Capítulo 2

Conceptos básicos

2.1 INTRODUCCIÓN

El objetivo de este capítulo es conocer en qué consiste la segmentación temporal de secuencias de vídeo: su necesidad en los contextos de la indexación de vídeo y del análisis del contenido, definiciones básicas de los conceptos que hemos utilizado a lo largo del proyecto, etc.

Posteriormente, mostraremos el *estado del arte* en estas técnicas, echando un vistazo a los soluciones propuestas por los investigadores a lo largo de estos últimos años. Se incluirán la referencias a los artículos donde aparecen, y los fundamentos de cada uno de los algoritmos.

Otro punto que también trataremos en este capítulo es el del diezmado de las secuencias. Como se verá, la mayoría de los trabajos realizados últimamente se centran en las secuencias comprimidas. En nuestro caso, hemos utilizado un formato propio y veremos el por qué del diezmado.

Finalmente, hemos incluido un resumen de *nuestro método*. Se trata de un conjunto de diagramas de bloques que nos darán una visión global de lo que se desarrollará en los capítulos posteriores.

2.2 NECESIDAD DE LA SEGMENTACIÓN TEMPORAL

Actualmente existen grandes cantidades de información de vídeo en distintos formatos (cintas analógicas de diferentes sistemas, CD-ROM's con imágenes comprimidas, bases de datos de programas de T.V., filmotecas, etc). Esta diversidad de formatos y fuentes tiende a desaparecer con el establecimiento de los nuevos estándares de codificación de imagen y vídeo. El objetivo de estos estándares digitales es doble: uniformizar los

sistemas y optimizar la cantidad de memoria necesaria para su almacenamiento. En el futuro, con toda esta información y con la que se generará, serán necesarios sistemas que pueden proveerla a grandes cantidades de *clientes*, que además demandarán servicios avanzados y personalizados.

Para soportar estas nuevas aplicaciones, será indispensable que los interfaces con el usuario, además de intuitivos y fáciles de manejar, sean potentes. Pero lo más importante, para poder dar respuesta a la peticiones de información, será la *preparación* previa de ésta. Las bases de datos deberán disponer de mecanismos eficientes de indexado de la información, que no serán más que sistemas para la descripción automática (o semiautomática, minimizando el trabajo manual de visionado) del contenido de la información audiovisual.

Pensando en secuencias de vídeo, esta claro que no es factible disponer de información sobre cada uno de sus fotogramas. Parace más útil poder dividir las secuencias en fragmentos, dentro de cada uno de los cuales exista conexión semántica. Básicamente, este ha sido el objetivo de nuestro proyecto: segmentar temporalmente las secuencias, para un posterior postprocesado de esta información.

2.2.1 Definición de plano

Acabamos de mencionar que las secuencias de vídeo se segmentan en trozos o fragmentos, cuyo contenido está relacionado. Habitualmente estos fragmentos representarán un evento o una secuencia de acciones contínuas, y esto es conocido como "plano" temporal [1]. Un plano es aquello capturado por una cámara entre las operaciones de *grabar* y *parar*.

Las fronteras entre planos de vídeo son conocidos comúnmente como cambios de escena y el acto de segmentar una secuencia de video en sus planos se llama detección de cambios de escena.

Hasta 1901, las películas de cine eran monoplanos. De esta forma su género era esclusivamente documental, no se podía contar una historía que tuviera más de un escenario o se desarrollara a lo largo del tiempo. En este año James Williamson en su película "*Fire!*", y con el fin de mostrar una casa en llamas cuando se inicia el fuego en el interior y como se consume vista desde fuera, puso su cámara en el interior y después la saco al exterior. El resultado así eran dos planos que se yuxtaponían. El *cut* o corte estaba inventado. Más adelante se atribuye la invención del montaje (y por tanto de la estructuración de la narración audiovisual) a Levin Fizhamon, que realizo en 1904 "*Rescued by Rover*", y que presentaba una historia estructurada en cinco planos. El corte funcionaba aquí como una elipsis espacio-temporal que permitía al realizador estructurar el argumento de la historia. Rápidamente, está sintaxis se enriqueció de nuevos elementos de estructuración. Se atribuye, por ejemplo, al gran cineasta David

[1] En la bibliografía en inglés se denomina *shot*.

Wark Griffith la invención del efecto de transición progresiva, algunos años más tarde. Esta riqueza de potencialidades ofrecidas por los efectos de transición es sin duda el origen de la siguiente cita de Orson Welles: *"La sola puesta en escena cobra una gran importancia a lo largo del montaje... Las imágenes en sí mismas no son suficientes: son muy importantes, pero no sólo las imágenes. Lo esencial es la duración de cada imagen, lo que es cada imagen; en esto consiste la elocuencia del cine y que se fabrica en la sala de montaje"* [14].

Hoy en día, todavía se mantienen estos dos caminos para que un plano pueda realizar una transición al siguiente: transición abrupta o corte y transición gradual. En las transiciones graduales, y con la aparición de los equipos digitales de edición, se han producido grandes avances y mejoras. Pero en cualquier caso, todavía se pueden clasificar en dos grandes grupos: fundidos y efectos de incrustaciones (por ejemplo, las cortinillas) [2].

(a) (b)

Figura 2.1: Ejemplo de transición abrupta: (a)Ultimo fotograma de un plano. (b)Primer fotograma del plano siguiente.

En algunos casos también puede ser interesante encontrar las variaciones dentro del plano. Esto es conocido como detección de cambios de escena intraplano o *intra-shot*. De esta manera se podrían encontrar efectos tales como: *zooms*, panorámicas (*panning*), etc. (para una mejor clasificación, consultar apéndice A). Las técnicas utilizadas para conseguir esta segmentación más crítica, se fundamentan en los vectores de movimiento y no las hemos estudiado en este proyecto.

[2] A lo largo del proyecto utilizaremos de manera indistinta los términos en castellano o en inglés para referirnos a: cortes o *cuts*, fundidos o *dissolves* y cortinillas o *wipes*. En caso de duda, cada uno de estos términos se define en el apéndice A.

(a) (b)

(c) (d)

Figura 2.2: Ejemplo de transición gradual (fundido con dos fotogramas intermedios): (a)Ultimo fotograma de un plano. (b)Fotograma intermedio del fundido. (c)Fotograma intermedio del fundido. (d)Primer fotograma del plano siguiente.

