PROYECTO FINAL DE CARRERA

HERRAMIENTRAS PARA LA INDEXACIÓN DE VÍDEO: EXTRACCIÓN DE IMÁGENES RELEVANTES Y ANÁLISIS DE IMÁGENES DE AGENCIA

Estudios: Ingeniería de Telecomunicación

Autor: Jonathan González Diéguez

Directores: Ferran Marqués Acosta

Antoni Gasull Llampallas

Año: 2011

Índice general

I. Colaboraciones

Este proyecto se ha realizado en el Grupo de Procesado de Imagen del departamento de Teoría de la Señal y Comunicaciones de la Universidad Politécnica de Cataluña. El desarrollo de los sistemas aplicados a la indexación de video queda enmarcado dentro de una colaboración con la Corporació Catalana de Mitjans Audiovisuals (CCMA) para el proyecto Buscamedia, bajo la dirección del Dr. Ferran Marqués y el Dr. Toni Gasull.

Buscamedia es un proyecto CENIT que pretende conseguir un avance significativo en las áreas de la semàntica, la producción audiovisual y la distribución de contenidos con el objetivo de crear un buscador semántico multimedia único en el mundo.

II. Resumen del Proyecto

En esta memoria se presenta, en primer lugar, un sistema de extracción de frames relevantes en secuencias de vídeo. Esto permite caracterizar con un número reducido de imágenes el contenido de una secuencia procedente de medios audiovisuales. De esta forma, se facilita la indexación automática o semiautomática y la posterior recuperación de contenido audiovisual. La elección de los frames que mejor representan la secuencia se realiza a partir de tres criterios: presencia de caras, presencia de texto y nitidez de la imagen. Además, como herramienta principal, se desarrolla un detector de cambios de escena basado en la publicación de Swain & Ballard *Color Indexing* [1].

En segundo lugar se presenta un sistema de reconocimiento de carátulas de agencias de noticias, las cuales suelen contener información sobre el contenido del video que les sigue. Mediante la extracción del texto podremos facilitar también las tareas de indexación y recuperación de contenidos en bases de datos.

III. Resum del Projecte

En aquesta memòria es presenta, en primer lloc, un sistema d'extracció de frames rellevants en seqüències de vídeo. Això ens permet caracteritzar amb un número reduït d'imatges el contingut d'una seqüència típica procedent de mitjans audiovisuals. D'aquesta manera es facilita la indexació automàtica o semiautomàtica i la posterior recuperació de contingut audiovisual. L'elecció dels frames que millor representen la seqüència es realitza a partir de tres criteris: presencia de cares, presencia de text i nitidesa de la imatge. A més, com a eina principal, es desenvolupa un detector de canvis d'escena basat en la publicació de Swain & Ballard *Color Indexing* [1].

En segon lloc, es presenta un sistema de reconeixement de caretes d'agència de notícies, les quals solen aportar informació del contingut del vídeo que precedeixen. Mitjançant l'extracció del text podrem facilitar també les tasques d'indexació i recuperació de continguts en bases de dades.

IV. Abstract

Firstly, in this report we will introduce a relevant frame extractor system from a video sequence. This allows us to characterize with a small number of images the content of a media sequence. Thus, the task of automatic audiovisual content indexing and retrieval becomes easier. The choice of frames that best represent the sequence occurs from three criteria: the presence of faces, the presence of text and image sharpness. Furthermore, as the main tool, it is developed a scene change detector based on the publication of Swain&Ballard *Color Indexing* [1].

Secondly we present a news agencies covers recognition system, which often contain information about the content of the video that follows. By extracting the text, we will also make easier the task of content indexing and retrieval from databases.

V. Introducción

El sector audiovisual se encuentra directamente afectado por factores internos (fragmentación de la audiencia, cambios en los modelos de consumo) y externos (crisis económica, globalización, popularización de las tecnologías de gestión digital para la creación, distribución y consumo de contenidos) que hacen que se vea inmerso actualmente en un momento de profunda reestructuración. Sigue siendo un sector estratégico, tanto por el potencial de crecimiento como por la capacidad de creación de opinión pública, que sin embargo ha de asumir la pérdida de privilegios históricos y orientarse claramente a mercado. En este aspecto, la eficiencia operativa es una necesidad vital para cualquier empresa audiovisual que desee asegurar su supervivencia. Y la innovación tecnológica es una de las claves para llevar a buen puerto estas mejoras operativas, y asegurarse de este modo la supervivencia a largo plazo y una cuenta de resultados saneada.

Puesto que la materia prima con la que operan las televisiones y otras entidades audiovisuales son los assets (activos) audiovisuales (es decir, tríadas de vídeo, audio, y metadatos), cualquier mejora en la indexación y/o recuperación de los mismos impactará directa y positivamente sobre el coste y tiempo empleado por los usuarios profesionales en la producción y gestión de los contenidos. De forma análoga, crecerá el número de servicios y la calidad percibida por el usuario final.

El presente documento introduce las dificultades de aplicar las tecnologías clásicas de procesado de imagen a la indexación automática o semi-automática de vídeos televisivos. Así mismo, cara a contemplar este reto desde una perspectiva global, se enumeran los inconvenientes que más gravemente acucian este tipo de innovación tecnológica, y se da una referencia del orden de magnitud y volumen de los mismos.

A continuación se presentan dos herramientas que ayudan a la indexación de assets audiovisuales:

- En primer lugar, un sistema que extrae las imágenes más relevantes, desde el punto de vista del contenido, de una secuencia de video. Para ello divide la secuencia en fragmentos donde se producen cambios de escena. Para cada escenario, extrae de forma inteligente la imagen más representativa o con información más relevante en cuanto a calidad de la imagen (evitar imágenes borrosas, fusionadas o mal enfocadas), presencia de texto y aparición de caras. De esta forma tendremos como resultado una representación de imágenes que permitirán contextualizar el video de entrada, permitiendo así una mejor indexación en una base de datos así como una mejor recuperación del mismo.

- En segundo lugar, una herramienta que detectará un conjunto particular de assets o secuencias: las noticias de agencia.
  Este tipo de información audiovisual consta de noticias de cualquier tipo y ámbito (deportes, sociedad, nacional, internacional, etc.) precedidas de su correspondiente portada. Esta portada contiene la carátula de la agencia con una serie de información de forma textual relativa a la noticia: título, descripción, fecha, etc. La detección de este tipo de imágenes y la posterior extracción de su contenido en formato texto permitirá obtener una mayor información sobre el contenido de la noticia, facilitando así la labor de indexación.

VI. Extractor de KeyFrames

1. Motivación

Varios estudios empíricos demuestran que son tres los parámetros básicos por los que un usuario profesional sin conocimientos técnicos evalúa y valora un sistema automático o semi-automático de indexación y/o de búsqueda de contenidos audiovisuales: la precisión, el tiempo de respuesta y la escalabilidad [2].

Según la funcionalidad escogida, el orden de prioridad de estos factores será uno u otro. Por ejemplo, en la búsqueda de contenidos en una base de datos prima el tiempo de respuesta y la escalabilidad, mientras que en la indexación automática de texto o caras en los mismos contenidos, la precisión y un tiempo de respuesta aceptable son los factores determinantes.

Sin duda, la variabilidad de la tipología de contenidos y el volumen diario de contenidos gestionados en el sector de la televisión son claros enemigos antagónicos de los tres parámetros antes mencionados. Para tener una idea del orden de magnitud, pondremos números a un ejemplo concreto, en este caso a la casuística de Televisió de Catalunya, una televisión autonómica que puede considerarse de tamaño medio dentro del espectro mundial de televisiones.

Por lo que respecta a la variabilidad de los contenidos, comentar que si observamos las horas de programación anual de TV3 (el canal generalista de la cadena, líder de audiencia en Cataluña), el 41% corresponden a noticias, el 15,6% a contenidos de entretenimiento, el 2,6% a contenidos dramáticos, el 6.7% a deportes, el 7,5% a publicidad, el 21,7% a contenidos de producción ajena (películas, documentales, etc), y el resto –un 4,9%- a otro tipo de contenidos (nuevos formatos, cultura y especiales, elementos de continuidad, patrocinios y documentales de producción interna). La diversidad de los contenidos es pues muy notable, y a parte se incrementa si tenemos en cuenta los contenidos de los otros seis canales de televisión del mismo ente televisivo[1]. A parte, cada tipo de contenido no es homogéneo en sí, sino que también incluye gran variabilidad en las temáticas y las instancias (caras, objetos, etc) que presenta. El ejemplo más claro es el de los telenoticias. En éstos es fácilmente observable que se tratan temas tan dispares como noticias de sociedad, de economía, de política nacional e internacional, de tecnología, de cultura, de meteorología, etc. Por otra parte, y tal como se verá más adelante, uno de los mayores inconvenientes para la automatización de un sistema de anotación es la falta de predictibilidad de los contenidos. Sin duda, ésta es máxima en los contenidos en directo (un 52% de la emisión de TV3 es en directo), donde inusualmente existe un guión pre-escrito.

Por lo que respecta al volumen de la información audiovisual que se gestiona en una televisión de este tamaño, comentar que para gestionar los contenidos audiovisuales de la cadena, se necesitan

[1] Canal 33, Canal 3XL, Club Super 3, canal 3/24, canal 3 esports I TV3 HD.

más de 850 estaciones digitales de trabajo, más de 60 servidores de ingesta, unos 95 servidores de playout y 65 servidores con programas autómatas, y más de 130 servidores de ficheros que gestionan unas 30.000 horas online, así como 2 robots con una capacidad de almacenamiento de 3 PetaBytes[2]. Si tenemos en cuenta que se pueden ingestar contenidos en el sistema desde cualquiera de las 25 estaciones dedicadas de ingesta[3], que cualquier periodista puede digitalizar una cinta desde su estación de trabajo, que el Departamento de Cambio de Formatos digitaliza ininterrumpidamente películas y documentales de producción ajena, y que el departamento de Documentación documenta anualmente de media unas 26.000 horas archivo[4] (el equivalente a más de 3 años de visionado continuo), concluimos que en una misma hora puede llegar a generarse un pico de más de 100 horas de nuevo contenido en el sistema. De hecho, cada hora se generan una media aproximada de unas 15 a 20 horas de contenido nuevo (la mayoría del cuál, eso sí, caduca al cabo de pocos días) [5].

Ante estos números, el gran dilema que se plantea es: ¿cómo anotar esta inmensa cantidad de nueva información que entra en un sistema digital gestor de contenidos audiovisuales para que sea más accesible al usuario? ¿Cómo mejorar por otra parte la anotación de todo el fondo de archivo ya existente de las televisiones?

Es obvio que es necesario un conjunto de herramientas automáticas que faciliten esta labor.

[2] El equivalente a unas 480.000 horas de vídeo

[3] Las estaciones de ingesta se utilizan para digitalizar señales provenientes del exterior: señales de satélite (noticias de agencia, enlaces propios por satélite, eventos deportivos), radioenlaces, o fibras ópticas dedicadas, como por ejemplo las que enlazan directamente el Parlament de Catalunya, el Camp Nou o el Estadio Olímpico con TV3.

[4] De éstas 26.000 horas, 10.000 corresponden aproximadamente a la digitalización del fondo de archivo (cintas históricas), y 16.000 corresponde a nuevo material del mismo año. Actualmente hay más de 140.000 horas de archivo digitalizadas. Es decir, una persona necesitaría más de 16 años sin descanso para poder visionar todo este contenido.

[5] Fuente: Corporació Catalana de Mitjans Audiovisuals (CCMA).

