempo real.

4.1 Cambios de Toma

La detección de cambios de toma está basada en la diferencia entre cuadros consecutivos para detectar cortes, usando el algoritmo descrito en [18].

Para obtener unos resultados eficientes sobre el dominio aceSVC, dicha diferencia se realiza sobre la capa espacial que permita emular los coeficientes dc de MPEG a la máxima resolución temporal (en nuestro caso se ha utilizado la versión QQCIF-25), mejorando la eficiencia y reduciendo así el tiempo de decodificación tal como muestran los resultados de la sección 4.5.

Los resultados experimentales han sido llevados a cabo sobre una secuencia 4CIF representativa de MPEG-7 Content Set [24][25][26] a 25 cuadros por segundo. Dicha secuencia se ha codificado con aceSVC utilizando la arquitectura t+2D, en la cual primero se realiza la transformación temporal y a continuación la espacial, con 4 descomposiciones temporales, 4 espaciales y 4 niveles de calidad.

La Tabla 8 muestra los resultados de la comparación de MPEG y dos versiones QQCIF de la secuencia News. Dicha tabla presenta el número de cuadros de la secuencia analizada (cuadros), el número de cambios de toma de la secuencia original (N), el número de tomas detectadas correctas (OK), las tomas no detectadas (NO), los falsos positivos (FP), el *recall* (R) y la precisión (P).

VERSIÓN	CUADROS	N	OK	NO	FP	R	P
QQCIF (88x72)	12500	51	46	5	16	0.9	0.74
QQCIF (88x72) Baja Calidad	12500	51	35	16	16	0.69	0.69
Coeficientes DC -MPEG- (88x72)	27400	97	96	1	2	0.99	0.98

Tabla 4-2 Comparación de los resultados obtenidos en los experimentos de la secuencia News entre el estándar MPEG y aceSVC

Adicionalmente, se ha testado el algoritmo con distintas resoluciones espaciales y de calidad, aun sabiendo que el algoritmo de detección de tomas no está preparado para ello. En estos casos, el argumento de entrada al algoritmo son los píxeles YUV de cada resolución testada. Los resultados obtenidos se encuentran recogidos en la Tabla 4-3.

VERSIÓN	CUADROS	N	OK	NO	FP	R	P
CIF (352x288)	12500	51	36	15	10	0.71	0.78
CIF (352x288) Baja Calidad	12500	51	32	19	26	0.63	0.55
QCIF (176x144)	12500	51	46	5	22	0.9	0.68
QCIF (176x144) Baja Calidad	12500	51	34	17	25	0.67	0.58

Tabla 4-3 Resultado de los experimentos de la secuencia News

Estos resultados de la Tabla 4-2 y Tabla 4-3 manifiestan que la resolución espacial afecta ligeramente en el rendimiento, además de que los resultados a resoluciones inferiores son ligeramente mejores que a mayor resolución. En cuanto a la escalabilidad de calidad, como era de esperar, los resultados de las versiones de baja calidad son mucho peores que los de alta calidad.

Comparando los resultados con los obtenidos con los coeficientes DC de MPEG, los cuales se encuentran descritos en [18] utilizando una muestra de 27400 cuadros en lugar de 12500, se aprecia que la precisión obtenida en la detección de los cambios abruptos es del 98% y un *recall* del 99%, por lo que se puede determinar que los resultados sobre aceSVC están bastante alejados de la precisión del algoritmo con vídeos MPEG.

La mayoría de los falsos positivos obtenidos en el experimento son debidos principalmente a los efectos de edición (ver Figura 4-1) a lo largo de la secuencia, sobre todo al principio de ésta. Secundariamente estos falsos positivos, aunque en menor cantidad, son producto de rápidos movimientos de cámara y a los efectos del flash de las cámaras de fotos.

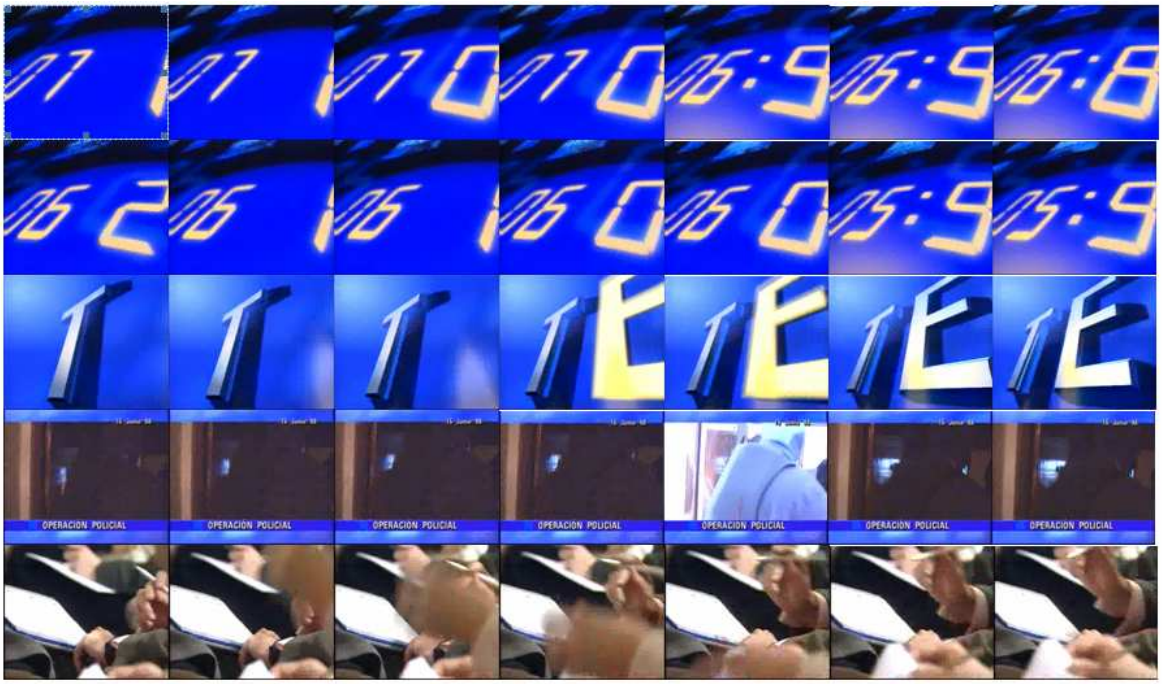

Figura 4-1 Selección de segmentos de secuencia que generan falsos positivos en el detector de tomas.

4.2 Actividad de Movimiento

La actividad de movimiento es una característica especialmente ligada a aspectos perceptuales, ya que ayuda a describir la *cantidad de acción* que muestra un segmento de vídeo. Esta característica es útil para tareas de indexación de vídeo o análisis semántico. Por tanto, el problema de la extracción de la actividad de movimiento ha sido explorado ampliamente en los últimos años y algunas de las aproximaciones existentes se han convertido en estándares. Este es el caso del descriptor MPEG-7 relacionado [27].

En los experimentos se ha utilizado la intensidad de la actividad de movimiento, definida como la desviación estándar de las magnitudes de los vectores de movimiento, normalizada por la resolución del cuadro y cuantificada por la resolución de la imagen. tal como se describe en [27].

Los resultados experimentales han sido llevados a cabo usando una secuencia con 4 tomas (variando la actividad de baja a media con algunos picos de alta actividad) formando un total de 1692 cuadros, a una resolución QQCIF, aunque se puede usar cualquier versión ya que en la arquitectura t+2D diferentes niveles espaciales tienen el mismo conjunto de vectores de movimiento para una resolución temporal concreta disponible en el GoP.

VERSIÓN / *THRESHOLD*	**Nº DE CUADROS P/S**	**T1**	**T2**	**T3**	**T4**
QQCIF-25	12.5	6.2363	17.1225	31.0505	51.2075
QQCIF-12.5	6.25	12.4726	34.2450	62.1010	102.4149
MPEG	22.91	3.4481	9.4673	17.1682	28.3133

Tabla 4-4 Umbrales (T1 a T4) para las diferentes versiones de la secuencia tenis calculados a partir del descriptor de actividad de movimiento de MPEG-7.

Para evaluar la eficiencia del algoritmo, los resultados se han realizado de manera subjetiva usando los umbrales que figuran en la Tabla 4-4. Las Figura 4-2 y Figura 4-3 muestran los resultados para MPEG y aceSVC respectivamente, donde act(k) indica la media de actividad para la toma k e indica el rango de actividad (1 indica muy poca actividad -intensidad de movimiento inferior a T1- mientras que 5 indica la actividad la actividad más alta -intensidad de movimiento superior a T4-).

