UNIVERSIDAD DE LAS PALMAS
DE GRAN CANARIA

DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA Y AUTOMÁTICA

TESIS DOCTORAL

Aportaciones Algorítmicas y Arquitecturales para la Estimación de Movimiento en Sistemas de Codificación Híbrida de Vídeo Basados en los Estándares H.263 y H.264/AVC

Sebastián Miguel López Suárez

Las Palmas de Gran Canaria, 2006

A mis padres y a
mi hermana Elena.

"La inspiración existe,
pero tiene que encontrarte trabajando."

Pablo Ruiz Picasso, pintor y escultor español (1881-1973).

RESUMEN

En estas dos últimas décadas se ha producido un espectacular avance de los estándares de compresión de imagen y vídeo, guiado por el objetivo de conseguir unas tasas de compresión cada vez más ambiciosas. No obstante, esta evolución, ha traído consigo un considerable aumento en los requisitos de procesamiento exigidos a los sistemas de codificación y decodificación compatibles con estos estándares de compresión. Este hecho cobra especial relevancia en la estimación de movimiento, pues ésta representa, por su elevado coste computacional, así como por su impacto sobre los niveles de compresión alcanzados, la etapa crítica de un codificador de vídeo compatible con cualquier estándar de compresión híbrida de vídeo conocido.

En este contexto, numerosos investigadores han propuesto multitud de estrategias de búsqueda de vectores de movimiento con el objetivo de reducir el esfuerzo computacional inherente al proceso de estimación de movimiento. Estas estrategias se

basan, por lo general, en un conjunto de consideraciones iniciales acerca de las características espaciales y temporales de la secuencia de vídeo a comprimir. Asimismo, con el objetivo de maximizar sus prestaciones para determinadas aplicaciones, la mayor parte de las estrategias de estimación de movimiento están particularizadas para un determinado estándar de codificación de vídeo y, dentro de éste, para un rango concreto del porcentaje de compresión. Estas características determinan que, ante cambios en la naturaleza de la secuencia de vídeo a comprimir, en las necesidades del usuario, y/o en los requisitos del estándar, las prestaciones de compresión proporcionadas por las estrategias de estimación de movimiento propuestas hasta ahora no estén garantizadas.

En esta Tesis Doctoral se propone un nuevo algoritmo de estimación de movimiento, denominado VBS-ACBM (Variable Block Size – Adaptive Cost Block Matching), que realiza el cálculo de vectores de movimiento para cualquiera de los tamaños de bloques de píxeles y precisión de los vectores de movimiento definidos por los estándares H.263 y H.264/AVC, y por lo tanto, de manera implícita, para los definidos por cualquier estándar de compresión de vídeo. Este algoritmo garantiza, gracias al uso de las estrategias adaptativas desarrolladas en esta Tesis, unas prestaciones de compresión óptimas para todo tipo de secuencias de vídeo y requisitos de compresión, con un coste computacional reducido.

De igual manera, se presentan en esta Tesis un conjunto de soluciones arquitecturales para la implementación eficiente del algoritmo VBS-ACBM en codificadores de vídeo con restricciones de funcionamiento en tiempo real. En particular, se aportan dos nuevas arquitecturas para la estimación de movimiento con precisión entera y posterior refinamiento sub-píxel de vectores de movimiento. La introducción de un conjunto de novedosas estrategias arquitecturales planteadas en esta Tesis permite obtener, en ambas arquitecturas, mejoras significativas con respecto a trabajos publicados en la bibliografía reciente.

AGRADECIMIENTOS

Tengo la inmensa suerte de poder afirmar que la realización de esta Tesis Doctoral se ha convertido en una experiencia inolvidable. Desde estas líneas, me gustaría expresar mi más sincero agradecimiento a las personas que lo han hecho posible.

A mis directores de Tesis, José Fco. López y Roberto Sarmiento, por la confianza que depositaron en mí desde el primer momento, así como por su incondicional apoyo durante todos estos años. El hecho de que dos personas a las que admiro profundamente, tanto desde el punto de vista profesional como personal, y que representan para mí un ejemplo a seguir, se hayan convertido no sólo en mis directores de Tesis sino también en verdaderos amigos, me lleva a reafirmarme en mi idea de que he tenido muchísima suerte.

A mis compañeros Valentín de Armas, Gustavo Marrero y Félix Tobajas les tengo que dar las gracias por tantas cosas que, para no aburrir al lector, y abusando de la confianza que tengo con ellos, las resumiré en un injusto "gracias por todo". En cualquier caso, lo que no quiero pasar por alto es aquello por lo que, sin duda alguna, les estoy más agradecido: su impagable amistad con la que me honran cada día.

A Antonio Núñez, por abrirme de par en par las puertas del Instituto Universitario de Microelectrónica Aplicada, dentro del cual he podido realizar esta Tesis Doctoral.

A Rubén Arteaga, Roberto Esper-Chaín, Francisco González, Luis Hernández, Juan Antonio Montiel, Héctor Navarro, Raúl Regidor, Carlos Javier Sosa y Óscar Tubío les agradezco su absoluta disposición a la hora de ayudarme a saltar muchos de los obstáculos con los que me he encontrado durante estos años.

A Ernesto Perea y a Viviana D'Alto, por brindarme la extraordinaria oportunidad de realizar una estancia de investigación en los centros de Milán y Padova de la compañía STMicroelectronics, permitiéndome participar de pleno en el excitante proyecto Nomadik. Asimismo, quisiera expresar mi agradecimiento a la Fundación Universitaria de Las Palmas y a La Caja de Canarias, por colaborar en la financiación de esta estancia por medio del programa Innova.

Fuera del ámbito meramente académico e investigador, me gustaría darle las gracias a mi otro grupo de amigos, por lo mucho que supone para mí que sean precisamente eso, mis amigos.

Gracias a ti, Vanessa, por tu aliento constante para que sacara adelante la Tesis, soportándome y animándome con infinita paciencia y generosidad en los momentos menos buenos, en los que has sido la única persona capaz de arrancarme una sonrisa. Para mi fortuna, esto no esto todo por lo que tengo que darte las gracias, al contrario, me queda lo más importante. Sin duda alguna, lo que más te agradezco son los inolvidables momentos que me has regalado a tu lado. Desde este "pizquito" de texto, de todo corazón, infinitas gracias.

Por último, gracias a mis padres, Sebastián y Micaela, porque ellos, sencillamente, me han dado todo a cambio de nada. Sólo espero que todos estos años de trabajo y silenciosos esfuerzos por su parte, dedicados en exclusividad a dar una formación y unos valores a sus dos hijos, hayan servido para que hoy se sientan tan orgullosos de mí, como yo lo estoy de tenerlos como padres. A ellos les debo todo lo que soy.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ACRÓNIMOS

4CIF:	*4 Common Intermediate Format.*
ACBM:	*Adaptive Cost Block Matching.*
ACBM-HW:	*Adaptive Cost Block Matching – HardWare.*
AMPD:	*Advanced Mode Pre-Decision.*
AVC:	*Advanced Video Coding.*
CD-ROM:	*Compact Disc – Read Only Memory.*
CGRA:	*Coarse-Grain Reconfigurable Architectures.*
CIF:	*Common Intermediate Format.*
CMOS:	*Complementary Metal Oxide Semiconductor.*
DCT:	*Discrete Cosine Transform.*
ENDIVIA:	*ENtorno de Diseño basado en IPs con soporte para Verificación y depuración, Integración con redes en chip y Aplicaciones multimedia.*
ETSII:	*Escuela Técnica Superior de Ingenieros Industriales.*
FIR:	*Finite Impulse Response.*
FPGA:	*Field Programmable Gate Array.*
FSBM:	*Full Search Block Matching.*

HDTV:	*High Definition TeleVision.*
HPR:	*Half Pixel Refinement.*
IMEC:	*Interuniversities MicroElectronics Center.*
IP:	*Intellectual Property.*
ISO:	*International Organization for Standardization.*
ITU:	*International Telecommunication Union.*
JPEG:	*Joint Photographic Experts Group.*
JVT:	*Joint Video Team.*
LUT:	*Look-Up Table.*
MPEG:	*Moving Picture Experts Group.*
NoC:	*Network on Chip.*
PBM:	*Predictive Block Matching.*
PBM-HW:	*Predictive Block Matching – HardWare.*
PE:	*Processing Element.*
PSNR:	*Peak Signal to Noise Ratio.*
QCIF:	*Quarter Common Intermediate Format.*
QPR:	*Quarter Pixel Refinement.*
RTL:	*Register Transfer Level.*
SAD:	*Sum of Absolute Differences.*
SOB:	*Start Of Block.*
SVC:	*Scalable Video Coding.*
VBS-ACBM:	*Variable Block Size – Adaptive Cost Block Matching.*
VBSME:	*Variable Block Size Motion Estimation.*
VCEG:	*Video Coding Experts Group.*
VOD:	*Video On Demand.*

CAPÍTULO 1

Introducción

Las técnicas involucradas en el proceso de compresión de vídeo se encuentran inmersas en un continuo proceso de transformación y evolución. Este hecho viene determinado por las necesidades de un mercado altamente dinámico, el cual incorpora rápidamente los resultados obtenidos en los diferentes centros de investigación. En este sentido, la estimación de movimiento, por su relevancia dentro del proceso de compresión híbrida de vídeo, continúa centrando los esfuerzos de numerosos investigadores, con el objetivo de proponer nuevos algoritmos y arquitecturas que contribuyan a mejorar sus prestaciones.

1 Introducción

En este primer capítulo se señalan los motivos que dan origen a enfocar esta Tesis en el proceso de estimación de movimiento, describiéndose asimismo los objetivos a alcanzar y la estructura en capítulos del presente trabajo.

1.1 Planteamiento del problema

Desde que a principios del siglo pasado Arthur Korn anunciara el nacimiento de la *telefotografía* [Kor07] al haber conseguido enviar con éxito una fotografía en escala de grises a través de una línea telegráfica, la transmisión de imágenes estáticas y/o en movimiento ha despertado un creciente interés en la comunidad científica que hoy en día permanece todavía vigente. Sin duda alguna, el continuo desarrollo de mecanismos de compresión de imagen y vídeo es uno de los factores que más ha contribuido al aumento de la eficiencia en este tipo de transmisiones. La considerable reducción del volumen de datos a transmitir, junto con el espectacular avance en los sistemas de telecomunicación, ha hecho que los once minutos y medio empleados por el profesor Korn en transmitir la pequeña imagen del rey Eduardo VII mostrada en la Figura 1.1, entre las ciudades de Munich y Nuremberg, resulten en la actualidad irrisorios.

Figura 1.1: Fotografía enviada por Arthur Korn en 1904 (izquierda) junto con la portada del número XXVII de la revista *Je sais tout* (derecha) en la que el propio Korn relataba sus experimentos en transmisión de imágenes.

Dentro de este nuevo escenario, el proceso de normalización y estandarización al que hemos asistido en estos últimos veinte años ha desempeñado un papel fundamental. Como resultado principal de este proceso, en la actualidad se dispone de un conjunto poderoso de estándares

de compresión de imagen y vídeo que permiten afrontar la gestión de este tipo de información con garantías. La progresiva evolución de dichos estándares ha hecho posible la aparición de aplicaciones y servicios como la videoconferencia, el vídeo bajo demanda (*Video On Demand – VOD*), la televisión de alta definición (*High Definition TeleVision – HDTV*) o la telefonía móvil de tercera y cuarta generación entre otras, provocando con ello un explosivo crecimiento de la industria del sector audiovisual.

Aunque los diferentes estándares de codificación de vídeo han sido desarrollados para aplicaciones muy diversas, existen una serie de similitudes que han determinado, y siguen determinando, gran parte de la actividad investigadora dentro de este campo:

- En todos los estándares se define una sintaxis de trama y un decodificador elemental, dejando absoluta libertad a los diseñadores en la definición del codificador, siempre y cuando éste genere tramas conforme a las directrices establecidas por el estándar.

- Con el objetivo de aumentar al máximo la eficiencia de compresión, la inmensa mayoría de los modelos e implementaciones de codificadores de vídeo están basados en esquemas de codificación híbrida [Hab74], [RPR77], consistentes en la reducción de las redundancias espaciales y temporales presentes en una señal de vídeo digital.

- La reducción de las redundancias temporales se lleva a cabo mediante la estimación del movimiento existente entre imágenes muestreadas en diferentes instantes. Este proceso resulta ser clave en la evaluación de las prestaciones de un codificador híbrido de vídeo, pues contribuye en gran medida al aumento de la eficiencia de compresión [BK97, pp. 105-117] a costa, sin embargo, de un elevado coste computacional [BK97, pp. 253-259]

La conjunción de estos tres factores ha determinado que la estimación de movimiento se haya convertido en un *tema caliente* de discusión. Así, y en paralelo al proceso de estandarización, los avances obtenidos dentro de este campo se han ido incorporando paulatinamente a los estándares de compresión, destacando los relativos al proceso de estimación de movimiento: precisión sub-píxel en los vectores de movimiento, estimación de movimiento bidireccional, estimación de movimiento con tamaño de bloque variable, ampliación del rango de los vectores de movimiento hasta apuntar fuera de la imagen, o estimación de movimiento multifotograma, entre otros [Sad02, pp. 30-65], [Ric03, pp.30-41].

Sin embargo, debido fundamentalmente al imparable avance de la industria audiovisual, las necesidades de compresión son cada vez más exigentes. Este hecho ha determinado que la investigación dentro de este campo se centre cada vez más en la consecución de un objetivo aparentemente contradictorio: ***más por menos***, o lo que es lo mismo, más compresión y mayor calidad de imagen por menos bits transmitidos y un menor uso de recursos. Para lograr este ambicioso objetivo, la comunidad científica ha puesto de manifiesto la imperiosa necesidad de seguir avanzando en el área de compresión de vídeo, y como consecuencia directa, en el proceso de estimación de movimiento, tanto desde el punto de vista algorítmico [Sik05] como arquitectural [TC04], [TCH+05].

1.2 Motivación de la Tesis

El proceso de estimación de movimiento en la compresión híbrida de vídeo consiste en determinar el movimiento existente entre dos imágenes pertenecientes a una misma secuencia de vídeo. Los algoritmos más utilizados para la implementación de la etapa de estimación de movimiento en los codificadores de vídeo actuales son los denominados algoritmos de ajuste de bloques (*block matching*), los cuales destacan por su simplicidad y

robustez [JJ81]. Estos algoritmos se basan en comparar cada bloque de $m \times n$ píxeles de la imagen que se desea codificar (bloque de referencia) sobre un área reducida (área de búsqueda) de una imagen anteriormente codificada. El resultado de la estimación es un vector, denominado *vector de movimiento*, que identifica el desplazamiento entre el bloque de referencia y el bloque *con mayor similitud* de los pertenecientes al área de búsqueda.

Este proceso de estimación de movimiento se realiza en la inmensa mayoría de los codificadores híbridos de vídeo, independientemente del estándar que utilicen. Por esta razón, es un elemento clave en plataformas de compresión de vídeo multiestándar, las cuales han cobrado especial interés en estos últimos años tanto en el ámbito universitario [Gar04] como en el sector industrial [PSA+03], [Nomadik04]. Estas plataformas han sido diseñadas con el objetivo de optimizar, tanto desde el punto de vista algorítmico como arquitectural, el proceso de codificación y decodificación de señales de vídeo de mediana y baja resolución espacio-temporal según los perfiles básicos de los estándares H.263 [H263] y MPEG-4 [MPEG4]. Sin embargo, esta integración de servicios y estándares ha de conseguirse bajo unos estrictos, a la par que paradójicos, requisitos: obtener aceleradores hardware de excelentes prestaciones, pero que por el contrario, presenten un consumo de potencia y un tamaño final mínimos, todo ello sin perder de vista el coste final de la implementación.

Este ámbito de aplicación se caracteriza igualmente por su alto dinamismo, en el sentido de que constantemente se están produciendo avances que son rápidamente incorporados por el sector. En este sentido, la aparición del estándar H.264/AVC [H264] ha supuesto una auténtica revolución en el campo de la codificación de vídeo, pues permite la obtención de mayores tasas de compresión que sus estándares predecesores, incorporando los mecanismos necesarios para garantizar la transmisión eficiente de la información comprimida a través de cualquier tipo de red de comunicación [TR03], [WSB+03], [OBL+04], [SW05]. No obstante, estas elevadas prestaciones de compresión no se obtienen a *coste cero*, sino a costa de un considerable incremento en el coste computacional [OBL+04], [SDL+04], especialmente

notable en la etapa de estimación de movimiento [ZHY+03], [CHC04a], [HHC+06]. Estos hechos ha motivado la reciente incorporación de dicho estándar en plataformas multiestándar [Nomadik06], así como el desarrollo de nuevas plataformas *hardware/software* capaces de codificar [4i2iENC05], [CCH+06b], [LW06] y decodificar [CJS+04], [HSM+04], [KJB+04], [4i2iDEC05], [CHC+05a], [PCJ+05], [LPK+06] vídeo según las directrices del estándar H.264/AVC.

Todas estas plataformas de codificación hacen uso de algún tipo de algoritmo de estimación de movimiento. Dentro de estos algoritmos, el más popular es, sin duda, el *algoritmo por búsqueda exhaustiva* (*Full Search Block Matching – FSBM*) el cual permite obtener altas tasas de compresión a costa de evaluar todas las posiciones posibles dentro del área de búsqueda [SR85]. Debido a su elevado coste computacional, se han realizado muchísimos esfuerzos con el objetivo de obtener estimadores de movimiento capaces de operar en sistemas bajo condiciones de tiempo real. Como resultado, en la actualidad se dispone de un variopinto conjunto de propuestas arquitecturales capaces de realizar el proceso de estimación de movimiento bajo las mencionadas condiciones, ya sea mediante la implementación eficiente en hardware del algoritmo de búsqueda exhaustiva o la implementación de algún tipo de *algoritmo de estimación de movimiento rápido*, basados en reducir el coste computacional demandado por el algoritmo de búsqueda exhaustiva sin sacrificar demasiado la eficiencia de compresión. Sin embargo, las soluciones presentadas hasta la fecha presentan al menos uno de los siguientes inconvenientes:

- **Rigidez computacional**. Los denominados *algoritmos rápidos de estimación de movimiento* presentan un coste computacional muy inferior al demandado por la solución exhaustiva [Kuh99, pp. 17-60], [ACB03], [LO04], [HCT+06]. Sin embargo, en la mayoría de ellos dicho coste es fijo, sin posibilidad de poder ser aumentado o

disminuido de acuerdo a las características de la secuencia de vídeo, a las necesidades del usuario de la aplicación de vídeo correspondiente y/o al estándar en uso.

- **Imposibilidad de adaptación a la secuencia de vídeo**. Este hecho determina que las prestaciones de dichas arquitecturas dependan extraordinariamente de las características espaciales y temporales de la secuencia de vídeo a comprimir. Si estas características (típicamente preestablecidas) no se cumplen, se obtiene una seria degradación en los niveles de compresión alcanzados, situación que se produce especialmente en las arquitecturas basadas en algún tipo de algoritmo rápido. Por otra parte, si el funcionamiento de la arquitectura se rige bajo las directrices del algoritmo de búsqueda exhaustiva, pueden darse situaciones de sobreesfuerzo computacional con determinadas secuencias, con el consiguiente aumento en la potencia dinámica disipada.

- **Imposibilidad de adaptación a las necesidades del usuario**. Los comités de expertos encargados de coordinar el proceso de estandarización coinciden en que el siguiente paso en este proceso debe consistir en ofrecer al usuario, dentro de los diferentes estándares de compresión, soluciones escalables. La prueba más evidente de esta tendencia la representa la futura ampliación del nuevo estándar H.264/AVC denominada SVC (*Scalable Video Coding*) [SVC], la cual propone soluciones de escalabilidad para el mencionado estándar y que, previsiblemente, se incorporará como anexo de éste de manera oficial a finales del año 2006. Sin embargo, esta tendencia no se ha visto reflejada en el proceso de estimación de movimiento, constatándose en la bibliografía la carencia de estimadores de movimiento que ofrezcan al usuario un conjunto de soluciones de compromiso eficaces en términos de niveles de compresión y uso de recursos.

- **Ausencia de comportamiento multiestándar**. El desarrollo de los estándares de codificación de vídeo ha venido acompañado del estudio de nuevas posibilidades en el proceso de estimación de movimiento. Así, en el reciente estándar H.264/AVC existe la posibilidad de calcular hasta un total de 41 vectores de movimiento con precisión de cuarto de píxel para un mismo macrobloque (bloque de 16×16 píxeles), mientras que en el pionero estándar H.261 [H261] sólo se permite un vector de movimiento con precisión de píxel por macrobloque. Si bien las propuestas más recientes cumplen total o parcialmente con las recomendaciones establecidas por H.264/AVC, y a su vez éste engloba a sus predecesores en lo que a estimación de movimiento se refiere, dichas arquitecturas son incapaces de adaptar su potencia de cómputo a las necesidades del estándar en uso por el codificador correspondiente.

Este conjunto de circunstancias, junto con el continuo avance de las plataformas de compresión de vídeo, determina que sea absolutamente necesario el diseño de nuevas etapas de estimación de movimiento multiestándar que soporten las recomendaciones establecidas por el estándar H.264/AVC y que a la vez, hagan un uso óptimo de un número de recursos escaso. Para ello, la triple adaptación de la arquitectura a las necesidades del usuario, a las características de las secuencias de vídeo a comprimir y a los requisitos del estándar de codificación en uso, representa una eficaz estrategia a seguir.

1.3 Objetivos de la Tesis

La finalidad de esta Tesis Doctoral es proporcionar soluciones algorítmicas y arquitecturales para etapas de estimación de movimiento multiestándar haciendo un uso inteligente de los recursos disponibles. Para conseguir este propósito, la Tesis se centra en alcanzar los siguientes objetivos:

- Estudio del estado del arte y en especial, de las arquitecturas propuestas en las que se implemente, con algún grado de adaptación de los recursos de cómputo, el proceso de estimación de movimiento mediante ajuste de bloques para cualquiera de los estándares de compresión de vídeo existentes. En este punto, es de especial importancia analizar y evaluar la estrategia de adaptación utilizada, tanto desde el punto de vista algorítmico como desde el punto de vista puramente arquitectural.

- Creación de un entorno de análisis que permita investigar en profundidad el proceso de estimación de movimiento y que a la vez, sirva como soporte para la elección de la estrategia de diseño que se aportará en esta Tesis. Este entorno se basará en la simulación y modelado de un estimador de movimiento genérico y permitirá extraer los parámetros esenciales del proceso sobre los cuales se construirán las propuestas desarrolladas en la Tesis.

- Aportar y evaluar un nuevo algoritmo que, en función de los parámetros mencionados en el punto anterior, sea capaz de estimar el movimiento entre dos imágenes de una secuencia de vídeo según las necesidades del estándar de codificación en uso, utilizando eficientemente los recursos disponibles y adaptándolos a las exigencias del usuario y a las características de la secuencia a comprimir. Para evaluar la bondad de este algoritmo, se utilizará una plataforma de validación que integre la solución aportada en codificadores basados en los estándares H.263 y H.264/AVC, de tal manera que cumpla con las características de estimación de movimiento para tamaños de bloque variables y precisión sub-píxel, tal y como se requiere por ambos estándares. Esta plataforma permitirá la evaluación de las prestaciones del algoritmo aportado para diferentes secuencias y tasas de compresión. En este punto es importante destacar que los resultados y conclusiones que se obtengan para el estándar H.263, serán absolutamente extrapolables para sistemas basados en MPEG-

4, al ser ambos estándares idénticos en sus perfiles básicos en cuanto al proceso de estimación de movimiento.

- Analizar las diferentes alternativas para el diseño arquitectural del algoritmo propuesto, desarrollando en paralelo un conjunto de técnicas que permitan aumentar las prestaciones de la arquitectura, tanto para la estimación de movimiento con precisión entera como para la estimación con precisión sub-píxel. Esta exploración permitirá evaluar y comparar, en términos arquitecturales, el conjunto de posibles soluciones.

- Proponer y evaluar una arquitectura que, una vez analizadas las diferentes alternativas arquitecturales, sea capaz de realizar el proceso de estimación de movimiento en tiempo real para formatos de vídeo de baja y media resolución según las directrices establecidas por los estándares H.263 y H.264/AVC de acuerdo al algoritmo propuesto. Asimismo, y de manera independiente al algoritmo propuesto, es también objetivo de esta Tesis mejorar las prestaciones arquitecturales de los trabajos previos más significativos recopilados en el estado del arte.

1.4 Organización de la Tesis

El trabajo desarrollado en esta Tesis Doctoral se ha estructurado en cinco capítulos, de los cuales el primero de ellos lo constituye el presente capítulo de introducción. El contenido del resto de los capítulos se describe a continuación.

Capítulo 2

En este capítulo se describe la evolución histórica del proceso de estandarización en compresión de imagen y vídeo, detallándose previamente la estructura de un codificador híbrido de vídeo genérico, base sobre la que se asientan los mencionados estándares. Asimismo, se recogen las novedades más significativas introducidas en el proceso de estimación de movimiento a lo largo del proceso de estandarización, finalizando el capítulo con una recopilación de los trabajos que, tanto desde el punto de vista algorítmico como arquitectural, constituyen el estado del arte en el ámbito de la estimación de movimiento.

Capítulo 3

En el capítulo 3 se describe el entorno de análisis propuesto en esta Tesis con el objetivo de interpretar, mediante el uso de parámetros, el proceso de estimación de movimiento en términos de la función de coste de Lagrange. A partir de los resultados obtenidos con dicho entorno, se presenta el algoritmo de estimación de movimiento adaptativa propuesto en esta Tesis, mostrándose los resultados obtenidos en términos de compresión y coste computacional. Por último, se detallan las variaciones introducidas con los objetivos de adaptar dicho algoritmo a estándares con estimación de movimiento de tamaño de bloque variable y facilitar su posterior implementación hardware.

Capítulo 4

En este capítulo se introduce la arquitectura de estimación de movimiento propuesta en esta Tesis. En particular, se describen por separado la arquitectura de estimación de movimiento con precisión entera y la arquitectura de refinamiento de vectores de movimiento a coordenadas de medio y cuarto de

píxel, poniendo de relevancia las aportaciones introducidas con el objetivo de mejorar sus prestaciones. Asimismo, se presenta una comparación en términos arquitecturales con trabajos previos recientemente publicados.

Capítulo

5 Por último, se presentan en el capítulo 5 las conclusiones extraídas a partir del trabajo desarrollado en esta Tesis así como las líneas de investigación futuras que se pretenden continuar.

1 Introducción

CAPÍTULO 2

Estado del arte de la codificación híbrida de vídeo

El avance de la industria audiovisual ha convertido la codificación y decodificación de imágenes y vídeo en un proceso que, aunque transparente para el cliente final, resulta de uso cotidiano para millones de usuarios de aplicaciones multimedia. Para llegar a esta situación ha resultado absolutamente necesario desarrollar un conjunto de estándares que no sólo han permitido regularizar dicho proceso, sino también, mejorarlo en términos de las tasas de compresión alcanzadas.

En este capítulo se revisan las bases sobre las que se asientan los estándares de compresión de imagen y vídeo actuales. Asimismo, se describen las modificaciones

que han ido incorporando estos estándares en relación con el proceso de estimación de movimiento con el objetivo de aumentar las prestaciones de los codificadores híbridos de vídeo basados en algún tipo de estándar. Dicho proceso de optimización ha llevado consigo un aumento exponencial de la complejidad de la estimación de movimiento por ajuste de bloques, despertando el interés de la comunidad científica por el desarrollo de nuevos algoritmos y arquitecturas capaces de acelerar el proceso de cálculo de vectores de movimiento. En este sentido, se recogen en este capítulo las aportaciones más significativas realizadas en esta área, poniendo de manifiesto aquellos aspectos aún sin resolver y que constituyen el objeto principal de esta Tesis.

2.1 Codificación híbrida de vídeo

La codificación o compresión híbrida de vídeo es un campo específico dentro del procesamiento de señal multidimensional cuyo objetivo fundamental es obtener una representación compacta de una señal de vídeo digital cualquiera mediante la reducción de las redundancias espaciales y temporales presentes en la señal original. Este proceso puede ser reversible (compresión sin pérdidas) o irreversible (compresión con pérdidas), obteniéndose de manera general mayores niveles de compresión, y por lo tanto una peor calidad de imagen, en el segundo caso.

En este apartado, se describen las técnicas sobre las que se fundamenta un codificador híbrido de vídeo genérico, así como las características más relevantes de los diferentes estándares que a lo largo de estos últimos años han sido desarrollados con el objetivo de normalizar, a la par que enriquecer y mejorar, dicho proceso de compresión. En este sentido, se resaltarán los conceptos teóricos directamente relacionados con los objetivos a conseguir en esta Tesis, pudiéndose obtener una visión más amplia y detallada acerca del proceso de compresión de vídeo en las publicaciones [BK97], [Ric02], [Ric03] y [Woo05], entre otras.

2.1.1 Esquema general de funcionamiento de un codificador híbrido de vídeo

El diagrama de bloques de un codificador híbrido de vídeo genérico se muestra en la Figura 2.1, distinguiéndose fundamentalmente tres unidades funcionales: unidad de preproceso, unidad de reducción de redundancias espaciales y unidad de reducción de redundancias temporales.

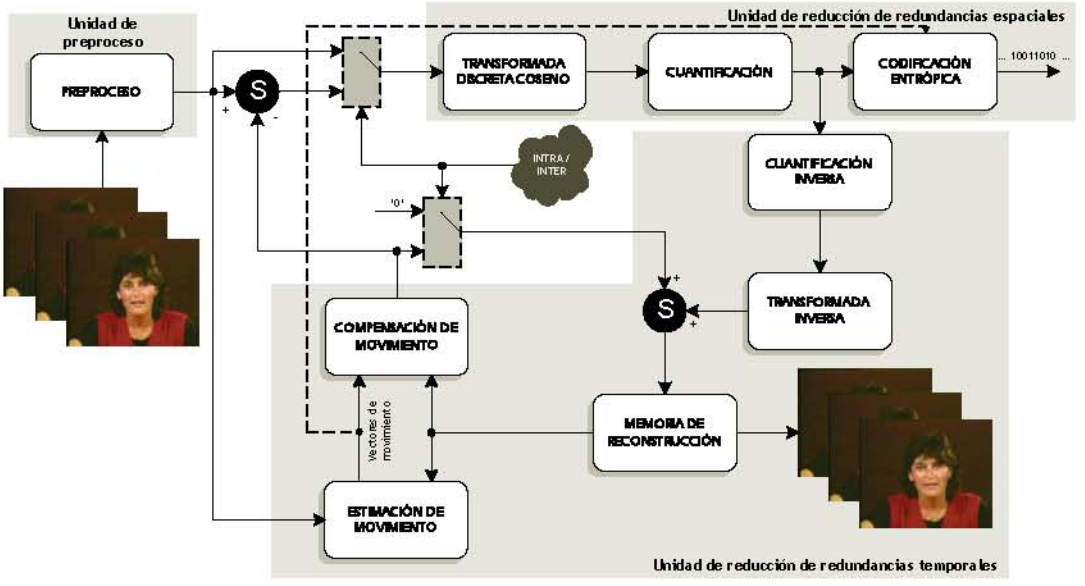


Figura 2.1: Diagrama de bloques de un codificador híbrido de vídeo genérico.

2.1.1.1 Unidad de preproceso

La unidad de preproceso tiene dos funciones claramente diferenciadas. En primer lugar, es la encargada de convertir el espacio de colores asociado a la secuencia de entrada de tamaño genérico $M \times N$ píxeles, al espacio propio del codificador de vídeo, resultando el cambio del espacio RGB (Rojo – Verde – Azul) al espacio YCbCr (Luminancia – Crominancia azul – Crominancia roja) el más habitual, pues este último representa el espacio de colores adoptado por los estándares de compresión de vídeo. El resultado de esta transformación son tres matrices de iguales dimensiones ($M \times N$ píxeles) que las originales, obtenidas mediante la aplicación directa de las siguientes ecuaciones [BT.601]:

$$Y = 0{,}299 \cdot R + 0{,}587 \cdot G + 0{,}114 \cdot B$$

$$Cb = 0{,}564 \cdot (B-Y) \qquad \text{(ec. 2.1)}$$

$$Cr = 0{,}713 \cdot (R-Y)$$

Por otro lado, el sistema visual humano es más sensible a las variaciones de la luminancia (*luminosidad* o *intensidad*) que a las variaciones del color de una imagen, debido fundamentalmente al mayor número de bastones con respecto al número de conos presentes en la retina humana [Kol03]. De esta manera, al realizar la mencionada transformación, se pasa de un espacio de colores en el cual sus tres componentes (RGB) poseen una entropía similar, a otro en el cual la información más importante para el ojo humano posee un grado de dispersión menor (YCbCr). Este hecho posibilita reducir la resolución de las dos componentes de crominancia sin que ello conlleve una pérdida apreciable de la calidad de la imagen, reduciendo el volumen de datos a almacenar y dando lugar a los formatos de diezmado conocidos como 4:2:2, 4:2:0 y 4:1:1. En cada uno de estos formatos, un conjunto de muestras de luminancia comparten un conjunto menor de muestras de ambas crominancias, tal y como se indica en la Figura 2.2, en la que también se muestra el formato sin diezmado (4:4:4). El esquema de muestreo YCbCr 4:2:0 se corresponde con el formato de entrada para los estándares de codificación de vídeo H.263 y H.264/AVC, y en general, por la mayoría de los estándares de compresión, y por lo tanto, será el utilizado en esta Tesis.

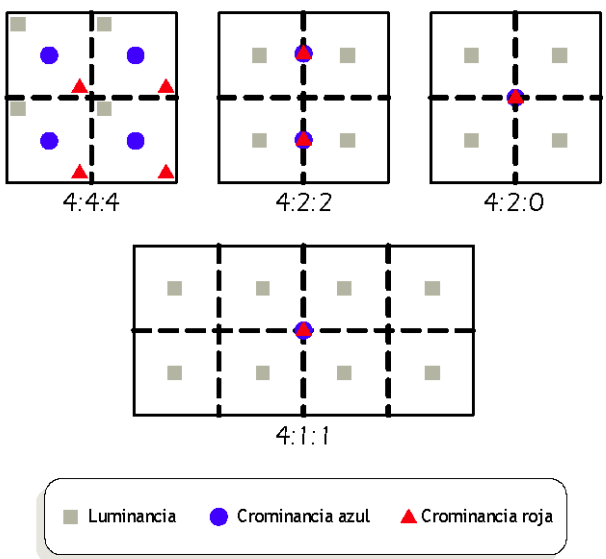


Figura 2.2: Distribución de muestras de luminancia y crominancias para los formatos de secuencia de entrada 4:4:4, 4:2:2, 4:2:0, y 4:1:1.

La segunda función de la unidad de preproceso consiste en dividir la secuencia de entrada en macrobloques, pues todo el procesamiento llevado a cabo por el codificador híbrido de vídeo se realiza a este nivel. Para el formato de muestreo YCbCr 4:2:0, cada macrobloque está compuesto por seis bloques de 8×8 píxeles, cuatro de ellos de muestras de luminancia, uno de crominancia azul y otro de crominancia roja, tal y como se indica en la Figura 2.3.

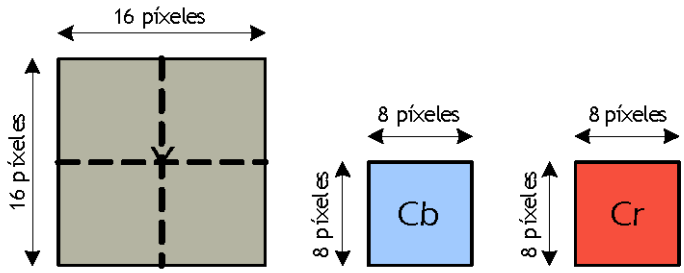


Figura 2.3: Estructura de un macrobloque.

De esta manera, los $(M \times N)/(16 \times 16)$ macrobloques presentes en cada uno de los fotogramas de una secuencia de vídeo de entrada serán procesados en orden lexicográfico (de izquierda a derecha y de arriba abajo) por el resto de bloques funcionales que componen el codificador híbrido hasta completar su procesamiento.

2.1.1.2 Unidad de reducción de redundancias espaciales

Cada uno de los fotogramas que componen una secuencia de vídeo real presenta, en la mayoría de los casos, una alta redundancia espacial en el sentido de que gran parte de los píxeles pertenecientes a un mismo fotograma son muy similares a sus inmediatos vecinos. Este hecho, junto con las características del sistema visual humano, es explotado por los bloques funcionales que componen la unidad de reducción de redundancias espaciales, con el objetivo de obtener una versión comprimida del fotograma sin que se produzca una pérdida apreciable de calidad.

Dentro de la unidad de reducción de redundancias espaciales (ver Figura 2.1) se llevan a cabo tres procesos de manera consecutiva para cada uno de los macrobloques de la secuencia de entrada:

- **Transformada discreta del coseno (DCT)**, encargada de obtener, para cada uno de los bloques de 8×8 píxeles que constituyen un macrobloque, su representación equivalente en el dominio frecuencial. De esta manera se obtiene una separación ordenada entre las componentes frecuenciales de la imagen bajo análisis, quedando de relevancia la cantidad de redundancia espacial presente en el bloque de píxeles procesado mediante la inspección de los valores de los coeficientes transformados obtenidos.

- **Cuantificación**. Este bloque funcional es el encargado de cuantificar los valores de los coeficientes obtenidos por la DCT, además de eliminar un número de coeficientes de alta frecuencia de acuerdo con el nivel de compresión que se pretenda alcanzar. Al ser la sensibilidad del sistema visual humano menor para las frecuencias espaciales altas, esta circunstancia se utiliza para transmitir una menor cantidad de información, traduciéndose este punto del proceso de codificación en la puesta a cero de determinados coeficientes transformados obtenidos a la salida de la DCT. Este proceso se lleva a cabo mediante la división entera de los coeficientes DCT del bloque de 8×8 píxeles bajo procesamiento entre los coeficientes de la *matriz de cuantificación* propia del cuantificador. Dicha matriz posee, por norma general, coeficientes de mayor magnitud para las posiciones matriciales asociadas a altas frecuencias espaciales, facilitando de esta manera la eliminación (puesta a cero) de las mismas. El número de coeficientes a eliminar y por lo tanto, el nivel de compresión que se puede alcanzar, estará determinado por un factor multiplicativo definido para toda la matriz, denominado *escalón de cuantificación* (Q_p). Es importante destacar que el

proceso de cuantificación es el único irreversible dentro de un codificador híbrido de vídeo y que por lo tanto, es el único responsable de que la compresión resultante sea con o sin pérdidas. En este sentido, en esta Tesis se indicará mediante la condición $Q_p = 0$ que el proceso de compresión se realiza sin pérdidas.

- **Codificación entrópica**. El último paso en el proceso de reducción de redundancias espaciales consiste en una codificación de longitud variable en la cual los símbolos con mayor probabilidad de aparición son codificados con una longitud de código menor y viceversa. Dichos símbolos se construyen a partir del número de ceros presentes en cada bloque de píxeles, obteniéndose así mejores prestaciones a medida que el número de ceros aumenta. En los actuales estándares de compresión de imagen y vídeo, esta codificación de longitud variable se puede realizar mediante codificadores de longitud de código entera (tipo *Huffman*) o mediante codificadores aritméticos [Ric03, pp. 62-72].

Una vez terminado este proceso, se obtiene una cadena de bits de tamaño considerablemente menor que la asociada a la secuencia original, siendo el flujo de procesamiento descrito la base de los estándares de compresión de imagen estática. Sin embargo, en el caso de imágenes en movimiento, el proceso de compresión se puede optimizar en mayor medida si no se codifica individualmente cada uno de los fotogramas (codificación tipo *INTRA*) sino con respecto a otros fotogramas previamente codificados, reduciéndose de esta manera las redundancias temporales (codificación tipo *INTER*).

2.1.1.3 Unidad de reducción de redundancias temporales

Entre dos fotogramas consecutivos de una misma secuencia de vídeo existen redundancias temporales en el sentido de que los píxeles de cada posición suelen ser, salvo situaciones de

cambio de contexto o movimientos bruscos, muy similares. Este hecho posibilita conseguir una mayor cantidad de redundancias espaciales al codificar un fotograma de manera diferencial con respecto a otro fotograma de la misma secuencia, pudiéndose obtener así mayores niveles de compresión.

El proceso necesario para obtener dicho fotograma *diferencia* se divide conceptualmente en los siguientes pasos:

- **Recuperación de imagen**. Este proceso se lleva a cabo mediante la concatenación de los bloques funcionales *Transformada Inversa* y *Cuantificación Inversa* mostrados en la Figura 2.1, obteniéndose una versión con pérdidas del bloque de píxeles que esté siendo procesado por el codificador debido al proceso de cuantificación. Estos píxeles son almacenados de manera ordenada en la *Memoria de Reconstrucción* de modo que siempre se disponga de los bloques previamente codificados correspondientes al fotograma actual y, al menos, de los del fotograma previamente codificado al completo. Puesto que el proceso de codificación entrópica es absolutamente reversible, los fotogramas que se almacenen en la mencionada memoria serán idénticos a los obtenidos por un decodificador compatible con el codificador utilizado.

- **Estimación de movimiento**. La estimación de movimiento se encarga de encontrar, para cada uno de los macrobloques de la secuencia de vídeo a comprimir, dónde se encuentra el macrobloque *más parecido* en un fotograma previamente codificado, y por lo tanto, almacenado en la memoria de reconstrucción. El resultado de dicha búsqueda es un *vector de movimiento* que indica de manera cartesiana la posición de dicho macrobloque dentro del fotograma codificado con anterioridad. Tal y como se indica en la Figura 2.4, en el contexto propio de la estimación de movimiento, el macrobloque del que se desea calcular su vector de movimiento asociado se denomina

macrobloque de referencia, mientras que la zona en la cual se realiza la búsqueda en un fotograma previamente codificado se denomina *área de búsqueda*.

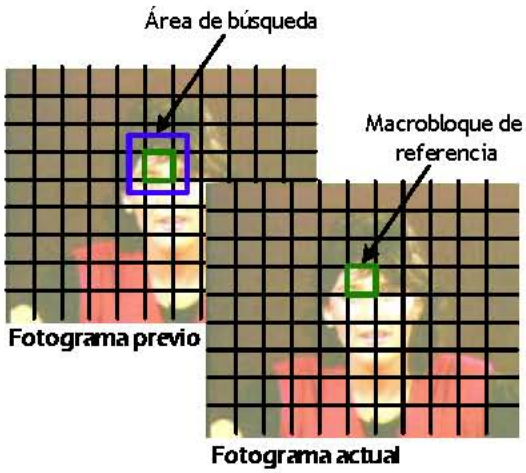


Figura 2.4: Macrobloque de referencia y área de búsqueda en el contexto propio de la estimación de movimiento en codificadores híbridos de vídeo.

A pesar de que existen varias métricas para encontrar el vector de movimiento de cada macrobloque [FS98], la más utilizada es la suma de diferencias absolutas (*Sum of Absolute Differences – SAD*) definida como:

$$SAD(u,v) = \sum_{i=1}^{m}\sum_{j=1}^{n}\left|r(i,j) - b(i+u,j+v)\right| \qquad \text{(ec. 2.2)}$$

$$-p1 < u,v < p2,$$

donde *p1* y *p2* son los límites en píxeles del área de búsqueda, *r(i,j)* y *b(i,j)* las muestras de luminancia del macrobloque de referencia y del área de búsqueda, respectivamente, *m* y *n* las dimensiones del macrobloque de referencia en píxeles (normalmente, $m = n = 16$) y *(u,v)* las coordenadas cartesianas del vector de movimiento evaluado. Este procedimiento se conoce como estimación de movimiento por ajuste de bloques (*block matching*) del cual, por su especial relevancia en el marco

de esta Tesis, se realizará posteriormente una descripción detallada, tanto desde el punto de vista algorítmico, como arquitectural.

- **Compensación de movimiento**. La compensación de movimiento obtiene de la memoria de reconstrucción el macrobloque señalado por el vector calculado por el estimador de movimiento. Así, a la salida del compensador de movimiento se obtiene un *predictor*, cuya resta con el macrobloque actual constituye el macrobloque diferencia que será procesado por la unidad de reducción de redundancias espaciales.

Finalmente, cabe señalar dentro de la estructura genérica de un codificador híbrido de vídeo un conjunto de elementos de control encargados de seleccionar el tipo de codificación para cada macrobloque de la secuencia de vídeo (*INTRA*/*INTER*). Obviamente, todos los macrobloques del primer fotograma han de ser necesariamente codificados como tipo *INTRA*, decidiendo el codificador el modo de codificación a utilizar para el resto de macrobloques.

2.1.2 Estándares de compresión de imagen y vídeo

La aparición de técnicas eficientes de compresión de datos multimedia ha determinado que, durante estos últimos veinte años, se haya realizado por parte de grupos de investigación y compañías del sector un exhaustivo esfuerzo de normalización, dando lugar a un potente conjunto de estándares de compresión de imagen y vídeo. Con el objetivo de caracterizar y comparar las prestaciones de cada estándar, se utilizan un conjunto de curvas conocidas como curvas de tasa de transmisión-distorsión (*rate-distortion*), en las que se representa la calidad de la imagen o secuencia decodificada frente a la tasa de transmisión utilizada. Para medir de manera objetiva la calidad de una imagen decodificada de $M \times N$ píxeles, la métrica

más utilizada es la relación de pico señal a ruido (*Peak Signal to Noise Ratio – PSNR*) expresada en unidades de decibelios, y definida como:

$$PSNR(dB) = 10 \cdot \log_{10} \frac{255^2}{\frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} \left| p_{orig}(i,j) - p_{dec}(i,j) \right|}, \qquad \text{(ec. 2.3)}$$

donde $p_{orig}(i,j)$ y $p_{dec}(i,j)$ representan los píxeles de la imagen original y decodificada, respectivamente. En secuencias de vídeo, cada uno de los puntos de la mencionada curva representa el PSNR medio obtenido considerando todos los fotogramas de la secuencia.

Por último, recordar que cada estándar sólo define la estructura del decodificador, dejando un holgado grado de libertad para el diseño del codificador siempre y cuando las tramas generadas por éste sean compatibles con el decodificador definido.

2.1.2.1 Estándares de compresión de imagen

Dentro de los estándares de compresión de imagen destaca, por su relevancia, el estándar JPEG (*Joint Photographic Experts Group*) [JPEG], que debe su nombre al grupo encargado de su desarrollo, fruto de la unión de los expertos en compresión de imágenes en color de las organizaciones ISO (*International Organization for Standardization*) e ITU (*International Telecommunication Union*). Dicho estándar permite comprimir imágenes digitales de tono continuo con unos rangos de compresión típicos de 10:1 hasta 50:1 garantizando una calidad de imagen aceptable [PM92].

Con el objetivo de aumentar las prestaciones de este estándar, el mismo comité de expertos ha desarrollado recientemente el estándar JPEG2000 [JPEG2K] que permite aumentar el

nivel de compresión entre un 11% y un 53% con respecto a su predecesor [CSE00] debido principalmente al uso de transformadas *wavelet* en lugar de la transformada DCT [LLT+01], [LCT+04], [LCL+05c] y codificadores aritméticos en la codificación entrópica del proceso de reducción de redundancias espaciales [ATL+02], [ALL+03].

2.1.2.2 Estándares de compresión de vídeo

Dentro de los estándares de compresión de vídeo, destacan como pioneros los estándares H.261 [H261] y MPEG-1 [MPEG1] desarrollados por el Grupo de Expertos en Codificación de Vídeo (*Video Coding Experts Group - VCEG*) de la ITU (*International Telecommunication Union*) y por el grupo de expertos en vídeo y audio digital MPEG (*Moving Pictures Expert Group*) de la organización ISO (*International Organization for Standardization*), respectivamente. Aunque coetáneos, el objetivo de cada uno de estos estándares es bien distinto, puesto que mientras que H.261 fue ideado para aplicaciones de videotelefonía, el objetivo fundamental de MPEG-1 es el almacenamiento eficiente de vídeo digital, típicamente en formato CD-ROM. De esta manera, H.261 alcanza sus máximas prestaciones para anchos de banda múltiplos enteros de 64 Kbps, mientras que MPEG-1 lo hace para tasas de transmisión de 1.5 Mbps.

Debido al tremendo éxito de ambos estándares, los posteriores trabajos de estandarización dieron lugar al nacimiento del estándar MPEG-2 (Recomendación H.262 de la ITU) [MPEG2]. El principal objetivo de este estándar es la codificación y transmisión eficiente de imágenes de televisión para tasas de transmisión por debajo de 10 Mbps, respetando la compatibilidad con los estándares H.261 y MPEG-1, y permitiendo como formato de entrada no sólo vídeo progresivo, sino también vídeo entrelazado. Para lograr este objetivo, MPEG-2 conserva las técnicas de compresión que resultaron ser más eficaces dentro del estándar

MPEG-1, destacando entre ellas la predicción bidireccional y la estimación de movimiento con precisión de medio píxel, además de introducir nuevos modos de predicción específicos para el formato entrelazado.

El estándar MPEG-4 [MPEG4] surge con el doble objetivo de aumentar las prestaciones de compresión alcanzadas por los estándares anteriores para tasas de transmisión bajas, además de proporcionar interactividad con el usuario. Para lograrlo, dicho estándar contempla el uso de técnicas avanzadas de compresión además de dividir la secuencia en *objetos audiovisuales*, permitiendo de esta manera el acceso y manipulación de su contenido. El estándar define un objeto audiovisual como cualquier representación de un objeto, natural o sintético, visual y/o sonoro, como por ejemplo audio natural o sintético, texturas y formas de objetos presentes en un fotograma, objetos sintéticos en dos y tres dimensiones o representaciones sintéticas de caras y cuerpos humanos, entre otros. Además, en el estándar MPEG-4 se presta especial atención a la transmisión eficiente del vídeo comprimido, dotándose de estrategias de protección ante errores y diversos grados de escalabilidad [PE02].

De manera paralela al desarrollo del estándar MPEG-4, el grupo VCEG de la ITU comenzó el desarrollo del estándar H.263 [H263]. La finalidad de este estándar no es otra que, a partir de la estructura básica del estándar H.261, mejorar las prestaciones de compresión para tasas de transmisión muy bajas, típicamente por debajo de 64 Kbps (nótese que el estándar H.261 estaba inicialmente concebido para la transmisión de vídeo en redes de conmutación de circuitos con un ancho de banda de $p \times 64$ Kbps, siendo p un número entero entre 1 y 30). Para conseguir este objetivo el estándar define un núcleo básico de compresión similar al especificado en el estándar H.261 y un conjunto de 18 modos avanzados de compresión opcionales en forma de anexos al estándar. En particular, la recomendación H.263 se compone del mencionado núcleo básico y cuatro de estos modos de codificación opcionales (anexos D, E, F y G), incluyéndose el resto de anexos en las recomendaciones conocidas como

H.263+ [H263+] y H.263++ [H263++] a razón de doce (anexos I a T) y dos (anexos U y V) modos opcionales, respectivamente.

La continua aparición, un tanto desordenada, de diferentes estándares de compresión de vídeo intentando mejorar las prestaciones de sus antecesores, así como la excesiva especialización de éstos para un conjunto específico de aplicaciones, llevó al grupo VCEG al comienzo del desarrollo de un nuevo estándar que presentara prestaciones de compresión superiores al resto para todo tipo de aplicaciones, denominándose dicho estándar H.26L (la letra 'L' responde a la filosofía de trabajo del grupo de VCEG en su intento de crear un estándar de larga duración, *long-term standard*). Una vez comenzado el desarrollo del estándar, la ISO, a través de su grupo MPEG, realizó en el año 2001 una llamada abierta a la comunidad científica internacional en búsqueda de nuevas soluciones dentro del campo de la codificación de vídeo avanzada (*Advanced Video Coding – AVC*). El objetivo de dicho proceso era encontrar nuevas soluciones que permitieran mejorar las prestaciones del estándar MPEG-4 que, aunque prometedor, no había causado un impacto en el mercado similar al de MPEG-2. Entre las propuestas recibidas se encontraban los primeros trabajos realizados por el VCEG dentro del marco del estándar H.26L, resultando ser la propuesta más innovadora a juicio de los expertos del grupo MPEG. Este hecho provocó que ambas organizaciones se fundieran en un nuevo grupo de trabajo conjunto (*Joint Video Team – JVT*) con el objetivo de crear *"una solución única para la próxima generación de estándares de codificación de vídeo"*. Los esfuerzos del grupo de trabajo JVT, dieron lugar al desarrollo de un nuevo estándar denominado H.264/AVC [H264], también conocido como JVT, H.26L o MPEG-4 *part 10*, que se encuentra hoy en día en continua ampliación y desarrollo y que, sin duda alguna, representa el *estado del arte* en el campo de la codificación híbrida de vídeo. En este sentido, el nuevo estándar consigue reducir, por término medio, las tasas de transmisión requeridas por un codificador MPEG-2 en un 50%, y en un 35 % las correspondientes a codificadores basados en los estándares H.263 o MPEG-4 [KA03], [OBL+04]. Este hecho se

puede comprobar en la Figura 2.5, en la que se muestra la evolución de la tasa de transmisión media conseguida por los estándares mencionados durante los últimos diez años para secuencias de 720×480 píxeles.

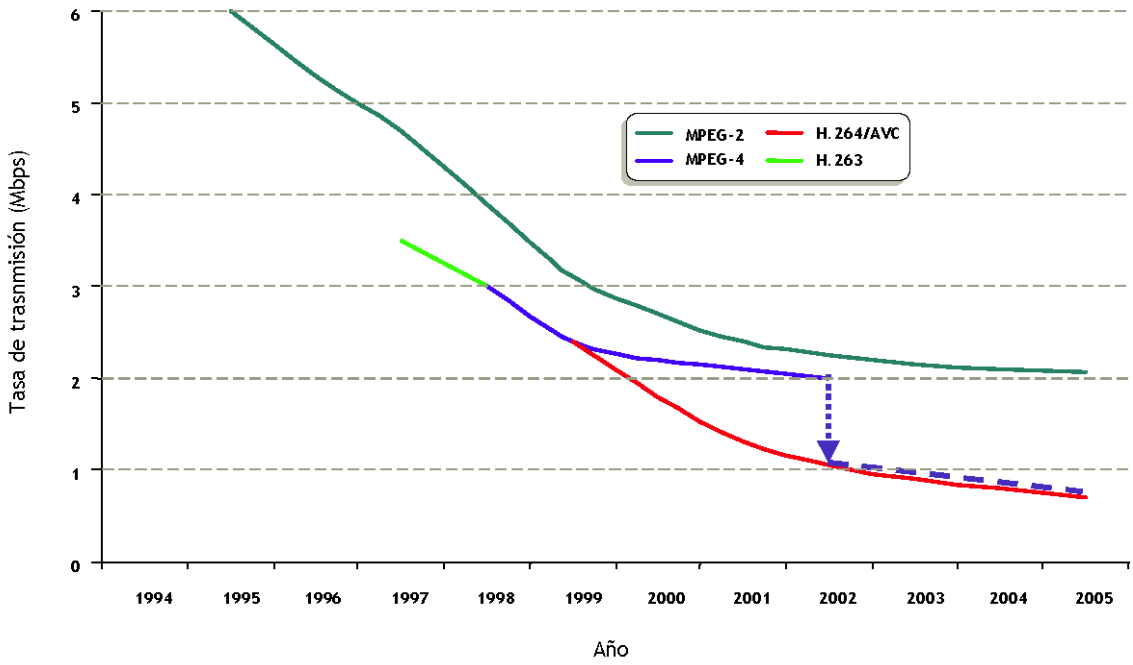


Figura 2.5: Evolución de la tasa de transmisión requerida por los estándares MPEG-2, MPEG-4, H.263 y H.264/AVC para codificar una secuencia de vídeo de 720×480 píxeles.

Asimismo, y a modo de resumen del proceso de estandarización, se muestra en la Figura 2.6 la evolución histórica de cada uno de los estándares mencionados en este apartado. En esta figura no se incluyen los estándares MPEG-7 [MPEG7] y MPEG-21 [MPEG21] debido a que su objetivo no es la compresión de vídeo. En particular, el estándar MPEG-7 representa una interfaz de descripción de contenidos multimedia diseñada para facilitar el acceso, recuperación, filtrado y manejo de datos multimedia. Para ello, este estándar se basa en la realización de bases de datos de información multimedia basadas en contenidos, las cuales permiten realizar búsquedas indexadas de imágenes y vídeo utilizando características como su color, textura, información de su forma y/o contorno de los objetos. En cuanto al estándar MPEG-21, su finalidad es permitir el uso transparente de recursos multimedia y aumentar su

acceso a través de un amplio rango de redes y dispositivos, proporcionando interoperabilidad entre distintos estándares. Así, el ámbito de MPEG-21 cubre funciones como creación, producción, consumo y uso de los contenidos; identificación, descripción y representación de los contenidos, o protección y gestión de la propiedad intelectual, entre otras.

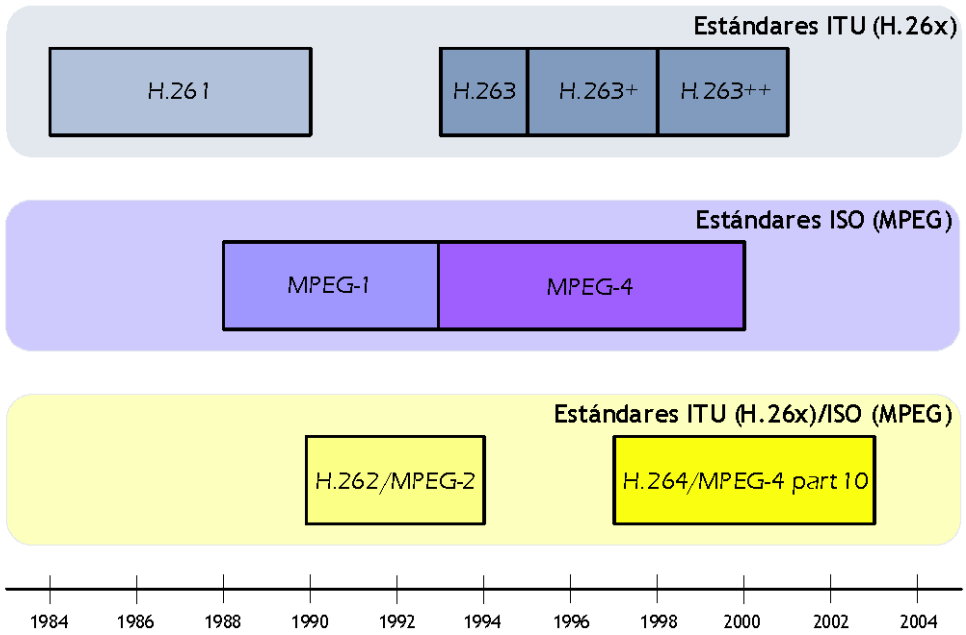


Figura 2.6: Evolución histórica del proceso de estandarización de compresión de vídeo.

De igual manera, tampoco se ha incluido en la Figura 2.6 la extensión del estándar H.264/AVC conocida como SVC (*Scalable Video Coding*), por estar en el momento de la escritura de esta Tesis en fase de desarrollo [SVC]. La idea que persigue dicha extensión del estándar es poder codificar una secuencia de vídeo a la mayor resolución espacio-temporal posible, pero permitiendo a los decodificadores realizar un proceso de decodificación escalonado, obteniendo así una secuencia decodificada con unas características espaciales y temporales acordes con la aplicación en uso y con las características del decodificador. Aunque esta posibilidad ya se había incluido de alguna manera en los estándares MPEG-2 y MPEG-4, la extensión SVC busca mejorar este proceso mediante la introducción de nuevas

técnicas y algoritmos de compresión optimizados para vídeo escalable [SMS+05], [SSM+05], [PKJ+06].

2.1.2.3 Novedades en la etapa de estimación de movimiento durante el proceso de estandarización

El desarrollo de los diferentes estándares mencionados en el apartado anterior ha llevado consigo la modificación y creación de nuevos algoritmos con el objetivo de llevar a cabo cada uno de los procesos propios del codificador híbrido de video de manera más eficiente, y como consecuencia, mejorar las prestaciones de éste. En este sentido, el hecho de que la etapa de estimación de movimiento sea la más influyente en cuanto a las variaciones de las prestaciones de compresión alcanzadas por un codificador híbrido de vídeo, ha determinado que los requisitos de dicha etapa sean cada vez más exigentes, particularmente a partir de la aparición del estándar H.264/AVC.

2.1.2.3.1 Mejoras en los estándares anteriores a H.264/AVC

Tomando como punto de referencia el estándar H.261, las dos primeras variantes a considerar son la introducción de vectores de movimiento con precisión de medio píxel y los fotogramas bidireccionales por parte del estándar MPEG-1.

Los **vectores con precisión de medio píxel** se obtienen típicamente mediante un proceso de búsqueda en dos pasos. En el primero de ellos se obtiene un vector de movimiento con precisión de píxel (precisión entera) según el procedimiento descrito en el apartado 2.1.1.3. Una vez que se ha obtenido un vector con precisión entera, se realiza un refinamiento alrededor de las ocho posiciones situadas a una distancia de medio píxel de ésta, tal y como se muestra en la Figura 2.7, en la que se señalan únicamente las muestras de medio píxel que le

corresponden al píxel situado en la esquina superior izquierda del macrobloque seleccionado por la búsqueda de precisión entera.

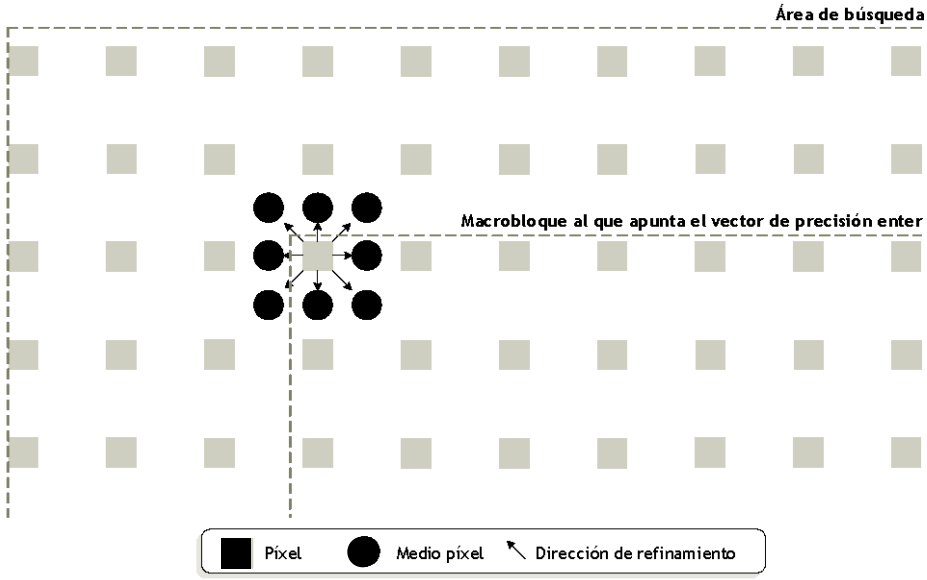


Figura 2.7: Proceso de refinamiento de medio píxel a partir de las coordenadas del vector de movimiento con precisión entera.

Las muestras de medio píxel necesarias para realizar el refinamiento, se calculan mediante interpolación bilineal, tal y como indica la Figura 2.8.

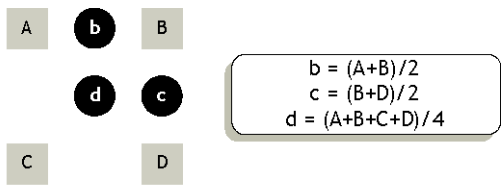


Figura 2.8: Cálculo de muestras de medio píxel mediante interpolación bilineal.

Los **fotogramas bidireccionales** (tipo B) son codificados usando la predicción de movimiento de dos fotogramas, un fotograma *pasado* y otro *futuro*, pudiendo ser ambos tipo *INTRA* (tipo I) o *INTER* (tipo P). Estos fotogramas no se utilizan en la predicción de otros fotogramas tipo B o P y por lo tanto, pueden tener más distorsión y ofrecer un factor de compresión más

alto que los fotogramas I o P. Para codificar o decodificar una imagen tipo B, el codificador y el decodificador necesitarán la imagen I o P que la precede y la imagen P o I que la sigue. El orden de las imágenes será por tanto modificado antes de la codificación, de forma que el codificador y el decodificador dispongan, antes que las imágenes B, de las imágenes I y/o P necesarias para su procesamiento.

Estas variaciones han resultado ser muy beneficiosas en términos de compresión y por ello han sido incorporadas de manera natural en todos los estándares posteriores. En cuanto al proceso de estimación de movimiento, las novedades más importantes por su impacto en los niveles de compresión alcanzados, aparte de las ya mencionadas, se localizan en los siguientes anexos del estándar H.263:

- **Modo de vectores de movimiento no restringidos** (*Unrestricted Motion Vector Mode*, Anexo D). Este modo opcional permite que los vectores de movimiento apunten fuera de un fotograma para macrobloques situados en los *bordes* del fotograma actual. Este hecho ocurrirá cuando parte del área de búsqueda quede fuera del fotograma previamente codificado tomado como referencia, interpolando el estimador de movimiento los píxeles necesarios hasta completar el área de búsqueda.

- **Modo de predicción avanzada** (*Advanced Prediction Mode*, Anexo F). Este modo opcional considera la posibilidad de que en un macrobloque hayan diferentes objetos moviéndose en diferentes sentidos y velocidades. Debido a esta razón, se permite la posibilidad de que existan cuatro vectores de movimiento para un mismo macrobloque (un vector de movimiento por cada bloque de 8×8 píxeles de luminancia). De esta manera, el codificador calcula, además de un vector de movimiento para cada macrobloque según el procedimiento descrito anteriormente, cuatro vectores más para cada uno de los bloques de 8×8 píxeles con el consiguiente incremento del coste computacional del proceso de estimación de movimiento. En

contraposición al aumento del coste computacional, el uso conjunto de los dos modos opcionales descritos supone un aumento de entre 0.2 y 1 dB en la calidad de la secuencia decodificada para una misma tasa de transmisión [CEG+98].

Es de destacar que estas mismas técnicas han sido incluidas de manera idéntica en el estándar MPEG-4 con beneficios similares en términos de compresión.

2.1.2.3.2 Mejoras introducidas por el estándar H.264/AVC

H.264/AVC representa el estándar de codificación de vídeo con mejores prestaciones de compresión, pero también, y como consecuencia directa de sus elevadas prestaciones, con mayor coste computacional asociado. Así, el complejo conjunto de herramientas de codificación introducido por el estándar H.264/AVC determina que un codificador basado en este estándar sea entre 10 y 100 veces más complejo que un codificador equivalente basado en el estándar MPEG-4 [OBL+04], [SDL+04].

Dentro de este proceso de codificación, la estimación de movimiento recomendada por el estándar puede suponer entre un 80% y un 95% del esfuerzo computacional total [ZHY+03], [CHC04a], [HHC+06] debido a las exigentes singularidades introducidas por H.264/AVC en cuanto al proceso de cálculo de vectores de movimiento se refiere. Estas características son las siguientes:

Estimación de movimiento para bloques de topología variable (*Variable Block Size Motion Estimation – VBSME*). Con el objetivo de reproducir de forma fidedigna el movimiento de una determinada escena, el estándar proporciona una mayor flexibilidad a la hora de la selección de los tamaños para realizar la estimación y posterior compensación de movimiento.

De esta manera, el estándar permite dividir un macrobloque de las siete maneras diferentes (en la terminología del estándar, siete *modos de estimación de movimiento)* que se muestran en la Figura 2.9, realizándose una estimación de movimiento multimodo.

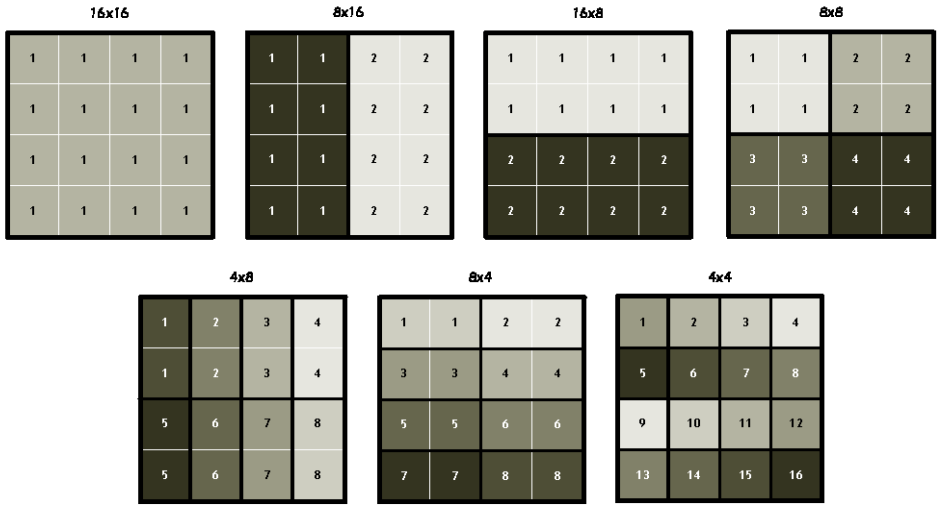


Figura 2.9: Modos de estimación de movimiento definidos por el estándar H.264/AVC.

De esta manera, el estándar H.264/AVC divide un macrobloque en particiones o modos superiores (16×16 hasta 8×8) y sub-particiones o modos inferiores (4×8 hasta 4×4), pudiendo llegar a tener entre uno y dieciséis vectores de movimiento en su posterior compensación de movimiento. Este hecho implica que, por cada macrobloque, se puedan tener que calcular 41 vectores de movimiento diferentes, frente al único vector calculado por H.261 y MPEG-1 o los 5 calculados en los codificadores H.263 y MPEG-4.

Compensación de movimiento a cuarto de píxel. El estándar H.264/AVC apuesta por incrementar la precisión de los vectores de movimiento a coordenadas de cuarto de píxel, hecho que había considerado el estándar MPEG-4 de manera opcional en codificadores de muy altas prestaciones. A diferencia de estándares anteriores, H.264/AVC emplea un filtro FIR (*Finite Impulse Response*) de seis coeficientes para calcular las muestras de medio píxel que se muestran en la Figura 2.10 a partir de las muestras de píxel.

La muestra de medio píxel *b* señalada en la Figura 2.10, al ser un medio píxel *horizontal* (tiene muestras de píxel a izquierda y derecha), se calcula de la siguiente manera:

$$b = (E - 5F + 20G + 20H - 5I + J)/32 \qquad \text{(ec. 2.4)}$$

De manera análoga, la muestra *h*, al ser un medio píxel *vertical* (tiene muestras de píxel arriba y abajo), es interpolada filtrando las muestras *A*, *C*, *G*, *M*, *R* y *T*. Una vez que todas las muestras de medio píxel horizontales y verticales han sido calculadas, las muestras de medio píxel diagonales se calculan filtrando seis muestras horizontales o seis muestras verticales previamente calculadas.

Las muestras de cuarto de píxel que se muestran en la Figura 2.10 como *a*, *c*, *d*, *n*, *f*, *i*, *k* y *q* se obtienen mediante interpolación bilineal, de acuerdo a:

$$a = (G + b)/2 \qquad \text{(ec. 2.5)}$$

Las otras muestras de cuarto de píxel etiquetadas como *e*, *g*, *p* y *r* también se calculan mediante interpolación bilineal, pero sólo usando muestras de medio píxel:

$$e = (b + h)/2 \qquad \text{(ec. 2.6)}$$

En este punto, es importante señalar que en el estándar H.264/AVC la estimación de movimiento a medio y cuarto de píxel se realiza mediante sucesivos refinamientos a partir de la posición indicada por los vectores de precisión entera. Sin embargo, este proceso debe

realizarse para los 41 vectores de movimiento calculados por macrobloque, aumentando drásticamente el esfuerzo computacional a realizar con respecto a estándares anteriores.