2.2.2 Indexación de vídeo digital

La indexación de vídeo no es algo totalmente nuevo. Desde hace muchos años la mayoría de las televisiones disponen de personal encargado del indexado manual de sus programas. En general, consiste simplemente en información sobre fechas de emisión, titulos de crédito y breve descripción del contenido. Pero para algunos casos, como los informativos, la descripción es muchísimo más detallada. La persona encargada del trabajo visiona el vídeo y describe todo lo que aparece en él, plano a plano. A pesar de la profesionalidad y destreza, resulta evidente que se pueden producir errores de dos tipos: el primero, considerado perceptual o subjetivo, es debido a la persona concreta que realiza el trabajo (una persona diferente, utilizaría otras palabras *clave*),

y el segundo tipo, serían errores introducidos por la variación del nivel de atención, cambio en las condiciones de trabajo, fatiga, motivación, etc.

Si bien esta clara la dificultad de conseguir una descripción automática, sobre todo de tipo textual, actualmente se trabaja en lograr aproximaciones lo más valiosas posibles. En breve, aparecerá el **MPEG-7** [8]. Se trata del último estándar para la codificación de imágenes y vídeo, pero que cambía totalmente la filosofía respecto a los anteriores estándares de la misma serie. Su objetivo es la descripción de la información multimedia, para su posterior codificación, todo ello enfocado a las futuras aplicaciones que hemos descrito ya. Dentro del complejo proceso a realizar, la indexación de vídeo es fundamental.

Un posible esquema para entender cómo funciona la indexación, podría ser el de la figura 2.3. Se puede observar que el primer paso es la **segmentación en *shots***.

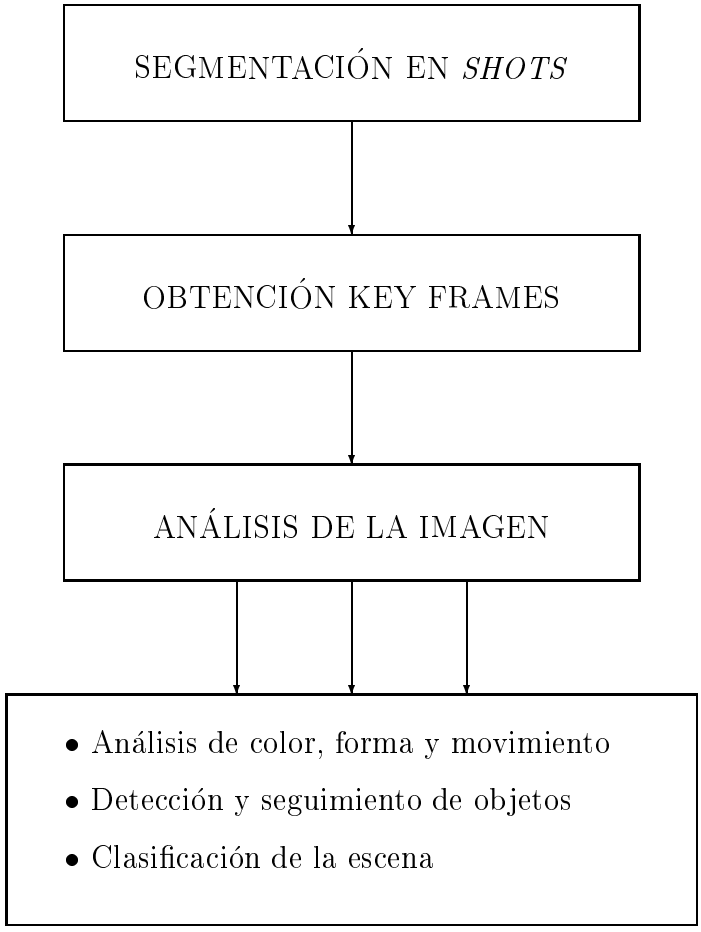

Figura 2.3: Esquema con los pasos llevados a cabo en la indexación de vídeo.

Como resultado de esta primera etapa se obtendrá como información básica el primer

y ultimo fotograma de cada uno de los planos. También es habitual disponer de otra información sobre el plano: tipo (variación lineal, fundido, cortinilla, etc), nivel de variación (es decir, si se producen muchas variaciones intraplano, debidas por ejemplo a movimientos bruscos de la cámara), etc.

El siguiente paso consiste en la **extracción de** uno o más fotogramas que caracterizan a todo el plano. Son los llamados ***keyframes*** [3]. Sobre las distintas alternativas para seleccionar estos fotogramas claves, profundizaremos en el capítulo 5.

A partir de los fotogramas claves se puede realizar un **análisis** más profundo **de la imagen**. Se ha pasado de disponer de una subsecuencia de fotogramas, que constituyen el plano detectado en la fase anterior, a tener solo uno más fotogramas (siempre será un número reducido) sobre los que aplicar todas las técnicas conocidas de análisis de imágenes. Se podrían realizar estudios sencillos: análisis de color, forma, movimiento, o más complejos: detección y seguimiento de objetos, detección de personas y "caras", etc. En cualquiera de los casos, se puede considerar que esta última fase de la indexación, es la étapa del análisis semántico o de contenido, propiamente dicho. Actualmente existen muchísimos investigadores dedicados a mejorar los algoritmos utilizados en esta fase, sin duda, la más compleja.

El objetivo inicial de nuestro proyecto era ocuparnos solo de la primera etapa. Posteriormente, nos pareció interesante realizar algún postprocesado de la información obtenida de la segmentación. Así pues, también hemos desarrollado algunas técnicas de extracción de fotogramas claves y de indexado semiautomático.

2.2.3 Restauración de películas antiguas

Ya hemos comentado anteriormente que existen multitud de fuentes de información de vídeo que tarde o temprano deberán pasarse a formato digital. Históricamente, las más antiguas son las primeras películas de cine (disponibles en formatos analógicos). La mayoría de este material se encuentra deteriorado por el paso del tiempo, malas condiciones de conservación, etc. Por tanto, una vez realizado el proceso de digitalización, podría pensarse en utilizar técnicas de restauración que mejoraran la calidad de la imagen [33] [32].

Los daños que pueden haber sufrido estas viejas películas pueden ser muy variados. Además, a las películas rodadas a partir de los años cincuenta (con la aparación del color) hay que añadirles el inconveniente de haber sufrido también decoloraciones. Los daños más habituales que podemos encontrar son: roturas, rayas, variación del brillo de unos fotogramas a otros, descentrado de las imágenes, decoloración, etc. A la hora de restaurar una película actualmente existen métodos manuales (fotograma a fotograma) o semiautomáticos, pero su principal inconveniente son sus precios desorbitados.

[3] En la bibliografía aparecen indistitamente las dos palabras juntas, *keyframes*, por tratarse de un término plenamente aceptado, o separadas *key frames*.

Evidentemente, la implementación de un método automático sería menos caro y podría ser utilizado para restaurar películas de distinto tipo [12].