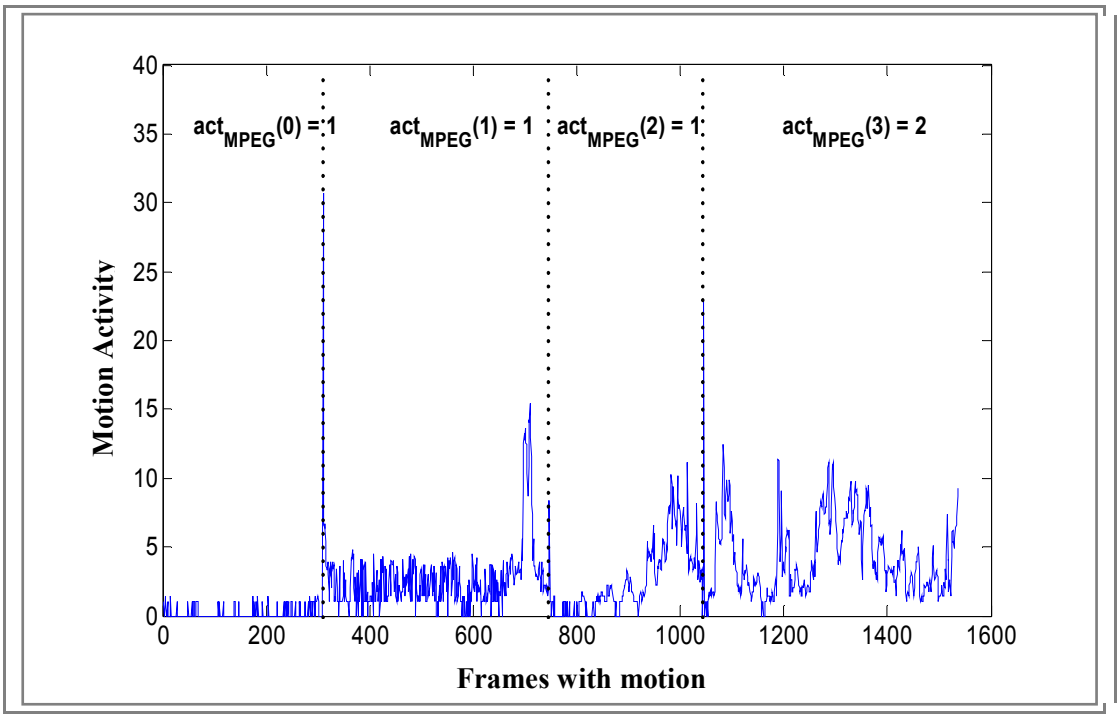

Figura 4-2 Rango y actividad de movimiento para el codec MPEG

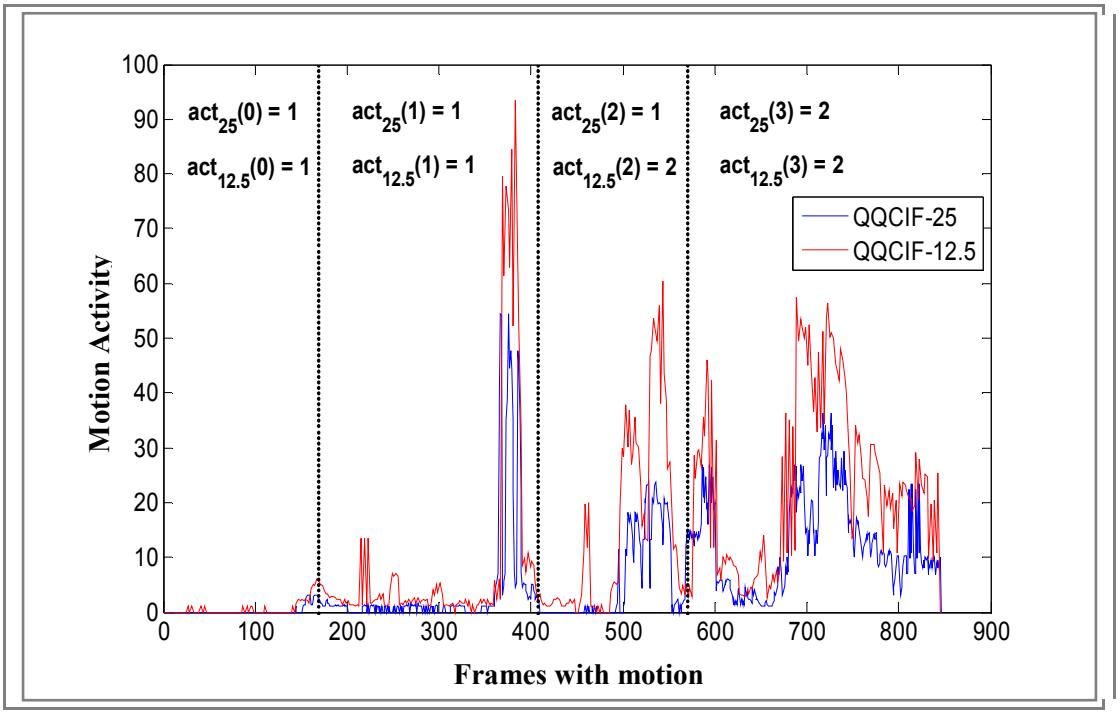

Figura 4-3 Rango y actividad de movimiento para la secuencia tenis usando diferentes resoluciones temporales del codec aceSVC

Para el caso de MPEG, el número de cuadros P por segundo, es decir cada 25 cuadros, que hay en un GoP, es superior al caso de aceSVC como consecuencia de que la composición del GoP de esta secuencia es IPPPPPPPPPP frente a la información que se puede extraer de una cierta capa temporal IPIPIPIPIPIP en el caso de aceSVC. Es por ello que como se observa en la Figura 4-2 y Figura 4-3, se presente más información de movimiento en MPEG, 1551 cuadros P, que en aceSVC, 846 cuadros P para QQCIF-25 y 423 para QQCIF-12.5 (en este caso los valores se han colocado bajo el mismo eje que el anterior).

La Figura 4-3 revela la actividad de movimiento obtenida usando dos capas temporales (25 y 12.5). Cabe destacar que un vector de movimiento de cada capa temporal describe un desplazamiento de los bloques sobre un intervalo temporal diferente. Ambas curvas muestran un similar comportamiento y también similar al obtenido con los vectores de MPEG (ver Figura 4-2). Considerando el valor medio de actividad de cada toma y cuantificado por los umbrales de la Tabla 4-4, los resultados son bastante similares.

4.3 Movimiento de Cámara

Para la estimación de movimiento de cámara, se ha empleado el algoritmo descrito en la sección 2.3.3.3, usando como vectores de movimiento los vectores de la subbanda LLL en lugar de los vectores de MPEG. Los resultados experimentales se han llevado a cabo usando la misma secuencia que la usada en la sección 4.2.

La Figura 4-4 (a) y Figura 4-5 (a) muestran el *ground-truth* obtenido de un proceso manual. Siendo el eje de abscisas el eje temporal, los valores representados son una indicación de si cada tipo de movimiento de cámara está presente o no. Los valores positivos o negativos están destinados a clasificar las dos direcciones diferentes que pueden tomar un patrón particular, siendo éstas derecha e izquierda, arriba y abajo, o *in* y *out* para los casos de *pan*, *tilt* y *zoom* respectivamente, mientras que los valores nulos representan ausencia de movimiento.

Asimismo, la Figura 4-4 (b) muestra los resultados obtenidos del algoritmo de movimiento de cámara integrado en el RC-CAT para la secuencia de MPEG. Como puede percibirse, después de compararlos con las curvas obtenidas con el *ground-truth,* los resultados son bastante similares.

En algunos casos, incluso para los evaluadores es realmente difícil decidir en qué cuadro empiezan los movimientos (siendo el peor caso cuando varios patrones participan de manera simultánea).

Además, y teniendo en cuenta la aplicación final de esta señal, que es la adaptación de contenidos, se puede argumentar razonablemente que buscar la ubicación exacta donde un movimiento de cámara comienza por medio de la selección del cuadro o del segmento donde comienza teniendo en cuenta la satisfacción máxima de los usuarios no tiene demasiado sentido. Una explicación plausible para ello es que incluso este usuario muchas veces no es capaz de establecer ese cuadro, además, el contenido de ese cuadro y sus cuadro vecinos serán prácticamente similares.