2. Estado del arte

El sistema implementado en la CCMA por la compañía Visual Century Research S.L. se compone de un módulo (plugin) o filtro DirectShow llamado CutDetector. Este filtro es capaz de reconocer los cambios de escena que aparecen en un vídeo. Éste será el sistema que se pretende mejorar.

Cada vez que se detecta un cambio de escena (cut) se almacenan el frame y su timecode. Este frame queda asignado como keyframe. Un keyframe es una imagen que caracteriza una escena. El timecode indica la posición del vídeo donde se ha encontrado el keyframe.

Como podemos ver, esto tan sólo lleva a una selección de imágenes relacionadas con las transiciones entre planos y no con la relevancia de la imagen en el plano.

El plugin de detección de cuts CutDetector es la pieza básica para indexar el vídeo y hacerlo recuperable. La detección de cuts permite segmentar el vídeo en shots (escenas, tomas de cámara, segmento entre cut y cut). El shot es la unidad mínima de vídeo tratada dentro de ViA2 Platform. El plugin CutDetector genera una única escena de todo el vídeo. Esta escena contiene todos los shots que se han detectado.

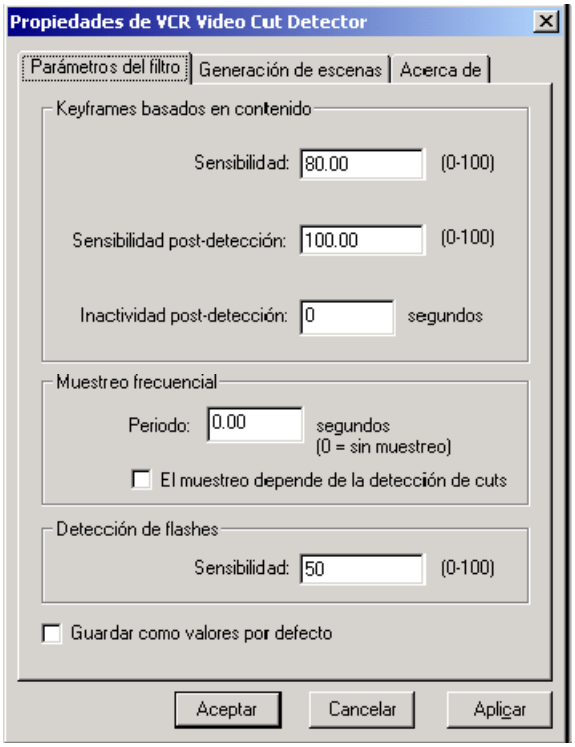
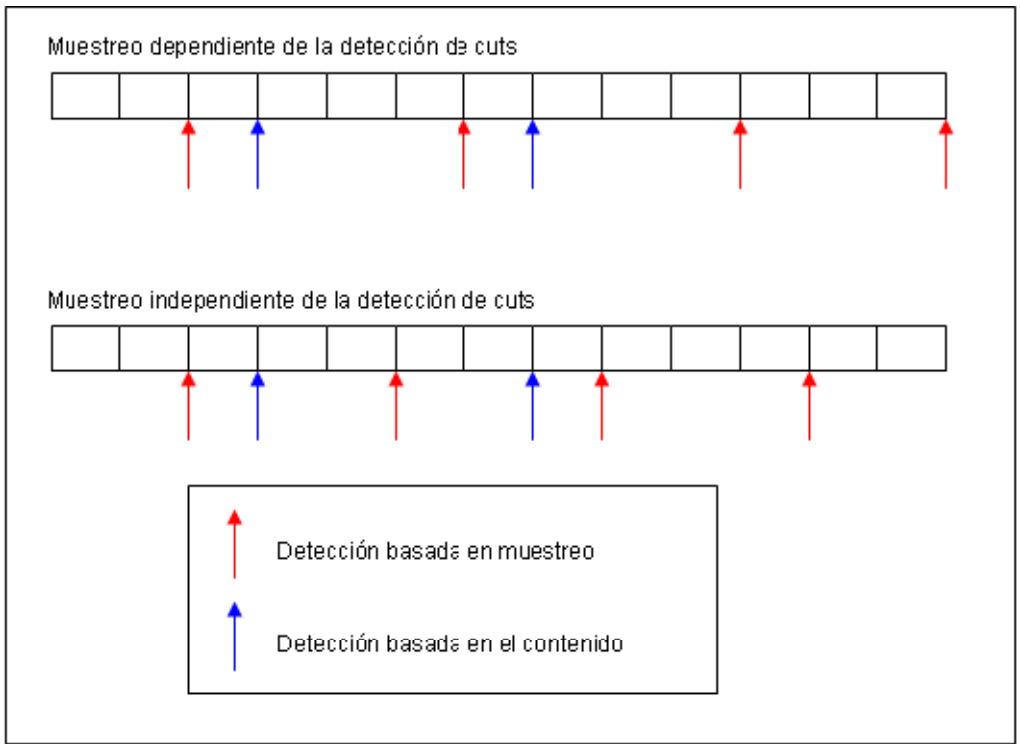


Figura 1: Interfaz del plugin detector de cambios de escena.

La detección de cuts desarrollada, puede basarse en el contenido o bien en un muestreo frecuencial. En la detección de cuts basada en el contenido, los cuts se detectan de manera inteligente analizando el contenido del vídeo. En la detección de cuts basada en un muestreo frecuencial, se define un periodo de un número de segundos y se genera un cut en cada periodo. CutDetector permite crear una escena cada número especificado de shots. Por ejemplo, si así se especifica en la hoja de propiedades, CutDetector generará automáticamente una escena cada 50 shots. Por tanto, las sesiones resultantes tendrán escenas de 50 shots creadas automáticamente por CutDetector. Esta opción resulta muy útil cuando las sesiones que se generan son muy largas y tienen miles de shots, ya que hace las hace mucho más manejables.



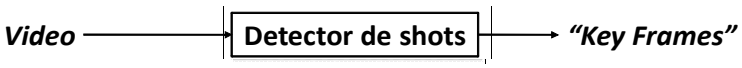
El método de detección se basa en la publicación *Color indexing* (Swain-Ballard, 1991) [1]. Este artículo demuestra que los histogramas de color proporciona una señal robusta y eficiente para la indexación en una gran base de datos de modelos. Los histogramas de color son representaciones estables en la presencia de oclusiones y cambios de vista, y que puede diferenciar entre un gran número de objetos.

Para identificar los cambios de escena, se utiliza la técnica llamada Intersección de Histogramas, la cual permite relacionar dos histogramas de entrada.

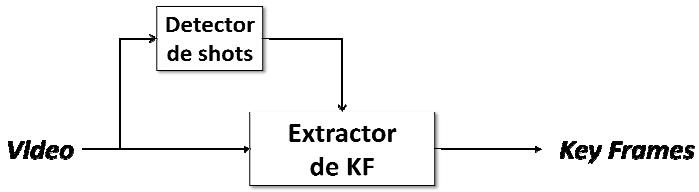
El primer paso para el desarrollo de nuestro sistema de extracción, será replicar el filtro DirectShow dentro de la librería desarrollada por el Grupo de Procesado de Imagen de la UPC, Image+. A continuación se desarrollará un conjunto de herramientas para que el sistema reconozca realmente una imagen relevante de la escena.

3. Esquema

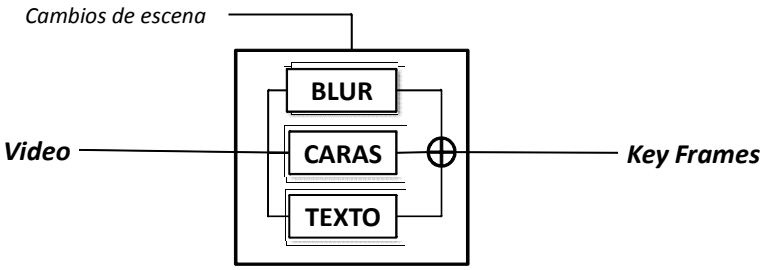
El sistema inicial consta de una entrada de video y un detector de cambios de escena (CutDetector). Nos proporciona a la salida una instantánea del momento en que se produce el cambio de imagen así como el timestamp.



Nuestro esquema detector de imágenes relevantes deberá tener una arquitectura similar, añadiéndole un bloque que se encargará de seleccionar el conjunto de imágenes que mejor representan el video de entrada.



Nuestro extractor de keyframes obtiene por un lado la propia secuencia de video, para poder analizar todos los frames, y por otro lado los timestamp de los cambios de escena. De esta forma, entre dos timestamps tendremos definida una escena y de ésta escogeremos un frame de entrada en base a 3 criterios:



El detector de blur nos informa sobre la borrosidad del frame.

El detector de caras nos indica la presencia, o no, de caras frontales en el frame actual.

El detector de texto analiza la posibilidad de la presencia de texto en la imagen.

A continuación se describen las principales características y funcionalidades de cada bloque así como las razones por las que se utilizan.

4. CutDetector: detector de cambio de escena

La primera pregunta que nos podemos plantear al efectuar el análisis de nuestra secuencia para extraer las imágenes más representativas, será saber cuántas serán necesarias. Es decir, cuantas imágenes necesitaremos para que un usuario tome consciencia del contexto del video.

La parte más elemental de cualquier secuencia audiovisual es la escena. Una escena se define como la unidad más pequeña de una secuencia, en la que aparecen los mismos personajes, objetos relevantes o entorno. Nuestro objetivo será detectar y dividir nuestro video en escenas. Para cada una de ellas, debemos extraer el frame que mejor representa a toda la escena teniendo en cuenta una serie de premisas que veremos más adelante. De esta forma, se obtienen un conjunto de imágenes que "resumen" el video.

Los **detectores de cambio de plano** son elementos que ayudan a la reducción del espacio de búsqueda, ya que se supone que los objetos se mantienen más o menos estables dentro del plano o escena y que un evento sucede dentro de un único plano (exceptuando el caso de repeticiones, cuya detección es un problema adicional). Las soluciones ya existentes a este problema son muy robustas, sobre todo si no se tiene en cuenta los cambios de planos más complejos debidos a efectos artísticos que, de hecho a este nivel, no son tan habituales[6].

Por su parte, la **detección de imágenes relevantes** es una técnica habitual de resumen del plano. El problema es que los criterios mediante los cuales se determinan las imágenes como relevantes están dirigidos, en el mejor de los casos[7], a facilitar la visualización por parte de un usuario humano y no la posterior detección de los objetos y eventos de interés. En los sistemas comerciales se suele utilizar el mismo detector de planos para generar también las imágenes relevantes, lo que suele llevar a una selección de imágenes relacionadas con las transiciones entre planos y no con la relevancia de la imagen en el plano.

Como ya hemos mencionado, el método de detección se basa en la publicación *Color Indexing* (Swain-Ballard, 1991) [1] por su robustez y eficiencia para la indexación en una gran base de datos de modelos. Los histogramas de color son representaciones estables en la presencia de oclusiones y cambios de vista, y que puede diferenciar entre un gran número de objetos. Dicho esto, ¿cómo los comparamos?

Para resolver el problema de la comparación, presenta la técnica llamada Intersección de Histogramas, la cual puede relacionar los histogramas del frame anterior (modelo) y del frame actual (imagen).

---

[6] Posteriormente se comentará el caso del texto en la escena, donde la creatividad artística genera una gran variabilidad que hace que el problema de su detección y reconocimiento automático sea una tarea muy complicada.

[7] Son habituales los sistemas que generan las imágenes relevantes a intervalos iguales de tiempo, sin atender a sus contenidos reales.