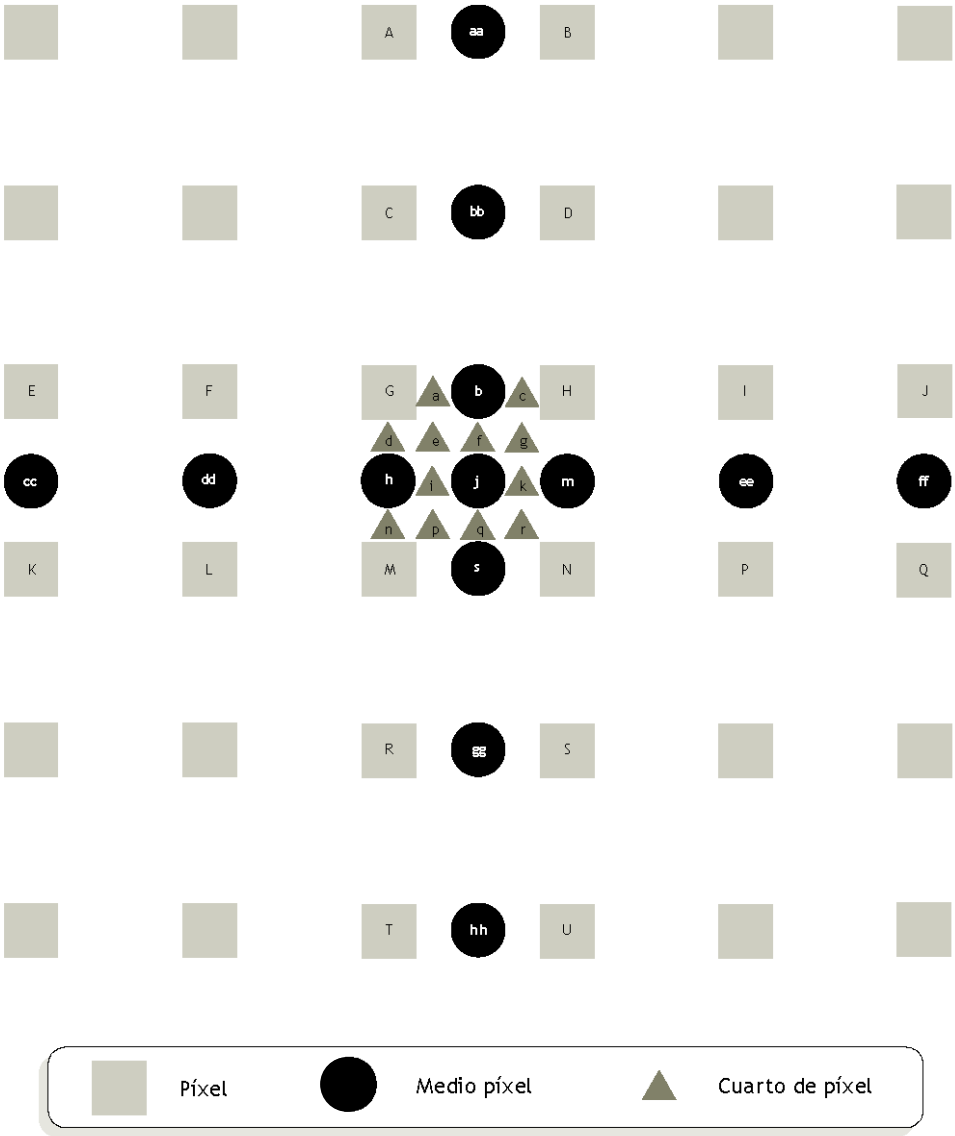


Figura 2.10: Cálculo de las muestras de medio y cuarto de píxel en el estándar H.264/AVC.

Estimación de movimiento multifotograma. En este estándar se introduce la posibilidad de realizar la estimación de movimiento no sólo con respecto al fotograma anterior (t-1), sino también con respecto a otros fotogramas ya procesados (t-f, con $2 \leq f \leq 5$) y almacenados en la memoria de reconstrucción. Esta característica implica no sólo un aumento del coste

computacional en el sentido de que se tienen que calcular 41 vectores de movimiento con precisión de cuarto de píxel por fotograma considerado, sino también un considerable aumento en las necesidades de memoria de almacenamiento del codificador.

Por último cabe destacar que, independientemente del proceso de estimación de movimiento, el estándar H.264/AVC introduce otras herramientas en el proceso de codificación con el objetivo de aumentar sus prestaciones de compresión, siendo la predicción *intraframe* y el filtrado de reducción de *efecto bloque* las más destacables [Ric03]. Aunque sus requisitos de procesamiento son mucho menos exigentes que los propios de la etapa de estimación de movimiento, su aplicación en codificadores de vídeo bajo condiciones de tiempo real ha motivado durante estos últimos años el diseño de arquitecturas hardware capaces de acelerar dichos procesos [LTC+06], [AKL+06].

2.2 Técnicas de optimización lagrangiana en codificadores híbridos de vídeo

Las secuencias de vídeo a comprimir normalmente contienen macrobloques con muy diferentes texturas y grados de movimiento, incluso en un mismo fotograma. Este hecho determina que los codificadores híbridos de vídeo necesiten de un conjunto de opciones para codificar cada macrobloque, maximizando la tasa de compresión alcanzada de acuerdo a las características espaciales y temporales de dicho macrobloque. En este sentido, a medida que el proceso de estandarización ha evolucionado, se ha ido ampliando el número de opciones diferentes para codificar un macrobloque, dependiendo el nivel de compresión alcanzado del conjunto de decisiones tomadas por el codificador. Por lo tanto, a nivel funcional, resulta imprescindible optimizar las decisiones de codificación para cada macrobloque con el objetivo de maximizar las prestaciones de compresión de un codificador basado en un estándar

cualquiera. Para este fin, las técnicas de optimización lagrangiana resultan ser extraordinariamente eficaces [WSJ+03], [MGL05].

2.2.1 Técnicas de optimización basadas en multiplicadores de Lagrange

En este apartado se explicarán las bases sobre las que se asientan las técnicas de optimización de Lagrange, con el objetivo de afrontar posteriormente su aplicación al caso de codificadores híbridos de vídeo. Para ello, sólo se describirán aquellos aspectos de importancia en el marco de esta Tesis, pudiéndose encontrar un mayor nivel de profundidad en [Eve63], [SG88], [SK97, pp. 43-72].

Con el objetivo de realizar una formulación adecuada del problema a resolver, consideremos un conjunto de K muestras agrupadas en un vector S tal que $S = (S_1, \ldots, S_K)$. Cada muestra S_k puede ser codificada, en principio, utilizando cualquiera de las diferentes opciones de codificación $O_k = (O_{k1}, \ldots, O_{kNk})$. Supongamos que I_k, perteneciente al conjunto de opciones O_k, representa la opción elegida para codificar la muestra S_k. De manera general, y siguiendo la notación empleada, las opciones seleccionadas para codificar el conjunto de muestras S vendrán determinadas por cada uno de los elementos del vector I definido como $I = (I_1, \ldots, I_K)$. Llegados a este punto, el problema de encontrar la combinación apropiada de opciones de codificación que minimice la distorsión D de un conjunto de muestras utilizando un número de bits menor que una constante R_c puede ser formulado de la siguiente manera:

$$min_I \{D(S,I)\} \text{ tal que } R(S,I) \leq R_c, \qquad \text{(ec. 2.7)}$$

donde $D(S,I)$ y $R(S,I)$ representan, respectivamente, la distorsión total y el número de bits resultantes de codificar las K muestras del vector S, seleccionando para ello las opciones de codificación I.

Una solución elegante a este problema se obtiene mediante el desacoplamiento de la ecuación 2.7, lo que conduce a encontrar un conjunto de opciones de codificación I de manera que:

$$I = \min_I \{J(S,I|\lambda)\}$$
$$J(S,I|\lambda) = D(S,I) + \lambda \cdot R(S,I), \qquad \text{(ec. 2.8)}$$

donde λ es una constante positiva denominada **constante o multiplicador de Lagrange** y la función $J(S,I|\lambda)$ a minimizar recibe el nombre de **función de coste de Lagrange**. La solución I que minimiza la función de coste es óptima en el sentido de que, si la constante R_c se corresponde con el valor del multiplicador de Lagrange seleccionado, dicha solución garantiza mínima distorsión.

Asumiendo que la distorsión y el número de bits son cantidades aditivas, y suponiendo que el valor de dichas cantidades depende única y exclusivamente de las opciones seleccionadas para codificar una muestra S_k específica, la ecuación 2.8 se puede reformular de la siguiente manera:

$$I = \min_I \{J(S,I|\lambda)\} = \min_I \sum_{k=1}^{K} J(S_k,I|\lambda) = \min_{Ik} \sum_{k=1}^{K} J(S_k,I_k|\lambda) \qquad \text{(ec. 2.9)}$$

De este modo, el problema descrito a través de la ecuación 2.7 puede ser resuelto de manera sencilla, seleccionando las opciones de codificación para cada una de las muestras S_k de manera totalmente independiente.

2.2.2 Aplicación de las técnicas de optimización lagrangiana a codificadores híbridos de vídeo

La implementación de las técnicas de optimización anteriormente descritas en codificadores híbridos de vídeo no es una tarea trivial, debido fundamentalmente a la multitud de opciones existentes para codificar cada uno de los fotogramas de la secuencia y al impacto que tienen las decisiones que se tomen en el resto del proceso de codificación de la secuencia de vídeo. Estas decisiones versan sobre aspectos tan críticos en un codificador de vídeo como la división del fotograma en conjuntos de píxeles con diferentes topologías; la posibilidad de codificar cada uno de estos conjuntos haciendo referencia a otros en el mismo o en un fotograma diferente, o decidir cuál es el conjunto de píxeles óptimo para hacer referencia a los píxeles de cada fotograma, entre otros.

Como puede deducirse, el conjunto de posibilidades a explorar es amplísimo y por lo tanto, imposible de evaluar fielmente en tiempo real para una secuencia cualquiera en los sistemas de compresión de vídeo convencionales. Este hecho ha determinado que en los codificadores híbridos de vídeo basados en alguno de los estándares conocidos, el espacio de exploración se haya acotado con el objetivo de evaluar, mediante funciones de coste independientes, sólo aquellas opciones de codificación que mayor impacto presenten en las prestaciones de compresión. De esta manera, las opciones a evaluar resultan ser las relacionadas con la selección del **modo de codificación** y del **conjunto de vectores de movimiento** óptimo para cada macrobloque [WSJ+03], [KKA05].

En particular, para el proceso de estimación de movimiento la función de coste J_{motion} a minimizar para cada macrobloque S_k es la siguiente:

$$J_{motion}(S_k, mv_k/\lambda_{motion}) = SAD\ (S_k, mv_k) + \lambda_{motion} \cdot R_{motion}(S_k, mv_k), \qquad \text{(ec. 2.10)}$$

donde mv_k representa el conjunto de vectores de movimiento posibles para el macrobloque S_k, $R_{motion}(S_k, mv_k)$ es el número de bits necesarios para codificar dichos vectores de movimiento, y $SAD\ (S_k, mv_k)$ representa el SAD obtenido para los vectores de movimiento bajo análisis.

En los estándares de codificación de vídeo, la codificación de los vectores de movimiento se realiza de forma diferencial con el objetivo de minimizar el número de bits R_{motion} necesarios para su transmisión. En el caso particular de los estándares H.263 y H.264/AVC, se codifica la diferencia entre el vector de movimiento calculado y un vector de movimiento predicho. Dicha predicción se obtiene, salvo para los macrobloques situados en los extremos de un fotograma, a partir de los vectores de los macrobloques vecinos MB_{left}, MB_{up} y $MB_{upright}$ señalados en la Figura 2.11.

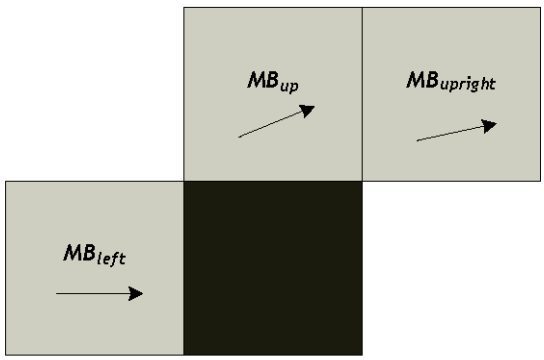


Figura 2.11: Macrobloques vecinos utilizados en la predicción del vector de movimiento.

Salvo para algunos casos particulares contemplados por ambos estándares (véase [H263, pp. 44-45] y [H264, pp. 129-130]), la predicción para el vector de movimiento del macrobloque que se está procesando se calcula como la mediana de los tres vectores vecinos anteriormente señalados. De esta manera, si los vectores de movimiento obtenidos para macrobloques vecinos son muy diferentes, el término R_{motion} aumentará drásticamente su valor, determinándose que el campo de vectores de movimiento obtenido es incoherente.

Con respecto a la determinación del valor de la constante λ_{motion}, se han realizado un gran número de experimentos con un conjunto de secuencias de entrada altamente heterogéneo que han sido publicados en [WG01] y [WSJ+03]. A partir de los resultados obtenidos, los autores proponen como valores óptimos del multiplicador de Lagrange λ_{motion} para su aplicación en codificadores basados en los estándares H.263 y H.264/AVC, los siguientes:

$$\lambda_{motion,\ H.263} = 0{,}92 \cdot Q_{p,H.263}$$
$$\lambda_{motion,\ H.264/AVC} = 0{,}92 \cdot (2^{(Qp,H.264/AVC - 12)/3})^{1/2} \qquad \text{(ec. 2.11)}$$

Por último, es importante destacar que para estándares que permiten estimación de movimiento con tamaño de bloque variable, este proceso de minimización es doble, en el sentido de que no sólo se persigue encontrar para cada modo de estimación de movimiento cuáles son los vectores óptimos, sino también cuál de todos los modos posibles es el mejor para el macrobloque bajo análisis. El conjunto de vectores de movimiento que presente una J_{motion} mínima será el considerado para la predicción del macrobloque bajo proceso, siempre y cuando el modo de codificación elegido por el codificador para el citado macrobloque sea *INTER*.

2.3 Estado del arte en estimación de movimiento

La estimación de movimiento constituye, sin lugar a dudas, la etapa dentro del proceso de codificación híbrida de vídeo sobre la que se ha realizado un mayor número de trabajos de investigación durante los últimos veinte años. Estos trabajos han centrado sus esfuerzos en dos aspectos:

- Obtener nuevos algoritmos de estimación de movimiento de bajo coste computacional que permitan obtener unas prestaciones de compresión similares a las ofrecidas por el algoritmo de búsqueda exhaustiva. De manera genérica, estos algoritmos reciben el nombre de *algoritmos rápidos de estimación de movimiento*.

- Desarrollar arquitecturas eficientes de estimación de movimiento para su posterior implementación en sistemas de compresión de vídeo en tiempo real.

En este apartado se resumen los trabajos más significativos en estas dos áreas, poniendo de manifiesto las limitaciones encontradas en cada caso y que serán posteriormente abordadas en esta Tesis.

2.3.1 Algoritmos rápidos de estimación de movimiento

El ingente número de publicaciones existente en este campo hace del todo inviable realizar, en el ámbito de esta Tesis, una recopilación exhaustiva de las propuestas realizadas durante las dos últimas décadas para la realización de la estimación de movimiento en codificadores híbridos de vídeo. En este sentido, se pueden encontrar recopilaciones de los algoritmos más significativos, entre otros, en los trabajos [FGW96], [CH97], [Kuh99, pp.17-60], [DH00],

[ACB03], [LO04], [HCT+06] y [Hui06], junto con una comparación de sus características en términos de calidad de imagen reconstruida y coste computacional asociado. De manera genérica, los algoritmos rápidos recogidos en estas publicaciones se pueden clasificar en:

- **Algoritmos basados en la reducción del número de posiciones a evaluar**. Estos algoritmos evalúan un subconjunto de candidatos dentro del área de búsqueda, seleccionado de manera dinámica las posiciones a evaluar a partir de los resultados parciales obtenidos. Para ello, se basan en la idea de que el SAD es una función monótona creciente, en el sentido de que su valor siempre aumenta a medida que la posición evaluada se aleja de la posición de mínimo SAD. Este hecho determina que estos algoritmos puedan quedar fácilmente *atrapados* en mínimos locales de la función SAD, proporcionando una pobre estimación de movimiento. Es de destacar que la mayor parte de los algoritmos rápidos de estimación de movimiento se encuentran dentro de este grupo.

- **Algoritmos basados en la evaluación de métricas simplificadas**. En este caso, se evalúan todas las posiciones del área de búsqueda haciendo uso de una métrica computacionalmente más sencilla que el SAD para determinar la posición ganadora, y por lo tanto, el vector de movimiento. Para ello, gran parte de los algoritmos dentro de este grupo proponen evaluar el SAD de cada posición sólo para un subconjunto de píxeles dentro del macrobloque [LZ93], [CS96], [WWK00] o bien, reducir el número de bits asociado a cada píxel mediante truncamiento [LCL+02].

- **Algoritmos jerárquicos**. Estos algoritmos se basan en realizar estimaciones de movimiento a diferentes niveles de resolución de imagen, desplazándose desde el nivel más bajo al más alto. De esta manera, se obtiene un primer vector de movimiento a partir del nivel más bajo, en el cual los bloques de píxeles son de dimensiones reducidas y por lo tanto, se puede establecer un área de búsqueda amplia. A partir de

esta primera estimación, se procede el refinamiento en los sucesivos niveles en los que, al ser los bloques de píxeles mayores, el área de búsqueda se va reduciendo proporcionalmente al tamaño del bloque. Normalmente, estos algoritmos suelen considerar entre dos y tres niveles de jerarquía [NKP+95], [LLS+01], [LL04].

- **Algoritmos predictivos**. Los algoritmos predictivos se basan en la coherencia del campo de vectores de movimiento de una secuencia de vídeo y por lo tanto, en la idea de que los vectores de movimiento de los macrobloques situados en una vecindad espacio-temporal deben ser similares. En este sentido, este tipo de algoritmos evalúan sólo aquellas posiciones apuntadas por un conjunto de vectores cuidadosamente seleccionado de entre los calculados en el fotograma anterior y en los macrobloques procesados del fotograma actual [HBH+93], [CFP02], [TAL02]. Los algoritmos predictivos han recibido una especial atención en estos últimos años debido a que, para diferentes estándares de compresión de vídeo, presentan mejores prestaciones que la mayor parte de los algoritmos rápidos clasificados en otros grupos de los anteriormente mencionados [CFP02], [HCT+06]. En particular, el algoritmo predictivo publicado en [CFP02] denominado algoritmo PBM (*Predictive Block Matching*) ha demostrado ser altamente eficiente en codificadores de vídeo basados en el estándar H.263, y por lo tanto, constituye una importante referencia en el marco de esta Tesis Doctoral.

  El algoritmo PBM realiza la estimación de movimiento en dos etapas. En la primera de ellas se evalúan las posiciones apuntadas por los cuatro vectores de movimiento correspondientes a los macrobloques señalados en la Figura 2.12 como $MB_{centre,T-1}$, $MB_{right,T-1}$, $MB_{left,T}$ y $MB_{up,T}$, además de la posición apuntada por el vector (0,0).

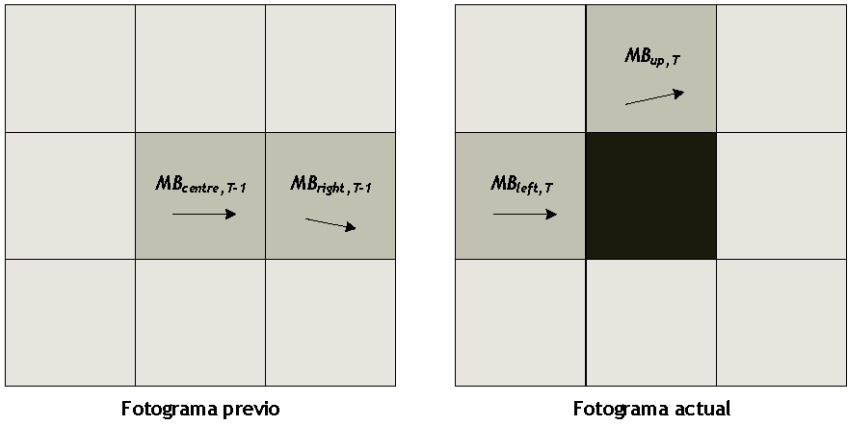


Figura 2.12: Vecindad espacio-temporal de vectores de movimiento utilizados por el algoritmo PBM en la primera etapa de estimación de movimiento.

A partir de la posición apuntada por el mejor de los cinco vectores evaluados, se procede a una segunda fase de refinamiento en la que se evalúan las ocho posiciones alrededor del vector obtenido en la primera etapa que se muestran en la Figura 2.13, obteniéndose finalmente un vector de movimiento con precisión de medio píxel.

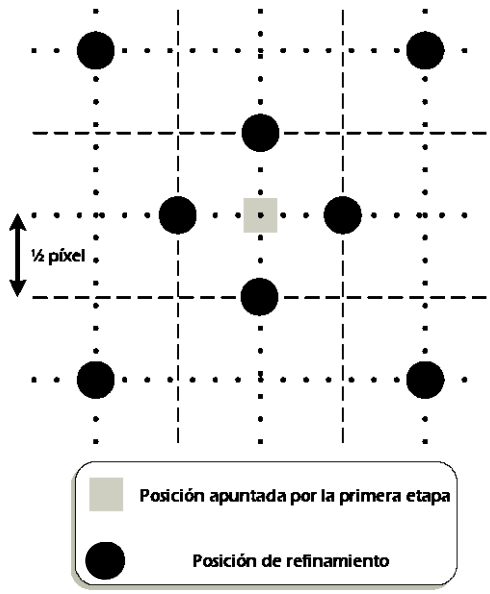


Figura 2.13: Posiciones de refinamiento evaluadas en la segunda etapa del algoritmo PBM.

De forma general, la principal desventaja de los algoritmos rápidos de estimación de movimiento existentes radica en que, directa o indirectamente, realizan un conjunto de suposiciones acerca de la secuencia de vídeo a codificar con el objetivo de reducir el coste computacional del proceso de estimación de movimiento. No obstante, estas hipótesis de partida conllevan que las prestaciones de compresión alcanzadas por los algoritmos rápidos de estimación de movimiento, indistintamente del grupo al que pertenezcan, sean extremadamente dependientes de la naturaleza de la secuencia de vídeo a codificar. En este sentido, los algoritmos rápidos de estimación de movimiento pueden presentar una seria degradación de sus prestaciones en aquellas secuencias de vídeo con características diferentes de las supuestas.

Este hecho ha motivado la búsqueda de nuevas soluciones para la realización del proceso de estimación de movimiento con un coste computacional reducido, pero manteniendo a la misma vez tasas de compresión altas para cualquier tipo de secuencia de vídeo. Esta es la finalidad de los denominados ***algoritmos de estimación de movimiento adaptativos***, en los cuales el coste computacional del proceso de estimación de movimiento varía de acuerdo a las características de la secuencia de vídeo a comprimir. Sin embargo, a pesar de que se han publicado numerosos trabajos en este campo, los algoritmos propuestos hasta la fecha presentan, al menos, alguna de las siguientes limitaciones:

- **El mecanismo de adaptación es exclusivo para tamaño de macrobloque**. A pesar de que, desde la aparición de los estándares H.263 y MPEG-4, y muy especialmente a partir del advenimiento del estándar H.264/AVC, se realiza una apuesta clara por incorporar diferentes tamaños de bloque en el proceso de estimación de movimiento, una gran parte de los algoritmos de estimación de movimiento adaptativos realizan el cálculo de un solo vector de movimiento por macrobloque [CP96], [FLM+98], [MKC00], [ANS01], [BJV01], [LC01], [TLW01], [MM02], [CTY+02], [Hos03],

[MKD03], [YL03], [CHW04], [JLB+04], [CD05], [HCW05], [LJI+05], [NM05], [AZL+06]. Este hecho determina que, en principio, resulte obligatorio repetir el procedimiento fijado por estos algoritmos 5 y 41 veces para los estándares H.263/MPEG-4 y H.264/AVC respectivamente, por lo que la reducción en el coste computacional total es insignificante. Es de destacar que, en el caso de los algoritmos adaptativos basados en la modificación de parámetros asociados al proceso de búsqueda exhaustiva, los cuales se definirán a continuación, este bucle repetitivo no es necesario, al poderse reaprovechar los resultados parciales obtenidos con los bloques de píxeles de menor tamaño [LTV+05a].

- **Ineficiente adaptación mediante ajuste dinámico del tamaño del área de búsqueda del algoritmo de búsqueda exhaustiva**. Un gran número de los algoritmos adaptativos publicados recientemente se basan en la sencilla estrategia de reducir el área de búsqueda en aquellos macrobloques en los que, de algún modo, se determine que sólo es necesario evaluar un pequeño número de posiciones alrededor del vector cero [FLM+95], [OL98], [PB98], [MS99], [OL00], [SM02], [SF04], [CCC05], [ZG05a]. Estos algoritmos son muy eficientes en el procesamiento, tanto de macrobloques sin movimiento (áreas de búsqueda muy reducidas) como de macrobloques con una gran cantidad de movimiento (áreas de búsqueda amplias). Sin embargo, en aquellos macrobloques con un grado de movimiento *medio*, el algoritmo de búsqueda exhaustiva requiere, por término medio, de un área de búsqueda de 24×24 píxeles para obtener unos resultados aceptables [SF04], lo cual supone la evaluación de 81 posiciones para obtener un vector de precisión entera. Para la inmensa mayoría de estos macrobloques, muchos de los algoritmos rápidos de estimación de movimiento son capaces de obtener un vector de movimiento de igual calidad realizando un número de operaciones considerablemente menor. Además, algunos de estos algoritmos proponen reajustar el área de búsqueda por fotograma en lugar de por macrobloque, lo que conlleva una pérdida apreciable en la calidad de la imagen

reconstruida [PB98], [MS99] que sería todavía de mayor consideración en secuencias con cambios de contexto.

- **El coste hardware asociado es elevado**. Gran parte de las propuestas adaptativas, en su empeño de alcanzar altas prestaciones de compresión con un bajo coste computacional, no prestan atención a los requisitos impuestos por los algoritmos diseñados con vistas a su implementación hardware para aplicaciones en tiempo real. Así, en varios algoritmos de estimación de movimiento adaptativos, se constata la existencia de dos aspectos que dificultan su realización en hardware:

  - Flujo de datos altamente irregular [MKC00], [Hos03], [YL03], [HCW05], [NM05], [ZG05b], [AZL+06], [LAL+06], determinado por el comportamiento altamente dinámico del algoritmo, estableciéndose un conjunto amplísimo de posibilidades de estimación de movimiento que se traduce en un elevado número de estados posibles por macrobloque.
  - Traslación dinámica del origen del área de búsqueda [FLM+95], [FLM+98], [OL98], [OL00], [YL03], [LJI+05], [NM05], [ZG05b], [LAL+06]. De esta manera, se imposibilita la reutilización de los píxeles del área de búsqueda entre dos macrobloques vecinos, con lo que se incrementa el número de accesos a la memoria de reconstrucción. Alternativamente, se puede aumentar el tamaño de la memoria interna del estimador de movimiento con el objetivo de poder reutilizar los mencionados píxeles, siendo absolutamente necesario en este caso acotar los movimientos del centro del área de búsqueda [CCC05].

- **El campo de vectores de movimiento obtenido presenta incoherencias**. Con el objetivo de adaptarse a secuencias con gran cantidad de movimiento, algunos algoritmos

adaptativos consideran la posibilidad de evaluar posiciones muy alejadas del vector (0,0), ya sea mediante una considerable traslación del centro del área de búsqueda [OL98], [OL00] o mediante la inspección de patrones de búsqueda alejados de dicho centro [HCW05]. Sin embargo, estos algoritmos no tienen en cuenta el incremento que puede experimentar el número de bits R_{motion} necesario para transmitir el vector calculado, a pesar de que la codificación diferencial de vectores de movimiento puede suponer hasta un 30% de la tasa de transmisión total para una secuencia de vídeo codificada por debajo de los 128 Kbps [NM05]. Además, los autores de estas publicaciones sólo establecen en sus resultados comparaciones en términos de coste computacional y SAD obtenido, sin realizar prueba alguna en ningún codificador de vídeo estándar que permita determinar si, efectivamente, se llega a producir una degradación de las prestaciones de compresión a causa del incremento experimentado por la función de Lagrange J_{motion}.

- **La idoneidad de la estrategia de adaptación para secuencias de vídeo con baja tasa de muestreo temporal no está garantizada**. Una situación típica en los codificadores de vídeo en tiempo real consiste es sub-muestrear la secuencia de vídeo a codificar con el objetivo de adaptarse fácilmente a una variación de las condiciones de compresión (cambios en el canal de transmisión, reducción del tiempo disponible para codificar la secuencia por requisitos de la aplicación, ...) o simplemente, con el fin de transmitir una menor cantidad de información. Sin embargo, gran parte de las decisiones tomadas en los algoritmos adaptativos expuestos, están basadas en los resultados obtenidos en los macrobloques del fotograma anterior al que se está procesando. Esta circunstancia conduce a pensar que los resultados para secuencias de baja tasa de muestreo temporal no serán tan satisfactorios como los publicados por los autores para una gran cantidad de secuencias de vídeo muestreadas a razón de 30 fotogramas por segundo. De hecho, prácticamente la totalidad de las publicaciones sólo ofrecen resultados para esta tasa de muestreo, obviando las posibles consecuencias derivadas

de su reducción. En este sentido, únicamente los trabajos publicados por Saponara y Fanucci en [SF04] presentan resultados para secuencias muestreadas a razón de 30 y 10 fotogramas por segundo, obteniéndose en este último caso unas notables prestaciones de compresión mediante el aumento del área de búsqueda del algoritmo de búsqueda exhaustiva.

2.3.2 Arquitecturas de estimación de movimiento

Al igual que ocurre con los algoritmos rápidos de estimación de movimiento, se han publicado numerosísimos trabajos relacionados con el desarrollo de arquitecturas de estimación de movimiento, especialmente para el caso de búsqueda exhaustiva. Diversas recopilaciones de los trabajos más significativos en esta área se pueden encontrar en [San98], [Kuh99, pp.119-174], [ESB00], [ESX+01], [ENB+02], [CCH+06a], [HCT+06].

Las arquitecturas presentadas hasta la fecha están compuestas por una red sistólica de elementos básicos, denominados elementos de proceso (*Processing Element – PE*), encargados de calcular la diferencia en valor absoluto entre un píxel del macrobloque de referencia y otro píxel de área de búsqueda. Dependiendo de la topología utilizada para unir dichos elementos de proceso, las arquitecturas se dividen en *arquitecturas unidimensionales* y *arquitecturas bidimensionales*. Las primeras son arquitecturas de bajo consumo de potencia, área reducida y sencillo control, típicamente utilizadas con secuencias de vídeo de mediana y baja resolución. Sin embargo, las arquitecturas bidimensionales son capaces de procesar un mayor número de posiciones por segundo, a costa de un considerable aumento en el número de elementos de proceso.

Dentro de cada uno de los dos tipos de arquitecturas anteriormente mencionados, se puede realizar una segunda división dependiendo de la asignación de las posiciones a evaluar entre los diferentes elementos de proceso. En este sentido, DeVos y Stegherr distinguen en [VS89] dos tipos de arquitecturas dependiendo del grado de paralelismo alcanzado en los elementos de proceso:

- Tipo 1: Arquitectura que dispone de tantos elementos de proceso como píxeles tiene el macrobloque de referencia. En este caso, todos los elementos de proceso contribuyen a calcular en paralelo el SAD de cada una de las posiciones de búsqueda siendo necesarios tantos ciclos como posiciones a evaluar para completar el proceso de estimación de movimiento.

- Tipo 2: Arquitectura que dispone de tantos elementos de proceso como posiciones a evaluar dentro del área de búsqueda. En este caso, cada elemento de proceso evalúa el SAD de una determinada posición, siendo necesarios tantos ciclos como píxeles tiene al macrobloque de referencia para completar el proceso de estimación de movimiento.

Esta misma clasificación es propuesta por Chen et al. en [CCH+06a] bajo la denominación de arquitecturas *intra-level* (tipo 1) e *inter-level* (tipo 2). No obstante, en ninguno de los dos trabajos se contempla la posibilidad de una arquitectura híbrida en la que, por ejemplo, un elemento de proceso evalúe una porción del SAD de varias posiciones de búsqueda. Por esta razón, en esta Tesis se propone una nueva clasificación que contiene a las anteriores:

- **Arquitectura no agrupada**: Arquitectura en la que cada elemento de proceso evalúa una o varias posiciones del área de búsqueda de manera individualizada, independientemente del número de elementos de proceso del que conste la arquitectura y del número de posiciones a evaluar.

- **Arquitectura agrupada**: Arquitectura en la que los elementos de proceso colaboran, de un modo cualquiera, en la evaluación de varias posiciones del área de búsqueda, independientemente del número de elementos de proceso del que conste la arquitectura y del número de posiciones a evaluar.

De esta manera, las arquitecturas tipo 1 o *intra-level* y las tipo 2 o *inter-level* representan casos particulares dentro de las arquitecturas agrupadas y no agrupadas, respectivamente. Así, se distinguen cuatro tipos de arquitecturas: unidimensionales agrupadas, unidimensionales no agrupadas, bidimensionales agrupadas y bidimensionales no agrupadas. Esta nueva nomenclatura permite clasificar de manera sencilla, no sólo las arquitecturas propuestas en esta Tesis, sino también las arquitecturas previas tanto de precisión entera como de precisión sub-píxel.

2.3.2.1 Arquitecturas de estimación de movimiento con precisión entera

Con independencia de sus características, y debido al creciente interés en el nuevo estándar H.264/AVC, se han propuesto recientemente varias arquitecturas capaces de realizar el proceso de estimación de movimiento con precisión entera para los siete modos definidos por el estándar. La mayor parte de estas propuestas se han centrado en la implementación del algoritmo de búsqueda exhaustiva, ya sea mediante el uso de arquitecturas unidimensionales no agrupadas [YM03], unidimensionales agrupadas [OLH05], [SLG+06] o bidimensionales agrupadas [HWH+03], [WYG+04], [DGH+05], [KHC05], [WG05a], [WG05b], [ZG05a], [CCH+06a], [CCH+06b]. Sin embargo, estas propuestas presentan las siguientes limitaciones:

1. **Rigidez computacional**. Salvo en el caso de la arquitectura propuesta por Zhang y Gao en [ZG05a], en la que se modifica el tamaño del área de búsqueda con las limitaciones que esto conlleva, ninguna de las mencionadas propuestas admite grado de adaptación alguno de los recursos al contenido de las secuencias de vídeo.

2. **Ausencia de estrategias de eliminación temprana de candidatos multimodo**. La técnica de eliminación temprana de candidatos consiste en dejar de calcular el SAD asociado a una posición del área de búsqueda cuando éste ha superado el valor del SAD mínimo provisional. Durante estos últimos años, se ha prestado especial atención en incrementar la eficacia de este proceso algorítmicamente [CP03], [HSC05], [MQ05], [ZQ05], así como en la aplicación de dicha técnica en arquitecturas sistólicas de estimación de movimiento para tamaño de macrobloque [DY98], [SR99], [Sou99], [ESB00], [ESX+01], [HCH+04], [LTC04]. En particular, estos trabajos demuestran que los elementos de proceso de la arquitectura pueden ser inhabilitados durante un número de ciclos que varía entre el 20% y el 65% del total requerido para completar el proceso de estimación de movimiento, reduciéndose la frecuencia de conmutación de la lógica combinacional de cada elemento de proceso. No obstante, y a pesar de estos prometedores resultados, ninguna de las arquitecturas con capacidad para calcular los 41 vectores establecidos por el estándar H.264/AVC ha intentando incorporar estos mecanismos de eliminación. Este hecho determina la existencia de un conjunto de nuevos aspectos a abordar:

   - Establecer los mecanismos necesarios para inhabilitar los elementos de proceso en arquitecturas de estimación de movimiento multimodo. Para ello, habrá que tener en cuenta dos aspectos fundamentales. El primero de ellos es que la decisión de inhabilitación ha de ser rápida al poder realizarse, en principio, para bloques de 4×4 píxeles. En segundo lugar, se debe tener en

cuenta el impacto de inhabilitar un elemento de proceso que está calculando el SAD asociado a una posición de 4×4 píxeles, sobre los modos superiores.

- Determinar si los beneficios obtenidos en términos de ciclos inactivos son similares para el caso de arquitecturas con estimación de movimiento multimodo, particularmente para arquitecturas compatibles con el estándar H.264/AVC.
- Establecer, en términos de área ocupada, el coste hardware resultante de incorporar mecanismos de eliminación temprana de candidatos en arquitecturas de estimación de movimiento multimodo. En este punto, es importante señalar que en ninguna de las arquitecturas referenciadas con eliminación temprana, los autores ofrecen datos objetivos acerca de este aspecto.

3. **Inexistencia de criterios en la agrupación de elementos de proceso**. La inmensa mayoría de las arquitecturas bidimensionales dedicadas el cálculo de vectores de movimiento mediante búsqueda exhaustiva para el estándar H.264/AVC están compuestas por un matriz de 16×16 elementos de proceso y un árbol de sumas. Estos elementos de proceso funcionan como un solo grupo, de manera que la arquitectura es *totalmente agrupada*. Las ventajas de esta agrupación de elementos de proceso son evidentes, al facilitarse enormemente la composición de los vectores de los modos superiores a partir de la obtención en paralelo de los SADs de cada uno de los bloques de 4×4 píxeles presentes en un macrobloque, sin necesidad de incorporar registros intermedios [HWH+03], [DGH+05], [KHC05], [WG05a]. En el caso de las arquitecturas unidimensionales, típicamente constituidas por un número de elementos de proceso menor, no existe un criterio claro a la hora de establecer grupos entre los elementos de proceso. Por lo tanto, resulta necesario estudiar las diferentes

posibilidades de agrupación de elementos de proceso, así como su impacto en las prestaciones de la arquitectura.

2.3.2.2 Arquitecturas de estimación de movimiento con precisión sub-píxel

A pesar de la importancia que posee el refinamiento de vectores a coordenadas sub-píxel en el estándar H.264/AVC, el número de publicaciones en esta área es muy reducido. De hecho, los trabajos más significativos se han centrado en los siguientes aspectos:

- Arquitecturas de refinamiento de vectores a coordenadas de medio píxel para estándares anteriores a H.264/AVC [GSJ+02], [SB03], [CCC04], [WYZ+04], [DRS05]. Obviamente, los requisitos de procesamiento en dichas arquitecturas son mucho menos exigentes, debido a que el número de modos de estimación de movimiento es menor (en estas arquitecturas sólo se contemplan tamaños de bloque de 16×16 y 8×8 píxeles) y sólo hay que realizar el refinamiento hasta coordenadas de medio píxel. Además, el método de interpolación bilineal implementado para generar las muestras de medio píxel es más sencillo que el filtrado propuesto por H.264/AVC. A partir del análisis de estas arquitecturas se observa que todas ellas son arquitecturas no agrupadas, en las que cada elemento de proceso evalúa una posición de refinamiento.

- Optimización del proceso de interpolación de muestras de medio y cuarto de píxel para decodificadores basados en el estándar H.264/AVC [LWW05], [SLG+05], [SN05], [HKL+06], sin necesidad de realizar el proceso de refinamiento propiamente dicho.

Para el caso particular del estándar H.264/AVC, sólo se han encontrado dos arquitecturas capaces de realizar dicho proceso de refinamiento según las directrices de dicho estándar. Dichos trabajos han sido publicados en [CHC04b] y [RB05], siendo relevante únicamente el primero de ellos. En este trabajo, realizado por Chen, Huang y Chen y publicado en [CHC04b], se presenta una arquitectura de refinamiento sub-píxel de vectores de movimiento como parte de un chip codificador en el que participan los autores [CCH+06b]. Se trata de una arquitectura formada por 36 elementos de proceso en la que se ha incluido la lógica necesaria para seleccionar, según el criterio lagrangiano recomendado por el estándar H.264/AVC, el mejor modo de estimación de movimiento al terminar el refinamiento de cuarto de píxel para cada uno de los macrobloques de la secuencia de vídeo bajo análisis. Los autores presentan, junto con la arquitectura, un algoritmo denominado AMPD (*Advanced Mode Pre-Decision*) el cual, sobre la base de los resultados obtenidos durante la búsqueda con precisión entera, elige sólo tres de los siete posibles modos de estimación de movimiento para los cuales se realizará el refinamiento sub-píxel a costa de una ligera pérdida de capacidad de compresión. De esta manera, y según los datos aportados por los autores, la arquitectura es capaz de procesar 49K macrobloques por segundo a una frecuencia de trabajo de 100 MHz utilizando para ello 79,4K puertas NAND2 equivalentes.

La segunda propuesta consiste en una arquitectura de refinamiento compuesta por 128 elementos de proceso (8 grupos de 16 elementos de proceso cada uno) que ha sido sintetizada sobre una FPGA modelo Virtex2 de Xilinx ocupando 14k slices y 28,5k LUTs, lo cual, a juicio de los autores, se corresponden con 225k puertas NAND2 equivalentes. Esta arquitectura, de manera contraria a la recomendación establecida por el estándar, realiza el proceso de búsqueda inspeccionado todas las posiciones contenidas en el área de búsqueda, independientemente de si éstas son de precisión entera o sub-píxel. Debido a este esquema de funcionamiento, esta arquitectura evalúa un número de posiciones de búsqueda elevadísimo y, como consecuencia directa, necesita interpolar un número de muestras sub-píxel que crece

de manera exponencial con el aumento del número de posiciones consideradas. Por último, es de destacar que los autores suponen que toda el área de búsqueda está pre-interpolada y ubicada en un conjunto de memorias consideradas para tal fin (no proponen ningún circuito de interpolación) y que además, no justifican los beneficios de buscar en todas las posibles posiciones en lugar de seguir el clásico esquema de búsqueda de precisión entera y posterior refinamiento sub-píxel.

Por último, es de resaltar el hecho de que en todos los trabajos mencionados, ya sea en arquitecturas de precisión entera o de precisión sub-píxel, el número de modos de estimación de movimiento evaluados es constante. En este sentido, las arquitecturas mencionadas no permiten incorporar los resultados aportados por numerosísimos trabajos que permiten reducir el número de modos de estimación de movimiento sin pérdidas apreciables en la calidad de la imagen reconstruida, entre los que destacan, entre otros, los publicados en [AKM+04], [JC04] y [WPL+05]. Por lo tanto, y puesto que estas estrategias han demostrado su validez en términos de compresión, resulta de interés investigar sus posibles beneficios en términos puramente arquitecturales.

2.4 Conclusiones

En este capítulo se han establecido las bases de la codificación híbrida de vídeo haciendo especial hincapié, por su relevancia en esta Tesis, en el proceso de estimación de movimiento. En particular, se ha descrito la evolución de la etapa de cálculo de vectores de movimiento a lo largo del procedimiento de estandarización, revisándose asimismo las aportaciones más significativas al proceso de estimación de movimiento, tanto desde el punto de vista algorítmico como arquitectural.

En primer lugar, se ha revisado la estructura interna de un codificador híbrido de vídeo genérico, pues constituye el núcleo básico de computación sobre el que se construyen los diferentes estándares de compresión de vídeo. En este análisis ha quedado de manifiesto la presencia de tres unidades funcionales claramente diferenciadas dentro del codificador: unidad de preproceso, unidad de reducción de redundancias espaciales y unidad de reducción de redundancias temporales, perteneciendo a esta última la etapa de estimación de movimiento. Igualmente, se han estudiado los requisitos impuestos por cada estándar a los vectores de movimiento calculados por macrobloque y en consecuencia, a los estimadores de movimiento. A partir de este estudio se concluye que el estándar H.264/AVC es el más exigente en cuanto al proceso de movimiento se refiere, al considerar un número variable de vectores de movimiento (entre 1 y 16) con precisión de cuarto de píxel por macrobloque que además, pueden apuntar a cualquiera de los cinco fotogramas previamente procesados por el codificador.

Esta creciente diversificación de las alternativas relacionadas con el proceso de estimación de movimiento en los estándares de compresión de vídeo, ha creado la necesidad de disponer de un conjunto de mecanismos que permitan decidir, para cada macrobloque, el mejor conjunto de opciones para maximizar la tasa de compresión alcanzada. Las técnicas de optimización lagrangiana introducidas en este capítulo permiten seleccionar el conjunto de vectores óptimo para cada macrobloque mediante la minimización de la función de coste J_{motion}. El estudio de dicha función de coste será de especial relevancia en la definición del algoritmo adaptativo de estimación de movimiento que se desarrollará en el siguiente capítulo de esta Tesis.

Por último, se han recopilado en este capítulo las aportaciones más significativas al proceso de estimación de movimiento. En dicha recopilación se ha prestado especial atención a los trabajos relacionados con estimación de movimiento adaptativa bajo los requisitos impuestos por el estándar H.264/AVC, tanto desde el punto de vista algorítmico como arquitectural.

Como resultado, se ha puesto de manifiesto para ambos casos un conjunto de problemas que serán abordados en los siguientes capítulos y cuya solución representa, en gran medida, las aportaciones de esta Tesis.

CAPÍTULO 3

Propuestas algorítmicas de estimación de movimiento adaptativa

La estimación de movimiento representa, sin lugar a dudas, la etapa crítica dentro de un codificador híbrido de vídeo debido a su elevado coste computacional e influencia en la tasa de compresión. Este hecho ha dado lugar al desarrollo de numerosos algoritmos rápidos de estimación de movimiento por parte de la comunidad científica, los cuales presentan una reducción del coste computacional con respecto al algoritmo de búsqueda exhaustiva. Sin embargo, las prestaciones de compresión ofrecidas por dichos algoritmos rápidos dependen extraordinariamente de las características espaciales y temporales de la secuencia de vídeo a comprimir.

Con el objetivo de eliminar esta dependencia, se propone en este capítulo un nuevo algoritmo adaptativo de estimación de movimiento. Este algoritmo permite, mediante adaptación a las características de la secuencia y a las exigencias del estándar en uso, obtener unos excelentes niveles de compresión para todo tipo de secuencias y tasas de transmisión, asegurando un coste computacional inferior al del algoritmo de búsqueda exhaustiva. Asimismo, el algoritmo propuesto supera las limitaciones encontradas en los trabajos previos que conforman el estado del arte en estimación de movimiento adaptativa.

3.1 Análisis del proceso de estimación de movimiento

En el capítulo anterior ha quedado de manifiesto que en los sistemas de codificación híbrida de vídeo con prestaciones de tiempo real basados en algún tipo de estándar de compresión, la etapa de estimación de movimiento se realiza mediante ajuste de bloques (ver apartado 2.1.1.3). El número de vectores de movimiento, y en consecuencia los diferentes tamaños de bloque a considerar dentro de un macrobloque, así como la precisión de dichos vectores, dependerán del estándar que se utilice. En cualquier caso, cada codificador deberá decidir cuántos vectores de movimiento deberá enviar por macrobloque y qué algoritmo de ajuste de bloques utilizará para su cálculo. Para ello, las ecuaciones de optimización en términos de multiplicadores de Lagrange introducidas en el capítulo anterior resultan ser una solución eficiente.

En esta sección se presenta un entorno de simulación que permite analizar de manera detallada el proceso de ajuste de bloques en términos de la ecuación de optimización de Lagrange aplicada al proceso de estimación de movimiento. A partir de este análisis, se extraerán una serie de conclusiones que serán claves en el diseño de la estrategia adaptativa de estimación de movimiento propuesta en esta Tesis.

3.1.1 Motivación del análisis

El algoritmo de búsqueda exhaustiva obtiene las mayores tasas de compresión dentro de los algoritmos de ajuste de bloques mediante la evaluación de todas las posibles posiciones dentro del área de búsqueda. Sin embargo, el principal problema que presenta este algoritmo, aparte de su elevado coste computacional, es que se basa exclusivamente en minimizar una métrica de error (SAD), sin tener en cuenta la consistencia de los vectores de movimiento

calculados [JZC03]. En el contexto de la estimación de movimiento, la consistencia del campo de vectores de movimiento se refiere al hecho de que no existan grandes diferencias entre vectores de movimiento de bloques vecinos, o dicho de otra manera, que el conjunto de vectores de movimiento obtenido sea "suave". Esta suposición está completamente justificada, pues, lo que no parece lógico es que bloques de píxeles que pertenecen a un mismo objeto dentro de una misma imagen presenten movimientos con velocidades y direcciones diametralmente opuestas.

Sin embargo, y debido a los problemas ya mencionados del algoritmo de búsqueda exhaustiva, una situación muy común que se produce tras una estimación de movimiento basada en este algoritmo es la obtención de un conjunto vectores de movimiento que no reproducen fielmente el movimiento real existente en la secuencia de vídeo, pudiendo obtenerse para bloques vecinos resultados completamente dispares. Debido a que, tal y como y se ha explicado, en los estándares de compresión de vídeo los vectores de movimiento se codifican de forma diferencial con respecto a los vectores de bloques vecinos, este hecho conlleva que se emplee un número excesivo de bits para la transmisión de todos los vectores de movimiento correspondientes a un fotograma, y por lo tanto un considerable aumento del valor de la función de Lagrange J_{motion}.

Asimismo, la obtención de *vectores de movimiento verdaderos* - entendiéndose por vector de movimiento verdadero aquel que reproduce el movimiento real de la escena que se analiza – resulta ser de vital importancia en aplicaciones que *cohabitan* con los sistemas de compresión de vídeo. Ejemplos claros de estas aplicaciones, en las que las prestaciones globales del sistema dependen extraordinariamente de la veracidad de los vectores de movimiento, son:

- **Aumento de la resolución temporal de la secuencia de vídeo**. En muchas ocasiones, con el objetivo de aumentar la tasa de compresión obtenida o bien ante situaciones variables en el canal de transmisión, se submuestrea en la dimensión temporal la

secuencia de vídeo a comprimir, disminuyendo así el número de fotogramas por segundo. Esta situación es corregida en el decodificador, realizando una interpolación temporal de los fotogramas suprimidos por el codificador a partir de la información contenida en las imágenes decodificadas [LKP03], [HLK04].

- **Aumento de la resolución espacial de la secuencia de vídeo mediante técnicas de *super-resolución***. Con el mismo propósito que para el caso anterior, la resolución espacial de una secuencia de vídeo puede ser disminuida antes de proceder a su transmisión. Las técnicas de *super-resolución* [Mar03] han demostrado constituir una eficaz alternativa en el aumento de la resolución en imágenes [CPL+06] y secuencias de vídeo, ya sean estas últimas en formato *crudo* [BCL+05], [CLP+05a], [CLP+05b] o comprimido [SKM+04].

- **Corrección de errores**. Durante la transmisión de vídeo comprimido sobre canales ruidosos es probable que parte o toda la información asociada a un macrobloque (o a un conjunto de ellos) no se reciba correctamente en su destino. En este caso, el decodificador debe tratar, en la medida de lo posible, de restaurar la información perdida. Para ello se hace uso de la información existente en macrobloques vecinos correctamente recibidos [SR01, pp. 217-250], [Sad02, pp.125-130], incluyendo sus vectores de movimiento [YB04], [YN04].

- **Transcodificación**. Se denomina así al proceso mediante el cual una trama de vídeo comprimido es convertida a otra de características diferentes, implicando cambios en la resolución espacial o temporal, en el nivel de compresión, o en la sintaxis de la trama, sin necesidad de realizar un ciclo completo de decodificación/codificación [Sad02, pp. 215-256], [XLS05]. Durante este proceso, es común la obtención de un nuevo conjunto de vectores de movimiento a partir de los vectores recibidos, ya sea

mediante reutilización directa de los vectores enviados [YVL+02] o mediante un proceso de re-estimación a partir de éstos [TS04].

- **Segmentación de objetos**. Como ya se ha explicado en el capítulo anterior, el estándar MPEG-4 introduce, como novedad con respecto a sus predecesores, la codificación de objetos audiovisuales. En este sentido, es necesario aislar los diferentes objetos que componen una escena. Para ello, se recurre frecuentemente a algoritmos de segmentación basados en los resultados obtenidos mediante ajuste de bloques rectangulares [TA02].

Por otra parte, diversos trabajos centrados exclusivamente en la necesidad de aumentar la tasa de compresión, han demostrado que un suavizado de los vectores de movimiento puede resultar altamente beneficioso para reducir el ancho de banda necesario en la transmisión de señales de vídeo [CK97], [AS00], [OWX03]. Sin embargo, las aportaciones relacionadas con el suavizado de vectores de movimiento en algoritmos de ajuste de bloques presentan un inconveniente común: los resultados son extremadamente dependientes de la secuencia de vídeo que se desea comprimir. Este hecho hace que los resultados obtenidos tras la aplicación de algoritmos de suavizado sean satisfactorios para algunas secuencias pero no para otras, llegando a situaciones en las que incluso se puede obtener una peor tasa de compresión tras el proceso de suavizado que la obtenida con el algoritmo de ajuste por bloques original [CK97], [JCK99], [TC01]. Este problema se interpreta claramente en términos de la ecuación de Lagrange, ya que se está disminuyendo el número de bits R_{motion} necesario para transmitir los vectores de movimiento a costa de un excesivo incremento en el SAD, provocando un aumento global de la función de coste J_{motion} (ver ecuación 2.10 en el capítulo anterior)

Por todo ello, queda de manifiesto la importancia de contar con algún tipo de entorno de simulación que permita analizar de manera detallada el grado de suavidad y veracidad de los vectores de movimiento tras aplicar un proceso de estimación de movimiento mediante ajuste

de bloques, así como las implicaciones que conlleva la *cantidad de suavizado* introducida en la tasa de compresión obtenida.

3.1.2 Entorno de análisis: propuesta sobre los parámetros de caracterización

La Figura 3.1 describe el novedoso entorno de análisis propuesto, el cual ha sido presentado en el congreso internacional *Euromicro* del año 2003 [LCL+03a]. La metodología de análisis se explica a continuación. En primer lugar, se toma un fotograma de una secuencia de vídeo cualquiera y se crea a partir de éste una secuencia de diez fotogramas mediante la introducción de nueve vectores de movimiento con precisión de píxel generados de manera aleatoria. Para ello se utiliza el bloque MOVER, el cual desplaza cada píxel de la imagen de referencia según lo indicado por el vector de movimiento correspondiente. Una vez que se ha creado una secuencia en la cual el movimiento está perfectamente controlado, se aplica sobre ella un algoritmo de ajuste de bloques por búsqueda exhaustiva FSBM (*Full Search Block Matching*), para finalmente comparar los vectores de movimiento obtenidos con los introducidos para la generación de la secuencia de vídeo, determinándose si los vectores de movimiento obtenidos son verdaderos o falsos. Es de destacar que el estimador de movimiento está integrado dentro de un codificador de vídeo H.263, permitiendo por lo tanto, el estudio de la variación de la veracidad de los vectores de movimiento con el escalón de cuantificación Q_p. En cualquier caso, y a pesar de haber utilizado el mencionado estándar, los resultados y conclusiones que se establecerán en este capítulo son totalmente extrapolables a cualquier codificador híbrido de vídeo basado en algún estándar de compresión.

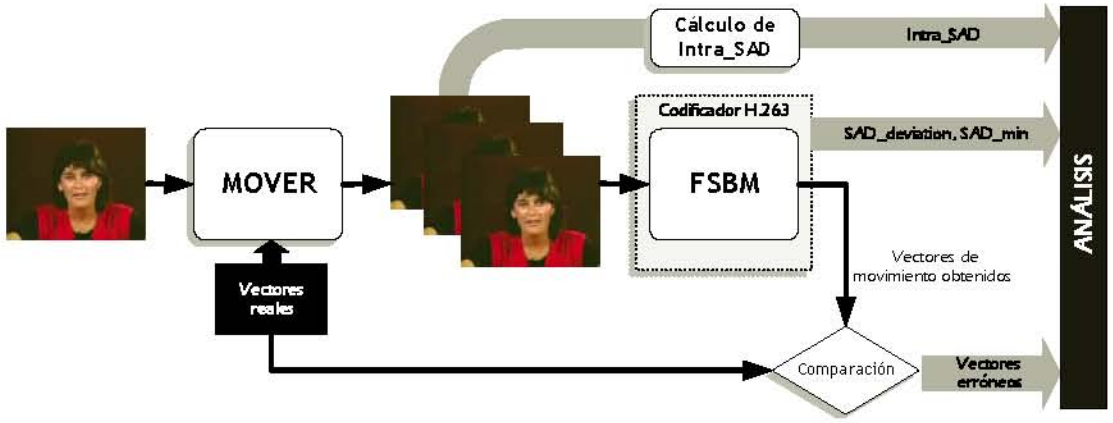


Figura 3.1: Diagrama de bloques del entorno de análisis propuesto.

Como se explicó en el apartado anterior, lo que se pretende realizar no es sólo una caracterización de los vectores de movimiento obtenidos con algoritmos de ajuste de bloques en términos de la veracidad de dichos vectores, sino también extraer una serie de conclusiones en términos de la ecuación de Lagrange para la optimización de la estimación de movimiento. Esta información será utilizada en la estrategia de estimación de movimiento adaptativa que se presenta en esta Tesis. Para ello, se han introducido, además del mínimo SAD (***SAD_min***), dos nuevos parámetros denominados ***Intra_SAD*** y ***SAD_deviation*** que se definen a continuación.

En primer lugar, es lógico pensar que los bloques que presentan una alta varianza espacial tienen una alta probabilidad de presentar vectores de movimiento verdaderos. Esto es debido a que este tipo de bloques presentará un buen ajuste en una sola posición del área de búsqueda. Según esta idea, el primer parámetro introducido para el proceso de caracterización es el que se ha denominado ***Intra_SAD***, definido de la siguiente forma:

$$Intra\_SAD_t^{xy} = \sum_{i=1}^{m}\sum_{j=1}^{n}\left|p_t^{xy}(i,j) - \mu_t^{xy}\right|, \qquad \text{(ec. 3.1)}$$

donde $p_t^{xy}(i,j)$ representa el valor de luminancia del píxel en la posición (i,j) dentro del bloque de $m \times n$ píxeles situado en las coordenadas (x,y) correspondiente al fotograma actual

(instante t) y μ_t^{xy} representa la media aritmética de todos los valores de luminancia de los píxeles que se encuentran en dicho bloque.

Por otra parte, es importante recordar que, de entre todas las posiciones evaluadas dentro del área de búsqueda por un algoritmo de búsqueda exhaustiva, el vector de movimiento finalmente elegido es calculado a partir de la posición que presenta la menor distorsión (SAD) con el bloque de referencia. Parece evidente pensar que si una posición candidata presenta un SAD muchísimo menor que el resto de las posiciones evaluadas, el vector resultante presenta una alta probabilidad de ser verdadero. Por ello, el segundo parámetro denominado ***SAD_deviation*** es definido para cada bloque de referencia de coordenadas *(x,y)* como:

$$SAD\_deviation_t^{xy} = \sum_{u=-p1}^{p2} \sum_{v=-p1}^{p2} \left| SAD(u,v) - SAD_{\min}^{xy} \right|, \qquad \text{(ec. 3.2)}$$

donde $SAD(u,v)$ representa el SAD de cada una de las posiciones (u,v) evaluadas dentro del área de búsqueda y SAD_{min}^{xy} representa el mínimo obtenido y que es, tal y como ya se ha comentado, el que determina el vector de movimiento para el bloque de referencia que se está analizando.

3.1.3 Resultados de la caracterización

Haciendo uso del esquema detallado en la Figura 3.1, se ha realizado un estudio en términos de los parámetros señalados para valores del escalón de cuantificación Q_p=2, 6, 10, 14, 18, 22, 26 y 30. En particular, se presentan los vectores de movimiento obtenidos con búsqueda exhaustiva con precisión de píxel sobre bloques de 16×16 píxeles (macrobloques) con un área de búsqueda de 46×46 píxeles ($p1$ = $p2$ = 15). Hay que destacar que, con el objetivo de evitar problemas en los bordes de las imágenes, la estimación de movimiento en este estudio

se realizó sólo para los *macrobloques centrales*, esto es, para aquellos macrobloques que no se encuentran ni en la primera ni en la última fila o columna de la imagen. Aunque estas pruebas se realizaron con todas las secuencias de tamaño QCIF (176×144 píxeles) descritas en el Anexo de esta Tesis, en las sucesivas figuras se presentan los resultados tomando como entrada del bloque MOVER el primer fotograma de la secuencia MISS AMERICA (el mismo fotograma que aparece en la Figura 3.1), pues resultan significativos de la tendencia general observada.

3.1.3.1 Resultados de veracidad de los vectores de movimiento en términos de *Intra_SAD* y *SAD_min*

En las Figuras 3.2 y 3.3 se presentan los resultados obtenidos para la mencionada secuencia de vídeo con los factores de cuantificación Q_p=30 y Q_p=2, respectivamente. Como se puede observar, se presentan seis gráficas por figura en las cuales aparecen los diferentes bloques de 16×16 píxeles para los que se obtuvieron vectores de movimiento, quedando representado en el eje horizontal el valor del *Intra_SAD* de cada uno de estos bloques y en el eje vertical el *SAD_min* derivado de la búsqueda exhaustiva llevada a cabo para el cálculo del vector de movimiento asociado a ese bloque, perteneciente al *vector de movimiento ganador*. En la esquina superior izquierda se presentan los bloques para los que se obtuvieron vectores de movimiento verdaderos en su componente horizontal (error = 0) mientras que en el resto se presentan, de izquierda a derecha y de arriba a abajo, los bloques con vectores de movimiento falsos (error = 1, error =2, error = 3, error = 4, y error ≥ 5 respectivamente), definiéndose el error como la diferencia en valor absoluto entre la componente horizontal del vector verdadero y la componente horizontal del vector calculado.

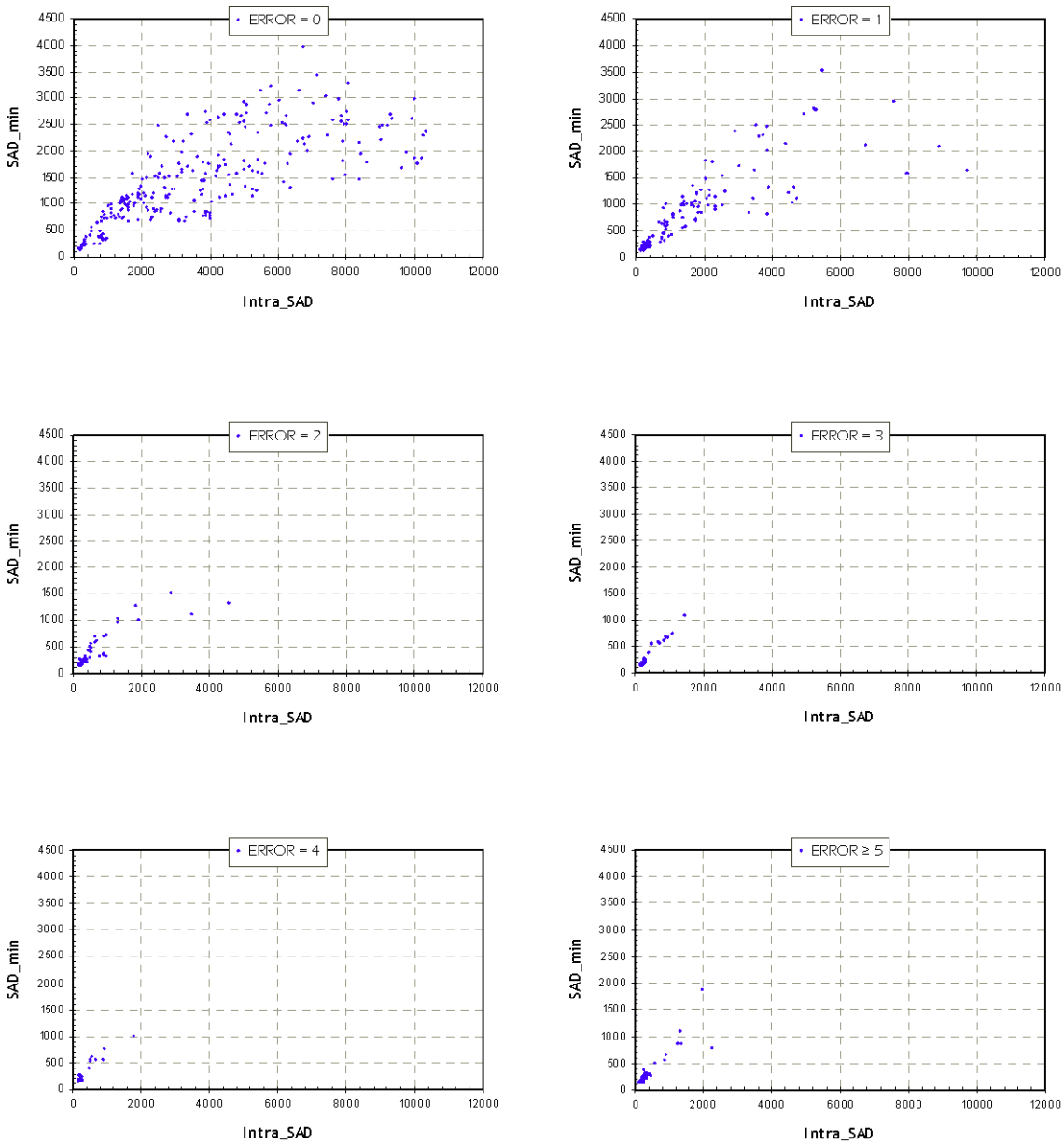


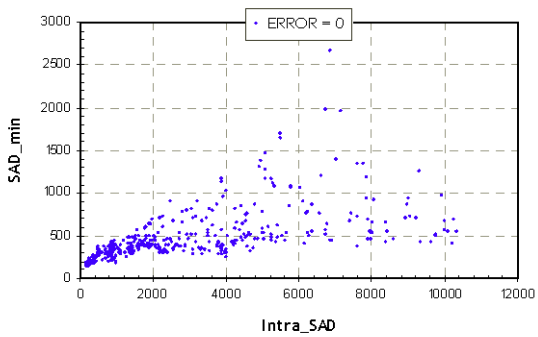
Figura 3.2: Resultados del proceso de caracterización en función de los parámetros *Intra_SAD* y *SAD_min* para $Qp = 30$.

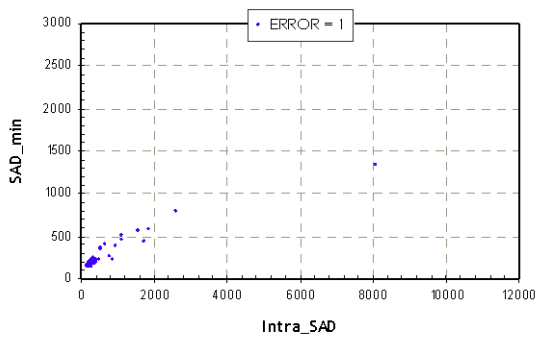
A partir de estas gráficas se extraen las siguientes conclusiones:

- Al aumentar el nivel de distorsión de la imagen recuperada con el aumento del escalón de cuantificación, el número de vectores erróneos es considerablemente mayor. De hecho, en el caso de estudio descrito, el porcentaje de vectores de movimiento verdaderos disminuye desde el 60.75% hasta el 38.23% al cambiar el escalón de cuantificación de Q_p=2 a Q_p=30.

- Debido a este aumento en el nivel de distorsión, los valores de *SAD_min* son mayores a medida que aumenta el valor del escalón de cuantificación, indicando una menor similitud entre imágenes. Asimismo, los macrobloques de alta varianza espacial (valores de *Intra_SAD* elevados) presentan altos valores de *SAD_min*, siendo este hecho más evidente a medida que aumenta el nivel de distorsión.

- Los macrobloques con valores altos de *Intra_SAD* presentan un mayor grado de probabilidad de ser verdaderos, probabilidad que aumenta con la disminución del escalón de cuantificación. Este hecho, junto con el anterior, desmiente la clásica suposición de que un vector de movimiento que tiene asociado un alto valor de *SAD_min* es fruto de una mala estimación de movimiento.





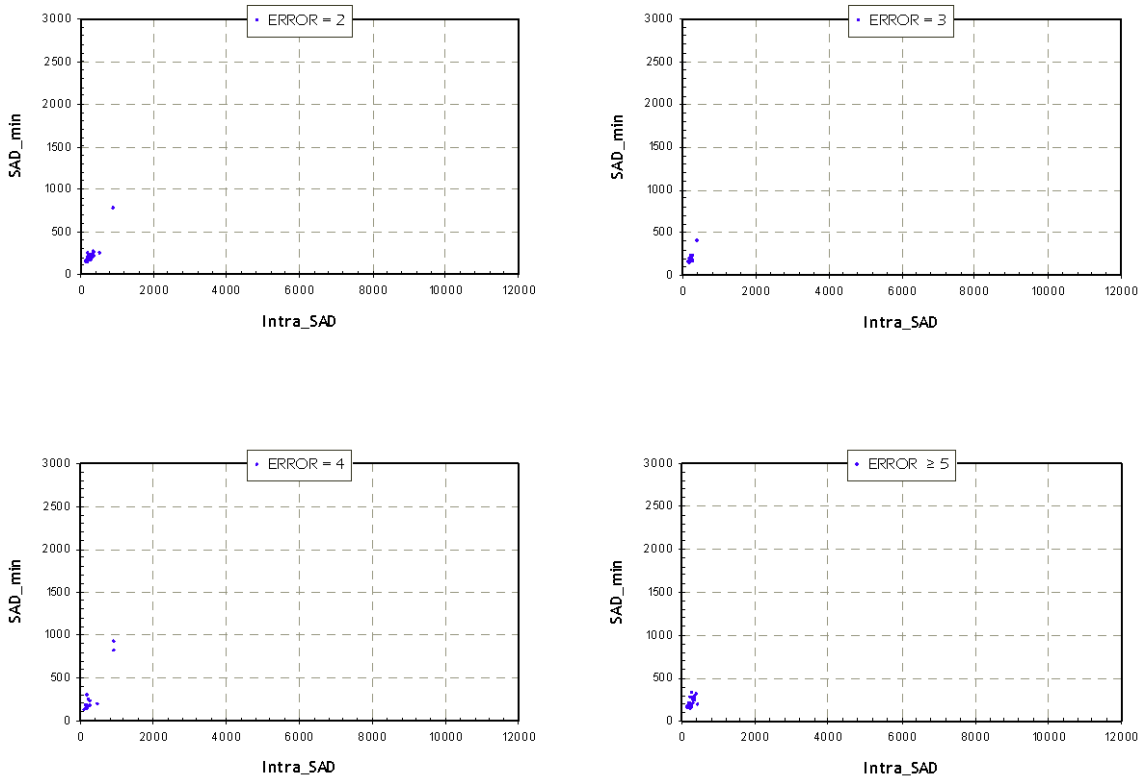


Figura 3.3: Resultados del proceso de caracterización en función de los parámetros *Intra_SAD* y *SAD_min* para $Qp = 2$.

3.1.3.2 Resultados de la veracidad de los vectores de movimiento en términos de *Intra_SAD* y *SAD_deviation*

Se presentan en las Figuras 3.4 (Q_p=30) y 3.5 (Q_p=2) las gráficas análogas a las anteriormente presentadas, pero esta vez en función de los parámetros *Intra_SAD* y *SAD_Deviation*.