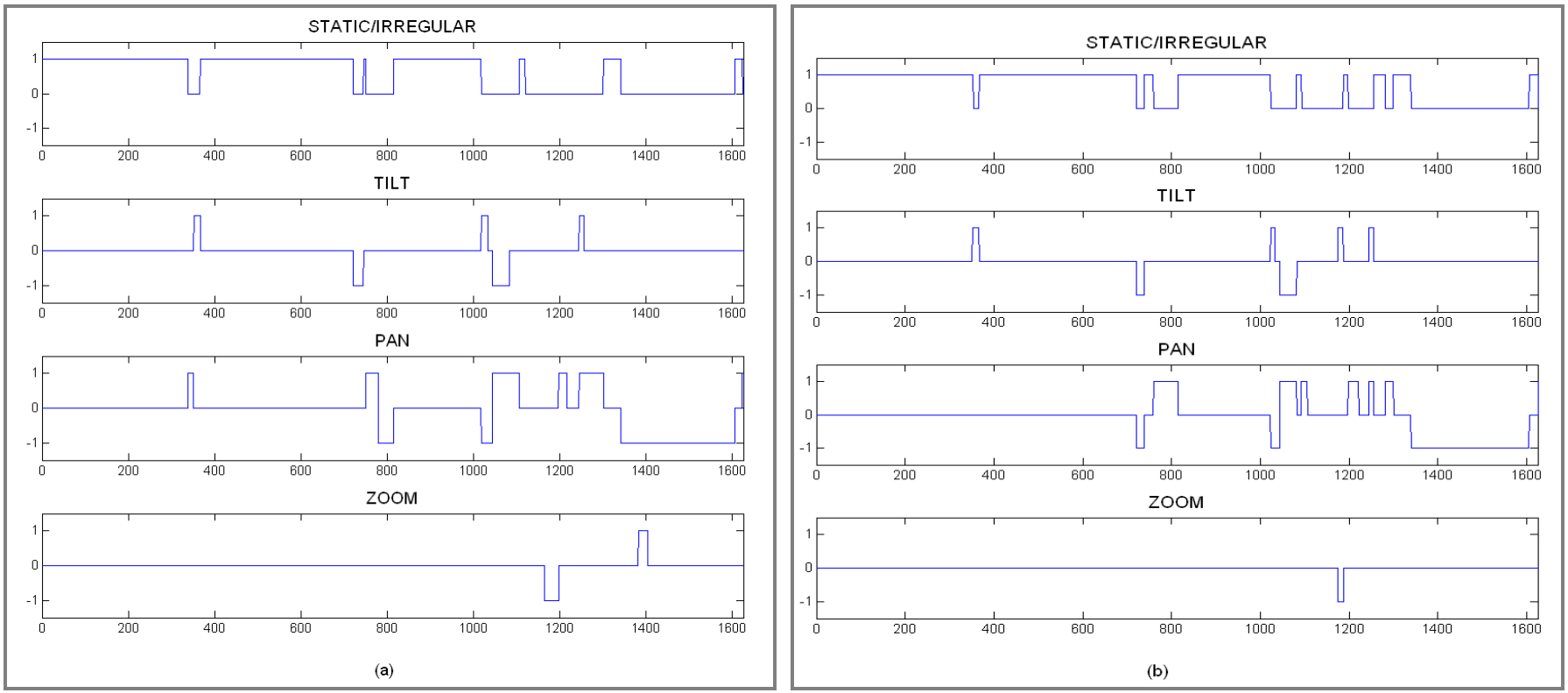

Figura 4-4 (a) Patrones de movimiento de cámara extraídos manualmente (*ground-truth*) para la secuencia tenis1.mpg. (b) Resultados de los patrones de movimiento obtenidos del algoritmo de movimiento de cámara integrado en el RC-CAT.

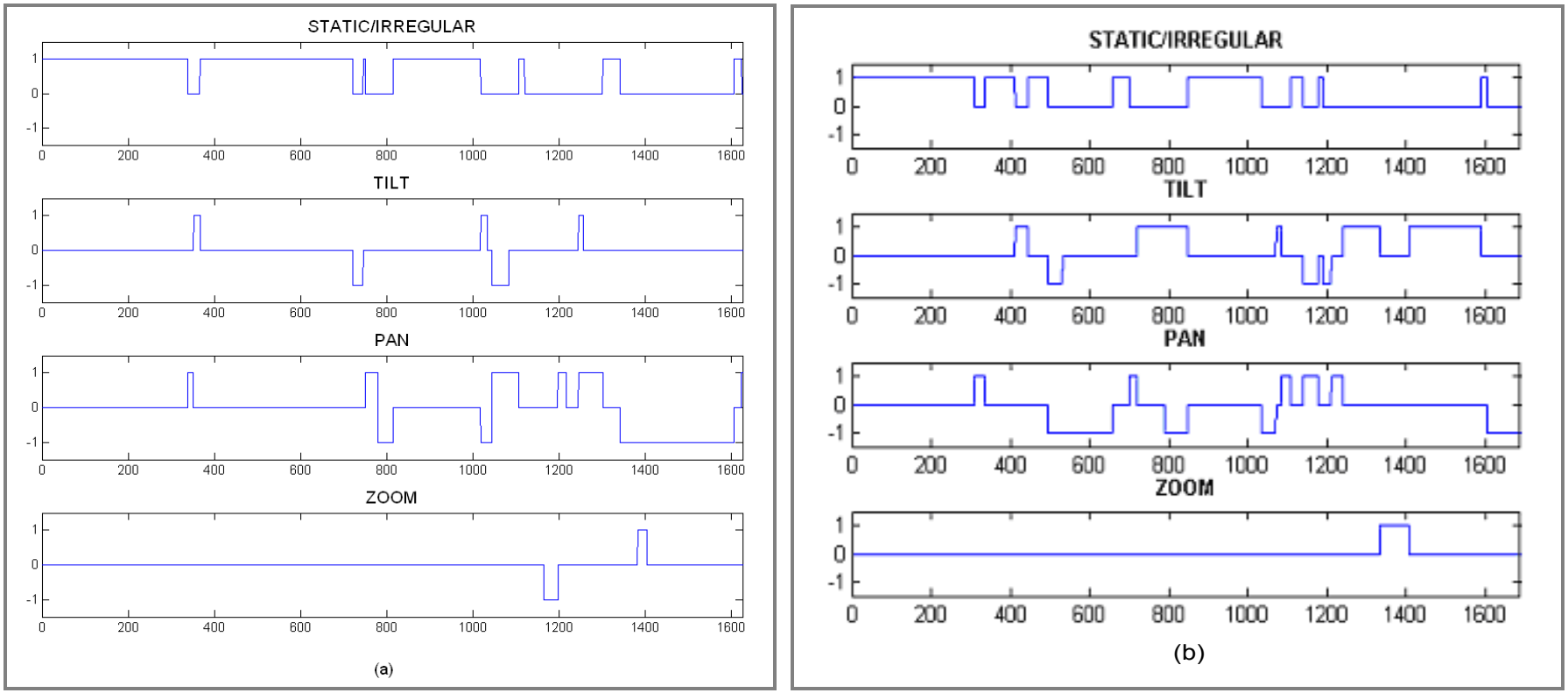

Figura 4-5 (a) Patrones de movimiento de cámara extraídos manualmente (*ground-truth*) para la secuencia tenis1.mpg. (b) Resultado de los patrones obtenidos en la secuencia de test usando el algoritmo de movimiento de cámara para el caso de aceSVC

Comparando los datos obtenidos de la Figura 4-5 (b) con el *ground truth* (a), el algoritmo presenta un peor comportamiento sobre aceSVC que sobre MPEG. Dicha degradación del comportamiento puede ser debida al hecho de que el algoritmo original fue diseñado y ajustado para el caso de MPEG-1/2, donde el tamaño de los macrobloques es constante. En este punto conviene recalcar que, a diferencia de otros eventos como los cambios de toma, la precisión a nivel de cuadro en el caso de movimiento de cámara no es tan importante.

A su vez, para aceSVC el tamaño de los bloques de compensación de movimiento es variable, pero éstos son replicados para simular un tamaño de bloque fijo (internamente por el API) como muestra la Figura 3-8. Esto puede conducir a vectores de movimiento incorrectos en bloques replicados de amplios bloques de compensación de movimiento, lo que puede provocar una peor estimación de los parámetros del modelo. Adicionalmente, se podría obtener una mejor estimación si se pudiera utilizar la secuencia con predicción unidireccional en lugar de bidireccional, pero debido a la limitaciones existentes, esta opción no se ha estudiado.

4.4 Adaptación espacial

La única adaptación espacial incluida en el RC-CAT consiste en una simplificación del fondo o *background*, por medio de una sustitución de éste por un color uniforme o de un filtrado paso bajo para emborronarlo, reduciendo en ambos casos la cantidad de información a codificar.

La característica espacial utilizada en los experimentos es la segmentación gruesa de la máscara extraída en el módulo de análisis, donde se espera que los objetos semánticamente significativos en el cuadro, en este caso asimilados a los objetos que presentan movimiento, estén presentes en el primer plano de la secuencia, mientras que los objetos del fondo se puedan separar del primer plano. Así pues, se conserva los objetos del primer plano degradando los objetos del fondo.

Para esta prueba se ha empleado una secuencia tenis de 550 cuadros con resolución 4CIF-25 y se ha codificado con cuatro resoluciones espaciales y temporales. Para simplificar, y reducir el número de bits necesarios para codificar el cuadro, se ha empleado el algoritmo descrito en la sección 2.3.3.3, donde las máscaras son extraídas de la subbanda LLL en distintas capas temporales, y son aplicadas a la resolución original del vídeo.

En este punto, conviene comentar que el algoritmo empleado para la obtención de las máscaras no está terminado, es decir, es un algoritmo de prueba que todavía se encuentra en fase de desarrollo y optimización; por ello, como se muestra a continuación, los resultados no son del todo los esperados.