4.1 Histograma de color

Dado un espacio de color discreto definido por unos ejes (por ejemplo, rojo, verde y azul), el histograma de color se obtiene discretizando los colores de la imagen y contando el número de veces que cada color discreto ocurre. Los histogramas son invariantes a translaciones o rotaciones de las imágenes y varían muy poco frente a cambios de ángulo, escala u oclusión. Por esta razón y aceptando como premisa que una misma escena se considerará única mientras sólo varíen estos parámetros, el estudio de la evolución de los histogramas de color para cada frame a lo largo del tiempo nos dirá en qué momento se produce un cambio de escena.

En la siguiente figura vemos para una imagen de ejemplo, su histograma a escala de grises, una representación habitual. También podríamos representar para cada canal (rojo, verde y azul) su nivel de luminancia. Sin embargo lo que buscamos es un histograma de los colores de la imagen, por tanto nos gustaría ver para cada color, cuantos píxeles contienen ese color.

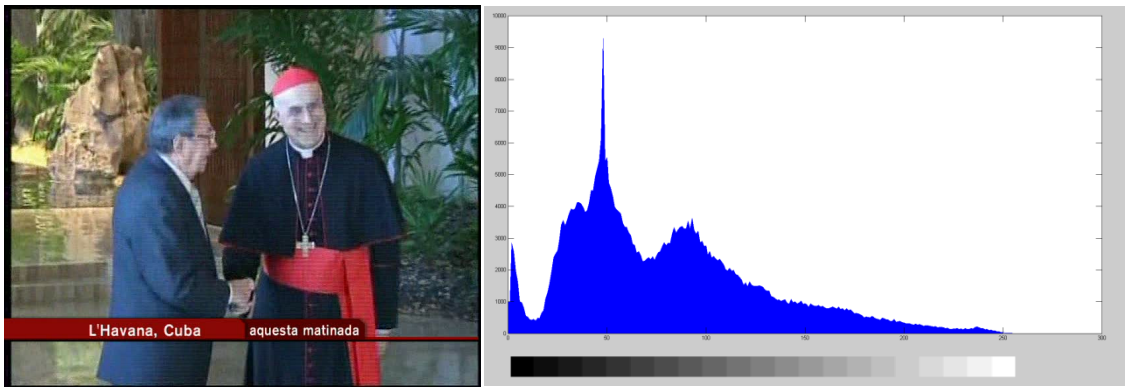


Figura 2: Imagen original e histograma en escala de grises

Para nuestro trabajo, basándonos en las pruebas realizadas y en las recomendaciones de Swain-Ballard, se ha escogido utilizar 512 bins para los histogramas. Queda asignada entonces una cuantificación de 3 bits, es decir, 8 niveles (de 0 hasta 7) para cada canal RGB (8x8x8). Lo que haremos será asignar a cada canal un peso.

Canal rojo * 1

Canal verde * 8

Canal azul * 64

Si a continuación sumamos los 3 canales en uno solo, obtendremos una matriz con mismo número de filas y columnas que la imagen original y con valores comprendidos entre 0 a 511. Cada uno de ellos representará únicamente a un color de la imagen. Por ejemplo, el valor 315 representara en (R,G,B) a (3,7,4) o el valor 511 representa al color blanco (7,7,7).

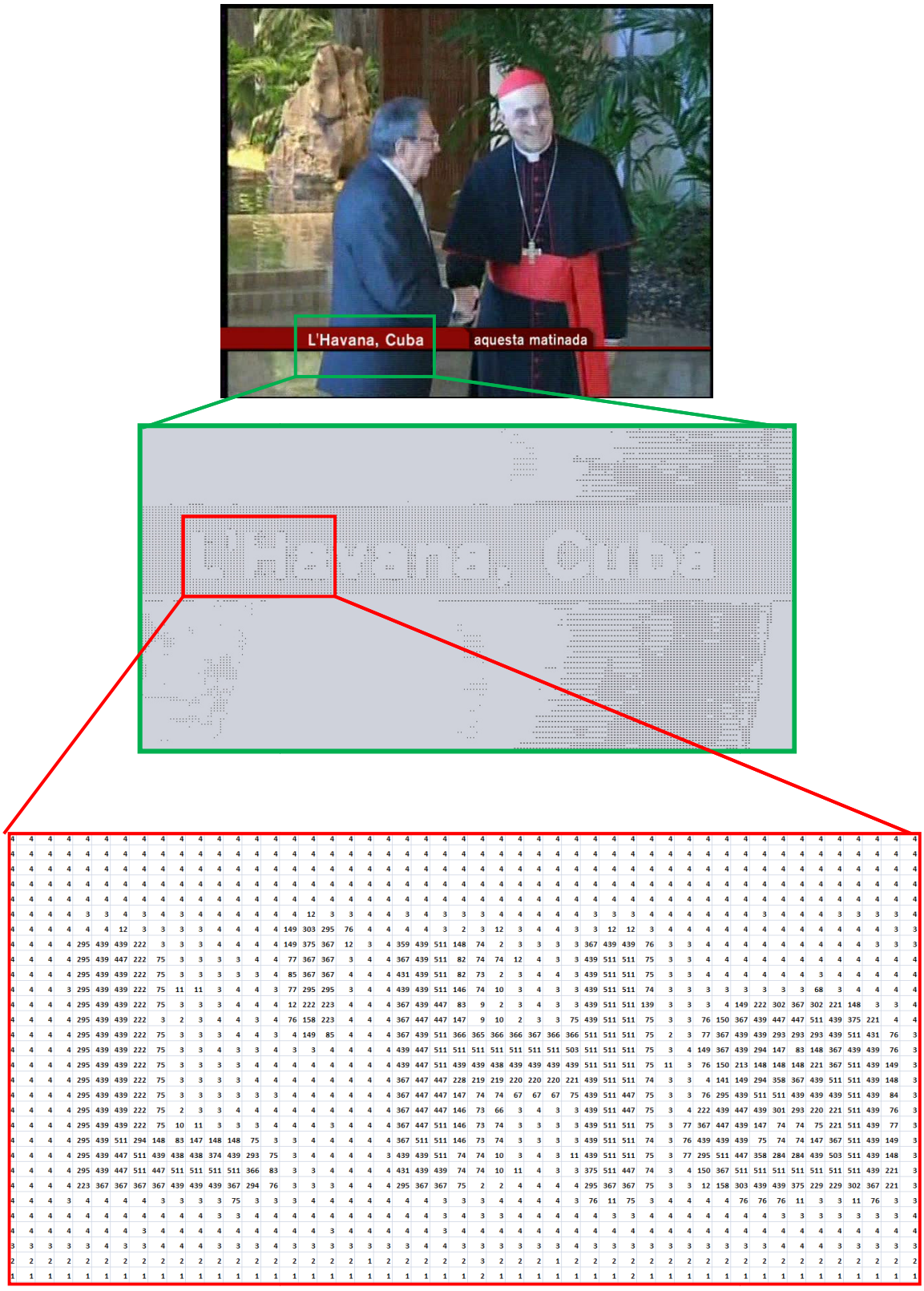


Figura 3: Detalle los valores de cada píxel, correspondientes a un único color

Los histogramas definen una función de equivalencia frente al surtido de colores posibles, estableciendo a dos colores como iguales si "caen" en la misma clase/barra (*bin*) del histograma. Esta función de equivalencia no es ideal para reconocimiento, ya que el rango de colores relativo que consideramos como el mismo color al dado puede ser mayor o menor, por lo que el color clasificado depende de dónde se sitúa dentro del bin. Idealmente, los colores considerados como iguales, estarían en una región centrada en el color, o en una región cuya forma dependiese de las posibles variaciones conocidas introducidas por cambios en iluminación, ruido de los sensores del dispositivo y la codificación. Se podrían definir histogramas con bins de forma Gaussiana u otras que tuviesen mejor en cuenta estos tipos de ruido, sin embargo los resultados con bins "clásicos" (un color de la imagen original únicamente pertenece a un conjunto) son suficientemente correctos.

Con tantos problemas, ¿cómo es que funciona? Los elementos que aparecen en una imagen (personajes, objetos,…) incluso el fondo tienden a tener superficies compuestas a partir de regiones de color. Debido a efectos de sombras y ruido de la cámara, estas regiones se difuminan en el espacio de color incluso más que el ancho del bin en el histograma. Sin embargo, cuando el histograma de la imagen y del modelo de la escena se comparan, se obtiene un valor alto de relación porque las regiones concuerdan bien, aún cuando no coinciden punto a punto.

Como característica adicional, el detector de cambios de escena es capaz de detectar flashes, cancelando la aparición de una nueva escena, o secuencias en negro. Para ello se mide el parámetro que llamaremos intensidad. La intensidad no es más que la suma de los valores de todos los píxeles en los tres canales normalizado por el número de píxeles. Una intensidad muy baja indicará la presencia de una escena negra y, por el contrario, una intensidad muy alta nos indicará la presencia de un destello o flash, en cuyos casos se debe evitar generar una nueva escena.

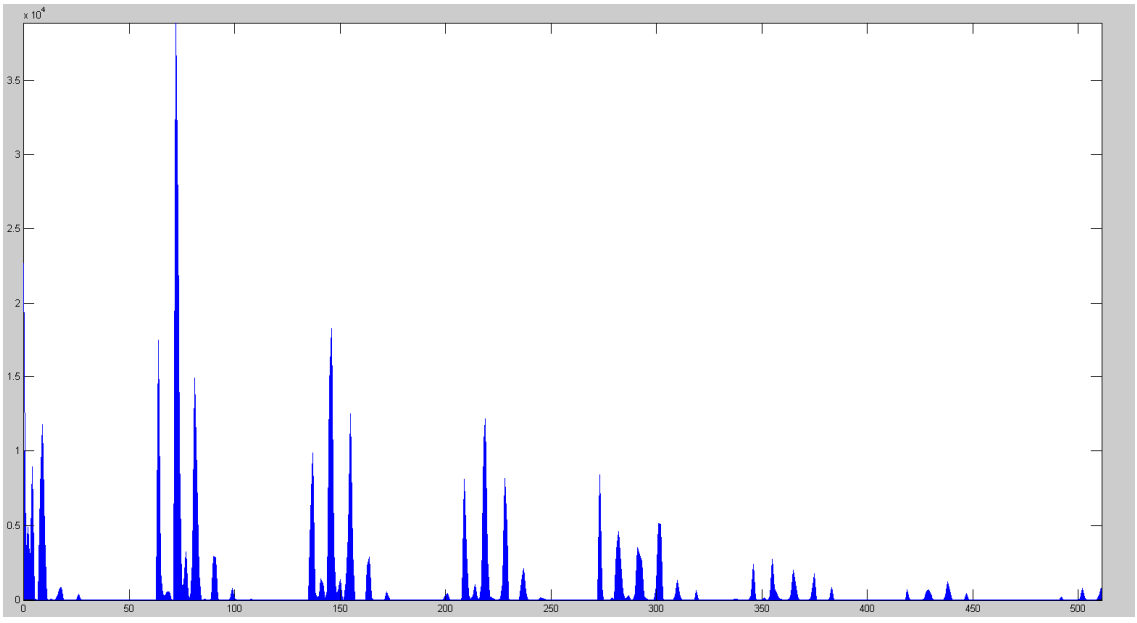


Figura 4: Histograma de color de la imagen original.

4.2 Comparación

Para comparar objetos basados en su Histograma de color, tenemos que ser capaces de juzgar su similitud con los histogramas de color. En el siguiente apartado se presenta un método para comparar imágenes e histogramas llamado Intersección de Histograma, el cual nos dice cuántos píxeles del histograma modelo se encuentran en la imagen a comparar. Este método encaja perfectamente con nuestras pretensiones porque no requiere de una extracción precisa de los objetos de su fondo o identificar oclusiones en primer plano (personas que deambulan por delante del objetivo, etc.).