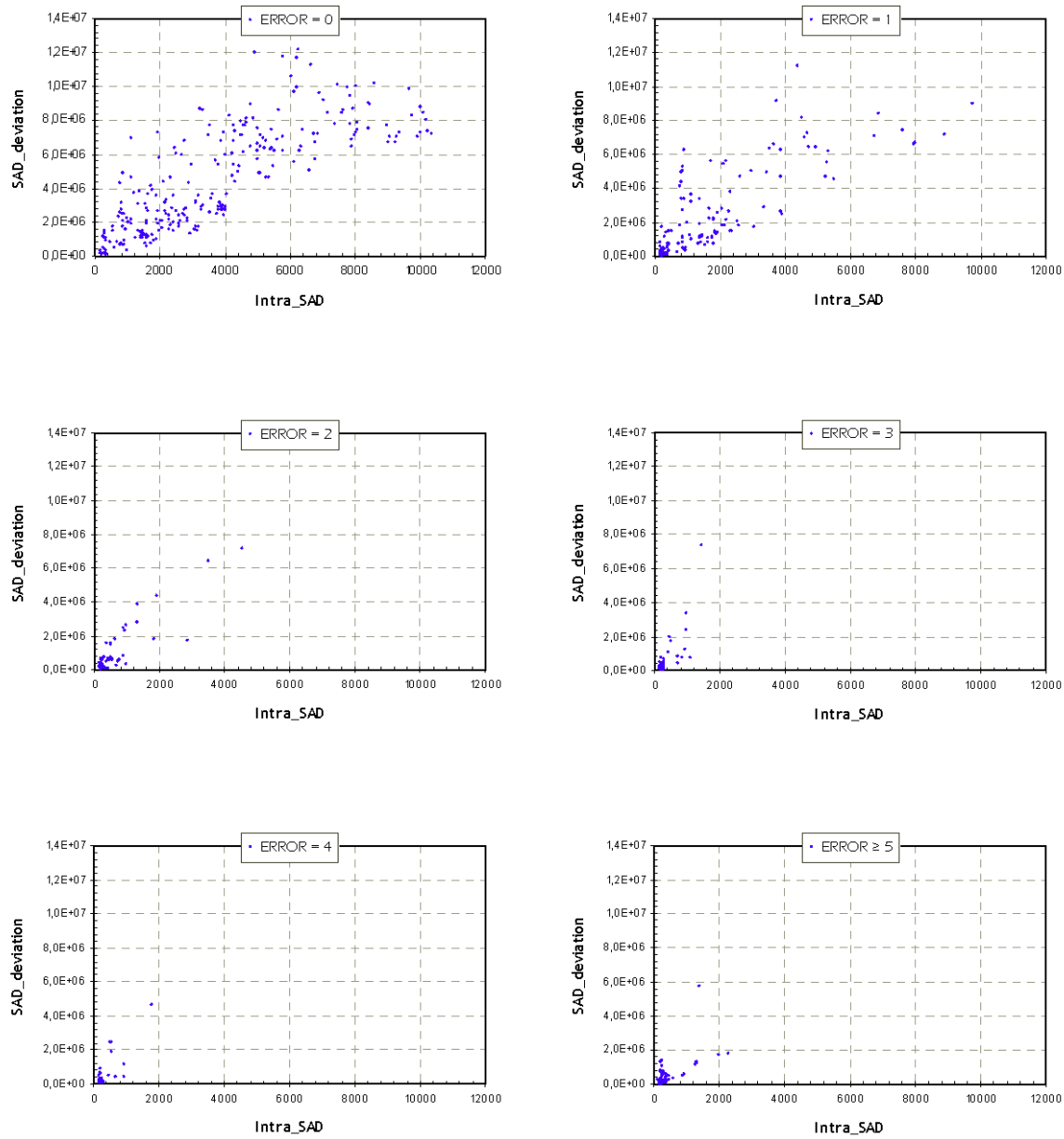
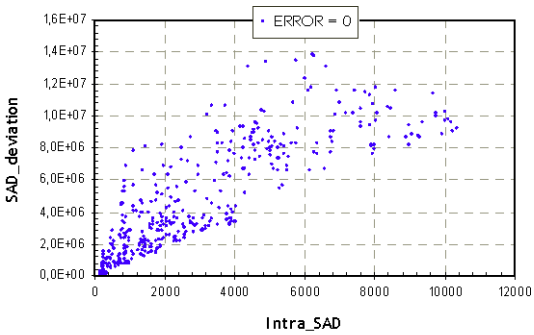


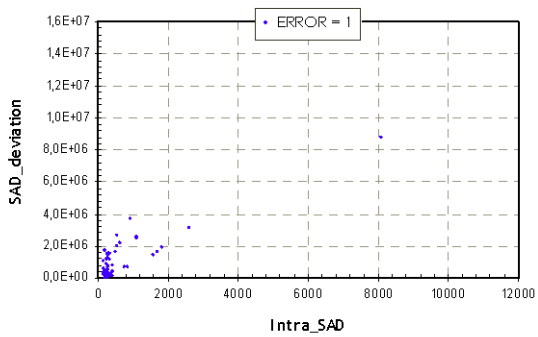
Figura 3.4: Resultados del proceso de caracterización en función de los parámetros *Intra_SAD* y *SAD_deviation* para $Qp = 30$.

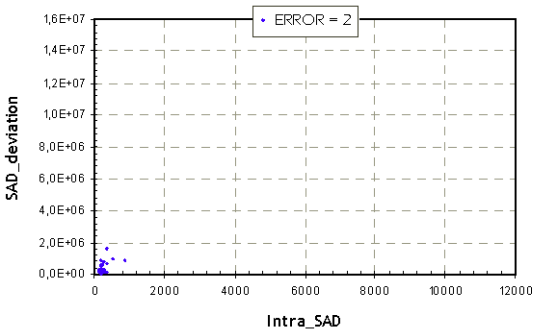
Además de confirmar las conclusiones establecidas en el apartado anterior acerca del porcentaje de vectores de movimiento verdaderos en función del escalón de cuantificación, estas gráficas permiten extraer nuevas conclusiones:

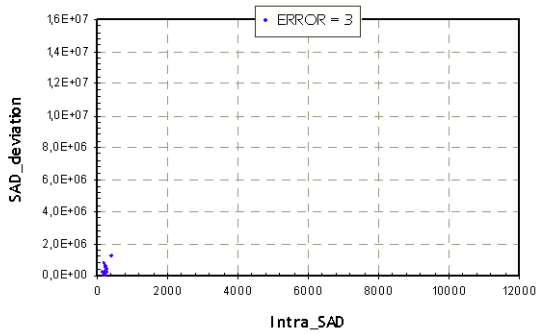
- Los macrobloques que presentan un alto valor de *Intra_SAD* tienen asociado un valor de *SAD_deviation* elevado.

- A pesar de que en la magnitud del *SAD_min* existen notables diferencias entre los dos casos extremos del valor de cuantificación seleccionados, estas diferencias son menos apreciables en lo que respecta al valor de *SAD_deviation*. Dicho de otra manera, los bloques de alta varianza espacial presentan un alto valor de *SAD_deviation*, independientemente del nivel de compresión.









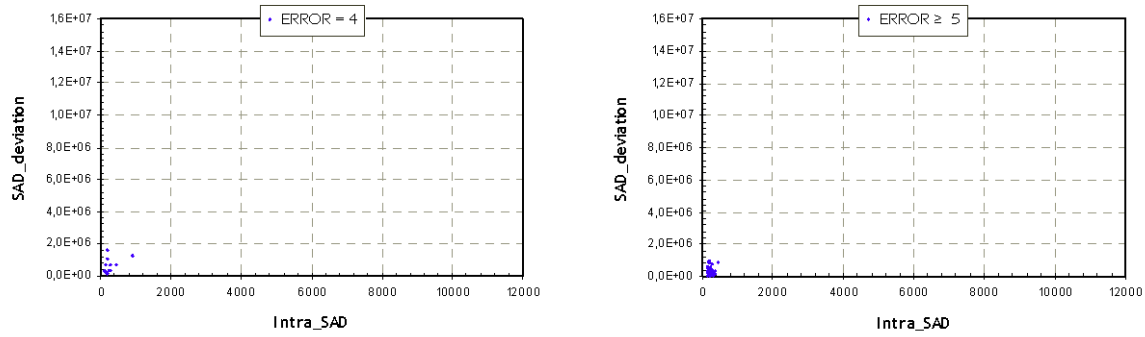


Figura 3.5: Resultados del proceso de caracterización en función de los parámetros *Intra_SAD* y *SAD_deviation* para $Qp = 2$.

3.1.3.3 Evolución del número de vectores de movimiento verdaderos con el nivel de compresión

Como último resultado de esta caracterización se presenta, para tres secuencias de vídeo obtenidas a partir del entorno de simulación propuesto tomando el primer fotograma de las secuencias SUZIE, MISS AMERICA y COASTGUARD, la tendencia del número de vectores de movimiento verdaderos con el valor del escalón de cuantificación.

Los resultados obtenidos se muestran en la Figura 3.6, confirmando claramente las afirmaciones establecidas en los apartados anteriores de esta misma sección 3.1.3. En este sentido, si se analizan los resultados obtenidos para un mismo valor del escalón de cuantificación, se constata que la mayor cantidad de vectores de movimiento verdaderos se obtiene para la secuencia con mayor actividad espacial, en este caso, la secuencia creada a partir del primer fotograma de SUZIE. Asimismo, se observa que para cualquiera de las tres secuencias de vídeo estudiadas, el número de vectores verdaderos obtenidos tras una estimación de movimiento por ajuste de bloques disminuye con el aumento del factor de cuantificación de manera no lineal.

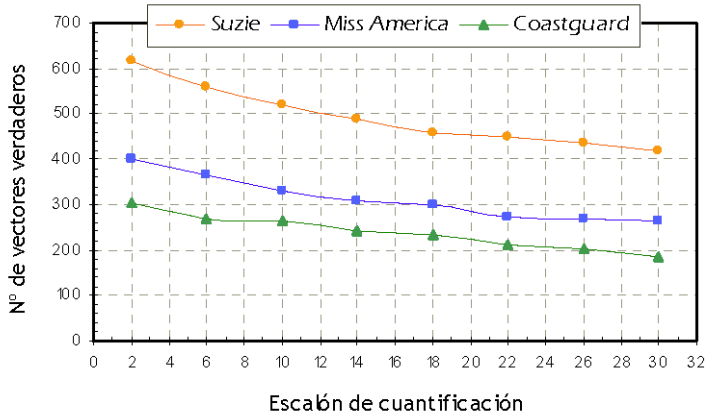


Figura 3.6: Evolución del número de vectores verdaderos para las tres secuencias estudiadas.

3.1.4 Análisis de los resultados y conclusiones

Los resultados obtenidos en el apartado anterior permiten establecer una serie de afirmaciones en cuanto al uso de un determinado algoritmo de estimación de movimiento por ajuste de bloques y su repercusión en la tasa de compresión. Así, analizando dichos resultados e interpretándolos en función de los parámetros propios de la ecuación de Lagrange para la estimación de movimiento, se establecen las siguientes conclusiones:

- En los bloques de alta actividad espacial, la evaluación de un número elevado de posiciones dentro del área de búsqueda es crítica.

  - Los bloques que presentan un *Intra_SAD* alto se caracterizan por tener asociado un elevado valor de *SAD_deviation*. Por lo tanto, si el algoritmo de ajuste de bloques no es capaz de encontrar la posición de mínimo SAD dentro del área de búsqueda, existen muchas probabilidades de que la función de coste J_{motion} crezca drásticamente, independientemente del nivel de compresión con el que se esté trabajando. Este hecho ocurre con frecuencia en

algoritmos rápidos de estimación de movimiento los cuales, debido a sus características intrínsecas de funcionamiento, quedan típicamente atrapados en mínimos locales.

- La aplicación del algoritmo de búsqueda exhaustiva resulta ser muy eficaz a la hora de proporcionar un vector de movimiento verdadero para este tipo de bloques, particularmente para valores bajos del escalón de cuantificación. Por lo tanto, al ser el movimiento real de las escenas generalmente *suave*, el término R_{motion} se beneficia también del uso de esta estrategia de búsqueda.

- En los bloques de baja actividad espacial, la obtención de un campo de vectores de movimiento homogéneo es crítica.

  - Los bloques que presentan un *Intra_SAD* bajo también presentan un valor de *SAD_deviation* reducido. Este hecho determina que la aplicación de un algoritmo rápido no suponga un riesgo elevado en cuanto al crecimiento de la función de coste J_{motion}, pues el SAD asociado a una posición no mínima no distará demasiado del *SAD_min* correspondiente a dicha posición de mínimo. Además, y de manera análoga a lo que ocurría en los bloques de alta actividad espacial, este hecho se da para todo el rango posible de valores del escalón de cuantificación.
  - La obtención de un campo de vectores de movimiento homogéneo es crucial en estos bloques, pues de no ser así, el término R_{motion} aumentará considerablemente. Este hecho se observa claramente en la Figura 3.7, en la cual se representa una imagen tipo P decodificada junto con los vectores de movimiento asociados a cada uno de sus macrobloques. En las zonas resaltadas en dicha figura se observa que el conjunto obtenido de vectores de movimiento es altamente incoherente, provocando un considerable aumento en el número de bits necesario para la transmisión diferencial de los vectores

de movimiento. Debido a la propia naturaleza de la ecuación de Lagrange, en la cual el término R_{motion} aparece multiplicado por el escalón de cuantificación, y al hecho aquí probado de que el número de vectores de movimiento falsos aumenta con el nivel de distorsión, este fenómeno es aún más crítico en las aplicaciones en las que se requiere de una alta tasa de compresión. Por lo tanto, la aplicación de un algoritmo de naturaleza predictiva resulta una excelente opción en estos bloques, puesto que se garantiza un bajo R_{motion} a la par que se consigue un elevado número de vectores de movimiento verdaderos [HBH+93], [HB98].

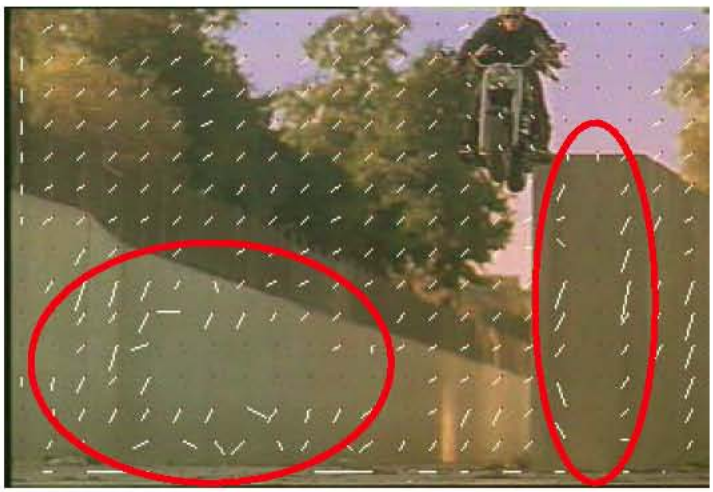


Figura 3.7: Incoherencia en el campo de vectores de movimiento en bloques de baja actividad espacial.

- **La elección de una determinada estrategia de estimación de movimiento en un codificador de vídeo no debe basarse exclusivamente en métricas de error.** Independientemente del nivel de compresión que se desee obtener, se ha puesto de manifiesto la inexactitud de la clásica afirmación de que un buen vector de movimiento es aquel que presenta un SAD pequeño. Así, la aplicación de un algoritmo de ajuste de bloques por búsqueda exhaustiva en bloques de baja actividad espacial garantiza la obtención de vectores de movimiento con SADs reducidos, y sin

embargo, conlleva un aumento excesivo del número de bits necesario para la transmisión de dichos vectores de movimiento.

3.2 Post-procesamiento de vectores de movimiento

En el apartado anterior se ha expuesto la importancia de obtener un campo de vectores de movimiento coherente para la optimización de la tasa de compresión de un codificador híbrido de vídeo. Este hecho cobra especial relevancia en los bloques de baja actividad espacial, en los que la aplicación de un algoritmo de ajuste de bloques por búsqueda exhaustiva, si bien encuentra el mínimo SAD posible dentro del área de búsqueda, resulta claramente ineficiente en cuanto a la transmisión de los vectores de movimiento calculados.

Por lo tanto, en codificadores de vídeo en los que se desee alta capacidad de compresión haciendo uso de un algoritmo de ajuste de bloques por búsqueda exhaustiva, resulta necesaria la modificación de los vectores obtenidos sin que ello suponga un detrimento en las prestaciones originales, independientemente de la secuencia de vídeo a comprimir. Una solución a este problema podría obtenerse mediante la aplicación de etapas de post-procesamiento de los vectores de movimiento. En este apartado se analiza esta posibilidad y se detallan los resultados de su aplicación a las secuencias que se utilizan como elementos de prueba.

3.2.1 Estrategia de post-procesamiento propuesta

La etapa de post-procesamiento propuesta, la cual ha sido publicada en la revista internacional *IEE Electronics Letters* [LCL+03b] y en la revista de ámbito local *Vector Plus* [LCL+05a], se basa en realizar un filtrado del vector de movimiento obtenido. Para ello, se

combinan los vectores obtenidos en una vecindad espacial de dicho vector de movimiento según la siguiente ecuación:

$$mv_{pp}{}^{0} = \frac{\sum_{i=0}^{8} W_t^i \cdot mv_t^i}{\sum_{i=0}^{8} W_t^0}, \qquad \text{(ec. 3.3)}$$

donde $mv_{pp}{}^{0}$ representa el vector de movimiento post-procesado, mv_t^i representa el conjunto de los vectores de movimiento incluidos en la vecindad espacial del vector de movimiento según indica la Figura 3.8 (aparece sombreado el vector de movimiento que se desea procesar), y W_t^i representa un conjunto de pesos dados a los vectores de movimiento señalados en dicha figura, permitiendo asignar diferentes niveles de importancia entre los vectores de movimiento participantes en el filtrado.

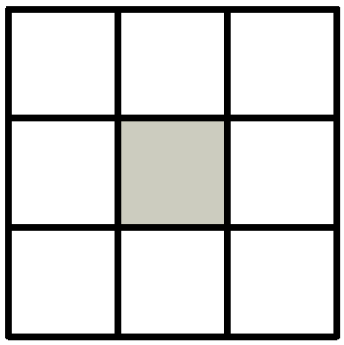


Figura 3.8: Vecindad espacial considerada para la etapa de post-procesamiento.

A la hora de asignar valores a los pesos W_t^i habrá que prestar especial atención a las conclusiones extraídas en el apartado anterior. En primer lugar, habrá que darle mayor peso a aquellos vectores de la vecindad que presenten un valor más alto de *Intra_SAD*, pues son los que contribuyen en mayor medida a la obtención de un conjunto de vectores de movimiento verdaderos. Sin embargo, no se debe olvidar que el objetivo de la etapa de post-

procesamiento aquí propuesta es optimizar la tasa de compresión de un codificador de vídeo basado en estimación de movimiento por búsqueda exhaustiva, y por lo tanto, habrá que tener especial cuidado en cambiar un vector de movimiento de un bloque que tiene asociado un elevado *SAD_Deviation*.

Con estas consideraciones, los pesos en la ecuación de post-procesamiento han sido asignados de la siguiente manera:

$$mv_{pp}{}^{0} = \frac{SAD\_deviation\_red_t^{0} \cdot mv_t^{0} \cdot Intra\_SAD_t^{0} + \sum_{i=1}^{8} w_t^{i} \cdot mv_t^{i} \cdot Intra\_SAD_t^{i}}{SAD\_deviation\_red_t^{0} \cdot Intra\_SAD_t^{0} + \sum_{i=1}^{8} w_t^{i} \cdot Intra\_SAD_t^{i}}, \qquad \text{(ec. 3.4)}$$

donde se ha establecido un nuevo conjunto de pesos w_t^i asociados a los vectores de movimiento vecinos y se ha añadido un nuevo parámetro definido como:

$$SAD\_deviation\_red_t^{0} = SAD\_max^{0} - SAD\_min^{0}, \qquad \text{(ec. 3.5)}$$

el cual representa la máxima variación del *SAD* para la posición evaluada.

3.2.2 Resultados obtenidos con la etapa de post-procesamiento

La etapa de post-procesamiento (*PP*) diseñada ha sido introducida en un codificador H.263 tal y como se indica en la Figura 3.9.

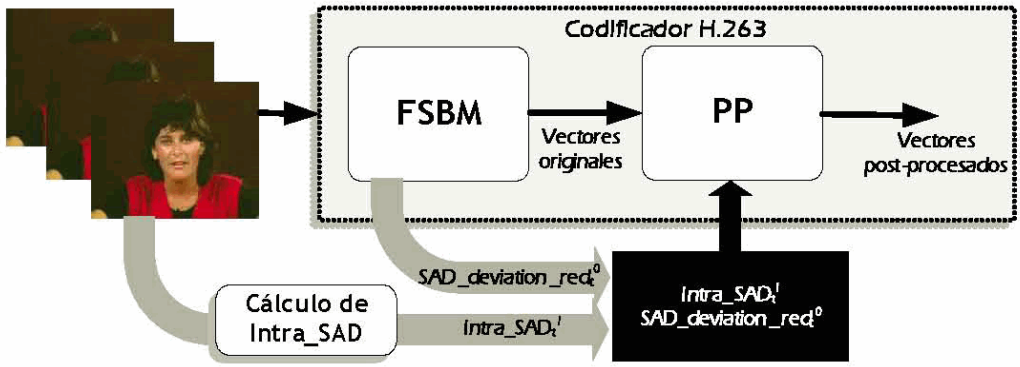


Figura 3.9: Introducción de la etapa de post-procesamiento (PP) en un codificador H.263.

Antes de presentar los resultados obtenidos, hay que destacar que, de los ocho vectores de movimiento alrededor del vector a procesar, sólo se introdujeron en la ecuación de post-procesado los vectores denominados mv_t^2, mv_t^3 y mv_t^4 debido a dos razones:

- En primer lugar, debido a que en los codificadores de vídeo el procesamiento de las imágenes se realiza en orden lexicográfico, por lo que en el momento en el que se post-procesa el vector mv_t^0 sólo han sido calculados los vectores mv_t^1, mv_t^2, mv_t^3 y mv_t^4.

- En segundo lugar, los vectores mv_t^2, mv_t^3 y mv_t^4 son los utilizados por el estándar H.263 para la codificación diferencial de los vectores de movimiento (salvo en los macrobloques que se encuentran situados en los bordes de la imagen).

Con el objetivo de evaluar las prestaciones de la etapa de post-procesamiento presentada, se eligieron cuatro secuencias con texturas y movimientos muy diferentes: CARPHONE, FOREMAN, MISS AMERICA y TABLE. Los resultados obtenidos se presentan en la Figura 3.10, en la que se comparan las prestaciones del codificador de vídeo con y sin etapa de post-procesamiento, habiendo aplicado en ambos casos ajuste de bloques mediante búsqueda

exhaustiva con precisión de medio píxel sobre bloques de 16×16 píxeles con un área de búsqueda de 46×46 píxeles ($p1=p2=15$).

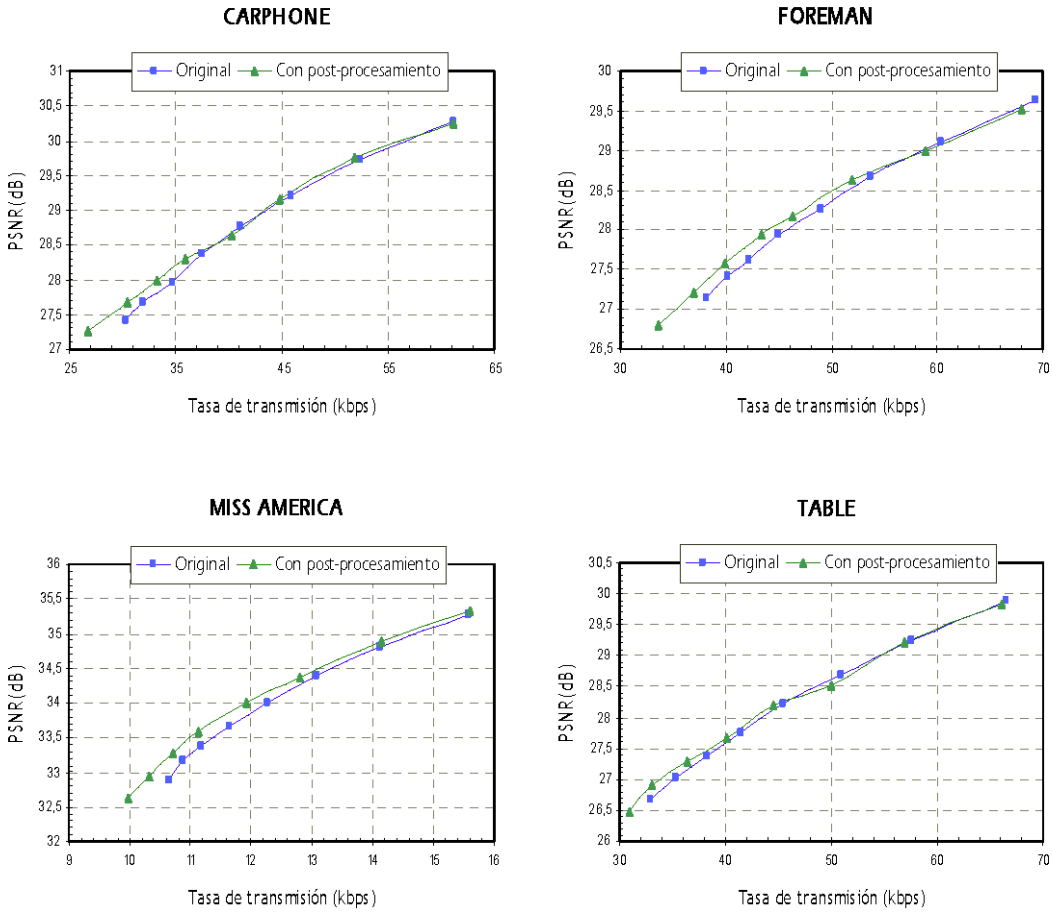


Figura 3.10: Prestaciones del algoritmo de post-procesamiento.

Tal y como se puede observar en cualquiera de las cuatro gráficas, la curva de prestaciones del codificador de vídeo mejora con la introducción de la etapa de post-procesamiento para la inmensa mayoría de los valores de tasa de transmisión simulados, independientemente de las características de la secuencia de vídeo. Este efecto se aprecia con mayor claridad para tasas bajas de transmisión, lo cual es completamente lógico según las afirmaciones realizadas en el apartado anterior. Sin embargo, esta modesta mejoría se obtiene a costa de un ligero aumento del coste computacional de una solución ya de por sí costosa computacionalmente,

pues han de calcularse los parámetros que aparecen en la ecuación 3.4, así como efectuar el post-procesamiento propiamente dicho.

Por lo tanto, esta etapa de post-procesamiento constituye sólo un primer paso en la consecución del objetivo marcado de obtener una solución válida para todo tipo de secuencias. No obstante, para conseguir la plenitud de los objetivos marcados en esta Tesis ha sido necesario desarrollar una estrategia radicalmente diferente a la de post-procesamiento, la cual se presentará en el siguiente apartado.

3.3 Propuestas algorítmicas para la estimación de movimiento

En el apartado anterior ha quedado de manifiesto que las mejoras introducidas mediante el uso de la etapa de post-procesamiento propuesta no se ven compensadas, en términos de compresión, con el aumento de coste computacional requerido. Es necesario, por tanto, buscar otro tipo de soluciones que den lugar a una estrategia global que haga que el resultado del proceso de estimación de movimiento sea independiente del tipo de secuencia, manteniendo el coste computacional lo más bajo posible. Esta aportación, que denominaremos **estimación de movimiento adaptativa (*Adaptive Cost Block Matching* - *ACBM*)**, constituye una de las propuestas fundamentales de esta Tesis. Básicamente, la estrategia se construye sobre la base de la combinación de diferentes algoritmos de estimación de movimiento por ajuste de bloques, y su principal objetivo es presentar un coste computacional adaptable, tanto a la secuencia de vídeo a comprimir, como a las necesidades de compresión. Estas propuestas algorítmicas han sido publicadas en la revista internacional *IEE Electronics Letters* [LLS05] y en el congreso internacional *Design, Automation and Test in Europe* (*DATE*) del año 2005 [LCL+05b].

3.3.1 Estimación de movimiento adaptativa

Los algoritmos rápidos de estimación de movimiento presentan un reducido coste computacional comparado con la solución por búsqueda exhaustiva. Sin embargo, la tasa de compresión alcanzada por estos algoritmos depende extraordinariamente de las características espaciales y temporales de la secuencia de vídeo a comprimir. En relación a este hecho, en el apartado 3.1 del presente trabajo se ha puesto de manifiesto la existencia de *bloques críticos* en los que es absolutamente necesario que el vector de movimiento apunte a la posición de mínimo SAD dentro área de búsqueda, o bien que éste sea coherente con los vectores de los bloques vecinos, para mantener una tasa de compresión elevada.

En este sentido, la estrategia propuesta se basa en calcular mediante un algoritmo rápido un primer vector de movimiento, para a continuación, evaluar la bondad de la estimación de movimiento realizada. Para ello se consideran, además de las características del propio vector de movimiento obtenido, las características de la secuencia a codificar y los requisitos de compresión. Si el vector obtenido no es satisfactorio en los términos anteriores, se descarta y se procede a la obtención de un vector de movimiento mediante búsqueda en un mayor número de posiciones del área de búsqueda.

En particular, el algoritmo rápido elegido es el publicado en [CFP02], denominado PBM (*Predictive Block Matching*), cuyo esquema de funcionamiento se ha detallado en el capítulo anterior. La elección de este algoritmo se debe a que es capaz de ofrecer tasas de compresión aceptables con un coste computacional extremadamente reducido, además de proporcionar un campo de vectores coherente, minimizando así el término R_{motion}. Además, de entre los algoritmos rápidos de estimación de movimiento más representativos, el algoritmo PBM resulta ser el de mejores prestaciones para un conjunto amplio de secuencias de vídeo, tal y como demuestran los trabajos [CFP02] y [HCT+06]. En el caso de que el resultado no sea

satisfactorio se ejecuta el algoritmo de búsqueda exhaustiva, tal y como se muestra en la Figura 3.11.

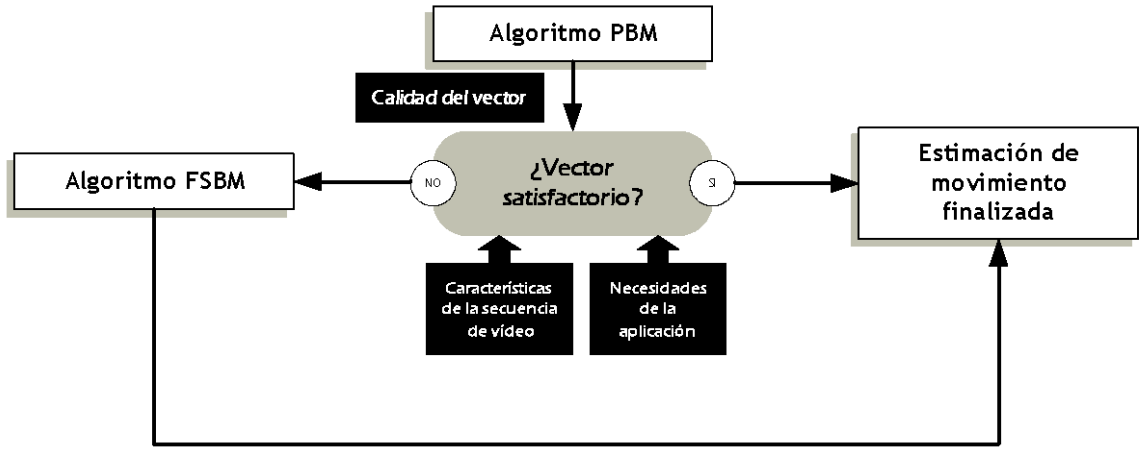


Figura 3.11: Esquema general de decisión dentro de la estrategia adaptativa.

3.3.2 Parámetros necesarios y criterios de decisión adaptativos

Una vez establecidas las líneas básicas de la estrategia de estimación de movimiento adaptativa, es necesario identificar un conjunto de parámetros que permita establecer, de manera sencilla, un criterio de decisión acerca de la validez del vector de movimiento obtenido por el algoritmo predictivo. Tal y como se muestra en la Figura 3.11, estos parámetros deberán proveer información acerca de tres factores:

- **Necesidades de la aplicación**. Obviamente, mientras mayor sea el número de bloques para los que el vector de movimiento calculado por el algoritmo predictivo es satisfactorio, menor será el coste computacional asociado. Sin embargo, para la inmensa mayoría de los casos, esta disminución del coste computacional llevará consigo una disminución de las prestaciones de compresión del codificador.

- **Características de la secuencia de vídeo a comprimir**. Una vez fijadas las necesidades de la aplicación, la adaptación a las singularidades de la secuencia a comprimir resulta clave para realizar un uso eficiente de los recursos de cómputo. Para ello, un parámetro clave es el *Intra_SAD* del bloque de referencia pues, tal y como ha quedado demostrado, su valor es indicativo de las probabilidades de éxito de un algoritmo de naturaleza predictiva. En particular, si el valor de *Intra_SAD* es bajo, el algoritmo PBM resulta una opción muy eficiente, decreciendo sus prestaciones a medida que el valor de *Intra_SAD* aumenta.

- **Calidad del vector de movimiento calculado por el algoritmo PBM**. Por último, es necesario evaluar de manera precisa la idoneidad del vector proporcionado por el algoritmo predictivo. Para este propósito, se inspecciona la magnitud del SAD del vector de movimiento calculado por el algoritmo PBM (de aquí en adelante *SAD_PBM*), pues mientras menor sea el *SAD_PBM*, menor será la función coste lagrangiana asociada a ese vector.

Con estos parámetros, y puesto que ya ha quedado de manifiesto en el presente capítulo que la valoración de una determinada estrategia de estimación de movimiento no debe estar basada exclusivamente en métricas de error, se establece como criterio de decisión la evaluación de la suma de los parámetros *Intra_SAD* y *SAD_PBM*, tal y como se indica en la Figura 3.12.

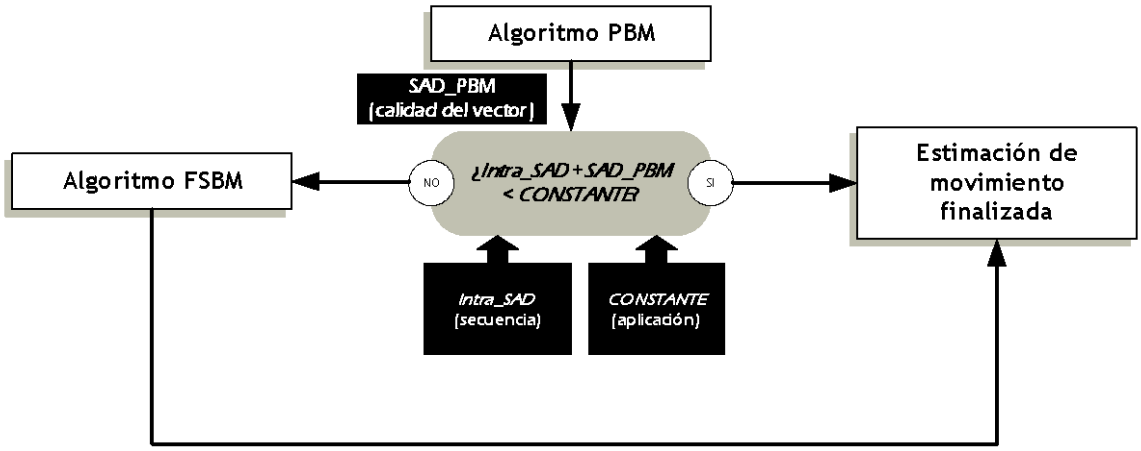


Figura 3.12: Esquema de decisión dentro de la estrategia adaptativa en función de los parámetros seleccionados.

3.3.3 Resultados de la estimación de movimiento adaptativa

Tal y como se ha expresado en el apartado anterior, el criterio de decisión vendrá determinado por una constante, *C*, que denominaremos *constante adaptativa*.

A continuación se presentan los resultados obtenidos con el criterio de decisión fijado previamente para diferentes valores de esta constante. En particular, los resultados se han obtenido con secuencias de muy diferentes características espaciales y temporales en las que todos los fotogramas, excepto el primero, han sido codificados en modo *INTER* con un codificador H.263. Los vectores de movimiento se han calculado con precisión de medio píxel sobre bloques de 16×16 píxeles con un área de búsqueda, en el caso de aplicar el algoritmo de búsqueda exhaustiva, de 46×46 píxeles ($p1=p2=15$). Asimismo, con el objetivo de evaluar la bondad del criterio de decisión para tasas de transmisión bajas, todas las secuencias han sido codificadas para valores del escalón de cuantificación entre 30 y 16. En todos los casos, se han elegido secuencias de tamaño QCIF muestreadas a 10 fotogramas por segundo debido a que, bajo estas condiciones de baja tasa de muestreo, la hipótesis de que el movimiento en

una secuencia de vídeo es suave y continuo pierde consistencia, y por lo tanto, se puede observar un alto porcentaje de casos en los que el algoritmo predictivo no es eficaz. Una vez fijado el criterio de decisión para estas condiciones, se investigará si es igualmente válido para tasas de transmisión y de muestreo temporal mayores.

En la Figura 3.13 se muestran los resultados obtenidos para las secuencias CARPHONE, FOREMAN, MISS AMERICA y TABLE para los siguientes valores de la constante adaptativa: C = 0, 5000, 10000, 15000 y 20000.

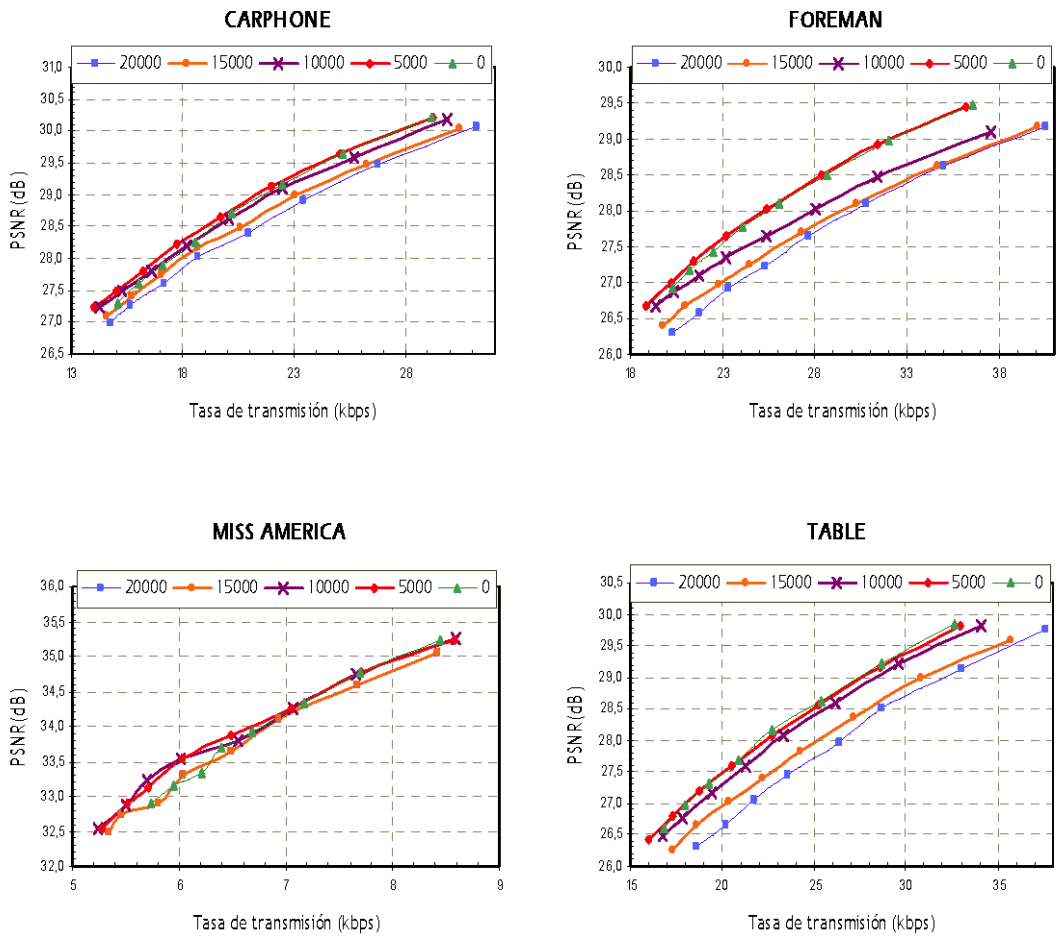


Figura 3.13: Resultados de compresión para diferentes valores de la constante adaptativa.

A partir de estas gráficas se extraen las siguientes conclusiones:

- Para tasas bajas de transmisión, las mejores prestaciones de codificación se obtienen para los valores de constante C = 5000 y C = 10000, incluso superiores a las obtenidas con C = 0. Por lo tanto, con estos valores se consigue *inyectar* una relación adecuada de vectores de movimiento obtenidos con el algoritmo predictivo que hacen que la función global de coste J_{motion} disminuya.

- Para tasas de transmisión mayores, las mejores prestaciones de codificación se obtienen para los valores de constante $C = 0$ y $C = 5000$. Al bajar el valor del escalón de cuantificación, el peso asignado al término R_{motion} disminuye y por lo tanto, cobra mucho mayor protagonismo dentro de la función de coste el SAD correspondiente al vector calculado.

- Las diferencias en las curvas de prestaciones provocadas por cambios en el valor de la constante son mucho más apreciables en el caso de secuencias con elevado movimiento y/o alta actividad espacial (FOREMAN y TABLE) que para el caso contrario (MISS AMERICA), en el cual apenas existen variaciones entre las curvas correspondientes a diferentes valores de la constante.

Con el objetivo de completar el análisis de los resultados obtenidos, se presentan en la Figura 3.14 el coste computacional (en términos del número medio de posiciones evaluadas por macrobloque dentro del área de búsqueda) obtenido para cada uno de los casos y secuencias estudiadas.

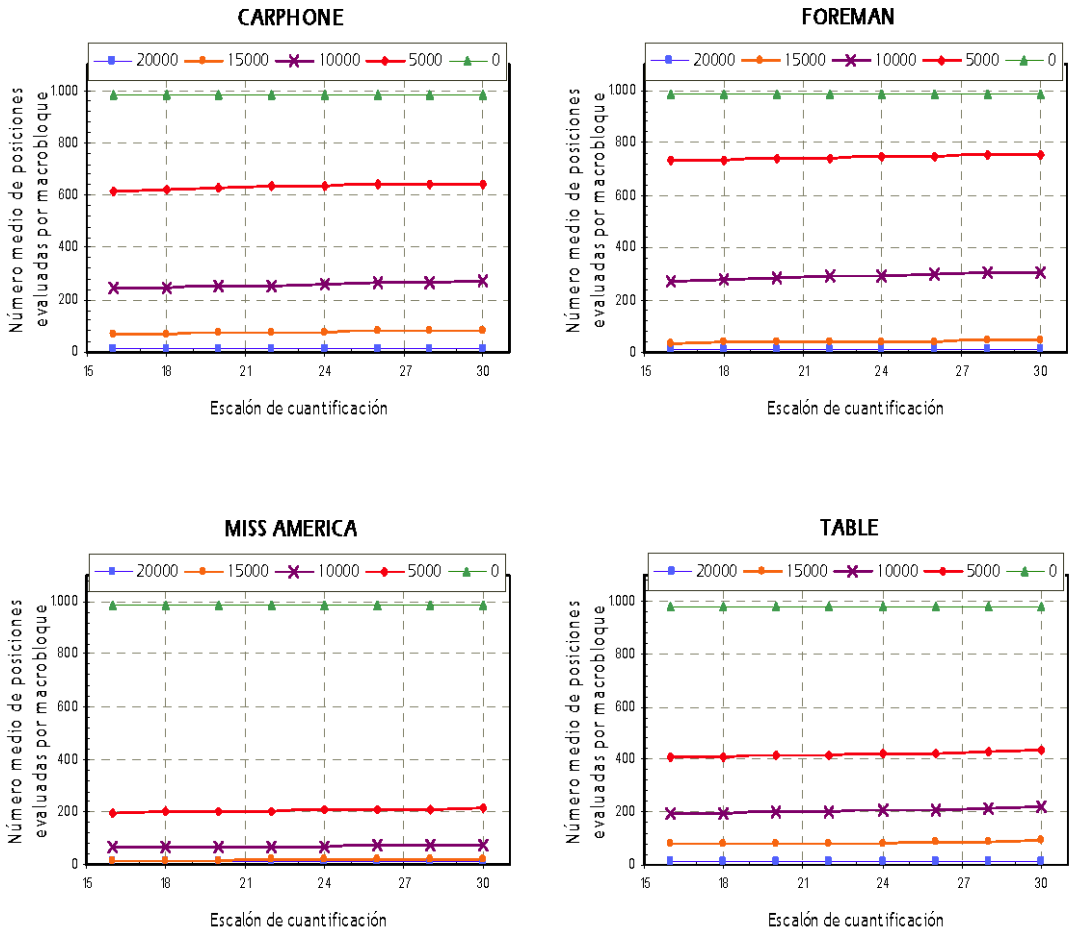


Figura 3.14: Número medio de posiciones evaluadas por macrobloque para diferentes valores de la constante adaptativa.

Como era de esperar, el número de posiciones evaluadas disminuye a medida que aumenta el valor de la constante. Además de esta evidencia, a partir de la observación de estas gráficas se puede afirmar lo siguiente:

- El número de posiciones evaluadas permanece estable aunque varíe el valor del escalón de cuantificación, y por lo tanto, el nivel de compresión.

- Para cada uno de los valores de la constante analizados, el menor coste computacional se obtiene para aquellas secuencias con poco movimiento y/o poca actividad espacial, como es el caso de la secuencia MISS AMERICA.

3.3.3.1 Ajuste de la constante adaptativa con Q_p

Con el objetivo de alcanzar siempre unas determinadas prestaciones de compresión, se introduce un ajuste cuadrático con Q_p de la constante adaptativa:

$$C(\alpha, \beta, Q_p) = \alpha + \beta \cdot Q_p^2 \qquad \text{(ec. 3.6)}$$

Como puede observarse, el valor de la constante adaptativa aumenta rápidamente con el nivel de compresión. Asimismo, con la finalidad de poder ajustar fácilmente su valor según las prestaciones requeridas, los parámetros α y β permiten realizar, respectivamente, un ajuste fino y un ajuste grueso del valor de esta constante. Con el objetivo de evaluar los cambios en las curvas correspondientes a las Figuras 3.13 y 3.14 mostradas anteriormente, se han fijado los valores de α y β de tal manera que se consigan, como mínimo, las mismas prestaciones que con $C = 0$ (equivalente a elegir siempre el vector de movimiento calculado por el algoritmo de búsqueda exhaustiva).

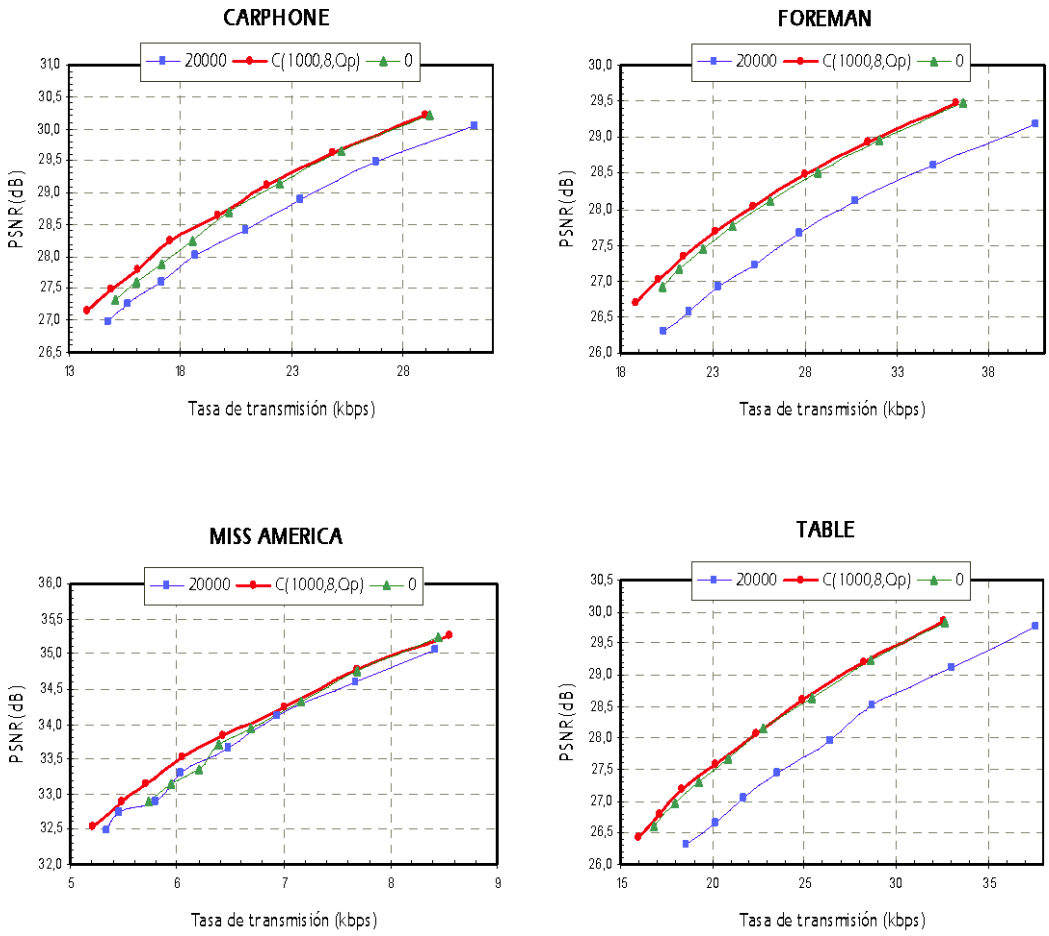


Figura 3.15: Resultados de compresión con ajuste de la constante adaptativa con Qp.

Para ello, y tras realizar numerosas simulaciones para satisfacer la condición impuesta, se fijan los valores $\alpha = 1000$ y $\beta = 8$, obteniéndose las curvas de compresión mostradas en la Figura 3.15. En esta figura se comparan las prestaciones obtenidas en los casos límite $C = 0$ (siempre se elige el vector proporcionado por el algoritmo *FSBM*) y $C = 20000$ (prácticamente igual a elegir siempre el vector proporcionado por el algoritmo *PBM*), con las obtenidas mediante el ajuste de la constante adaptativa con Q_p. Los valores de α y β se han fijado con el objetivo de obtener, para cada valor posible del escalón de cuantificación, un valor de la constante adaptativa que ofrezca prestaciones de compresión óptimas para cualquier secuencia de vídeo según lo establecido en la Figura 3.13. Asimismo, la relación

cuadrática de la constante adaptativa con el escalón de cuantificación permite evitar la rápida pérdida de calidad que se produce en secuencias de vídeo con movimientos caóticos (como es el caso de FOREMAN) para valores altos de la mencionada constante a medida que aumenta el valor de Q_{p*} manteniendo el coste computacional por debajo del exhibido por el algoritmo de búsqueda exhaustiva.

De esta manera, al introducir una constante ajustable se alcanza el objetivo de conseguir unas prestaciones de compresión máximas y constantes, en el sentido de que siempre se alcanzan las prestaciones máximas de compresión si se seleccionan adecuadamente los valores de los parámetros α y β. Este objetivo se obtiene realizando un esfuerzo computacional variable, tal y como se muestra en la Figura 3.16.

A partir del estudio de estas gráficas, se puede observar cómo el algoritmo propuesto cumple con uno de los objetivos de esta Tesis, que no es otro que la doble adaptación del coste computacional a las necesidades de compresión y a las características de la secuencia de vídeo a comprimir:

- **Adaptación a las necesidades de compresión**. Como puede observarse, para cada secuencia individual el coste computacional varía con el valor del escalón de cuantificación, manteniendo siempre unos requisitos de máxima compresión.

- **Adaptación a la secuencia a comprimir**. Para lograr este objetivo, el algoritmo realiza un uso inteligente de los recursos, obteniéndose un coste computacional acorde con las peculiaridades de la secuencia a comprimir.

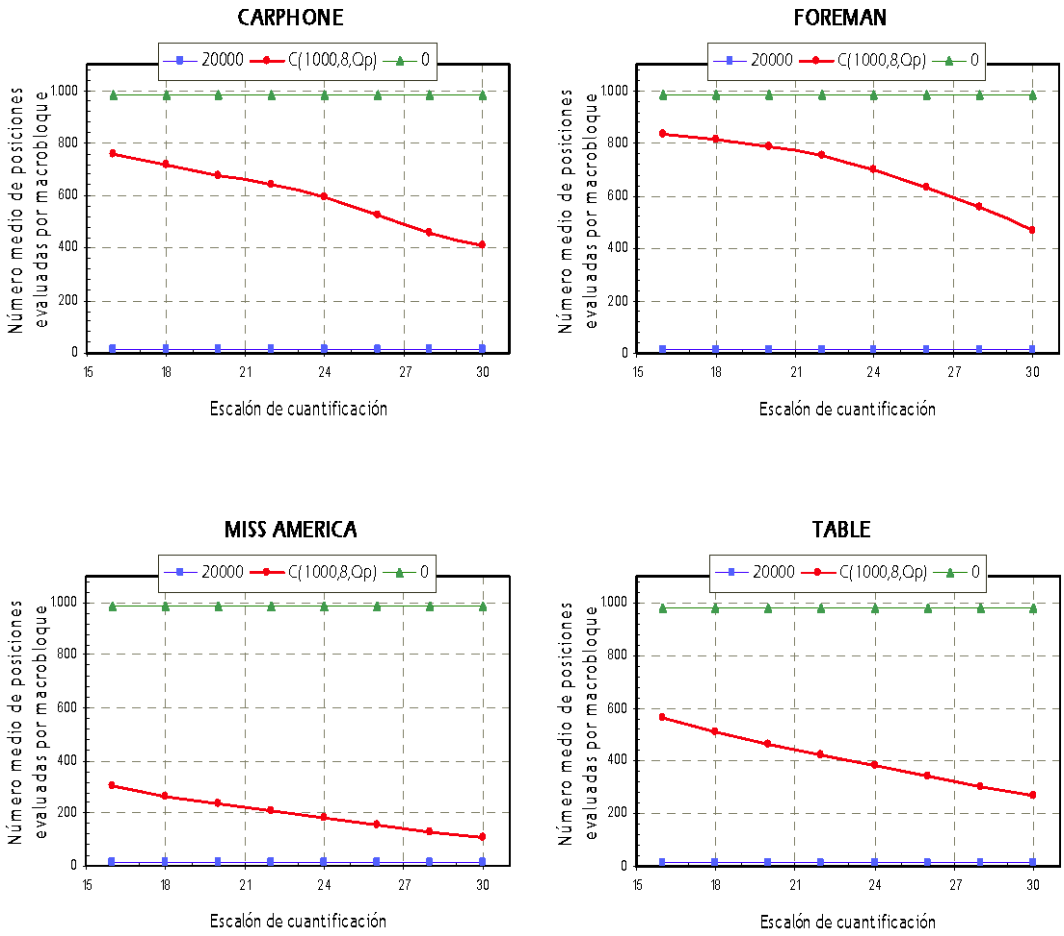


Figura 3.16: Número medio de posiciones evaluadas por macrobloque con ajuste de la constante adaptativa con *Qp*.

Sin embargo, a pesar de que el objetivo de la doble adaptabilidad está cumplido, todavía se puede reducir el coste computacional global de la solución aportada. Para ello, habrá que tener en cuenta el hecho de que en algunas ocasiones el algoritmo PBM proporciona un vector válido aunque no se cumplan las especificaciones establecidas por el criterio de decisión.

3.3.3.2 Ajuste de la estimación de movimiento adaptativa para bloques de alta actividad espacial

En este capítulo ha quedado expuesto que los bloques de alta actividad espacial presentan, por lo general, un alto valor de *SAD_min*. Por lo tanto, para estos bloques es prácticamente imposible que, según el criterio dinámico establecido, el algoritmo PBM encuentre un vector de movimiento válido. Sin embargo, existen determinadas situaciones en las que el vector encontrado por el algoritmo PBM es el mejor posible, aún cuando no se cumple lo especificado en el criterio de decisión.

Esta situación se dará con frecuencia en aquellas secuencias en las que exista un *fondo de imagen* con alta actividad espacial. Dicho de otra manera, esta situación ocurrirá en zonas estáticas de alto *Intra_SAD*, o lo que es lo mismo, macrobloques de textura heterogénea con movimiento cero. Las zonas señaladas en la imagen de la Figura 3.17 representan un claro ejemplo de esta situación.

Figura 3.17: Zonas estáticas de alta actividad espacial en la secuencia DEADLINE.

Por lo tanto, es necesario detectar estas situaciones en las que el algoritmo PBM encuentra un buen vector de movimiento, a pesar de que no se cumple lo especificado en el criterio de decisión. Para ello, se evalúa y se compara el *SAD_PBM* con el *Intra_SAD* del bloque correspondiente, estimándose que el vector obtenido es satisfactorio, en términos de

compresión, si el primero de estos parámetros es significativamente menor que el segundo. Por lo tanto, el esquema de decisión en este caso toma la siguiente forma:

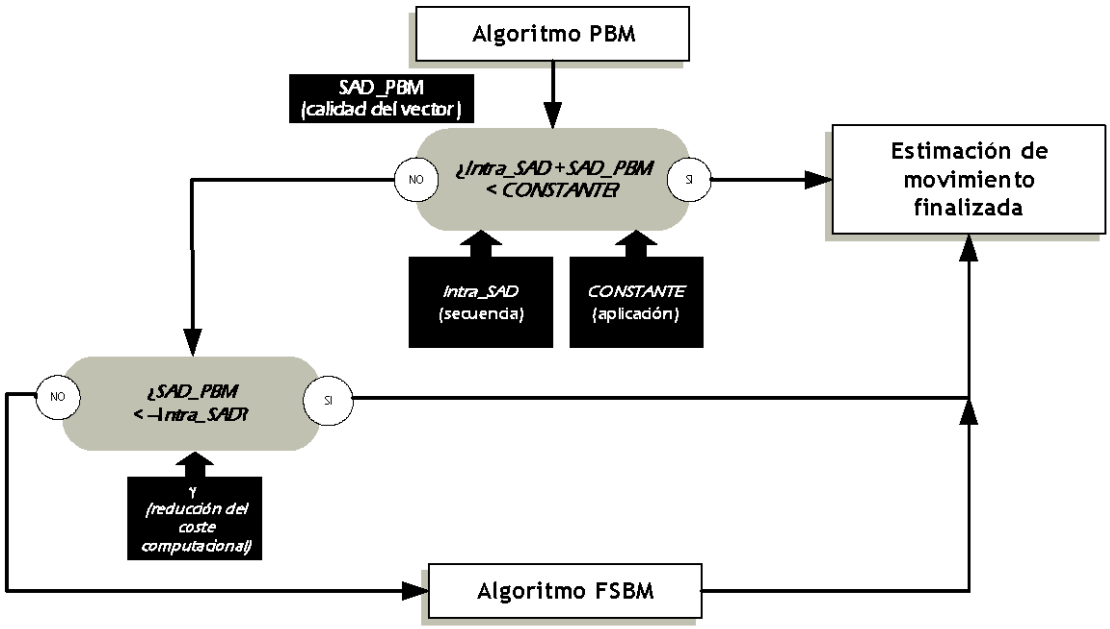


Figura 3.18: Esquema de decisión con ajuste de la constante adaptativa para bloques de alta actividad espacial.

En este sentido, tal y como puede apreciarse en la Figura 3.18, se ha introducido en el criterio de decisión un nuevo parámetro γ que establece cuántas veces ha de ser menor el *SAD_PBM* con respecto al *Intra_SAD* del macrobloque bajo análisis para considerar el vector de movimiento predictivo satisfactorio.

Para evaluar el efecto de la introducción de esta nueva condición, se realizaron simulaciones para diferentes valores de γ con diferentes secuencias. En la Figura 3.19 se muestran los resultados obtenidos con $\gamma = 0, 1/8, 1/4$ y $1/2$.

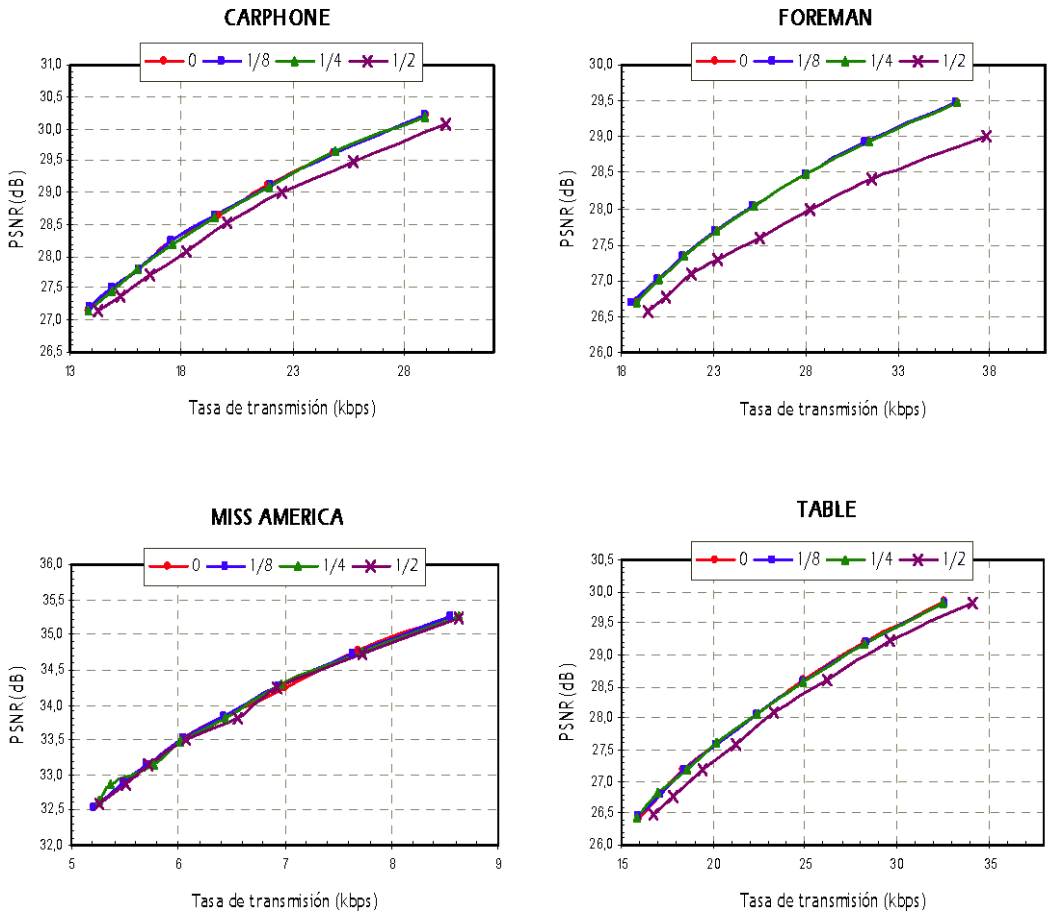


Figura 3.19: Resultados de compresión para diferentes valores de γ.

A partir de estas gráficas se deduce que, si se desean mantener las prestaciones máximas de compresión, el valor máximo de la constante de reducción del coste computacional debe ser $\gamma = 1/4$, pues para valores mayores se obtiene un deterioro de dichas prestaciones en secuencias con alta actividad espacial y/o alto movimiento.

En la Figura 3.20, se muestra la reducción obtenida en el esfuerzo computacional mediante la variación del criterio de decisión para diferentes valores de γ, comprobándose la considerable

reducción del número de posiciones evaluadas por macrobloque para el valor $\gamma = 1/4$ escogido.

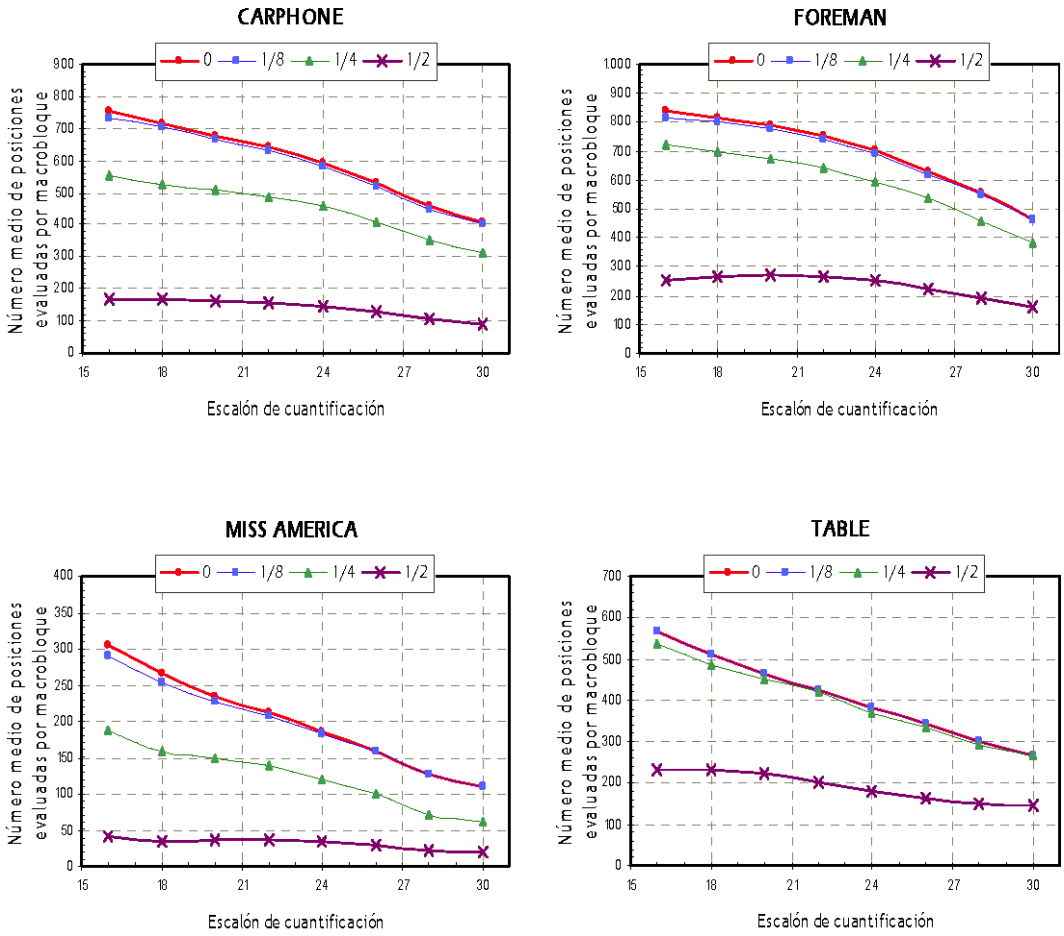


Figura 3.20: Número medio de posiciones evaluadas por macrobloque para diferentes valores de γ.

3.3.4 Evaluación de prestaciones

En este apartado se muestran los resultados finales obtenidos con la solución final adoptada para la estimación de movimiento adaptativa por ajuste de bloques (*Adaptive Cost Block Matching – ACBM*) con los parámetros $\alpha = 1000$, $\beta = 8$ y $\gamma = 0.25$. Para realizar una

correcta caracterización del algoritmo ACBM, se han evaluado sus prestaciones para tasas de muestreo temporal y factores de cuantificación diferentes a los considerados hasta ahora.

3.3.4.1 Prestaciones de codificación para tasas de compresión elevadas y diferentes tasas de muestreo temporal

En este caso se evalúan las prestaciones del algoritmo ACBM para factores de cuantificación comprendidos entre 16 y 30 utilizando las mismas secuencias que hasta ahora, pero con diferentes tasas de muestreo temporal. En particular, se muestran los resultados obtenidos para tasas de muestreo de 30 y 5 fotogramas por segundo – hasta ahora los resultados obtenidos han sido siempre para secuencias muestreadas a 10 fotogramas por segundo -, comparando en ambos casos los resultados con los obtenidos haciendo uso de los algoritmos PBM y FSBM.

Tal y como se advierte a partir de las gráficas de la Figura 3.21, las diferencias entre el algoritmo predictivo seleccionado y el algoritmo propuesto son más evidentes a medida que disminuye la tasa de muestreo. Este hecho es debido a que, en secuencias sub-muestreadas temporalmente, la suposición de que el movimiento es suave pierde validez. Sin embargo, el algoritmo ACBM es capaz de obtener siempre las mejores prestaciones de compresión posibles, independientemente de la tasa de muestreo temporal y sin necesidad de recalcular los valores de los parámetros α, β y γ.

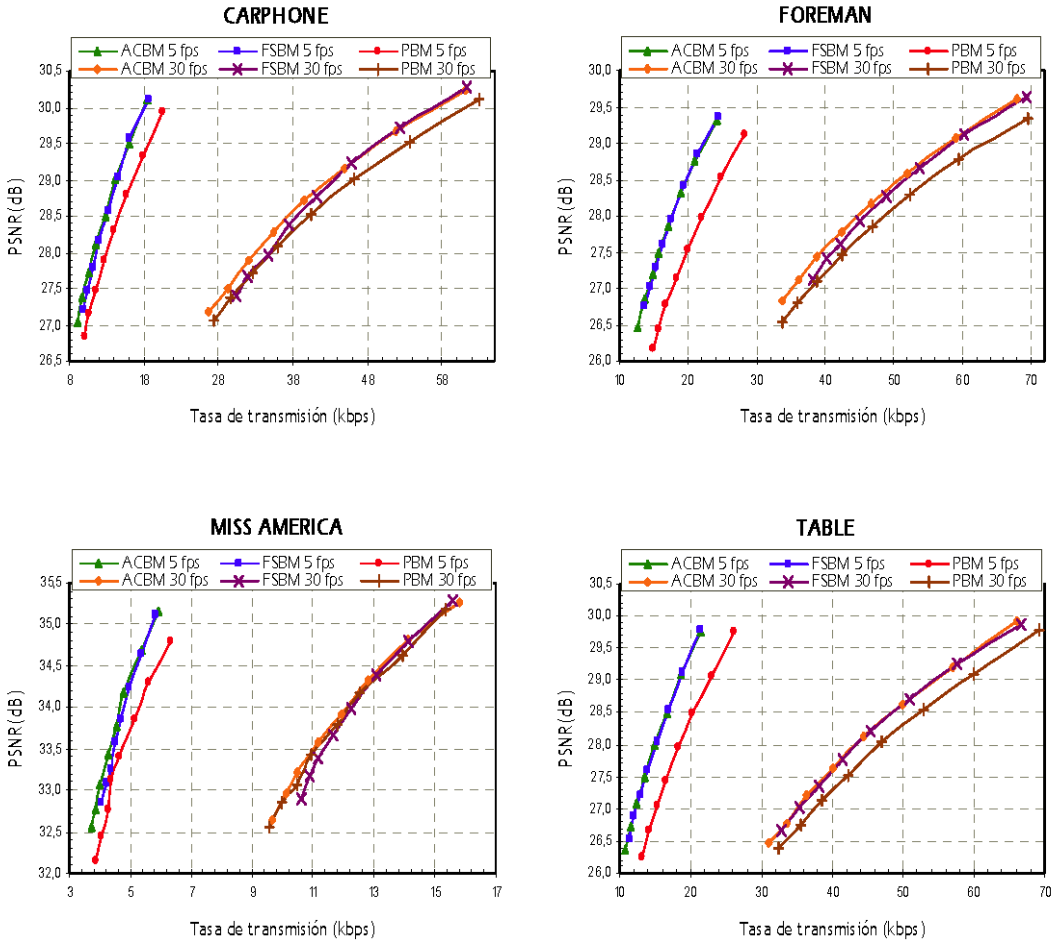


Figura 3.21: Prestaciones de compresión con factores de cuantificación elevados para las tasas de muestreo temporal estudiadas.

3.3.4.2 Prestaciones de codificación para tasas de compresión reducidas y baja tasa de muestreo temporal

En este apartado se muestran los resultados obtenidos para factores de cuantificación comprendidos entre 2 y 14, con el objetivo de verificar las prestaciones obtenidas con el algoritmo propuesto para otros valores del escalón de cuantificación. En la Figura 3.22 se muestran los resultados obtenidos para las mismas secuencias muestreadas a 5 fotogramas por segundo, pues como ha quedado demostrado, es para bajas tasas de muestreo temporal donde se obtienen las peores prestaciones del algoritmo predictivo.