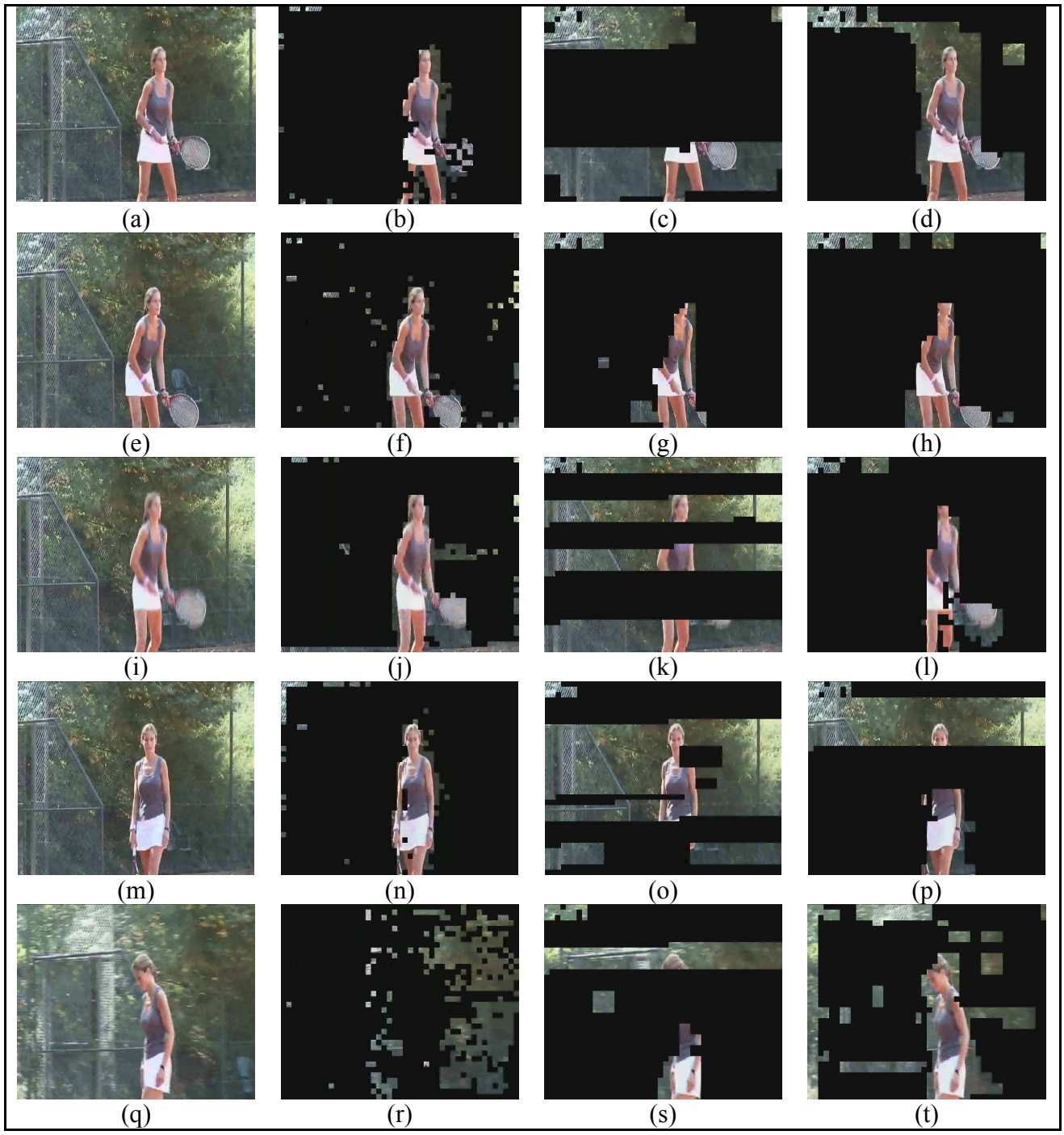

Figura 4-6 Cuadros originales de la secuencia tenis -(a), (e), (i), (m), (q)- y los resultantes de aplicar el algoritmo de segmentación de movimiento. Los casos (b), (f), (j), (n) y (r) son el resultado obtenido sobre MPEG, mientras que (c), (g), (k), (o) y (s) están referidos a las máscaras obtenidos para aceSVC al mismo nivel temporal que la secuencia temporal (25 cuadros por segundo), y (d), (h), (l), (p) y (t) están referidos a un nivel temporal inferior (12.5 cuadros por segundo).

Como puede contemplarse en la Figura 4-6, a veces la estimación de movimiento de la cámara no es tan precisa como se desearía en la segmentación. Esto es debido principalmente al hecho de que, a excepción de los dispositivos controlados mecánicamente, es muy difícil lograr un patrón de movimiento completamente uniforme, como se asume en el algoritmo, por lo que el primer plano de las máscaras obtenidas pueden presentar bastante error (casos (n), (k), (o), (p), (r), (s),…).

A diferencia con MPEG y debido a que el algoritmo realiza *tracking* del movimiento, los resultados sobre aceSVC son menos precisos. Estas diferencias son producto de que en MPEG el GOP de 12 cuadros sigue el siguiente patrón IBBPBBPBBPBB, por lo que al haber sólo un cuadro I a lo largo del GOP es posible hacer *tracking* del movimiento.

Por el contrario, el algoritmo no es muy efectivo para aceSVC, ya que por ejemplo un GOP de 16 cuadros de una secuencia a 25 cuadros por segundo se establece según el patrón IPIPIPIPIPIPIPIP, es decir, únicamente los cuadros impares presentan información de movimiento, perdiéndose por tanto la información del movimiento anterior e imposibilitando hacer *tracking*.

Finalmente, cabe destacar que como se demuestra en la sección 4.2, los vectores de movimiento son de mayor amplitud en la versión a 12.5 que a 25 cuadros por segundo. Esto es debido a que el salto del movimiento es de dos cuadros en lugar de uno y por ello se presentan mejores resultados a 12.5 cuadros por segundo que a 25.

Por último, recalcar que los resultados obtenidos de la segmentación de objetos no son muy concluyentes, ni para MPEG (aunque se obtengan unos resultados razonables) ni para aceSVC, ya que el algoritmo de segmentación empleado es un mero prototipo y no está terminado. Sin embargo, para ser una primera versión los resultados pueden ser considerados aceptables.

4.5 Coste computacional

Tal como se ha comentado en la sección 4.1, la razón más importante en el análisis para la utilización de la capa espacial más baja en un vídeo aceSVC es la ganancia en eficiencia, en términos de reducción de coste computacional.

El tiempo total empleado en la decodificación del vídeo viene definido por la siguiente fórmula:

$$T_{total} = t_{lectura\ del\ bitstream} + t_{decodificación\ de\ EZBC} + t_{decodifiacion\ de\ los\ vectores\ de\ movimiento}$$

Para poder extraer la información espacial de una subbanda determinada, únicamente se necesita realizar la decodificación EZBC. Adicionalmente, para obtener los vectores de movimiento es necesario decodificarlos.

La Figura 14 muestra los tiempos medios de decodificación de la secuencia de test utilizada en la sección 4.1 a diferentes resoluciones, incluida la resolución 4CIF. Como puede observarse, la primera gran contribución del tiempo de decodificación lo representa la decodificación de EZBC. El eje vertical de la izquierda está referido al tiempo empleado para la lectura del *bitstream* y la decodificación de los vectores de movimiento, mientras que el eje de la derecha se refiere al tiempo de decodificación de la subbanda (EZBC).

Como era de esperar, a bajas resoluciones el tiempo de decodificación se reduce como consecuencia de la menor cantidad de datos (píxeles) a procesar. Aunque se ha medido el coste computacional a distintas calidades, no se han mostrado en la gráfica porque los resultados obtenidos eran muy similares.

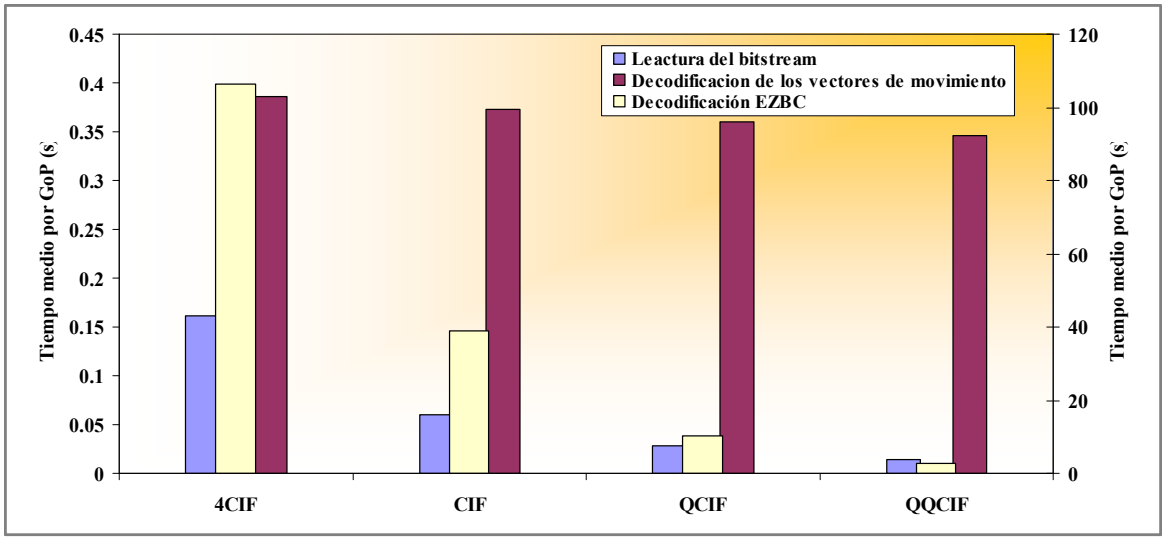

Figura 4-7 Tiempo medio de decodificación de un GoP a diferentes escalas espaciales: el eje izquierdo indica el tiempo medio por GoP empleado en la lectura del *bitstream* y la decodificación de los vectores de movimiento, y el derecho indica el tiempo para la decodificación EZBC.

En conclusión, hay que tener un balance entre precisión de coeficientes y eficiencia (tiempo de decodificación). Como es evidente cuanto menor sea la resolución empleada, menor será la precisión obtenida de los coeficientes YUV, pero menor será a su vez el tiempo de decodificación ya que la cantidad de datos es menor y por tanto el proceso de adaptación será más rápido.

Para que el lector pueda hacerse una idea sobre el coste computacional que lleva adaptar una secuencia entera en el RC-CAT, se expone en la Tabla 4-5 el tiempo total del análisis y generación del vídeo adaptado con adaptación espacial (*Uniform-filling*), temporal (*Video slideshow*), cambio de resolución y de calidad, empleando en ambos codecs la misma secuencia de la sección 4.1 pero únicamente con 540 cuadros.

	4CIF	CIF	QCIF	QQCIF	MPEG
Tiempo de adaptación (s)	2313.36	835.92	375.84	226.80	54.15
Velocidad de adaptación (cuadros por seg.)	0.23	0.65	1.44	2.38	9.97

Tabla 4-5 Tiempo total de adaptación para MPEG y diferentes versiones de aceSVC de un segmento de 540 cuadros de la secuencia news.