4.2.1 Intersección de histogramas

Dado un par de histogramas, I y M, cada uno con n bins, la intersección de los histogramas se define como:

$$H_i(I, M) = \sum_{j=1}^{n} min(I_j, M_j)$$

El resultado de la intersección de un histograma modelo H(M) con un histograma de una imagen H(I) es el número de píxeles en M que coinciden con píxeles del mismo color en la imagen I.

Para obtener un valor entre 0 y 1 normalizamos la intersección por el número de píxels en el histograma modelo. El valor es entonces:

$$H(I, M) = \frac{\sum_{j=1}^{n} min(I_j, M_j)}{\sum_{j=1}^{n} M_j}$$

Las diferentes pruebas realizadas dan un valor superior a 0.8 para frames consecutivos considerados de la misma escena. Cuando se produce un cambio de escena por tanto, el matching es un valor inferior a este umbral, el cual podemos ajustar según la sensibilidad que queramos proporcionar al detector.

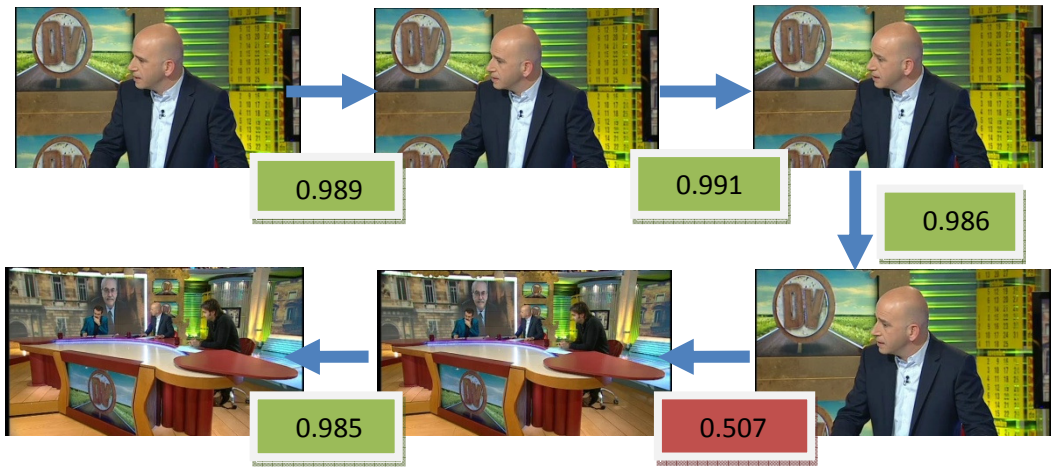


Figura 5: Ejemplo de comparación de imágenes con los valores de matching. En rojo, cambio de escena detectado.

5. Análisis de blurring

En secuencias de video del ámbito televisivo es frecuente encontrar transiciones entre escenas realizadas mediante la técnica de fundido. Esta técnica es más agradable a la vista que una transición repentina y consiste en sumar los últimos frames de la escena con los primeros de la siguiente, consiguiendo un efecto de transparencia. Mediante el uso de los pesos adecuados en la suma, suavizaremos la transición de una escena a la siguiente. Según el efecto que se le quiera dar esto puede durar desde un par a decenas de frames.

Realmente algunos extractores de keyframes convencionales son directamente el detector de cambio de escena, por lo que el frame escogido contiene un fundido de dos escenas, algo que no es representativo de una ni de otra, por no hablar de la calidad de imagen.

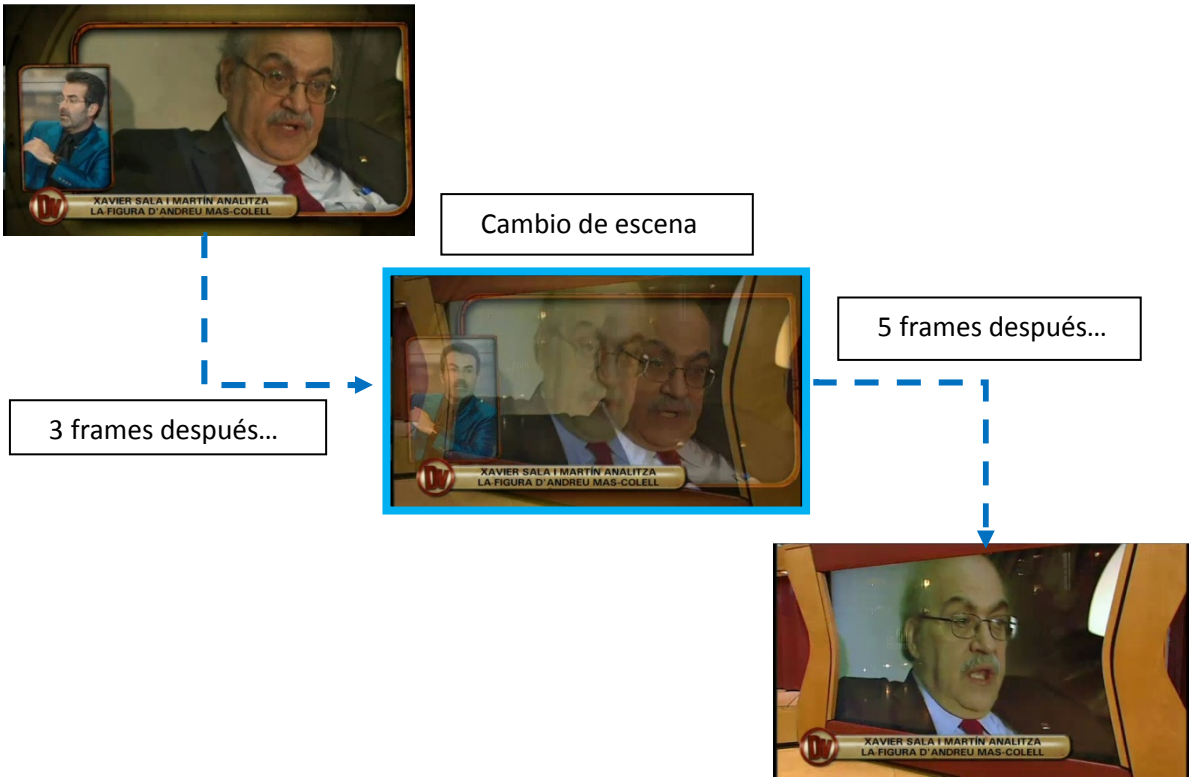


Figura 6: Ejemplo de uso de la técnica de fusión y el resultado del detector.

Por otro lado, algo relacionado con la calidad de la imagen son los errores en el enfoque o cualquier otro efecto de difuminado. Si tenemos que escoger una imagen que caracterice una escena es obvio que debemos asegurarnos de que tenga una buena calidad en términos de nitidez.

Para ello, un bloque básico para escoger un buen conjunto de frames candidatos a ser keyframes será el de análisis de blurring (borrosidad). Una sencilla, aunque efectiva, forma de detectar estos emborronamientos es calculando la energía de los contornos que aparecen en la imagen. Necesitamos pues, un detector de contornos.

5.1 Detector de contornos

En el área de procesamiento de imágenes, la detección de los bordes de una imagen es de suma importancia y utilidad, pues facilita muchas tareas, entre ellas el reconocimiento de objetos y la segmentación de regiones.

Existe una larga lista de detectores de contornos conocidos: Sobel, Prewitt, Laplaciano, Roberts, etc. En este proyecto nos hemos centrado en el detector de Canny [3], considerado uno de los mejores métodos y disponible en la librería Image+.

5.1.1 Criterios del Algoritmo de Canny

En 1986, Canny propuso un método para la detección de bordes, el cual se basaba en tres criterios, estos son:

- Un criterio de detección expresa el hecho de evitar la eliminación de bordes importantes y no suministrar falsos bordes.
- El criterio de localización establece que la distancia entre la posición real y la localizada del borde se debe minimizar.
- El criterio de una respuesta que integre las respuestas múltiples correspondientes a un único borde.

5.1.2 Algoritmo de Canny para la detección de bordes

Uno de los métodos relacionados con la detección de bordes es el uso de la primera derivada, porque toma el valor de cero en todas las regiones donde no varía la intensidad y tiene un valor constante en toda la transición de intensidad. Por tanto un cambio de intensidad se manifiesta como un cambio brusco en la primera derivada [1], característica que es usada para detectar un borde, y en la que se basa el algoritmo de Canny.

El algoritmo de Canny consiste en tres grandes pasos:

- Obtención del gradiente: en este paso se calcula la magnitud y orientación del vector gradiente en cada píxel.
- Supresión no máxima o de los no máximos: en este paso se logra el adelgazamiento del ancho de los bordes, obtenidos con el gradiente, hasta lograr bordes de un píxel de ancho.
- Histéresis de umbral: en este paso se aplica una función de histéresis basada en dos umbrales; con este proceso se pretende reducir la posibilidad de aparición de contornos falsos.

5.1.2.1. Obtención del gradiente

Para la obtención del gradiente, lo primero que se realiza es la aplicación de un filtro gaussiano a la imagen original con el objetivo de suavizar la imagen y tratar de eliminar el posible ruido existente. Sin embargo, se debe de tener cuidado de no realizar un suavizado excesivo, pues se podrían perder detalles de la imagen y provocar un pésimo resultado final. Este suavizado se obtiene promediando los valores de intensidad de los píxeles en el entorno de vecindad con una máscara de convolución de media cero y desviación estándar σ. En la figura X se muestran dos ejemplos de máscaras que se pueden usar para realizar el filtrado gaussiano.

Una vez que se suaviza la imagen, para cada píxel se obtiene la magnitud y módulo (orientación) del gradiente, obteniendo así dos imágenes.

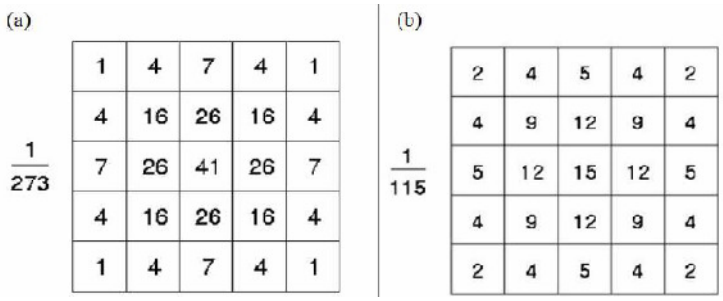


Figura 7. Máscaras de convolución recomendadas para el obtener el filtro gaussiano. La máscara (a) fue obtenida de [4], mientras que la máscara (b) fue obtenida de [5].

5.1.2.2. Supresión no máxima al resultado del gradiente

Las dos imágenes generadas en el paso anterior sirven de entrada para generar una imagen con los bordes adelgazados. El procedimiento es el siguiente: se consideran cuatro direcciones identificadas por las orientaciones de 0º, 45º, 90º y 135º con respecto al eje horizontal. Para cada píxel se encuentra la dirección que mejor se aproxime a la dirección del ángulo de gradiente.

Posteriormente se observa si el valor de la magnitud de gradiente es más pequeño que al menos uno de sus dos vecinos en la dirección del ángulo obtenida en el paso anterior. De ser así se asigna el valor 0 a dicho píxel, en caso contrario se asigna el valor que tenga la magnitud del gradiente.

La salida de este segundo paso es la imagen con los bordes adelgazados, es decir, después de la supresión no máxima de puntos de borde.

5.1.2.3. Histéresis de umbral a la supresión no máxima

La imagen obtenida en el paso anterior suele contener máximos locales creados por el ruido. Una solución para eliminar dicho ruido es la histéresis del umbral.