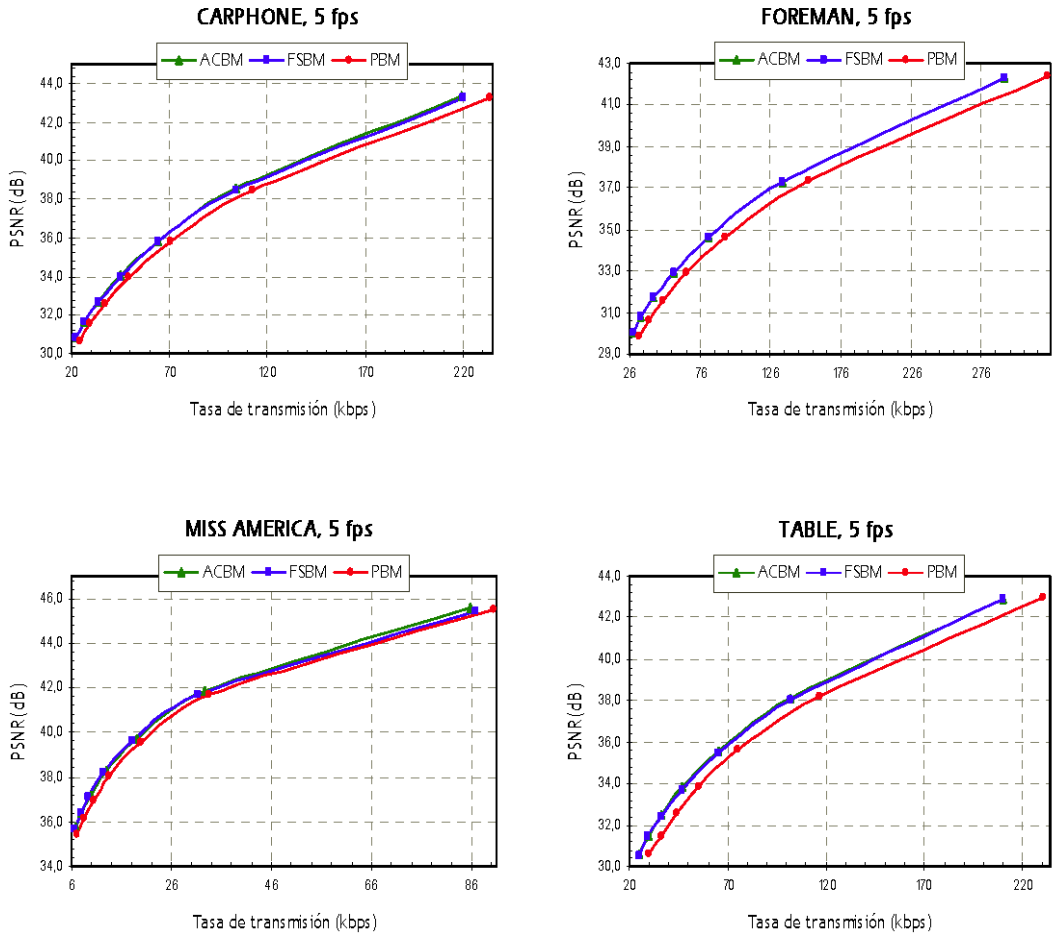


Figura 3.22: Prestaciones de compresión con factores de cuantificación bajos para secuencias muestreadas a 5 fotogramas por segundo.

A partir de estas gráficas, y de los resultados anteriores, se deduce que el algoritmo ACBM es capaz de obtener las mejores prestaciones de compresión independientemente del escalón de cuantificación utilizado. Por lo tanto, la solución final propuesta es capaz de adaptar automáticamente el coste computacional, no sólo a las características espaciales y temporales de la secuencia de vídeo a comprimir, sino también al nivel de compresión a obtener.

3.3.4.3 Coste computacional de la solución propuesta

Por último, en la Figura 3.23 se muestra el coste computacional, en términos del número medio de posiciones evaluadas por macrobloque, asociado a la solución propuesta para los dos casos extremos de muestreo temporal estudiados.

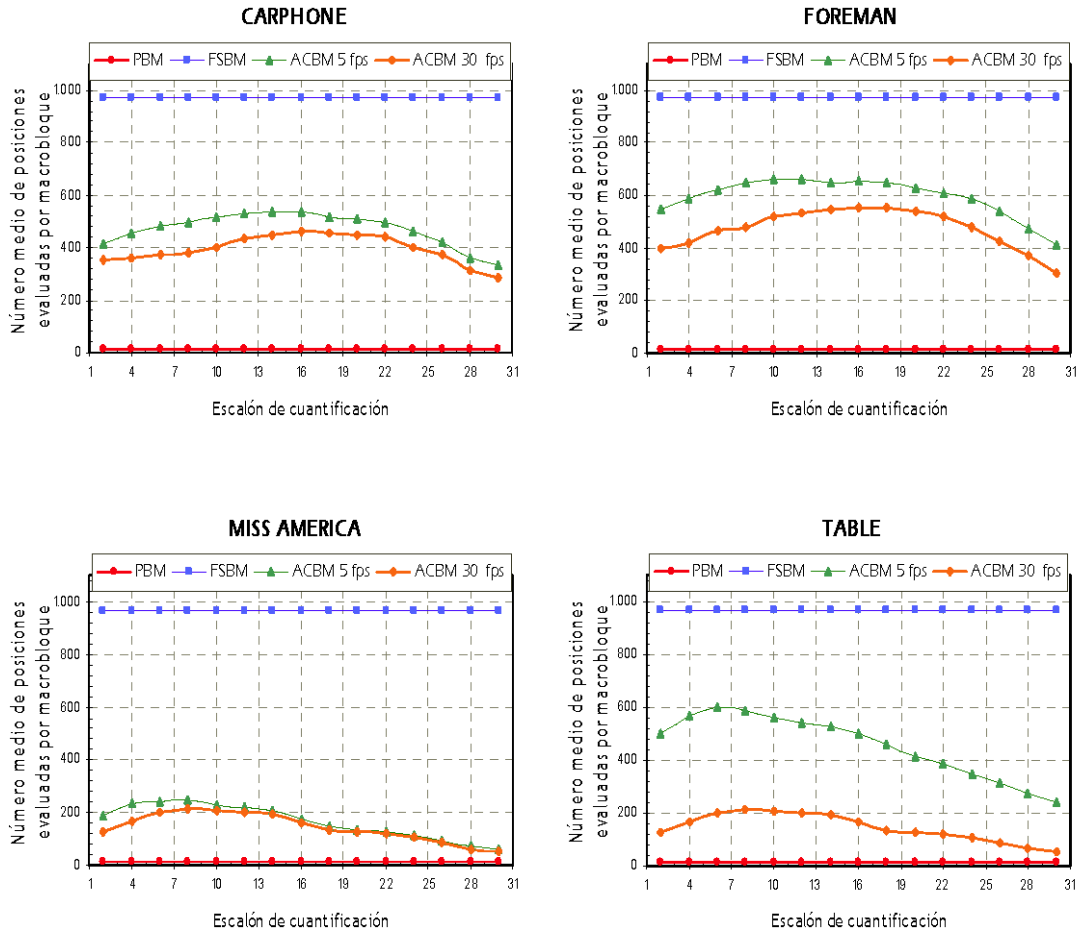


Figura 3.23: Número medio de posiciones evaluadas por macrobloque con criterio de decisión dinámico.

A partir de estas gráficas se extraen las siguientes conclusiones:

- El algoritmo propuesto es capaz de adaptar automáticamente el coste computacional a las características espaciales y temporales de la secuencia a comprimir. En particular, a medida que disminuye la tasa de muestreo, el algoritmo incrementa el esfuerzo computacional debido al empeoramiento de las prestaciones del algoritmo predictivo.

- En todos los casos, el coste computacional es claramente inferior al asociado al algoritmo FSBM. Sin embargo, y tal y como se ha demostrado anteriormente, las prestaciones de compresión obtenidas son ligeramente superiores.

- El coste computacional obtenido no crece monótonamente con el valor del escalón de cuantificación. Este hecho es debido a que, a medida que decrece el nivel de distorsión, los vectores de movimiento óptimos para compresión se asemejan en mayor medida a los vectores que reproducen el movimiento real de la escena y por lo tanto, existe mayor probabilidad de que entre el conjunto de posiciones calculadas por el algoritmo predictivo, exista al menos una satisfactoria.

3.3.5 Reducción del coste hardware asociado al algoritmo ACBM propuesto

El algoritmo ACBM propuesto en esta Tesis representa una estrategia eficaz de estimación de movimiento y, tal como se ha mostrado, permite obtener unas excelentes prestaciones mediante la adaptación del esfuerzo computacional a las características de la secuencia de vídeo y a las necesidades de compresión. Sin embargo, ciertas aplicaciones requieren que el hardware utilizado sea de bajo coste. En este sentido el algoritmo ACBM, propuesto como

una combinación del PBM y del FSBM, presenta un conjunto de características que dificultan la utilización de un hardware reducido. Entre ellas las más importantes son las siguientes:

- **El algoritmo PBM toma los vectores correspondientes a la vecindad espacio-temporal en coordenadas de medio píxel.** Este hecho hace que, cada vez que se calcule el SAD de uno de los candidatos elegidos, sea necesario interpolar la zona a la que dicho vector candidato apunta.

- **El algoritmo PBM evalúa durante la fase de refinamiento posiciones de píxel y medio píxel indistintamente.** Esta característica del algoritmo PBM dificulta en gran medida su implementación, pues en las arquitecturas de estimación de movimiento se realiza de manera separada la búsqueda con precisión entera y la búsqueda sub-píxel, dependiendo esta última del vector de movimiento obtenido con precisión entera.

- **El área de búsqueda, en el caso de que se tenga que ejecutar el algoritmo FSBM, resulta ser de 46×46 píxeles ($p_1 = p_2 = 15$).** Puesto que cada píxel esta codificado con 8 bits, el tamaño de la memoria *cache* necesaria para el almacenamiento del área de búsqueda es de 16928 bits. Además, el número de posiciones a evaluar para cada macrobloque sobre el que se decide ejecutar el algoritmo de búsqueda exhaustiva es de $(p_1 + p_2 + 1)^2$, más 8 posiciones para obtener un vector de coordenadas de medio píxel.

Con el objetivo de reducir el coste de la implementación hardware del algoritmo ACBM, manteniendo la calidad del resultado final, en esta Tesis se proponen realizar las siguientes modificaciones en los algoritmos PBM y FSBM:

- **Se reduce el número de vectores candidatos a evaluar durante la primera fase del algoritmo PBM.** En lugar de elegir cuatro vectores a partir de la vecindad espacio-temporal del macrobloque bajo análisis, se seleccionan sólo tres. En particular, se eligen los vectores de los macrobloques superior e izquierdo dentro del mismo fotograma, así como el vector correspondiente a la misma posición de macrobloque en el fotograma anterior (macrobloques $MB_{up,T}$, $MB_{left,T}$ y $MB_{centre,T-1}$ en la terminología de la Figura 2.12). La elección de estos vectores se debe a que son los que presentan mayor correlación espacio-temporal con el vector de movimiento a calcular [CFP02].

- **Los cuatro vectores candidatos elegidos durante la primera fase del algoritmo PBM, así como las posiciones correspondientes a la segunda fase de refinamiento, han de estar en coordenadas de píxel.** De esta manera, y tras la evaluación de los tres vectores de la vecindad espacio-temporal más el vector cero, se procede a la fase de refinamiento en la forma en la que se indica en la Figura 3.24 (b), sustituyendo la fase original de refinamiento mostrada en la Figura 3.24 (a).

- **El refinamiento a coordenadas sub-píxel es común para ambos algoritmos.** Una vez evaluadas las doce posiciones correspondientes al algoritmo PBM, se determina si el vector de coordenadas de píxel obtenido es satisfactorio. En caso afirmativo, se procede al refinamiento a coordenadas de medio píxel y se concluye la estimación de movimiento. En caso contrario, se realiza la búsqueda exhaustiva con precisión entera, realizándose después el correspondiente refinamiento de medio píxel.

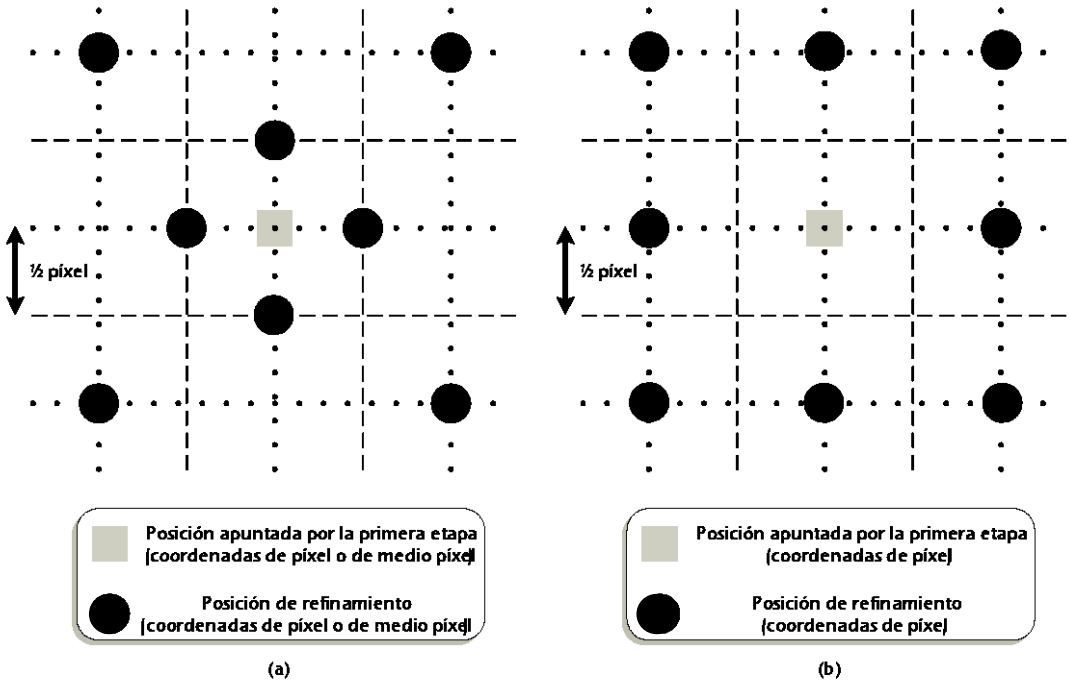


Figura 3.24: Etapa de refinamiento en el algoritmo PBM (a) y propuesta de modificación para reducir el coste hardware asociado (b).

- **En los casos en los que se tenga que ejecutar el algoritmo FSBM, el área de búsqueda se reduce a 31×31 píxeles.** De esta manera, el tamaño de la memoria interna del estimador de movimiento será de 7688 bits y el número de posiciones a evaluar dentro del área de búsqueda será igual a 256.

Este nuevo algoritmo predictivo propuesto, cuyo coste de implementación es significativamente menor que el correspondiente al algoritmo PBM original, se denominará, en el marco de esta Tesis, algoritmo PBM-HW (*Predictive Block Matching – HardWare*). De manera análoga, el algoritmo adaptativo que hace uso de dicho algoritmo predictivo PBM-HW se denominará ACBM-HW (*Adaptive Cost Block Matching – HardWare*).

Llegados a este punto, es necesario evaluar las prestaciones del algoritmo ACBM con las modificaciones introducidas. Dicha evaluación se muestra gráficamente en la Figura 3.25, en la cual se han utilizado las secuencias de estudio muestreadas a 10 fotogramas por segundo,

presentándose también los resultados obtenidos con las dos áreas de búsqueda definidas. En estas gráficas se constata que las modificaciones del algoritmo ACBM no conllevan pérdidas apreciable en las prestaciones de compresión con respecto al original salvo para el caso de la secuencia con mayor cantidad de movimiento, esto es, la secuencia FOREMAN. Sin embargo, y tal y como se puede observar en la gráfica correspondiente a esta secuencia, este empeoramiento está motivado única y exclusivamente por la disminución del tamaño del área de búsqueda. A partir de este resultado, se puede afirmar que las modificaciones introducidas sobre el algoritmo PBM con el objetivo de facilitar su posterior implementación hardware, no conlleva pérdidas apreciables en las prestaciones del algoritmo ACBM final.

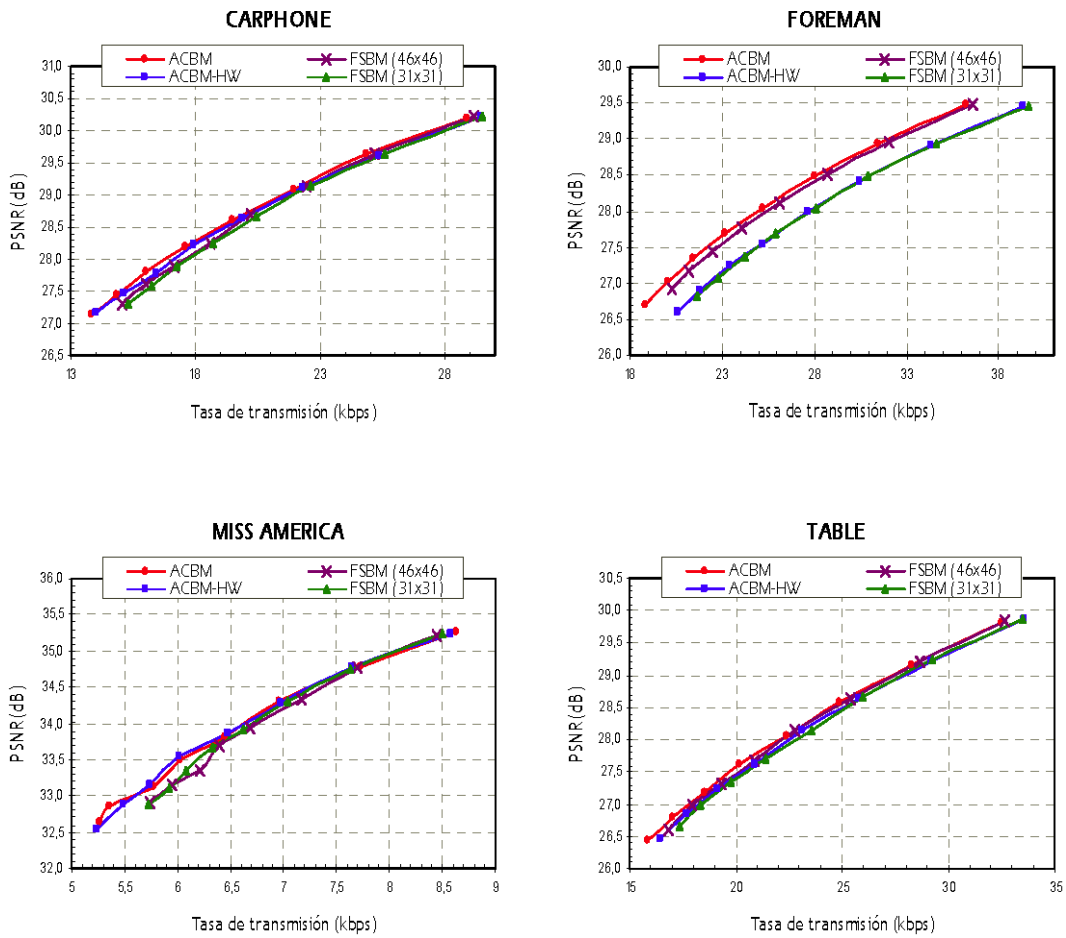


Figura 3.25: Comparación de prestaciones de compresión entre el algoritmo ACBM original y ACBM modificado.

Para completar el análisis efectuado en este apartado, se muestra en la Figura 3.26 el coste computacional de la versión modificada del algoritmo ACBM, comprobándose la considerable reducción en el número medio de posiciones evaluadas.

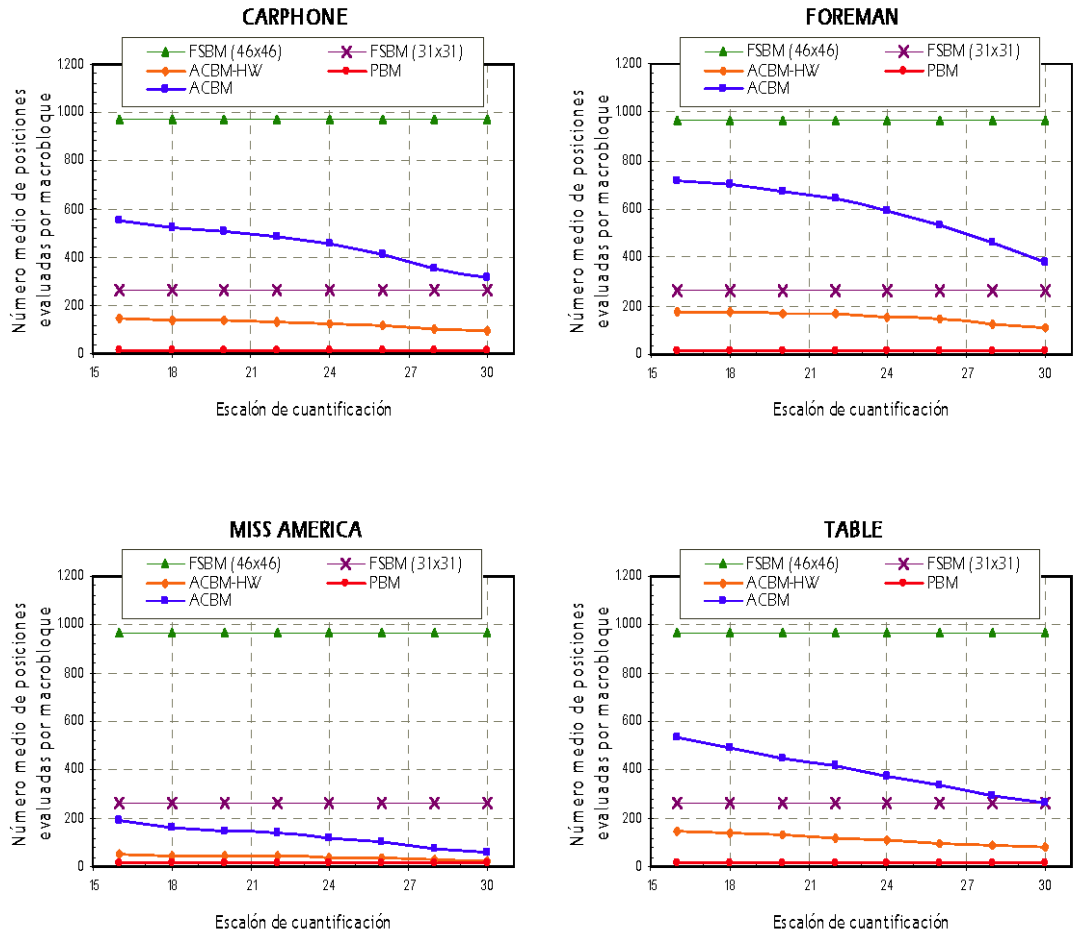


Figura 3.26: Número medio de posiciones evaluadas por macrobloque en la solución final adoptada.

3.4 Estimación de movimiento adaptativa para bloques de topología variable

En el apartado anterior se han mostrado los resultados obtenidos con la estrategia diseñada para estimación de movimiento en macrobloques (16×16 píxeles), usando para su validación el estándar H.263. Sin embargo, tanto en este estándar como en H.264/AVC, se introduce la posibilidad de realizar la estimación de movimiento para diferentes tamaños dentro de un mismo macrobloque (estimación de movimiento multimodo). En particular, el estándar H.263 soporta de manera opcional que un macrobloque pueda codificarse haciendo uso de un solo vector de movimiento o de 4 vectores de movimiento, uno para cada bloque de 8×8 píxeles. Asimismo, tal y como se recoge en el capítulo anterior, el estándar H.264/AVC permite dividir un macrobloque de siete maneras diferentes, de forma que un macrobloque pueda tener asociado un número de vectores de movimiento variable comprendido entre 1 y 16.

En este apartado se presentan las modificaciones realizadas en el algoritmo ACBM con el objetivo de obtener vectores de movimiento para tamaños de bloque variables siguiendo la misma estrategia adaptativa. Igualmente, se presentan los resultados obtenidos con las modificaciones introducidas para el caso de los estándares H.263 y H.264/AVC, utilizando para ello las secuencias habituales de test.

3.4.1 Análisis de la utilización de bloques de tamaño variable en la codificación híbrida de vídeo

En general y de manera muy resumida, el algoritmo ACBM descrito hasta ahora considera que el vector proporcionado por el algoritmo predictivo es satisfactorio cuando se produce alguna de estas dos condiciones:

- **El macrobloque de referencia presenta un alto grado de homogeneidad**. Este hecho se controla dentro del algoritmo mediante la inspección del parámetro *Intra_SAD*, favoreciendo el vector obtenido por el algoritmo predictivo.

- **El macrobloque de referencia presenta un excelente *encaje* con respecto a alguna posición del área de búsqueda**. Para ello, se evalúa el SAD asociado al vector obtenido por el algoritmo PBM (*SAD_PBM*), determinando si éste es lo suficientemente pequeño como para que el vector obtenido se considere satisfactorio en términos de compresión.

Además de estas características, en la adaptación del algoritmo ACBM para su utilización en estimación de movimiento con bloques de tamaño variable, se ha estudiado la frecuencia con la que los estándares H.263 y H.264/AVC utilizan las diferentes topologías de bloque. Para ello, y puesto que el estándar H.264/AVC contempla un mayor número de posibilidades en cuanto a los tamaños y formas de bloque, se ha analizado la probabilidad de que un macrobloque tipo *INTER* sea codificado haciendo uso de un patrón determinado. En particular, se ha utilizado un codificador H.264/AVC en su *perfil básico* (*baseline profile*) con las secuencias de estudio muestreadas a 10 fotogramas por segundo, empleando en la estimación de movimiento el algoritmo de búsqueda exhaustiva con un área de búsqueda de 31×31 píxeles. Los resultados obtenidos se muestran en las gráficas de la Figura 3.27, representándose en el eje izquierdo el porcentaje de macrobloques que han sido codificados con una única partición (16×16) y en el eje derecho la tasa de transmisión obtenida en la codificación de la secuencia con el escalón de cuantificación seleccionado.

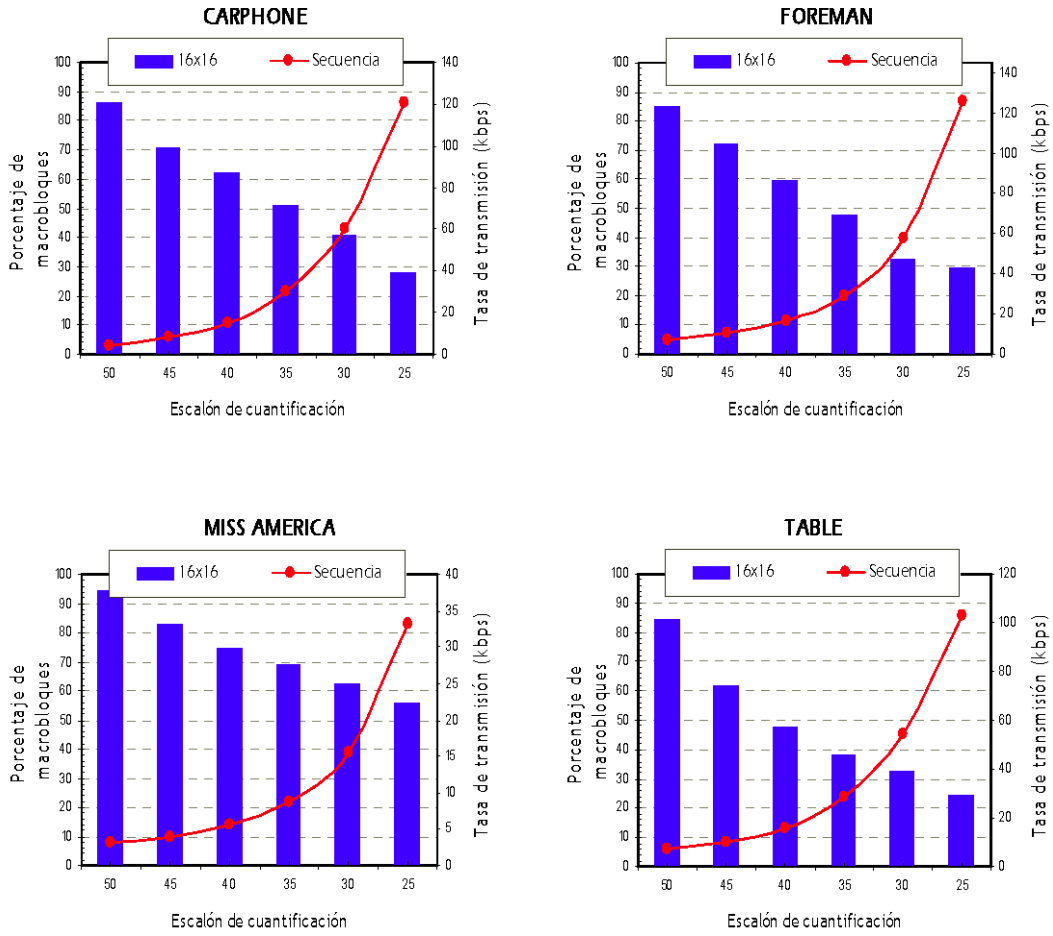


Figura 3.27: Estudio de modos de codificación utilizados por el estándar H.264/AVC.

A partir del estudio realizado se concluye que para tasas bajas de transmisión, el estándar realiza un uso masivo del modo 16×16 que resulta todavía más evidente en secuencias con gran cantidad de bloques homogéneos, como por ejemplo, MISS AMERICA. Además, para todas las secuencias y tasas de transmisión estudiadas, el modo 16×16 resulta ser el más frecuente. Por lo tanto, a mayor tasa de compresión y mayor nivel de homogeneidad en el macrobloque a codificar, mayor será la probabilidad de que el macrobloque sea codificado haciendo uso de un solo vector de movimiento. En gran parte de estos casos, tal y como ha quedado demostrado en este capítulo, el vector 16×16 ofrecido por el algoritmo PBM

proporciona un resultado satisfactorio, y en los que no sea así, lo más probable es que el estándar H.264/AVC elija algún otro modo de codificación.

3.4.2 Propuesta para la utilización de bloques de tamaño variable en el algoritmo ACBM

Sobre la base de estas afirmaciones, el algoritmo ACBM ha sido modificado para su funcionamiento en el proceso de estimación de movimiento con bloques de topología variable, tal y como se resume en la Figura 3.28, en la que se incluyen igualmente los cambios introducidos en el apartado anterior con el objetivo de reducir el coste de implementación del algoritmo. Esta nueva versión del algoritmo, denominada VBS-ACBM (*Variable Block Size – Adaptive Cost Block Matching*), opera por lo tanto según el siguiente esquema:

1. Se calcula un solo vector de movimiento haciendo uso del algoritmo PBM-HW para todo el macrobloque. Este vector de movimiento, según las modificaciones anteriormente introducidas, posee coordenadas de precisión entera (precisión de píxel).

2. Se evalúa si dicho vector es satisfactorio según las necesidades de compresión.

3. En el caso de que sea satisfactorio, se realiza el refinamiento del vector a coordenadas sub-píxel (dependiendo del estándar, medio o cuarto de píxel) y termina la estimación de movimiento.

4. En caso contrario, se evalúa el algoritmo de búsqueda exhaustiva para todos los tamaños y formas de bloque definidos por el estándar en uso, realizándose posteriormente el refinamiento sub-píxel.

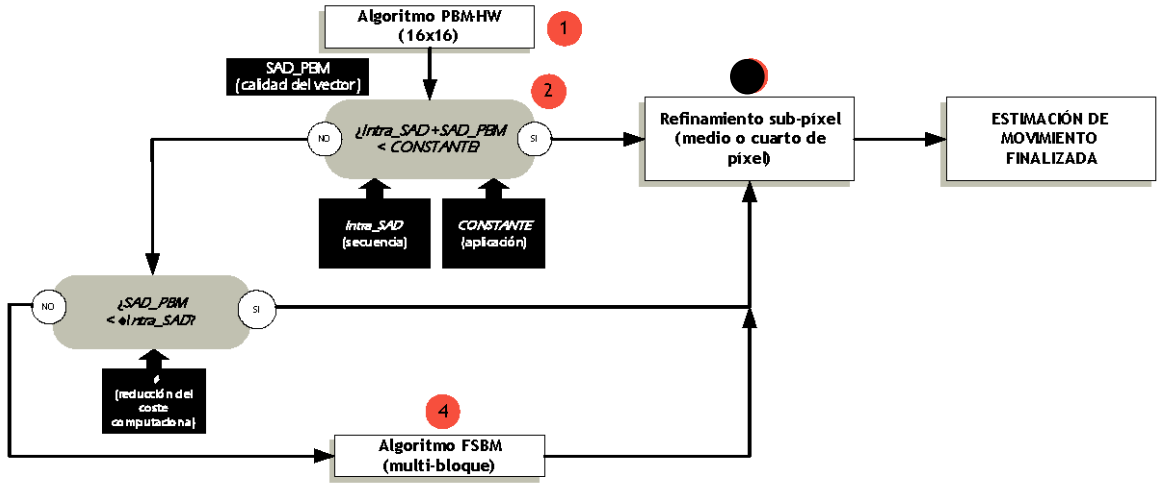


Figura 3.28: Esquema de decisión en el algoritmo VBS-ACBM.

3.4.3 Resultados obtenidos con el algoritmo VBS-ACBM para el estándar H.264/AVC

Con el fin de evaluar las prestaciones del algoritmo VBS-ACBM, se muestran los resultados de compresión/calidad obtenidos con las secuencias de estudio muestreadas a 10 fotogramas por segundo. Hay que destacar que, con el doble objetivo de adaptar el algoritmo a los posibles valores del escalón de cuantificación en el estándar H.264/AVC (de 0 a 51) sin dejar de conseguir unas prestaciones de compresión competitivas con las obtenidas por búsqueda exhaustiva en bloques de tamaño variable, los valores de β y γ se han fijado a 3 y 1/8 respectivamente. Con estos valores, los resultados obtenidos son los que se muestran en la Figura 3.29, en los que se comparan las prestaciones de compresión del algoritmo propuesto con los obtenidos aplicando búsqueda exhaustiva para todos los modos posibles y búsqueda predictiva calculando un vector de movimiento por macrobloque.

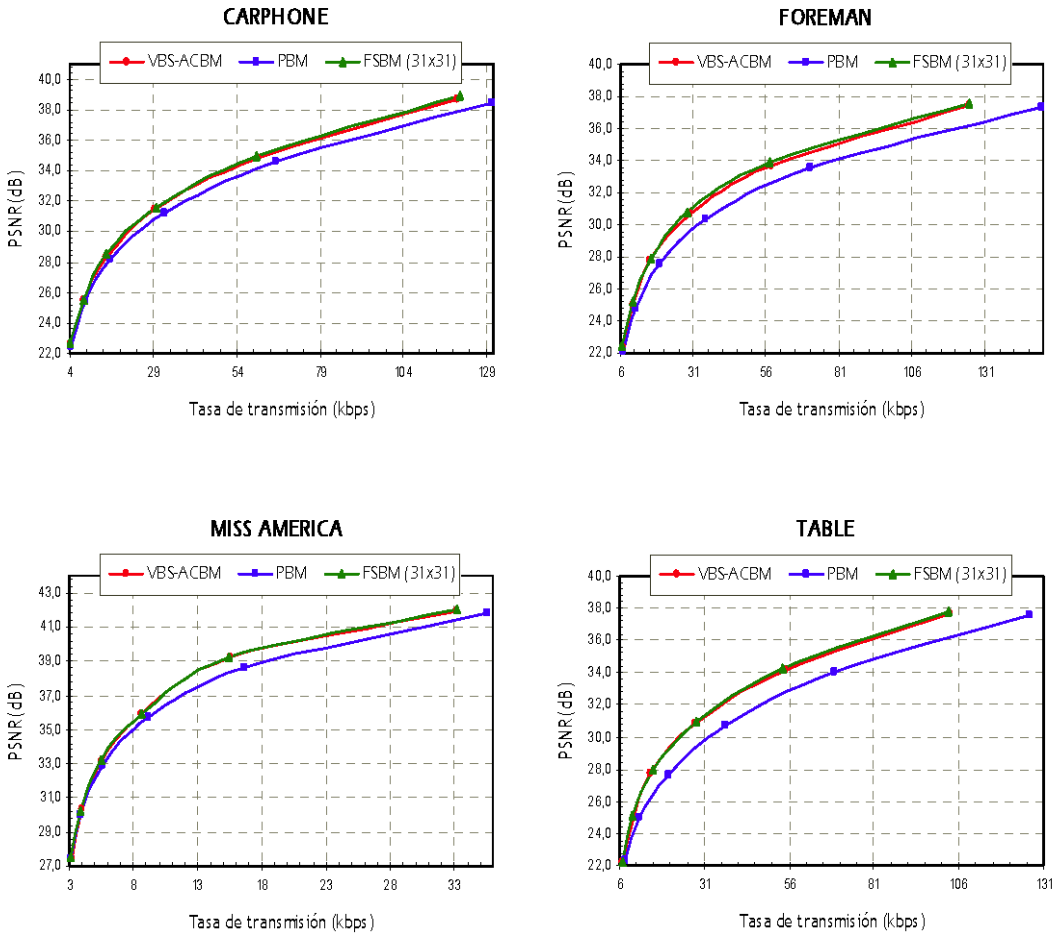


Figura 3.29: Prestaciones de compresión obtenidas con el algoritmo VBS-ACBM para el estándar H.264/AVC.

A partir de estos resultados se verifica que el algoritmo VBS-ACBM es capaz de obtener unas prestaciones de compresión idénticas a las obtenidas por el algoritmo FSBM con todos los modos posibles de estimación de movimiento activados, independientemente de las características de la secuencia de vídeo a comprimir. Sin embargo, tal y como se demuestra en la Figura 3.30, este objetivo se consigue con una importante reducción en el coste computacional gracias a la estrategia de adaptación seguida.

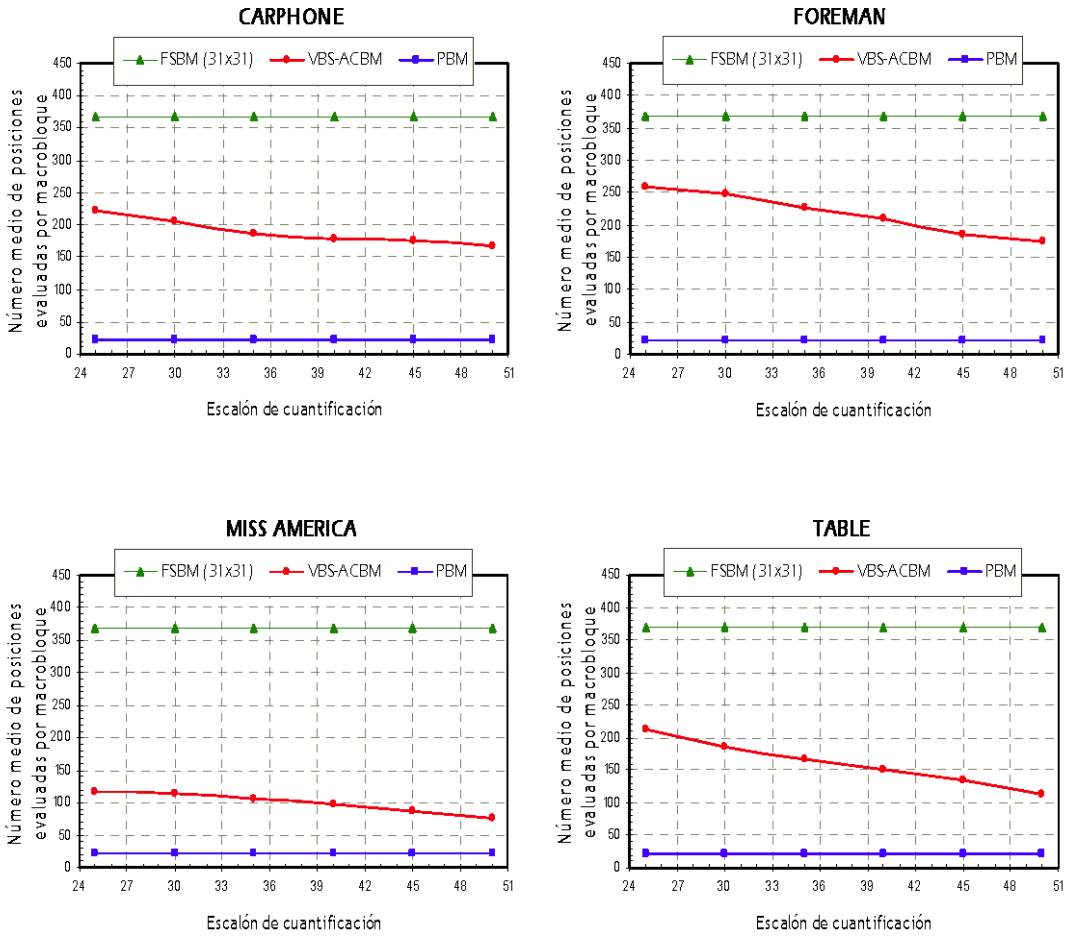


Figura 3.30: Número medio de posiciones evaluadas por macrobloque con el algoritmo VBS-ACBM para el estándar H.264/AVC.

Por último, es necesario explicar dos aspectos para la correcta comprensión de estas gráficas. En primer lugar, el algoritmo PBM evalúa ahora 21 posiciones, 8 más que en el caso anterior. Este aumento en el número de posiciones evaluadas es debido a que el estándar H.264/AVC trabaja con vectores de precisión de cuarto de píxel, y por lo tanto, se necesitan 8 posiciones más para el refinamiento de los vectores de movimiento. En segundo lugar, el algoritmo FSBM, en su primera tarea de búsqueda de vectores de movimiento con precisión de píxel, continúa evaluando 256 posiciones a pesar de que sea necesario calcular un mayor número de vectores de movimiento. Este hecho es posible debido a que se reutilizan los cálculos

realizados en el cómputo de los vectores de movimiento de los bloques de menor tamaño para construir los vectores de movimiento de los modos superiores. Sin embargo, en las etapas de refinamiento de medio y cuarto de píxel de los vectores de movimiento esta reutilización no es posible. Por ello, es necesario refinar todos los vectores de manera independiente, resultando que para cada modo posible de estimación de movimiento hay que evaluar el equivalente a 16 posiciones de macrobloque con el objetivo de obtener vectores de movimiento con precisión de cuarto de píxel.

3.5 Conclusiones

En este capítulo se ha presentado un novedoso algoritmo adaptativo de estimación de movimiento denominado ACBM (*Adaptive Cost Block Matching*) que combina los beneficios aportados por el algoritmo de búsqueda exhaustiva FSBM (*Full Search Block Matching)* y el algoritmo predictivo PBM (*Predictive Block Matching*). Dicho algoritmo es capaz de obtener, como mínimo, las mismas prestaciones que el algoritmo de búsqueda exhaustiva realizando un esfuerzo computacional considerablemente menor, adaptándose a las características espaciales y temporales de la secuencia de vídeo a comprimir, e independiente de la tasa de transmisión requerida y/o el estándar utilizado.

Para definir la estrategia de adaptación con éxito, se ha desarrollado un entorno de simulación que ha permitido, mediante el estudio de los parámetros *Intra_SAD* y *SAD_Deviation*, analizar con detalle el proceso de estimación de movimiento desde el punto de vista de la función de coste de Lagrange J_{motion}. Este análisis ha facilitado la identificación, en términos de compresión y coste computacional, de dos tipos de macrobloques críticos. En primer lugar, los macrobloques de alta actividad espacial, en los que es necesario evaluar un gran número de posiciones dentro del área de búsqueda para evitar un notable aumento del SAD, y en consecuencia, del valor de la función J_{motion}. En segundo lugar, los bloques de baja

actividad espacial, en los que sólo es necesario evaluar un número reducido de candidatos, siempre y cuando se garantice que el vector de movimiento resultante sea coherente con respecto a los de los macrobloques vecinos, pues de lo contrario, se incrementaría el número de bits R_{motion}.

A raíz de las conclusiones extraídas, se ha propuesto una etapa de post-procesamiento de vectores de movimiento que permite aumentar ligeramente las prestaciones de compresión obtenidas mediante el algoritmo de búsqueda exhaustiva para cualquier tipo de secuencia. Esta etapa de post-procesamiento resulta de utilidad en estimadores de movimiento por búsqueda exhaustiva, pues permite obtener un conjunto de vectores de movimiento que reproducen de manera más fidedigna el movimiento real en la escena, corrigiendo los defectos del algoritmo exhaustivo en zonas de baja actividad espacial. De esta manera, se obtienen beneficios en aplicaciones complementarias al proceso de decodificación, tales como corrección de errores, transcodificación, o aumento de la resolución espacio-temporal de la secuencia de vídeo comprimida.

Con el objetivo de obtener una solución de coste computacional reducido, se ha propuesto el algoritmo ACBM. En este algoritmo se introduce una constante adaptativa que, al depender directamente del valor del escalón de cuantificación Q_p, permite establecer un compromiso entre el coste computacional y el nivel de compresión. Los resultados obtenidos demuestran que el algoritmo ACBM, evaluando un número de posiciones significativamente menor, es capaz de obtener unas prestaciones de compresión iguales a las ofrecidas por el algoritmo de búsqueda exhaustiva para todo el rango de valores posibles de Q_p con secuencias de muy diferentes características espaciales y tasas de muestreo temporal (desde 5 hasta 30 fotogramas por segundo). Asimismo, se han introducido un conjunto de modificaciones en el algoritmo ACBM que facilitarán en gran medida su posterior implementación hardware, sin que esto conlleve una degradación de sus prestaciones. Dicha implementación es abordada en

el siguiente capítulo, introduciéndose durante su desarrollo las aportaciones arquitecturales de esta Tesis.

Por último, se ha propuesto un nuevo algoritmo denominado VBS-ACBM (*Variable Block Size – Adaptive Cost Block Matching*) para estándares con estimación de movimiento con bloques de tamaño variable, incluyendo el estándar H.264/AVC con sus siete modos de estimación de movimiento. Al conservar las características del algoritmo ACBM anteriormente mencionadas, el nuevo algoritmo VBS-ACBM representa una importante contribución en el área de la estimación de movimiento adaptativa, al eliminar las deficiencias encontradas en trabajos previos garantizando, a la misma vez, unas prestaciones de compresión óptimas para todo tipo de secuencias.

CAPÍTULO 4

Arquitecturas multiestándar de estimación de movimiento adaptativa

En el capítulo anterior se ha demostrado que el algoritmo VBS-ACBM aportado representa, mediante la adaptación del esfuerzo computacional a las características espacio-temporales de la secuencia a comprimir, una eficaz solución para realizar la estimación de movimiento en codificadores híbridos que permitan el uso de vectores de movimiento multimodo. Sin embargo, para su utilización en aplicaciones con restricciones de funcionamiento en tiempo real, es necesario aportar nuevas soluciones arquitecturales adecuadas a la implementación del proceso de estimación de movimiento según las directrices del algoritmo VBS-ACBM de manera eficiente.

En este capítulo se proponen nuevas soluciones arquitecturales adaptadas al algoritmo VBS-ACBM mediante la división del cálculo de vectores de movimiento en dos procesos: estimación de movimiento con precisión entera y posterior refinamiento sub-píxel. Así, en esta Tesis se propone una nueva arquitectura unidimensional agrupada de estimación de movimiento con precisión entera mediante búsqueda predictiva/exhaustiva, que permite obtener los vectores de movimiento correspondientes a un macrobloque de acuerdo al estándar H.264/AVC y, por defecto, los vectores demandados por cualquiera de los estándares anteriores. Asimismo, se propone en esta Tesis una nueva arquitectura unidimensional no agrupada de refinamiento sub-píxel. Dicha arquitectura permite calcular los vectores de movimiento de acuerdo a lo establecido por los estándares H.263 y H.264/AVC, refinando en cada caso, bien 5 vectores de movimiento a coordenadas de medio píxel mediante interpolación bilineal, o bien 41 vectores de movimiento a coordenadas de cuarto de píxel mediante ecuaciones específicas de interpolación, respectivamente. Para ambas arquitecturas propuestas se obtienen mejoras significativas con respecto a trabajos predecesores recientemente publicados.

4.1 Introducción

Durante los últimos años se han propuesto numerosas arquitecturas de estimación de movimiento que persiguen, como objetivo principal, calcular en tiempo real los vectores de movimiento correspondientes a una determinada secuencia de vídeo bajo los requisitos establecidos por un estándar de compresión específico. Para ello, diversos autores han propuesto el uso de arquitecturas sistólicas, tanto para la obtención de vectores de movimiento de precisión entera (precisión de píxel) como para el posterior refinamiento de éstos a coordenadas sub-píxel (precisión de medio píxel y, en menor medida, cuarto de píxel). Estas arquitecturas han sido analizadas en el capítulo 2 de esta Tesis, detallándose con especial énfasis las deficiencias encontradas en arquitecturas adaptativas y/o capaces de cumplir con los estrictos requerimientos del estándar H.264/AVC.

La propuesta arquitectural aportada en esta Tesis realiza el proceso de estimación de movimiento según las directrices definidas por el algoritmo VBS-ACBM expuesto en el capítulo anterior. Esta arquitectura es capaz de obtener vectores de movimiento a partir de esquemas predictivos y/o mediante búsqueda exhaustiva con precisión de medio y cuarto de píxel. Asimismo, en el caso de aplicarse búsqueda exhaustiva, la arquitectura propuesta permite obtener vectores de movimiento para cualquiera de los siete modos de estimación de movimiento definidos por el estándar H.264/AVC, dentro de los cuales se encuentran los modos específicos de cualquier estándar anterior a éste, lo que convierte la arquitectura propuesta en multiestándar. Para lograr este objetivo de manera eficiente, y con el objetivo de cubrir las carencias detectadas en trabajos previos, se ha estudiado en esta Tesis un conjunto de novedosas estrategias independientes de la capacidad de adaptación ofrecida por el algoritmo VBS-ACBM. Dichas estrategias, cuya incorporación en la arquitectura propuesta en esta Tesis se describirá en detalle durante este capítulo, se detallan a continuación:

A) Mecanismos de eliminación temprana de candidatos en arquitecturas multimodo. El proceso del cálculo del SAD correspondiente a un determinado vector de movimiento puede ser detenido si el valor acumulado hasta ese momento es mayor que el SAD mínimo provisional. Los beneficios obtenidos mediante esta técnica, así como sus implicaciones arquitecturales, han sido estudiados por otros autores para arquitecturas basadas en estándares de topología fija [DY98], [Sou99], [SR99], [LTC04], [DRS05].

En arquitecturas de estimación de movimiento de precisión entera para topología de macrobloque se obtiene una reducción media en el consumo de potencia de entre un 50% y un 65% en arquitecturas sistólicas unidimensionales [SR99], [LTC04] y de entre un 20% y un 40% en arquitecturas sistólicas bidimensionales [DY98], [Sou99], mientras que para las arquitecturas típicas de refinamiento de medio píxel el porcentaje de reducción es de un 20% [DRS05]. Sin embargo, y a pesar de la decidida apuesta dentro de los estándares de compresión de vídeo por el uso de etapas de estimación de movimiento de topología de bloque variable, no existe ningún trabajo en la bibliografía disponible acerca del uso de los mencionados mecanismos de eliminación de candidatos en arquitecturas multimodo.

En este capítulo se propone una arquitectura de estimación de movimiento con capacidad de eliminación temprana de candidatos tanto durante la estimación con precisión entera como en el posterior refinamiento a coordenadas sub-píxel, respetando los requisitos establecidos por los estándares H.263 y H.264/AVC en cuanto al uso de bloques de tamaño variable y precisión de los vectores. Asimismo, se investiga por primera vez si las técnicas de eliminación temprana de candidatos son igualmente efectivas, en términos de ciclos inactivos y coste hardware asociado, para arquitecturas de estimación de movimiento con topología variable.

B) Elección flexible del número de modos y precisión de los vectores de movimiento por macrobloque. Los estándares de compresión de vídeo definen un número máximo de modos de estimación de movimiento, así como una precisión máxima para los vectores de

movimiento de cada macrobloque. Mediante el estudio de las características de la secuencia de vídeo a codificar puede reducirse en el codificador el número de modos a evaluar, así como la precisión de los vectores de movimiento, sin pérdidas apreciables en los niveles de compresión, tal y como demuestran los trabajos referenciados en la parte final del apartado 2.3.2.2 de esta Tesis. Este hecho no es considerado por las arquitecturas de estimación de movimiento publicadas hasta la fecha, en las cuales siempre se realiza la estimación de movimiento para un número de modos y una precisión en los vectores de movimiento constantes.

En este capítulo se investiga, sobre la base de la arquitectura propuesta, los beneficios arquitecturales derivados de considerar una total flexibilidad en el número de modos de estimación de movimiento y en la precisión de los vectores de movimiento por macrobloque.

C) Agrupación de elementos de proceso. En los trabajos previos a esta Tesis no se han estudiado las implicaciones que resultan de la manera en que se agrupan los elementos de proceso en una arquitectura de estimación de movimiento.

En este capítulo se estudian dichos efectos en términos arquitecturales que, como consecuencia de lo establecido en los dos puntos anteriores, resultan ser totalmente novedosos en el ámbito de las arquitecturas de estimación de movimiento multimodo.

A lo largo del presente capítulo se detallan los estudios realizados en relación con estos aspectos, tanto en el proceso de estimación de movimiento con precisión entera como en el de refinamiento de vectores de movimiento a coordenadas sub-píxel, mostrándose las contribuciones propias de esta Tesis dentro de ambos ámbitos.

4.2 Características generales de la arquitectura propuesta

La estructura a nivel de módulos funcionales de la arquitectura propuesta en esta Tesis Doctoral para la estimación de movimiento según el algoritmo VBS-ACBM se muestra de forma esquemática en la Figura 4.1.

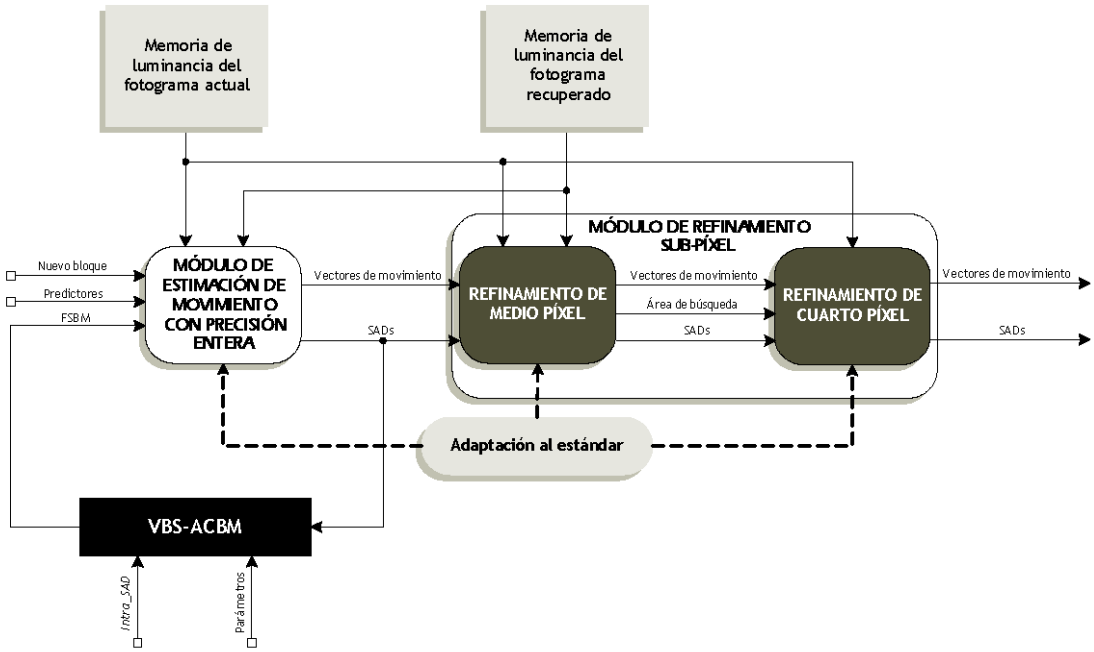


Figura 4.1: Esquema general de la arquitectura propuesta para la estimación de movimiento según el algoritmo VBS-ACBM.

Tal y como puede deducirse a partir de la Figura 4.1, la arquitectura propuesta consta de dos módulos arquitecturales claramente diferenciados:

1. **Módulo de estimación de movimiento con precisión entera**. El módulo de estimación de movimiento con precisión entera propuesto en esta Tesis calcula los vectores de movimiento con precisión entera mediante búsqueda predictiva o exhaustiva. El tipo de búsqueda dependerá de si la estimación se está realizando por primera vez para un nuevo macrobloque o, si por el contrario, ya se ha realizado una primera estimación predictiva pero el vector obtenido no es satisfactorio en los términos establecidos por

el algoritmo VBS-ACBM propuesto. En el caso de realizarse la búsqueda exhaustiva, este proceso se realizará sólo para aquellos modos que sea necesario y que en el contexto de esta Tesis se denominan ***modos activos***. Esta condición, tal y como se indicó anteriormente, representa una novedad con respecto a las arquitecturas de búsqueda exhaustiva precedentes y ha sido introducida con el objetivo de estudiar los posibles beneficios arquitecturales asociados a ella, así como proporcionar una adaptación total de la arquitectura al estándar en uso. En cualquier caso, este módulo es capaz de obtener los vectores de movimiento para los siete modos definidos por el estándar H.264/AVC y por lo tanto, para cualquiera de sus estándares predecesores.

2. **Módulo de refinamiento sub-píxel**. A partir de las coordenadas de los vectores de movimiento con precisión de píxel calculados por el módulo de estimación de movimiento con precisión entera, el módulo de refinamiento sub-píxel se encarga de refinar dichos vectores, obteniendo el mismo número de vectores en coordenadas de cuarto de píxel. Es importante destacar que, tomando como referencia el estándar H.264/AVC, el refinamiento de vectores de movimiento a coordenadas de cuarto de píxel supone un 45% del esfuerzo computacional medio del proceso de estimación de movimiento, proporcionando una mejora en la calidad de la secuencia de 4 dBs de media para una misma tasa de transmisión [CHC04b]. Sin embargo, y a pesar de estos relevantes números, las aportaciones algorítmicas y arquitecturales realizadas en el marco del refinamiento de vectores de movimiento en la bibliografía reciente son prácticamente inexistentes cuando se comparan con los trabajos relacionados con la estimación de movimiento con precisión entera. Para realizar este proceso de manera eficiente, el módulo de refinamiento propuesto en esta Tesis está compuesto por dos sub-módulos:

- **Sub-módulo de refinamiento de medio píxel**. A partir de las posiciones apuntadas por los vectores de movimiento proporcionados por el módulo de estimación de movimiento con precisión entera, este sub-módulo realiza el refinamiento de los vectores recibidos a coordenadas de medio píxel. Para ello, después de completar una primera etapa en la que se calculan los valores de las posiciones de medio píxel necesarias, se realiza el refinamiento propiamente dicho mediante la inspección de ocho posiciones de medio píxel situadas alrededor de la posición con precisión entera apuntada. De nuevo, este proceso sólo se realizará para aquellos modos que hayan sido activados, de acuerdo a los vectores calculados por el módulo de estimación de movimiento con precisión entera. Asimismo, este proceso sólo se realizará si el estándar lo requiere, o lo que es lo mismo, si permite realizar la compensación de movimiento en coordenadas de medio píxel.
- **Sub-módulo de refinamiento de cuarto píxel.** Este sub-módulo es análogo al de refinamiento de medio píxel, con la salvedad de que el refinamiento se produce mediante la inspección de ocho posiciones de cuarto de píxel alrededor del conjunto de vectores de medio píxel.

Tal y como se puede deducir a partir de las características descritas, de las tres grandes novedades introducidas por el estándar H.264/AVC en cuanto al proceso de estimación de movimiento se refiere [WSB+03], la arquitectura propuesta en esta Tesis aborda de manera implícita dos de ellas - estimación de movimiento para bloques de topología variable y precisión de hasta cuarto de píxel -, no abordando la estimación de movimiento multifotograma. Este hecho se debe fundamentalmente a las razones siguientes:

En primer lugar, la arquitectura propuesta es absolutamente *transparente* con respecto a los fotogramas utilizados para la estimación de movimiento. Tal y como se detallará posteriormente, la arquitectura completa el proceso de estimación de movimiento a partir de

los datos que se encuentran en sus memorias *cache*, independientemente de los fotogramas de los que proceden dichos datos.

En segundo lugar, recientes estudios han demostrado que las mejoras introducidas al considerar la posibilidad de estimar el movimiento teniendo en cuenta varios fotogramas de búsqueda son apenas apreciables, particularmente en el caso de elevados factores de compresión [ABP+03], pudiéndose provocar incluso un empeoramiento de las prestaciones del codificador [ABC+03]. Asimismo, existen numerosos trabajos en los que, a partir de estas mismas observaciones, se concluye que la estimación de movimiento multifotograma puede ser obviada en la mayor parte de los macrobloques de una secuencia cualquiera [CCL+04], [SS04], [HHC+06].

Por último, su coste hardware asociado es extremadamente alto al incrementar, tanto el coste computacional del proceso de estimación de movimiento como, sobre todo, el área final del chip.

En los sucesivos apartados del presente capítulo se analizan las características de cada uno de los módulos definidos, quedando patente la incorporación del conjunto de estrategias anteriormente mencionadas, así como las aportaciones más destacadas en el ámbito de la Tesis.

4.3 Estimación de movimiento con precisión entera

En este apartado se comenzará con una descripción de la arquitectura propuesta y del funcionamiento del módulo de estimación de movimiento con precisión entera para el caso en el que el cálculo de los vectores de movimiento se realice aplicando el algoritmo de búsqueda

exhaustiva. Posteriormente se describirán los cambios a introducir para la incorporación del algoritmo predictivo, ya que éste se puede considerar como un caso particular (de complejidad reducida), contemplado por el caso más general (búsqueda exhaustiva).

En esta Tesis se propone para la implementación del proceso de estimación de movimiento con precisión entera una arquitectura sistólica unidimensional y parametrizable compuesta por N grupos de M elementos de proceso cada uno. Dicha arquitectura está basada en trabajos realizados por el autor de esta Tesis Doctoral para estándares con un solo vector de movimiento por macrobloque, los cuales han sido publicados en la revista internacional *Elsevier Microelectronics Journal* [LCL+02] y en los congresos internacionales *SPIE International Symposium on Smart Electronics and MEMS* [LCL+01] y *The 28th Annual Conference of the IEEE Industrial Electronics Society (IECON 2002)* [LTL+02]. Esta arquitectura realiza el proceso de estimación de movimiento con precisión entera según los requerimientos del estándar H.264/AVC incorporando, como estrategias novedosas, mecanismos de eliminación temprana de candidatos y selección de modos activos. En la Figura 4.2 se muestra la estructura a nivel de bloques funcionales del módulo de estimación de movimiento con precisión entera.

Cada uno de los N grupos de elementos de proceso de esta arquitectura, evalúa una posición de macrobloque dentro del área de búsqueda, colaborando los M elementos de proceso PE_i^j ($1 \leq i \leq M$ y $1 \leq j \leq N$) en esta tarea. A partir de cada una de las posiciones de macrobloque evaluadas, la arquitectura es capaz de calcular los 41 vectores de movimientos por macrobloque definidos por el estándar H.264/AVC. Para ello, tanto el macrobloque de referencia como el macrobloque correspondiente a la región del área de búsqueda bajo evaluación, se dividen en 16 bloques de 4×4 píxeles, obteniéndose en primer lugar los SADs correspondientes a este tamaño de bloque.

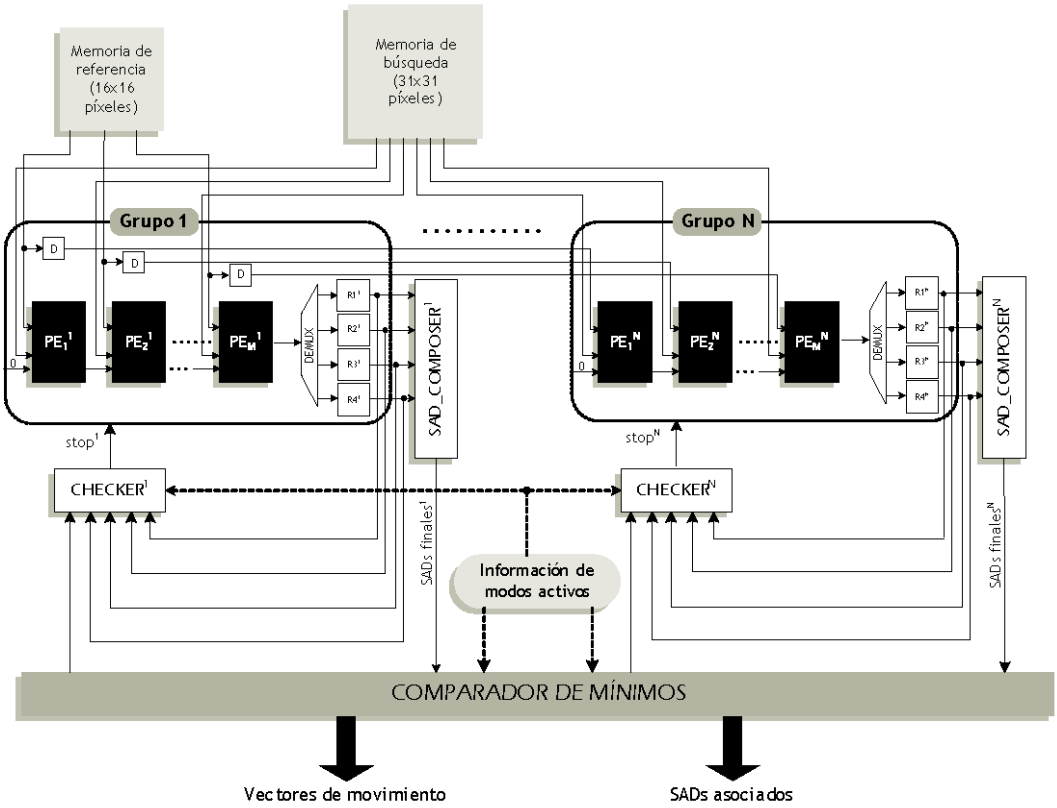


Figura 4.2: Esquema general de la arquitectura propuesta para el módulo de estimación de movimiento con precisión entera.

A partir de los resultados obtenidos para los bloques de 4×4 píxeles, el bloque funcional asociado a cada grupo, denominado en esta Tesis como SAD_COMPOSER, se encarga de componer los SADs para el resto de los modos de estimación de movimiento. De esta manera se consigue una reutilización eficiente de los SADs asociados a los bloques de 4×4 píxeles para el cálculo de los bloques de tamaño superior. Este hecho implica que el recorrido del área de búsqueda debe realizarse una sola vez para el cálculo de los vectores del menor modo de estimación de movimiento, y no una vez para cada modo.

El bloque funcional denominado CHECKER, es el elemento esencial en la eliminación temprana de candidatos. Mediante la inspección de los SADs que obtiene cada bloque funcional SAD_COMPOSER y los SADs asociados a las posiciones mínimas provisionales,

almacenados en el comparador de mínimos, cada CHECKER decide si puede deshabilitar a los elementos de proceso de su grupo durante la evaluación de una posición candidata dentro del área de búsqueda. En el caso de estándares que permitan el uso de varias topologías de bloque en la compensación de movimiento, si se desea reutilizar eficientemente los resultados de los bloques menores en la construcción de los vectores de movimiento de modos superiores, el mecanismo para la detención no es inmediato. En particular, para detener el cálculo del SAD de un determinado bloque de píxeles, es necesario inspeccionar si todos los SADs que se vayan a construir por parte del SAD_COMPOSER correspondiente, han sobrepasado igualmente sus mínimos provisionales.

El esquema del funcionamiento de la arquitectura propuesta se muestra en la Figura 4.3, detallándose las posiciones evaluadas por cada grupo de elementos de proceso, así como el orden seguido por los grupos de elementos de proceso en la evolución de las posiciones evaluadas para un área de búsqueda de 31×31 píxeles. Esta figura muestra que la primera posición evaluada por un grupo cualquiera está distanciada *M* píxeles en horizontal con respecto a la primera posición evaluada por su grupo vecino. Además, una vez se ha completado la evaluación de la primera posición correspondiente a cada grupo, la siguiente posición se encuentra en un desplazamiento vertical de un píxel hasta completar, para el caso del área de búsqueda previamente definida, 16 posiciones separadas por este desplazamiento vertical unitario. Una vez completada esta etapa, el grupo realiza un *salto horizontal*, de manera que la siguiente posición a evaluar vendrá determinada por un desplazamiento horizontal de 1 píxel con respecto a la primera posición evaluada. Este proceso será repetido por todos los grupos, de tal manera que el número de saltos siempre sea menor que *M*.

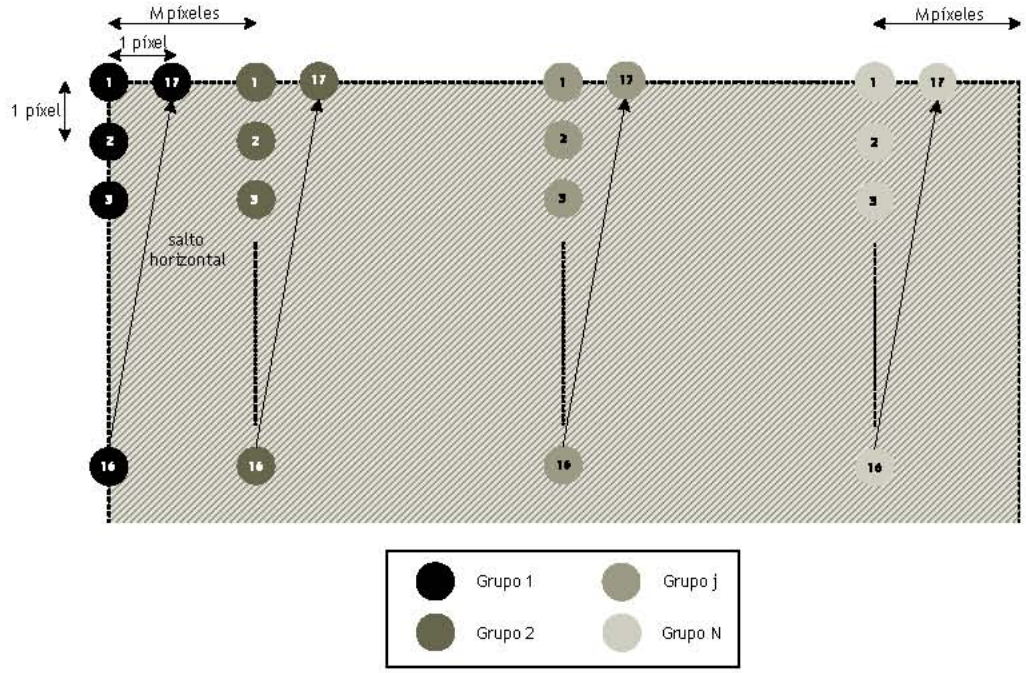


Figura 4.3: Evaluación de posiciones dentro del área de búsqueda por parte de cada uno de los grupos de elementos de proceso.

4.3.1 Análisis de la arquitectura propuesta

Las prestaciones de la arquitectura propuesta en la Figura 4.2 dependerán, en gran medida, de los valores seleccionados para *M* y *N*. Por esta razón, y puesto que la influencia de dichos parámetros no ha sido estudiada en ninguno de los trabajos previos a esta Tesis, es necesario evaluar, en términos arquitecturales, las consecuencias de elegir un número determinado de grupos con una cantidad determinada de elementos de proceso por cada grupo. Asimismo, en el marco de esta Tesis, resulta de extraordinaria importancia analizar qué valores de *M* y *N* son los más adecuados para la posterior implementación de la búsqueda predictiva contemplada por el algoritmo VBS-ACBM.

Sobre la base de la arquitectura propuesta, se han elegido para realizar este análisis, un conjunto significativo de casos en los que se cumple que $M \times N = 16$. La arquitectura

seleccionada, compuesta por 16 elementos de proceso, representa una estructura unidimensional *base* que, una vez optimizada para unos valores particulares de *M* y *N*, es fácilmente escalable en el caso de que se requieran requisitos de procesamiento más exigentes. Esta escalabilidad es la que lleva a seleccionar estos casos de estudio.