Las diversas técnicas digitales utilizadas (filtrado morfológico temporal, compensación de brillo, compensación de movimiento, etc) se fundamentan en localizar "diferencias" entre fotogramas consecutivos. Surge de esta forma la necesidad de segmentar una película completa, en sus planos temporales, y luego aplicar los algoritmos de segmentación, tal y como aparece en la figura 2.4.

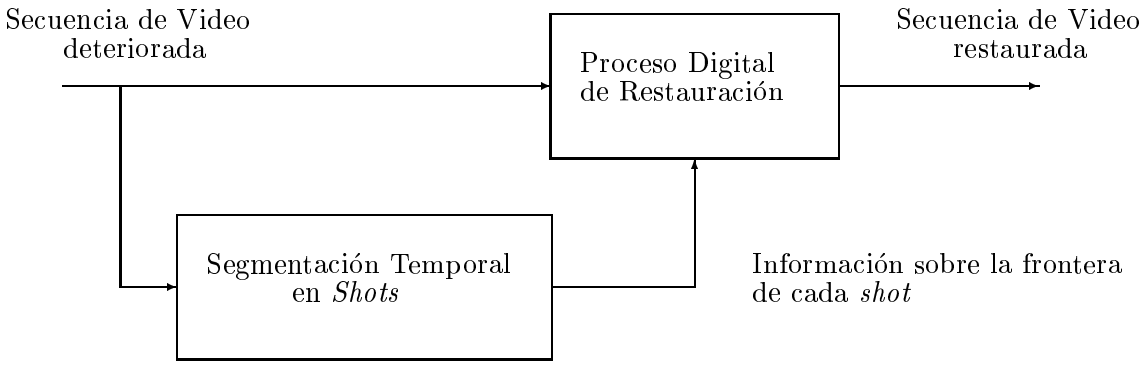

Figura 2.4: Utilidad de la segmentación temporal en el proceso de restauración.

Uno de los profesores miembros del Grupo de Procesado de Imágenes y Vídeo trabaja en su tesis para desarrollar nuevos algoritmos de restauración, por lo que hemos intentado que parte de nuestro trabajo pudiera ser utilizado tal y como se ha descrito.

2.3 APROXIMACIONES AL PROBLEMA

La segmentación temporal o detección de cambios de escena en secuencias de vídeo es un tema sobre el que se ha trabajado mucho. Las primeras publicaciones sobre estás técnicas son de hace más de diez años, y desde entonces no han parado de introducirse novedades y mejoras. En este apartado, queremos hacer un breve resumen de las técnicas más conocidas (incluyendo las referencias a los artículos donde han sido publicadas). No se trata de ningún estudio exhaustivo, sirva simplemente como "golpe de vista". Existen dos artículos de Boreczky *et al.* [5] y de Joly [14], que sí realizan un estudio comparativo serio. En ambos casos, se trata de comparar la bondad de los métodos utilizando un modelo experimental. Otro artículo que también da una buena visión sobre la indexación de video en general, y la segmentación en particular es el de Ahanger y Little [2].

Las primeras técnicas estaban enfocadas a la detección de cortes, pero más recientemente el trabajo se ha orientado a la detección de transiciones graduales. La mayor

parte de ellas, utilizan para la detección de fronteras entre planos: diferencias entre pixels, diferencias estadísticas, comparación de histogramas, diferencias en los bordes, diferencias de compresión, y vectores de movimiento.

Para comparar todos estos métodos, se deberían tener en cuenta los siguientes elementos:

- Fiabilidad en la detección de los efectos de transición. En la bibliografía científica se han propuesto incluso ratios que la miden [4].

- La capacidad de los algoritmos de tratar diferentes tipos de transiciones. Como ya hemos comentado, se distinguen principalmente tres tipos de efectos: cortes, fundidos y efectos de incrustación (ej. cortinillas). A estos dos últimos, los llamaremos también efectos progresivos.

- La posibilidad de aplicar los tratamientos sobre secuencias comprimidas o particularmente descomprimidas (sobre esto se profundizará en el siguiente apartado).

- La cantidad de cálculos necesarios para efectuar el proceso, y por tanto, el tiempo de procesado.

A continuación, realizaremos el análisis por autores [5].

Nagasaka y Tanaka [22] experimentan con varias técnicas de comparación, incluyendo la diferencia de la suma de niveles de gris, suma de las diferencias de los niveles de gris, diferencia de los histogramas de nivel de gris, comparación de paletas de color, diferencia de los histogramas de color y comparación χ^2 de histogramas de color. Cada una de las comparaciones son aplicadas sólo a una porción de la imagen. Los autores concluyen que el método más robusto es el de las comparaciones χ^2 de histogramas de color. Para protegerse frente a "ruidos" momentaneos, como flashes de cámara, dividen la imagen completa de cada fotograma en 4×4 regiones rectangulares de igual tamaño y comparan cada región par a par. Las grandes diferencias son desechadas, y la detección es dada por aquellas que permanecen. Su método es robusto frente a efectos como *zoom* o *panning*, pero falla al intentar detectar efectos especiales, del tipo fundido.

Otsuji *et al.* [15] usan como información el nivel de brillo, para calcular las diferencias basadas en el histograma, y las diferencias basadas en los *pixels*. Las transiciones contínuas no son consideradas. En una publicación posterior, estos mismos autores proponen un filtro "adaptado" que mejora la detección de los cortes.

[4] Para verificar la bondad de nuestros métodos, hemos utilizado uno de ellos en las conclusiones finales de este proyecto.

[5] Hemos elegido hacerlo según este criterio, frente a otra posible clasificación, ya que la mayoría se centran en una técnica en concreto, intentando mejorarla.

Hsu y Harashima formulan los cambios de escena y actividad como discontinuidades de movimiento [13]. La caracterización de actividad se desarrolla considerando el signo de una Gaussiana y midiendo la curvatura de superficies espacio-temporales.

Hampapur *et al.* [10] y, Aigrain y Joly [3] usan métodos basados en modelos que capturan diferentes tipos de transiciones entre planos. La resolución de la imagen se reduce para aumentar la velocidad de computación y para protegerse frente a excesivos movimientos de cámara u objetos.

Shahraray [30] detecta cambios abruptos y graduales basandose en un filtrado temporal del movimiento entre fotogramas consecutivos. Un proceso bloque a bloque, se desarrolla en cada bloque de la primera imagen, para encontrar la región que mejor se ajusta de la segunda imagen. Un filtrado estadístico no lineal es utilizado para generar los valores de ajuste.