A partir de los datos obtenidos, se deduce que la velocidad de adaptación conseguida en MPEG es 42 veces superior a la conseguida con aceSVC a la resolución 4CIF. Aunque no se consiga la velocidad en tiempo real (25 cuadros por segundo) en ninguno de los casos, no quiere decir que la herramienta RC-CAT no opere en tiempo real, ya que aunque algunas operaciones de análisis no funcionen en la actualidad a tiempo real en un PC estándar, el enfoque online asegura que para sistemas potentes, al menos en el caso de MPEG, se cumple el objetivo.

5 Conclusiones y trabajo futuro

5.1 Conclusiones

El RC-CAT proporciona una adaptación de contenido eficiente y *online* realizada en el momento de la adaptación de la secuencia. En él se incluyen herramientas que guiadas por técnicas que operan en el dominio comprimido, consiguen eludir la decodificación entera de la secuencia de vídeo consiguiendo, de este modo, procesar el flujo de una manera más eficiente.

Esta herramienta permite realizar una adaptación temporal de la secuencia de entrada. Para realizar dicha adaptación, el RC-CAT se basa en los resultados de los algoritmos de detección de cambios de toma, la actividad de movimiento y el movimiento de cámara integrados en la misma. Estos resultados guiarán el proceso de selección de los cuadros clave significativos de la secuencia.

A partir de los cuadros clave, se ofrecen tres posibilidades de adaptación temporal de la secuencia, conocidas como *Video slideshow, Video skim* y *Image story-board.* La diferencia entre un tipo y otro radica principalmente en la replicación o no de los cuadros clave obtenidos y en el formato de salida del la secuencia adaptada.

En la adaptación *Video slideshow*, la secuencia adaptada consiste en un vídeo MPEG-1/2 en el cual los cuadros clave son replicados para conservar la longitud de la secuencia original. En el *Video skim* únicamente los cuadros claves son almacenados conformando un vídeo MPEG-1/2. Finalmente, la adaptación *Image story-board* generará una colección de imágenes en formato JPEG de los cuadros claves de la secuencia.

La otra gran funcionalidad de esta herramienta consiste en la adaptación espacial de la secuencia. Este tipo de adaptación que se basa en el resultado del algoritmo de segmentación gruesa, algoritmo que permite obtener una detección gruesa de los objetos en movimiento, ofrece dos tipos de adaptaciones para resaltar los objetos de primer plano: *Uniform-filling* y *Low-pass filtering.* En la adaptación *Uniform-filling*, el *background* es sustituido por un color uniforme, mientras que en *Low-pass filtering* el *background* es filtrado por un filtro paso bajo.

En el presente proyecto se han analizado y estudiado las herramientas para realizar una adaptación temporal y / o espacial haciendo especial hincapié en el uso de secuencias representativas que componen *MPEG-7 Content Set* [24][25][26].

Conviene destacar que todas las limitaciones que se han encontrado a lo largo del desarrollo de este proyecto, en concreto las limitaciones de la librería que se esperaba que permitiera obtener información simultánea de las distintas capas espaciales, temporales y calidades en las que se había codificado la secuencia, han condicionado la consecución de los objetivos planteados en este proyecto y se han estado barajando distintas alternativas hasta que se optó por la planteada en la sección 3.4.

La solución adoptada consiste en emplear tres versiones adaptadas de la secuencia de entrada para obtener una adaptación completa de la secuencia de entrada. En base a ello, las versiones empleadas han sido 4CIF-25 para adaptar la secuencia, QQCIF-25 para obtener los cambios de toma de la secuencia y QQCIF-12.5 para obtener una mejor predicción de los vectores de movimiento.

Esta solución obliga a restringir las características que se puede extraer, lo que afecta a la calidad de los algoritmos, además de exigir operar *offline* lo que imposibilita que el RC-CAT mantenga la funcionalidad de operar *online*.

En la prueba realizada sobre el detector de cambios de toma, se observa el efecto de la reducción de la resolución y de la calidad en cuanto a eficiencia y resultados. Aunque el algoritmo se comporta de forma más eficaz en el codec MPEG-1/2, a raíz de los experimentos se demuestra que el esquema de detección de tomas puede aplicarse al codificador escalable obteniendo resultados aceptables aunque con menor precisión que en MPEG.

A su vez, del dominio escalable se deduce que emplear una baja resolución espacial y / o versiones de baja calidad reduce notablemente el coste computacional, como era de esperar; la resolución espacial afecta ligeramente a la detección de los cambios de toma y el nivel de calidad degrada los resultados notablemente. Por lo tanto, la conclusión del trabajo es que el detector de tomas puede ser empleado con éxito en las versiones de baja resolución de secuencias de vídeo con el fin de aumentar la eficiencia, pero no sobre las versiones de baja calidad.

En el caso de la actividad de movimiento, los resultados muestran un comportamiento similar en ambos dominios, lo cual valida el algoritmo empleado para calcular de manera eficiente la actividad de movimiento a partir del dominio aceSVC.

Sin embargo, como la media de actividad en aceSVC es menor que en MPEG cuando hay poco movimiento y es muy superior cuando hay mucho movimiento los resultados del movimiento de cámara no son tan certeros, conduciendo a movimientos de cámara erróneos. No obstante, ello no significa que el algoritmo no sea válido, ya que teniendo en cuenta la aplicación final de esta señal, el hecho de que se produzca un cambio en el movimiento de la cámara es una señal de que el algoritmo funciona, aunque esté optimizado para el dominio MPEG.

En cuanto a la adaptación espacial, al estar diseñados los algoritmos para operar directamente en el dominio comprimido MPEG-1/2, los resultados que pueden conseguirse utilizando el dominio escalable pueden ser mucho menos precisos, sobre todo cuando se trata con características espaciales.

De esta manera, los resultados obtenidos dependen de los vectores rechazados en el algoritmo de movimiento de cámara, es decir, aquellos que no siguen el movimiento estimado. Debido a que el algoritmo de segmentación hace *traking* del movimiento y a que sólo los cuadros impares del GOP presentan información de movimiento, se hace imposible realizar un seguimiento del movimiento, lo que conduce a que los resultados sean menos eficaces que en el domino MPEG.

No obstante, las máscaras resultantes de este algoritmo presentan bastante ruido cuando el movimiento de cámara es complejo, debido a la dependencia en la precisión de la estimación del movimiento de cámara. Aun así, al tratarse de una versión inicial del algoritmo, los resultados de la segmentación de objetos para ambos codecs no son concluyentes aunque son un buen comienzo para próximas mejoras del mismo.

Finalmente, se concluye que los experimentos han demostrado la viabilidad de la aplicación de las mismas técnicas en ambos codecs. No obstante, debido en primer lugar a las limitaciones de la primera versión de la librería y a que originalmente el RC-CAT y los algoritmos de análisis se habían diseñado para aplicaciones sobre el dominio MPEG-1/2, los resultados obtenidos sobre este codec son mejores que para el caso a estudio en este proyecto, el codec escalable aceSVC.

También cabe destacar que el objetivo global de este proyecto consistía en poder realizar una adaptación *online* del codec aceSVC, pero que por motivos ajenos a la implementación y debido entre otras cosas a la lentitud del decodificador aceSVC no se ha podido conseguir dicho objetivo.

5.2 Trabajo futuro

Teniendo en cuenta todas las limitaciones que se ha presentado en el desarrollo del proyecto, el principal trabajo futuro que se plantea, sería conseguir establecer contacto con la QMUL para que desarrollaran una nueva versión de la librería que no tuviera una funcionalidad tan limitada como la anterior.

Esta nueva versión tendría que solucionar las limitaciones planteadas en la sección 3.3.1 para permitir extraer información simultánea de las distintas capas QTS (calidad, temporal y espacial), además de ser más eficientes en cuanto al tiempo del decodificador en la obtención de las características básicas para poder conseguir el objetivo global del proyecto: tener una adaptación a tiempo real.

A día de hoy, la QMUL ha dejado de trabajar en el tema del decodificador escalable por lo que ha sido imposible obtener una mejora de la versión actual para realizar los experimentos.

En caso de no obtener una librería mejorada, se podría contactar con la QMUL para que proporcionaran el código de la misma y así poder adaptarlo en función de nuestras necesidades y requerimientos.

A la espera de una mejora de la librería de aceSVC que permitiera entre otras cosas obtener información simultánea de distintas capas, y por tanto obtener un GOP más parecido al de MPEG donde sólo el primer cuadro no presenta información de movimiento, también se propone en primer lugar la optimización de los algoritmos de análisis para el codificador escalable aceSVC con el fin de que se obtengan unos resultados más parecidos a los obtenidos con MPEG-1/2 y se mejore la eficiencia de los mismos para este tipo de codec. Aun así con esta propuesta y con la librería anterior no sería posible conseguir la adaptación a tiempo real, pero si se obtendrían mejores resultados.

Referencias

[1] J-R. Ohm, "Advances in Scalable Video Coding", Proceedings of the IEEE, 93 (1), January 2005.

[2] F. Pereira, P. Van Beek, A.C. Kot, J. Ostermann, "Special issue on analysis and understanding for video adaptation", Circuits and Systems for Video Technology, IEEE Transactions on, October 2005.

[3] N. Sprljan, M. Mrak, G.C.K. Abhayaratne, E. Izquierdo, "A Scalable Coding Framework For Efficient Video Adaptation," Proc. Int. Work. on Image Analysis for Multimedia Interactive Services (WIAMIS), April 2005.

[4] S. Jeannin and A. Divakaran, "MPEG-7 visual motion descriptors", Circuits and Systems for Video Technology, IEEE Transactions on, June 2001.