El proceso consiste en tomar la imagen obtenida del paso anterior, tomar la orientación de los puntos de borde de la imagen y tomar dos umbrales, el primero más pequeño que el segundo. Para cada punto de la imagen se debe localizar el siguiente punto de borde no explorado que sea mayor al segundo umbral. A partir de dicho punto seguir las cadenas de máximos locales conectados en

ambas direcciones perpendiculares a la normal del borde siempre que sean mayores al primer umbral. Así se marcan todos los puntos explorados y se almacena la lista de todos los puntos en el contorno conectado. Es así como en este paso se logra eliminar las uniones en forma de Y de los segmentos que confluyan en un punto.

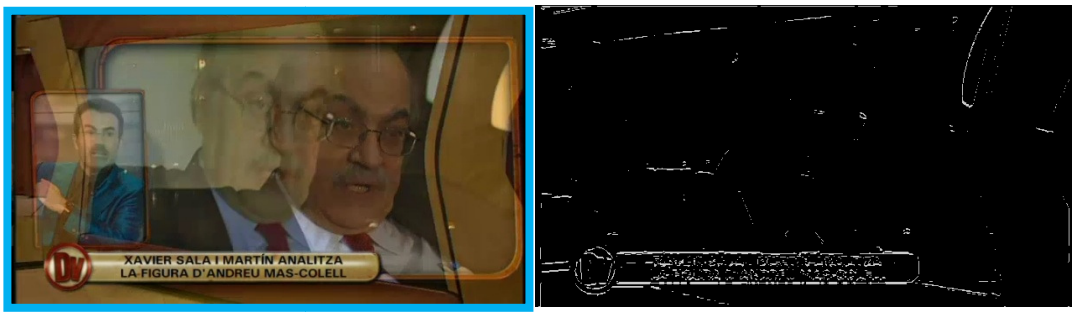
5.1.2.4. Un cuarto paso

Frecuentemente, es común se realice en el algoritmo de Canny un cuarto y último paso. Este paso consiste en cerrar los contornos que pudiesen haber quedado abiertos por problemas de ruido.

Un método muy utilizado es el algoritmo de Deriche y Cocquerez. Este algoritmo utiliza como entrada una imagen binarizada de contornos de un píxel de ancho. El algoritmo busca los extremos de los contornos abiertos y sigue la dirección del máximo gradiente hasta cerrarlos con otro extremo abierto.

El procedimiento consiste en buscar para cada píxel uno de los ocho patrones posibles que delimitan la continuación del contorno en tres direcciones posibles. Esto se logra con la convolución de cada píxel con una máscara específica. Cuando alguno de los tres puntos es ya un píxel de borde se entiende que el borde se ha cerrado, de lo contrario se elige el píxel con el valor máximo de gradiente y se marca como nuevo píxel de borde y se aplica nuevamente la convolución. Estos pasos se repiten para todo extremo abierto hasta encontrar su cierre o hasta llegar a cierto número de iteraciones determinado.

a)



b)

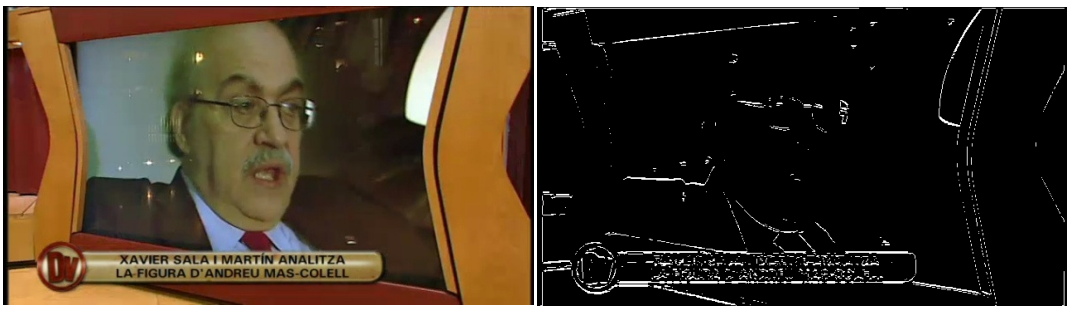


Figura 8: Resultado del filtro Canny aplicado a dos frames: a) imagen con fundido identificada como cambio de escena y b) primera imagen posterior sin fundido.

6. Detección de texto

Las herramientas de **reconocimiento de texto en la escena** se basan siempre en un primer paso cuya misión es determinar la ubicación aproximada del texto en la imagen. En este punto, cabe distinguir el texto artificial incrustado en un vídeo (como puede ser el nombre del conductor de un noticiario o el texto natural en la escena), del texto que aparece en una pancarta o un cartel que han sido filmados. Detectar el primer tipo de texto es más factible que detectar el segundo. Con posterioridad a la fase de localización del texto en la escena, se segmenta este texto (ya sea a nivel de frase, palabra o carácter) y se reconoce. El problema principal de este tipo de tecnología es la segmentación del texto. La gran mayoría de los sistemas conocidos no consiguen detectar el texto y, por tanto, tampoco consiguen segmentarlo. Ello es especialmente acuciante en el caso del texto natural. ¿Por qué esta dificultad? Básicamente, porque las imágenes habituales tienen una gran presencia de texturas semejantes a las del texto que dificultan enormemente la ubicación aproximada pero automática del texto. De esta manera, la gran mayoría de los sistemas incorpora un paso para marcar manualmente el área de la imagen aproximada donde está presente el texto (por ejemplo, el patrón de una imagen de público de un estadio de futbol puede confundirse fácilmente con los patrones de letras). Por supuesto, este funcionamiento no es aceptable en el ámbito de la gestión documental de material televisivo.

El objetivo de esta herramienta que conforma el extractor de KeyFrames es no sólo detectar qué imágenes contienen texto (principalmente texto artificial incrustado o texto en pastilla), sino escoger el frame con mayor cantidad de texto posible. No nos interesa simplemente captar un rótulo cuando aparecen las primeras palabras, sino que, ya que tenemos que escoger uno entre decenas o centenares de frames, que sea el que contenga la mayor información posible. Esta situación es típica en rótulos de texto que aparecen desde un extremo y se van desplazando a lo largo de la imagen.

6.1 Detección

El texto incrustado o en pastilla se puede describir como texto añadido dentro de una barra rectangular, con una cierta textura, alineado horizontalmente y que contrasta con el color de la propia barra. Estas características se traducen en dos tipos de descriptores: de textura y geométricos. Éstos son típicamente utilizados para *text candidate spotting* y *text characteristic verification* respectivamente. Las zonas texturizadas se pueden detectar utilizando análisis de wavelets. Por el alto coste computacional que requieren ambos procesos, puede utilizarse únicamente el primero.

La Transformada Wavelet es eficiente para el análisis local de señales no estacionarias y de rápida transitoriedad y, al igual que la Transformada de Fourier con Ventana, mapea la señal en una representación de tiempo-escala. Se preserva el aspecto temporal de las señales. La diferencia está en que la Transformada Wavelet provee un análisis en múltiples resoluciones con ventanas

dilatadas. El análisis de las frecuencias de mayor rango se realiza usando ventanas angostas y el análisis de las frecuencias de menor rango se hace utilizando ventanas anchas.

Sin embargo, esta aproximación produce bastantes falsos positivos (que tendrán que ser filtrados posteriormente utilizando descriptores geométricos) y algunas pérdidas en áreas poco contrastadas. Por otro lado, el método propuesto en *Region-Based Caption Text Extraction* [6] del Grupo de Procesado de Imagen de la Universidad Politécnica de Cataluña, combina el análisis de texturas mediante wavelets y el modelo de la imagen basado en regiones para las características geométricas.

6.1.1 Text Candidate Spotting

En una primera etapa, se analiza toda la imagen a nivel de textura en múltiples resoluciones utilizando la descomposición wavelet de Haar.

La descomposición de Haar, propuesta en 1909 por Alfred Haar, se considera como el primer wavelet conocido y el más simple posible. La desventaja técnica del wavelet de Haar es que no es continuo y por lo tanto es no derivable. Esta propiedad, de cualquier forma, es una ventaja para el análsis de señales con transiciones repentinas.

La función wavelet madre de las funciones de Haar ψ(t) puede ser descrita como

$$\psi(t) = \begin{cases} 1 & 0 \le t < 1/2, \\ -1 & 1/2 \le t < 1, \\ 0 & \text{De otra forma.} \end{cases}$$

y su función escalar φ(t) puede ser descrita como

$$\phi(t) = \begin{cases} 1 & 0 < t < 1, \\ 0 & \text{De otra forma.} \end{cases}$$

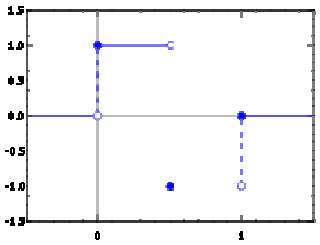


Figura 9: Wavelet de Haar

Tal y como se propone en [5], descriptores de textura como los coeficientes de la Transformada Discreta Wavelet (DWT) dan suficiente información para determinar dónde se sitúan en la imagen las áreas texturizadas. Se propone utilizar la potencia de las subbandas LH y HL en una transformada Haar aplicada sobre una ventana deslizante de tamaño HxW, siendo W mayor que H debido a la horizontalidad del texto:

$$P_{LH}^{l}(m,n) = \frac{1}{WH} \sum_{i=1}^{W} \sum_{j=1}^{H} LH^{l}(m+i, n+j)^2$$

donde l indica el nivel de descomposición y la expresión análoga para P^{l}_{HL}. La ventana se desliza con una superposición de la mitad del tamaño de la ventana en ambas direcciones. Se analizan ambas bandas ya que la potencia de la DWT en ventanas que contienen texto presentan valores altos (>T1) en alguna de las dos bandas y un valor destacable (>T2) en la otra. Así pues, los píxeles de una ventana son considerados como candidatos de pertenecer a texto si:

$$((P_{LH}^{l} > T_1) \wedge (P_{HL}^{l} > T_2)) \vee ((P_{LH}^{l} > T_2) \wedge (P_{HL}^{l} > T_1) \quad (1)$$

donde T1 y T2 son dos umbrales siendo T1 más restrictivo (T1>T2).

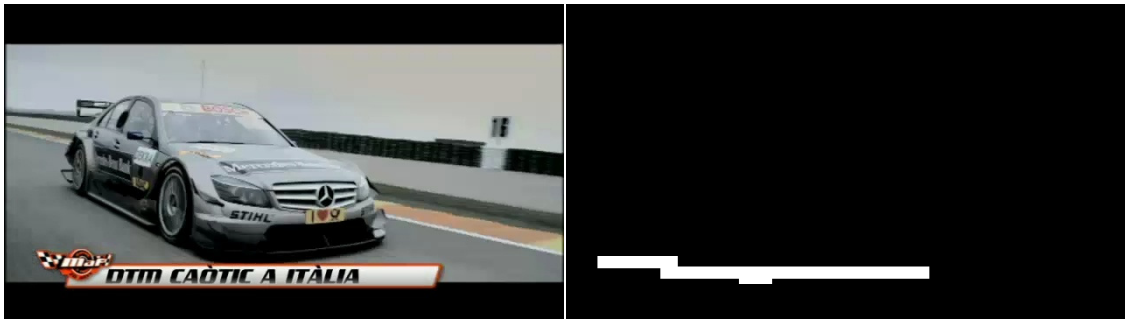
Con un modelo jerárquico de imagen basado en regiones (*Binary Partition Tree* [7]), se marcan las regiones de la BPT, asi como sus ancestros, si contienen píxeles candidatos. Esta acción es muy conservadora, pero en este paso prima más no tener pérdidas de posibles candidatos.