4.3.1.1 Primer caso de estudio: *M* = 1 y *N* = 16

En este caso, cada uno de los 16 elementos de proceso de la arquitectura evalúa una posición de manera individual, sin colaborar con ningún otro elemento de proceso en esta tarea y por lo tanto, actuando en su conjunto como una arquitectura *unidimensional no agrupada* que ha sido presentada en el congreso internacional *IEEE International Symposium on Circuits and Systems* (*ISCAS)* [LTV+05a]. De esta manera, y tras un breve periodo de latencia inicial, los 16 elementos de proceso trabajan en paralelo hasta completar la evaluación de la totalidad de posiciones que se encuentran dentro del área de búsqueda, tal y como se muestra en la Tabla 4.1. En dicha tabla se observa la evolución temporal de los cálculos realizados por cada elemento de proceso PE_1^j, mostrándose los ciclos en los que los bloques funcionales SAD_COMPOSER asociados a los dos primeros grupos obtienen los SADs correspondientes a todos los modos de estimación de movimiento para la primera posición de macrobloque evaluada por ambos grupos. Asimismo, se indica en la última columna de la tabla el número de ciclos de reloj que transcurre hasta que se obtienen cada uno de los vectores de movimiento para el macrobloque bajo procesamiento.

T	PE_1^1	PE_1^2	...	PE_1^{16}	SAD_CMP^1	SAD_CMP^2	...	SAD_CMP^{16}	MVs
0	|r0,0-b0,0|		...				...		
1	|r0,1-b0,1|	|r0,0-b0,1|	...				...		
2	|r0,2-b0,2|	|r0,1-b0,2|	...				...		
3	|r0,3-b0,3|	|r0,2-b0,3|	...				...		
...	...	...	...				...		
14	|r0,14-b0,14|	|r0,13-b0,14|	...				...		
15	|r0,15-b0,15|	|r0,14-b0,15|	...	|r0,0-b0,15|			...		
16	|r1,0-b1,0|	|r0,15-b0,16|	...	|r0,1-b0,16|			...		
17	|r1,1-b1,1|	|r1,0-b1,1|	...	|r0,2-b0,17|			...		
...	...	...	...	...			...		
54	|r3,6-b3,6|	|r3,5-b3,6|	...	|r2,7-b2,22|			...		

55	|r3,7-b3,7|	|r3,6-b3,7|	…	|r2,8-b2,23|			…		
56	|r3,8-b3,8|	|r3,7-b3,8|	…	|r2,9-b2,24|			…		
…	…	…	…	…			…		
59	|r3,11-b3,11|	|r3,10-b3,11|	…	…	8×4a		…		
60	|r3,12-b3,12|	|r3,11-b3,12|	…	…		8×4a	…		
…	…	…	…	…			…		
67	|r4,3-b4,3|	|r4,2-b4,3|	…	|r3,4-b3,19|	8×4b		…		
68	|r4,4-b4,4|	|r4,3-b4,4|	…	|r3,5-b3,20|		8×4b	…		
…	…	…	…	…			…		
74	…	…	…	…			…	8×4a	
…	…	…	…	…			…		
114	|r7,2-b7,2|	|r7,1-b7,2|	…	|r6,3-b6,18|			…		
115	|r7,3-b7,3|	|r7,2-b7,3|	…	|r6,4-b6,19|			…		
…	…	…	…	…			…		
119	|r7,7-b7,7|	|r7,6-b7,7|	…	|r6,8-b6,23|	4×8a		…		
120	|r7,8-b7,8|	|r7,7-b7,8|	…	|r6,9-b6,24|		4×8a	…		
…	…	…	…	…			…		
123	|r7,11-b7,11|	|r7,10-b7,11|	…	|r6,12-b6,27|	4×8b ; 8×4c		…		
124	|r7,12-b7,12|	|r7,11-b7,12|	…	|r6,13-b6,28|	8×8a	4×8b; 8×4c	…		
125	…	…	…	…		8×8a	…		
126	…	…	…	…			…		
127	|r7,15-b7,15|	|r7,14-b7,15|	…	|r7,0-b7,15|	4×8c		…		
128	|r8,0-b8,0|	|r7,15-b8,0|	…	|r7,1-b7,16|		4×8c	…		
129	|r8,1-b8,1|	|r8,0-b8,1|	…	|r7,2-b7,17|			…		
130	|r8,2-b8,2|	|r8,1-b8,2|	…	|r7,3-b7,18|			…		
131	|r8,3-b8,3|	|r8,2-b8,3|	…	|r7,4-b7,19|	4×8d ; 8×4d		…		
132	|r8,4-b8,4|	|r8,3-b8,4|	…	|r7,5-b7,20|	8×8b	4×8d ; 8×4d	…	4×8a	
133	|r8,5-b8,5|	|r8,4-b8,5|	…	|r7,6-b7,21|	16×8a	8×8b	…		
134	|r8,6-b8,6|	|r8,5-b8,6|	…	|r7,7-b7,22|		16×8a	…		
…	…	…	…	…			…		
247	|r15,7-b15,7|	|r15,6-b15,7|	…	|r14,8-b14,23|			…		
248	|r15,8-b15,8|	|r15,7-b15,8|	…	|r14,9-b14,24|			…		
…	…	…	…	…			…		
252	|r15,12-b15,12|	|r15,11-b15,12|	…	|r14,13-b14,28|	4×8f ; 8×4g		…		
253	|r15,13-b15,13|	|r15,12-b15,13|	…	|r14,14-b14,29|	8×8c	4×8f ; 8×4g	…		
254	|r15,14-b15,14|	|r15,13-b15,14|	…	|r14,15-b14,30|	8×16a	8×8c	…		
255	|r15,15-b15,15|	|r15,14-b15,15|	…	|r15,0-b15,15|		8×16a	…		
256	|r0,0-b1,0|	|r15,15-b15,16|	…	|r15,1-b15,16|			…		
…	…	…	…	…			…		
259	|r0,3-b1,3|	|r0,2-b1,3|	…	|r15,4-b15,19|	4×8h ; 8×4h		…		
260	|r0,4-b1,4|	|r0,3-b1,4|	…	|r15,5-b15,20|	8×8d	4×8h ; 8×4h	…		
261	|r0,5-b1,5|	|r0,4-b1,5|	…	|r15,6-b15,21|	8×16b; 16×8b	8×8d	…		
262	|r0,6-b1,6|	|r0,5-b1,6|	…	|r15,7-b15,22|	16×16	8×16b; 16×8b	…		
263	|r0,7-b1,7|	|r0,6-b1,7|	…	|r15,8-b15,23|		16×16	…		
…	…	…	…	…			…		
3910	|r4,6-b19,6|	|r4,5-b19,6|	…	|r3,7-b18,22|			…		4×4a
…	…	…	…	…			…		
3914	|r4,10-b19,10|	|r4,9-b19,10|	…	|r3,11-b18,26|			…		4×4b
3915	|r4,11-b19,11|	|r4,10-b19,11|	…	|r3,12-b18,27|			…		8×4a
…	…	…	…	…			…		
3918	|r4,14-b19,14|	|r4,13-b19,14|	…	|r3,15-b18,30|			…		4×4c
	…	…	…	…			…		
3922	|r5,2-b20,2|	|r5,1-b20,2|	…	|r4,3-b19,17|			…		4×4d
3923	|r5,3-b20,3|	|r5,2-b20,3|	…	|r4,4-b19,18|			…		8×4b
…	…	…	…	…			…		
3974	|r8,6-b23,6|	|r8,5-b23,6|	…	|r7,7-b22,21|			…		4×4e
3975	|r8,7-b23,7|	|r8,6-b23,7|	…	|r7,8-b22,22|			…		4×8a
…	…	…	…	…			…		
3978	|r8,10-b23,10|	|r8,9-b23,10|	…	|r7,11-b22,26|			…		4×4f
3979	|r8,11-b23,11|	|r8,10-b23,11|	…	|r7,12-b22,27|			…		4×8b 8×4c
3980	|r8,12-b23,12|	|r8,11-b23,12|	…	|r7,13-b22,28|			…		8×8a
3981	|r8,13-b23,13|	|r8,12-b23,13|	…	|r7,14-b22,29|			…		
3982	|r8,14-b23,14|	|r8,13-b23,14|	…	|r7,15-b22,30|			…		4×4g
3983	|r8,15-b23,15|	|r8,14-b23,15|	…	|r8,0-b23,15|			…		4×8c

…			…				…		
3986	|r9,2-b24,2|	|r9,1-b24,2|	…	|r8,3-b23,18|			…		4×4h
3987	|r9,3-b24,3|	|r9,2-b24,3|	…	|r8,4-b23,19|			…		4×8d 8×4d
3988	|r9,4-b24,4|	|r9,3-b24,4|	…	|r8,5-b23,20|			…		8×8b
3989	|r9,5-b24,5|	|r9,4-b24,5|	…	|r8,6-b23,21|			…		16×8a
…			…				…		
4095	|r15,15-30,15|	|r15,14-30,15|	…	|r15,0-b30,15|			…		
…		…	…	…			…		
4102			…	|r15,7-b30,22|			…		4×4m
4103			…	|r15,8-b30,23|			…		4×8e
…			…	…			…		
4106			…	|r15,11-b30,26|			…		4×4n
4107			…	|r15,12-b30,27|			…		4×8f 8×4g
4108			…	|r15,13-b30,28|			…		8×8c
4109			…	|r15,14-b30,29|			…		8×16a
4110			…	|r15,15-b30,30|			…		4×4o
4111			…				…		4×8g
…			…				…		
4114			…				…		4×4p
4115			…				…		4×8h 8×4h
4116			…				…		8×8c
4117			…				…		8×16b 16×8b
4118			…						16×16

Tabla 4.1: Diagrama de tiempos del módulo de estimación de movimiento con precisión entera para $M = 1$ y $N = 16$.

4.3.1.2 Segundo caso de estudio: $M = 4$ y $N = 4$

Este caso de estudio implica la existencia de cuatro grupos de cuatro elementos de proceso cada uno por lo que, después de un periodo inicial de latencia, se evalúan cuatro posiciones del área de búsqueda en paralelo. De esta manera, se obtiene una arquitectura sistólica *unidimensional agrupada*.

En este caso, cada grupo se encarga de la evaluación de 64 posiciones hasta completar la totalidad de posiciones dentro del área de búsqueda, tal y como se indica en la Tabla 4.2. En esta tabla se ha omitido la información relativa a los ciclos en los que los respectivos bloques SAD_COMPOSER obtienen los SADs de modos superiores, así como los ciclos en los que el comparador de mínimos obtiene los vectores de movimiento finales. Este hecho es debido a que, una vez completado el cálculo de los diferentes bloques de 4×4 píxeles dentro de un macrobloque, el número de ciclos que transcurre hasta la obtención del SAD asociado a

modos superiores, y de los vectores de movimiento finales, es el mismo que para el caso anterior de estudio, puesto que las arquitecturas de los bloques SAD_COMPOSER y del comparador de mínimos no sufren variación alguna.

T	PE_1^1	PE_2^1	...	PE_4^1	PE_1^2	...	PE_1^3	...	PE_1^4
0	|r0,0-b0,0|		...			...		...	
1	|r0,4-b0,4|	|r0,1-b0,1|	...		|r0,0-b0,4|	...		...	
2	|r0,8-b0,8|	|r0,5-b0,5|	...		|r0,4-b0,8|	...	|r0,0-b0,8|	...	
3	|r0,12-b0,12|	|r0,9-b0,9|	...	|r0,3-b0,3|	|r0,8-b0,12|	...	|r0,4-b0,12|	...	|r0,0-b0,12|
4	|r1,0-b1,0|	|r0,13-b0,13|	...	|r0,7-b0,7|	|r0,12-b0,16|	...	|r0,8-b0,16|	...	|r0,4-b0,16|
5	|r1,4-b1,4|	|r1,1-b1,1|	...	|r0,11-b0,11|	|r1,0-b1,4|	...	|r0,12-b0,20|	...	|r0,8-b0,20|
6	|r1,8-b1,8|	|r1,5-b1,5|	...	|r0,15-b0,15|	|r1,4-b1,8|	...	|r1,0-b1,8|	...	|r0,12-b0,24|
7	|r1,12-b1,12|	|r1,9-b1,9|	...	|r1,3-b1,3|	|r1,8-b1,12|	...	|r1,4-b1,12|	...	|r1,0-b1,12|
...	...	...	...	...	...	...	...	...	...
60	|r15,0-b15,0|	|r14,13-b14,13|	...	|r14,7-b14,7|	|r14,12-b14,16|	...	|r14,8-b14,16|	...	|r14,4-b14,16|
61	|r15,4-b15,4|	|r15,1-b15,1|	...	|r14,11-b14,11|	|r15,0-b15,4|	...	|r14,12-b14,20|	...	|r14,8-b14,20|
62	|r15,8-b15,8|	|r15,5-b15,5|	...	|r14,15-b14,15|	|r15,4-b15,8|	...	|r15,0-b15,8|	...	|r14,12-b14,24|
63	|r15,12-b15,12|	|r15,9-b15,9|	...	|r15,3-b15,3|	|r15,8-b15,12|	...	|r15,4-b15,12|	...	|r15,0-b15,12|
64	|r0,0-b1,0|	|r15,13-b15,13|	...	|r15,7-b15,7|	|r15,12-b15,16|	...	|r15,8-b15,16|	...	|r15,4-b15,16|
65	|r0,4-b1,4|	|r0,1-b1,1|	...	|r15,11-b15,11|	|r0,0-b1,4|	...	|r15,12-b15,20|	...	|r15,8-b15,20|
66	|r0,8-b1,8|	|r0,5-b1,5|	...	|r15,15-b15,15|	|r0,4-b1,8|	...	|r0,0-b1,8|	...	|r15,12-b15,24|
...	...	...	...	...	...	...	...	...	...
1023	|r15,12-b30,12|	|r15,9-b30,9|	...	|r15,3-b30,3|	|r15,8-b30,12|	...	|r15,4-b30,12|	...	|r15,0-b30,12|
1024	|r0,0-b0,1|	|r15,13-b30,13|	...	|r15,7-b30,7|	|r15,12-b30,16|	...	|r15,8-b30,16|	...	|r15,4-b30,16|
1025	|r0,4-b0,5|	|r0,1-b0,2|	...	|r15,11-b30,11|	|r0,0-b0,5|	...	|r15,12-b30,20|	...	|r15,8-b30,20|
...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...
4095	|r15,12-b30,15|	|r15,9-b30,12|	...	|r15,3-b30,6|	|r15,8-b30,15|	...	|r15,4-b30,15|	...	|r15,0-b30,15|
4096		|r15,13-b30,16|	...	|r15,7-b30,10|	|r15,12-b30,19|	...	|r15,8-b30,19|	...	|r15,4-b30,19|
4097			...	|r15,11-b30,14|		...	|r15,12-b30,23|	...	|r15,8-b30,23|
4098			...	|r15,15-b30,18|		...		...	|r15,12-b30,27|

Tabla 4.2: Diagrama de tiempos simplificado del módulo de estimación de movimiento con precisión entera para $M = 4$ y $N = 4$.

4.3.1.3 Tercer caso de estudio: *M* = 16 y *N* = 1

Este último caso de estudio se caracteriza por la existencia de un único grupo formado por la totalidad de elementos de proceso disponibles, obteniéndose una arquitectura *unidimensional totalmente agrupada*, de la cual se muestra su correspondiente diagrama de tiempos en la Tabla 4.3.

T	PE_1^1	PE_2^1	PE_3^1	PE_4^1	...	PE_{16}^1
0	|r0,0-b0,0|	...	...	...	...	...
1	|r1,0-b1,0|	|r0,1-b0,1|	...	...	...	...
2	|r2,0-b2,0|	|r1,1-b1,1|	|r0,2-b0,2|	...	...	...
3	|r3,0-b3,0|	|r2,1-b2,1|	|r1,2-b1,2|	|r0,3-b0,3|	...	...
...	...	...	...	...	...	...
15	|r15,0-b15,0|	|r14,1-b14,1|	|r13,2-b13,2|	|r12,3-b12,3|	...	|r0,15-b0,15|
16	|r0,0-b1,0|	|r15,1-b15,1|	|r14,2-b14,2|	|r13,3-b14,3|	...	|r1,15-b1,15|
...	...	...	...	...	...	...
256	|r0,0-b0,1|	|r15,1-b30,1|	|r14,2-b29,2|	|r13,3-b28,3|	...	|r1,15-b16,15|
257	|r1,0-b1,1|	|r0,1-b0,2|	|r15,2-b30,2|	|r14,3-b29,3|	...	|r2,15-b17,15|
258	|r2,0-b2,1|	|r1,1-b1,2|	|r0,2-b0,3|	|r15,3-b30,3|	...	|r3,15-b18,15|
259	|r3,0-b3,1|	|r2,1-b2,2|	|r1,2-b1,3|	|r0,3-b0,4|	...	|r4,15-b19,15|
...	...	...	...	...	...	...
3840	|r0,0-b0,15|	|r15,1-b30,15|	|r14,2-b29,16|	|r13,3-b28,17|	...	|r1,15-b16,29|
3841	|r1,0-b1,15|	|r0,1-b0,16|	|r15,2-b30,16|	|r14,3-b29,17|	...	|r2,15-b17,29|
3842	|r2,0-b2,15|	|r1,1-b1,16|	|r0,2-b0,17|	|r15,3-b30,17|	...	|r3,15-b18,29|
3843	|r3,0-b3,15|	|r2,1-b2,16|	|r1,2-b1,17|	|r0,3-b0,18|	...	|r4,15-b19,29|
...	...	...	...	...	...	...
4095	|r15,0-b30,15|	|r14,1-b29,16|	|r13,2-b28,17|	|r12,3-b27,18|	...	|r0,15-b15,30|
4096		|r15,1-b30,16|	|r14,2-b29,17|	|r13,3-b28,18|	...	|r1,15-b16,30|
4097			|r15,2-b30,17|	|r14,3-b29,18|	...	|r2,15-b17,30|
4098				|r15,3-b30,18|	...	|r3,15-b18,30|
...					...	...
4110					...	|r15,15-b30,30|

Tabla 4.3: Diagrama de tiempos simplificado del módulo de estimación de movimiento con precisión entera para $M = 16$ y $N = 1$.

4.3.1.4 Análisis arquitectural de los casos de estudio seleccionados

El análisis del funcionamiento de las tres arquitecturas de estudio, detallado ciclo por ciclo en las Tablas 4.1, 4.2 y 4.3, permite establecer las siguientes conclusiones para cada uno de los casos considerados:

A) Arquitectura con $M = 1$ y $N = 16$

- En este caso, se necesita leer un único píxel de referencia por ciclo de reloj. Este hecho es debido a que el primer elemento de proceso accede directamente a los datos de la memoria de referencia, transmitiéndolos hacia el resto de elementos de proceso a través de la línea de retardos correspondiente. Asimismo se necesitan, como máximo, dos píxeles correspondientes al área de búsqueda por ciclo de reloj.

- El número de ciclos que transcurre desde que se termina de evaluar la contribución al SAD de una fila de un determinado bloque de 4×4 píxeles, hasta que se comienza a evaluar la contribución de la siguiente fila del mismo bloque, es elevado. Este hecho facilita enormemente el diseño del bloque CHECKER, tal y como se detallará en el siguiente apartado.

B) Arquitectura con $M = 4$ y $N = 4$

- En este caso, por ciclo de reloj se necesitan, como máximo, cuatro y siete píxeles diferentes del área de referencia y del área de búsqueda, respectivamente.

- El número de ciclos que transcurre desde que se termina de evaluar la contribución al SAD de una fila de un determinado bloque de 4×4 píxeles, hasta que se comienza a evaluar la contribución de la siguiente fila del mismo bloque, es considerablemente menor que para el anterior caso de estudio. De hecho, y según lo expuesto en la Tabla 4.2, un ciclo después de que el último elemento de proceso dentro de un grupo realice la última operación correspondiente a una fila de píxeles, el primer elemento de proceso de ese mismo grupo comienza con la evaluación de la siguiente fila de píxeles., lo cual supone un factor determinante en el diseño del bloque CHECKER.

C) Arquitectura con $M = 16$ y $N = 1$

- En este caso se necesitan, como máximo, dieciséis píxeles nuevos de la memoria de referencia y de la memoria de búsqueda por cada ciclo de reloj.

- Resulta imposible introducir de manera eficiente técnicas de eliminación temprana de candidatos en los términos planteados en esta Tesis. Este hecho es debido a que, un

ciclo de reloj después de la evaluación de la última resta correspondiente a una fila de un bloque de 4×4 píxeles cualquiera, se comienza con la evaluación de la misma fila correspondiente al bloque de 4×4 píxeles inmediatamente inferior dentro del mismo macrobloque.

Este conjunto de observaciones realizadas para cada uno de los casos de estudio considerados, permiten extraer dos primeras conclusiones generales. En primer lugar, a medida que aumenta el número de elementos de proceso por grupo, los requisitos en cuanto al diseño de las memorias de referencia y de búsqueda son más exigente. En segundo lugar, mientras mayor sea el número de elementos de proceso por grupo, más difícil resulta la eliminación de candidatos a nivel de bloques de 4×4 píxeles. En particular, para el último de los casos de estudio considerados, resulta imposible detener los elementos de proceso del grupo durante la estimación de movimiento en bloques de 4×4 píxeles. Esta característica, en cuanto a que los vectores de movimiento para bloques de 4×4 píxeles son requeridos sólo por el reciente estándar H.264/AVC, no había sido considerada en las arquitecturas con capacidad de eliminación temprana presentadas por otros autores.

4.3.2 Descripción arquitectural detallada de los casos de estudio propuestos

Independientemente de los valores que se establezcan para los parámetros *M* y *N*, existen para todos los casos posibles de estudio un conjunto de bloques funcionales comunes. A continuación se detalla la arquitectura interna de cada uno de estos bloques funcionales, así como la función principal de cada uno de ellos dentro de la arquitectura de estimación de movimiento con precisión entera correspondiente a cada uno de los casos de estudio seleccionados.

4.3.2.1 Elemento de proceso (PE)

El elemento de proceso (*Processing Element – PE*) es la unidad funcional básica dentro de la arquitectura general propuesta, siendo la función principal del elemento de proceso *i-ésimo* perteneciente al grupo *j-ésimo* calcular el valor absoluto de la resta entre un píxel del macrobloque de referencia r_i^j y un píxel cualquiera del área de búsqueda b_i^j. El valor resultante se suma al valor obtenido por el elemento de proceso que le precede en su grupo, transmitiéndose el resultado final hacia el siguiente elemento de proceso dentro del mismo grupo a través de un registro de 12 bits, siempre y cuando existan ambos elementos de proceso vecinos dentro del mismo grupo. El número de bits de dicho registro viene determinado por el valor máximo que puede tomar el SAD para un bloque de 4×4 píxeles.

La arquitectura del elemento de proceso *i-ésimo* correspondiente al grupo *j-ésimo* se muestra en la Figura 4.4, observándose la posibilidad de introducir los correspondientes valores de píxel a la entrada del elemento de proceso, o por el contrario, introducir valores nulos. Esta alternativa se contempla con el objetivo de detener el funcionamiento de la lógica combinacional de cada elemento de proceso una vez que se hayan verificado las condiciones necesarias para proceder a la eliminación temprana de la posición que se está evaluando, y por lo tanto, a la detención del elemento de proceso correspondiente. En este sentido, los ciclos de reloj durante los cuales no se producen conmutaciones en la lógica combinacional asociada a la implementación cada elemento de proceso, se denominan en esta Tesis *ciclos inactivos*. La presencia de estos ciclos en una arquitectura de estimación de movimiento basada en elementos de proceso, garantiza una importante reducción en el consumo de potencia de la misma, tal y como han demostrado los trabajos publicados en [SR99], [Sou99], [LTC04] y [DRS05].

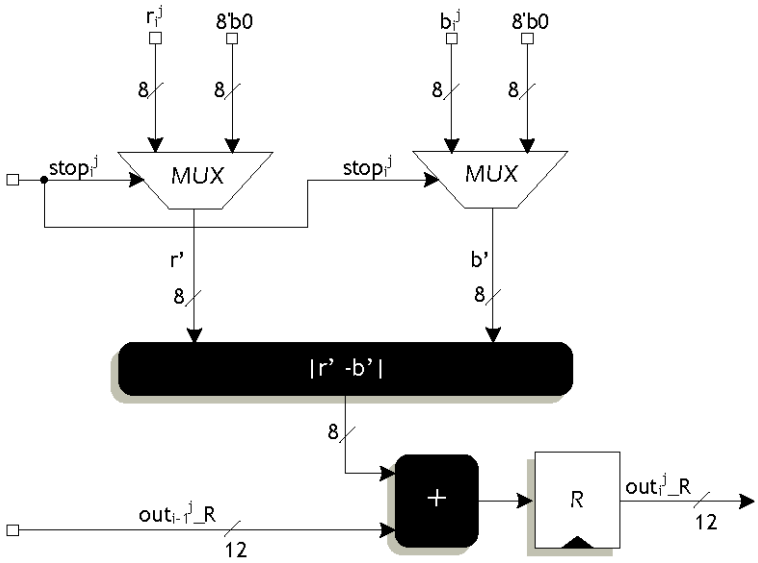


Figura 4.4: Estructura interna de un elemento de proceso genérico.

4.3.2.2 Grupo de elementos de proceso (GRUPO)

Este bloque funcional está constituido por una asociación de elementos de proceso que cooperan en la evaluación del SAD correspondiente a una posición dentro del área de búsqueda, tal y como se muestra en la Figura 4.5, en la que se representa el diagrama de bloques correspondiente a un grupo *j-ésimo* genérico. Este tipo de agrupación de elementos de proceso permite estudiar las prestaciones de la arquitectura dependiendo del número de elementos de proceso por grupo, tal y como se analizará posteriormente en este capítulo.

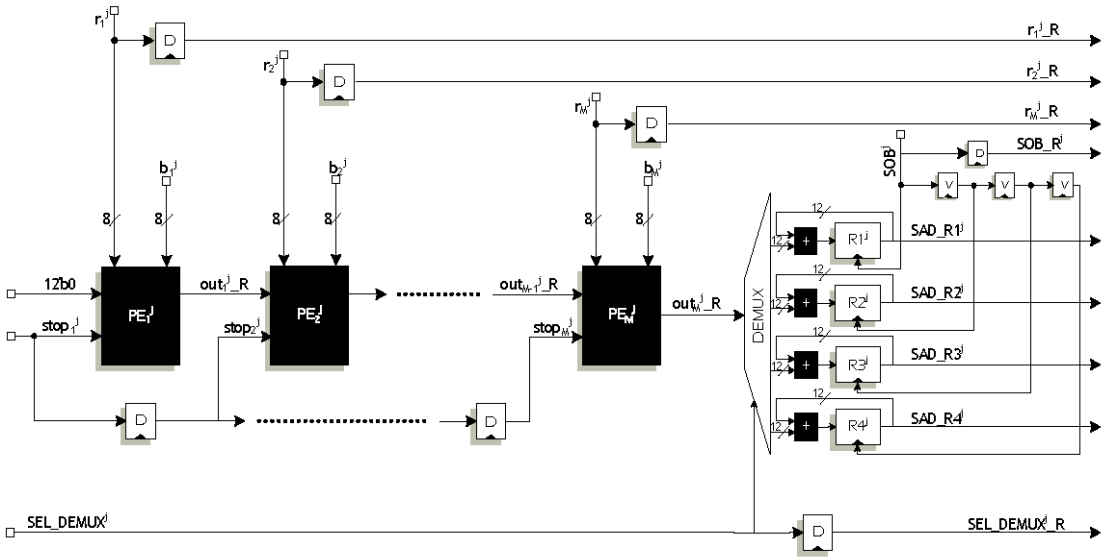


Figura 4.5: Diagrama de bloques correspondiente a un grupo de elementos de proceso.

En la Figura 4.5 se observa que el cálculo de los SADs correspondientes a una determinada posición evaluada por un grupo de elementos de proceso se lleva a cabo haciendo uso de cuatro registros de acumulación. De esta manera, se consigue diferenciar el SAD correspondiente a cada uno de los 16 bloques de 4×4 píxeles que conforman la posición de macrobloque que se está evaluando. En particular, tal y como se muestra en la Figura 4.6, cada uno de estos cuatro registros almacenará los SADs de cuatro bloques 4×4 diferentes.

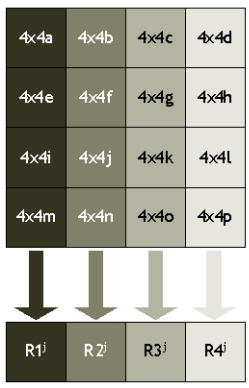


Figura 4.6: Propuesta de almacenamiento en cuatro registros de acumulación de los SADs correspondientes a los bloques 4×4 dentro de un macrobloque.

Mediante el uso de la estrategia propuesta se consigue evitar la utilización de la lógica de multiplexación de 1-a-16 propuesta por otros autores en [YM03] y [CCH+06a], con el consiguiente impacto en el área final del circuito sin ningún coste adicional. Una vez que se ha completado el cálculo del SAD de un bloque 4×4 cualquiera, éste se envía a su respectivo bloque SAD_COMPOSER con el objetivo de calcular los SADs de bloques superiores a partir de los valores acumulados en los mencionados registros. Un ciclo después, dicho registro se inicializa mediante la activación de la señal *Start Of Block* (SOB) correspondiente. Este proceso se repetirá para el resto de los registros, provocándose su inicialización después de un número variable de ciclos (retardo de V ciclos) con respecto al registro $R1^j$, dependiendo el valor de V de los valores seleccionados de los parámetros M y N.

Por último, es importante destacar que, puesto que entre el funcionamiento de dos grupos consecutivos sólo existe un ciclo de retardo, cada grupo envía a través de líneas de retardo unitario (D) los datos del macrobloque de referencia, así como las señales de control de los cuatro registros de acumulación a su grupo inmediatamente posterior, facilitándose así la estrategia de accesos a memoria y el control global de la arquitectura. Asimismo, y utilizando también una línea de retardo unitario, cada elemento de proceso transmite a su inmediato vecino el valor de la señal $stop_i^j$, consiguiéndose así una correcta detención del cálculo de los SADs asociados a la posición de búsqueda que el grupo *j-ésimo* se encuentre evaluando.

4.3.2.3 Constructor de SADs de modos superiores (SAD_COMPOSER)

Este bloque funcional es el encargado de construir, a partir de las cuatro salidas de cada grupo, todos los SADs de los modos superiores. Para ello, se utiliza un conjunto de ocho registros de 12 bits que son actualizados mediante dos señales de habilitación debidamente retardadas en tiempo entre sí, de manera que se actualice correctamente el valor del bloque 4×4 que se ha acabado de procesar dentro del macrobloque. A partir de los valores

almacenados en estos ocho registros, y haciendo uso de un conjunto de sumadores de ancho de palabra variable, se obtienen el resto de los SADs de todos los modos superiores, transmitiéndose al comparador de mínimos los valores de SAD almacenados en los registros (R) resaltados en la Figura 4.7.

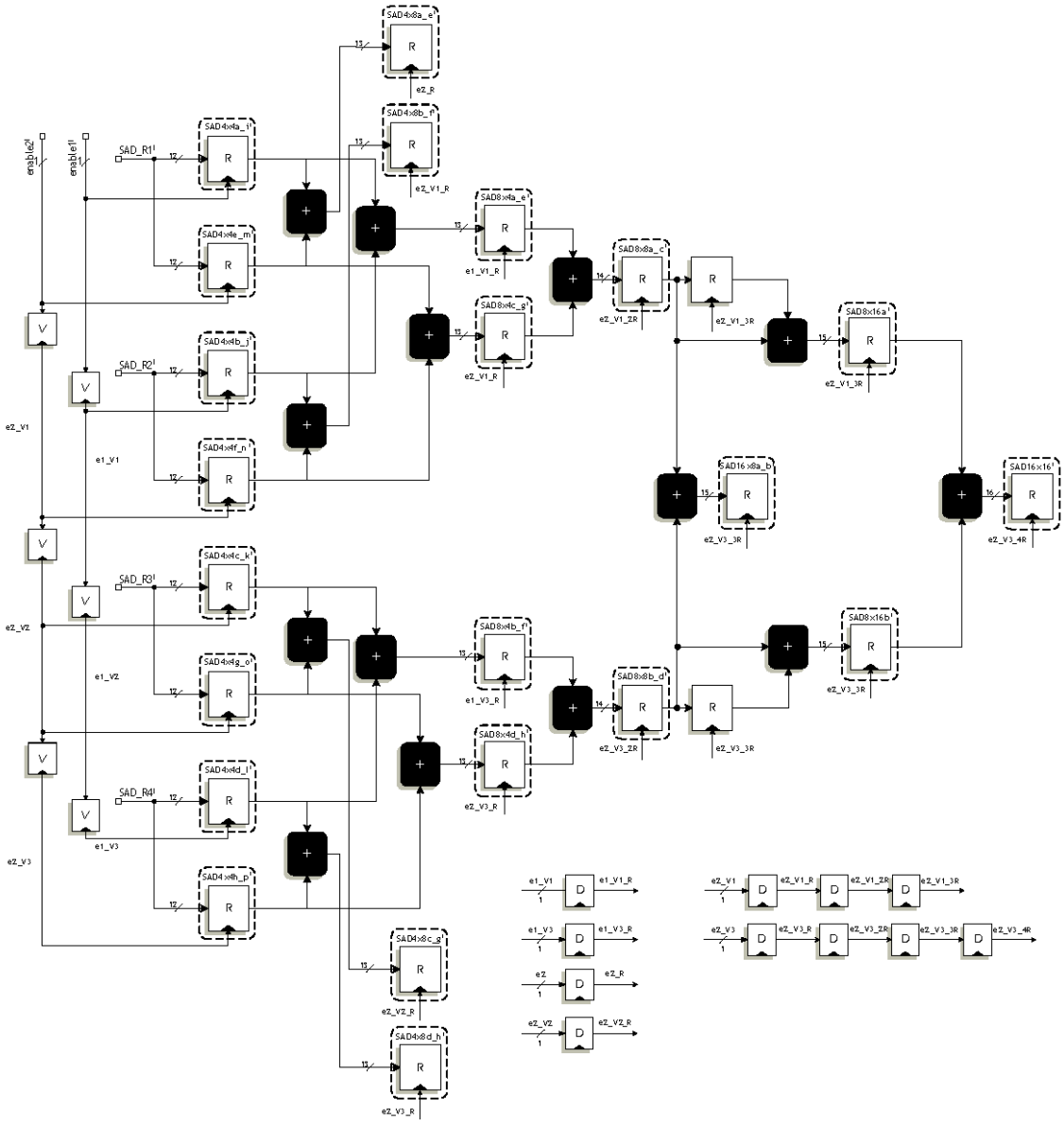


Figura 4.7: Estructura y control del bloque SAD_COMPOSER.

Es importante destacar que el cálculo y posterior transferencia hacia el comparador de mínimos de los SADs correspondientes a modos superiores se produce en dos etapas. De esta manera, durante la primera etapa se calculan los SADs asociados a la mitad superior del macrobloque (4×8a, 4×8b, 4×8c, 4×8d; 8×4a, 8×4b, 8×4c, 8×4d; 8×8a, 8×8b y 16×8a) mientras que durante la segunda etapa se calculan los correspondientes a la mitad inferior del macrobloque (4×8e, 4×8f, 4×8g, 4×8h; 8×4e, 8×4f, 8×4g, 8×4h; 8×8c, 8×8d y 16×8b), a la vez que se completa el cálculo de los SADs asociados a modos que implican a ambas mitades del macrobloque (8×16a, 8×16b y 16×16).

4.3.2.4 Comparador de mínimos

El comparador de mínimos está compuesto por 41 comparadores independientes, uno por cada vector de movimiento a calcular de acuerdo al estándar H.264/AVC. El diagrama de bloques correspondiente a este módulo se presenta en la Figura 4.8, mostrándose la transmisión de los vectores de movimiento de cada bloque a su correspondiente comparador de mínimos a través de un multiplexor en el que se selecciona la salida de cada uno de los bloques SAD_COMPOSER.

A su vez, cada uno de los comparadores realizará la operación de comparación solamente en el caso de que se encuentre activa su señal de habilitación correspondiente, verificando si el SAD recibido es menor que el almacenado hasta ese momento y actualizando, en caso afirmativo, el vector de movimiento correspondiente.

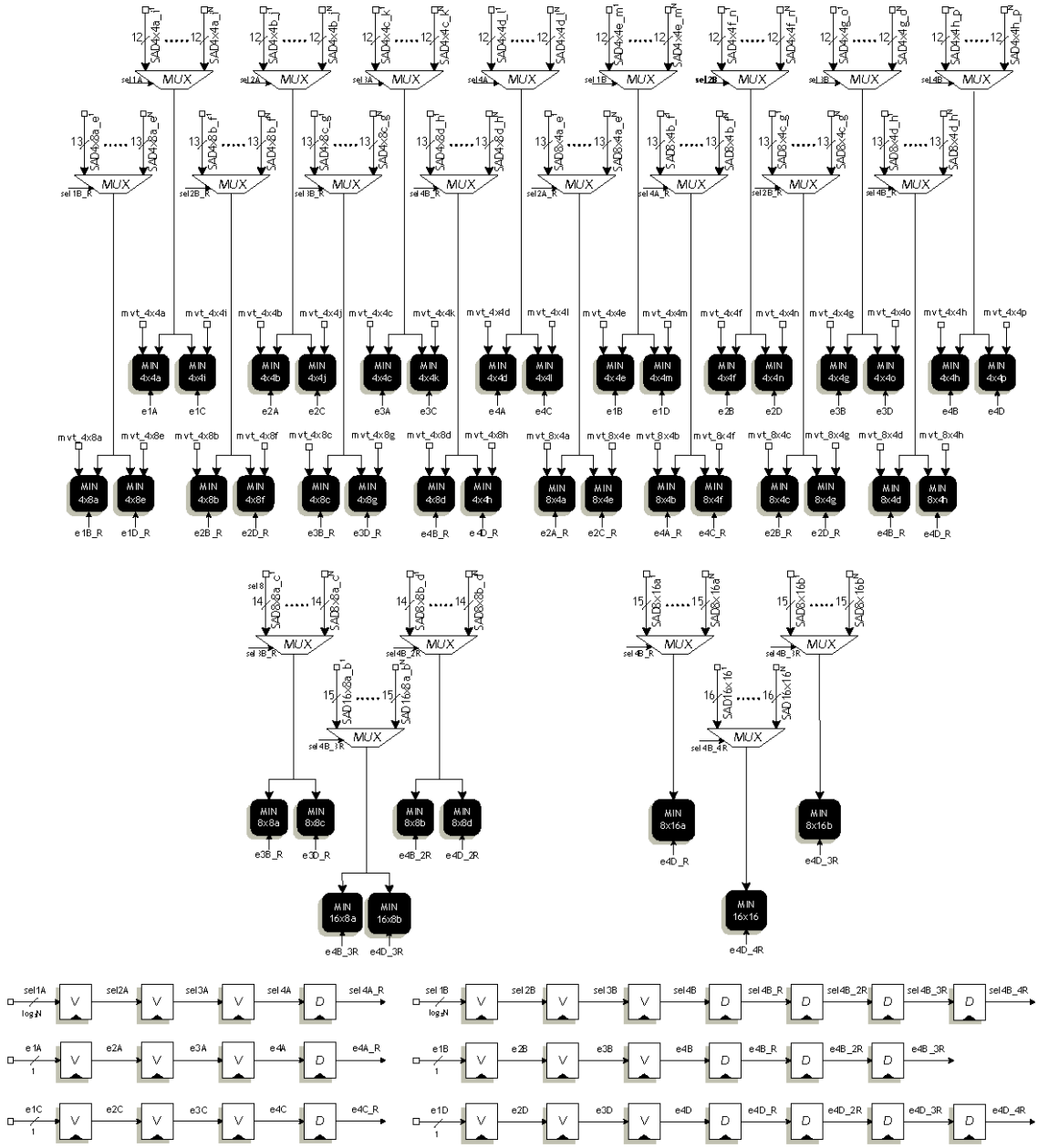


Figura 4.8: Estructura y control del bloque comparador de mínimos.

Para habilitar la operación de un determinado comparador será necesario, no sólo que su multiplexor asociado se encuentre transmitiendo los correspondientes SADs, sino también que el modo de estimación al que dicho elemento de comparación está asociado haya sido activado en el registro de configuración que se presenta en la Figura 4.9.

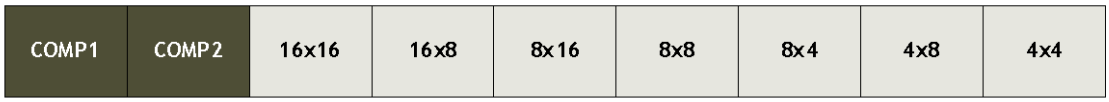


Figura 4.9: Campos del registro de configuración.

La introducción de este registro de configuración en la arquitectura de estimación de movimiento permite, a diferencia de otros trabajos recientemente publicados, realizar el proceso de estimación de movimiento exclusivamente para un conjunto de modos activos, investigándose en esta Tesis los beneficios arquitecturales que se derivan de esta característica.

Por último, los dos bits más significativos del mencionado registro son utilizados exclusivamente por el módulo CHECKER, el cual se detalla a continuación.

4.3.2.5 Comprobación y activación de eliminación temprana de candidatos (CHECKER)

Este bloque funcional es el encargado de detener a los elementos de proceso de un grupo determinado cuando todos los SADs correspondientes a la posición que está siendo evaluada por el grupo de elementos de proceso han superado sus respectivos mínimos. Para ello, cada módulo CHECKER compara individualmente los SADs registrados en el comparador de mínimos con los valores parciales de los SADs obtenidos hasta el momento de la comparación, dotando a la arquitectura global de capacidad de eliminación temprana de candidatos.

Es importante destacar que, si bien en la bibliografía reciente se encuentran arquitecturas en las que se implementa algún tipo de estrategia de eliminación temprana de candidatos, como en [Sou99], [SR99], [ESX+01], [MGS+02], [HCH+04] y [LTC04], un análisis detallado de éstas evidencia las siguientes limitaciones:

- Permiten únicamente el cálculo de un vector por macrobloque (correspondiente al modo 16×16) y por lo tanto, la estrategia de eliminación temprana se aplica únicamente para este modo de estimación de movimiento.

- El número de comparaciones que se realizan entre el valor parcial del SAD que se está calculando y el mínimo provisional, no es adaptable a la secuencia de vídeo que se está procesando. De manera general, en las mencionadas arquitecturas se produce una comparación cada vez que se realiza una nueva acumulación, sin tener en cuenta el coste de la propia operación de comparación.

Estas dos limitaciones son resueltas por la arquitectura propuesta en esta Tesis, y en particular por el bloque funcional CHECKER, mediante la introducción de dos nuevos procedimientos:

- **Creación de árboles de dependencias**. A partir de la información presente en los siete bits menos significativos del registro de configuración presentado en la Figura 4.9, se inicializan 41 registros de un bit que indican si cada uno de los bloques asociados a un modo de estimación de movimiento particular deben ser calculados. Una vez que se han determinado cuáles de los 41 vectores de movimiento correspondientes a un macrobloque deben ser calculados, en el caso de que el SAD parcial correspondiente a uno de ellos supere su mínimo provisional, su registro correspondiente será puesto a cero. Sólo en el caso de que todos los registros de un mismo árbol de dependencias se encuentren a cero, se podrá detener el funcionamiento del grupo de elementos de proceso que esté calculando los mencionados vectores de movimiento. En la Figura 4.10 se muestra un ejemplo de la dependencia que presentan los diferentes modos de estimación de movimiento con los resultados obtenidos en los bloques de 4×4 píxeles, en particular, con el bloque 4×4c.

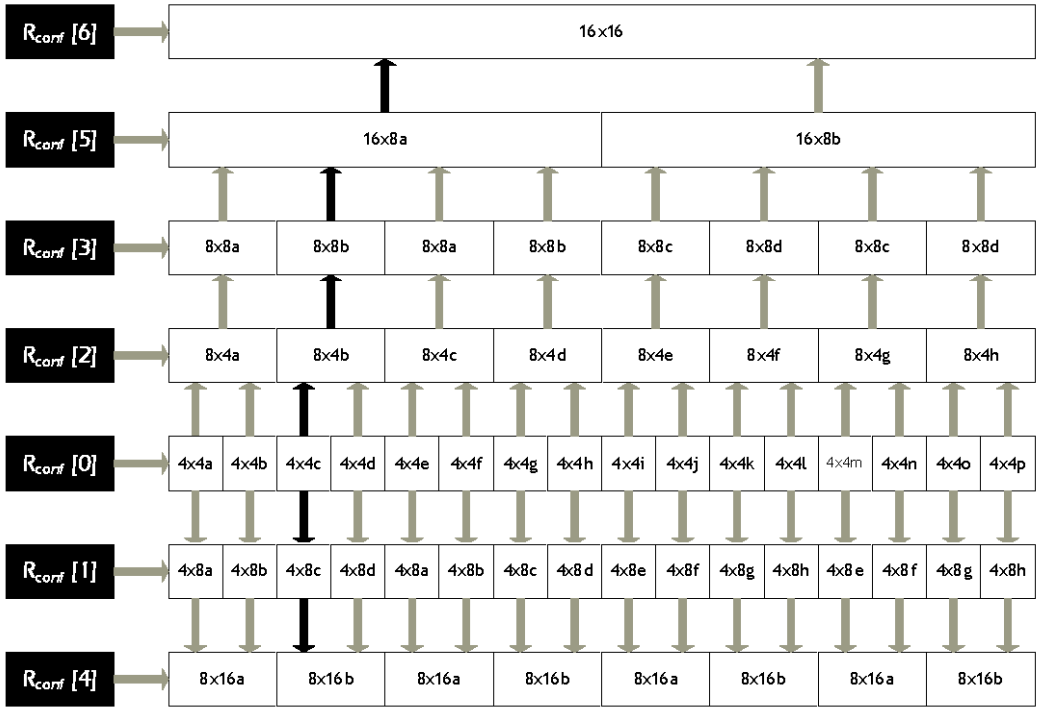


Figura 4.10: Árbol de dependencias.

- **Flexibilidad en el número de comparaciones**. Los dos bits más significativos del registro de configuración permiten introducir diferentes patrones de comparación. En particular, en la arquitectura propuesta, se permiten tres posibles formas de comparación, mostradas en la Figura 4.11: al final de cada bloque 4×4 (A), a mitad de cada bloque 4×4 (B) y al final de cada línea (C). Debido a que el registro de configuración puede ser establecido para cada macrobloque, el número de comparaciones a realizar puede ser modificado con facilidad dependiendo de las características espaciales y temporales de la secuencia de vídeo a comprimir.

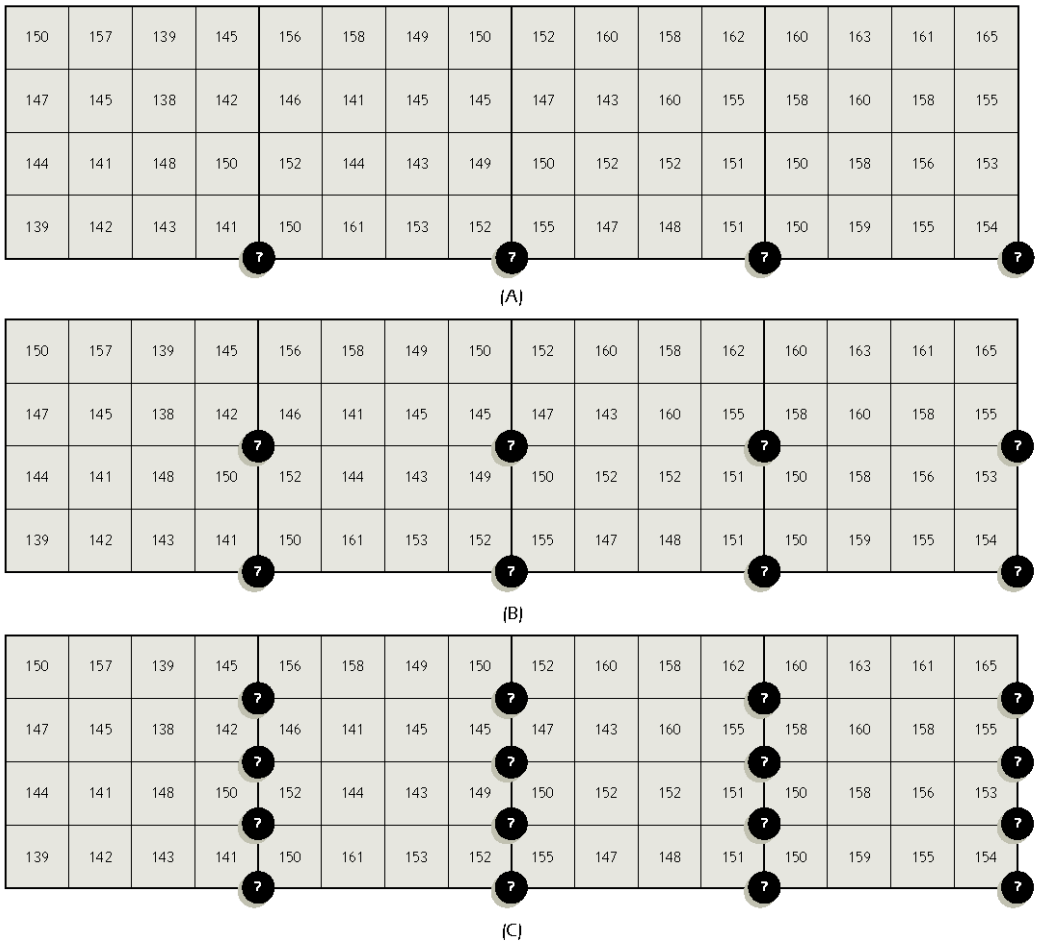


Figura 4.11: Diferentes posibilidades de comparación en cada uno de los módulos CHECKER.

Debido a que en la arquitectura propuesta se ha restringido el número de elementos de proceso a 16, la arquitectura interna de los CHECKERs dependerá en gran medida de los valores de M y N seleccionados, tal y como se mencionó en el apartado anterior.

En particular, para la arquitectura con $M = 1$ y $N = 16$, al transcurrir un número de ciclos elevado desde que se termina de evaluar la contribución al SAD de una fila de un determinado bloque de 4×4 píxeles hasta que se comienza a evaluar la contribución de la siguiente fila del mismo bloque, la arquitectura del bloque CHECKER es prácticamente idéntica a la diseñada para el módulo SAD_COMPOSER (véase Figura 4.7) en cuanto a los

recursos de cómputo se refiere. Obviamente, la temporización de las señales de control variará sensiblemente, pues el valor de los registros debe actualizarse según el patrón de comparación elegido y los modos que se hayan activado en el registro de configuración. Asimismo, ha de incluirse un conjunto de comparadores cuyo objetivo es comparar el valor de cada uno de los registros con sus respectivos mínimos provisionales de tal manera que, si el SAD acumulado para un bloque cualquiera es mayor que el mínimo provisional, se introduce un cero lógico en el registro correspondiente dentro del árbol de dependencias. Durante este proceso de comparación, si se alcanza la situación en la que todos los registros asociados a una misma *rama* del árbol de dependencias están a cero, se habilita la señal $stop_j$ con el objetivo que se detenga la operación del grupo *j-ésimo* al que está asociado el bloque CHECKER correspondiente.

Sin embargo, en el caso de la arquitectura con $M = 4$ y $N = 4$, el primer elemento de proceso de un grupo comienza con la evaluación de cada fila de píxeles, un ciclo después de que el último elemento de proceso dentro de ese mismo grupo haya concluido con el procesamiento de la fila de píxeles inmediatamente anterior. Este hecho determina la incorporación de dos variantes en la estructura de los bloques CHECKER con respecto al anterior caso de estudio:

- En primer lugar, se adelanta el envío de los SADs parciales hacia los respectivos bloques CHECKER en una cantidad igual al tiempo de procesamiento de media línea de píxeles. Para conseguir este propósito, se ha introducido una pequeña variación en la arquitectura interna del grupo de elementos de proceso con respecto a la presentada en la Figura 4.5, tal y como se muestra en la Figura 4.12.

- En segundo lugar, se elimina el almacenamiento en registros de los SADs correspondientes a todos los modos intermedios hasta llegar al modo 16×16. De esta manera se acelera, en cuanto al número de ciclos empleados se refiere, el cálculo de

los SADs parciales, a costa de disminuir la frecuencia máxima de funcionamiento del bloque CHECKER.

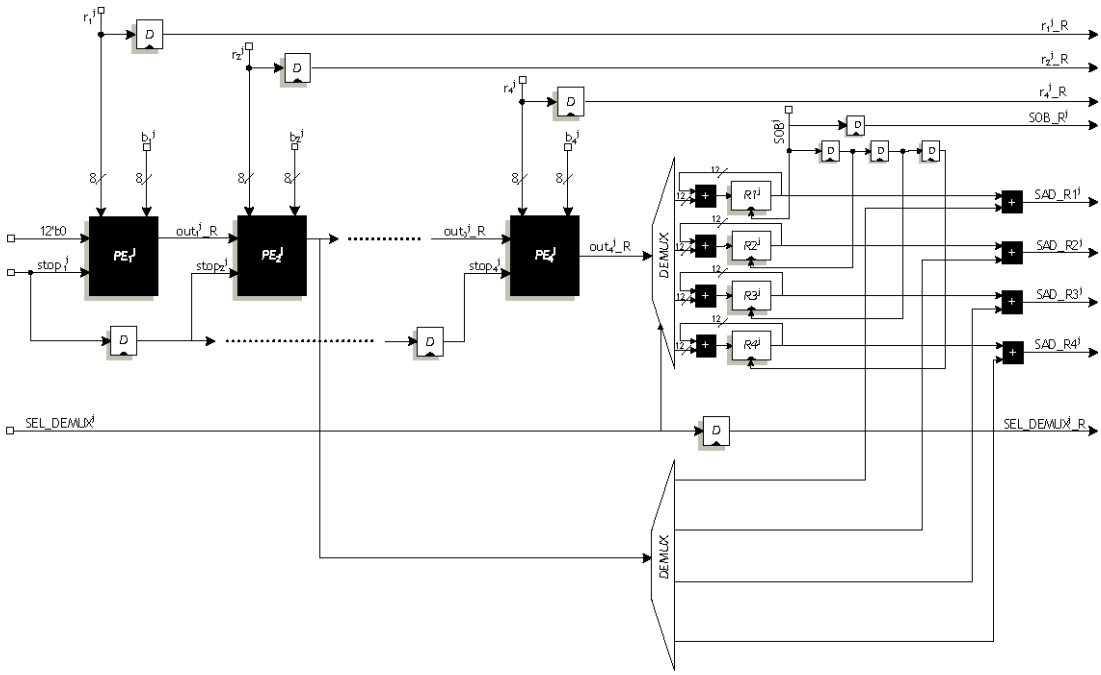


Figura 4.12: Modificación del grupo de elementos de proceso para el caso de estudio $M = 4$ y $N = 4$.

Por último, para la arquitectura de estudio con $M = 1$ y $N = 16$, no se propone ninguna arquitectura específica para el bloque CHECKER, al imposibilitar el propio flujo de datos de la arquitectura la incorporación de mecanismos de eliminación temprana a nivel de bloques de 4×4 píxeles.

4.3.3 Comparación de las arquitecturas de estudio seleccionadas

En el presente apartado, se realiza una detallada comparación de los casos de estudio seleccionados en términos de prestaciones y coste hardware asociado.

4.3.3.1 Comparación de prestaciones

Las arquitecturas de estudio expuestas presentan características muy similares en cuanto a la latencia y número de ciclos necesarios para procesar un macrobloque. Sin embargo, resulta de interés realizar un estudio comparativo acerca de la eficacia de cada arquitectura en términos de eliminación temprana de candidatos, pues este hecho influirá directamente en el número de ciclos inactivos de los elementos de proceso de las mismas. En este punto, debido a las razones anteriormente comentadas, se elimina del estudio comparativo la arquitectura totalmente agrupada considerada con $M = 1$ y $N = 16$.

Para realizar esta comparación se han utilizado un conjunto de casos significativos a partir de las secuencias descritas en el Anexo I de esta Tesis, realizándose la estimación de movimiento con cada arquitectura para diferentes valores del escalón de cuantificación y configuraciones del registro de configuración descrito en la Figura 4.9. La descripción de los diferentes casos de análisis considerados se muestra en la Tabla 4.4.

Caso	Modos activos	Tipo de comparación	Valor del registro de configuración
1L	16×16, 8×16, 16×8, 8×8 8×4, 4×8, 4×4	Al final de cada línea de 4×4 píxeles	011111111
1M	16×16, 8×16, 16×8, 8×8 8×4, 4×8, 4×4	Al final de cada dos líneas de 4×4 píxeles	101111111
2L	16×16, 8×8, 4×4	Al final de cada línea de 4×4 píxeles	011001001
2M	16×16, 8×8, 4×4	Al final de cada dos líneas de 4×4 píxeles	101001001
3L	16×16, 8×8	Al final de cada línea de 4×4 píxeles	011001000
3M	16×16, 8×8	Al final de cada dos líneas de 4×4 píxeles	101001000
3F	16×16, 8×8	Al final de cada bloque de 4×4 píxeles	111001000
4L	16×16	Al final de cada línea de 4×4 píxeles	011000000
4M	16×16	Al final de cada dos líneas de 4×4 píxeles	101000000
4F	16×16	Al final de cada bloque de 4×4 píxeles	111000000

Tabla 4.4: Casos seleccionados para la comparación arquitectural de prestaciones en términos de eliminación temprana de candidatos.

Tal y como puede observarse a partir de la mencionada tabla, los casos 1L y 1M corresponden a la activación de todos los modos posibles de estimación de movimiento definidos por el estándar H.264/AVC, mientras que en el resto de los casos sólo algunos de estos modos permanecen activos. En particular, los casos 2L y 2M corresponden a la activación de los modos de topología cuadrada dentro del estándar H.264/AVC pues, a partir de los estudios mencionados en el apartado 2.3.2.2 del anterior capítulo acerca de la disminución del número de modos activos, se desprende como conclusión general que estos modos *cuadrados* son los que mayor relevancia tienen en las prestaciones de compresión de un codificador basado en el estándar H.264/AVC. Asimismo, en los casos 3L, 3M y 3F permanecen activos los modos permitidos por las opciones avanzadas de codificación establecidas en los estándares H.263 y MPEG-4 (cuatro vectores de movimiento por macrobloque). Por último, en los casos 4L, 4M y 4F sólo se permite un vector de movimiento por macrobloque, reproduciéndose así las condiciones de los estándares de compresión de vídeo H.261, MPEG-1 y MPEG-2 en cuanto al proceso de estimación y compensación de movimiento se refiere.

Los resultados correspondientes a la arquitectura de estudio con $M = 1$ y $N = 16$ se presentan en la Figura 4.13, los cuales han sido obtenidos para un conjunto de macrobloques con diferentes características de movimiento y textura. En particular, se muestran los resultados obtenidos con macrobloques sin movimiento tanto de baja (PAMPHLET) como de alta actividad espacial (DEADLINE), así como con macrobloques de alta actividad espacial y diferentes vectores de movimiento asociados (SUZIE y TABLE). Además, para cada uno de estos casos se han reproducido los resultados, tanto para el caso de secuencias sin comprimir ($Qp = 0$) como para rangos de compresión medios y altos ($Qp = 25$ y $Qp = 50$ respectivamente) dentro del estándar H.264/AVC.

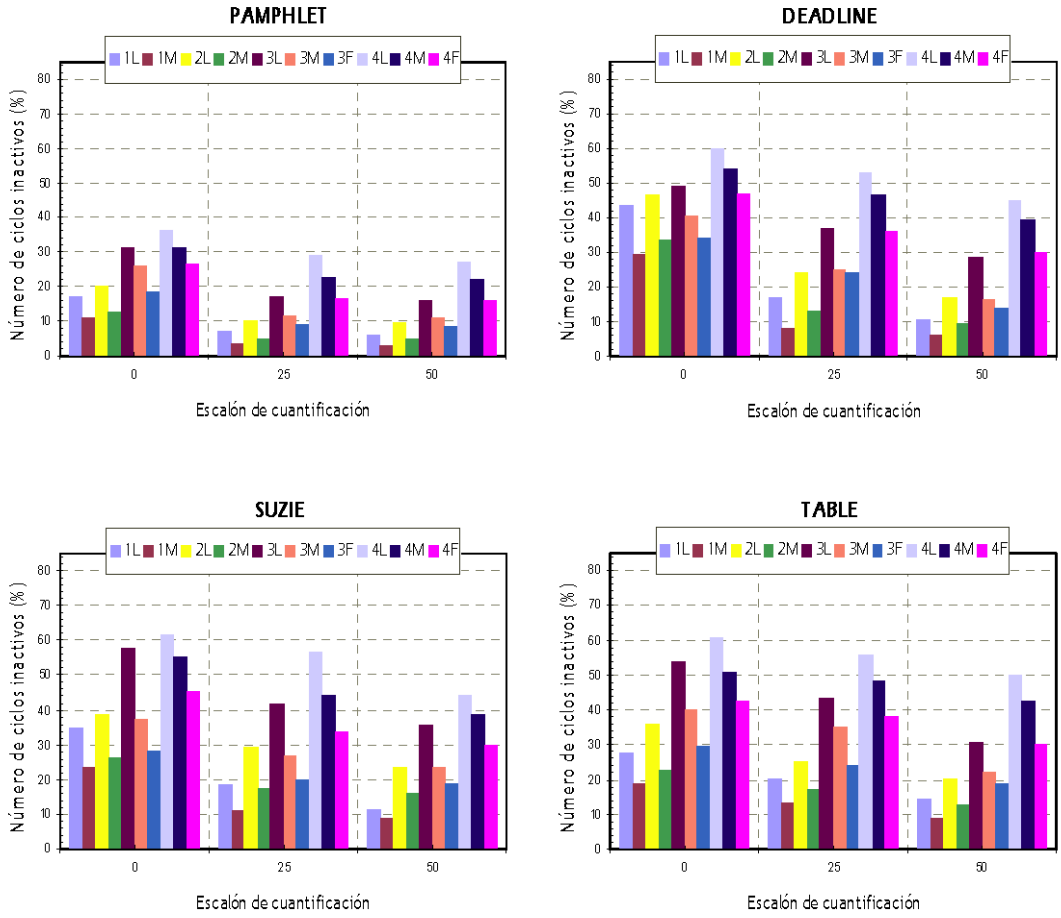


Figura 4.13: Resultados en términos de eliminación temprana de candidatos para la arquitectura de estudio con $M = 16$ y $N = 1$.

A partir de las gráficas de la Figura 4.13 se extraen las siguientes conclusiones:

- El número de ciclos inactivos disminuye sensiblemente en macrobloques de baja actividad espacial, independientemente del valor de registro de configuración. Este hecho es debido a fundamentalmente a dos razones:
  - La desviación de los SADs con respecto a la posición de mínimo es pequeña, tal y como se demostró en el capítulo anterior de esta Tesis.

- El ritmo de crecimiento del SAD para cualquiera de las posiciones evaluadas es bajo, al ser los píxeles de dichas posiciones muy similares a los del macrobloque de referencia.

- El número de ciclos inactivos disminuye a medida que aumenta el valor del escalón de cuantificación, independientemente del valor del registro de configuración. Esta afirmación está en clara concordancia con la anterior, pues a medida que aumenta el valor del escalón de cuantificación, disminuye la actividad espacial de los macrobloques del área de búsqueda.

- El número de ciclos inactivos aumenta a medida que el número de modos activos disminuye, puesto que en el bloque CHECKER de cada uno de los grupos debe satisfacerse un menor número de condiciones para detener el funcionamiento de los elementos de proceso del grupo.

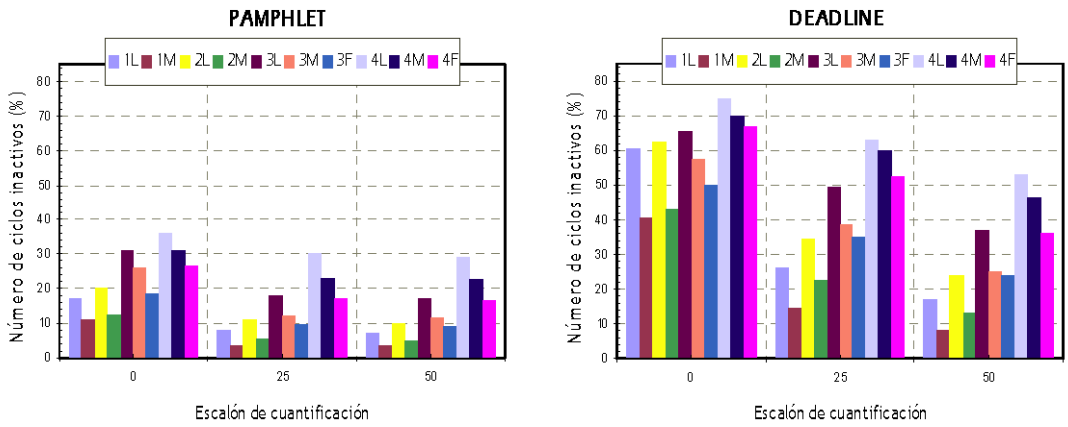
- Para un mismo número de modos activos, el número de ciclos inactivos aumenta a medida que el número de comparaciones es mayor.

Con el objetivo de comparar las prestaciones de las arquitecturas descritas, en la Figura 4.14 se presentan las mismas gráficas particularizadas para la arquitectura con $M = 4$ y $N = 4$. El análisis conjunto de las gráficas mostradas en las Figuras 4.13 y 4.14 permite establecer las siguientes conclusiones generales:

- Las afirmaciones establecidas para la arquitectura de estudio con $M = 16$ y $N = 1$, acerca de las variaciones en el porcentaje de ciclos inactivos, son totalmente extrapolables para el caso en el que $M = 4$ y $N = 4$.

- El número de ciclos inactivos depende en gran medida de la rapidez con la que se evalúe el vector ganador. Este hecho se aprecia al estudiar con detenimiento las gráficas de ambas arquitecturas para los casos de movimiento nulo (secuencias PAMPHLET y DEADLINE), en las que el porcentaje de eliminación es superior en el caso de la arquitectura con $M = 4$ y $N = 4$ que en la definida con $M = 16$ y $N = 1$, ya que en la primera de ellas se obtienen antes los SADs asociados a la posición (0,0) para cada uno de los siete posibles modos de estimación de movimiento. Esta característica contribuye a elevar los porcentajes medios de eliminación en la arquitectura con $M = 4$ y $N = 4$, pues las secuencias reales de vídeo poseen un alto grado de macrobloques sin movimiento [BTG+01].

- La antelación en el envío de los SADs parciales hacia los respectivos bloques CHECKER descrito en el apartado 4.2.2.2 de esta Tesis conlleva un deterioro mínimo en las prestaciones de la arquitectura con M=4 y N=4 en términos de eliminación temprana de candidatos para vectores de movimiento no nulos.



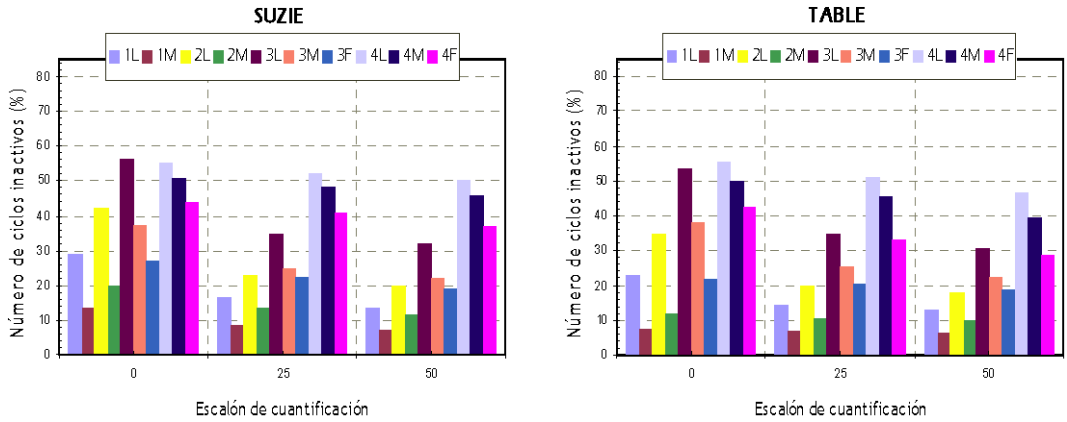


Figura 4.14: Resultados en términos de eliminación temprana de candidatos para la arquitectura de estudio con $M = 4$ y $N = 4$.

4.3.3.2 Comparación de coste hardware asociado

Con el objetivo de estimar y comparar el área ocupada por cada una de las propuestas arquitecturales presentadas, se exponen en esta sección los resultados obtenidos tras realizar la síntesis lógica preliminar de ambas arquitecturas.

Este proceso ha sido realizado haciendo uso de las herramientas de síntesis de *Synopys* con tecnología CMOS de 0,25 μm, mostrándose los resultados obtenidos estableciéndose como objetivo una frecuencia de funcionamiento para la arquitectura de 100 MHz. En particular, la Tabla 4.5 muestra el área ocupada en términos de miles de puertas NAND2 equivalentes por los bloques funcionales genéricos de ambas arquitecturas, englobando el bloque funcional MÍNIMOS al comparador de mínimos y al conjunto de M bloques CHECKER de cada arquitectura.

	Arquitectura con $M = 16$ y $N = 1$	Arquitectura con $M = 4$ y $N = 4$
GRUPO	0,57	2,10
SAD_COMPOSER	2,95	2,95
CHECKER	7,10	7,10
MÍNIMOS	122,01	36,35
Arquitectura completa	183,42	61,81

Tabla 4.5: Numero de puertas NAND2 equivalentes (miles) para cada bloque funcional y arquitectura.

Los resultados obtenidos para la arquitectura completa son absolutamente lógicos en ambos casos pues, a medida que disminuye el número de grupos, disminuye también el número de bloques funcionales SAD_COMPOSER y CHECKER a considerar.

4.3.3.3 Conclusiones globales del estudio

A partir de los resultados obtenidos para cada uno de los casos de estudio, la arquitectura unidimensional agrupada con $M = 4$ y $N = 4$ representa la mejor opción para estimadores de movimiento en sistemas de codificación H.264/AVC, debido a las siguientes razones:

- A pesar de que en la arquitectura totalmente agrupada caracterizada por $M = 1$ y $N = 16$, el número de grupos, y por lo tanto el área del circuito, es menor, ésta no puede incorporar mecanismos de eliminación temprana para bloques de 4×4 píxeles. No obstante, el flujo de datos en esta arquitectura permite eliminar candidatos a nivel de bloques de 8×8 píxeles, por lo que representa una opción válida para sistemas de codificación MPEG-4 y H.263 o bien para codificadores de vídeo H.264/AVC en los que no exista eliminación temprana de candidatos.

- La arquitectura agrupada con $M = 4$ y $N = 4$ presenta unos porcentajes medios de eliminación similares a los obtenidos a partir de la arquitectura unidimensional no

agrupada con $M = 16$ y $N = 1$, e incluso superiores para aquellos macrobloques con movimiento nulo. Este hecho determina que en secuencias de vídeo *reales* (no sintéticas), el porcentaje medio de eliminación sea superior para el caso de la arquitectura agrupada $M = 4$ y $N = 4$, al ser el vector de movimiento nulo muy frecuente.

- Estos porcentajes de eliminación se obtienen haciendo uso de un área tres veces menor que para el caso de la arquitectura no agrupada, sin que existan diferencias significativas en la latencia de ambas arquitecturas.

- Por último, la arquitectura agrupada $M = 4$ y $N = 4$ evalúa cuatro posiciones de búsqueda en paralelo. Este hecho facilita la implementación arquitectural del algoritmo VBS-ACBM propuesto en esta Tesis en el cual, gracias a los cambios descritos en el apartado 3.3.5, se inspeccionan cuatro posiciones de macrobloque dentro del área de búsqueda durante la primera fase de la búsqueda predictiva y ocho durante la segunda. La incorporación del algoritmo PBM-HW en la arquitectura seleccionada se describe en el siguiente apartado del presente capítulo.

No obstante, resulta conveniente destacar que, independientemente de la arquitectura seleccionada, se ha demostrado en esta Tesis que la posibilidad de seleccionar un subconjunto de modos activos para cada macrobloque, permite incrementar de manera drástica el porcentaje de ciclos inactivos para cualquier secuencia de vídeo.