[5] Habibollah Danyali, "Highly Scalable Wavelet Image and Video Coding for Transmission Over Heterogeneous Networks", 2004

[6] Entregable D 3.1 "Scalable content and metadata encoding techniques to support scalability in fixed and wireless communications systems: Evaluation of the state-of-the-art and Initial developments" del Proyecto AceMedia • IST-IP-001765: Integrating Knowledge, Semantics and Content for User-centred Intelligent Media Services., Apr. 2005

[7] Entregable D 3.4 "State-of-the-art analysis and update for content engineering technologies to support adaptability" del Proyecto AceMedia • IST-IP-001765: Integrating Knowledge, Semantics and Content for User-centred Intelligent Media Services., 2005

[8] Entregable D 3.5 "Enhancements to scalable video coding framework" del Proyecto AceMedia • IST-IP-001765: Integrating Knowledge, Semantics and Content for User-centred Intelligent Media Services., Jan. 2006

[9] Entregable ID 3.7 "Release of the first version of aceMedia SVC" del Proyecto AceMedia • IST-IP-001765: Integrating Knowledge, Semantics and Content for User-centred Intelligent Media Services., 2006

[10] Entregable ID 3.22 "Release of the third version of aceMedia SVC with enhanced tools" del Proyecto AceMedia • IST-IP-001765: Integrating Knowledge, Semantics and Content for User-centred Intelligent Media Services., Nov. 2006

[11] Yonghui Wang, "Fully Scalable Video Coding Using Redundant-Wavelet Multihypothesis And Motion-Compensated Temporal Filtering", Dec 2003

[12] F. Verdicchio, Y. Andreopoulos, T. Clerckx, J. Barbarien, A. Munteanu, J. Cornelis, P. Schelkens "Scalable video coding base on motion-compensated temporal filtering: complexity and functionality analysis", 2004

[13] M. Van der Schaar and Iphilip A. Chow, "MULTIMEDIA OVER IP AND WIRELESS NETWORKS COMPRESSION, NETWORKING, AND SYSTEMS": 117-158, 2007

[14] L. Herranz, F. Tiburzi and J. Bescos "Extraction of Motion Activity from Scalable-Coded Video Sequences", Dec. 2006

[15] Entregable D 3.9 “Media adaptation into the Cross Media Engine (CME): results, progress and future plans” del Proyecto AceMedia • IST-IP-001765: Integrating Knowledge, Semantics and Content for User-centred Intelligent Media Services., Dec. 2006

[16] Entregable D 3.12 “Media adaptation into the Cross Media Engine (CME)” del Proyecto AceMedia • IST-IP-001765: Integrating Knowledge, Semantics and Content for User-centred Intelligent Media Services., Dec. 2007

[17] Entregable D 3.8 “Specification and approaches to cross-media adaptation and interoperability” del Proyecto AceMedia • IST-IP-001765: Integrating Knowledge, Semantics and Content for User-centred Intelligent Media Services., July 2006

[18] J. Bescós, “Real-time Shot Change Detection over On-line MPEG-2 Video”, *IEEE Tr. On Circuits and Systems for Video Technology*, 14 (4): 475-484, Apr. 2004

[19] R, Narasimha, A. Savakis, R. M. Rao, R. De Queiroz, “Key frame extraction using MPEG-7 motion descriptors”, Conference Record of the Thirty-Seventh Asilomar Conference on Signals, Systems and Computers, 2003. Volume 2, 9-12 Nov. 2003. 1575 – 1579. Vol2

[20] G. B. Rath and A. Makur, “Iterative least squares and compression based estimations for a four-parameter linear global motion model and global motion compensation”, Circuits and Systems for Video Technology, IEEE Transactions on, vol. 9, pp. 1075–1099, Oct. 1999

[21] V. Mezaris, I. Kompatsiaris, M.G Strintz, “Video object segmentation using Bayes-based temporal tracking and trajectory-based region merging”, Circuits and Systems for Video Technology, IEEE Transactions on, 14 (6), June 2004

[22] Entregable ID 3.10 “Real-time analysis and adaptation into the CME” del Proyecto AceMedia • IST-IP-001765: Integrating Knowledge, Semantics and Content for User-centred Intelligent Media Services., Nov. 2005

[23] K. A. Peker, A. Divakaran and H. Sun, “Constant pace skimming and temporal sub-sampling of video using motion activity,” Proc. IEEE Int’l Conf. on Image Processing, Thessaloniki, Greece Oct. 2001

[24] MPEG Requirements Group, Description of MPEG-7 Content Set, MPEG, Atlantic City, NJ, Oct. 1998.

[25] Licensing Agreement for MPEG-7 Content Set, Atlantic City, NJ, Oct. 1998.

[26] Guide to obtaining the MPEG-7 Content Set, MPEG, Rome, Italy, Dec. 1998.

[27] I Leszek Cieplinski, Whoi-Yul Kim, Jens-Rainer Ohm, Mark Pickering, Akio Yamada: MPEG-7 Part 3. International Organization For Standarization Organisation Internationale De Normalisation ISO/IEC JTC1/SC29/WG11 Coding of Moving Pictures and Associated Audio, 87-92 (July 2001)

[28] A. Isar, S. Moga and X. Lurton, “A Statistical Analysis of the 2D Discrete Wavelet Transform“, May 2005

[29] M. Mastriani, “Denoising and Compression in Wavelet Domain via Projection onto Approximation Coefficients”, Dec. 2008

[30] Ömer Nezih Gerek and Dogan Gökhan Ece, “A 2d representation for analysis and coding of power quality events”, Aug. 2003

Glosario

API	Application Programming Interface
aceSVC	aceMedia Scalable Video Codec
SVC	Scalable Video Codec
RC-CAT	Real-time Content-based Content Adaptation Tool
MPEG	Moving Pictures Experts Group
4CIF	resolución (en píxeles) de 704x576
CIF	resolución (en píxeles) de 352x288
QCIF	resolución (en píxeles) de 176x144
QQCIF	resolución (en píxeles) de 88x72
MCTF	Motion Compensated Temporal Filtering
HVSBM	Hierarchical Variable Size Block Matching
DWT	Discrete Wavelet Transform

Anexos

A Manual de instalación

Para la instalación del sistema completo, se necesitará el sistema operativo Linux instalado en el PC, con los siguientes paquetes instalados:

- cpp y cpp-4.1
- g++ y g++-4.1
- gcc, gcc-4.1, gcc-4.1-base y gcc-4.2-base
- gcj-4.2-base
- gij y gij-4.2
- javacc y java-common
- liba52-0.7.4
- libc6-dev
- libgcj8-1
- libmpeg2-4
- sablevm
- libsablevm1-dev
- xlib1g-dev

En la versión actual, los ejecutables **SVCenc**, **SVCdec** y **SVCextr** tienen que situarse en el mismo directorio que la librería **libmmvsvc**, ya que necesitan estar enlazados a ella. Para ello, hay que ejecutar el script **vincLib**, que se encuentra en la carpeta **software/Linux**.

.\vincLib

Vinculada la librería, la ejecución de la herramienta de análisis RC-CAT únicamente necesita del fichero ejecutable, **test_adaptation** y del fichero de configuración que contiene la información necesaria para adaptar la secuencia.

.\test_adaptation <fichero de configuración>

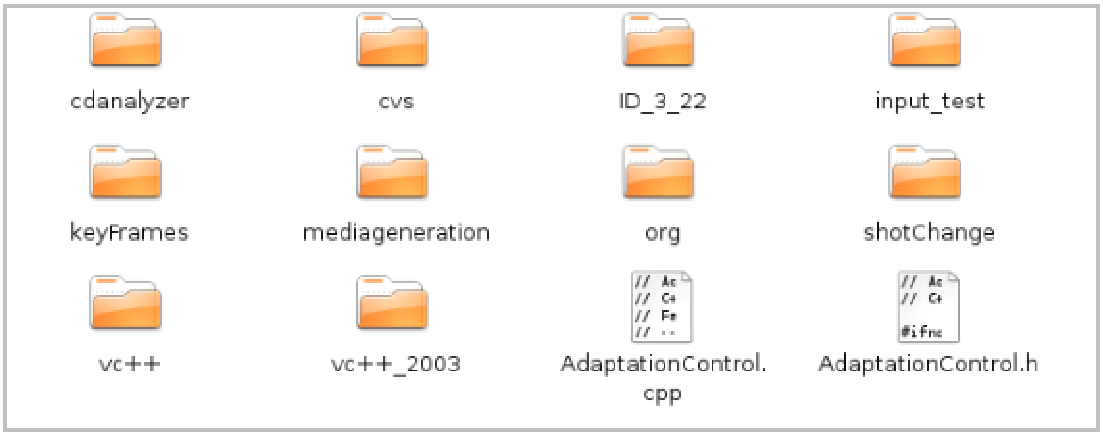

Figura A-1 Directorio principal de la herramienta RC-CAT

A modo de ejemplo, en la carpeta **rc-cat** se encuentra el fichero **RC-CAT.config**, ver Figura A-2, que contiene los parámetros de adaptación de la secuencia de entrada.

```
#
# RC-CAT CONFIGURATION FILE
#
================================================================================
================================================================================
#
# INPUT FILE
# ------------------------------------------------------------------------
VIDEO_ENTRADA =../videos/video.svc  # en formato svc o m2v
#
# OUTPUT FILE
# ------------------------------------------------------------------------
VIDEO_SALIDA =../output_test/rc-cat_test                    # ruta
donde se almacenará el fichero adaptado
#
# ADAPTATION PARAMETERS
# ------------------------------------------------------------------------
CHANNEL_BITRATE = 640000            # KB/s
OUT_WIDTH = 704                     # ancho del fichero de salida
OUT_HEIGHT = 576                    # alto del fichero de salida

TEST_T_ADAPTATION_TYPE = 2          # 0: AT_IMAGE_STORYBOARD, 1:
AT_VIDEO_SKIMMING, 2: AT_VIDEO_SLIDESHOW, 3: AT_VIDEO_NO_ADAPTATION

TEST_S_ADAPTATION_TYPE = 1           # 0: ADAPTATION_TYPE_NONE, 1:
ADAPTATION_TYPE_NEWCOLOR, 2: ADAPTATION_TYPE_FILTER
```

Figura A-2 Fichero de configuración para adaptar las secuencias

Los datos de entrada y la salida se especifican por medio de los parámetros "VIDEO_ENTRADA" y "VIDEO_SALIDA" respectivamente. Los parámetros de adaptación que se definen son parámetros de información de codificación (tamaño del fichero multimedia de salida y el *bitrate*) y parámetros de información del tipo de adaptación, temporal y espacial, a realizar.