6.1.2 Text characteristic verification

Para cada nodo seleccionado, se estima una serie de descriptores para verificar si la región contiene texto en pastilla. Primero, se calcula un descriptor de textura basado en regiones de igual forma que en (1), pero sobre los píxeles interiores para evitar la influencia de los coeficientes de la DWT en los bordes de la región debido al gradiente. A continuación, se modifica el área de soporte de los nodos candidatos rellenando agujeros y realizando una apertura morfológica con un elemento estructurante pequeño (típicamente 9x9).

Finalmente, se establece la región conexa mayor como el área de soporte para calcular los descriptores geométricos: rectangularidad, relación de aspecto, altura, área y compacidad.

a) b)



c)

Figura 10: a) Ejemplo de imagen con texto en pastilla. b) Máscara resultado de la descomposición Haar. c) Superposición de imagen original y máscara.

7. Detección de caras

La detección y reconocimiento de caras humanas es una tecnología que funciona correctamente en entornos muy controlados (iluminaciones controladas y grupos reducidos de personas), o también en situaciones donde la interactuación humana es posible (anotación de álbumes personales). Sin embargo, los algoritmos de aplicación genérica actuales no son útiles en el ámbito televisivo (suelen producir un gran número de falsos positivos) y en la gran mayoría de sistemas se reduce su uso a contextos muy determinados (entrevistas).

La detección de caras es un caso específico de detección de clases-objeto, cuya principal tarea es encontrar la posición y tamaño de los objetos en una imagen perteneciente a una clase dada. Los algoritmos de detección en un primer momento se enfocaban a la detección de caras humanas frontales, pero hoy en día intentan ser mucho más genéricos detectando desde múltiples ángulos. Sin embargo, la detección de caras sigue siendo un difícil desafío debido a la gran variabilidad en tamaño, forma, color y textura de las caras humanas.

Nuestro objetivo en este caso, es crear un sistema que, para una escena, nos indique aquellas en las que hay presencia de caras frontales y cuantas.

Profundizando un poco más y a modo de valorar opciones, los métodos de detección de caras pueden clasificarse en las siguientes categorías [8,9]:

- Basados en el conocimiento (Knowledge-based): Estas técnicas se basan en reglas que codifican el conocimiento humano sobre la relación entre características faciales (Yang [10])
- Técnicas de características invariantes: consisten en encontrar caractarísticas estructurales que permanecen invariantes ante variaciones de postura e iluminación (Yow & Cipolla [11])
- Template matching: se basan en el uso de un patrón de caras estándar que puede ser predefinido manualmente o parametrizado mediante una función. Así pues, la detección consiste en calcular la correlación entre la imagen de entrada y el patrón (Yuille [12]).
- Basados en apariencia: en este caso, los modelos se generan entrenando una colección de imagenes y contienen las variaciones representativas de las clases de caras. Ejemplos de este tipo son: Neural Networks (Juell [13]), Support Vector Machine (Schulze [14]), Hidden Markov Models (Rabiner [15]) y Eigenfaces (Turk & Pentland [16])
- Basados en el color: estas técnicas se basan en la detección de píxeles que tienen un color similar al de la piel humana. Para ello se puede utilizar cualquier espacio de color (Zarit [17]: RGB, RGB normalizado, HSV, CIE-xyz, CIE-LUV, etc.).
- Adaptive Boosting (AdaBoost): el método consiste en la creación de un detector robusto a partir de un conjunto de clasificadores débiles (weak classifiers) para características locales contrastadas encontradas en posiciones específicas de la cara (Viola & Jones [18]).
- Aproximaciones basadas en video: este tipo de detectores explota la relación temporal entre frames, integrando detección y seguimiento en un mismo sistema. Así pues, las caras se detectan en una secuencia de video, en vez de una detección frame a frame (Lee [19]), (Qian [20]).

Obviamente estas categorias están interrelacionadas y pueden ser combinadas para mejorar las ratios de detección. El compromiso consiste en utilizar el máximo número de elementos

incorrelados posible sin penalizar el tiempo de computación. Normalmente el comportamiento se compara en términos de ratios de detección y ratios de falsas alarmas. Los errores típicos son:

- Falsos negativos: caras no detectadas correctamente, debidas a una ratio de detección baja.
- Falsos positivos: detección como cara de algo que no lo es, debido a una ratio de falsa alarma alta.

En este proyecto se ha utilizado el método AdaBoost de Viola&Jones [18] debido a su bajo coste computacional una vez entrenado el sistema.

7.1 Viola and Jones

Paul Viola y Michael Jones presentaron en el International Journal of Computer Vision de 2001 el método *Robust real-time object detection* para la detección de caras. El método se basa en unir clasificadores muy simples y agresivos en cascada, los cuales miran características faciales muy específicas. Esta técnica utiliza Adaptative Boosting (conocido como AdaBoost) y hereda la idea de clasificadores débiles Haar de la primera aproximación de Freund y Schapire [21].

La idea principal es que, si se escogen bien, la unión de estos clasificadores débiles (weak classifiers) generan un clasificador robusto. Por ejemplo, podemos tener un conjunto de 32 clasificadores de los cuales, los primeros, descartarían características extremas como fondos y formas de una cara, mientras que el resto escanearían la imagen buscando la posición de ojos y nariz, entre un tamaño máximo y mínimo. Estos parámetros deben ser elegidos con cierto cuidado, ya que la identificación global del sistema puede empeorar.

Las cascadas de clasificadores se obtienen tras un proceso de aprendizaje amplio de cara/no cara. Se necesitan miles de ejemplos de diferentes tamaños para obtener cascadas con ratios de falsos positivos bajas. Una vez pasada la fase de entrenamiento, el algoritmo es computacionalmente muy rápido. La principal desventaja es que la salida es completamente binaria: hay caras o no hay caras, sin proporcionar un valor de confianza o probabilidad de acierto, aunque sí podemos saber cuántas.

Figura 11: Ejemplo de detección de diversas caras en una imagen utilizando Viola&Jones.

8. Algoritmo

8.1 Puntuación basada en blurring.

Tal y como apuntábamos en el apartado 5, una manera de reconocer los frames con menor difuminado trataría de calcular la energía de los contornos de la imagen.

La implementación de esto se traduce en aplicar el filtro Canny a la imagen de entrada para obtener una imagen con los contornos detectados. La energía total de la imagen podemos interpretarla como la suma de los píxeles de esta máscara. Aplicando este concepto a todas las imágenes de una misma escena podemos obtener un valor numérico que representa la energía en cada una de ellas. Las imágenes que hayan dado como resultado un valor de energía mayor, serán las escogidas para representar la escena.

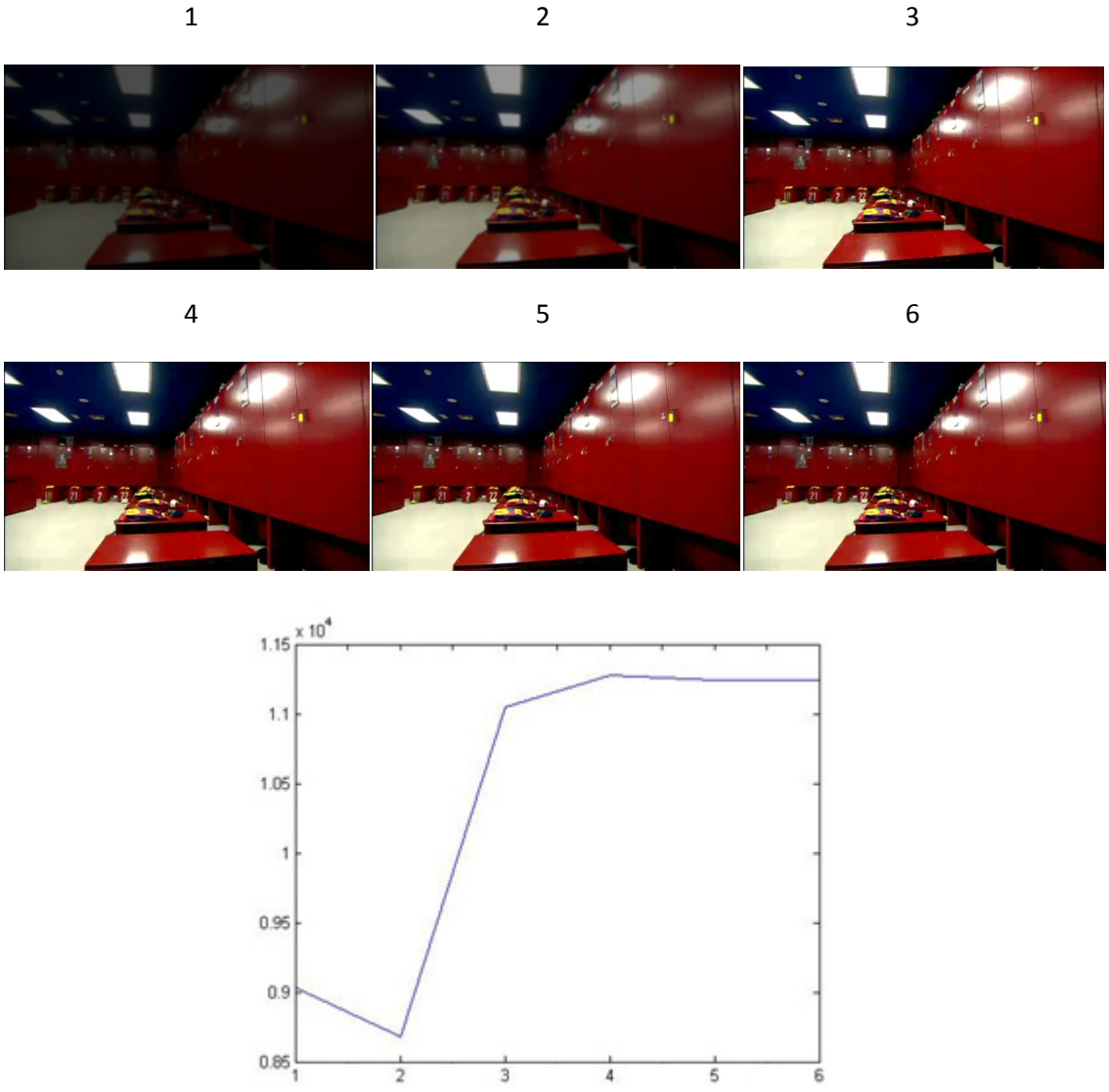


Figura 12: Secuencia de imágenes con una mejora progresiva en la nitidez. El gráfico muestra el aumento de la energía en los contornos.

Un valor mayor de energía puede ser debido a dos factores: el primero que, efectivamente, la imagen presenta mayor nitidez que el resto y el segundo que aparecen en la imagen un mayor número de objetos o texturas que contienen bordes, cosa que también lo podemos calificar como más representativo puesto que contiene más información visual.

La salida del bloque será por tanto la secuencia de frames que forman parte de la escena ordenados de mayor a menor energía de contornos. En el ejemplo de la figura anterior, la primera imagen seria la número 4.

8.1.1 Ejemplos

Los siguientes ejemplos muestran el análisis de los contornos a lo largo de la secuencia. Para cada uno se ilustra un gráfico que contiene la cantidad de energía de contornos para cada frame durante N frames. Sobre él se destacan algunas peculiaridades que son analizadas posteriormente. Las líneas verticales rojas delimitan las diferentes escenas.