Zhang *et al.* [7] también hacen uso de la diferencia de los histogramas de color y de umbrales globales para detectar cambios de escena. Cómo en el esquema de Nagasaka y Tanaka, se usa un umbral global para tomar la decisión. Para localizar los fundidos, proponen una técnica de doble umbral, utilizando un umbral "pequeño" para detectar el comienzo de la transición gradual. Se marca este fotograma de comienzo y se comparan los sucesivos fotogramas para comprobar si se supera el segundo umbral (más grande), en este punto, el algoritmo decidirá si hay transición o no, siempre que todas las diferencias fotograma a fotograma de los intermedios hayan superado el primer umbral. También se considera el uso del análisis de flujo óptico para detectar los movimientos de cámara. Consiguen un alto porcentaje de detecciones. Las mayores causas de fallos vienen dadas por la similitud entre los histogramas de color de diferentes planos, cuando el contenido de color es muy similar o por cambios bruscos de luminancia debidos a flashes.

Con la aparición de MPEG, muchos investigadores se han centrado en la detección de los cambios de escena sobre video comprimido. Meng *et al.* [19] usan la varianza de los coeficientes *dc* (todo esto se desarrollará en le siguiente apartado) en fotogramas I y P y la información de los vectores de movimiento para caracterizar los cambios de escena. Sethi y Patel [29] y posteriormente Sethi *et al.* [31], han usado solo los coeficientes *dc* de los fotogramas I para desarrollar un test de hipótesis usando el histograma de luminancia. Se asume que la separación entre dos *frames* I es fija y pequeña. De todas formas, la localización exacta de los cambios abruptos no puede ser realizada con este método.

Yeo y Liu [35] proponen algoritmos rápidos para operar directamente sobre *Motion* JPEG o MPEG comprimidos. Sus algoritmos operan directamente sobre una pequeña fracción de los datos comprimidos (se realiza una descompresión parcial). Su método es efectivo para detectar cortes, transiciones graduales y escenas con *flashlight*.

Los últimos métodos propuestos trabajan casi todos sobre técnicas estadísticas en vídeo comprimido, como el de Bouthemy*et al.* [6]. Estos autores utilizan estimación de movimiento dominante para localizar los cortes y caracterizar los movimientos de

cámara. Y los más recientes de Hanjalic*et al.* [11] y de Vanconcelos y Lippman [34] que establecen modelos estadísticos bayesianos de repetición de transiciones.

2.4 VÍDEO COMPRIMIDO, IMÁGENES "ENTERAS" E IMÁGENES DIEZMADAS

En esta sección vamos a introducir algunos conceptos básicos sobre vídeo comprimido en MPEG que, aunque nosotros en el PDIWIN32 utilizamos un formato propio para las secuencias (ver apéndice B), nos han servido como fundamento de nuestro métodos, y que además permitirán una fácil extrapolación de nuestro trabajo para usarse sobre MPEG.

La compresión de imágenes se plantea en el momento en que se pretende codificar una imagen como una señal digital. El problema reside en la cantidad de bits que se necesitan para realizar dicha codificación. La codificación de vídeo MPEG consigue altas tasas de compresión de la señal de vídeo eliminando no sólo redundancias espaciales sino también temporales. Para conseguirlo se basa en dos filosofías fundamentales: la codificación para una imagen fija, es decir, eliminar redundancias dentro de una misma imagen y la determinación de vectores de movimiento que es base para eliminar redundancias entre imágenes sucesivas.

Por el tipo de codificación que usa MPEG (codificación entrópica), las imágenes se tratan y codifican de distinta forma. Los distintos tipos de imágenes usados son:

- Imágenes intraframe, **I**: Se codifican como si fuesen imágenes fijas utilizando una norma similara a JPEG, por tanto, para decodificar una imagen de este tipo no hacen falta otras imágenes de la secuencia, sino sólo ella misma.

- Imágenes interframe, **P**: Es la predicción de una imagen con compensación de movimiento a partir de la I o P anterior. Para decodificar una imagen de este tipo se necesita, además de ella misma, la I o P anterior.

- Imágenes interframe bidireccionales, **B**: Son imágenes que se codifican utilizando la I o P anterior y la I o P siguiente.

- Imágenes intraframe de baja resolución, **I**: Son imágenes de las mismas características que las I pero con menos resolución. Se usan en aplicaciones que no necesitan gran calidad, como en el avance rápido.

Lo interesante para un preprocesado de la información, como es la segmentación temporal, sería no tener que decodificar totalmente la secuencia comprimida en MPEG. A continuación describiremos el método que utiliza las llamadas imágenes *DC* y secuencias *DC*, cómo se pueden extraer de forma eficiente de vídeos comprimidos, y por qué es útil para rapidez y eficiencia en operaciones de análisis de escenas.

2.4.1 Imagen *DC* y secuencia *DC*

[6] Las imágenes *DC* son versiones reducidas espacialmente de las imágenes originales. Cada una de estas imágenes reducidas, una vez extraídas, pueden también ser usadas para otras aplicaciones, como por ejemplo la detección de cambios de escena. Algunos autores las han utilizado para una detección eficiente de los planos [35], para generación automática de una representación compacta de la estructura y contenido de documentos de vídeo [16], y para aplicaciones no lineales de vídeo *browsing* [7].

Una imagen se divide en bloques de $N \times N$ pixels. El valor del pixel (i, j) de la *dc-image* es la media de los valores de los pixels del bloque (i, j) de la imagen original. La secuencia formada, para cada fotograma, de esta manera se llama *dc-sequence*.

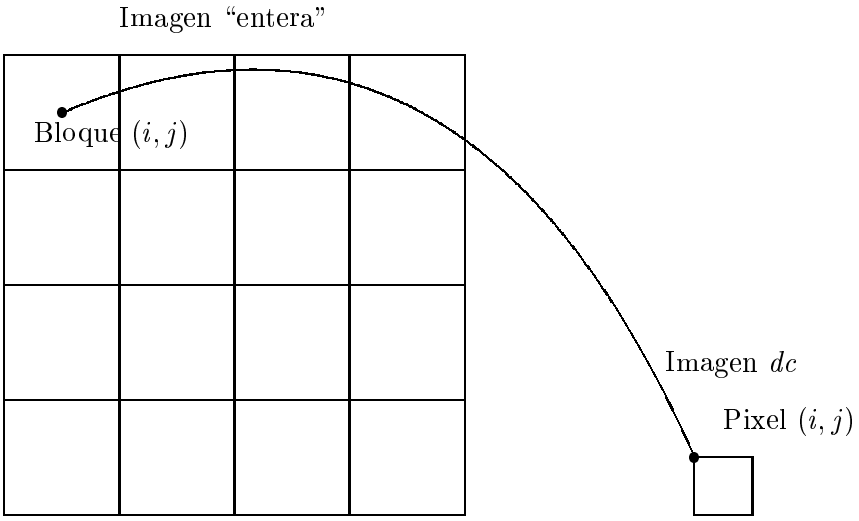

Figura 2.5: Imagen "entera" dividida en bloques frente a la imagen *dc*.