4.3.4 Incorporación de la estimación de movimiento predictiva en la arquitectura de precisión entera

Para poder realizar el proceso de estimación de movimiento según las directrices del algoritmo VBS-ACBM con la arquitectura unidimensional agrupada seleccionada, es necesario incorporar en esta arquitectura el algoritmo PBM-HW. En este apartado se describen los cambios a realizar en la arquitectura con el objetivo de soportar el algoritmo PBM-HW, así como los resultados obtenidos en términos arquitecturales.

4.3.4.1 Modificaciones en la arquitectura de precisión entera para la implementación del algoritmo PBM-HW

Sin duda alguna, el mayor obstáculo para la implementación del algoritmo PBM-HW sobre la arquitectura seleccionada viene determinado por la propia naturaleza del algoritmo. A diferencia del algoritmo FSBM de búsqueda exhaustiva, el algoritmo PBM-HW evalúa un conjunto de posiciones que no siguen un patrón determinado, pues éstas dependen exclusivamente de vectores de movimiento previamente calculados. Este hecho determina que los elementos de proceso de los diferentes grupos no puedan compartir píxeles del área de búsqueda, como ocurría para el algoritmo de búsqueda exhaustiva. Este comportamiento se puede observar en la Tabla 4.6, en la que se muestran los píxeles procesados por cada grupo de elementos de proceso por ciclo de reloj tanto para la primera fase el algoritmo PBM-HW (ciclos en color amarillo), como para la segunda fase de refinamiento (ciclos en color azul).

Tal y como se muestra en la Tabla 4.6, durante la primera fase se evalúan las posiciones de macrobloque del área de búsqueda apuntadas por los vectores correspondientes, en la terminología propia del algoritmo PBM-HW, a los macrobloques $MB_{up,T}$, $MB_{left,T}$ y $MB_{centre,T-1}$ así como la posición correspondiente al vector cero. Los píxeles del área de búsqueda correspondientes a estas posiciones se indican en la Tabla 4.6 como b_u, b_l, b_c y b_z

respectivamente. Una vez que se obtienen, a través de los bloques SAD_COMPOSER de cada grupo, los SADs de las cuatro posiciones de 16×16 píxeles evaluadas, se procede a la fase de refinamiento a partir de las coordenadas de la posición con menor SAD asociado. En esta fase, se evalúan ocho posiciones en una distancia de un píxel alrededor del mejor vector obtenido durante la primera fase. Los píxeles del área de búsqueda correspondientes a estas ocho posiciones de refinamiento se indican en la Tabla 4.6 como b_{Rul}, b_{Ru}, b_{Rur}, b_{Rl}, b_{Rr}, b_{Rdl}, b_{Rd} y b_{Rdr}. Una vez evaluadas estas ocho posiciones, se obtiene un vector de movimiento para tamaño de bloque de 16×16 píxeles transcurriendo, desde que se iniciara la búsqueda, 221 ciclos.

T	PE_1^1	PE_2^1	...	PE_4^1	PE_1^2	...	PE_1^3	...	PE_1^4
0	$\|r0,0-b_u0,0\|$		...			...		...	
1	$\|r0,4-b_u0,4\|$	$\|r0,1-b_u0,1\|$	...		$\|r0,0-b_c0,0\|$	...		...	
2	$\|r0,8-b_u0,8\|$	$\|r0,5-b_u0,5\|$	...		$\|r0,4-b_c0,4\|$	...	$\|r0,0-b_z0,0\|$	...	
3	$\|r0,12-b_u0,12\|$	$\|r0,9-b_u0,9\|$	...	$\|r0,3-b_u0,3\|$	$\|r0,8-b_c0,8\|$	...	$\|r0,4-b_z0,4\|$	...	$\|r0,0-b_l0,0\|$
4	$\|r1,0-b_u1,0\|$	$\|r0,13-b_u0,13\|$	...	$\|r0,7-b_u0,7\|$	$\|r0,12-b_c0,12\|$	...	$\|r0,8-b_z0,8\|$	...	$\|r0,4-b_l0,4\|$
5	$\|r1,4-b_u1,4\|$	$\|r1,1-b_u1,1\|$	...	$\|r0,11-b_u0,11\|$	$\|r1,0-b_c1,0\|$	...	$\|r0,12-b_z0,12\|$	...	$\|r0,8-b_l0,8\|$
6	$\|r1,8-b_u1,8\|$	$\|r1,5-b_u1,5\|$	...	$\|r0,15-b_u0,15\|$	$\|r1,4-b_c1,4\|$	...	$\|r1,0-b_z1,0\|$	...	$\|r0,12-b_l0,12\|$
7	$\|r1,12-b_u1,12\|$	$\|r1,9-b_u1,9\|$	...	$\|r1,3-b_u1,3\|$	$\|r1,8-b_c1,8\|$	...	$\|r1,4-b_z1,4\|$	...	$\|r1,0-b_l1,0\|$
...	...	...	...	...	...	...	...	...	...
60	$\|r15,0-b_u15,0\|$	$\|r14,13-b_u14,13\|$	...	$\|r14,7-b_u14,7\|$	$\|r14,12-b_c14,12\|$	...	$\|r14,8-b_z14,8\|$	...	$\|r14,4-b_l14,4\|$
61	$\|r15,4-b_u15,4\|$	$\|r15,1-b_u15,1\|$	...	$\|r14,11-b_u14,11\|$	$\|r15,0-b_c15,0\|$	...	$\|r14,12-b_z14,12\|$	...	$\|r14,8-b_l14,8\|$
62	$\|r15,8-b_u15,8\|$	$\|r15,5-b_u15,5\|$	...	$\|r14,15-b_u14,15\|$	$\|r15,4-b_c15,4\|$	...	$\|r15,0-b_z15,0\|$	...	$\|r14,12-b_l14,12\|$
63	$\|r15,12-b_u15,12\|$	$\|r15,9-b_u15,9\|$	...	$\|r15,3-b_u15,3\|$	$\|r15,8-b_c15,8\|$	...	$\|r15,4-b_z15,4\|$	...	$\|r15,0-b_l15,0\|$
64		$\|r15,13-b_u15,13\|$	...	$\|r15,7-b_u15,7\|$	$\|r15,12-b_c15,12\|$	...	$\|r15,8-b_z15,8\|$	...	$\|r15,4-b_l15,4\|$
65			...	$\|r15,11-b_u15,11\|$		...	$\|r15,12-b_z15,12\|$	...	$\|r15,8-b_l15,8\|$
66			...	$\|r15,15-b_u15,15\|$		...		...	$\|r15,12-b_l15,12\|$
...	...	...	...	...	...	...	...	...	...
73	**16×16$_u$**		...			...		...	
74			...		**16×16$_c$**	...		...	
75			...			...	**16×16$_z$**	...	
76			...			...		...	**16×16$_l$**
...			...			...		...	
80	$\|r0,0-b_{Ru}0,0\|$		...			...		...	
81	$\|r0,4-b_{Ru}0,4\|$	$\|r0,1-b_{Ru}0,1\|$	...		$\|r0,0-b_{Ru}0,0\|$	...		...	
82	$\|r0,8-b_{Ru}0,8\|$	$\|r0,5-b_{Ru}0,5\|$	...		$\|r0,4-b_{Ru}0,4\|$	...	$\|r0,0-b_{Rur}0,0\|$	...	
83	$\|r0,12-b_{Ru}0,12\|$	$\|r0,9-b_{Ru}0,9\|$	...	$\|r0,3-b_{Ru}0,3\|$	$\|r0,8-b_{Ru}0,8\|$	...	$\|r0,4-b_{Rur}0,4\|$	...	$\|r0,0-b_{Rl}0,0\|$
84	$\|r1,0-b_{Ru}1,0\|$	$\|r0,13-b_{Ru}0,13\|$	...	$\|r0,7-b_{Ru}0,7\|$	$\|r0,12-b_{Ru}0,12\|$	...	$\|r0,8-b_{Rur}0,8\|$	...	$\|r0,4-b_{Rl}0,4\|$
85	$\|r1,4-b_{Ru}1,4\|$	$\|r1,1-b_{Ru}1,1\|$	...	$\|r0,11-b_{Ru}0,11\|$	$\|r1,0-b_{Ru}1,0\|$	...	$\|r0,12-b_{Rur}0,12\|$	...	$\|r0,8-b_{Rl}0,8\|$
86	$\|r1,8-b_{Ru}1,8\|$	$\|r1,5-b_{Ru}1,5\|$	...	$\|r0,15-b_{Ru}0,15\|$	$\|r1,4-b_{Ru}1,4\|$	...	$\|r1,0-b_{Rur}1,0\|$	...	$\|r0,12-b_{Rl}0,12\|$
87	$\|r1,12-b_{Ru}1,12\|$	$\|r1,9-b_{Ru}1,9\|$	...	$\|r1,3-b_{Ru}1,3\|$	$\|r1,8-b_{Ru}1,8\|$	...	$\|r1,4-b_{Rur}1,4\|$	...	$\|r1,0-b_{Rl}1,0\|$
...	...	...	...	...	...	...	...	...	...
140	$\|r15,0-b_{Ru}15,0\|$	$\|r14,13-b_{Ru}14,13\|$	...	$\|r14,7-b_{Ru}14,7\|$	$\|r14,12-b_{Ru}14,12\|$	...	$\|r14,8-b_{Rur}14,8\|$	...	$\|r14,4-b_{Rl}14,4\|$
141	$\|r15,4-b_{Rdl}15,4\|$	$\|r15,1-b_{Ru}15,1\|$	...	$\|r14,11-b_{Ru}14,11\|$	$\|r15,0-b_{Ru}15,0\|$	...	$\|r14,12-b_{Rur}14,12\|$	...	$\|r14,8-b_{Rl}14,8\|$
142	$\|r15,8-b_{Rdl}15,8\|$	$\|r15,5-b_{Ru}15,5\|$	...	$\|r14,15-b_{Ru}14,15\|$	$\|r15,4-b_{Ru}15,4\|$	...	$\|r15,0-b_{Rur}15,0\|$	...	$\|r14,12-b_{Rl}14,12\|$
143	$\|r15,12-b_{Rdl}15,12\|$	$\|r15,9-b_{Ru}15,9\|$	...	$\|r15,3-b_{Ru}15,3\|$	$\|r15,8-b_{Ru}15,8\|$	...	$\|r15,4-b_{Rur}15,4\|$	...	$\|r15,0-b_{Rl}15,0\|$
144	$\|r0,0-b_{Rdl}0,0\|$	$\|r15,13-b_{Ru}15,13\|$	...	$\|r15,7-b_{Ru}15,7\|$	$\|r15,12-b_{Ru}15,12\|$	...	$\|r15,8-b_{Rur}15,8\|$	...	$\|r15,4-b_{Rl}15,4\|$
145	$\|r0,4-b_{Rdl}0,4\|$	$\|r0,1-b_{Rdl}0,1\|$	...	$\|r15,11-b_{Ru}15,11\|$	$\|r0,0-b_{Rd}0,0\|$	...	$\|r15,12-b_{Rur}15,12\|$	...	$\|r15,8-b_{Rl}15,8\|$
146	$\|r0,8-b_{Rdl}0,8\|$	$\|r0,5-b_{Rdl}0,5\|$	...	$\|r15,15-b_{Ru}15,15\|$	$\|r0,4-b_{Rd}0,4\|$	...	$\|r0,0-b_{Rdr}0,0\|$	...	$\|r15,12-b_{Rl}15,12\|$
147	$\|r0,12-b_{Rdl}0,12\|$	$\|r0,9-b_{Rdl}0,9\|$	...	$\|r0,3-b_{Rdl}0,3\|$	$\|r0,8-b_{Rd}0,8\|$	...	$\|r0,4-b_{Rdr}0,4\|$	...	$\|r0,0-b_{Rr}0,0\|$
148	$\|r1,0-b_{Rdl}1,0\|$	$\|r0,13-b_{Rdl}0,13\|$	...	$\|r0,7-b_{Rdl}0,7\|$	$\|r0,12-b_{Rd}0,12\|$	...	$\|r0,8-b_{Rdr}0,8\|$	...	$\|r0,4-b_{Rr}0,4\|$

149	$\|r1,4-b_{Rdl}1,4\|$	$\|r1,1-b_{Rdl}1,1\|$	...	$\|r0,11-b_{Rdl}0,11\|$	$\|r1,0-b_{Rd}1,0\|$	...	$\|r0,12-b_{Rdr}0,12\|$	...	$\|r0,8-b_{Rr}0,8\|$
150	$\|r1,8-b_{Rdl}1,8\|$	$\|r1,5-b_{Rdl}1,5\|$	...	$\|r0,15-b_{Rdl}0,15\|$	$\|r1,4-b_{Rd}1,4\|$	...	$\|r1,0-b_{Rdr}1,0\|$	...	$\|r0,12-b_{Rr}0,12\|$
151	$\|r1,12-b_{Rdl}1,12\|$	$\|r1,9-b_{Rdl}1,9\|$	...	$\|r1,3-b_{Rdl}1,3\|$	$\|r1,8-b_{Rd}1,8\|$	...	$\|r1,4-b_{Rdr}1,4\|$	...	$\|r1,0-b_{Rr}1,0\|$
...	...	...	...	...	...	...	...	...	...
204	$\|r15,0-b_{Rdl}15,0\|$	$\|r14,13-b_{Rdl}14,13\|$	...	$\|r14,7-b_{Rdl}14,7\|$	$\|r14,12-b_{Rd}14,12\|$	...	$\|r14,8-b_{Rdr}14,8\|$	...	$\|r14,4-b_{Rr}14,4\|$
205	$\|r15,4-b_{Rdl}15,4\|$	$\|r15,1-b_{Rdl}15,1\|$	...	$\|r14,11-b_{Rdl}14,11\|$	$\|r15,0-b_{Rd}15,0\|$	...	$\|r14,12-b_{Rdr}14,12\|$	...	$\|r14,8-b_{Rr}14,8\|$
206	$\|r15,8-b_{Rdl}15,8\|$	$\|r15,5-b_{Rdl}15,5\|$	...	$\|r14,15-b_{Rdl}14,15\|$	$\|r15,4-b_{Rd}15,4\|$	...	$\|r15,0-b_{Rdr}15,0\|$	...	$\|r14,12-b_{Rr}14,12\|$
207	$\|r15,12-b_{Rdl}15,12\|$	$\|r15,9-b_{Rdl}15,9\|$	...	$\|r15,3-b_{Rdl}15,3\|$	$\|r15,8-b_{Rd}15,8\|$	...	$\|r15,4-b_{Rdr}15,4\|$	...	$\|r15,0-b_{Rr}15,0\|$
208		$\|r15,13-b_{Rdl}15,13\|$	...	$\|r15,7-b_{Rdl}15,7\|$	$\|r15,12-b_{Rd}15,12\|$	...	$\|r15,8-b_{Rdr}15,8\|$	...	$\|r15,4-b_{Rr}15,4\|$
209			...	$\|r15,11-b_{Rdl}15,11\|$		...	$\|r15,12-b_{Rdr}15,12\|$	...	$\|r15,8-b_{Rr}15,8\|$
210			...	$\|r15,15-b_{Rdl}15,15\|$		...		...	$\|r15,12-b_{Rr}15,12\|$

Tabla 4.6: Diagrama de tiempos simplificado del módulo de estimación de movimiento con precisión entera para el algoritmo PBM-HW.

Para realizar las operaciones indicadas en la Tabla 4.6, la solución propuesta en esta Tesis consiste, en primer lugar, en diseñar la memoria de búsqueda como una memoria de único puerto de 32 palabras de 32 bytes cada una. En cada palabra de esta memoria se almacena una fila completa de píxeles correspondientes al área de búsqueda. De esta manera, una vez conocida la coordenada vertical del primer vector a evaluar durante la primera fase, se lee de memoria la fila de píxeles correspondiente a dicha coordenada. A partir de la coordenada horizontal del vector candidato se eligen, mediante el uso de un multiplexor a la entrada del grupo de elementos de proceso, los 16 píxeles correspondientes a la posición a evaluar de entre los 32 píxeles leídos. Estos píxeles son almacenados en registros de desplazamiento para ser utilizados durante los siguientes cuatro ciclos de reloj. Este proceso se repite para los grupos de elementos de proceso consecutivos, utilizando cada grupo de elementos de proceso el único puerto de la memoria durante un ciclo de reloj.

Una vez que el primer grupo de elementos de proceso ha concluido con los píxeles de la primera fila de la posición que tiene que evaluar, lee de la memoria y comienza a procesar los píxeles de la segunda fila de dicha posición. Este procedimiento se realiza hasta que se termina con la primera fase de la búsqueda predictiva, repitiéndose de igual manera durante la fase de refinamiento. Las modificaciones introducidas en el grupo de cuatro elementos de proceso para realizar el procesamiento de datos establecido en la Tabla 4.6 se resumen, de manera esquematizada, en la Figura 4.15. Por sencillez, en esta figura se ha obviado la línea de registros destinada a la transferencia de los píxeles de referencia entre grupos.

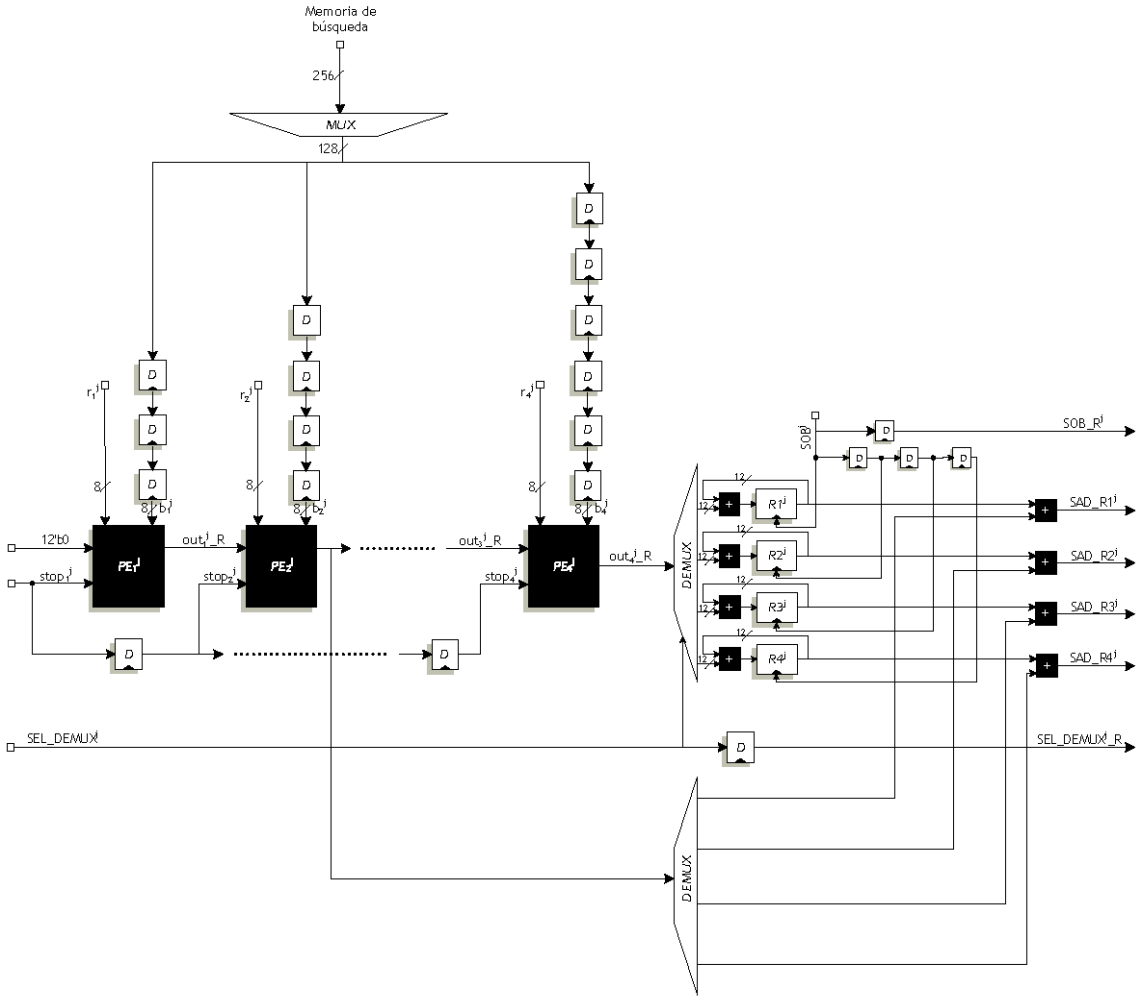


Figura 4.15: Grupo de elementos de proceso modificado para la implementación del algoritmo PBM-HW.

4.3.4.2 Resultados obtenidos

Los cambios introducidos con el objetivo de implementar el algoritmo PBM-HW sobre la arquitectura seleccionada determinan que, bajo las mismas condiciones de síntesis impuestas en la obtención de los resultados mostrados en la Tabla 4.5, el número de puertas de la arquitectura propuesta en esta Tesis asciende a 70,93K NAND2 equivalentes.

Es importante reseñar que, para cada una de las doce posiciones de 16×16 píxeles del área de búsqueda evaluadas por el algoritmo PBM-HW, se ha obtenido el SAD asociado a partir de la suma de los bloques de 4×4 píxeles correspondientes a estas posiciones. Este hecho ha posibilitado que, haciendo uso de los bloques SAD_COMPOSER y del comparador de mínimos multimodo propuestos en esta Tesis, se haya obtenido durante este proceso de búsqueda un primer conjunto de valores de los SADs correspondientes a todos los modos activos. A su vez, estos valores de SAD calculados resultan ser, por lo general, cercanos a los correspondientes a las posiciones de mínimo para cada uno de los modos activos. Este hecho contribuye de manera decisiva a que, para aquellos macrobloques en los que el algoritmo VBS-ACBM decide que tras la búsqueda predictiva es necesario realizar una búsqueda exhaustiva, el porcentaje de ciclos inactivos en los elementos de proceso aumente considerablemente. En particular, para los casos seleccionados en la Figura 4.14, se obtiene un incremento en el número de modos inactivos entre el 15% y el 25% por término medio, dependiendo de la secuencia y del número de modos activos.

Por último es de destacar que, para completar el proceso de estimación de movimiento según el algoritmo VBS-ACBM propuesto en esta Tesis en la arquitectura propuesta, sólo sería necesario incluir el cálculo del *Intra_SAD* y la gestión de los vectores utilizados durante la primera fase del algoritmo predictivo PBM-HW. El hecho de que no se haya considerado la lógica de cómputo necesaria para ello se debe a que, típicamente, los codificadores de vídeo basados en algún estándar de compresión realizan total o parcialmente esta tarea. De hecho, los estándares H.263 [H263, ap.III, pp.15] y MPEG-4 [Gar04, pp.111] recomiendan utilizar la media aritmética de los valores de píxel del macrobloque para decidir si éste es codificado como *INTRA* o *INTER*, mientras que en codificadores H.264/AVC dicha media es utilizada durante la *intra predicción* 16×16 [H264,pp.102]. Asimismo, en los tres estándares mencionados se realiza la codificación diferencial de vectores de movimiento que engloban a los elegidos por el algoritmo PBM-HW dentro del fotograma bajo procesamiento y que, por lo tanto, están disponibles para su uso por parte del estimador de movimiento [H263, pp.45],

[MPEG4, pp.178], [H.264, pp. 115-117]. En cualquier caso, si estos valores no fueran calculados por otros módulos del codificador correspondiente, su inclusión dentro de la arquitectura propuesta podría realizarse sin excesiva dificultad.

4.4 Refinamiento sub-píxel de vectores de movimiento

El módulo de refinamiento sub-píxel propuesto en esta Tesis realiza el refinamiento de vectores de movimiento a coordenadas de medio y cuarto de píxel para cualquiera de los modos definidos por los estándares H.263 y H.264/AVC. Para ello, este módulo de refinamiento calcula las muestras de medio y cuarto de píxel según los filtros de interpolación propuestos por el estándar H.263 (interpolación bilineal) o H.264/AVC (filtro FIR de seis coeficientes), para a continuación proceder al propio refinamiento de vectores de movimiento, independientemente del estimador de movimiento con precisión entera utilizado. Asimismo, gracias a las modificaciones introducidas en el apartado 3.3.5 en cuanto al refinamiento de vectores en el algoritmo PBM, la etapa de refinamiento que se presenta es totalmente compatible con el algoritmo VBS-ACBM introducido en esta Tesis.

Debido a que los vectores de movimiento con precisión entera pueden apuntar a cualquier zona dentro del área de búsqueda, es imposible reutilizar en este módulo los resultados del refinamiento de los vectores correspondientes a un modo de estimación de movimiento en el cálculo de los vectores de otro modo cualquiera. Este hecho determina, sin lugar a dudas, la complejidad arquitectural de este módulo al ser necesario evaluar un gran número de posiciones que, aunque se encuentran en un área de búsqueda muy reducida, no están relacionadas entre sí. Bajo estas consideraciones, y con el objetivo de realizar el refinamiento de vectores de movimiento de manera eficiente, el módulo de refinamiento sub-píxel propuesto opera dentro de un mismo modo de estimación de movimiento a nivel de bloques

de 4×4 píxeles, obteniendo los vectores de topologías superiores para un macrobloque cualquiera mediante la correcta combinación de los resultados parciales obtenidos.

La funcionalidad inherente a este proceso se ha dividido en esta Tesis entre dos sub-módulos: refinamiento de medio píxel y refinamiento de cuarto de píxel, tal y como se indica en la Figura 4.1. De esta manera se obtiene un alto grado de paralelismo durante el proceso de refinamiento, al permitir que inmediatamente después de que el sub-módulo de refinamiento de medio píxel finalice el procesamiento de un modo de estimación de movimiento, se proceda al refinamiento de cuarto de píxel de los vectores correspondientes a este modo mientras se comienzan a refinar a coordenadas de medio píxel los vectores asociados al siguiente modo. De manera análoga al módulo de estimación de movimiento con precisión entera cabe señalar que, a diferencia de las escasas aportaciones realizadas por otros autores, las etapas de refinamiento de medio y cuarto de píxel propuestas en esta Tesis son adaptables a las necesidades del estándar en uso y al número de modos activos, además de incorporar la eliminación temprana de candidatos de manera sencilla. Independientemente de estas características, el módulo de refinamiento sub-píxel propuesto en esta Tesis representa una aportación arquitectural significativa, al permitir refinar vectores de movimiento a coordenadas de cuarto de píxel para todos los modos definidos por el estándar H.264/AVC. A continuación, se analizan las características principales de la arquitecturas propuestas para ambos sub-módulos, y que han sido publicadas en el congreso internacional *SPIE Microtechnologies for the New Millenium 2005* [LTV+05b].

4.4.1 Sub-módulo de refinamiento de medio píxel

El sub-módulo de refinamiento de medio píxel propuesto en esta Tesis recibe los vectores de movimiento calculados por el módulo de estimación de movimiento con precisión entera, junto con sus correspondientes SADs. A partir de esta información, calcula las muestras de

medio píxel necesarias de acuerdo con el estándar en uso y evalúa las ocho posiciones que se encuentran a una distancia de medio píxel de cada uno de los vectores con precisión entera recibidos, obteniendo como resultado final un conjunto de vectores de movimiento de mayor precisión que los originales.

La arquitectura del sub-módulo de refinamiento de medio píxel multiestándar que se presenta en esta Tesis está compuesta, de modo genérico, por el conjunto de bloques funcionales que se muestra en la Figura 4.16. Tal y como se aprecia en dicha figura, el sub-módulo de refinamiento de medio píxel recibe 16 vectores de movimiento de precisión entera (*mv_4×4a_ip* ... *mv_4×4p_ip*) con sus correspondientes SADs asociados (*SAD_4×4a_ip* ... *SAD_4×4p_ip*), independientemente del modo de estimación de movimiento al que correspondan dichos vectores. Así, en el caso en el que se estén refinando los vectores de movimiento correspondientes al modo 16×16, se recibirán 16 vectores de movimiento iguales junto con 16 SADs del mismo valor, mientras que para el caso del modo 4×4 todas estas entradas podrán tener, a priori, valores diferentes.

A partir de las coordenadas de los vectores de movimiento recibidos, se realiza la interpolación de las muestras de precisión de medio píxel para cada uno de los bloques de 4×4 píxeles utilizando los interpoladores de medio píxel. Finalmente, una vez interpoladas las muestras de medio píxel, se realiza el refinamiento mediante la evaluación de ocho posiciones de medio píxel alrededor de la posición apuntada por el vector de precisión entera. Dicha evaluación de posiciones de medio píxel es realizada por una arquitectura sistólica unidimensional no agrupada de ocho elementos de proceso, obteniéndose como resultado 16 vectores de movimiento con precisión de medio píxel (*mv_4×4a_hp* ... *mv_4×4p_hp*) junto con sus respectivos SADs (*SAD_4×4a_hp* ... *SAD_4×4p_hp*). La estructura interna del mencionado sistólico de elementos de proceso, al igual que la del resto de los bloques

funcionales que componen el módulo de refinamiento de medio píxel, se detalla en los sucesivos apartados.

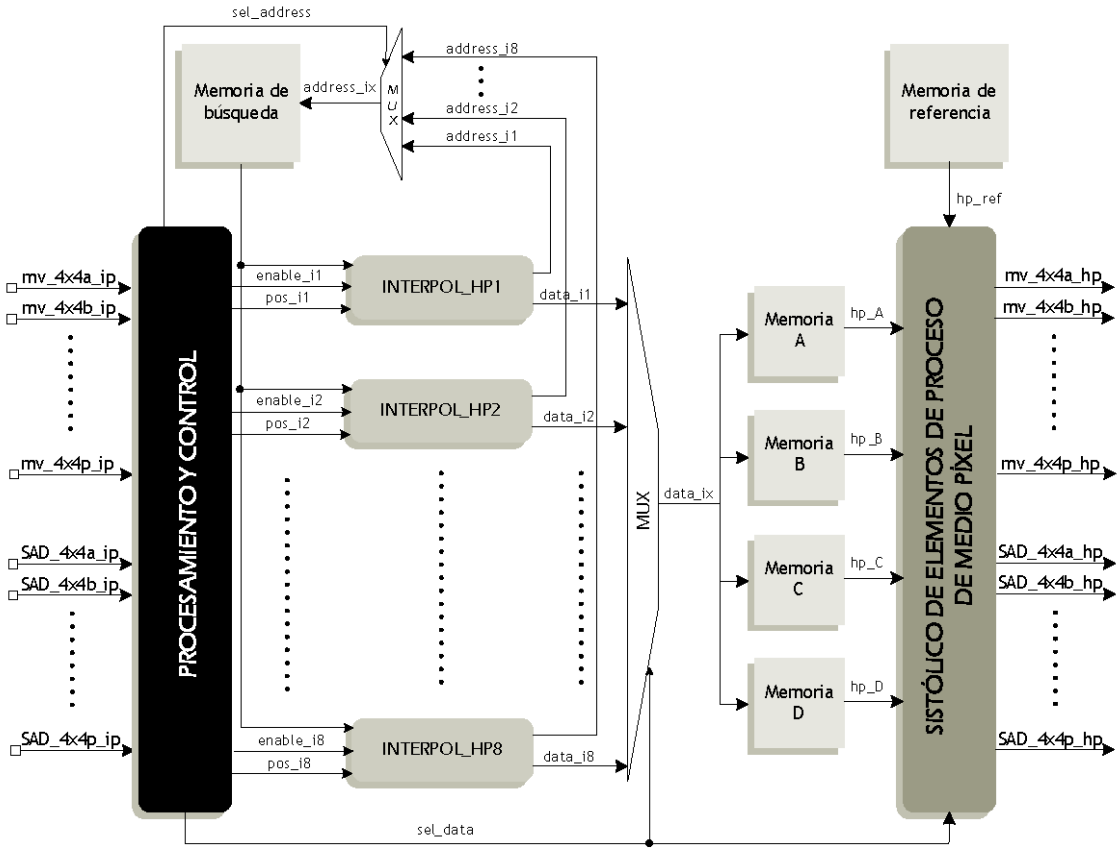


Figura 4.16: Arquitectura del módulo de refinamiento de medio píxel a nivel de bloques funcionales.

4.4.1.1 Interpolador de medio píxel (INTERPOL_HP)

Este bloque funcional se encarga de interpolar las muestras de medio píxel necesarias para realizar el refinamiento del vector de movimiento correspondiente a un bloque de 4×4 píxeles, con independencia del estándar de compresión de vídeo que se utilice.

En este punto es necesario considerar dentro del sub-módulo de refinamiento de medio píxel dos nuevas variables de configuración que indiquen si el refinamiento debe ser realizado

(estándares con precisión de vectores de movimiento superior a un píxel), y en su caso, cómo debe ser realizado (método de interpolación para calcular las muestras de medio píxel). Para ello, se han añadido dos nuevos campos en el registro de configuración que indican si el estándar permite el refinamiento de vectores de movimiento a coordenadas de medio píxel (*Half Pixel Refinement - HPR*) y si las muestras de medio píxel deben ser calculadas mediante el filtro de seis coeficientes introducido por el estándar H.264/AVC o mediante la clásica interpolación bilineal de estándares anteriores (*F/B*), tal y como se muestra en la Figura 4.17. De esta manera, en esta Tesis se pretende no sólo diseñar módulos de interpolación multiestándar, sino de modo genérico, acomodar de manera sencilla dentro de la arquitectura nuevos métodos de interpolación que permiten obtener resultados muy similares a los ofrecidos por la interpolación establecida por los estándares, pero con un coste hardware considerablemente menor [LYL+05].

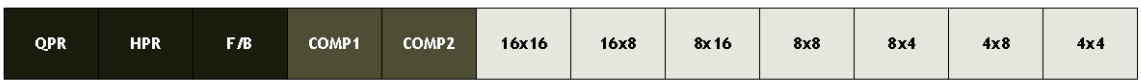


Figura 4.17: Campos del registro de configuración modificado.

Asimismo, se ha introducido un tercer campo que indica si se debe realizar el refinamiento de vectores a coordenadas de cuarto de píxel (*Quarter Pixel Refinement - QPR*), aspecto a tener en cuenta por el módulo de refinamiento de cuarto de píxel.

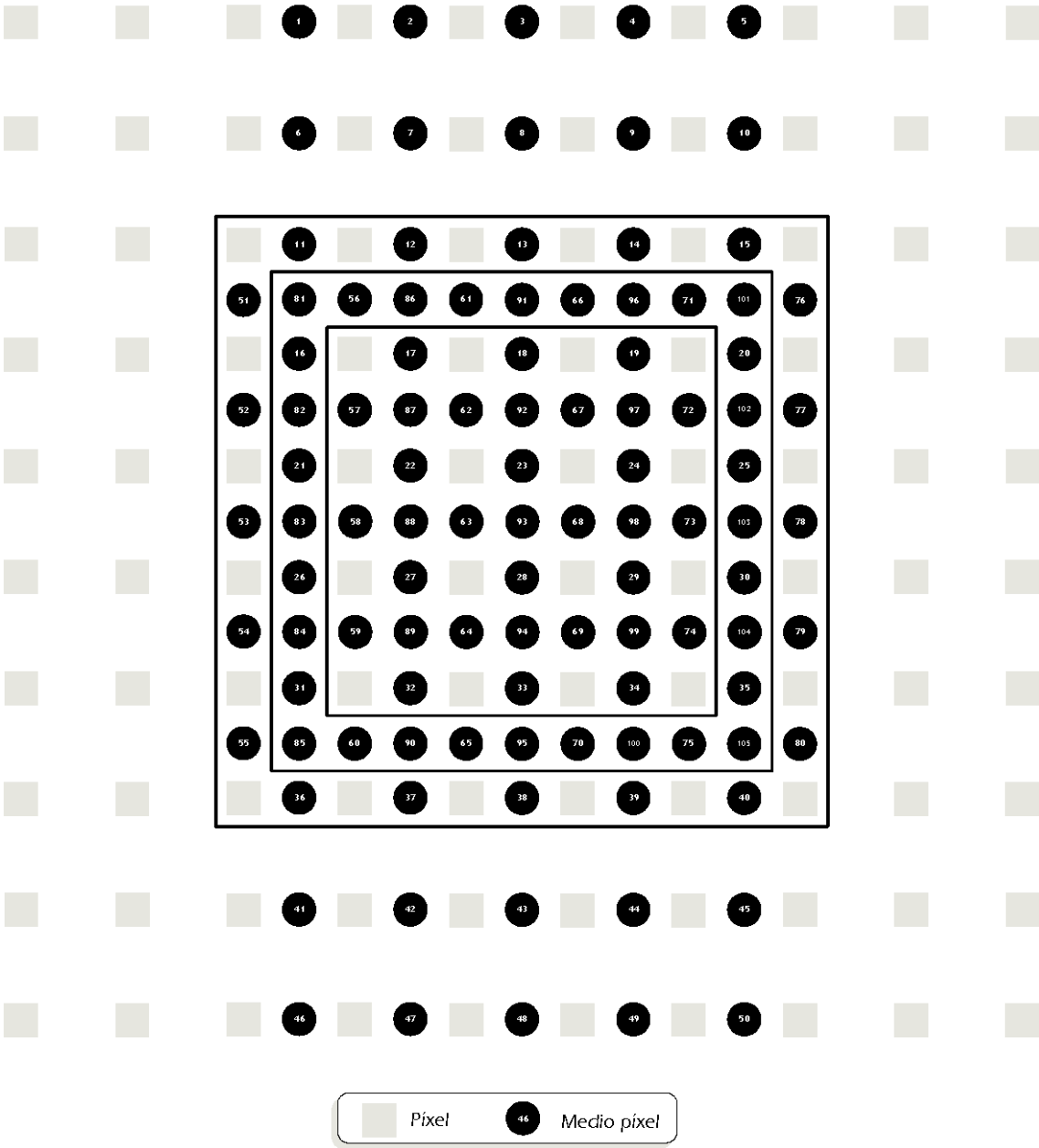


Figura 4.18: Distribución de los píxeles utilizados y las muestras de medio píxel calculadas por un bloque interpolador del módulo de refinamiento de medio píxel.

Por claridad, se detallará el funcionamiento del bloque interpolador considerando que el cálculo de las muestras de medio píxel se realiza según el filtrado definido por el estándar H.264/AVC, indicándose al final del presente apartado las variaciones introducidas para la realización de la interpolación bilineal. En este contexto, las muestras de medio píxel que

calcula un bloque interpolador, junto con los píxeles necesarios para su cálculo, se muestran en la Figura 4.18, indicándose asimismo el orden en el que dichas muestras de medio píxel son interpoladas. En la mencionada figura, los píxeles hacia los que apunta el vector de movimiento con precisión entera quedan encuadrados en el recuadro interior mientras que las muestras de medio píxel necesarias para realizar el proceso de refinamiento en sí, aparecen bajo el recuadro intermedio.

Tal y como se deduce a partir de la Figura 4.18, cada interpolador calcula en primer lugar las muestras de medio píxel *horizontales* (de la muestra 1 a la 50), a continuación las muestra de medio píxel *verticales* (de la muestra 51 a la 80) y por último las muestras de medio píxel ubicadas entre cuatro píxeles, o *diagonales* (de la muestra 81 a la 105), pues estas últimas tienen que ser necesariamente obtenidas a partir de muestras de medio píxel previamente calculadas. Para el cálculo de estas muestras, independientemente de que éstas sean *horizontales*, *verticales* o *diagonales*, cada bloque interpolador presenta una estructura de filtrado como la mostrada en la Figura 4.19. Mediante el uso de esta estructura de interpolación se realiza el cálculo de las muestras de medio píxel según las ecuaciones establecidas en el apartado 2.1.2.3.2, usando exclusivamente sumas y desplazamientos.

Una vez finalizado el proceso de interpolación, todos los píxeles y muestras de medio píxel que se encuentran dentro del recuadro exterior en la Figura 4.18 se envían al sub-módulo de refinamiento de cuarto de píxel, pues tal y como se detallará posteriormente, estas muestras son necesarias para realizar dicha operación de refinamiento. Durante este proceso de transferencia, las muestras de medio píxel bajo el recuadro intermedio de la Figura 4.18 se almacenan en cuatro memorias que se utilizarán como entradas a la estructura sistólica de ocho elementos de proceso encargada de completar el refinamiento de vectores.

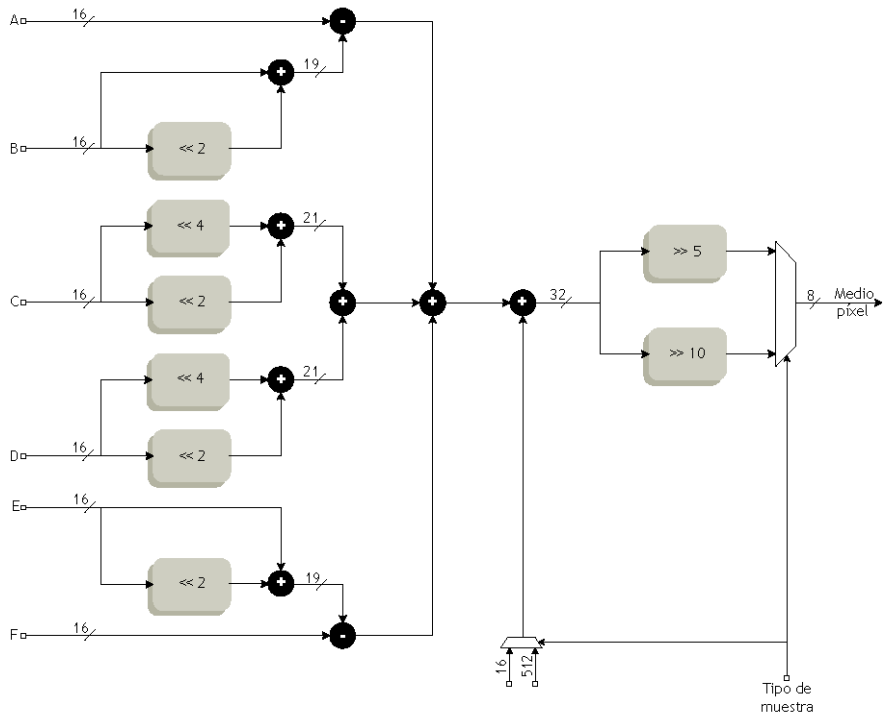


Figura 4.19: Estructura de filtrado de cada bloque interpolador de medio píxel.

4.4.1.1.1 Modificaciones arquitecturales para la incorporación del algoritmo de interpolación bilineal

A continuación se describen las modificaciones consideradas en el bloque de interpolación con el objetivo de soportar, no sólo el algoritmo de filtrado para la interpolación de muestras de medio píxel fijado por el estándar H.264/AVC, sino también el algoritmo bilineal establecido por los estándares predecesores. Mediante la configuración del bit *F/B* en el registro de configuración, la arquitectura propuesta permite realizar el refinamiento de vectores de movimiento a coordenadas de medio píxel en codificadores de vídeo basados en los estándares H.263, MPEG-4 y H.264/AVC, lo cual representa una considerable novedad con respecto a trabajos previos.

Para garantizar este comportamiento multiestándar, se ha añadido al bloque de interpolación de medio píxel una estructura de cómputo similar a la mostrada en la Figura 4.19 que permita realizar el cálculo de las muestras de medio píxel según las ecuaciones definidas en el apartado 2.1.2.3.1 de esta Tesis. La distribución de las muestras de medio píxel calculas y de las muestras de precisión entera correspondientes a un bloque de 4×4 píxeles, se muestra en la Figura 4.20, donde los recuadros mantienen el mismo significado que en la Figura 4.18. En este caso, el procedimiento a realizar por el bloque interpolador resulta ser considerablemente más sencillo debido, fundamentalmente, a las siguientes razones:

- Para interpolar una muestra de medio píxel sólo se necesitan dos píxeles (muestras *horizontales* y *verticales*) o cuatro píxeles (muestras *diagonales*) en lugar de los seis píxeles requeridos por el estándar H.264/AVC.

- Todas las muestras de medio píxel se calculan sin necesidad de reutilizar muestras de medio píxel anteriormente calculadas, lo que determina que el número de muestras de medio píxel a calcular se reduzca en un 19% con respecto al caso en el que se aplica el filtrado establecido por el estándar H.264/AVC.

Estos hechos determinan que el número de ciclos necesario para calcular las muestras de medio píxel correspondientes a un bloque de 4×4 píxeles mediante interpolación bilineal sea menor, y que por lo tanto, la temporización que determina el bloque de procesamiento y control sea ligeramente diferente. En cualquier caso, aunque el número de ciclos cambie, el funcionamiento global del módulo de refinamiento de medio píxel sigue siendo el mismo, manteniéndose igualmente el número total de bloques interpoladores.

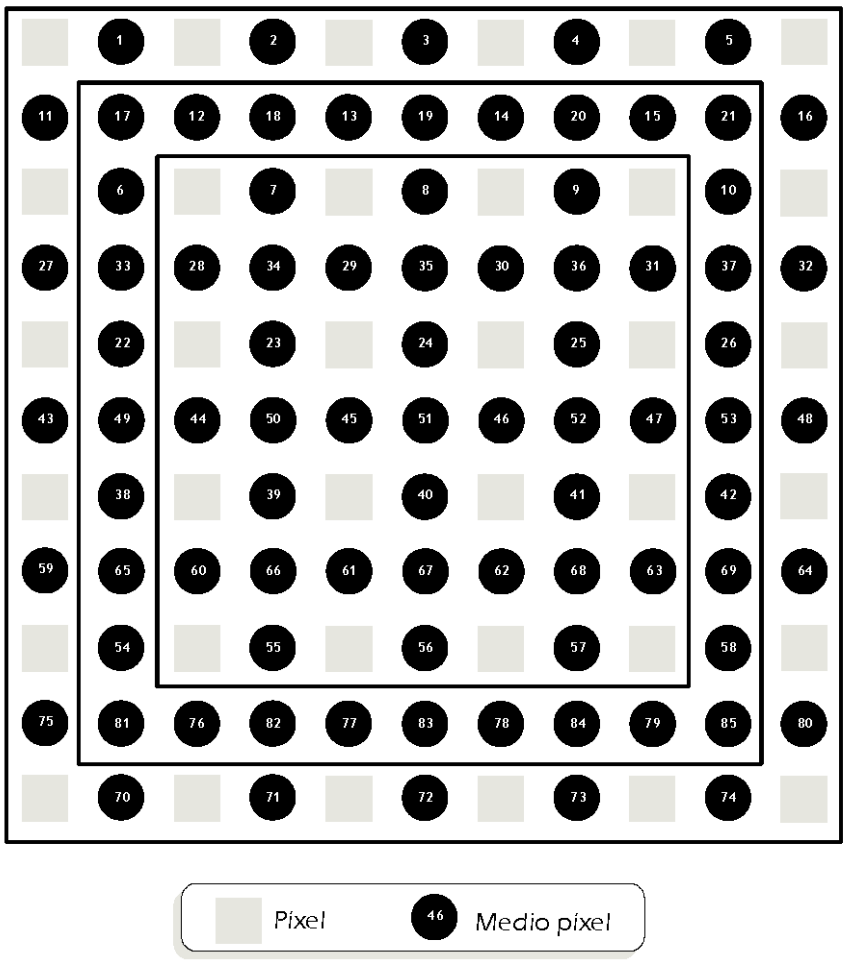


Figura 4.20: Distribución de los píxeles utilizados y las muestras de medio píxel calculadas por un bloque interpolador mediante interpolación bilineal.

4.4.1.2 Sistólico de elementos de proceso de medio píxel

Este bloque funcional está compuesto fundamentalmente por una arquitectura sistólica unidimensional no agrupada de ocho elementos de proceso, encargándose del refinamiento de vectores una vez calculadas las correspondientes muestras de medio píxel. Cada uno de los elementos de proceso evalúa una de las ocho posibles posiciones de refinamiento de medio píxel, obteniéndose de esta manera la arquitectura de ocho grupos de un sólo elemento de

proceso (M = 1 y N = 8) que se muestra en la Figura 4.21, señalándose claramente la dirección de refinamiento asociada a cada elemento de proceso.

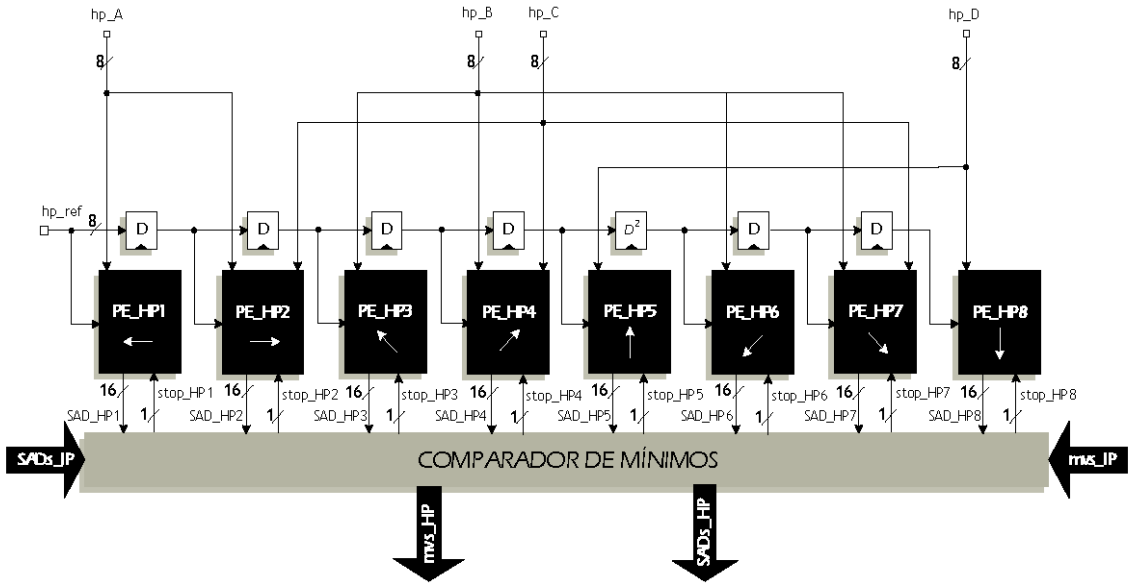


Figura 4.21: Arquitectura sistólica de refinamiento de vectores de movimiento a coordenadas de medio píxel con M=1 y N=8.

Dentro de la arquitectura propuesta, cada elemento de proceso se encarga de la evaluación de una sola posición de refinamiento, tal y como indica la Tabla 4.7, en la que se muestran las muestras de medio píxel y los píxeles de referencia procesados por cada uno de los elementos de proceso de la arquitectura para el refinamiento de un bloque genérico de 4×4 píxeles. A partir del estudio de la mencionada tabla, se puede observar que la distribución de los elementos de proceso y de las muestras de medio píxel en cuatro memorias no es ni mucho menos casual, obteniéndose un alto grado de reutilización de los píxeles de referencia y de las muestras de medio píxel de búsqueda mediante la introducción de seis retardos unitarios y una unidad de retardo doble entre los elementos de proceso.

T	hp_ref	hp_A	hp_B	hp_C	hp_D	PE_HP1	PE_HP2	PE_HP3	PE_HP4	PE_HP5	PE_HP6	PE_HP7	PE_HP8
0	r0,0	hp(16)	...	...	...	|r0,0 - hp(16)|	...	...	...	...	...	...	...
1	r0,1	hp(17)	...	...	...	|r0,1 - hp(17)|	|r0,0 - hp(17)|	...	...	...	...	...	...
2	r0,2	hp(18)	hp(81)	...	...	|r0,2 - hp(18)|	|r0,1 - hp(18)|	|r0,0 - hp(81)|	...	...	...	...	...
3	r0,3	hp(19)	hp(86)	...	...	|r0,3 - hp(19)|	|r0,2 - hp(19)|	|r0,1 - hp(86)|	|r0,0 - hp(86)|	...	...	...	...
4	r1,0	hp(21)	hp(91)	hp(20)	hp(56)	|r1,0 - hp(21)|	|r0,3 - hp(20)|	|r0,2 - hp(91)|	|r0,1 - hp(91)|	|r0,0 - hp(56)|	...	...	...
5	r1,1	hp(22)	hp(96)	...	hp(61)	|r1,1 - hp(22)|	|r1,0 - hp(22)|	|r0,3 - hp(96)|	|r0,2 - hp(96)|	|r0,1 - hp(61)|	...	...	...
6	r1,2	hp(23)	hp(82)	hp(101)	hp(66)	|r1,2 - hp(23)|	|r1,1 - hp(23)|	|r1,0 - hp(82)|	|r0,3 - hp(101)|	|r0,2 - hp(66)|	|r0,0 - hp(82)|	...	...
7	r1,3	hp(24)	hp(87)	...	hp(71)	|r1,3 - hp(24)|	|r1,2 - hp(24)|	|r1,1 - hp(87)|	|r1,0 - hp(87)|	|r0,3 - hp(71)|	|r0,1 - hp(87)|	|r0,0 - hp(87)|	...
8	r2,0	hp(26)	hp(92)	hp(25)	hp(57)	|r2,0 - hp(26)|	|r1,3 - hp(25)|	|r1,2 - hp(92)|	|r1,1 - hp(92)|	|r1,0 - hp(57)|	|r0,2 - hp(92)|	|r0,1 - hp(92)|	|r0,0 - hp(57)|
9	r2,1	hp(27)	hp(97)	...	hp(62)	|r2,1 - hp(27)|	|r2,0 - hp(27)|	|r1,3 - hp(97)|	|r1,2 - hp(97)|	|r1,1 - hp(62)|	|r0,3 - hp(97)|	|r0,2 - hp(97)|	|r0,1 - hp(62)|
10	r2,2	hp(28)	hp(83)	hp(102)	hp(67)	|r2,2 - hp(28)|	|r2,1 - hp(28)|	|r2,0 – hp(83)|	|r1,3 - hp(102)|	|r1,2 - hp(67)|	|r1,0 - hp(83)|	|r0,3 - hp(102)|	|r0,2 - hp(67)|
11	r2,3	hp(29)	hp(88)	...	hp(72)	|r2,3 - hp(29)|	|r2,2 - hp(29)|	|r2,1 - hp(88)|	|r2,0 – hp(88)|	|r1,3 - hp(72)|	|r1,1 - hp(88)|	|r1,0 - hp(88)|	|r0,3 – hp(72)|
12	r3,0	hp(31)	hp(93)	hp(30)	hp(58)	|r3,0 - hp(31)|	|r2,3 - hp(30)|	|r2,2 - hp(93)|	|r2,1 - hp(93)|	|r2,0 – hp(58)|	|r1,2 - hp(93)|	|r1,1 - hp(93)|	|r1,0 - hp(58)|
13	r3,1	hp(32)	hp(98)	...	hp(63)	|r3,1 - hp(32)|	|r3,0 - hp(32)|	|r2,3 - hp(98)|	|r2,2 - hp(98)|	|r2,1 - hp(63)|	|r1,3 - hp(98)|	|r1,2 - hp(98)|	|r1,1 - hp(63)|
14	r3,2	hp(33)	hp(84)	hp(103)	hp(68)	|r3,2 - hp(33)|	|r3,1 - hp(33)|	|r3,0 - hp(84)|	|r2,3 - hp(103)|	|r2,2 - hp(68)|	|r2,0 – hp(84)|	|r1,3 - hp(103)|	|r1,2 - hp(68)|
15	r3,3	hp(34)	hp(89)	...	hp(73)	|r3,3 - hp(34)|	|r3,2 - hp(34)|	|r3,1 - hp(89)|	|r3,0 - hp(89)|	|r2,3 - hp(73)|	|r2,1 - hp(89)|	|r2,0 – hp(89)|	|r1,3 - hp(73)|
16	...	...	hp(94)	hp(35)	hp(59)	...	|r3,3 - hp(35)|	|r3,2 - hp(94)|	|r3,1 - hp(94)|	|r3,0 - hp(59)|	|r2,2 - hp(94)|	|r2,1 - hp(94)|	|r2,0 – hp(59)|
17	...	...	hp(99)	...	hp(64)	...	...	|r3,3 - hp(99)|	|r3,2 - hp(99)|	|r3,1 - hp(64)|	|r2,3 - hp(99)|	|r2,2 - hp(99)|	|r2,1 - hp(64)|
18	...	...	hp(85)	hp(104)	hp(69)	...	...	...	|r3,3 - hp(104)|	|r3,2 - hp(69)|	|r3,0 - hp(85)|	|r2,3 - hp(104)|	|r2,2 - hp(69)|
19	...	...	hp(90)	...	hp(74)	...	...	...	...	|r3,3 - hp(74)|	|r3,1 - hp(90)|	|r3,0 - hp(90)|	|r2,3 - hp(74)|
20	...	...	hp(95)	...	hp(60)	...	...	...	...	...	|r3,2 - hp(95)|	|r3,1 - hp(95)|	|r3,0 - hp(60)|
21	...	...	hp(100)	...	hp(65)	...	...	...	...	...	|r3,3 - hp(100)|	|r3,2 - hp(100)|	|r3,1 - hp(65)|
22	...	...	hp(105)	...	hp(70)	...	...	...	...	...	...	|r3,3 - hp(105)|	|r3,2 - hp(70)|
23	...	...	...	...	hp(75)	...	...	...	...	...	...	...	|r3,3 - hp(75)|

Tabla 4.7: Diagrama de tiempos del sistólico de elementos de proceso de medio píxel para un bloque genérico de 4×4 píxeles.

La arquitectura de un elemento de proceso de refinamiento de medio píxel genérico (dos entradas para las muestra de medio píxel del área de búsqueda) se muestra en la Figura 4.22. En esta arquitectura se ha incorporado, de forma análoga a lo establecido para los elementos de proceso del módulo de estimación de movimiento con precisión entera, eliminación temprana de candidatos mediante la comparación del SAD acumulado con el SAD transmitido por la etapa de precisión de píxel y la posterior activación de la señal *stop_HPi*. En este caso, al ser imposible reutilizar los resultados obtenidos en la evaluación de topologías de bloque pequeñas para la evaluación de tamaños de bloque mayores, el proceso de comparación y posterior eliminación es considerablemente más sencillo.

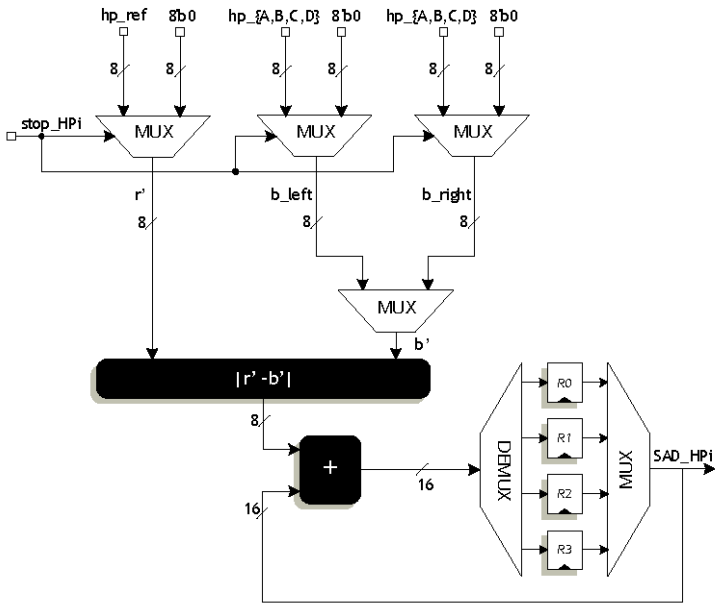


Figura 4.22: Elemento de proceso genérico de la arquitectura sistólica de refinamiento de medio píxel.

Puesto que cada elemento de proceso opera a nivel de bloques independientes de 4×4 píxeles y sin embargo, debe ser capaz de evaluar la posición de refinamiento para cualquiera de las siete topologías de bloque establecidas por el estándar H.264/AVC, resulta imprescindible incorporar dentro de la estructura básica del elemento de proceso mecanismos que permitan realizar dicho cálculo de manera eficiente. Con este objetivo, el elemento de proceso de refinamiento de medio píxel propuesto en esta Tesis presenta cuatro registros que permiten almacenar los SADs parciales y totales para cada topología de bloque de la manera que muestra la Figura 4.23. Así, se facilita enormemente el refinamiento de vectores de movimiento para cualquier modo de estimación de movimiento y la eliminación temprana de candidatos, que se realiza mediante simple comparación de los valores almacenados en estos registros con los SADs enviados por la etapa de estimación de movimiento con precisión entera. En esta figura también se indica, mediante círculos, el momento en el que cada uno de estos registros se inicializa.

Cada elemento de proceso evalúa el SAD asociado a su dirección de refinamiento de los 16 bloques de 4×4 píxeles en orden lexicográfico, almacenando los resultados obtenidos en los registros correspondientes según lo mencionado anteriormente. Este proceso se repetirá para todos los modos de estimación de movimiento activos resultando, en el peor de los casos, que cada uno de los elementos de proceso de la arquitectura sistólica de refinamiento de medio píxel debe procesar 112 bloques de 4×4 píxeles con el objetivo de obtener, mediante la asociación de los ocho elementos de proceso, un total de 41 vectores de movimiento de medio píxel de precisión.

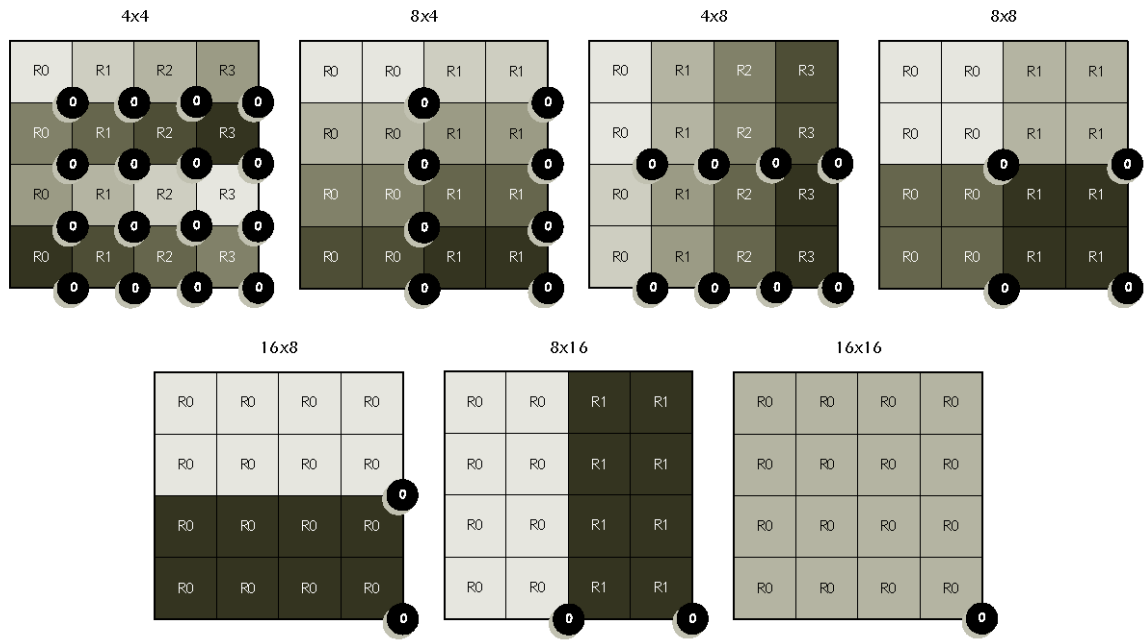


Figura 4.23: Almacenamiento de SADs en un elemento de proceso de medio píxel para los diferentes modos de estimación de movimiento.

4.4.1.3 Procesamiento y control

Este bloque funcional se encarga de procesar la información enviada por el módulo de estimación de movimiento con precisión entera para, de acuerdo a esta información y al valor

del registro de configuración, realizar de manera eficiente el control global del sub-módulo de refinamiento de medio píxel.

En este sentido, una vez que se han recibido los vectores de movimiento con precisión de píxel, y siempre y cuando haya sido activado el refinamiento de medio píxel en el registro de configuración, se comienza con el refinamiento de éstos para los modos de estimación de movimiento activos, desde el modo activo de topología de bloque *menor* (4×4) hasta la topología de bloque *mayor* (16×16). Para cada uno de ellos, el refinamiento comienza mediante la activación de los bloques de interpolación, cada uno de los cuales procesa un bloque de 4×4 píxeles independientemente del modo de estimación de movimiento bajo análisis. Con el objetivo de evitar la presencia de *ciclos muertos* durante el proceso de refinamiento, así como obtener un alto grado de utilización del hardware del módulo de medio píxel, en esta Tesis se propone un uso escalonado de ocho bloques de interpolación, tal y como se indica en el diagrama de tiempos mostrado en la Figura 4.24. En esta figura, el número de ciclos que se muestra representa el caso correspondiente a realizar la interpolación de medio píxel mediante el filtro establecido por el estándar H.264/AVC.

El número de interpoladores seleccionado permite que la arquitectura de elementos de proceso empiece con el refinamiento de un nuevo bloque de 4×4 píxeles un ciclo después de haber terminado con el bloque actual, maximizándose así sus prestaciones. Asimismo, gracias a la estrategia diseñada, sólo transcurren siete ciclos de reloj desde que cada interpolador termina de procesar un bloque de 4×4 píxeles hasta que es habilitado de nuevo para comenzar con un nuevo bloque de píxeles permitiendo, una vez superada la latencia inicial, obtener los vectores de todos los modos de estimación de movimiento activos con un alto grado de eficiencia de acuerdo a los recursos hardware empleados.

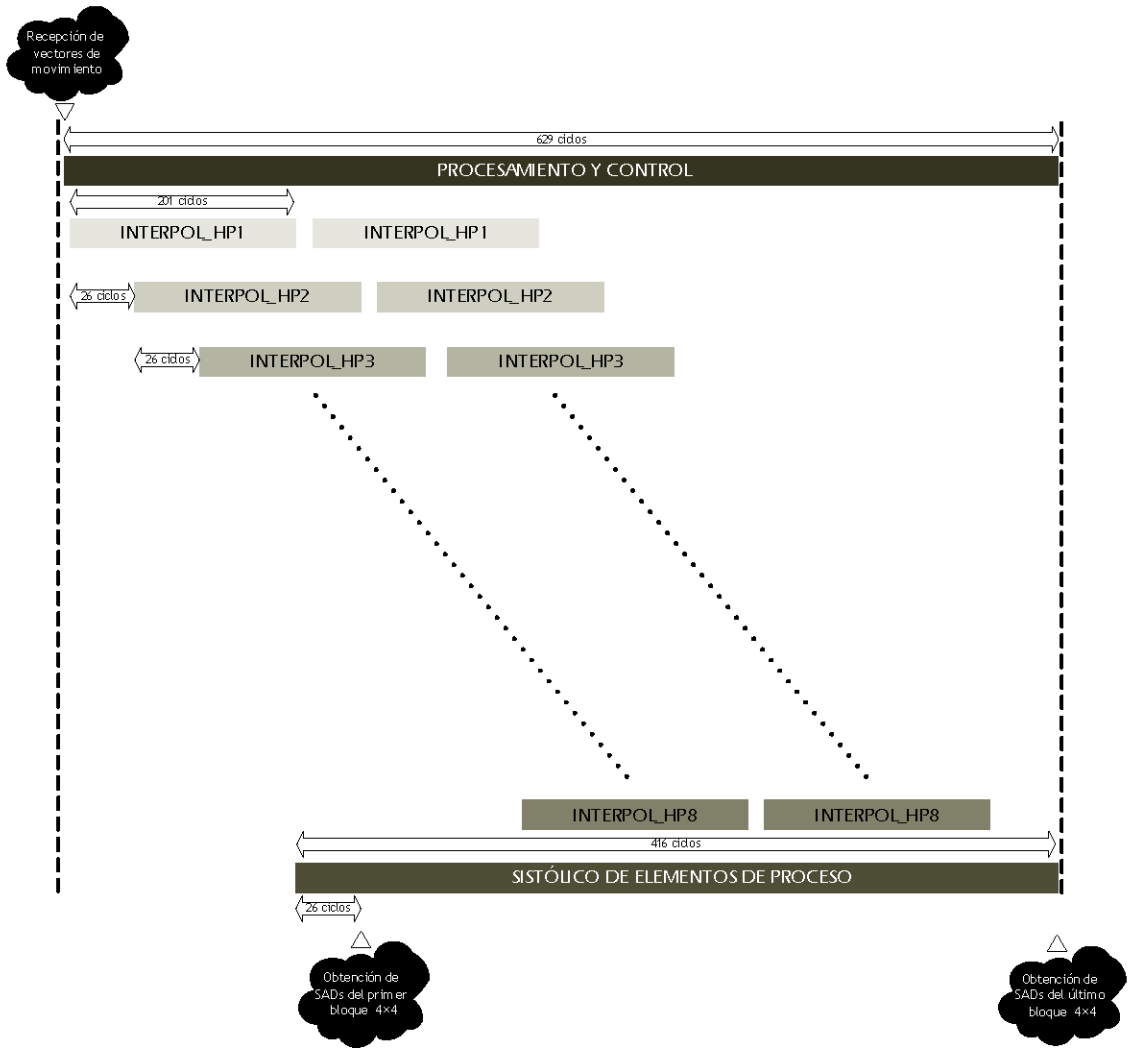


Figura 4.24: Diagrama de tiempos de refinamiento de vectores a coordenadas de medio píxel para un modo de estimación de movimiento cualquiera.

4.4.2 Sub-módulo de refinamiento de cuarto de píxel

El sub-módulo de refinamiento de cuarto de píxel representa la última etapa dentro del proceso de estimación de movimiento. En la arquitectura propuesta en esta Tesis, éste recibe del sub-módulo de refinamiento de medio píxel, tanto los vectores de movimiento a refinar (*mv_4×4a_hp* ... *mv_4×4p_hp*), como los píxeles (*s_4×4a_ip* ... *s_4×4p_ip*) y muestras

de medio píxel (*s_4×4a_hp* ... *s_4×4p_hp*) necesarias para realizar el proceso de interpolación de las muestras de cuarto de píxel. A partir de estas muestras de cuarto de píxel, este sub-módulo procede al refinamiento de vectores a coordenadas de cuarto de píxel mediante la arquitectura mostrada en la Figura 4.25.

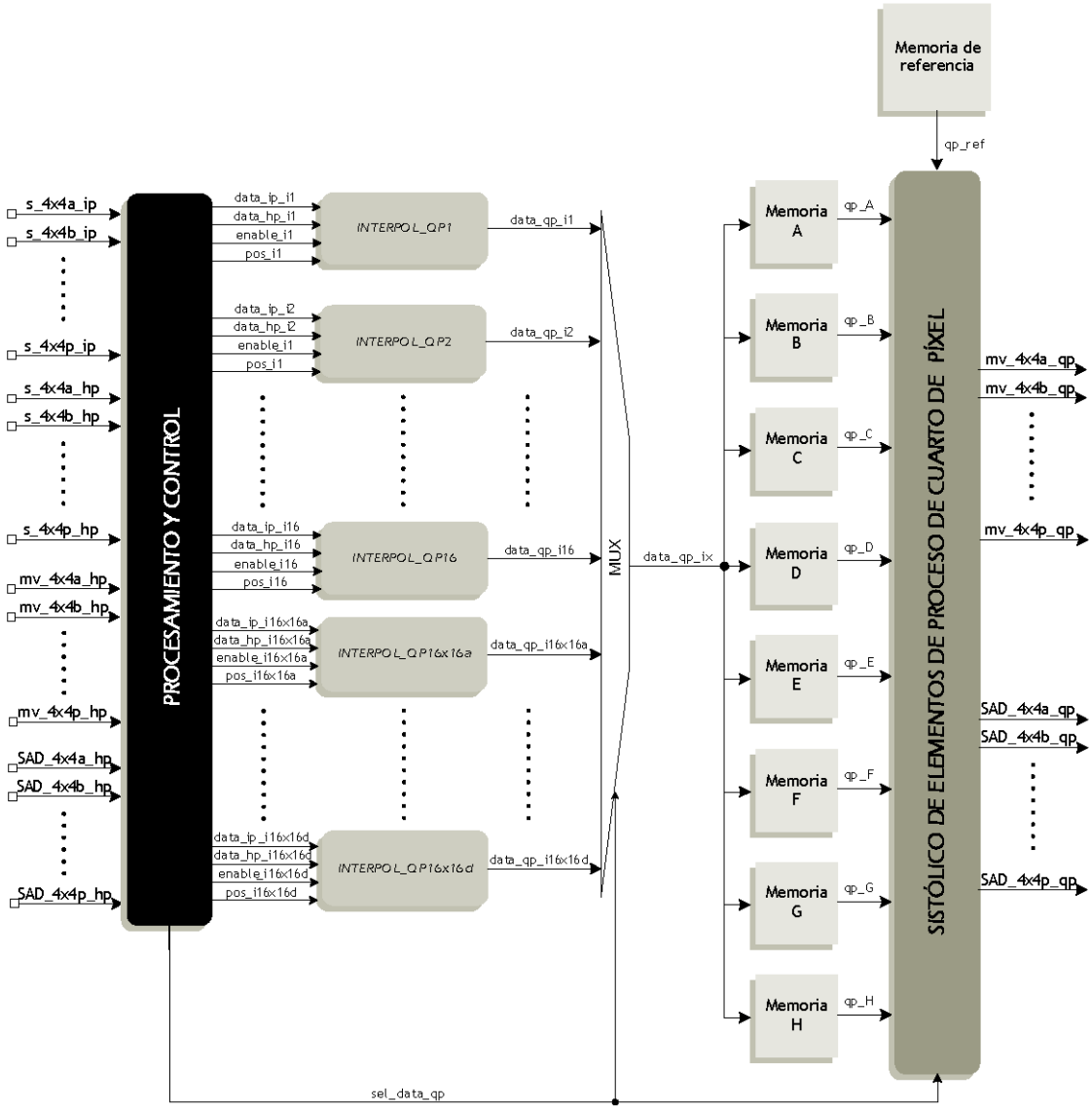


Figura 4.25: Arquitectura del módulo de refinamiento de cuarto de píxel a nivel de bloques funcionales.