8.1.1.1. Secuencia "Parlament"

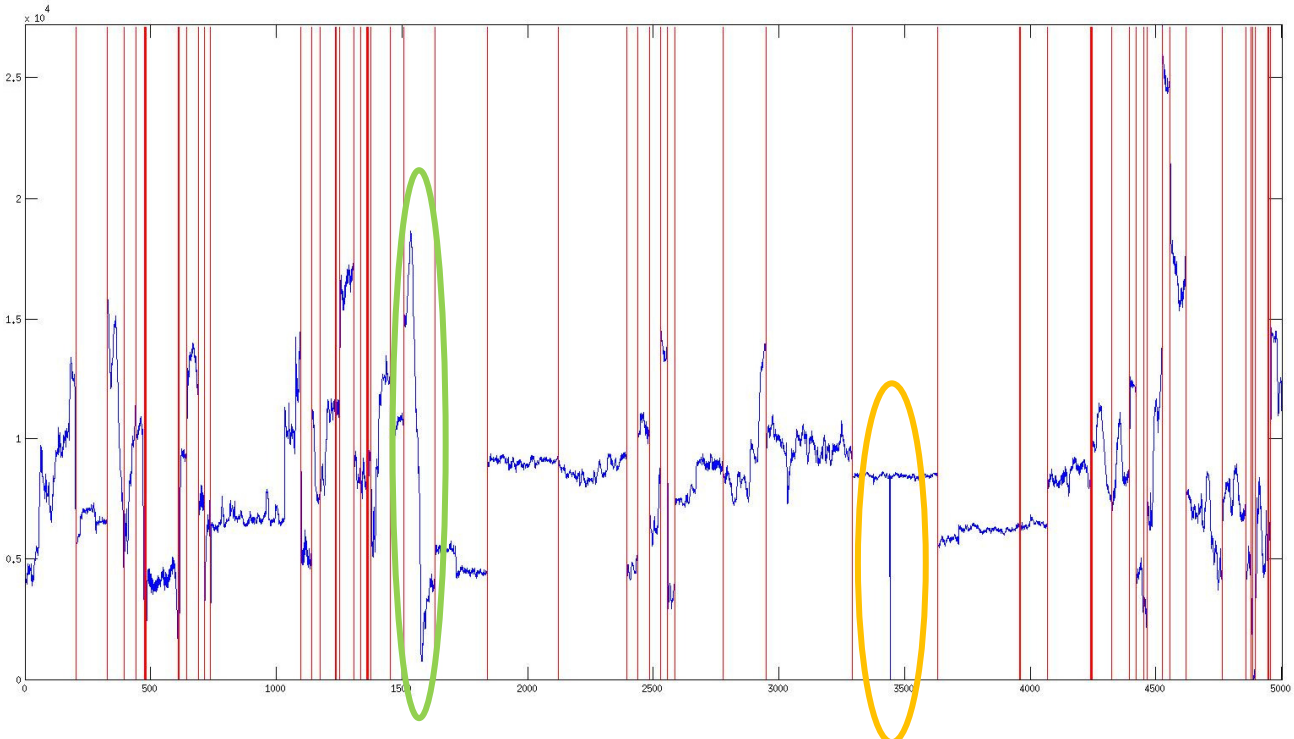


Figura 13: Energía de contornos en los 5000 primeros frames de la secuencia.

En la zona destacada en verde se produce una caída importante de la energía de contornos. Esto es debido a que durante la escena, el plano pasa de una zona luminosa a una zona oscura, en la que es difícil apreciar detalles, por lo que tendrá menor relevancia.

Figura 13.1: Detalle de la causa del desvanecimiento de la energía: la imagen se oscurece.

En la zona destacada de color naranja, aparece una caída repentina y una posterior recuperación de la energía. Análizandolo con detalle, se produce cuando se han unido con un frame blanco dos secuencias correspondientes a la misma escena (distintas declaraciones). El detector de cambios de escena acertadamente ha estimado que es la misma, ya que ha estimado que el frame blanco es un destello.

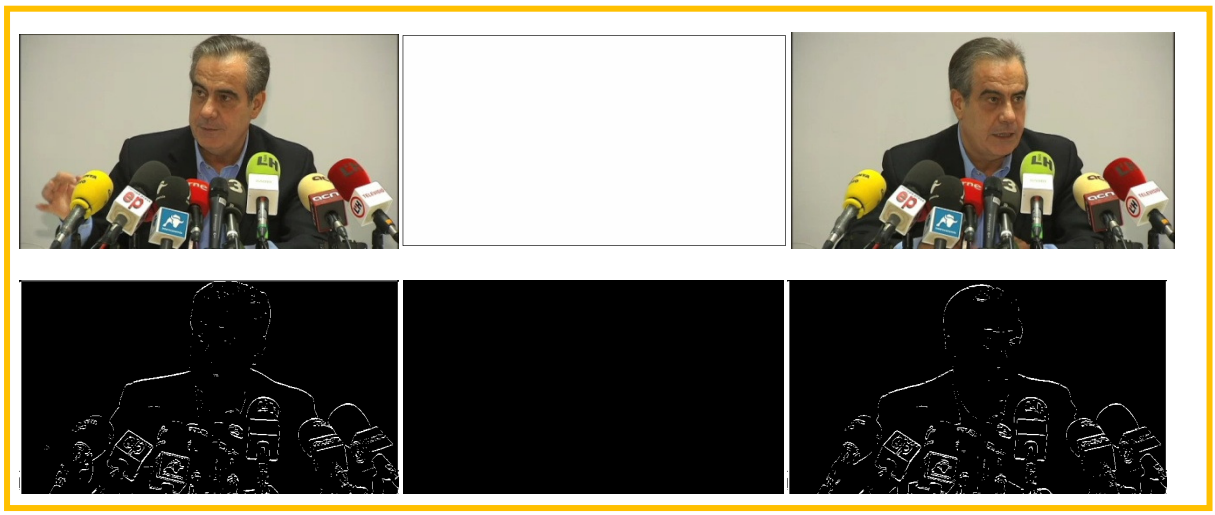


Figura 13.2: Detalle de la caída brusca de energía: corte entre diferentes declaraciones.

8.1.1.2. Secuencia "España Directo"

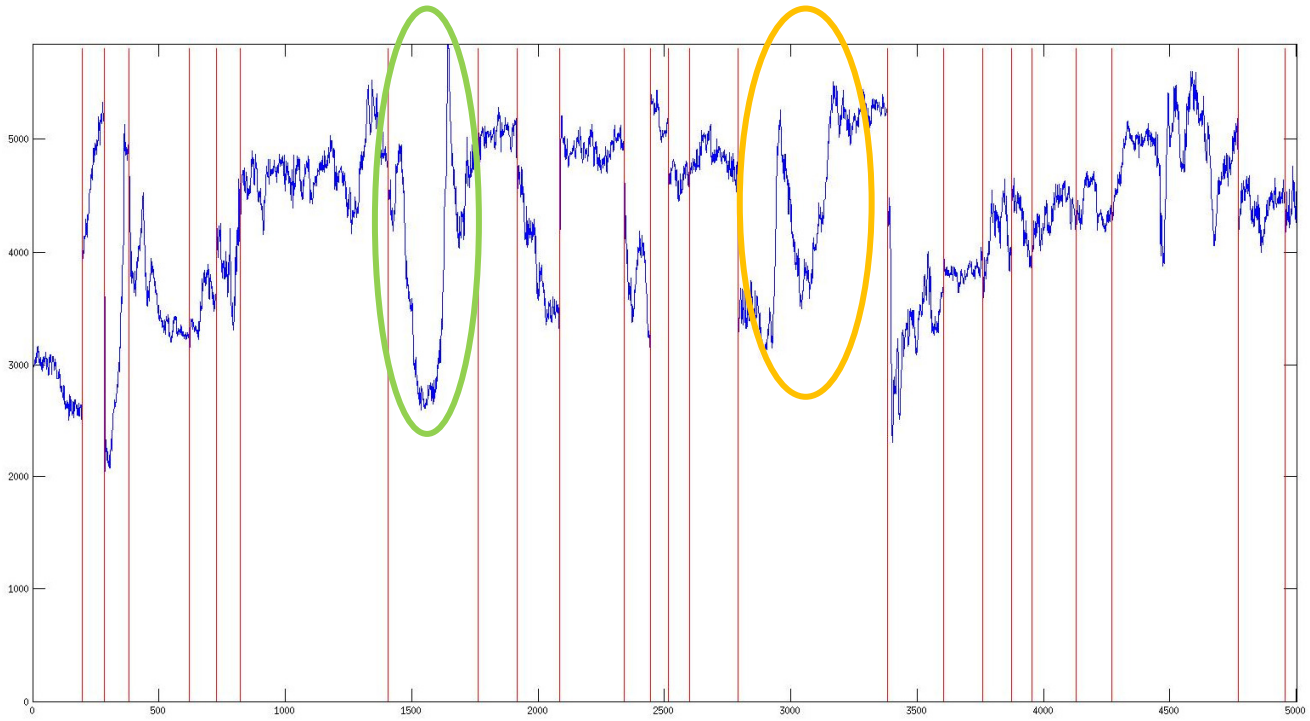


Figura 14: Evolución de la energía de contornos en la secuencia.

Durante esta secuencia se mantiene en todo momento el mismo escenario y la cámara se va desplazando por él.

En el primer caso, de color verde, la cámara da un cuarto de vuelta alrededor de los protagonistas para mostrar lo que tienen delante y retorna a su posición inicial. Los planos con ambos protagonistas se caracterizan por una mayor energía. Sin duda los planos en los que aparecen más detalles contendrán más energía, lo que nos dará una mejor puntuación a frames semánticamente más importantes.

Figura 14.1: Ejemplo de rotación en la escena y vuelta a la posición inicial.

En el segundo caso, de color naranja, la cámara parte de un plano lejano y acaba enfocando a los protagonistas iniciales. Seguimos obteniendo una mayor energía cuando se enfoca a los protagonistas.

Figura 14.2: Ejemplo en el que se centra la atención en algo lejano y progresivamente se mueve hasta un primer plano.

8.1.1.3. Secuencia "Pit Stop"

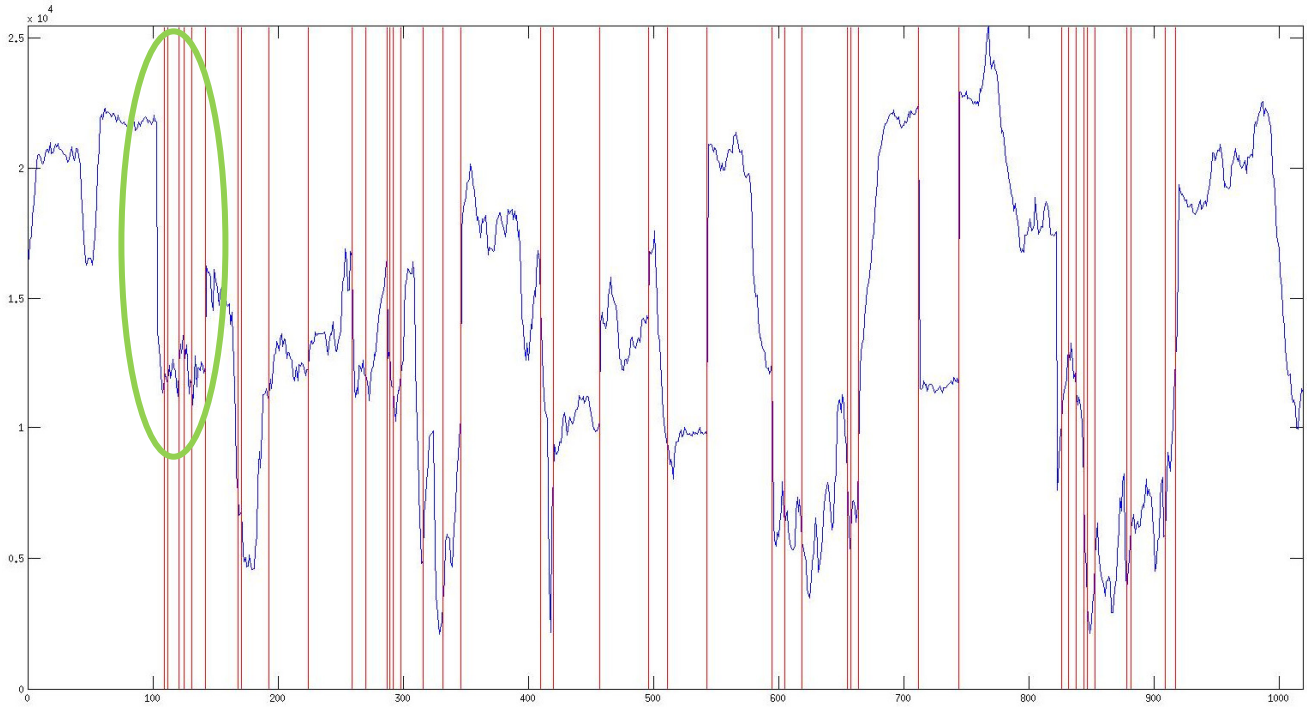


Figura 15: Evolución de la energía de contornos en la secuencia.