En la figura 2.6 se puede observar una imagen original de tamaño 512×512 y la correspondiente *dc-image* de tamaño 64×64, usando $N = 8$.

Las ventajas del uso de las imágenes *dc* y de las secuencias *dc* para el análisis de escenas es triple:

- Las operaciones se pueden desarrollar directamente sobre los datos comprimidos (más concretamente, realizando solo una decodificación parcial), eliminando la necesidad de una descompresión del fotograma completo.

- Se trabaja con una pequeña fracción de los datos originales. Esto reducirá la carga computacional de los algoritmos y por consiguiente el tiempo de procesado.

[6] O también *DC-image* y *DC-sequence* respectivamente.

(a)

(b)

Figura 2.6: Imagen de *Lena*: (a)Imagen "entera" de 512×512 pixels. (b)Imagen *DC* de tamaño 64×64 pixels.

- El proceso de "promediado" o filtrado paso-bajo de la imagen introduce otra mejora. Por una parte eliminará gran número de falsas alarmas en la detección de *cuts*, al eliminar parte del movimiento de la cámara a través de la imagen. Y además, también minimizará el efecto del movimento de los objetos en la imagen (en el caso de objetos menores de 8×8 pixels, lo eliminará totalmente).

2.4.2 Relación con la Transformada Discreta del Coseno, DCT

Para JPEG y *frames I* de MPEG, la imagen original se agrupa en bloques de 8×8 pixels, y el término *dc* o término de la "componente contínua", $c(0,0)$ de la Transformada Discreta del Coseno 2-D esta relacionado con los valores de los pixels, $f(i,j)$ a través de 2.1:

$$c(0,0) = \frac{1}{8} \sum_{x=0}^{7} \sum_{y=0}^{7} f(x,y) \tag{2.1}$$

Mientras que la imagen *dc* es mucho más pequeña que la imagen original, todavía contiene suficiente información. Esto sugiere que las variaciones de escena de naturaleza global (es decir, descartando el movimiento de objetos) que se producen en la imagen original, se produzcan también en la imagen *dc*.

Para formar la *dc-image* de una imagen original no comprimida (como hacemos nosotros en nuestro proyecto) se requieren $O(N^2)$ operaciones por bloque, mientras que en el caso de imágenes con compresión basada en la DCT, el término *dc* es una

versión escalada del valor de la media de un bloque. Cuando $N = 8$, de 2.1, el valor de la media es $\frac{1}{8}$ del término *dc* de la DCT de un bloque.

2.4.3 Secuencia *DC* de una secuencia codificada con MPEG

La extracción de la *dc-image* en las *I-frames* de MPEG es trivial. A continuación veremos cómo se extraen las imágenes *dc* correspondientes a las *P-frames* y *B-frames* directamente de la secuencia de vídeo comprimido. Para más detalles mirar en [35].

Para obtener los coeficientes *dc* de las imágenes P, usando los coeficientes DCT de la imagen I previa, utilizaremos la siguiente nomenclatura: P_{ref} denotará el bloque de interés de la imagen, P, $P_1, \ldots, P_4$ son los cuatro bloques vecinos originales de la imagen I previa, de ellos proviene P_{ref}, ver figura 2.7.

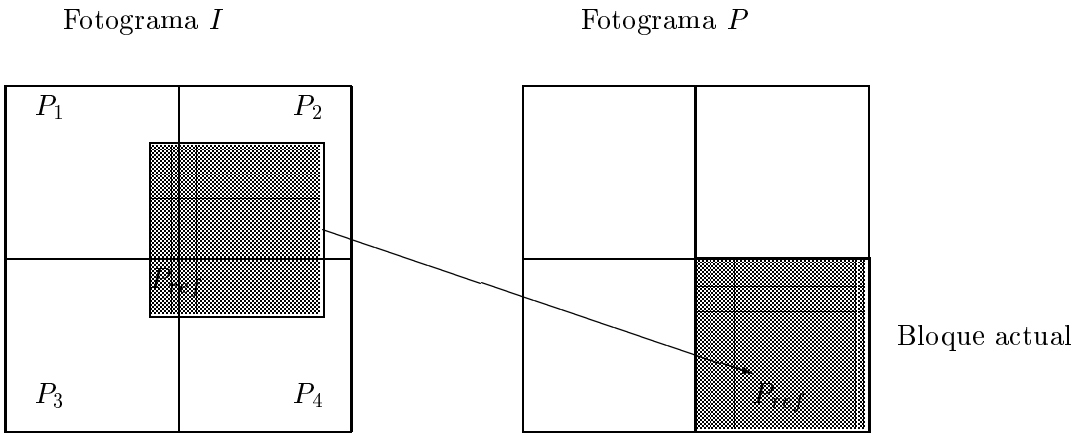

Figura 2.7: Bloque de referencia P_{ref}, vectores de movimiento y bloques originales, del anterior fotograma I, y Bloque actual del fotograma P.

El vector de movimiento es $(\Delta x, \Delta y)$, y representa lo que se han movido $P_1, \ldots, P_4$ desde la imagen I hasta la imagen P, y que ha tenido como consecuencia el desplazamiento de P_{ref} a su posición actual. Dada la linealidad de la DCT, el coeficiente *dc* de P_{ref} es de la forma

$$dc(P_{ref}) = \sum_{i=1}^{4} \{ \sum_{m=0}^{7} \sum_{l=0}^{7} \varpi_{ml}^{i} [DCT(P_i)]_{ml} \} \tag{2.2}$$

donde la componente (i, j) de P se denota por $[P]_{ij}$. El factor o peso ϖ_{ml}^{i} pondera la contribución de $[DCT(P_i)]_{ml}$. Para calcular 3.3, se necesitan como máximo 256 multiplicaciones. Siguiendo la aproximación que aparece en [35], se puede decir que

$\varpi^i_{ml} = [DCT(S_{i1})]_{0m} \times [DCT(S_{i2})]_{l0}$. S_{i1} es una matriz de la forma

$$\begin{pmatrix} 0 & 0 \\ I_{h_i} & 0 \end{pmatrix} \text{o} \begin{pmatrix} 0 & I_{h_i} \\ 0 & 0 \end{pmatrix}$$

y S_{i2} es una matriz de la forma

$$\begin{pmatrix} 0 & 0 \\ I_{\varpi_i} & 0 \end{pmatrix} \text{o} \begin{pmatrix} 0 & I_{\varpi_i} \\ 0 & 0 \end{pmatrix}$$

donde I_n es una matriz identidad de tamaño n. En estas ecuaciones, h_i y ϖ_i son el peso y la anchura que se superpone P_{ref} con P_i.