Para completar el proceso de estimación de movimiento con eficacia, el sub-módulo de refinamiento de cuarto de píxel opera de acuerdo a la misma filosofía que el módulo de

refinamiento de medio píxel, obteniéndose los vectores correspondientes a cada uno de los modos de estimación de movimiento activos a partir del análisis individual de los 16 bloques de 4×4 píxeles que componen un macrobloque, independientemente del modo de estimación de movimiento bajo análisis. En este sentido, la arquitectura de la etapa de refinamiento de cuarto de píxel aportada en esta Tesis (mostrada en la Figura 4.25) es muy similar a la del sub-módulo de refinamiento de medio píxel anteriormente expuesta, así como su esquema general de funcionamiento a nivel de bloques funcionales.

El sub-módulo de refinamiento de cuarto de píxel recibe, para cada bloque de 4×4 píxeles, las muestras de píxel y medio píxel que se encuentran dentro del recuadro exterior de la Figura 4.18 o de la Figura 4.20, dependiendo de si el cálculo de las muestras de medio píxel se ha realizado de acuerdo al filtro de seis coeficientes propuesto por el estándar H.264/AVC, o bien mediante interpolación bilineal, respectivamente. Una vez recibido el correspondiente vector de movimiento, se realiza la interpolación de las muestras de cuarto de píxel que sean necesarias para proceder al refinamiento por parte del sistólico de ocho elementos de proceso, obteniéndose como resultado final un vector de movimiento en coordenadas de cuarto de píxel, con su correspondiente SAD asociado, para cada uno de los bloques de 4×4 píxeles que constituyen un macrobloque.

Al igual que para el resto de módulos que componen la arquitectura de estimación de movimiento multiestándar propuesta en esta Tesis, en los siguientes apartados se detalla el funcionamiento y la estructura interna de cada uno de los bloques funcionales del módulo de refinamiento de cuarto de píxel definidos en la Figura 4.25. Es de destacar que el módulo de refinamiento de cuarto de píxel, y en consecuencia, los bloques funcionales que a continuación se detallan, realizará su función sólo en el caso en el que campo *QPR* (*Quarter Pixel Refinement*) del registro de configuración esté activado. En caso contrario, el módulo de refinamiento de cuarto de píxel no realizará función alguna, finalizando el proceso de estimación con el refinamiento de medio píxel descrito en el apartado anterior.

4.4.2.1 Interpolador de cuarto de píxel (INTERPOL_QP)

Este bloque funcional se encarga de interpolar las muestras de cuarto de píxel necesarias para realizar el refinamiento del vector de movimiento correspondiente a un bloque de 4×4 píxeles. Puesto que la compensación de movimiento a coordenadas de cuarto de píxel sólo posee carácter obligatorio en el estándar H.264/AVC, cada uno de los interpoladores realiza el cálculo de las muestras de cuarto de píxel según la interpolación bilineal establecida por dicho estándar y que ha sido descrita en el capítulo 2 de esta Tesis.

Para realizar su función, cada uno de los bloques interpoladores recibe los píxeles y las muestras de medio píxel enviadas por el módulo de refinamiento de medio píxel correspondiente al bloque de 4×4 píxeles que se está procesando. A continuación, y una vez que se ha calculado y enviado por parte del módulo de refinamiento de medio píxel el vector de movimiento, se procede a realizar el cálculo de las muestras de cuarto de píxel. De esta manera, única y exclusivamente se calculan las muestras de cuarto de píxel que son necesarias para el proceso de refinamiento, pues en el caso de no esperar por el vector de movimiento de medio píxel, sería necesario interpolar un 134% de muestras adicionales o lo que es lo mismo 2752 muestras de cuarto de píxel de más por macrobloque y modo de estimación de movimiento activo (19264 muestras adicionales en el caso de estar todos los modos de estimación de movimiento activos). A modo de ejemplo de este proceso de cálculo, en la Figura 4.26 se muestran las 128 muestras de cuarto de píxel calculadas por un bloque interpolador para un bloque de 4×4 píxeles cualquiera, así como la distribución de los datos en las memorias correspondientes. En este caso particular, tras realizarse el proceso de refinamiento de medio píxel, el vector de movimiento recibido posee un desplazamiento de (-1,-1) medios píxeles con respecto al vector de movimiento de precisión entera.

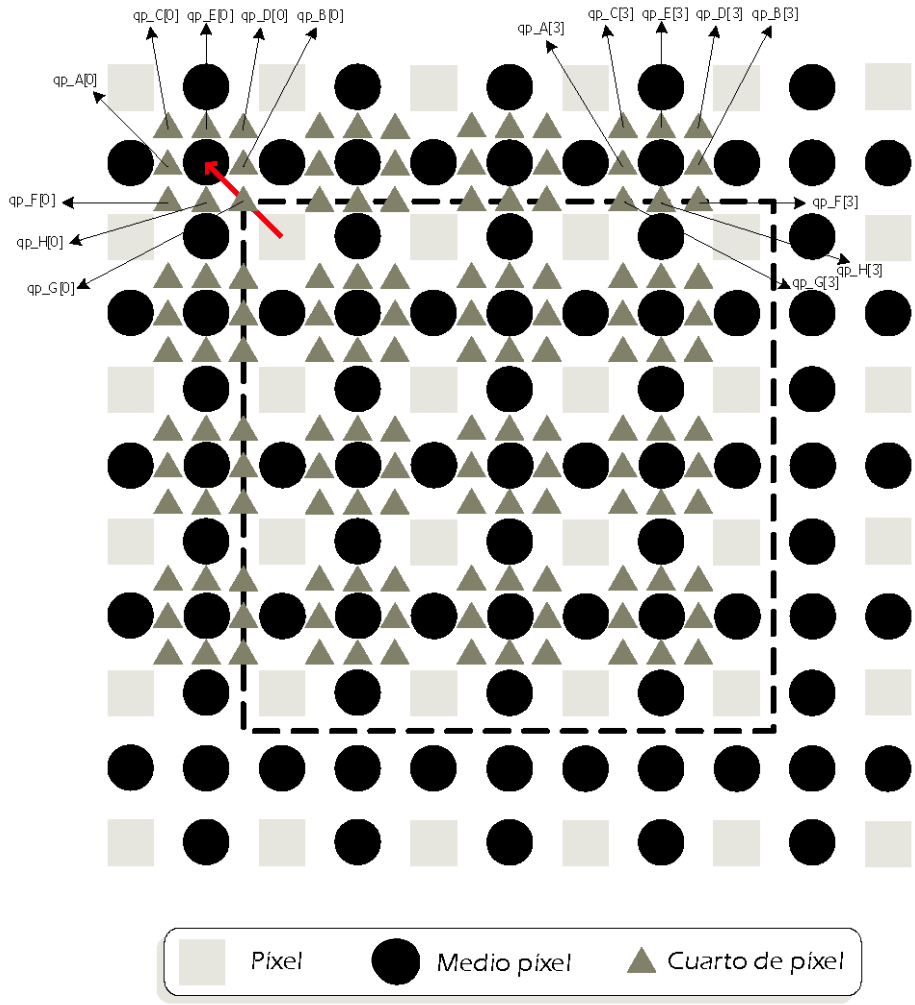


Figura 4.26: Distribución de las muestras de píxel y medio píxel utilizadas, y muestras de cuarto de píxel calculadas por un bloque interpolador de cuarto de píxel para un desplazamiento determinado.

En principio, para realizar la interpolación de las muestras de cuarto de píxel según el procedimiento descrito, se necesitan 16 interpoladores de cuarto de píxel, de manera que cada uno de ellos se encarga de procesar un bloque de 4×4 píxeles para cada modo de estimación de movimiento activo. Debido a las razones anteriormente comentadas, cada bloque interpolador empezará a calcular las muestras correspondientes al bloque de píxeles bajo análisis una vez que haya recibido su vector de movimiento correspondiente. Sin embargo, esta circunstancia conlleva la existencia de un singular *conflicto* cuando se encuentran activos los modos de estimación de movimiento 8×16 y 16×16. En esta situación,

al estar activo el modo 8×16, el primer SAD válido se obtiene una vez procesados los catorce primeros bloques de 4×4 píxeles, ya que se sigue un orden de procesamiento lexicográfico y el primer vector 8×16 se obtiene tras completar el procesamiento de dos columnas enteras. Hasta que no se conozca el valor del mencionado vector de movimiento, los interpoladores necesitan mantener los píxeles y muestras de medio píxel correspondientes a los mencionados bloques de 4×4 píxeles. El conflicto surge debido a que el módulo de refinamiento de medio píxel no espera a que se calculen los dos vectores de movimiento del modo 8×16 para empezar con el cálculo de las muestras de medio píxel correspondientes al modo 16×16. Por esta razón, cuando se esté procesando el catorceavo bloque de 4×4 píxeles correspondiente al modo 8×16, seis de los ocho bloques interpoladores de medio píxel ya habrán comenzado con el procesamiento del modo 16×16, y por lo tanto, estarán transmitiendo datos útiles al módulo de refinamiento de cuarto de píxel.

Con el objetivo de no aumentar la latencia de la etapa de refinamiento sub-píxel por la existencia de un caso particular que no siempre se produce (modos 8×16 y 16×16 activos), la solución adoptada en esta Tesis ha sido considerar cuatro nuevos bloques de interpolación de cuarto de píxel, reflejados en la arquitectura del módulo de refinamiento de cuarto de píxel de la Figura 4.25 como *INTERPOL_QP16×16a* ... *INTERPOL_QP16×16d*. De esta manera, las muestras de cuarto de píxel correspondientes a los cuatro primeros bloques de 4×4 píxeles del modo de estimación de movimiento 16×16 serán calculadas por parte de estos interpoladores, encargándose el resto de los bloques interpoladores del quinto bloque de píxeles en adelante.

4.4.2.2 Sistólico de elementos de proceso de cuarto de píxel

Este bloque funcional está compuesto, fundamentalmente, por una arquitectura sistólica unidimensional no agrupada de ocho elementos de proceso, encargándose del refinamiento de vectores una vez calculadas las correspondientes muestras de cuarto de píxel por parte de los bloques interpoladores. Cada uno de los elementos de proceso evalúa una de las ocho posibles posiciones de refinamiento de cuarto de píxel, obteniéndose de esta manera la arquitectura de ocho grupos de un sólo elemento de proceso ($M = 1$ y $N = 8$) mostrada en la Figura 4.27, en la que se señala claramente la dirección de refinamiento asociada a cada elemento de proceso.

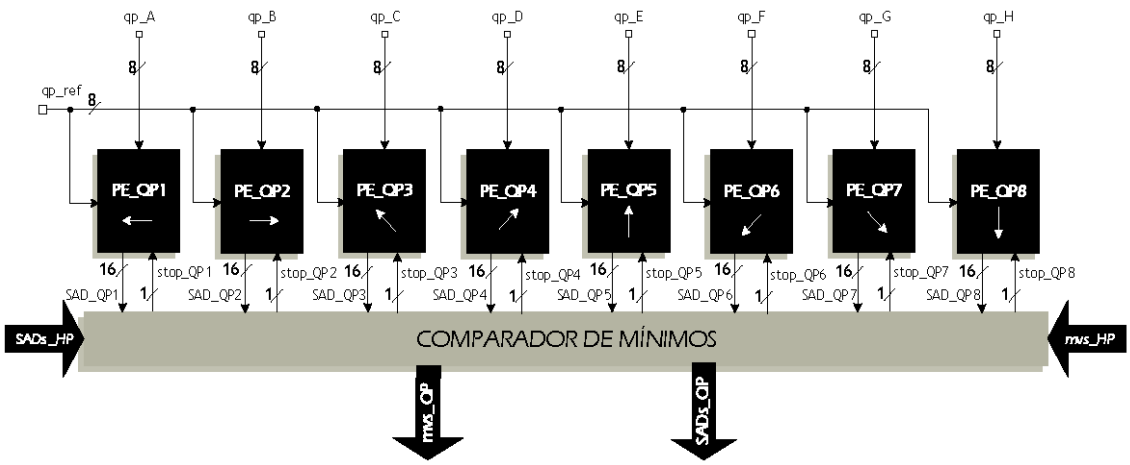


Figura 4.27: Arquitectura sistólica con $M=1$ y $N=8$ de refinamiento de vectores de movimiento a coordenadas de cuarto de píxel.

La arquitectura interna de cada uno de los elementos de proceso, representada en la Figura 4.28, es muy similar a la definida para el módulo de refinamiento de medio píxel, operando de la misma manera a la hora de obtener el SAD asociado a su posición de refinamiento para topologías superiores al bloque de 4×4 píxeles, así como en términos de eliminación temprana de candidatos.

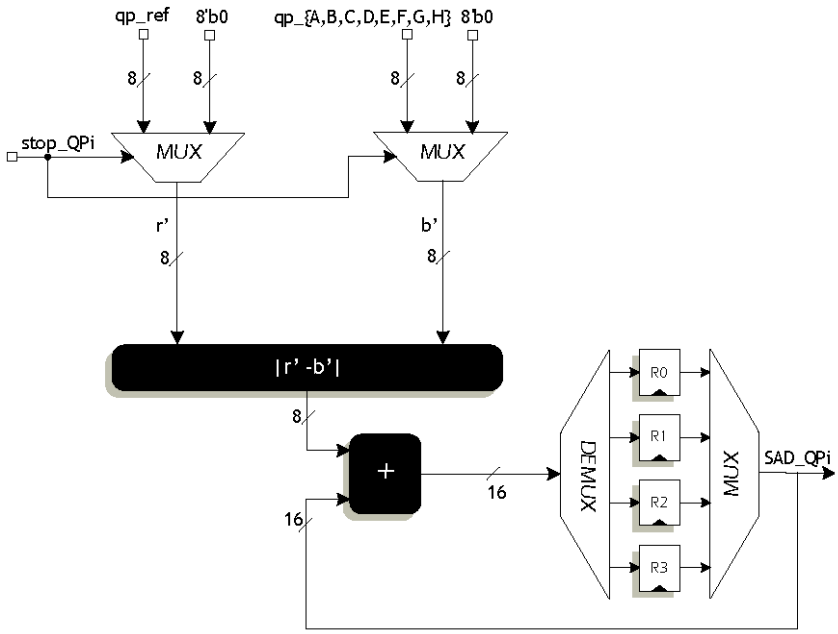


Figura 4.28: Elemento de proceso genérico de la arquitectura sistólica de refinamiento de cuarto de píxel.

La principal diferencia de la arquitectura de refinamiento de cuarto de píxel con respecto a la de medio píxel o a la de búsqueda con precisión entera, está motivada por la naturaleza del proceso de refinamiento de cuarto de píxel, en el cual cada una de las muestras interpoladas se utiliza exclusivamente en una sola de las posiciones de búsqueda. Este hecho determina que el grado de reutilización de las muestras de cuarto de píxel sea nulo y que por lo tanto, se elimine la línea de retardos que comunica a los elementos de proceso en las arquitecturas sistólicas anteriormente presentadas. De esta manera, los elementos de proceso operan en paralelo durante todos los ciclos de refinamiento de cuarto de píxel, según el flujo de datos definido en la Tabla 4.8.

T	PE_QP1	PE_QP2	PE_QP3	PE_QP4	PE_QP5	PE_QP6	PE_QP7	PE_QP8
0	|r0,0 – qp_A[0]|	|r0,0 – qp_B[0]|	|r0,0 – qp_C[0]|	|r0,0 – qp_D[0]|	|r0,0 – qp_E[0]|	|r0,0 – qp_F[0]|	|r0,0 – qp_G[0]|	|r0,0 – qp_H[0]|
1	|r0,1 – qp_A[1]|	|r0,1 – qp_B[1]|	|r0,1 – qp_C[1]|	|r0,1 – qp_D[1]|	|r0,1 – qp_E[1]|	|r0,1 – qp_F[1]|	|r0,1 – qp_G[1]|	|r0,1 – qp_H[1]|
2	|r0,2 – qp_A[2]|	|r0,2 – qp_B[2]|	|r0,2 – qp_C[2]|	|r0,2 – qp_D[2]|	|r0,2 – qp_E[2]|	|r0,2 – qp_F[2]|	|r0,2 – qp_G[2]|	|r0,2 – qp_H[2]|
3	|r0,3 – qp_A[3]|	|r0,3 – qp_B[3]|	|r0,3 – qp_C[3]|	|r0,3 – qp_D[3]|	|r0,3 – qp_E[3]|	|r0,3 – qp_F[3]|	|r0,3 – qp_G[3]|	|r0,3 – qp_H[3]|
4	|r1,0 – qp_A[4]|	|r1,0 – qp_B[4]|	|r1,0 – qp_C[4]|	|r1,0 – qp_D[4]|	|r1,0 – qp_E[4]|	|r1,0 – qp_F[4]|	|r1,0 – qp_G[4]|	|r1,0 – qp_H[4]|
5	|r1,1 – qp_A[5]|	|r1,1 – qp_B[5]|	|r1,1 – qp_C[5]|	|r1,1 – qp_D[5]|	|r1,1 – qp_E[5]|	|r1,1 – qp_F[5]|	|r1,1 – qp_G[5]|	|r1,1 – qp_H[5]|
6	|r1,2 – qp_A[6]|	|r1,2 – qp_B[6]|	|r1,2 – qp_C[6]|	|r1,2 – qp_D[6]|	|r1,2 – qp_E[6]|	|r1,2 – qp_F[6]|	|r1,2 – qp_G[6]|	|r1,2 – qp_H[6]|
7	|r1,3 – qp_A[7]|	|r1,3 – qp_B[7]|	|r1,3 – qp_C[7]|	|r1,3 – qp_D[7]|	|r1,3 – qp_E[7]|	|r1,3 – qp_F[7]|	|r1,3 – qp_G[7]|	|r1,3 – qp_H[7]|
8	|r2,0 – qp_A[8]|	|r2,0 – qp_B[8]|	|r2,0 – qp_C[8]|	|r2,0 – qp_D[8]|	|r2,0 – qp_E[8]|	|r2,0 – qp_F[8]|	|r2,0 – qp_G[8]|	|r2,0 – qp_H[8]|
9	|r2,1 – qp_A[9]|	|r2,1 – qp_B[9]|	|r2,1 – qp_C[9]|	|r2,1 – qp_D[9]|	|r2,1 – qp_E[9]|	|r2,1 – qp_F[9]|	|r2,1 – qp_G[9]|	|r2,1 – qp_H[9]|
10	|r2,2 – qp_A[10]|	|r2,2 – qp_B[10]|	|r2,2 – qp_C[10]|	|r2,2 – qp_D[10]|	|r2,2 – qp_E[10]|	|r2,2 – qp_F[10]|	|r2,2 – qp_G[10]|	|r2,2 – qp_H[10]|
11	|r2,3 – qp_A[11]|	|r2,3 – qp_B[11]|	|r2,3 – qp_C[11]|	|r2,3 – qp_D[11]|	|r2,3 – qp_E[11]|	|r2,3 – qp_F[11]|	|r2,3 – qp_G[11]|	|r2,3 – qp_H[11]|
12	|r3,0 – qp_A[12]|	|r3,0 – qp_B[12]|	|r3,0 – qp_C[12]|	|r3,0 – qp_D[12]|	|r3,0 – qp_E[12]|	|r3,0 – qp_F[12]|	|r3,0 – qp_G[12]|	|r3,0 – qp_H[12]|
13	|r3,1 – qp_A[13]|	|r3,1 – qp_B[13]|	|r3,1 – qp_C[13]|	|r3,1 – qp_D[13]|	|r3,1 – qp_E[13]|	|r3,1 – qp_F[13]|	|r3,1 – qp_G[13]|	|r3,1 – qp_H[13]|
14	|r3,2 – qp_A[14]|	|r3,2 – qp_B[14]|	|r3,2 – qp_C[14]|	|r3,2 – qp_D[14]|	|r3,2 – qp_E[14]|	|r3,2 – qp_F[14]|	|r3,2 – qp_G[14]|	|r3,2 – qp_H[14]|
15	|r3,3 – qp_A[15]|	|r3,3 – qp_B[15]|	|r3,3 – qp_C[15]|	|r3,3 – qp_D[15]|	|r3,3 – qp_E[15]|	|r3,3 – qp_F[15]|	|r3,3 – qp_G[15]|	|r3,3 – qp_H[15]|

Tabla 4.8: Diagrama de tiempos del sistólico de elementos de proceso de cuarto de píxel para un bloque genérico de 4×4 píxeles.

4.4.2.3 Procesamiento y control

Este bloque funcional se encarga de procesar la información enviada por el módulo de refinamiento de medio píxel (vectores de movimiento, píxeles y muestras de medio píxel) para, de acuerdo a esta información y al valor del registro de configuración, realizar de manera eficiente el control global del módulo de refinamiento de cuarto de píxel.

Las tareas de control a realizar por este bloque funcional están fuertemente determinadas por dos condicionantes: el funcionamiento segmentado del módulo de refinamiento de cuarto de píxel y el modo de estimación de movimiento bajo análisis. El primero de ellos se refiere al hecho de que los interpoladores de medio píxel operan con un retardo de 26 ciclos entre ellos, y que por lo tanto, la recepción de los píxeles y muestras de medio píxel necesarias para la interpolación de las muestras de cuarto de píxel correspondientes a cada uno de los bloques de 4×4 píxeles ha de estar acorde con las mencionadas características temporales. Sin embargo, a pesar de que esta recepción escalada de los datos es fija, el instante en el cual se empiezan a procesar dichos datos por parte de los interpoladores de cuarto de píxel dependerá del modo de estimación de movimiento. Así, sólo cuando se estén calculando los dieciséis vectores de movimiento correspondientes al modo 4×4 se obtendrá un funcionamiento de los interpoladores de cuarto de píxel totalmente segmentado, mientras

para el resto de modos o topologías de estimación de movimiento existirán, para uno o más de los interpoladores de cuarto de píxel, ciclos de espera. En particular, la Figura 4.29 muestra el comportamiento a nivel de bloques funcionales del módulo de refinamiento de cuarto de píxel para la obtención de vectores de movimiento correspondientes al modo 4×4.

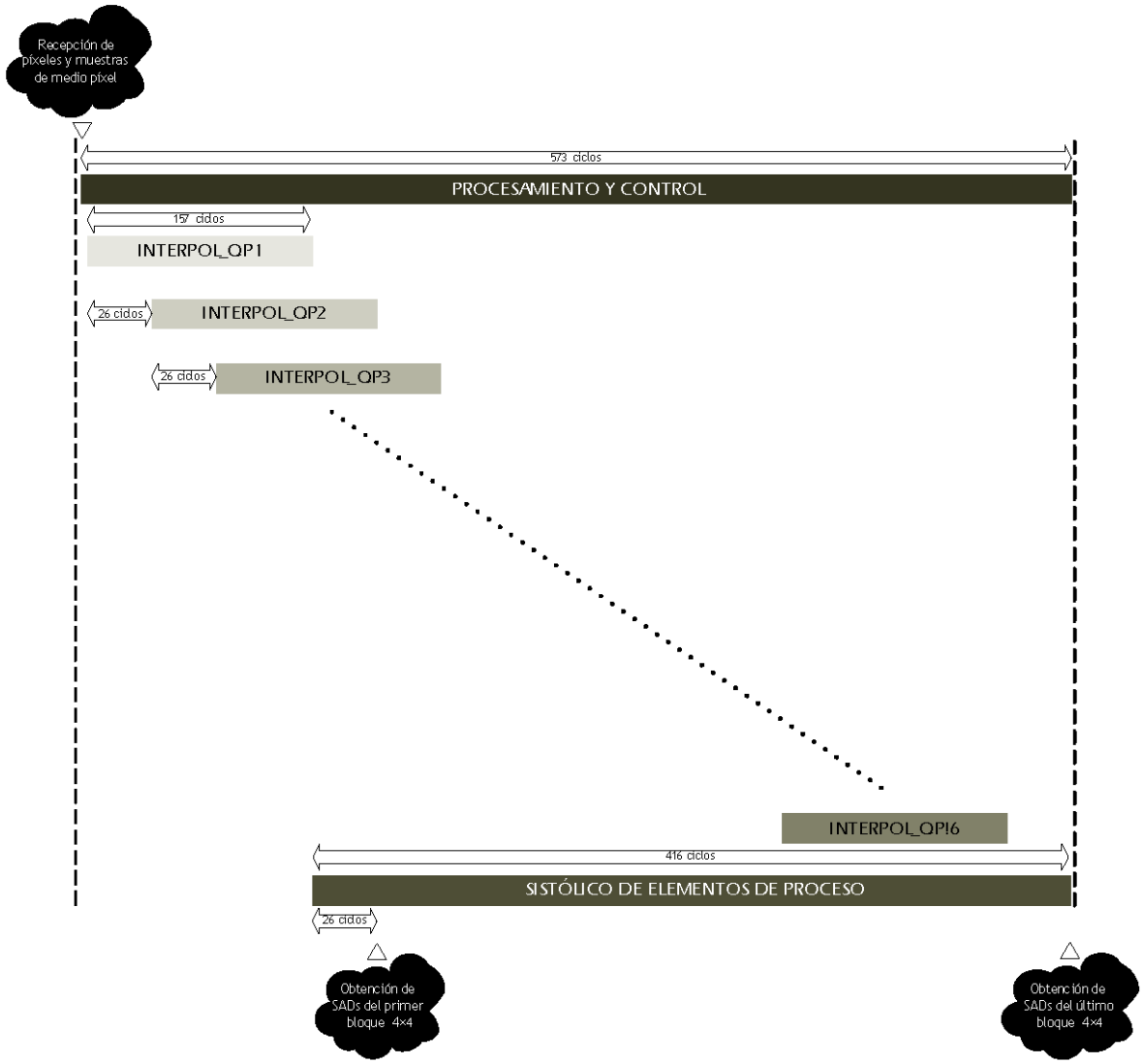


Figura 4.29: Diagrama de tiempos de refinamiento de vectores a coordenadas de cuarto de píxel para el modo de estimación de movimiento 4×4.

4.4.3 Resultados obtenidos con la etapa de refinamiento propuesta

Una vez analizada la arquitectura de la etapa de refinamiento sub-píxel propuesta en esta Tesis, en este apartado se presentan los resultados obtenidos en términos de latencia y coste hardware asociado.

4.4.3.1 Ciclos de trabajo de la arquitectura propuesta para diferentes casos de estudio

La introducción del registro de configuración propuesto en esta Tesis permite examinar y comparar las prestaciones de la etapa de refinamiento sub-píxel propuesta para diferentes casos de estudio. En este sentido, resulta de gran interés evaluar la latencia de la arquitectura de refinamiento propuesta para un conjunto de casos significativos del proceso de refinamiento. Así, se ha evaluado en primer lugar la latencia obtenida por macrobloque en el caso en el que sólo esté activo el refinamiento de medio píxel para diferentes combinaciones de modos de estimación de movimiento activos, y en segundo lugar, la latencia obtenida tras realizar el proceso de refinamiento de medio y cuarto de píxel para la misma combinación de modos. Los resultados obtenidos para cada caso se muestran en la Figura 4.30 (a) y (b), respectivamente.

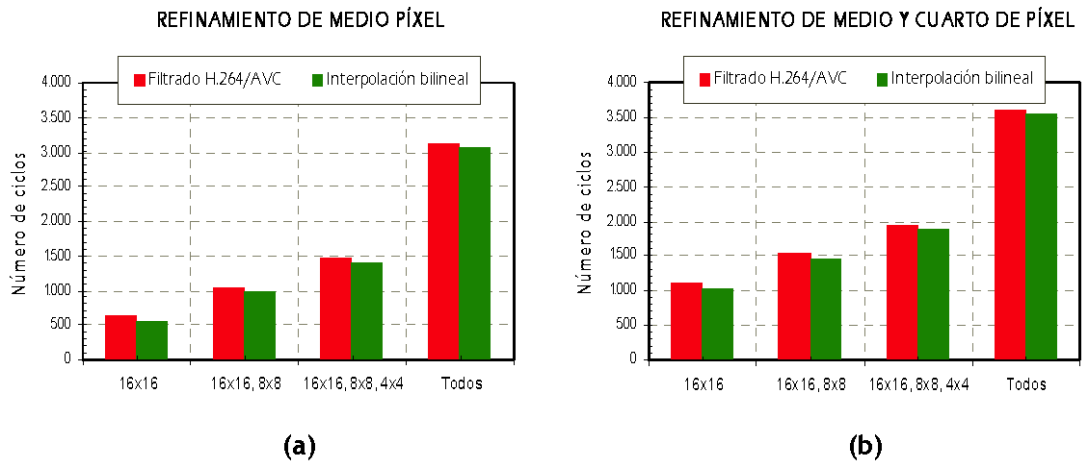


Figura 4.30: Ciclos de trabajo por macrobloque obtenidos para el refinamiento sub-píxel en diferentes casos de estudio.

A partir de estos resultados se extraen las siguientes conclusiones:

- El paralelismo introducido entre las etapas de refinamiento de medio píxel y refinamiento de cuarto de píxel es efectivo, minimizándose el número de ciclos de espera desde que se obtienen los vectores de medio píxel hasta que se éstos son refinados a coordenadas de cuarto de píxel para cualquier combinación de modos activos. En los casos mostrados en la Figura 4.30, al estar activo el modo 16×16, el sub-módulo de cuarto de píxel debe esperar, por las razones anteriormente comentadas, a que la etapa de refinamiento de medio píxel termine de procesar el último bloque de 4×4 píxeles para comenzar con el proceso de refinamiento a coordenadas de cuarto de píxel. Este hecho determina que la diferencia en ciclos para todos los casos de estudio entre el refinamiento de medio píxel y el refinamiento de cuarto de píxel sea constante e igual a 479 ciclos.

- El número de ciclos de refinamiento depende significativamente del número de modos activos, alcanzándose unos porcentajes de reducción en la latencia del proceso de refinamiento del 80% y 70% para las etapas de refinamiento de medio píxel y cuarto de píxel, respectivamente. De esta manera, se obtiene una arquitectura de refinamiento con una latencia acorde al número de modos activos, lo cual supone una considerable novedad con respecto a las arquitecturas de refinamiento previamente publicadas por otros autores, tanto para el estándar H.264/AVC como para estándares anteriores. Esta característica resulta de especial relevancia en los siguientes casos:

    - **Estimación de movimiento mediante el algoritmo VBS-ACBM propuesto en esta Tesis**. En este caso, el algoritmo predictivo PBM-HW sólo se ejecuta

para el modo 16×16, y por lo tanto, sólo se necesita refinar un vector de movimiento por macrobloque.

- **Estimación de movimiento multiestándar**. Dentro de los estándares H.263 y MPEG-4 sólo es necesario refinar a coordenadas de medio píxel los vectores de movimiento correspondientes a dos modos de estimación de movimiento (16×16 y 8×8), mientras que en el caso del estándar H.264/AVC es necesario, en principio, refinar los vectores correspondientes a los siete modos definidos por el estándar hasta coordenadas de cuarto de píxel. A partir de la información mostrada en al Figura 4.30 se comprueba que la arquitectura de refinamiento propuesta realiza un esfuerzo computacional adaptado al estándar en uso.
- **Estimación de movimiento simplificada dentro del estándar H.264/AVC.** Tal y como ya se ha comentado en esta Tesis, numerosos trabajos han demostrado el éxito de las estrategias basadas en calcular los vectores de movimiento para un número variable de modos por macrobloque. Mediante la manipulación del registro de configuración, estas estrategias son fácilmente acomodables dentro de la arquitectura propuesta.

- El uso de algoritmos de interpolación bilineal permite obtener una reducción en el número de ciclos inferior al 10% para todos los casos de estudio.

- El número de ciclos invertido para realizar el refinamiento de todos los modos según las directrices del estándar H.264/AVC es muy similar al necesario para realizar la estimación de movimiento con precisión entera. Este hecho es de especial relevancia en codificadores de vídeo basados en este estándar, en los cuales la estimación de movimiento con precisión entera y el refinamiento sub-píxel *ocupan* dos etapas del *pipeline* global del codificador [CCH+06b], [LW06].

Tomando como referencia el caso más desfavorable (refinamiento de todos los modos de estimación de movimiento según las exigencias del estándar H.264/AVC), la arquitectura propuesta es capaz de realizar en tiempo real el refinamiento de secuencias de vídeo CIF@30fps (352×288 píxeles a razón de 30 fotogramas por segundo) y 4CIF@15fps (704×576 píxeles a razón de 15 fotogramas por segundo) con una frecuencia de trabajo de 50MHz y 100 MHz respectivamente. En cualquier caso, la arquitectura propuesta es fácilmente escalable, pudiéndose aumentar su capacidad de procesamiento mediante la unión de sub-módulos de refinamiento que refinen en paralelo los vectores de movimiento correspondientes a diferentes modos de estimación de movimiento. De esta manera, considerando dos sub-módulos de refinamiento de medio píxel y otros dos de refinamiento de cuarto de píxel idénticos a los propuestos en esta Tesis, el refinamiento en tiempo real de secuencias de vídeo 4CIF@30fps es perfectamente abordable.

4.4.3.2 Coste hardware de la arquitectura propuesta

La Tabla 4.9 muestra los resultados obtenidos, en términos de miles de puertas NAND2 equivalentes, tras realizar la síntesis de la arquitectura de refinamiento sub-píxel con tecnología CMOS de 0,25 μm para una frecuencia objetivo de 100 MHz.

	Sub-módulo de refinamiento de medio píxel	Sub-módulo de refinamiento de cuarto de píxel
PE	0,71	0,63
Sistólico de PEs	11,71	11,04
Interpolador de medio píxel	4,83	0
Interpolador de cuarto de píxel	0	0,96
Arquitectura completa	54,22	33,49

Tabla 4.9: Número de puertas NAND2 equivalentes (miles) para cada bloque funcional de los sub-módulos de refinamiento de medio y cuarto de píxel.

Asimismo, se muestran en la Tabla 4.10 los requisitos de almacenamiento demandados por cada uno de elementos funcionales de los su-módulos de refinamiento de medio y cuarto de píxel.

	Requisitos de almacenamiento (bytes)
Interpolador de medio píxel	280
Interpolador de cuarto de píxel	121
Memoria A de medio píxel	48
Memoria B de medio píxel	68
Memoria C de medio píxel	24
Memoria D de medio píxel	60
Memoria A de cuarto píxel	64
Memoria B de cuarto píxel	64
Memoria C de cuarto píxel	64
Memoria D de cuarto píxel	64
Memoria E de cuarto píxel	64
Memoria F de cuarto píxel	64
Memoria G de cuarto píxel	64
Memoria H de cuarto píxel	64

Tabla 4.10: Requisitos de almacenamiento (en bytes) del módulo de refinamiento sub-píxel propuesto.

La validez de estos resultados se determinará en el apartado 4.6 del presente capítulo, en el que se compararán las prestaciones de la arquitectura de refinamiento propuesta con las ofrecidas por las arquitecturas aportadas por otros autores.

4.5 Metodología de verificación

La arquitectura de estimación de movimiento propuesta en esta Tesis ha sido descrita a nivel *RTL* mediante el uso del lenguaje de descripción hardware *Verilog*. Para verificar su funcionamiento, se ha empleado una metodología de verificación que consiste en comparar, macrobloque a macrobloque, los vectores de movimiento calculados por los estimadores de movimiento de los códigos *C* de los estándares H.263 [H263code] y H.264/AVC [H264code]

con los vectores calculados por la arquitectura propuesta. En esta Tesis, se han ampliado ambos códigos de referencia mediante la introducción del algoritmo VBS-ACBM con el objetivo de obtener los resultados mostrados en el capítulo 3 y realizar su posterior verificación arquitectural usando la metodología propuesta.

Esta metodología de verificación se resume en la Figura 4.31. Tal y como se muestra en dicha figura, mediante la variación de los parámetros de configuración que determinan el procesamiento a realizar por el codificador descrito en lenguaje *C*, se consigue hacer pasar al estimador de movimiento de dicho codificador por un conjunto de estados aleatorio. Para cada uno de estos estados, se extraen las entradas al estimador de movimiento y las salidas proporcionadas por éste para cada conjunto de entradas. El funcionamiento del estimador de movimiento descrito a nivel *RTL* se considera correcto si, para las mismas entradas, es capaz de obtener las mismas salidas *bit a bit* para todos los casos de test. En caso contrario, es necesario modificar el código *RTL* o los ficheros de test hasta obtener un grado de coincidencia total.

Obviamente, el éxito de esta estrategia de verificación depende en gran medida del número de estados por los que se consiga hacer pasar al codificador a través de la variación de los estímulos de configuración. Por esta razón, durante la verificación llevada a cabo se han utilizado todas las secuencias de vídeo definidas en el Anexo de esta Tesis, realizando una variación de los parámetros de entrada relacionados con el estimador de movimiento (modos activos, precisión del refinamiento, número de comparaciones y valor del escalón de cuantificación) para todos los casos. Asimismo, este proceso se ha repetido para las arquitecturas tras el proceso de síntesis, garantizándose su correcto funcionamiento.

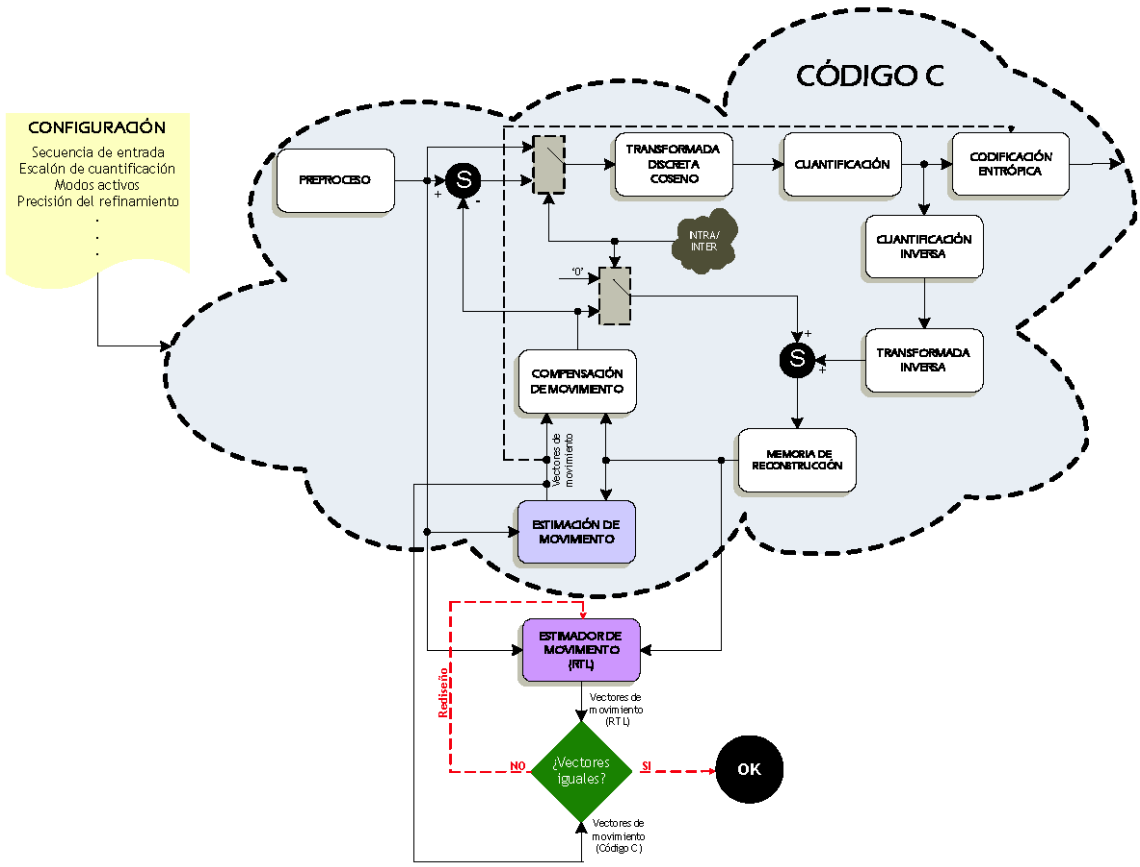


Figura 4.31: Estrategia de verificación empleada en esta Tesis.

En la Tabla 4.11 se resumen, en función de los valores del escalón de cuantificación y del registro de configuración modificado (ver Figura 4.17), los casos de test verificados para cada una de las secuencias de entrada. Mediante la asignación de valores al registro de configuración, se consigue realizar la verificación arquitectural para diferentes modos activos, número de comparaciones durante la eliminación temprana de candidatos y algoritmos de interpolación de las muestras de medio píxel, realizándose para todos los casos el refinamiento de medio y cuarto de píxel.

Nombre de la secuencia	Valores del escalón de cuantificación	Valores del registro de configuración
TABLE (300 fotogramas)	10,30,50	111011111111 111101111111 110011001001 110101001001
DEADLINE (1375 fotogramas)	10,30,50	111011001000 110101001000 111111001000 110011000000
FOREMAN (400 fotogramas)	10,30,50	110101000000 111111000000 110011111111 111101111111
MISS AMERICA (150 fotogramas)	10,30,50	110011001001 110101001001 111011001000 111101001000
PAMPHLET (300 fotogramas)	10,30,50	111111001000 111011000000 110101000000 110111000000
SUZIE (40 fotogramas)	10,30,50	111011111111 110101111111 111011001001 110101001001
COASTGUARD (100 fotogramas)	10,30,50	110011001000 111101001000 110111001000 111011000000

Tabla 4.11: Casos de test verificados en las arquitecturas propuestas.

4.6 Comparación con trabajos previos

Independientemente de las características en términos de adaptabilidad y niveles de compresión ofrecidas por el algoritmo de estimación movimiento VBS-ACBM propuesto en esta Tesis, resulta interesante analizar y comparar desde un punto de vista *puramente arquitectural* la arquitectura propuesta con respecto a los trabajos que definen el estado

actual del arte para los procesos de estimación de movimiento, tanto con precisión entera, como con precisión sub-píxel.

4.6.1 Comparación con arquitecturas previas de estimación de movimiento con precisión entera

Dentro de las arquitecturas de estimación de movimiento con precisión entera y capacidad para obtener vectores de movimiento para todos los modos definidos por el estándar H.264/AVC, existen propuestas tanto de arquitecturas unidimensionales como bidimensionales. A pesar de que las propuestas de esta Tesis se corresponden con arquitecturas unidimensionales, y es por lo tanto con este tipo de arquitecturas con el que se realizará una pormenorizada comparación, se presentan de igual manera las características generales de los trabajos más significativos dentro de las arquitecturas bidimensionales con el objetivo de realizar un análisis comparativo completo.

Dentro de las arquitecturas con topología bidimensional de elementos de proceso, destacan en número de propuestas las formadas por una matriz de 16×16 elementos de proceso y un árbol de sumas, que facilitan enormemente la composición de los vectores de los modos superiores a partir de la obtención en paralelo de los SADs de cada uno de los bloques de 4×4 píxeles presentes en un macrobloque [HWH+03], [DGH+05], [KHC05], [WG05a]. A partir de la estructura arquitectural *base* compuesta por 256 elementos de proceso, destacan dos variantes propuestas recientemente. La primera de ellas [ZG05a], considera una matriz de 16×32 elementos de proceso, de los cuales sólo 256 de ellos permanecen activos durante el proceso de estimación de movimiento, realizándose en los 256 elementos restantes una precarga del resto de píxeles que conforman el área de búsqueda. De esta manera se consigue, a partir de la recirculación de los datos desde la matriz *inactiva* hasta la matriz *activa*, reducir el número de accesos a memoria a costa de aumentar el área ocupada por la arquitectura. La

segunda de las variantes [CCH+06a], consiste en la unión de ocho módulos bidimensionales de estructura idéntica a la presentada en [HWH+03], constituyendo así una arquitectura *tridimensional* de 2048 elementos de proceso que, a su vez, ha sido incluida en el chip codificador H.264/AVC diseñado por el mismo grupo de investigación [CHC+05b], [HCT+05], [CLC06]. El acoplamiento de estructuras bidimensionales permite aumentar por un factor de ocho el número de posiciones evaluadas por ciclo de reloj, consiguiendo a la misma vez un altísimo grado de reutilización de los píxeles del área de búsqueda. Cabe destacar que los autores proponen como alternativa una solución de coste reducido en la cual se realiza un sub-muestreo previo al proceso de estimación de movimiento, resultando ocho matrices compuestas por 128 elementos de proceso, con la consiguiente pérdida en los niveles de compresión alcanzados [CCH+06b].

En cualquier caso, todas las propuestas mencionadas están basadas en una matriz *base* de 16×16 elementos de proceso *totalmente agrupada*, encontrándose las principales diferencias entre ellas en la manera en la que los píxeles circulan dentro de la mencionada matriz. Así, para completar el proceso de estimación de movimiento, en los trabajos [DGH+05], [ZG05a], [WG05a] y [CCH+06a] los píxeles de referencia permanecen almacenados en registros internos de los elementos de proceso, circulando los píxeles del área de búsqueda por la matriz de acuerdo a un orden lexicográfico de evaluación de candidatos [DGH+05], [ZG05a] o bien siguiendo un orden *serpenteado* [WG05a], [CCH+06a] que permite aumentar la reutilización de píxeles entre posiciones de búsqueda consecutivas. Sin embargo, en [KHC05] se proponen movimientos horizontales y verticales para los píxeles de búsqueda y de referencia, respectivamente, simplificando así el flujo de datos dentro de la matriz de proceso.

Con respecto a las topologías de proceso unidimensionales, la primera arquitectura capaz de realizar la estimación de movimiento mediante búsqueda exhaustiva para todos los modos de estimación de movimiento definidos por el estándar H.264/AVC fue propuesta por Yap y

McCanny en [YM03]. Se trata de una arquitectura *unidimensional no agrupada* compuesta por 16 elementos de proceso, cuyo flujo de datos se corresponde con el definido en la Tabla 4.1, y en la que los datos de síntesis aportados por los autores (108K puertas equivalentes con tecnología de 0,13 μm) indican un claro desaprovechamiento del área del circuito debido fundamentalmente a dos razones. La primera de ellas, tal y como se ha demostrado en esta Tesis, es una consecuencia directa del uso de elementos de proceso absolutamente *desagrupados*, lo que conduce inevitablemente al uso de un elemento de construcción de SADs (en la terminología de esta Tesis, SAD_COMPOSER) de vectores de tamaños de bloques superiores por cada elemento de proceso, en vez de utilizar uno solo de estos *constructores* por grupo de elementos de proceso. La segunda razón está directamente relacionada con el diseño del elemento de proceso en sí, quedando perfectamente definida y caracterizada en los trabajos dirigidos por el Profesor Liang-Gee Chen de la Universidad Nacional de Taiwan [CCH+06a]. En esta publicación, se realiza un estudio acerca de las implicaciones de soportar diferentes tamaños de bloque en arquitecturas de estimación de movimiento *clásicas* (con capacidad para calcular exclusivamente vectores de movimiento para topología de macrobloque), afirmándose que en arquitecturas sistólicas *no agrupadas* (en la terminología de la referida publicación, arquitecturas *inter-level*) el elemento de proceso tiene que modificarse irremediablemente según lo establecido en la Figura 4.32, donde n representa el número de bloques de menor tamaño presentes en un macrobloque completo (en el caso del estándar H.264/AVC, $n = 16$).

El trabajo de Yap y McCanny está basado en la arquitectura *unidimensional no agrupada* propuesta por Yang, Sun y Wu en [YSW89] para estimación de movimiento en bloques de 16×16 píxeles, introduciendo modificaciones en los elementos de proceso que conducen a una solución muy similar a la señalada en la Figura 4.32. En este sentido, en esta Tesis se ha aportado una nueva solución que permite, para cada uno de los grupos de elementos de proceso, reducir los multiplexores y demultiplexores de 16-a-1 a 4-a-1, con el consiguiente impacto en el área final del circuito. De esta manera, en esta Tesis se obtiene una solución

mejor en términos de área que la propuesta en [CCH+06a] para posibilitar la obtención de vectores de movimiento para varias topologías de macrobloques en arquitecturas *unidimensionales no agrupadas* con un esfuerzo de rediseño reducido.

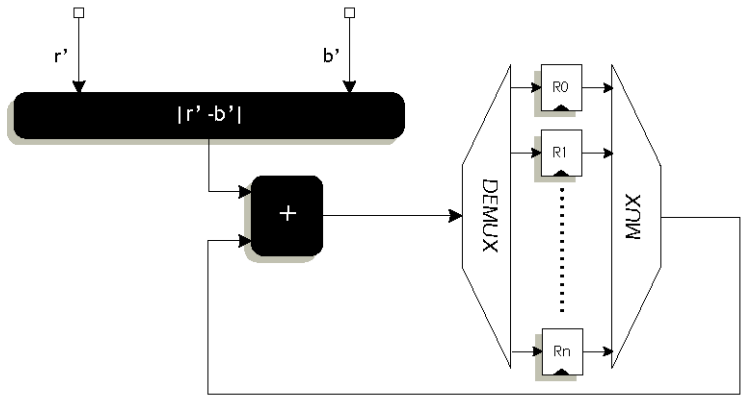


Figura 4.32: Modificación del elemento de proceso propuesta en [CCH+06a] con el objetivo de soportar diferentes modos de estimación de movimiento.

Con posterioridad a los trabajos de Yap y McCanny han aparecido dos nuevas publicaciones que, de manera indirecta, corroboran los resultados obtenidos en esta Tesis. Ou, Le y Hwang proponen en la primera de estas publicaciones una arquitectura *unidimensional agrupada* formada por 256 elementos de proceso agrupados de cuatro en cuatro [OLH05] de manera similar a la organización de elementos de proceso propuesta en esta Tesis. En particular, el mencionado trabajo propone un segundo nivel de agrupación, de manera que cuatro grupos de cuatro elementos de proceso cada uno constituyen, en la terminología propuesta por los autores, un *SAD module*. Cada uno de los dieciséis *SAD modules* de la arquitectura se encarga de obtener el SAD de un bloque de 4×4 píxeles para todas las posiciones del área de búsqueda, existiendo por lo tanto un único SAD_COMPOSER que, a partir de los resultados de cada *SAD module*, obtiene los vectores para todos los modos de estimación de movimiento definidos por el estándar H.264/AVC. Los autores no aportan ningún dato acerca del área ocupada por cada *SAD module*, proporcionado únicamente el área total de la arquitectura que resulta ser de 597K puertas equivalentes con tecnología UMC de 0,18 μm.

La segunda de las mencionadas arquitecturas unidimensionales ha sido propuesta recientemente por Song, Liu, Goto e Ikenaga en [SLG+06]. Se trata de una arquitectura *unidimensional agrupada* compuesta por ocho grupos de dieciséis elementos de proceso cada uno. En este sentido, cada uno de los grupos definidos es prácticamente idéntico al correspondiente a la arquitectura estudiada en el apartado 4.2.2.3, existiendo igualmente un SAD_COMPOSER por grupo. Mediante la introducción de ocho de estos grupos se consigue aumentar de manera sencilla el número de posiciones de búsqueda por segundo, al operar cada uno de ellos sobre una porción del área de búsqueda de manera absolutamente independiente. Los autores presentan en la citada publicación el área ocupada, no sólo por la arquitectura completa, sino también por cada uno de los grupos considerados, resultando ser esta última de 20,94K puertas equivalentes incluyendo el SAD_COMPOSER y el comparador de mínimos con tecnología de 0,18 μm. Con el objetivo de comparar estos resultados con la arquitectura propuesta en esta Tesis con $M = 16$ y $N = 1$, se ha realizado una síntesis de esta última, resultando un total de 21,3K puertas equivalentes con tecnología de 0,25 μm, quedando de manifiesto la similitud de ambas propuestas. En cualquier caso, ninguna de las propuestas publicadas en [OLH05] y [SLG+06] contempla el uso de algoritmos adaptativos (ambas emplean exclusivamente el algoritmo de búsqueda exhaustiva) ni la incorporación de mecanismos de eliminación temprana, como es el caso de la arquitectura de precisión entera aportada en esta Tesis.

Los dos trabajos mencionados acerca de arquitecturas unidimensionales demuestran el alto grado de escalabilidad que poseen este tipo de arquitecturas, pudiéndose acoplar con gran facilidad un número determinado de grupos de proceso unidimensionales con el objetivo de aumentar la capacidad de procesamiento de la arquitectura de manera sencilla. En este sentido, y puesto que las arquitecturas bidimensionales con menos de 256 elementos de proceso resultan ser ineficientes con respecto a las unidimensionales (en [WYG+04] y [WG05b] se presentan sendas arquitecturas bidimensionales con 16 y 64 elementos de proceso, respectivamente, con peores prestaciones que cualquier arquitectura unidimensional

con el mismo número de elementos de proceso), en esta Tesis se ha realizado un estudio acerca de la idoneidad de la unión de grupos de cuatro elementos de proceso para realizar la estimación de movimiento con diferentes formatos de vídeo. En particular, la Figura 4.33 muestra las puertas lógicas necesarias para realizar la estimación de movimiento en secuencias con formato QCIF@30fps (176×144 píxeles por fotograma, a razón de 30 fotogramas por segundo), CIF@30fps (352×288 píxeles por fotograma, a razón de 30 fotogramas por segundo) y 4CIF@30fps (704×576 píxeles por fotograma, a razón de 30 fotogramas por segundo), siendo necesario uno, dos y ocho grupos de cuatro elementos con la estructura propuesta en esta Tesis, respectivamente, operando a una frecuencia de trabajo de 50 MHz para el primer caso y de 100 MHz para los dos restantes. Asimismo, en la Figura 4.33 se muestra el número de puertas lógicas equivalentes correspondiente a cada una de las arquitecturas bidimensionales basadas en una matriz de 256 elementos de proceso anteriormente señaladas, así como los resultados obtenidos par las propuestas realizadas en esta Tesis, con y sin posibilidad de eliminación temprana de candidatos (g4×4_e y g4×4 respectivamente).

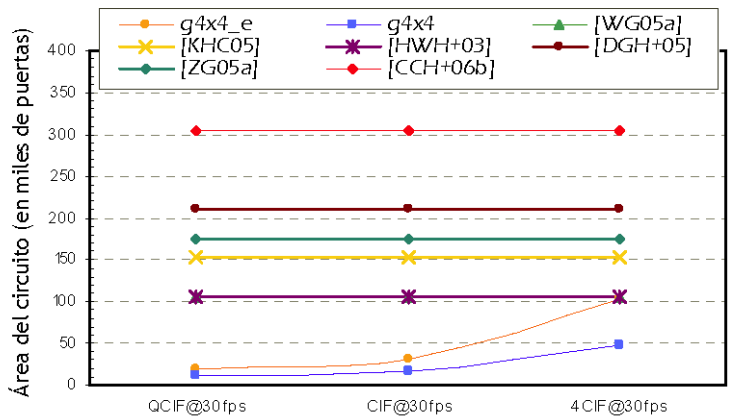


Figura 4.33: Número de puertas NAND2 equivalentes para diferentes formatos de vídeo y arquitecturas.

Tal y como se muestra en la Figura 4.33, para los tres formatos de vídeo considerados resulta más conveniente, en términos de área ocupada por el circuito, utilizar una arquitectura

basada en los grupos de elemento de proceso propuestos en esta Tesis, resultando las arquitecturas bidimensionales una mejor opción para formatos de vídeo de alta definición. Asimismo, se aprecia que el elemento de proceso en las arquitecturas bidimensionales presenta un coste hardware, por término medio, inferior al de las arquitecturas unidimensionales presentadas en esta Tesis. Este hecho se debe a que, al estar formadas las arquitecturas bidimensionales por una matriz de 16×16 elementos de proceso, sólo es necesario almacenar en registros los SADs de los bloques de 4×4 píxeles, calculándose el resto de los SADs sin necesidad de almacenamiento intermedio.

Por último, a modo de resumen, se muestran en la Tabla 4.12 las características más significativas de las arquitecturas de estimación de movimiento con precisión entera, incluyéndose las propuestas arquitecturales propuestas en esta Tesis compuestas por grupos de cuatro elementos de proceso, con y sin capacidad de eliminación temprana de candidatos (g4×4_e y g4×4, respectivamente).

4.6.2 Comparación con arquitecturas previas de estimación de movimiento con precisión sub-píxel

Resulta difícil comparar las prestaciones de las arquitecturas de refinamiento de medio y cuarto de píxel desarrolladas en esta Tesis teniendo en cuenta la escasez de trabajos publicados en esta área, como ya se indicó en el capítulo 2 del presente trabajo.

En este sentido, por su similitud con las arquitecturas de refinamiento a coordenadas de medio y cuarto de píxel propuestas en esta Tesis, resulta de utilidad comparar la arquitectura de refinamiento propuesta con la desarrollada por Chen, Huang y Chen en [CHC04b].

	Tipo de arquitectura	Número de PEs	Tecnología (µm)	Número de NAND2 equivalentes	Capacidad de procesamiento
[HWH+03]	Bidimensional agrupada	256	0,35	105,58K	720×480@30fps +SA: 63×47 ++f: 66,67 MHz
[YM03]	Unidimensional no agrupada	16	0,13	108K	704×576@13fps SA: 31×31 f: 100 MHz
[DGH+05]	Bidimensional agrupada	256	0,18	210K	720×576@30fps SA: 81×81 f: 260 MHz
[KHC05]	Bidimensional agrupada	256	0,18	154K	720×576@15fps SA: 79×79 f: 100 MHz
[OLH05]	Unidimensional agrupada	256	0,18	597K	1920×1080@60fps SA: 31×31 f: 123,49 MHz
[WG05a]	Bidimensional agrupada	256	0,25	105K	720×576@30fps SA: 31×31 f: 52 MHz
[ZG05a]	Bidimensional agrupada	512	0,18	174K	+++N/A
[CCH+06b]	Bidimensional agrupada	1024	0,18	305,21K	1280×720@30fps SA: 143×79 f: 108 MHz
[SLG+06]	Unidimensional agrupada	16/128	0,18	20,94K/180,57K	352×288@30fps SA: 47×47 f: 228 MHz/ 720×480@30fps SA: 71×63 f: 228 MHz
g4×4	Unidimensional agrupada	4/8/32	0,25	12,86K/17,77K/47,23K	176×144@30fps SA: 31×31 f: 50 MHz/ 352×288@30fps SA: 31×31 f: 100 MHz/ 704×576@30fps SA: 31×31 f: 100 MHz/
g4×4_e	Unidimensional agrupada	4/8/32	0,25	19,96K/31,97K/104,03K	176×144@30fps SA: 31×31 f: 50 MHz/ 352×288@30fps SA: 31×31 f: 100 MHz/ 704×576@30fps SA: 31×31 f: 100 MHz/

+SA: Área de búsqueda ++f: frecuencia +++N/A: No facilitado por los autores

Tabla 4.12: Resumen de las características de las arquitecturas de estimación de movimiento con precisión entera analizadas.

.A diferencia de la arquitectura de refinamiento propuesta en esta Tesis, en la cual se opta por una estrategia de interpolación distribuida, el trabajo presentado por Chen *et al.* propone el uso de un solo bloque de interpolación compartido por los 36 elementos de proceso de los que consta la arquitectura. Una vez interpolados todos los datos correspondientes a un modo de estimación de movimiento cualquiera, se procede al refinamiento de vectores de movimiento. Esta estrategia resulta ser altamente beneficiosa para los modos de estimación de movimiento asociados a topologías de bloque de tamaño elevado, pues se consigue un alto porcentaje de reutilización de los datos interpolados. Sin embargo, esta estrategia pierde eficacia para bloques de píxeles de tamaños pequeños. De hecho, según los datos aportados por los autores, el porcentaje de utilización de los elementos de proceso de la arquitectura varía entre el 73% y el 40% dependiendo de si el refinamiento se realiza para el modo 16×16 o 4×4, respectivamente. Es importante recordar que dicha arquitectura funciona bajo las directrices del algoritmo AMPD (*Advanced Mode Pre-Decision*) presentado por los autores, y que por lo tanto, sólo realiza el refinamiento de tres de los siete posibles modos de estimación de movimiento establecidos por el estándar H.264/AVC. Dicho algoritmo penaliza claramente el uso del modo 4×4, obteniéndose una pérdida de calidad de hasta 0,2 dB en la calidad de la imagen para las dos secuencias de test consideradas por los autores. Sin embargo, las dos secuencias de vídeo consideradas por los autores presentan idénticas características temporales, lo cual hace imposible determinar la pérdida de calidad obtenida mediante el uso del algoritmo AMPD para secuencias de características diferentes a las consideradas. Bajo estas condiciones, esta arquitectura es capaz de procesar 49K macrobloques por segundo a una frecuencia de trabajo de 100 MHz utilizando para ello 79,4K puertas NAND2 equivalentes, sin especificar los autores la cantidad de memoria necesaria para ello.

A modo de comparación, y según los datos expuestos en la Figura 4.30 acerca de los ciclos necesarios para refinar los vectores de movimiento correspondientes a tres modos de estimación de movimiento, la arquitectura de refinamiento propuesta en esta Tesis es capaz

de procesar, a la misma frecuencia de trabajo y para el mismo número de modos activos que en el caso de la propuesta [CHC04b], 53K macrobloques por segundo utilizando 8,31K puertas lógicas de más, con un porcentaje de utilización de los elementos de proceso cercano al 100%. Además, mediante el uso la arquitectura propuesta en esta Tesis no existen, a diferencia de la propuesta realizada en [CHC04b], pérdidas de calidad en la imagen reconstruida para cualquier secuencia de vídeo, independientemente de sus características espaciales y temporales.

Por último, es de destacar que la arquitectura descrita en [CHC04b] representa la única aportación publicada hasta la fecha para el refinamiento de vectores a coordenadas sub-píxel según las directrices establecidas por el estándar H.264/AVC. Puesto que en dicha arquitectura sólo se realiza el refinamiento de tres modos de estimación de movimiento, podemos afirmar que la arquitectura de refinamiento sub-píxel propuesta en esta Tesis representa una novedosa aportación en este campo al posibilitar, si el codificador de vídeo utilizado así lo requiere, el refinamiento de todos los modos de estimación de movimiento definidos por el estándar H.264/AVC para formatos de vídeo 4CIF@15fps o inferiores, con un coste hardware reducido. De igual manera, es de destacar que el algoritmo AMPD o cualquier otro que permita reducir el número de modos de refinamiento activos, es fácilmente acomodable dentro de la arquitectura de refinamiento propuesta. Asimismo, a diferencia de la arquitectura publicada en [CHC04b], la arquitectura aportada incorpora mecanismos de selección de modos activos, cuyos beneficios, en términos de latencia, han quedado demostrados en esta Tesis.

4.7 Conclusiones

En este capítulo se ha propuesto una arquitectura de estimación de movimiento para sistemas de codificación de vídeo H.263 y H.264/AVC que permite la obtención de vectores de movimiento según las directrices del algoritmo VBS-ACBM, propuesto en el capítulo 3 de esta Tesis. En dicha arquitectura, el proceso de estimación de movimiento se ha dividido en dos etapas claramente diferenciadas: estimación de movimiento con precisión entera y refinamiento sub-píxel. Para ambas etapas se han presentado sendas propuestas arquitecturales en las cuales, independientemente de la adaptabilidad aportada de forma inherente por el algoritmo VBS-ACBM, se ha incluido un amplio conjunto de aportaciones con el objetivo de obtener unas prestaciones superiores a las ofrecidas por los trabajos previos recogidos en la bibliografía reciente. Entre estas novedades destacan la posibilidad de realizar el proceso de búsqueda sólo para aquellos modos de estimación que se deseen (el contexto de esta Tesis, *modos activos*), y la incorporación de mecanismos de eliminación temprana de candidatos, lo cual representa una novedad con respecto a arquitecturas de estimación de movimiento multimodo previamente publicadas. En ambos casos, se obtienen unas características de latencia y frecuencia de funcionamiento que permiten procesar holgadamente secuencias de video CIF@30fps (352×288 píxeles muestreados a razón de 30 fotogramas por segundo) con un coste hardware reducido.

La arquitectura de estimación de movimiento con precisión entera aportada en esta Tesis es capaz de realizar dicho proceso de búsqueda haciendo uso del algoritmo de búsqueda exhaustiva FSBM, o bien mediante el algoritmo predictivo PBM-HW propuesto en esta Tesis. En este capítulo se ha planteado, en primer lugar, la implementación del algoritmo de búsqueda exhaustiva sobre una arquitectura unidimensional *base* fácilmente escalable compuesta por 16 elementos de proceso. Para ello se ha realizado un novedoso estudio acerca de las diferentes posibilidades de agrupación de estos elementos de proceso, concluyéndose que una arquitectura agrupada formada por cuatro grupos de cuatro elementos de proceso

constituye la mejor opción en términos de latencia, coste hardware y eficiencia de eliminación de candidatos. Este análisis constituye una de las aportaciones de esta Tesis Doctoral, al no existir en las publicaciones previas ningún criterio específico de agrupación de elementos de proceso. Asimismo, en este estudio se ha puesto de relevancia que, mediante la incorporación de la posibilidad de efectuar la estimación de movimiento propuesta en esta Tesis, se puede incrementar el número de ciclos inactivos en los elementos de proceso de la arquitectura. Una vez evaluados los diferentes casos de estudio, se han descrito los cambios necesarios a realizar en la arquitectura propuesta para realizar el proceso de estimación de movimiento según el algoritmo PBM-HW, valorándose sus implicaciones arquitecturales en cuanto al área ocupada por el circuito y al incremento de los porcentajes de eliminación.

La etapa de refinamiento sub-píxel propuesta en esta Tesis se lleva a cabo mediante dos arquitecturas que permiten realizar el refinamiento de medio píxel y de cuarto de píxel con un alto grado de paralelismo y de utilización de los recursos hardware. En ambas arquitecturas, las propuestas realizadas de utilizar una estrategia de interpolación distribuida y una nueva arquitectura para cada elemento de proceso, ambas optimizadas para el procesamiento de bloques de 4×4 píxeles, permiten refinar con eficacia los vectores de movimiento con precisión entera para cualquiera de los modos establecidos por los estándares H.263 y H.264/AVC. Además, se ha demostrado que la posibilidad introducida en esta Tesis de controlar el número de modos activos y la precisión de los vectores de movimiento posee un impacto crucial sobre la latencia del circuito de refinamiento sin coste hardware adicional. De esta manera, y a diferencia de los trabajos previos presentados por otros autores, se realiza un esfuerzo computacional acorde al estándar en uso.

Por último, con el objetivo de contrastar la validez de las propuestas arquitecturales realizadas en este capítulo, se han comparado las prestaciones de la arquitectura propuesta con los más recientes y significativos trabajos publicados, tanto para la estimación con

precisión entera, como para el refinamiento sub-píxel. Dentro de la estimación de movimiento con precisión entera, y en lo que respecta a la comparación con otras arquitecturas unidimensionales, se ha demostrado que las propuestas realizadas de agrupación de elementos de proceso y de reducción de la lógica de multiplexado de cada elemento de proceso, conllevan una considerable reducción del área final del circuito. Asimismo, se ha puesto de manifiesto que ésta representa, para una misma frecuencia de funcionamiento, una opción mejor que cualquiera de las arquitecturas bidimensionales compatibles con el estándar H.264/AVC publicadas hasta la fecha para formatos de vídeo 4CIF@30fps (704×576 píxeles muestreados a razón de 30 fotogramas por segundo) o inferiores. Con respecto a la arquitectura de refinamiento sub-píxel compatible con los estándares H.263 y H.264/AVC, y debido a la escasez de aportaciones en este campo, sólo se ha podido comparar con un trabajo previamente publicado. En este caso, se ha demostrado que, en igualdad de condiciones, la arquitectura propuesta presenta unas prestaciones análogas a las publicadas por los autores de este trabajo pudiendo realizar, si se requiere el refinamiento de vectores de movimiento para cualquiera de los modos definidos por el estándar H.264/AVC. Además, a diferencia de dicho trabajo, la arquitectura propuesta realiza el refinamiento de vectores de movimiento en un número de ciclos acorde con las características del estándar en uso y/o las necesidades de compresión en cuanto al número de modos activos y tipo de interpolación utilizada, sin que para ello se sacrifique, en modo alguno, la calidad de la secuencia de vídeo reconstruida en sistemas de codificación híbrida de vídeo basados en los estándares H.263 y H.264/AVC.

Las conclusiones extraídas de esta comparativa, junto con las novedosas características de eliminación temprana multimodo y número variable de modos de estimación de movimiento que presenta la arquitectura de estimación de movimiento propuesta, hacen que ésta represente una importante aportación dentro del área de arquitecturas de estimación de movimiento.

CAPÍTULO 5

Conclusiones y líneas futuras

En este capítulo se presentan las conclusiones extraídas en el presente trabajo, así como las líneas futuras de investigación que se pretenden continuar.

5.1 Conclusiones

Los codificadores de vídeo basados en estándares se fundamentan en la utilización de estrategias de compresión híbrida, explotando las redundancias espaciales y temporales presentes en una señal de vídeo con el objetivo de llevar a cabo su compresión. Para realizar este proceso de manera eficiente, el diseño de la etapa de estimación de movimiento representa, por su impacto en las prestaciones de compresión así como por su elevado coste computacional, un factor crítico. De hecho, gran parte de los esfuerzos realizados por los investigadores dentro del campo de la compresión de vídeo se han centrado en la optimización algorítmica y arquitectural de dicha etapa, tal y como demuestra el elevado número de publicaciones al respecto. Sin embargo, tal y como ha quedado reflejado en el capítulo 2 de esta Tesis, ni los algoritmos, ni aún menos las arquitecturas disponibles, han alcanzado un grado de madurez que haga pensar que este esfuerzo investigador no continuará durante los próximos años.

En esta Tesis se han estudiado los trabajos previos más significativos dentro de esta doble vertiente algorítmica y arquitectural, haciendo hincapié en aquellas soluciones de bajo coste computacional que permiten, mediante adaptación a las características de la secuencia de vídeo a comprimir, obtener niveles de compresión razonables. Además, se ha examinado la validez de dichas propuestas para los estándares H.263 y H.264/AVC, en cuanto a su capacidad de calcular vectores de movimiento con precisión sub-píxel para diferentes topologías de bloque, así como las implicaciones arquitecturales de éstas. Este estudio ha puesto de manifiesto que, si bien la estimación de movimiento adaptativa representa una excelente opción para el cálculo eficiente de vectores de movimiento, todavía existen muchos aspectos sobre los que es necesario continuar investigando. En particular, este análisis de los trabajos previos a esta Tesis Doctoral ha permitido establecer la siguiente conclusión: no existe ningún estimador de movimiento adaptativo capaz de obtener vectores de movimiento que, cumpliendo con los requisitos de precisión (hasta cuarto de píxel) y tamaños de bloque

(desde 4×4 hasta 16×16 píxeles) establecidos por el estándar H.264/AVC, garanticen altos niveles de compresión para secuencias de vídeo de cualquier naturaleza espacio-temporal con un coste computacional reducido.