La adaptación espacial se define por medio del parámetro "TEST_S_ADAPTATION_TYPE" que toma valor 0 cuando no se realiza ningún tipo de adaptación espacial, valor 1 cuando el *background* de cada imagen es sustituido por un color uniforme (color negro) y valor 2 cuando el *background* es filtrado para emborronarlo resaltando así los objetos del primer plano.

Los datos auxiliares, máscaras de objetos obtenidas a través de los algoritmos de análisis del RC-CAT, que necesita la aplicación para adaptar espacialmente la secuencia de entrada se localizarán en la carpeta "outputs" del directorio principal.

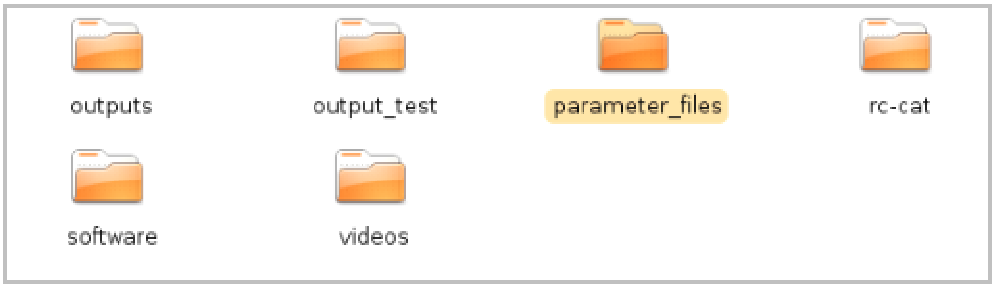

Figura A-3 Directorio raíz del CD

El contenido multimedia adaptado será un vídeo en formato MPEG-1/2 en caso de que el parámetro "TEST_T_ADAPTATION_TYPE" sea igual a 1 (vídeo *skimming,* definido como *Type II Videos* en la sección 2.3.4.1), 2 (vídeo *slideshow*, definido como *Type I Videos*) ó 3 (no se realiza adaptación temporal), o una colección de imágenes en formato *jp*g. Estos archivos se almacenarán en la ruta indica por el parámetro "VIDEO_SALIDA".

Para codificar un vídeo en formato aceSVC, éste ha de tener el formato YUV. En caso de no disponer del vídeo en este formato, se aconseja el empleo de **ffmpeg.exe** en el sistema operativo MS Windows, para convertir el vídeo al formato deseado.

Como ejemplo, para convertir una secuencia en formato m2v al formato yuv con resolución 4CIF (704x576) y un *framerate* de 25 cuadros por segundo, la secuencia a teclear en la línea de comandos es la siguiente:

Ruta del ffmpeg.exe –i <vídeo de entrada> –s <tamaño del vídeo a generar> -r <tasa de cuadros del vídeo a generar> <ruta y nombre del vídeo adaptado (con extensión yuv)>

(ffmpeg.exe –i "video.m2v" –s 704x576 -r 25 "video.yuv")

Una vez obtenido el vídeo descomprimido, la aplicación que permite codificar el vídeo es el **SVCEncoder**. Esta aplicación y el resto de las que se van a explicar a continuación se encuentran explicadas más en detalle en [10].

Como dato de entrada el codificador necesita del fichero de parámetros de codificación (este fichero ha de tener la extensión .par). Como ejemplo se muestra en el anexo B el fichero de parámetros empleados en los experimentos. La secuencia en la línea de comandos para codificar el vídeo es la siguiente:

.\SVCEncoder <path del fichero de parámetros de codificación>
(.\SVCEncoder ..\parameter_files\parameter_file.par)

Si lo que se desea es trabajar con una versión adaptada del *bitstream*, se invoca al extractor, **SVCextr**, seguido de los parámetros de adaptación para llevar a cabo la extracción deseada:

SVCextr <fichero comprimido de entrada> [-opciones] <fichero adaptado>

Opciones:

- -help: muestra la ayuda de la función
- -q[#]: extrae una capa de calidad predefinida
- -b[bitrate]: adapta la secuencia a un bitrate determinado. El bitrate se expresa en kbps o Mbps por medio de las letras "k" y "M" respectivamente. Por ejemplo, una adaptación a 128 kbps se expresa como –b128k
- -s[#]: número de capas espaciales a extraer. Por ejemplo si la resolución original es 4CIF y se la resolución objetivo es QCIF, la extracción se expresa como –s2. Hay que tener en cuenta que únicamente las capas especificadas en el parámetro *TargetDecPoints* del fichero de codificación pueden ser extraídas.
- -t[#]: número de capas temporales a extraer. Por ejemplo si el *frame rate* de la secuencia original es 25 fps y el la resolución objetivo es 12.5 fps, la extracción se expresa como –t1. Hay que tener en cuenta que únicamente las capas especificadas en el parámetro *TargetDecPoints* del fichero de codificación pueden ser extraídas.

Finalmente para reconstruir la secuencia comprimida, se utiliza el decodificador **(SVCdec)** invocado mediante la siguiente secuencia:

SVCdec <SecuenciaComprimida.svc> <SecuenciaRecontruida.yuv>

Todos estos programas se podrán encontrar en la carpeta **software** incluida en directorio principal del CD.

B Fichero de parámetros de codificación empleados en los experimentos

```
SourceVideoFile = .\..\..\mpeg_yuv\video.yuv
FrameWidth = 704
FrameHeight = 576
FrameRate = 25


#MotionParameters = DecompositionLabel: LambdaTemporal,
MacroblockSize, MotionEstimationLev, {MotionLevParams},
{MotionLevParams},...
MotionParameters = T00: 60, 128, 1, {2, 32, 4, 0}
MotionParameters = T01: 36
MotionParameters = T02: 40
MotionParameters = T03: 25
MotionParameters = T04: 75

Decomposition = L: T,LIFTING, 1x3, <T00>;
                  {
                     L: T,LIFTING, 1x3, <T01>;
                     {
                        L: T,LIFTING, 1x3, <T02>;
                        {
                              L: T,LIFTING, 1x3, <T03>;
                              {
                                    L: S,LIFTING,9x7,4;
                                    H: S,LIFTING,9x7,4;
                              }
                              H: S,LIFTING,9x7,4;
                        }
                        H: S,LIFTING,9x7,4;
                     }
                     H: S,LIFTING,9x7,4;
                  }

TargetDecPoints =  QQCIF, 6.25Hz, <42k, 50k, 58k, 66k, 74k>;
QQCIF, 12.5Hz, <58k, 66k, 74k, 82k, 90k>; QQCIF, 25Hz, <74k, 82k,
90k, 98k, 106k>; QCIF, 6.25Hz, <96k, 112k, 128k, 160k, 192k>;
QCIF, 12.5Hz, <128k, 160k, 192k, 224k, 256k>; QCIF, 25Hz, <192k,
224k, 256k, 288k, 320k>; CIF, 6.25Hz, <256k, 320k, 384k, 448k,
512k>; CIF, 12.5Hz, <384k, 448k, 512k, 640k, 768k>; CIF, 25Hz,
<512k, 640k, 768k, 896k, 1024k>; 4CIF, 6.25Hz, <768k, 896k,
1024k, 1280k, 1536k>;
 4CIF, 12.5Hz, <1024k, 1280k, 1536k, 1792k, 2048k>; 4CIF, 25Hz,
<1536k, 1792k, 2048k, 2560k, 3072k>

CompressedVideoFile = video.svc
```

PRESUPUESTO

1) Ejecución Material

- Compra de ordenador personal (Software incluido)..... ..... 2.000 €
- Alquiler de impresora láser durante 12 meses..... 100 €
- Material de oficina ..... 250 €
- Total de ejecución material ..... 2.350 €

2) Gastos generales

- 16 % sobre Ejecución Material ..... 376 €

3) Beneficio Industrial

- 6 % sobre Ejecución Material ..... 141 €

4) Honorarios Proyecto

- 1200 horas a 15 € / hora ..... 18000 €

5) Material fungible

- Gastos de impresión ..... 60 €
- Encuadernación ..... 200 €

6) Subtotal del presupuesto

- Subtotal Presupuesto ..... 20610 €

7) I.V.A. aplicable

- 16% Subtotal Presupuesto ..... 3297.6 €

8) Total presupuesto

- Total Presupuesto ..... 23907,6 €

Madrid, Marzo de 2009

El Ingeniero Jefe de Proyecto

Fdo.: Andrés Cortés Marlia
Ingeniero Superior de Telecomunicación

PLIEGO DE CONDICIONES

Este documento contiene las condiciones legales que guiarán la realización, en este proyecto, de un Scalable Video Analysis For Content Based Adaptation. En lo que sigue, se supondrá que el proyecto ha sido encargado por una empresa cliente a una empresa consultora con la finalidad de realizar dicho sistema. Dicha empresa ha debido desarrollar una línea de investigación con objeto de elaborar el proyecto. Esta línea de investigación, junto con el posterior desarrollo de los programas está amparada por las condiciones particulares del siguiente pliego.