Hay cuatro posibles localizaciones del subbloques de interés: arriba-izquierda, arriba-derecha, abajo-derecha, y abajo-izquierda. Las matrices que corresponden a cada caso vienen dadas por la tabla 2.1 . Operando, se puede obtener que el peso ϖ^i_{00} de

Subbloque	**Posición**	S_{i1}	S_{i2}
P_1	inferior derecha	$\begin{pmatrix} 0 & I_{h_1} \\ 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 \\ I_{\varpi_1} & 0 \end{pmatrix}$
P_2	inferior izquierda	$\begin{pmatrix} 0 & I_{h_2} \\ 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & I_{\varpi_2} \\ 0 & 0 \end{pmatrix}$
P_3	superior derecha	$\begin{pmatrix} 0 & 0 \\ I_{h_3} & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 \\ I_{\varpi_3} & 0 \end{pmatrix}$
P_4	superior izquierda	$\begin{pmatrix} 0 & 0 \\ I_{h_4} & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & I_{\varpi_4} \\ 0 & 0 \end{pmatrix}$

Tabla 2.1: Matrices S_{i1} y S_{i2}.

$[DCT(P_i)]_{00}$ es $(h_i \times \varpi_i)/64$. De tal forma, que la ecuación 3.3 puede escribirse como

$$[DCT(P_{ref})]_{00} = \sum_{i=1}^{4} \frac{h_i \varpi_i}{64} [DCT(P_i)]_{00} + c \tag{2.3}$$

donde

$$c = \sum_{i=1}^{4} (\sum_{(m,l)\neq(0,0)} \varpi^i_{ml} [DCT(P_i)]_{ml}).$$

El primer término puede ser visto como una aproximación de $[DCT(P_{ref})]_{00}$ dada por la suma ponderada de los coeficientes dc de $P_1, \ldots, P_4$, donde el peso $(h_i \times \varpi_i)/64$ es simplemente la fracción de área ocupada por el subbloque. El último término de 2.3, c, sirve como corrección de esta aproximación. En la práctica, se comprueba que c es muy pequeño.

Según los experimentos realizados por los investigadores que utilizan estas aproximaciones, las imágenes *dc* obtenidas para los *frames* P y B sirven para los propósitos del análisis de escenas. Además, el coste por coeficiente *dc* en imágens P y B se reduce a tan solo cuatro multiplicaciones. Las aproximaciones de primer orden descritas, reciben el nombre de aproximación *P-first-B-first* y la reconstrucción exacta de *frames* P y B se llama *P-exact-B-exact* [35].

2.4.4 Funciones para la obtención de la secuencia *DC* con el PDIWin32

Ya hemos comentado varias veces que en el PDIWIN32 trabajamos con un formato propio de secuencias de vídeo digitales. Para lo que nos interesa en este apartado, decir simplemente que se trata de un formato no comprimido, donde los fotogramas se almacenan como imágenes completas.

Cuando queramos procesar una secuencia para segmentarla temporalmente, el primer paso que realizaremos siempre será el de obtener la correspondiente secuencia *dc* o, como terminología propia, secuencia diezmada. Tal y como hemos descrito en el apartado anterior, lo que habrá que hacer es dividir la imagen de cada fotograma en bloques de 8×8 pixels, y el valor del pixel (i, j) de la *dc-image* será la media de los valores de los pixels del bloque (i, j) de la imagen original. Esta operación lineal puede ser vista como dos operaciones lineales en cascada: filtrado paso-bajo y diezmado por 8 de la imagen.

Al inicio de este proyecto, cuando nos enfrentamos con la necesidad de añadir una función al programa que realizara este proceso, vimos que ya existía una similar que filtraba paso-bajo y diezmada por 2. Debido a las operaciones en que consiste el diezmado, resulta evidente que el diezmado por 8 se consigue realizando tres diezmados sucesivos por 2, tal y como se observa en la figura 2.8. Para que lo anterior funcione sin problemas, únicamente ha de tenerse cuidado a la hora de reservar memoria para la imagen destino. Si vamos a diezmar una imagen de tamaño $m \times n$ por un factor N, la imagen resultado tendrá un tamaño $m/N \times n/N$, siempre y cuando m y n sean divisibles por N, en caso contrario ha de elegirse el número entero correspondiente. Así pues, hemos programado una función:

```
P8TamagnoDiezmar(int *anch,int *alt,int *anch2,int *alt2,int factor),
```

que nos devuelve el tamaño (`anch2` y `alt2`) que tendrá la imagen diezmada por `factor`, sí la imagen original tenía un tamaño `anch` y `alt`.

El siguiente paso ha consistido en programar una función que filtrara-diezmara por 8 una imagen (para ello como ya se ha comentado, se realizan tres llamadas a la que filtra-diezma por 2):

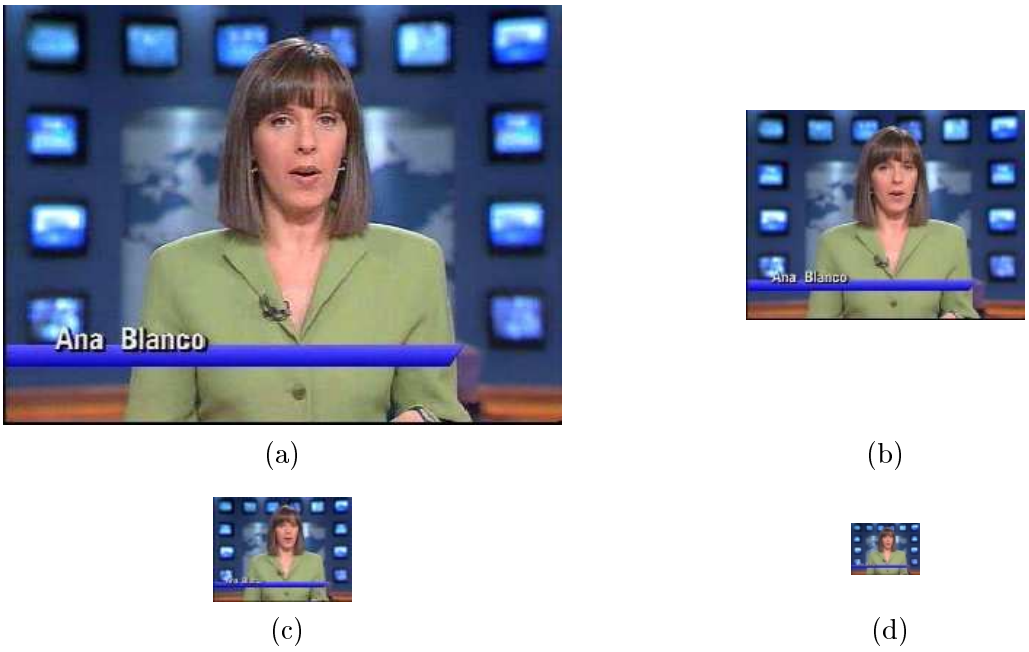

Figura 2.8: Ejemplo de Filtrado Paso-Bajo y Diezmado por 8 en tres pasos: (a)Imagen original. (b)Imagen filtrada y diezmada por 2. (c)Imagen filtrada y diezmada por 4. (d)Imagen definitiva, filtrada y diezmada por 8.