Con el objetivo de solventar esta carencia se ha introducido, en primer lugar, un entorno de simulación que permite analizar el proceso de estimación de movimiento desde el punto de vista de la función de coste de Lagrange para la estimación de movimiento. Este entorno de análisis ha permitido la identificación de *macrobloques críticos* en los que, para obtener una tasa de compresión elevada, es imprescindible obtener un vector de movimiento que minimice el residuo de movimiento compensado a costa de evaluar un número elevado de posiciones. Por otro lado, el análisis realizado ha posibilitado la detección de macrobloques en los que no es necesario evaluar un gran número de posiciones para calcular su vector de movimiento, siempre y cuando se garantice que el vector resultante es coherente con los vectores de los macrobloques situados en su vecindad espacial.

A partir de estas observaciones, se ha aportado un nuevo algoritmo de estimación de movimiento denominado ACBM (*Adaptive Cost Block Matching*) basado en la combinación del algoritmo de búsqueda exhaustiva FSBM (*Full Search Block Matching*) y el algoritmo de búsqueda predictiva PBM (*Predictive Block Matching*). Este algoritmo es capaz de proporcionar, como mínimo, las mismas prestaciones de compresión que el algoritmo de búsqueda exhaustiva con un coste computacional significativamente menor. Para ello, el algoritmo ACBM se basa en una novedosa estrategia de adaptación a las necesidades de compresión y a las características de la secuencia de vídeo a comprimir. Para comprobar la eficacia del algoritmo ACBM propuesto, se ha incluido dicho algoritmo en un codificador H.263 genérico y se han realizado diversas simulaciones para secuencias de entrada de muy diferentes características espaciales (diferentes texturas) y temporales (diferentes tasas de muestreo temporal que varían desde 5 hasta 30 fotogramas por segundo), calculándose un

solo vector de movimiento con precisión de medio píxel por macrobloque. Los resultados obtenidos demuestran que el algoritmo ACBM es capaz de obtener unas prestaciones de compresión óptimas, con una reducción en el coste computacional de entre el 30% y el 95% comparado con la solución de búsqueda exhaustiva para un área de búsqueda de 46×46 píxeles, dependiendo dicho esfuerzo computacional de las características de la secuencia de vídeo y del valor del escalón de cuantificación Q_p.

Con el objetivo de garantizar la validez del algoritmo propuesto para sistemas de codificación de vídeo en tiempo real basados en el estándar H.264/AVC, o en cualquiera de sus estándares predecesores, ha sido necesario realizar un conjunto de cambios encaminados a facilitar la implementación hardware del algoritmo ACBM propuesto y que a la vez, permitieran obtener vectores de movimiento para todos los modos de estimación de movimiento definidos por el estándar H.264/AVC. En este sentido, se ha propuesto en esta Tesis un nuevo algoritmo, denominado PBM-HW (*Predictive Block Matching – HardWare*) que, conservando la naturaleza predictiva del algoritmo PBM original, presenta unas características que facilitan su implementación en una arquitectura sistólica de estimación de movimiento típica.

Asimismo, y tras realizar un estudio del uso de los diferentes modos de estimación de movimiento por parte de un codificador H.264/AVC genérico, se ha propuesto un nuevo algoritmo de estimación de movimiento denominado VBS-ACBM (*Variable Block Size – Adaptive Cost Block Matching*) basado en la combinación de los algoritmos PBM-HW y FSBM. Gracias a la estrategia de adaptación propuesta, el algoritmo VBS-ACBM obtiene los vectores de movimientos demandados por el estándar H.264/AVC evaluando entre un 20% y un 80% del número total de posiciones inspeccionadas por el algoritmo FSBM para un área de búsqueda de 31×31 píxeles. Los resultados de simulación obtenidos utilizando el código *C* del estándar H.264/AVC, indican que esta reducción en el coste computacional se obtiene manteniendo la tasa de compresión resultante de aplicar el algoritmo de búsqueda exhaustiva

para todos los macrobloques de una secuencia, independientemente de las características de ésta.

Una vez contrastadas las prestaciones del algoritmo VBS-ACBM, se han propuesto en esta Tesis Doctoral una doble arquitectura – arquitectura de estimación de movimiento con precisión entera y arquitectura de refinamiento sub-píxel - para la obtención de vectores de movimiento en tiempo real según las directrices del algoritmo VBS-ACBM. Esta novedosa arquitectura es capaz de procesar secuencias CIF@30fps (352×288 píxeles muestreados a razón de 30 fotogramas por segundo) para una frecuencia de trabajo de 55 MHz, utilizando para ello 158,64K puertas NAND2 equivalentes en tecnología CMOS de 0,25 μm. Independientemente de las características favorables que otorga el algoritmo VBS-ACBM a la arquitectura en términos de adaptabilidad, latencia y procesamiento multiestándar, ésta se beneficia de un conjunto de aportaciones arquitecturales realizadas en esta Tesis que son absolutamente extrapolables a otras arquitecturas de estimación de movimiento con precisión entera o sub-píxel, indistintamente del algoritmo de ajuste de bloques que utilicen.

La arquitectura unidimensional de estimación de movimiento con precisión entera aportada es capaz de obtener vectores de movimiento para tamaño de bloque de 16×16 píxeles mediante el algoritmo PBM-HW y/o para todos los tamaños de bloque definidos por el estándar H.264/AVC mediante búsqueda exhaustiva. La arquitectura propuesta incorpora, a diferencia del resto de arquitecturas de estimación de movimiento con precisión entera publicadas hasta la fecha para codificadores H.264/AVC, mecanismos de eliminación temprana de candidatos y de selección de modos de estimación de movimiento activos. Para aprovechar al máximo estas características, se han analizado en esta Tesis, de manera novedosa, un conjunto de posibles configuraciones para los elementos de proceso de la arquitectura, demostrándose que la agrupación de cuatro elementos de proceso representa la mejor opción en arquitecturas unidimensionales en términos de área ocupada y porcentajes

de eliminación de candidatos. Asimismo se ha demostrado, mediante la realización de diferentes simulaciones, que la reducción del número de modos activos y la evaluación previa de las posiciones determinadas por el algoritmo PBM-HW proporcionan un considerable incremento del número de ciclos inactivos de los elementos de proceso de la arquitectura.

La arquitectura de refinamiento sub-píxel propuesta en esta Tesis permite refinar los vectores calculados por cualquier arquitectura de precisión entera a coordenadas de medio y cuarto de píxel, para cualquiera de las topologías de bloque y métodos de interpolación contemplados por los estándares H.263 y H.264/AVC, lo cual la convierte en una contribución relevante, mas aún si se tiene en cuenta la escasez de propuestas arquitecturales al respecto. La arquitectura propuesta consta de una etapa de refinamiento de medio píxel y otra de refinamiento de cuarto de píxel que operan con un alto grado de paralelismo gracias a la estrategia diseñada para el procesamiento independiente de bloques de 4×4 píxeles. Los resultados aportados en esta Tesis demuestran que, la posibilidad introducida en esta arquitectura de realizar el refinamiento según las necesidades del estándar de compresión en cuanto al número y precisión sub-píxel de los vectores de movimiento de cada macrobloque, permite obtener reducciones drásticas en la latencia de la arquitectura. De esta manera, y a diferencia de los escasos trabajos previos, se obtiene una eficiente adaptación del número de ciclos a las características del codificador utilizado.

La validez de las arquitecturas de estimación de movimiento con precisión entera y de refinamiento sub-píxel aportadas, ha quedado demostrada mediante la comparación de las prestaciones de ambas con las propuestas aparecidas en la bibliografía más reciente. Este hecho, junto con los resultados obtenidos con el algoritmo VBS-ACBM aportado, tanto en términos de compresión como de coste computacional, hace que la combinación de las propuestas algorítmicas y arquitecturales realizadas en esta Tesis Doctoral constituya una contribución relevante en el campo de la estimación de movimiento multiestándar.

A modo de resumen, las aportaciones más significativas realizadas en esta Tesis Doctoral son las siguientes:

1. Se ha desarrollado un nuevo entorno de simulación que permite, mediante el estudio de parámetros relacionados con las características espaciales y temporales de la secuencia de vídeo a codificar, realizar un análisis detallado del proceso de estimación de movimiento desde el punto de vista de la función de coste de Lagrange J_{motion}.

2. Se ha propuesto una etapa de post-procesamiento de vectores de movimiento para la corrección de las limitaciones del algoritmo de estimación de movimiento por búsqueda exhaustiva en zonas de baja actividad espacial, con la consiguiente mejora de la tasa de compresión del sistema de codificación de vídeo.

3. Se ha propuesto un nuevo algoritmo de estimación de movimiento adaptativa, denominado ACBM (*Adaptive Cost Block Matching*), capaz de obtener, para secuencias de muy diferentes características espaciales y tasas de muestreo temporal, una reducción drástica en el coste computacional con respecto al algoritmo de búsqueda exhaustiva, garantizando idénticas prestaciones de compresión que el algoritmo exhaustivo.

4. Se ha adaptado el algoritmo ACBM para su posterior implementación hardware, dando lugar a un nuevo algoritmo denominado ACBM-HW (*Adaptive Cost Block Matching - HardWare*) que, haciendo uso del algoritmo de búsqueda exhaustiva y del algoritmo PBM-HW (*Predictive Block Matching - HardWare*) propuesto en esta Tesis, conserva las prestaciones de compresión del algoritmo ACBM original.

5. Se ha desarrollado un nuevo algoritmo, denominado VBS-ACBM (*Variable Block Size – Adaptive Cost Block Matching*), para estándares con estimación de movimiento con bloques de tamaño variable, incluyendo el estándar H.264/AVC con sus siete modos de estimación de movimiento. Dicho algoritmo garantiza, con un coste computacional significativamente menor, unas prestaciones de compresión iguales a las ofrecidas por el algoritmo de búsqueda exhaustiva multimodo.

6. Se han analizado y evaluado, de forma novedosa, las diferentes posibilidades de agrupación de elementos de proceso en arquitecturas de estimación de movimiento con precisión entera multimodo unidimensionales, en términos de latencia, coste hardware, y eficacia en la eliminación temprana de candidatos. Como resultado de este análisis, se ha establecido que la agrupación de cuatro elementos de proceso representa la mejor opción en función de los parámetros anteriormente mencionados.

7. Se han demostrado, tanto para arquitecturas multimodo de estimación de movimiento con precisión entera, como para arquitecturas multimodo de refinamiento sub-píxel, los beneficios de incorporar mecanismos de eliminación temprana de candidatos y de selección de modos activos en términos de ciclos inactivos en los elementos de proceso de las primeras arquitecturas, y de latencia en el caso de las segundas.

8. Se ha diseñado, sintetizado y evaluado una novedosa arquitectura multiestándar de estimación de movimiento con precisión entera, capaz de realizar dicho proceso de búsqueda haciendo uso del algoritmo de búsqueda exhaustiva multimodo, o bien, mediante el algoritmo predictivo PBM-HW propuesto. Dicha arquitectura es capaz de procesar, en tiempo real, secuencias de vídeo CIF@30fps (352×288 píxeles a razón de 30 fotogramas por segundo) a una frecuencia de trabajo de 55 MHz, incorporando mecanismos de eliminación temprana de candidatos y de selección de modos activos,

y utilizando para ello 70,93K puertas NAND2 equivalentes en tecnología CMOS de 0,25 μm.

9. Se ha diseñado, sintetizado y evaluado una novedosa arquitectura multiestándar de refinamiento de vectores de movimiento a coordenadas de medio píxel y/o cuarto de píxel. Dicha arquitectura es capaz de procesar, en tiempo real, secuencias de vídeo CIF@30fps (352×288 píxeles a razón de 30 fotogramas por segundo) a una frecuencia de trabajo de 45 MHz, incorporando mecanismos de eliminación temprana de candidatos y de selección de modos activos, y utilizando para ello 87,71K puertas NAND2 equivalentes en tecnología CMOS de 0,25 μm y 5,4 Kbytes de almacenamiento interno. La unión de esta arquitectura de refinamiento sub-píxel con la anteriormente mencionada de búsqueda de precisión entera, garantiza la obtención de vectores de movimiento, según las directrices del algoritmo VBS-ACBM aportado en esta Tesis Doctoral, en sistemas de codificación híbrida de vídeo con restricciones de funcionamiento en tiempo real basados en los estándares H.263 y H.264/AVC.

5.2 Líneas futuras

Las líneas de investigación que se pretenden continuar a partir del trabajo desarrollado en esta Tesis son las siguientes:

- Integración del estimador de movimiento propuesto en una plataforma hardware de compresión de vídeo. Una de las líneas prioritarias de investigación de la *División de Diseño de Sistemas Integrados* del *Instituto Universitario de Microelectrónica Aplicada* consiste en el desarrollo de una plataforma hardware/software de compresión H.264/AVC en la que los diferentes núcleos *IP* (*Intellectual Property*)

estén conectados mediante redes de conmutación integradas, más conocidas como redes-en-chip (*Network on Chip - NoC*). Dentro de esta plataforma de codificación está previsto que el núcleo de estimación de movimiento de la plataforma siga las bases establecidas en esta Tesis. De esta manera, se podrá evaluar con mayor fiabilidad la validez de las propuestas realizadas en esta Tesis, así como el impacto de los posibles cambios a realizar. Es de destacar que esta línea de investigación se enmarca dentro del proyecto *ENDIVIA* (***EN****torno de* ***D****iseño basado en* ***I****Ps con soporte para* ***V****erificación y depuración,* ***I****ntegración con redes en chip y* ***A****plicaciones multimedia*), financiado por el *Ministerio de Educación y Ciencia* (TEC2005-08138-C02-01/MIC) y que en la actualidad está siendo desarrollado en colaboración con la *División de Ingeniería Electrónica* de la *ETSII* de la *Universidad Politécnica de Madrid*.

- Validación de las propuestas realizadas en sistemas de codificación de vídeo escalable. El grupo de trabajo *JVT* (*Joint Video Team*), encargado del desarrollo del estándar H.264/AVC, ha apostado claramente, como siguiente paso del proceso de estandarización, por introducir diferentes niveles de escalabilidad dentro de la estructura del codificador híbrido. En este sentido, sería interesante comprobar si la estrategia de estimación de movimiento adaptativa propuesta en esta Tesis sigue siendo igualmente válida en este contexto, así como el impacto de las modificaciones a realizar, en caso de que fueran necesarias.

- Exploración de metodologías de diseño alternativas. Dentro del área de investigación destinada a la obtención de sistemas de compresión de vídeo para aplicaciones con restricciones de funcionamiento en tiempo real, están cobrando especial relevancia nuevas metodologías de diseño que permiten, de manera *semi-automática*, implementar bloques funcionales descritos en lenguajes de alto nivel (típicamente *C*), sobre una arquitectura determinada. De esta manera, se puede obtener un codificador

y/o decodificador híbrido de vídeo que, aunque típicamente presenta menores prestaciones que las obtenidas a través de la síntesis e implementación de un código descrito a nivel *RTL*, requiere de un esfuerzo de diseño considerablemente menor. En este sentido, la *División de Diseño de Sistemas Integrados* del *Instituto Universitario de Microelectrónica Aplicada* colabora con el *IMEC* (*Interuniversities MicroElectronics Center*) de Bélgica en la aplicación de estas metodologías para la implementación de codificadores y decodificadores H.264/AVC sobre arquitecturas reconfigurables de *grano grueso* (*Coarse-Grain Reconfigurable Architectures – CGRA*). Los primeros resultados, en cuya obtención ha participado el autor de esta Tesis Doctoral, demuestran la validez de esta metodología para los procesos de compensación de movimiento [HKL+06] y filtrado adaptativo [AKL+06] establecidos por el estándar H.264/AVC.

- Análisis y exploración arquitectural para aplicaciones de elevada resolución espacial y temporal. En este sentido, resulta de especial interés el estudio de arquitecturas para la estimación de movimiento en codificadores de vídeo integrados en cámaras de vídeo-vigilancia, pues ésta es una de las aplicaciones que, hoy en día, presenta mayor demanda dentro del mercado audiovisual. Debido a la elevada resolución espacio-temporal de las secuencias de entrada, resulta imprescindible la utilización de estimadores de movimiento con una gran capacidad de procesamiento. Sin embargo, con el objetivo de minimizar el tamaño de las cámaras, es necesario que el sistema de codificación, y por lo tanto, el estimador de movimiento, ocupe un área lo más pequeña posible. Este hecho determina la necesidad de investigar nuevos algoritmos y arquitecturas capaces de cumplir con dichos requisitos, probablemente basados en las características espaciales y temporales de las secuencias de vídeo captadas por las cámaras.

BIBLIOGRAFÍA

En esta bibliografía se recogen las publicaciones mencionadas en esta Tesis Doctoral. Asimismo, a la derecha de cada referencia, se indica el capítulo (c) y la página (p) en la que dicha referencia ha sido citada en el presente trabajo.

[4i2iDEC05]	"H.264/MEG-4 part 10 baseline video decoder IP core", 4i2i Communications Ltd., Technical Specifications, diciembre 2005.	(c1,p7)
[4i2iENC05]	"H.264 video encoder IP core", 4i2i Communications Ltd., Technical Specifications, agosto 2005.	(c1,p7)
[ABC+03]	D. ALFONSO, D. BAGNI, L. CELETTO Y L. PEZZONI, "Detailed rate-distortion analysis of H.264 video coding standard and comparison to MPEG-2/4", *Proceedings SPIE Visual Communications and Image Processing*, vol. 5150, pp. 891-902, junio 2003.	(c4,p131)
[ABP+03]	D. ALFONSO, D. BAGNI, D. PAU Y A. CHIMIENTO, "A performance analysis of H.264 video coding standard", *Proceedings Picture Coding Symposium*, pp. 23-28, abril 2003.	(c4,p131)
[ACB03]	M.E. AL-MUALLA, C.N. CANAGARAJAH Y D.R. BULL, "Reduced complexity motion estimation techniques: review and comparative study", *Proceedings IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, vol. 2, pp. 607-610, diciembre 2003.	(c1,p7) (c2,p46)
[AKL+06]	C. ARBELO, A. KANSTEIN, S. LÓPEZ, J.F. LÓPEZ Y M. BEREKOVIC, "Mapping control-intensive video kernels onto a coarse-grain reconfigurable architecture: the H.264/AVC deblocking filter", *Proceedings Design, Automation and Test in Europe Conference and Exhibition (DATE), enviado para su publicación*, 2006.	(c2,p39) (c5,p225)
[AKM+04]	A. AHMAD, N. KHAN, S. MASUD Y M.A. MAUD, "Efficient block size selection in H.264 video coding standard", *IEE Electronics Letters*, vol. 40, num. 1, pp. 19-21, enero 2004.	(c2,p60)
[ALL+03]	A. ÁLVAREZ, S. LÓPEZ, J.F. LÓPEZ Y R. SARMIENTO, "A 0.25µm technology arithmetic codec for mobile multimedia communicators", *Proceedings SPIE VLSI Circuits and Systems, International Symposium on Microtechnologies for the New Millennium*, vol. 5117, pp. 361-369, mayo 2003.	(c2,p27)
[ANS01]	A. AHMED, S.K. NANDY Y P. SATHYA, "Content adaptive motion estimation for mobile video encoders", *Proceedings IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 2, pp. 237-240, mayo 2001.	(c2,p49)
[AS00]	L. ALPARONE Y L. SANTURRI, "Weighted median refinement of motion vectors in H.263-based video coding", *Proceedings IEEE Packet Video Workshop*, mayo 2000.	(c3,p68)

[ATL+02] A. ÁLVAREZ, F. TOBAJAS, S. LÓPEZ, A, TEJERA, J.F. LÓPEZ Y R. SARMIENTO, "Design and implementation of an arithmetic codec for wavelet based image compression", *Proceedings XVII Design of Integrated Circuits and Systems Conference (DCIS)*, pp. 655-660, noviembre 2002. (c2,p27)

[AZL+06] I. AHMAD, W. ZHENG, J. LUO Y M. LIOU, "A fast adaptive motion estimation algorithm", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, num. 3, pp. 420-438, marzo 2006. (c2,p50) (c2,p51)

[BCL+05] D. BARRETO, G.M. CALLICÓ, S. LÓPEZ, L. GARCÍA Y A. NÚÑEZ, "Real-time super-resolution over raw video sequences", *SPIE Proceedings of VLSI Circuits and Systems, International Symposium on Microtechnologies for the New Millennium*, vol. 5837, pp. 628-637, mayo 2005. (c3,p67)

[BJV01] W. BURLESON, P. JAIN Y S. VENKATRAMAN, "Dynamically parameterized architectures for power-aware video coding: motion estimation and DCT", *Proceedings Second International Workshop on Digital and Computational Video*, pp. 4-12, febrero 2001. (c2,p49)

[BK97] V. BHASKARAN Y K. KONSTANTINIDES, Image and Video Compression Standards: Algorithms and Architectures, *Springer*, 1997. (c1,p4) (c2,p17)

[BT.601] Recommendation ITU-R BT.601-5, "Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios", octubre 1995. (c2,p18)

[BTG+01] W. BURLESON, R. TESSIER, D. GOECKEL, S. SWAMINATHAN, P. JAIN, J. EUH, S. VENKATRAMAN Y V. THYAGARAJAN, "Dynamically parameterized algorithms and architectures to exploit signal variations for improved performance and reduced power", *Proceedings IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP)*, vol. 2, pp. 901-904, mayo 2001. (c4,p160)

[CCC04] Y.C. CHANG, W.M. CHAO Y L.G. CHEN, "Platform-based MPEG-4 video encoder SOC design", *Proceedings IEEE Workshop on Signal Processing Systems*, pp. 251-256, octubre 2004. (c2,p58)

[CCC05] Y.H. CHEN, T.C. CHEN Y L.G. CHEN, "Hardware oriented content-adaptive fast algorithm for variable block-size integer motion estimation in H.264", *Proceedings International Symposium on Intelligent Signal Processing and Communication Systems*, pp. 341-344, diciembre 2005. (c2,p50) (c2,p51)

[CCH+06a] C.Y. CHEN, S.Y. CHIEN, Y.W. HUANG, T.C. CHEN, T.C. WANG Y L.G. CHEN, "Analysis and architecture design of variable block-size motion estimation for H.264/AVC", *IEEE Transactions on Circuits and Systems I*, vol. 53, num. 2, pp. 578-593, febrero 2006. (c2,p53) (c2,p54) (c2,p55) (c4,p146) (c4,p203) (c4,p204) (c4,p205)

[CCH+06b] C.Y. CHEN, S.Y. CHIEN, Y.W. HUANG, C.H. TSAI, C.Y. CHEN, T.W. CHEN Y L.G. CHEN, "Analysis and architecture design of an HDTV720p 30 frames/s H.264/AVC encoder", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, num. 6, pp. 673-688, junio 2006. (c1,p7) (c2,p55) (c2,p59) (c4,p196) (c4,p203) (c4,p209)

[CCL+04] M.J. CHEN, Y.Y. CHIANG, H.J. LI Y M.C. CHI, "Efficient multi-frame motion estimation algorithms for MPEG-4 AVC/JVT/H.264", *Proceedings International Symposium on Circuits and Systems (ISCAS)*, vol. 3, pp. 737-740, mayo 2004. (c4,p131)

[CD05] H.W. CHENG Y L.R. DUNG, "A content-based methodology for power-aware motion estimation architecture", *IEEE Transactions on Circuits and Systems II*, vol. 52, num. 10, pp. 631-635, octubre 2005. (c2,p50)

[CEG+98] G. COTÉ, B. EROL, M. GALLANT Y F. KOSSENTINI, "H.263+: video coding at low bit rates", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, num. 7, pp. 849-866, noviembre 1998. (c2,p35)

[CFP02] A. CHIMIENTI, C. FERRARIS Y D. PAU, "A complexity-bounded motion estimation algorithm", *IEEE Transactions on Image Processing*, vol. 11, num. 4, pp. 387-392, abril 2002. (c2,p47) (c3,p88) (c3,p109)

[CH97] S.C. CHENG Y H.M. HANG, "A comparison of block-matching algorithms mapped to systolic-array implementation", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, num. 5, pp. 741-757, octubre 1997. (c2,p45)

[CHC+05a] T.W. CHEN, Y.W. HUANG, T.C. CHEN, Y.H. CHEN, C.Y. TSAI Y L.G. CHEN, "Architecture design of H.264/AVC decoder with hybrid task pipelining for high definition videos", *Proceedings IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 3, pp. 2931-2934, mayo 2005. (c1,p7)

[CHC+05b] S.Y. CHIEN, Y.W. HUANG, C.Y. CHEN. H.H. CHEN Y L.G.CHEN, "Hardware architecture design of video compression for multimedia communication systems", *IEEE Communications Magazine*, vol. 43, num. 8, pp. 122-131, agosto 2005. (c4,p203)

[CHC04a] T.C. CHEN, Y.W. HUANG Y L.G. CHEN, "Analysis and design of macroblock pipelining for H.264/AVC VLSI architecture", *Proceedings IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 2, pp. 273-276, mayo 2004. (c1,p7) (c2,p35)

[CHC04b] T.C. CHEN, Y.W. HUANG Y L.G. CHEN, "Fully utilized and reusable architecture for fractional motion estimation of H.264/AVC", *Proceedings IEEE International Acoustics, Speech and Signal Processing (ICASSP)*, vol. 5, pp. 17-21, mayo 2004. (c2,p59) (c4,p129) (c4,p208) (c4,p211)

[CHW04] C.Y. CHO, S.Y. HUANG Y J.S. WANG, "A-TDB: a self adaptive block-matching algorithm for varying motion contents", *Proceedings Data Compression Conference (DCC)*, pp. 532, marzo 2004. (c2,p50)

[CJS+04] S.K. CHOI, J.G. JEON, W.S. SHIM, W.K. JANG Y V.H.S. HA, "Design and implementation of H.264-based video decoder for digital multimedia broadcasting", *Proceedings IEEE International Conference on Multimedia and Expo (ICME)*, vol. 1, pp. 149-152, junio 2004. (c1,p7)

[CK97] Y.K. CHEN Y S.Y. KUNG, "Rate optimization by true motion estimation", *Proceedings IEEE Workshop on Multimedia Signal Processing*, pp. 187-194, junio 1997. (c3,p68)

[CLC06] T.C. CHEN, C.J LIAN Y L.G. CHEN, "Hardware architecture design of an H.264/AVC video codec", *Proceedings Asia and South Pacific Conference on Design Automation*, pp. 750-757, enero 2006. (c4,p203)

[CLP+05a] G.M. CALLICÓ, S. LÓPEZ, R. PESET, R. SETHURAMAN, J.F. LÓPEZ, R. SARMIENTO Y A. NÚÑEZ, "Low-cost implementation of a super-resolution algorithm for real-time video applications", *Proceedings IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 6, pp. 6130-6133, mayo 2005. (c3,p67)

[CLP+05b] G.M. CALLICÓ, S. LÓPEZ, R. PESET, R. SETHURAMAN, A. NÚÑEZ, J.F. LÓPEZ, M. MARRERO Y R. SARMIENTO, "Practical considerations for real-time super-resolution implementation techniques over video coding platforms", *Proceedings SPIE VLSI Circuits and Systems, International Symposium on Microtechnologies for the New Millennium*, vol. 5837, pp. 613-627, mayo 2005. (c3,p67)

[CP03] C. HO Y L.M. PO, "Adjustable partial distortion search algorithm for fast block motion estimation", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, num. 1, enero 2003. (c2,p56)

[CP96] C.K. CHEUNG Y L.M. PO, "A hybrid adaptive search algorithm for fast block motion estimation", *Proceedings International Symposium on Signal Processing and its Applications (ISSPA)*, pp. 365-368, agosto 1996. (c2,p49)

[CPL+06] G.M. CALLICÓ, R. PESET, S. LÓPEZ, J.F. LÓPEZ, A. NÚÑEZ, R. SETHURAMAN Y R. SARMIENTO, "Low-cost super-resolution algorithms implementation over a HW/SW video compression platform", *EURASIP Journal on Applied Signal Processing*, vol. 2006, art. ID 84614, *aceptado para su publicación*, 2006. (c3,p67)

[CS96] Y.L. CHAN Y W.C. SIU, "New adaptive pixel decimation for block motion vector estimation", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, num. 1, pp.113-118, febrero 1996. (c2,p46)

[CSE00] C. CHRISTOPOULOS, A. SKODRAS Y T. EBRAHIMI, "The JPEG2000 still image coding system: and overview", *IEEE Transactions on Consumer Electronics*, vol. 46, num. 4, pp. 1103-1127, noviembre 2000. (c2,p27)

[CTY+02] J. CHOI, N. TOGAWA, M. YANAGISAWA Y T. OHTSUKI, "VLSI architecture for a flexible motion estimation with parameters", *Proceedings Design Automation Conference (DAC)*, pp. 7-11, enero 2002. (c2,p49)

[DGH+05] L. DENG, W. GAO, M.Z. HU Y Z.Z. JI, "An efficient hardware implementation for motion estimation of AVC standard", *IEEE Transactions on Consumer Electronics*, vol. 51, num. 4, pp. 1360-1366, noviembre 2005. (c2,p55) (c2,p57) (c4,p202) (c4,p203) (c4,p209)

[DH00] C. DU Y Y. HE, "Comparative study of motion estimation for low-bit-rate video coding", *Proceedings SPIE Visual Communications and Image Processing*, vol. 4067, pp. 1239-1249, junio 2000. (c2,p45)

[DRS05] T. DIAS, N. ROMA Y L. SOUSA, "Efficient motion vector refinement architecture for sub-pixel motion estimation systems", *IEEE Workshop on Signal Processing Systems Design and Implementation*, pp. 313-318, noviembre 2005. (c2,p58) (c4,p126) (c4,p143)

[DY98] V.L. DO Y K.Y. YUN, "A low power VLSI architecture for full-search block-matching motion estimation", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, num. 4, pp. 393-398, agosto 1998. (c2,p56) (c4,p126)

[ENB+02] M.A. ELGAMEL, B.R. NALLAMILLI, M.A. BAYOUMI Y S. MASHALY, "Systolic array architectures for full-search block matching motion estimation", *Proceedings International Workshop on Digital Computational Video*, pp. 108-115, noviembre 2002. (c2,p53)

[ESB00] M.A. ELGAMEL, A.M. SHAMS Y M.A. BAYOUMI, "A comparative analysis for low power motion estimation VLSI architectures", *Proceedings IEEE Workshop on Signal Processing Systems (SIPS)*, pp. 149-158, octubre 2000. (c2,p53) (c2,p56)

[ESX+01] M.A. ELGAMEL, A.M. SHAMS, X. XUELING Y M.A. BAYOUMI, "Enhanced low power motion estimation VLSI architectures for video compression", *Proceedings IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 4, pp. 474-477, mayo 2001. (c2,p53) (c2,p56) (c4,p150)

[Eve63] H. EVERETT III, "Generalized lagrange multiplier method for solving problems of optimum allocation of resources", *Operations Research*, vol. 11, num. 3, pp. 399-417, mayo-junio 1963. (c2,p40)

[FGW96] B. FURHT, J. GREENBERG Y R. WESTWATER, Motion estimation algorithms for video compression, *The International Series in Engineering and Computer Science*, Springer, 1996. (c2,p46)

[FLM+95] J. FENG, K.T. LO, H. MEHRPOUR Y A.E. KARBOWIAK, "Adaptive block matching motion estimation algorithm for video coding", *IEE Electronics Letters*, vol. 31, num. 18, pp. 1542-1543, agosto 1995. (c2,p50) (c2,p51)

[FLM+98] J. FENG, K.T. LO, H. MEHRPOUR Y A.E. KARBOWIAK, "Adaptive block matching algorithm for video compression", *IEE Proceedings Vision, Image and Signal Processing*, vol. 145, num. 3, pp. 173-178, junio 1998. (c2,p49) (c2,p51)

[FS98] V. FOTOPOULOS Y A.N. SKODRAS, "sMAE: an improved block matching criterion", *Proceedings International Conference on Electronics, Circuits and Systems*, vol. 3, pp. 519-522, septiembre 1998. (c2,p24)

[Gar04] MATÍAS JAVIER GARRIDO GONZÁLEZ, "Arquitectura versátil para la codificación de vídeo multi-estándar: aportaciones metodológicas para el diseño de sistemas reutilizables y sistemas en un chip", *Tesis Doctoral*, 2004. (c1,p6) (c4,p168)

[GSJ+02] M.J. GARRIDO, C. SANZ, M. JIMÉNEZ Y J.M. MENESES, "An FPGA implementation of a flexible architecture for H.263 video coding", *IEEE Transactions on Consumer Electronics*, vol. 48, num. 4, pp. 1056-1066, noviembre 2002. (c2,p58)

[H261] ITU-T Recommendation H.261, "Video codec for Audiovisual Services at p×64 kbit/s", 1993. (c1,p9) (c2,p27)

[H263]	ITU-T Recommendation H.263, "Video coding for low bit rate communication", version 1, 1995.	(c1,p6) (c2,p28) (c2,p44) (c4,p168)
[H263+]	ITU-T Recommendation H.263, "Video coding for low bit rate communication", version 2, 1998.	(c2,p29)
[H263++]	ITU-T Recommendation H.263, "Video coding for low bit rate communication" version 3, 2000.	(c2,p29)
[H263code]	Disponible en: ftp:// bonde.nta.no/pub/tmn [Online].	(c4,p198)
[H264]	Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264/ISO/IEC 14496-10 AVC), 2003.	(c1,p6) (c2,p29) (c2,p44) (c4,p168)
[H264code]	Disponible en: http://iphome.hhi.de/suehring/tml/index.htm [Online].	(c4,p198)
[Hab74]	A. HABIBI, "Hybrid coding of pictorial data", *IEEE Transactions on Communications*, vol. COM-22, num. 5, pp. 614-615, mayo 1974.	(c1,p4)
[HB98]	G. DE HAAN Y P.W.A.C. BIEZEN, "An efficient true-motion estimator using candidate vectors from a parametric motion model", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, num. 1, pp. 85-91, febrero 1998.	(c3,p81)
[HBH+93]	G. DE HAAN, P.W.A.C. BIEZEN, H. HUIJGEN Y O.A. OJO, "True-motion estimation with 3-D recursive search block matching", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 3, num. 5, pp. 368 – 379, octubre 1993.	(c2,p47) (c3,p81)
[HCH+04]	Y.W. HUANG, S.Y. CHIEN, B.Y. HSIEH Y L.G. CHEN, "Global elimination algorithm and architecture design for fast block matching motion estimation", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, num. 6, pp. 898-907, junio 2004.	(c2,p56) (c4,p150)
[HCT+05]	Y.W. HUANG, T.C. CHEN, C.H. TSAI, C.Y. CHEN, T.W. CHEN, C.S. CHEN, C.F. SHEN, S.Y. MA, T.C. WANG, B.Y. HSIEH, H.C. FANG Y L.G. CHEN, "A 1.3 TOPS H.264/AVC single-chip encoder for HDTV applications", *Digest of Technical Papers, IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 1, pp. 128-129, febrero 2005.	(c4,p203)

[HCT+06] Y.W. HUANG, C.Y. CHEN, C.H. TSAI, C.F. SHEN Y L.G. CHEN, "Survey on block matching motion estimation algorithms and architectures with new results", *Journal of VLSI Signal Processing*, vol. 42, num. 3, pp. 297-320, marzo 2006. (c1,p7) (c2,p47) (c2,p53) (c3,p88)

[HCW05] S.Y. HUANG, C.Y. CHO Y J.S. WANG, "Adaptive fast block-matching algorithm by switching search patterns for sequences with wide-range motion content", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, num. 11, pp. 1373-1384, noviembre 2005. (c2,p50) (c2,p51)

[HHC+06] Y.W. HUANG, B.Y. HSIEH, S.Y. CHIEN, S.Y. MA Y L.G. CHEN, "Analysis and complexity reduction of multiple reference frames motion estimation in H.264/AVC", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, num. 4, pp. 507-522, abril 2006. (c1,p7) (c2,p35) (c4,p131)

[HKL+06] H. HERNÁNDEZ, A. KANSTEIN, S. LÓPEZ, J.F. LÓPEZ Y M. BEREKOVIC, "Mapping of the H.264/AVC motion compensation algorithm onto coarse-grain reconfigurable array", *Proceedings XXI Design of Integrated Circuits and Systems (DCIS) Conference, aceptado para su publicación*, 2006. (c2,p58) (c5,p225)

[HLK04] T. HA, S. LEE Y J. KIM, "Motion compensated frame interpolation by new block-based motion estimation algorithm", *IEEE Transactions on Consumer Electronics*, vol. 50, num. 2, pp. 752-759, mayo 2004. (c3,p67)

[Hos03] P.I. HOSUR, "Motion adaptive search for fast motion estimation", *IEEE Transactions on Consumer Electronics*, vol. 49, num. 4, pp. 1330-1340, noviembre 2003. (c2,p49) (c2,p51)

[HSC05] K.C. HUI, W.C. SIU Y Y.L. CHAN, "New adaptive partial distortion search using clustered pixel matching error characteristic", *IEEE Transactions on Image Processing*, vol. 14, num. 5, pp. 597-607, mayo 2005. (c2,p56)

[HSM+04] Y. HU, A. SIMPSON, K. MCADOO Y J. CUSH, "A high definition H.264/AVC hardware video decoder core for multimedia SoC's", *Proceedings IEEE International Symposium on Consumer Electronics*, pp. 385-389, septiembre 2004. (c1,p7)

[Hui06] C. HUI, Analysis and motion estimation strategies for frame and video object coding, *ProQuest/UMI*, 2006. (c2,p46)

[HWH+03] Y.W. HUANG, T.C. WANG, B.Y. HSIEH Y L.G. CHEN, "Hardware architecture design for variable block size motion estimation in MPEG-4 AVC/JVT/ITU-T H.264", *Proceedings International Symposium on Circuits and Systems (ISCAS)*, vol. 2, pp. 796-799, mayo 2003. (c2,p55) (c2,p57) (c4,p202) (c4,p203) (c4,p209)

[JC04] X. JING Y L.P. CHAU, "Fast approach for H.264 inter mode decision", *IEE Electronics Letters*, vol. 40, num. 17, pp. 1050-1052, agosto 2004. (c2,p60)

[JCK99] J.C.H. JU, Y.K. CHEN Y S.Y. KUNG, "A fast rate-optimized motion estimation algorithm for low-bit-rate video coding", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, num. 7, pp. 994-1002, octubre 1999. (c3,p68)

[JJ81] J.R. JAIN Y A.K. JAIN, "Displacement measurement and its application in interframe image coding", *IEEE Transactions on Communications*, vol. COM-29, num. 12, pp. 1799-1808, diciembre 1981. (c1,p6)

[JLB+04] P. JAIN, A. LAFFELY, W. BURLESON, R. TESSIER Y D. GOECKEL, "Dynamically parameterized algorithms and architectures to exploit signal variations", *Journal of VLSI Signal Processing*, vol. 36, num.1, pp. 27-40, noviembre 2004. (c2,p50)

[JPEG] ISO/IEC JTC1 10918-1, ITU-T Recommendation T.81, "Information Technology – Digital compression and coding of continuous-tone still images: requirements and guidelines", 1994. (c2,p26)

[JPEG2K] ISO/IEC JTC1/SC29/WG1, "Information technology - JPEG 2000 image coding system: Core coding system", marzo 2000. (c2,p26)

[JZC03] X. JING, C. ZHU Y L.P. CHAU, "Smooth constrained block matching criterion for motion estimation", *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 3, pp. 661-664, abril 2003. (c3,p66)

[KA03] N. KAMACI Y Y. ALTUNBASAK, "Performance comparison of the emerging H.264 video coding standard with the existing standards", *Proceedings International Conference on Multimedia and Expo (ICME)*, vol. 1, pp. 345-348, julio 2003. (c2,p29)

[KHC05] M. KIM, I. HWANG Y S.I. CHAE, "A fast VLSI architecture for full-search variable block size motion estimation in MPEG-4 AVC/H.264", *Proceedings Asia and South Pacific Design Automation Conference (ASP-DAC)*, vol. 1, pp. 631-634, enero 2005. (c2,p55) (c2,p57) (c4,p202) (c4,p203) (c4,p209)

[KJB+04] H.Y. KANG, K.A. JEONG, J.Y. BAE, Y.S. LEE Y S.H. LEE, "MPEG4 AVC/H.264 decoder with scalable bus architecture and dual memory controller", *Proceedings International Symposium on Circuits and Systems (ISCAS)*, vol. 2, pp. 145-148, mayo 2004. (c1,p7)

[KKA05] H. KIM, N. KAMACI Y Y. ALTUNBASAK, "Low-complexity rate-distortion optimal macroblock mode selection and motion estimation for MPEG-like video coders", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, num. 7, pp. 823-834, julio 2005. (c2,p42)

[Kol03] HELGA KOLB, "How the Retina Works," *American Scientist*, vol. 91, num.1, pp. 28-35, enero 2003. (c2,p19)

[Kor07] ARTHUR KORN, "La télégraphie des images", *Je sais tout*, num. 27, abril 1907. (c1,p3)

[Kuh99] PETER KUHN, Algorithms, complexity, analysis and VLSI architectures for MPEG-4 motion estimation, *Kluwer Academic Publishers*, 1999. (c1,p7) (c2,p45) (c2,p53)

[LAL+06] Y. LIANG, I. AHMAD, J. LUO, Y. SUN Y V. SWAMINATHAN, "A fast adaptive motion estimation using hierarchical history of motion intensity in H.264/AVC", *IEEE Transactions on Circuits and Systems for Video Technology, aceptado para publicación*, 2006 (c2,p51)

[LC01] J.H. LIM Y H.W. CHOI, "Adaptive motion estimation algorithm using spatial and temporal correlation", *Proceedings IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, pp. 473-476, agosto 2001. (c2,p49)

[LCL+01] J.F. LÓPEZ, P. CORTÉS, S. LÓPEZ Y R. SARMIENTO, "Gallium Arsenide Processing Elements for Motion Estimation Full Search Algorithm", *SPIE Proceedings Electronics and Structures for MEMS*, vol. 4591, pp. 101-112, noviembre 2001. (c4,p132)

[LCL+02] J.F. LÓPEZ, P. CORTÉS, S. LÓPEZ Y R. SARMIENTO, "Design of a 270 MHz/340 mW processing element for high performance motion estimation systems application", *Microelectronics Journal*, vol. 33, num. 12, pp. 1123-1134, diciembre 2002. (c2,p46) (c4,p132)

[LCL+03a] S. LÓPEZ, G.M. CALLICÓ, J.F. LÓPEZ Y R. SARMIENTO, "High confident characterization for block-matching true motion vectors", *Proceedings of the Work In Progress Session of the 29th Euromicro Conference*, pp. 79-80, septiembre 2003. (c3,p69)

[LCL+03b] S. LÓPEZ, G.M. CALLICÓ, J.F. LÓPEZ Y R. SARMIENTO, "Adaptive motion vector post-processing for low cost rate-distortion optimisation", *IEE Electronics Letters*, vol. 39, num. 24, pp. 1720-1721, diciembre 2003. (c3,p82)

[LCL+05a] S. LÓPEZ, G.M. CALLICÓ, J.F. LÓPEZ Y R. SARMIENTO, "Compresión de vídeo mediante técnicas de post-procesamiento", *Revista Vector PLUS*, Fundación Universitaria de Las Palmas, num. 25, pp. 27-36, enero 2005. (c3,p82)

[LCL+05b] S. LÓPEZ, G.M. CALLICÓ, J.F. LÓPEZ Y R. SARMIENTO, "A high quality/low computational cost technique for block matching motion estimation", *Proceedings Design, Automation and Test in Europe Conference and Exhibition (DATE)*, vol. 3, pp. 2-7, marzo 2005. (c3,p87)

[LCL+05c] S. LÓPEZ, G.M. CALLICÓ, J.F. LÓPEZ Y R. SARMIENTO, "A low-cost bidimensional smart pixel network for video coding operations", *Proceedings SPIE VLSI Circuits and Systems, International Symposium on Microtechnologies for the New Millennium*, vol. 5837, pp. 638-649, mayo 2005. (c2,p27)

[LCT+04] S. LÓPEZ, R. CALZADA, A. TEJERA, J.F. LÓPEZ Y R. SARMIENTO, "Real time smart pixels processing array for mobile multimedia applications", *Proceedings XIX Integrated Circuits and Systems Conference (DCIS)*, pp. 386-391, noviembre 2004. (c2,p27)

[LJI+05] S. LI, Y. JIANG, T. IKENAGA Y S. GOTO, "Content-based motion estimation with extended temporal-spatial analysis", *IEICE Transactions on Information and Systems*, vol. E88-D, num. 7, pp. 1561-1567, julio 2005. (c2,p50) (c2,p51)

[LKP03] S.H. LEE, O. KWON Y R.H. PARK, "Weighted-adaptive motion-compensated frame rate up-conversion", *IEEE Transactions on Consumer Electronics*, vol. 49, num. 3, pp. 485-492, agosto 2003. (c3,p67)

[LL04] J.H. LEE Y N.S. LEE, "Variable block size motion estimation algorithm and its hardware architecture for H.264/AVC", *Proceedings International Symposium on Circuits and Systems (ISCAS)*, vol. 3, pp. 741-744, mayo 2004. (c2,p47)

[LLS+01] J.H. LEE, K.W. LIM, B.C. SONG Y J.B. RA, "A fast multi-resolution block matching algorithm and its LSI architecture for low bit-rate video coding", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, num. 12, pp. 1289-1301, diciembre 2001. (c2,p47)

[LLS05] S. LÓPEZ, J.F. LÓPEZ Y R. SARMIENTO, "Cost-adaptive motion estimation strategy for high-performance video encoders", *IEE Electronics Letters*, vol. 41, num. 4, pp. 182-183, febrero 2005. (c3,p87)

[LLT+01] J.F. LÓPEZ, S. LALCHAND, F. TOBAJAS, S. LÓPEZ, A. NÚÑEZ Y R. SARMIENTO, "Gallium Arsenide multiplierless filter bank for two dimensional discrete wavelet transform (2D-DWT) computation", *Proceedings SPIE International Symposium on Smart Electronics and MEMs*, vol. 4951, pp. 273-280, noviembre 2001. (c2,p27)

[LO04] Y. LIU Y S. ORAINTARA, "Complexity comparison of fast block-matching motion estimation algorithms", *Proceedings IEEE International Acoustics, Speech and Signal Processing (ICASSP)*, vol. 3, pp. 341-344, mayo 2004. (c1,p7) (c2,p46)

[LPK+06] S.H. LEE, J.H. PARK, S.W. KIM, S.J. KO Y S. KIM, "Implementation of H.264/AVC decoder for mobile video applications", *Proceedings Asia and South Pacific Conference Design Automation*, pp. 120-121, enero 2006. (c1,p7)

[LTC+06] S. LÓPEZ, F. TOBAJAS, G.M. CALLICÓ, P.PÉREZ Y R. SARMIENTO, "A novel high performance architecture for H.264/AVC deblocking filtering", *IEE Electronics Letters, enviado para su publicación*, 2006. (c2,p39)

[LTC04] S.S. LIN, P.C. TSENG Y L.G. CHEN, "Low power parallel tree architecture for full search block-matching motion estimation", *Proceedings International Symposium on Circuits and Systems (ISCAS)*, vol. 2, pp. 313-316, mayo 2004. (c4,p126) (c4,p143) (c4,p150)

[LTL+02] J.F. LÓPEZ, F. TOBAJAS, S. LÓPEZ, P. CORTÉS, S. LALCHAND Y R. SARMIENTO, "VLSI video processing elements for real time applications", *Proceedings 28th Annual Conference of the IEEE Industrial Electronics Society (IECON 2002)*, noviembre 2002. (c4,p132)

[LTV+05a] S. LÓPEZ, F. TOBAJAS, A. VILLAR, V. DE ARMAS, J.F. LÓPEZ Y R. SARMIENTO, "Low cost efficient architecture for H.264 motion estimation", *Proceedings IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 1, pp. 412-415, mayo 2005. (c2,p50) (c4,p136)

[LTV+05b] S. LÓPEZ, F. TOBAJAS, A. VILLAR, J. BIENES, V. DE ARMAS, G.M. CALLICÓ, J.F. LÓPEZ Y R. SARMIENTO, "A quarter pixel precision motion estimation architecture for H.264/AVC video coding", *SPIE Proceedings of VLSI Circuits and Systems, International Symposium on Microtechnologies for the New Millennium*, vol. 5837, pp. 174-184, mayo 2005. (c4,p170)

[LW06] V. LIGUORI Y K. WONG, "Designing a real-time HDTV 1080p baseline H.264/AVC encoder core", *DesignCon 2006*, 2006. (c1,p7)

[LWW05] M. LI, R. WANG Y W. WU, "The high throughput and low memory access design of sub-pixel interpolation for H.264/AVC HDTV decoder", *Proceedings IEEE Workshop on Signal Processing Systems Design and Implementation*, pp. 296-301, noviembre 2005. (c2,p58)

[LYL+05] W.N. LIE, H.C. YEH, T.C.I. LIN Y C.F. CHEN, "Hardware-efficient computing architecture for motion compensation interpolation in H.264 video coding", *Proceedings IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 3, pp. 2136-2139, mayo 2005. (c4,p173)

[LZ93] B. LIU Y A. ZACCARIN, "New fast algorithms for the estimation of block motion vectors", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 3, num. 2, pp. 148-157, abril 1993. (c2,p46)

[Mar03] GUSTAVO MARRERO CALLICÓ, "Real-time and low-cost super-resolution algorithms onto hybrid video encoders", *Tesis Doctoral*, Universidad de Las Palmas de Gran Canaria, julio 2003. (c3,p67)

[MGL05] S. MA, W. GAO Y Y. LU, "Rate-distortion analysis for H.264/AVC video coding and its application to rate control", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, num. 12, pp. 1533-1544, diciembre 2005. (c2,p40)

[MGS+02] H. MAHMOUD, S. GOEL, M. SHAABAN, T. DARWISH Y M. BAYOUMI, "A low power VLSI architecture for multi-stage interval-based motion estimation (MIME) algorithm", *Proceedings International Workshop on Digital and Computational Video*, pp. 159-166, noviembre 2002. (c4,p150)

[MKC00] F. MOSCHETTI, M. KUNT Y F. CALVANO, "A nested-multilevel redundancy exploitation for fast block matching", *Proceedings International Conference on Image Proceesing (ICIP)*, vol. 1, pp. 856-859, septiembre 2000. (c2,p49) (c2,p51)

[MKD03] F. MOSCHETTI, M. KUNT Y E. DEBES, "A statistical adaptive block-matching motion estimation", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, num. 4, pp. 417-431, abril 2003. (c2,p50)

[MM02] V.G. MOSHNYAGA Y K. MASUNAGA, "Reducing computational complexity of adaptive motion estimation through binary comparison", *Proceedings International Symposium on Circuits and Systems (ISCAS)*, vol. 2, pp. 484-487, mayo 2002. (c2,p49)

[MPEG1] ISO/IEC 11172: "Information technology—coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s", 1993. (c2,p27)

[MPEG2] ISO/IEC 13818–2: "Generic coding of moving pictures and associated audio information—Part 2: Video", (*también ITU-T Recommendation H.262*), 1994. (c2,p27)

[MPEG21] ISO/IEC JTC1/SC29/WG11N5525, "MPEG-21, Overview v.9", 2003. (c2,p30)

[MPEG4] ISO/IEC 14496–2: "Information technology—coding of audiovisual objects—part 2: visual", 2000. (c1,p6) (c2,p28) (c4,p169)

[MPEG7] ISO/IEC 15938-3: "Multimedia Content Description Interfaces. Part 3: Visual", 2002. (c2,p30)

[MQ05] B. MONTRUCCHIO Y D. QUAGLIA, "New sorting-based lossless motion estimation algorithms and a partial distortion elimination performance analysis", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, num. 2, pp. 210-220, febrero 2005. (c2,p56)

[MS99] J. MINOCHA Y N.R. SHANBHAG, "A low power data-adaptive motion estimation algorithm", *Proceedings IEEE Workshop on Multimedia Signal Processing*, pp. 685-690, septiembre 1999. (c2,p50)

[NKP+95] K.M. NAM, J.S. KIM, R.H. PARK Y Y.S. SHIM, "A fast hierarchical motion vector estimation algorithm using mean pyramid", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, num. 4, pp. 344-351, agosto 1995. (c2,p47)

[NM05] Y. NIE Y K.K. MA, "Adaptive irregular pattern search with matching prejudgment for fast block-matching motion estimation", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, num. 6, pp. 789-794, junio 2005. (c2,p50) (c2,p51)

[Nomadik04] "Nomadik – open multimedia platform for next generation mobile devices", STMicroelectronics, TA305 Technical article, 2004. (c1,p4)

[Nomadik06] "Nomadik mobile multimedia application processor", STMicroelectronics, STn8815 Data Brief, 2006. (c1,p7)

[OBL+04] J. OSTERMANN, J. BORMANS, P. LIST, D. MARPE, M. NARROSCHKE, F. PEREIRA, T. STOCKHAMMER Y T. WEDI, "Video coding with H.264/AVC: tools, performance and complexity", *IEEE Circuits and Systems Magazine*, vol. 4, num. 1, pp. 7-28, enero 2004. (c1,p6) (c2,p29) (c2,p35)

[OL00] H.S. OH Y H.K. LEE, "Block-matching algorithm based on an adaptive reduction of the search area for motion estimation", *Real-Time Imaging Journal*, vol.6, num.5, pp. 407-414, octubre 2000. (c2,p50) (c2,p51)

[OL98] H.S. OH Y H.K. LEE, "Adaptive adjustment of the search window for block-matching algorithm with variable block size", *IEEE Transactions on Consumer Electronics*, vol. 44, num. 3, pp. 659-666, agosto 1998. (c2,p50) (c2,p51)

[OLH05] C.M. OU, C.F. LE Y W.J. HWANG, "An efficient VLSI architecture for H.264 variable block size motion estimation", *IEEE Transactions on Consumer Electronics*, vol. 51, num. 4, pp. 1291-1299, noviembre 2005. (c2,p55) (c4,p205) (c4,p206) (c4,p209)

[OWX03] E. ONG, H. WANG Y P. XUE, "Video coding based on true motion estimation", *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 409-412, abril 2003. (c3,p68)

[PB98] S.R. PARK Y W. BURLESON, "Reconfiguration for power saving in real-time motion estimation", *Proceedings IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP)*, vol. 5, pp. 3037-3040, mayo 1998. (c2,p51)

[PCJ+05] S. PARK, H. CHO, H. JUNG Y D. LEE, "An implemented of H.264 video decoder using hardware and software", *Proceedings IEEE Custom Integrated Circuits Conference*, pp. 271-275, septiembre 2005. (c1,p7)

[PKJ+06] S.H. PARK, H.K. KIM, J.W. JUNG Y S.J. KO, "Efficient SVC encoding scheme for the video transmission over various networks", *Proceedings International Conference on Consumer Electronics (ICCE)*, pp. 487-488, enero 2006. (c2,p32)

[PM92] W.B. PENNEBAKER Y J.L. MITCHELL, JPEG still data compression standard, *Springer*, 1992. (c2,p26)

[PSA+03] R. PESET-LLOPIS, R. SETHURAMAN, C. ALBA-PINTO, H. PETERS, S. MAUL Y M. OOSTERHUIS, "A low-cost and low-power multi-standard video encoder", *Proceedings IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and Systems Synthesis*, pp. 97-102, octubre 2003 . (c1,p6)

[RB05] C.A. RAHMAN Y W. BADAWY, "A quarter pel full search block motion estimation architecture for H.264/AVC", *Proceedings IEEE International Conference on Multimedia and Expo (ICME)*, pp. 414-417, julio 2005. (c2,p59)

[Ric02] I. RICHARDSON, Video codec design: developing image and video compression systems, *John Wiley & Sons*, 2002. (c2,p17)

[Ric03]	I. RICHARDSON, H.264 and MPEG-4 video compression: video coding for next generation multimedia, *John Wiley & Sons*, 2003.	(c1,p5) (c2,p17) (c2,p22) (c2,p39)
[RPR77]	J.A. ROESSE, W.K. PRATT Y G.S. ROBINSON, "Interframe cosine transform image coding", *IEEE Transactions on Communications*, vol. COM-25, num. 11, pp. 1329-1339, noviembre 1977.	(c1,p4)
[Sad02]	A.H. SADKA, Compressed video communications, *John Wiley & Sons*, 2002.	(c1,p5) (c3,p67)
[San98]	CÉSAR SANZ ÁLVARO, "Arquitecturas VLSI para la estimación de movimiento en codificación de imágenes", *Tesis Doctoral*, Universidad Politécnica de Madrid, marzo 1998.	(c2,p53)
[SB03]	M. SAYED Y W. BADAWY, "A half-pel motion estimation architecture for MPEG-4 applications", *Proceedings IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 2, pp. 792-795, mayo 2003.	(c2,p58)
[SDL+04]	S. SAPONARA, K. DENOLF, G. LAFRUIT, C. BLANCH Y J. BORMANS, "Performance and complexity co-evaluation of the advanced video coding standard for cost-effective multimedia communications", *EURASIP Journal on Applied Signal Processing*, vol. 2, pp. 220-235, 2004.	(c1,p6) (c2,p35)
[SF04]	S. SAPONARA Y L. FANUCCI, "Data-adaptive motion estimation algorithm and VLSI architecture design for low-power video systems", *IEE Proceedings on Computers and Digital Techniques*, vol. 151, num. 1, pp. 51-59, enero 2004.	(c2,p50) (c2,p53)
[SG88]	Y. SHOHAM Y A. GERSHO, "Efficient bit allocation for an arbitrary set of quantizers", *IEEE Transactions on Speech Signal Processing*, vol. 36, num. 9, pp. 1445-1453, septiembre 1988.	(c2,p40)
[Sik05]	T. SIKORA, "Trends and perspectives in image video coding", *Proceedings of the IEEE*, vol. 93, num. 1, pp. 6-17, enero 2005.	(c1,p5)
[SK97]	G.M. SCHUSTER Y A. KATSAGGELOS, Rate-Distortion Based Video Compression: Optimal Video Frame Compression and Object Boundary Encoding, *Springer*, 1997.	(c2,p40)
[SKM+04]	C.A. SEGALL, A.K. KATSAGGELOS, R. MOLINA Y J. MATEOS, "Bayesian resolution enhancement of compressed video", *IEEE Transactions on Image Processing*, vol. 13, num. 7, pp. 898-911, Julio 2004.	(c3,p67)

[SLG+05] Y. SONG, Z. LIU, S. GOTO Y T. IKENAGA, "A VLSI architecture for motion compensation interpolation in H.264/AVC", *Proceedings International Conference on ASIC (ASICON)*, vol. 1, pp. 279-282, octubre 2005. (c2,p58)

[SLG+06] Y. SONG, Z. LIU, S. GOTO Y T. IKENAGA, "Scalable VLSI architecture for variable block size integer motion estimation in H.264/AVC", *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E89-A, num. 4, pp. 979-987, abril 2006. (c2,p55) (c4,p206) (c4,p209)

[SM02] A. SHARIFINEJAD Y H. MEHRPOUR, "A fast full search block matching algorithm using three window search based on the statistical analysis of the motion vectors", *Proceedings IEEE International Conference on Communications (ICC)*, vol. 1, pp. 104-108, abril-mayo 2002. (c2,p50)

[SMS+05] H. SCHWARZ, D. MARPE, T. SCHIERL Y T. WIEGAND, "Combined scalability support for the scalable extension of H.264/AVC", *Proceedings IEEE International Conference on Multimedia and Expo (ICME)*, julio 2005. (c2,p31)

[SN05] T. SIHVO Y J. NIITTYLAHTI, "H.264/AVC interpolation optimization", *Proceedings IEEE Workshop on Signal Processing Systems Design and Implementation*, pp. 307-312, noviembre 2005. (c2,p58)

[Sou99] L.A. SOUSA, "Applying conditional processing to design low-power array processors for motion estimation", *Proceedings International Conference on Image Processing (ICIP)*, vol. 2, pp. 769-773, octubre 1999. (c2,p56) (c4,p126) (c4,p143) (c4,p150)

[SR01] M.T. SUN Y A. REIBMAN, Compressed video over networks, *Marcel Dekker*, 2001. (c3,p67)

[SR85] R. SRINIVASAN Y K.R. RAO, "Predictive coding based on efficient motion estimation", *IEEE Transactions on Communications*, vol. COM-33, num. 8, pp. 888-896, agosto 1985. (c1,p7)

[SR99] L. SOUSA Y N. ROMA, "Low-power array architectures for motion estimation", *Proceedings Workshop on Multimedia Signal Processing*, pp. 679-684, septiembre 1999. (c2,p56) (c4,p126) (c4,p143) (c4,p150)

[SS04] Y. SU Y M.T. SUN, "Fast multiple reference frame motion estimation for H.264", *IEEE Transactions on Circuits and Systems*, vol. 16, num. 3, pp. 447-452, marzo 2006. (c4,p131)

[SSM+05] R.SCHAFER, H. SCHWARZ, D. MARPE, T. SCHIERL Y T. WIEGAND, "MCTF and scalability extension of H.264/AVC and its applications to video transmission, storage and surveillance", *Proceedings SPIE Visual Communications and Image Processing (VCIP)*, vol. 5960, pp. 343-354, julio 2005. (c2,p31)

[SVC] Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6), "Joint scalable video model JSVM-6", marzo-abril 2006. (c1,p8) (c2,p31)

[SW05] G.J. SULLIVAN Y T. WIEGAND, "Video compression – from concepts to the H.264/AVC standard", *Proceedings of the IEEE*, vol. 93, num. 1, pp. 18-31, enero 2005. (c1,p6)

[TA02] Y. TSAIG Y A. AVERBUCH, "Automatic segmentation of moving objects in video sequences: a region labelling approach", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, num. 7, pp. 597-612, julio 2002. (c3,p68)

[TAL02] A.M. TOURAPIS, O.C. AU Y M.L. LIOU, "Highly efficient predictive zonal algorithms for fast block-matching motion estimation", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, num. 10, pp. 934-947, octubre 2002. (c2,p47)

[TC01] D.S. TURAGA Y T. CHEN, "Estimation and mode decision for spatially correlated motion sequences", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, num. 10, pp. 1098-1107, octubre 2001. (c3,p68)

[TC04] P.C. TSENG Y L.G. CHEN, "Hardware architecture design for visual processing: present and future", *Proceedings IEEE Asia-Pacific Conference on Advanced System Integrated Circuits*, pp. 6-9, agosto 2004. (c1,p5)

[TCH+05] P.C. TSENG, Y.C. CHANG, Y.W. HUANG, H.C. FANG, C.T. HUANG Y L.G. CHEN, "Advances in hardware architectures for image and video coding – a survey", *Proceedings of the IEEE*, vol. 93, num. 1, pp. 184-197, enero 2005. (c1,p5)

[TLW01] P.L. TAI, C.T. LIU Y J.S. WANG, "Complexity-adaptive search algorithm for block motion estimation", *Proceedings International Conference on Image Processing (ICIP)*, vol. 2, pp. 969-972, octubre 2001. (c2,p49)

[TR03] A. TAMHANKAR Y K.R. RAO, "An overview of H.264/MPEG-4 part 10", *Proceedings EURASIP Conference focused on Video/Image Processing and Multimedia Communications*, vol. 1, pp. 1-51, julio 2003. (c1,p6)

[TS04] Y.P. TAN Y H. SUN, "Fast motion re-estimation for arbitrary downsizing video transcoding using H.264/AVC standard", *IEEE Transactions on Consumer Electronics*, vol. 50, num. 3, pp. 887-894, agosto 2004. (c3,p68)

[VS89] L. DE VOS Y M. STEGHERR, "Parameterizable VLSI architectures for the full-search block-matching algorithm", *IEEE Transactions on Circuits and Systems*, vol. 36, num. 10, pp. 1309-1316, octubre 1989. (c2,p54)

[WG01] T. WIEGAND Y B. GIROD, "Lagrange multiplier selection in hybrid video coder control", *Proceedings International Conference on Image Processing (ICIP)*, vol. 3, pp. 542-545, octubre 2001. (c2,p44)

[WG05a] C. WEI Y M.Z. GANG, "A novel VLSI architecture for VBSME in MPEG-4 AVC/H.264", *Proceedings IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 2, pp. 1794-1797, mayo 2005. (c2,p55) (c2,p57) (c4,p202) (c4,p203) (c4,p209)

[WG05b] C. WEI Y M.Z. GANG, "Reconfigurable VLSI architecture for VBSME in MPEG-4 AVC/H.264", *Proceedings International Conference on ASIC*, vol. 1, pp. 265-269, octubre 2005. (c2,p55) (c4,p206)

[Woo05] C. WOOTTON, A Practical Guide to Video and Audio Compression: From Sprockets and Rasters to Macro Blocks, *Focal Press*, 2005. (c2,p17)

[WPL+05] D. WU, F. PAN, K.P. LIM, S. WU, Z.G. LI, X. LIN, S. RAHARDJA Y C.C. KO, "Fast intermode decision in H.264/AVC video coding", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, num. 6, pp. 953-958, julio 2005. (c2,p60)

[WSB+03] T. WIEGAND, G.J. SULLIVAN, G. BJØNTERGAARD Y A. LUTHRA, "Overview of the H.264/AVC video coding standard", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, num. 7, pp. 560-576, julio 2003. (c1,p6) (c4,p130)

[WSJ+03] T. WIEGAND, H. SCHWARZ, A. JOCH, F. KOSSENTINI Y G.J. SULLIVAN, "Rate-constrained coder control and comparison of video coding standards", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, num. 7, pp. 688-703, julio 2003. (c2,p40) (c2,p42) (c2,p44)

[WWK00] Y. WANG, Y. WANG Y H. KURODA, "A globally adaptive pixel-decimation algorithm for block-motion estimation", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, num. 6, pp. 1006-1011, septiembre 2000. (c2,p46)

[WYG+04] C. WEI, Z. YAN, M.Z. GANG Y L.Z. QIANG, "VLSI architecture design for variable-size block motion estimation in MPEG-4 AVC/H.264", *Proceedings IEEE Asia-Pacific Conference on Circuits and Systems*, vol. 1, pp. 617-620, diciembre 2004. (c2,p55) (c4,p206)

[WYZ+04] H. WEI-FENG, Z. YAN, G. ZHI-QIANG Y M. ZHI-GANG, "Implementation of half-pel motion estimation IP core for MPEG-4 ASP@L5 texture coding", *Proceedings IEEE Asia-Pacific Conference on Circuits and Systems*, vol. 1, pp. 149-152, diciembre 2004. (c2,p58)

[XLS05] J. XIN, C.W. LIN Y M.T. SUN, "Digital video transcoding", *Proceedings of the IEEE*, vol. 93, num. 1, pp. 84-97, enero 2005. (c3,p67)

[YB04] R. YANG Y M.S. BROWN, "Decoder motion vector estimation for scalable video error concealment", *Proceedings IEEE International Conference on Multimedia and Expo (ICME)*, vol. 3, pp. 1739-1742, junio 2004. (c3,p67)

[YL03] H.S. YOON Y G.S. LEE, "Motion estimation based on spatio-temporal correlations", *Proceedings International Conference on Image Processing (ICIP)*, vol. 2, pp. 359-362, septiembre 2003. (c2,p50) (c2,p51)

[YM03] S.Y. YAP Y J.V. MCCANNY, "A VLSI architecture for advanced video coding motion estimation", *Proceeedings of the Application-Specific Systems, Architectures and Processors*, pp. 293-301, junio 2003. (c2,p55) (c4,p146) (c4,p204) (c4,p209)

[YN04] B. YAN Y K.W. NG, "A novel motion vector recovery algorithm for error concealment in video transmission", *Proceedings IEEE Consumer Communications and Networking Conference*, pp. 621-623, enero 2004. (c3,p67)

[YSW89] K.M. YANG, M.T. SUN Y L. WU, "A family of VLSI designs for motion compensation block-matching algorithm", *IEEE Transactions on Circuits and Systems*, vol. 36, num. 10, pp. 1317-1325, octubre 1989. (c4,p204)

[YVL+02] P. YIN, A. VETRO, B. LIU Y H. SUN, "Drift compensation for reduced spatial resolution transcoding", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, num. 11, pp. 1009-1020, noviembre 2002. (c3,p68)

[ZG05a] L. ZHANG Y W. GAO, "Improved FFSBM algorithm and its VLSI architecture for variable block size motion estimation of H.264", *Proceedings International Symposium on Intelligent Signal Processing and Communication Systems*, pp. 445-448, diciembre 2005. (c2,p50) (c2,p55) (c4,p202) (c4,p203) (c4,p209)

[ZG05b] L. ZHANG Y W. GAO, "Hybrid algorithm with adaptive complexity for integer pel motion estimation of H.264", *Proceedings IEEE International Conference on Multimedia and Expo (ICME)*, pp. 101-104, julio 2005. (c2,p51)

[ZHY+03] J. ZHANG, Y. HE, S. YANG Y Y. ZHONG, "Performance and complexity joint optimization for H.264 video coding", *Proceedings International Symposium on Circuits and Systems (ISCAS)*, vol. 2, pp. 888-891, mayo 2003. (c1,p7) (c2,p35)

[ZQ05] C. ZHU Y W.S. QI, "Predictive fine granularity successive elimination for fast optimal block-matching motion estimation", *IEEE Transactions on Image Processing*, vol. 14, num. 2, pp. 213-221, febrero 2005. (c2,p56)

ANEXO

Secuencias de vídeo utilizadas

En este anexo se resumen las principales características espaciales y temporales de las secuencias de vídeo utilizadas en esta Tesis Doctoral.

A.1 Características de las secuencias de vídeo utilizadas

En esta Tesis se han utilizado numerosas secuencias de vídeo en formato QCIF (*Quarter Common Intermediate Format*, 176×144 píxeles) para evaluar las prestaciones de los algoritmos y arquitecturas propuestas. A lo largo de la Tesis, se muestran alternativamente resultados obtenidos con las secuencias cuyas características se resumen en la Tabla A.1.

Nombre de la secuencia	Características espaciales y temporales de la secuencia
TABLE (300 fotogramas)	▪ Secuencia con mezcla de texturas homogéneas y heterogéneas ▪ Gran cantidad de movimiento en objetos de diferentes tamaños ▪ Existencia de cambios de contexto y diferentes enfoques (*zoom*) ▪ No existe movimiento de cámara
DEADLINE (1375 fotogramas)	▪ Secuencia con gran cantidad de texturas heterogéneas ▪ Poco movimiento limitado al objeto central de la secuencia ▪ No existen cambios ni en el contexto ni en el enfoque ▪ No existe movimiento de cámara
FOREMAN (400 fotogramas)	▪ Secuencia con gran cantidad de texturas heterogéneas ▪ Gran cantidad de movimiento, con muy diferentes sentidos y velocidades ▪ Existencia de cambios de contexto sin diferentes enfoques ▪ Existe movimiento de cámara
MISS AMERICA (150 fotogramas)	▪ Secuencia con gran cantidad de texturas homogéneas ▪ Poco movimiento limitado al objeto central de la secuencia ▪ No existen cambios ni en el contexto ni en el enfoque ▪ No existe movimiento de cámara
PAMPHLET (300 fotogramas)	▪ Secuencia con mezcla de texturas homogéneas y heterogéneas ▪ Gran cantidad de movimiento, con muy diferentes sentidos y velocidades ▪ Existencia de cambios de contexto sin diferentes enfoques ▪ No existe movimiento de cámara
SUZIE (40 fotogramas)	▪ Secuencia con mezcla de texturas homogéneas y heterogéneas ▪ Gran cantidad de movimiento, con muy diferentes sentidos y velocidades ▪ No existen cambios ni en el contexto ni en el enfoque ▪ No existe movimiento de cámara
COASTGUARD (100 fotogramas)	▪ Secuencia con gran cantidad de texturas homogéneas ▪ Poca cantidad de movimiento, exclusivamente en sentido horizontal ▪ No existen cambios ni en el contexto ni en el enfoque ▪ Existe movimiento de cámara

Tabla A.1: Características espaciales y temporales de las secuencias de vídeo utilizadas.

Asimismo, en la Figura A.1 se puede observar un fotograma de muestra correspondiente a cada una de las secuencias descritas.

Figura A.1: Fotograma de muestra de las secuencias TABLE (A), DEADLINE (B), FOREMAN (C), MISS AMERICA (D), PAMPHLET (E), SUZIE (F) y COASTGUARD (G).