Supuesto que la utilización industrial de los métodos recogidos en el presente proyecto ha sido decidida por parte de la empresa cliente o de otras, la obra a realizar se regulará por las siguientes:

Condiciones generales

1. La modalidad de contratación será el concurso. La adjudicación se hará, por tanto, a la proposición más favorable sin atender exclusivamente al valor económico, dependiendo de las mayores garantías ofrecidas. La empresa que somete el proyecto a concurso se reserva el derecho a declararlo desierto.

2. El montaje y mecanización completa de los equipos que intervengan será realizado totalmente por la empresa licitadora.

3. En la oferta, se hará constar el precio total por el que se compromete a realizar la obra y el tanto por ciento de baja que supone este precio en relación con un importe límite si este se hubiera fijado.

4. La obra se realizará bajo la dirección técnica de un Ingeniero Superior de Telecomunicación, auxiliado por el número de Ingenieros Técnicos y Programadores que se estime preciso para el desarrollo de la misma.

5. Aparte del Ingeniero Director, el contratista tendrá derecho a contratar al resto del personal, pudiendo ceder esta prerrogativa a favor del Ingeniero Director, quien no estará obligado a aceptarla.

6. El contratista tiene derecho a sacar copias a su costa de los planos, pliego de condiciones y presupuestos. El Ingeniero autor del proyecto autorizará con su firma las copias solicitadas por el contratista después de confrontarlas.

7. Se abonará al contratista la obra que realmente ejecute con sujeción al proyecto que sirvió de base para la contratación, a las modificaciones autorizadas por la superioridad o a las órdenes que con arreglo a sus facultades le hayan comunicado por escrito al Ingeniero Director de obras siempre que dicha obra se haya ajustado a los preceptos de los pliegos de condiciones, con arreglo a los cuales, se harán las modificaciones y la valoración de las diversas unidades sin que el importe total pueda exceder de los presupuestos aprobados. Por consiguiente, el número de unidades que se consignan en el proyecto o en el presupuesto, no podrá servirle de fundamento para entablar reclamaciones de ninguna clase, salvo en los casos de rescisión.

8. Tanto en las certificaciones de obras como en la liquidación final, se abonarán los trabajos realizados por el contratista a los precios de ejecución material que figuran en el presupuesto para cada unidad de la obra.

9. Si excepcionalmente se hubiera ejecutado algún trabajo que no se ajustase a las condiciones de la contrata pero que sin embargo es admisible a juicio del Ingeniero Director de obras, se dará conocimiento a la Dirección, proponiendo a la vez la rebaja de precios que el Ingeniero estime justa y si la Dirección resolviera aceptar la obra, quedará el contratista obligado a conformarse con la rebaja acordada.

10. Cuando se juzgue necesario emplear materiales o ejecutar obras que no figuren en el presupuesto de la contrata, se evaluará su importe a los precios asignados a otras obras o materiales análogos si los hubiere y cuando no, se discutirán entre el Ingeniero Director y el contratista, sometiéndolos a la aprobación de la Dirección. Los nuevos precios convenidos por uno u otro procedimiento, se sujetarán siempre al establecido en el punto anterior.

11. Cuando el contratista, con autorización del Ingeniero Director de obras, emplee materiales de calidad más elevada o de mayores dimensiones de lo estipulado en el proyecto, o sustituya una clase de fabricación por otra que tenga asignado mayor precio o ejecute con mayores dimensiones cualquier otra parte de las obras, o en general, introduzca en ellas cualquier modificación que sea beneficiosa a juicio del Ingeniero Director de obras, no tendrá derecho sin embargo, sino a lo que le correspondería si hubiera realizado la obra con estricta sujeción a lo proyectado y contratado.

12. Las cantidades calculadas para obras accesorias, aunque figuren por partida alzada en el presupuesto final (general), no serán abonadas sino a los precios de la contrata, según las condiciones de la misma y los proyectos particulares que para ellas se formen, o en su defecto, por lo que resulte de su medición final.

13. El contratista queda obligado a abonar al Ingeniero autor del proyecto y director de obras así como a los Ingenieros Técnicos, el importe de sus respectivos honorarios facultativos por formación del proyecto, dirección técnica y administración en su caso, con arreglo a las tarifas y honorarios vigentes.

14. Concluida la ejecución de la obra, será reconocida por el Ingeniero Director que a tal efecto designe la empresa.

15. La garantía definitiva será del 4% del presupuesto y la provisional del 2%.

16. La forma de pago será por certificaciones mensuales de la obra ejecutada, de acuerdo con los precios del presupuesto, deducida la baja si la hubiera.

17. La fecha de comienzo de las obras será a partir de los 15 días naturales del replanteo oficial de las mismas y la definitiva, al año de haber ejecutado la provisional, procediéndose si no existe reclamación alguna, a la reclamación de la fianza.

18. Si el contratista al efectuar el replanteo, observase algún error en el proyecto, deberá comunicarlo en el plazo de quince días al Ingeniero Director de obras, pues transcurrido ese plazo será responsable de la exactitud del proyecto.

19. El contratista está obligado a designar una persona responsable que se entenderá con el Ingeniero Director de obras, o con el delegado que éste designe, para todo relacionado con ella. Al ser el Ingeniero Director de obras el que interpreta el proyecto, el contratista deberá consultarle cualquier duda que surja en su realización.

20. Durante la realización de la obra, se girarán visitas de inspección por personal facultativo de la empresa cliente, para hacer las comprobaciones que se crean oportunas. Es obligación del contratista, la conservación de la obra ya ejecutada hasta la recepción de la misma, por lo que

el deterioro parcial o total de ella, aunque sea por agentes atmosféricos u otras causas, deberá ser reparado o reconstruido por su cuenta.

21. El contratista, deberá realizar la obra en el plazo mencionado a partir de la fecha del contrato, incurriendo en multa, por retraso de la ejecución siempre que éste no sea debido a causas de fuerza mayor. A la terminación de la obra, se hará una recepción provisional previo reconocimiento y examen por la dirección técnica, el depositario de efectos, el interventor y el jefe de servicio o un representante, estampando su conformidad el contratista.

22. Hecha la recepción provisional, se certificará al contratista el resto de la obra, reservándose la administración el importe de los gastos de conservación de la misma hasta su recepción definitiva y la fianza durante el tiempo señalado como plazo de garantía. La recepción definitiva se hará en las mismas condiciones que la provisional, extendiéndose el acta correspondiente. El Director Técnico propondrá a la Junta Económica la devolución de la fianza al contratista de acuerdo con las condiciones económicas legales establecidas.

23. Las tarifas para la determinación de honorarios, reguladas por orden de la Presidencia del Gobierno el 19 de Octubre de 1961, se aplicarán sobre el denominado en la actualidad "Presupuesto de Ejecución de Contrata" y anteriormente llamado "Presupuesto de Ejecución Material" que hoy designa otro concepto.

Condiciones particulares

La empresa consultora, que ha desarrollado el presente proyecto, lo entregará a la empresa cliente bajo las condiciones generales ya formuladas, debiendo añadirse las siguientes condiciones particulares:

1. La propiedad intelectual de los procesos descritos y analizados en el presente trabajo, pertenece por entero a la empresa consultora representada por el Ingeniero Director del Proyecto.

2. La empresa consultora se reserva el derecho a la utilización total o parcial de los resultados de la investigación realizada para desarrollar el siguiente proyecto, bien para su publicación o bien para su uso en trabajos o proyectos posteriores, para la misma empresa cliente o para otra.

3. Cualquier tipo de reproducción aparte de las reseñadas en las condiciones generales, bien sea para uso particular de la empresa cliente, o para cualquier otra aplicación, contará con autorización expresa y por escrito del Ingeniero Director del Proyecto, que actuará en representación de la empresa consultora.

4. En la autorización se ha de hacer constar la aplicación a que se destinan sus reproducciones así como su cantidad.

5. En todas las reproducciones se indicará su procedencia, explicitando el nombre del proyecto, nombre del Ingeniero Director y de la empresa consultora.

6. Si el proyecto pasa la etapa de desarrollo, cualquier modificación que se realice sobre él, deberá ser notificada al Ingeniero Director del Proyecto y a criterio de éste, la empresa consultora decidirá aceptar o no la modificación propuesta.

7. Si la modificación se acepta, la empresa consultora se hará responsable al mismo nivel que el proyecto inicial del que resulta el añadirla.

8. Si la modificación no es aceptada, por el contrario, la empresa consultora declinará toda responsabilidad que se derive de la aplicación o influencia de la misma.

9. Si la empresa cliente decide desarrollar industrialmente uno o varios productos en los que resulte parcial o totalmente aplicable el estudio de este proyecto, deberá comunicarlo a la empresa consultora.

10. La empresa consultora no se responsabiliza de los efectos laterales que se puedan producir en el momento en que se utilice la herramienta objeto del presente proyecto para la realización de otras aplicaciones.

11. La empresa consultora tendrá prioridad respecto a otras en la elaboración de los proyectos auxiliares que fuese necesario desarrollar para dicha aplicación industrial, siempre que no haga explícita renuncia a este hecho. En este caso, deberá autorizar expresamente los proyectos presentados por otros.

12. El Ingeniero Director del presente proyecto, será el responsable de la dirección de la aplicación industrial siempre que la empresa consultora lo estime oportuno. En caso contrario, la persona designada deberá contar con la autorización del mismo, quien delegará en él las responsabilidades que ostente.