`P8Filtradiezma(imgdes *srcimg, imgdes *desimg)`,

donde `*srcimg` y `*desimg` son respectivamente el puntero a la imagen de entrada y a la imagen de salida. Por último, es necesaria una función que realice el proceso sobre la secuencia completa (sin comprimir):

`int IMP8FiltradiezmaSec(FILE *fileor,char *nomfiledest)`.

Los parámetros son:

- `*fileor`: Puntero al fichero *VAL* de la secuencia a diezmar (previamente abierto).
- `*nomfiledest`: Cadena de texto con el nombre del fichero *VAL* donde guardar la secuencia diezmada.

Esta última función toma los fotogramas uno a uno, y les aplica `P8Filtradiezma`.

2.4.5 Obtención de la secuencia *DC* con el PDIWin32

En la introducción a este proyecto, ya hemos mencionado que el PDIWin32 puede ser "personalizado" por cada programador, para añadir las funciones que necesite. La

forma habitual de trabajar en los proyectos fin de carrera, es añadir una nueva entrada de menú (en nuestro caso la hemos llamado **Jesús A.**), y que el resto de funciones y "POPUP's" cuelguen de ésta. En nuestro menú, aparece **Filtra-Diezma x 8** [7], que funciona para imágenes y para secuencias.

Posteriormente nos ha parecido interesante incluir en el núcleo general del PDIWIN32 una función que permitiese realizar la operación de Filtrado-Diezmado sobre imágenes y secuencias, pero con tres posibilidades: por 2, por 4 y por 8 (dando más generalidad y posibilidades). Para programar estas operaciones, solo se han tenido que realizar pequeñas modificaciones en el código de las funciones descritas anteriormente. Su situación dentro de los menús del PDIWIN32 se muestra en la figura 2.10.

Figura 2.9: Entrada de menú que realiza el filtrado-diezmado por 8 de una imagen o una secuencia.

[7] También aparece la llamada a **Filtra-Diezma x 8 Lista → Val**, que partiendo de una lista de ficheros (con imágenes "completas") generá una secuencia *VAL* con todos los fotogramas diezmados por 8. Para conocer el manejo de las listas de ficheros remitimos al lector al apéndice B.

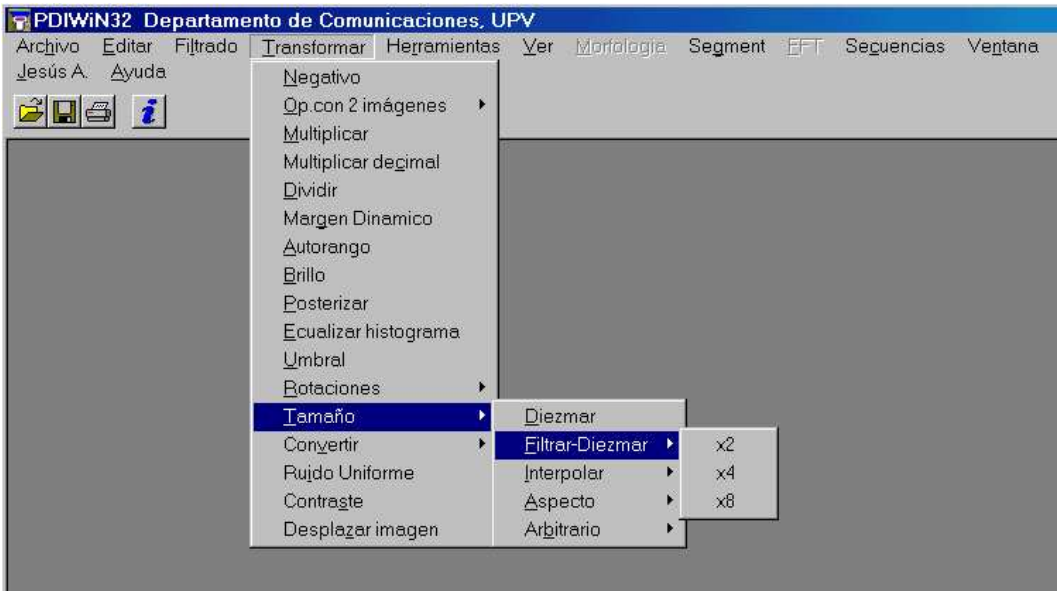

Figura 2.10: Entradas de menú que realizan el filtrado-diezmado por 2, 4 y 8 de una imagen o una secuencia, dentro del PDIWIN32 "general".

2.5 ESQUEMA GENERAL DEL ALGORITMO DE SEGMENTACIÓN UTILIZADO

Este último apartado del capítulo tienen como objetivo mostrar, muy esquemáticamente, el método de segmentación *implementado* en el proyecto. No entraremos en los fundamentos de cada étapa, ya que esto se desarrollará en profundidad en los capítulos siguientes.

Observemos en primer lugar, la figura 2.11, donde aparecen la étapas de las que consta nuestro método. En la sección anterior, ya hemos justificado el por qué de comenzar con la obtención de la secuencia *dc*, mediante el **Filtrado Paso-Bajo y Diezmado $\times$ 8**. A continuación aparece una segundad etapa denominada **Compensación de Brillo y Contraste**. Uno de los objetivos de este trabajo, es poder segmentar secuencias de vídeo de películas antiguas y que probablemente estarán muy deterioradas. Como se verá posteriormente, nuestros algoritmos se fundamentan en las diferencias del nivel de gris entre fotogramas sucesivos y al mismo tiempo, las variaciones aleatorias del brillo medio en de los fotogramas, es uno de los efectos más usuales de deterioro en las películas antiguas. La conclusión que se puede preveer (y que además hemos comprobado experimentalmente) es que la variaciones aleatorias del brillo en los frames de un plano, generarán falsas transiciones (ya sean de tipo abrupto o gradual). Para solucionar este problema inicial, hemos propuesto la posibilidad de compensar el brillo y contraste de los fotogramas de la secuencia. En el próximo capítulo entraremos en más profundidad, pero básicamente consiste en igualar el brillo medio de cada fotograma, al brillo medio de los fotogramas de una determinada ventana temporal que se va desplazando. Esta etapa es opcional, y carece de sentido cuando la secuencia a segmentar se encuentra en perfecto estado.

La siguiente etapa, la hemos denominado **Análisis Temporal**. Consiste en el cálculo de una serie de parámetros que carácterizan a la secuencia, después estos datos se guardan en ficheros de texto, para que su contenido puede ser utilizado en las etapas siguientes. Se realizan tres tipos de operaciones a los frames:

- *Operaciones con 1 frame*: Se calcula el valor de la *Media* y *Varianza* de cada fotograma.

- *Operaciones con 2 frames*: Esta información es la base de la detección de cortes. Se calcular dos parámetros: *Módulo de la diferencia de los frames n y n-1* y *Correlación entre los frames n y n-1.*

- *Operaciones con 3 frames*: Para este caso también se calculan otros dos parámetros: *Diferencia entre los frames n-1 y n+1* y *Correlación entre los frames n-1, n y n+1.* Este último dato, es el utilizado junto a la Varianza para detectar los fundidos.

Seguidamente, se debe proceder a la **Segmentación Temporal**. De esta etapa, decir únicamente que consiste en la detección de las fronteras entre planos. Se realiza por separado la detección de: Cortes, Fundidos y Cortinillas (utilizando los parámetros calculados anteriormente) y como resultado de ella, se vuelven a obtener otros ficheros de texto, necesarios para el **Postprocesado** de la información. En esta parte se realizarán principalmente dos cosas: combinar la información procedente de la fase anterior, para poder tener definitivamente delimitados los planos y, por otro lado obtener los fotogramas clave de cada plano. También hemos incluido algunas facilidades para la visualización sobre la secuencia de estos resultados.

Por último, se ha añadido una última etapa, consistente en la generación de **Información textual** sobre la secuencia. Todos los ficheros de texto obtenidos anteriormente, contendrán básicamente "números", interpretables en las funciones del PDIWIN32, pero que no dirán nada para el usuario ajeno al programa. Por eso, la inclusión de esta aplicación, que generará un fichero de texto describiendo todo el proceso de segmentación y que además también permite que se indexe el contenido de la secuencia, con una ventana que visualiza cada plano y permite describirlo textualmente.

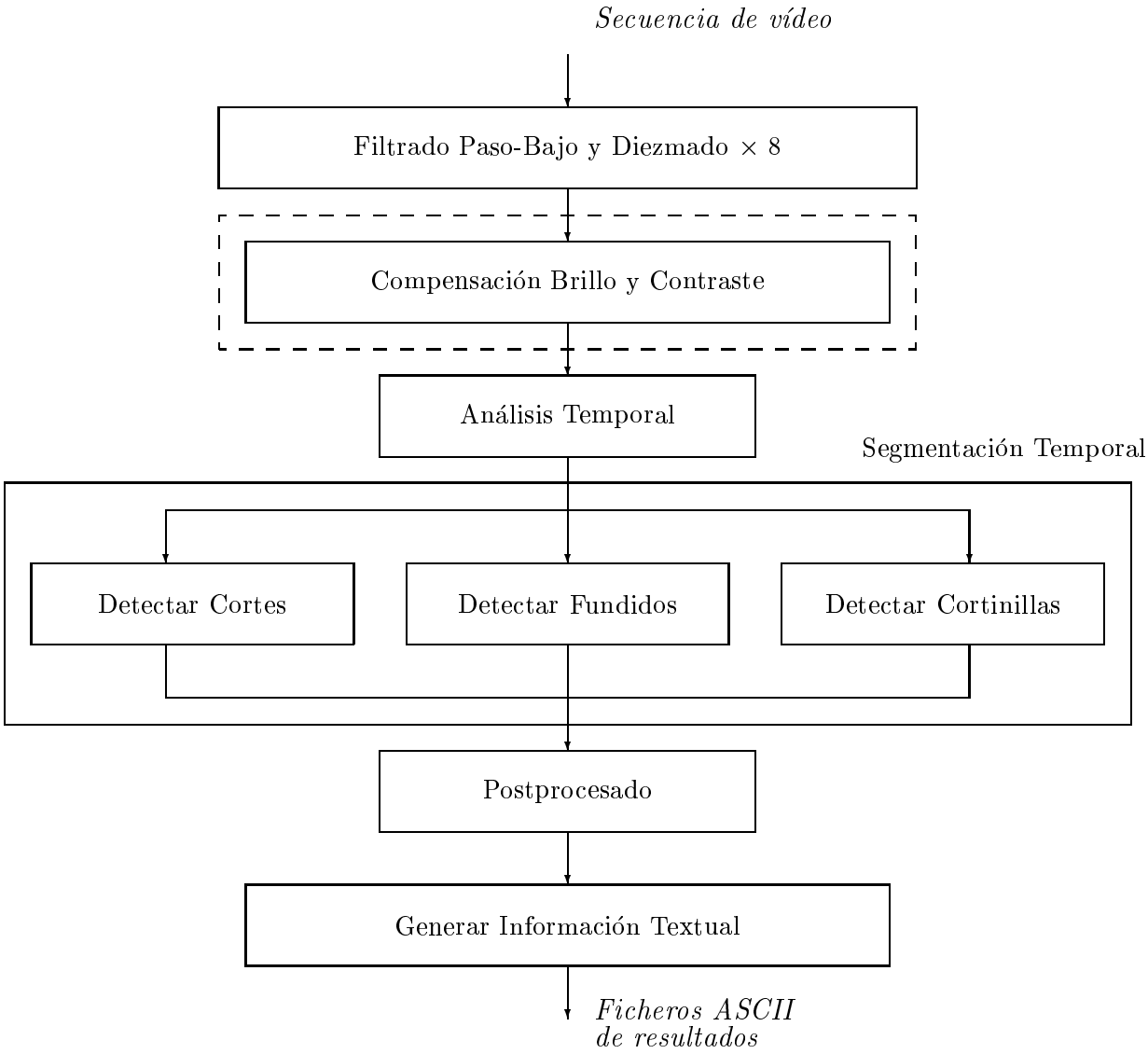

Figura 2.11: Diagrama con las distintas etapas del método de segmentación desarrollado en este proyecto.