En este caso se muestra la dependencia con la presencia de texto: el texto se desvanece y decae la energía. Posteriormente vuelve a aparecer un nuevo texto y la energía vuelve a aumentar.

Esta característica nos ayudará a puntuar mejor frames que contienen texto.

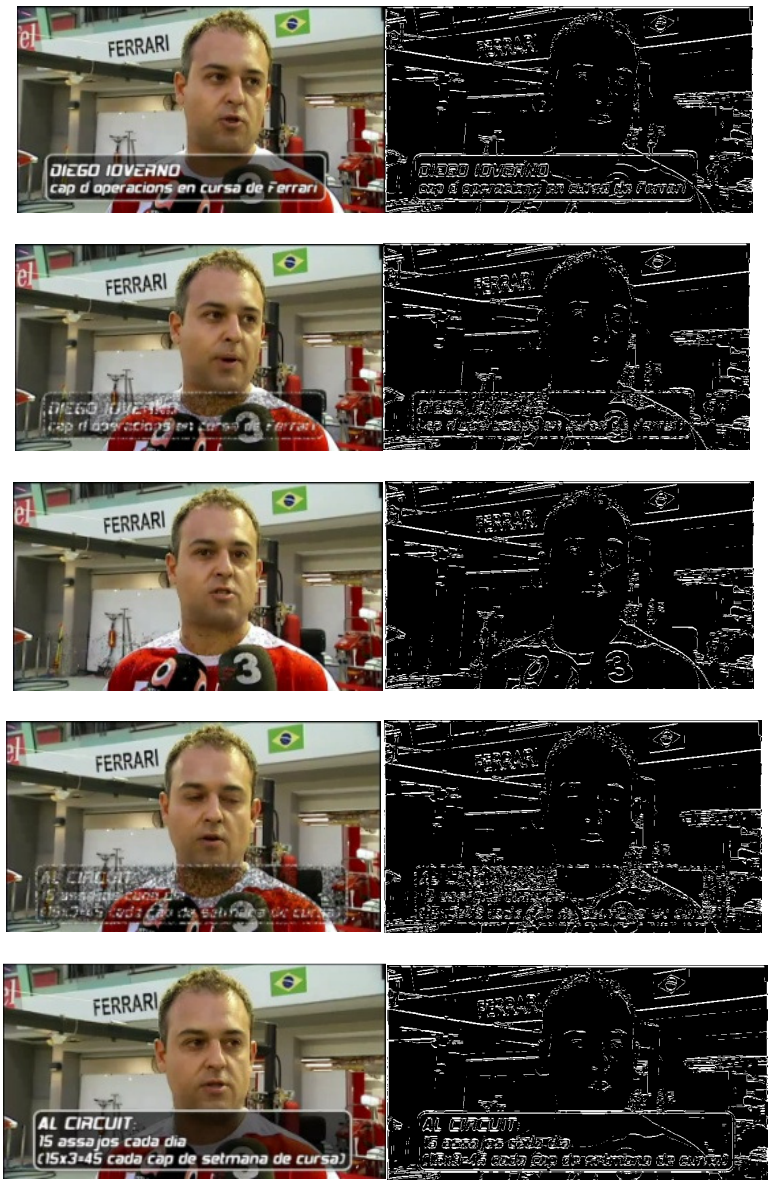


Figura 15.1: Ejemplo de desvanecimiento de energía causada por la desaparición y la posterior aparición de texto en pastilla.

8.2 Puntuación basada en presencia de texto.

Como hemos visto en el bloque de detección de texto basado en la wavelet de Haar, el resultado en imágenes naturales genera muchos falsos positivos. El primer paso para diseñar un sistema que nos devuelva un valor de confianza sobre la presencia de texto y de la cantidad de texto presente, será reducir al máximo estos falsos positivos. Sin embargo, algo positivo es la baja probabilidad de no detección, ya que consideramos peor el hecho de que el texto no se detecte.

Para reducir la cantidad de falsos positivos y tener así una medida fiable de cuánta presencia de texto hay (nos interesa extraer la imagen con más texto), deberemos analizar la redundancia temporal que genera el texto gracias a su estaticidad.

Durante una secuencia en la que tenemos texto visible, lo más lógico es que se muestre un tiempo prudencial para que el usuario pueda leerlo o bien, si se trata de una imagen natural en la que el texto no forma parte del foco de atención, al encontrarse durante la misma escena lo detectaremos durante diversos frames consecutivos. Por otro lado, zonas detectadas como texto pero que en realidad se tratan de zonas con texturas y geometría que confunden a nuestro detector, al realizar pequeñas variaciones en el ángulo de visión o en la distancia del plano, es probable que se reduzca la zona o bien se dejen de detectar.

Figura 16: Secuencia en la que aparece texto en pastilla por la derecha. Debajo, las zonas detectadas con la descomposición Haar.

Nuestro sistema consistirá en un “mapa de calor” en el que se mostrará a lo largo de una misma escena, las zonas donde se ha detectado un posible texto. Al final de la misma, debido a la estaticidad del texto, quedarán como “zonas más calientes” las más repetidas durante la escena. Por el contrario, detecciones momentáneas o la no detección en esas zonas, quedarán como “zonas frías”, pudiéndolas descartar.

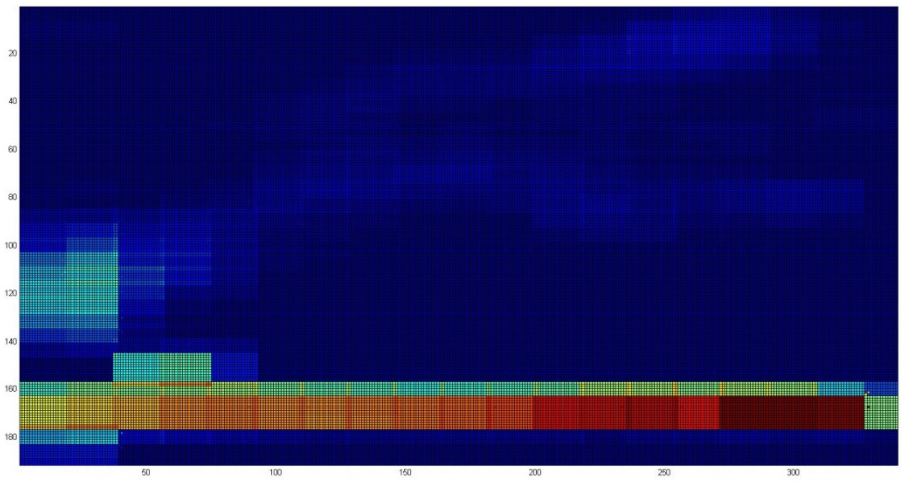


Figura 17: Mapa de calor de la secuencia anterior.

Una vez generado nuestro mapa de calor de la escena, lo utilizaremos para la construcción de una máscara en la que solo tendremos las zonas más calientes. Para ello, buscaremos la zona que más se ha repetido (la de mayor calor) y aplicando un cierto umbral por debajo de su valor (alrededor de un 25%), binarizaremos la imagen. De esta forma, obtendremos una máscara con las zonas que se han repetido en un 75% de veces respecto a la que más ha aparecido como candidata a texto. Estas son las regiones que tienen mayor probabilidad de contener texto.

Figura 18: Máscara modelo generada.

Llegados a este punto, tan solo nos hace falta saber, como para cada bloque, cuales son los frames que nos representan mejor la escena. La respuesta es sencilla: serán aquellos que contengan, en la mayor parte posible, las mismas zonas que han quedado definidas en nuestra máscara a la que llamaremos modelo. Para buscar estos candidatos, miraremos las máscaras de cada uno y únicamente nos centraremos en las zonas coincidentes con la máscara modelo. Por tanto, aquellos que presentaban posibles detecciones de texto en zonas donde luego no se han vuelto a repetir,

quedan descartadas para el estudio. Además, de las zonas coincidentes miraremos cuanto coinciden, dando la máxima puntuación a aquellas que contienen completamente la máscara modelo (y quizá zonas adicionales) y menor puntuación cuanto menor sea la coincidencia.

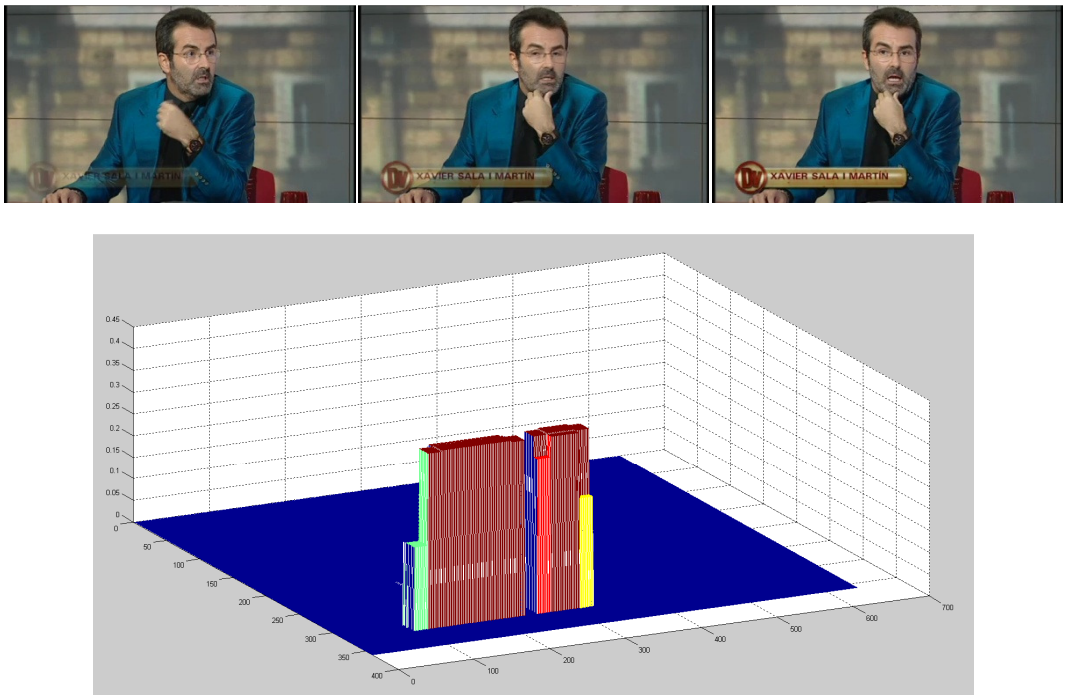


Figura 19: Vista en 3D del “mapa de calor”.

Figura 20: Máscara modelo

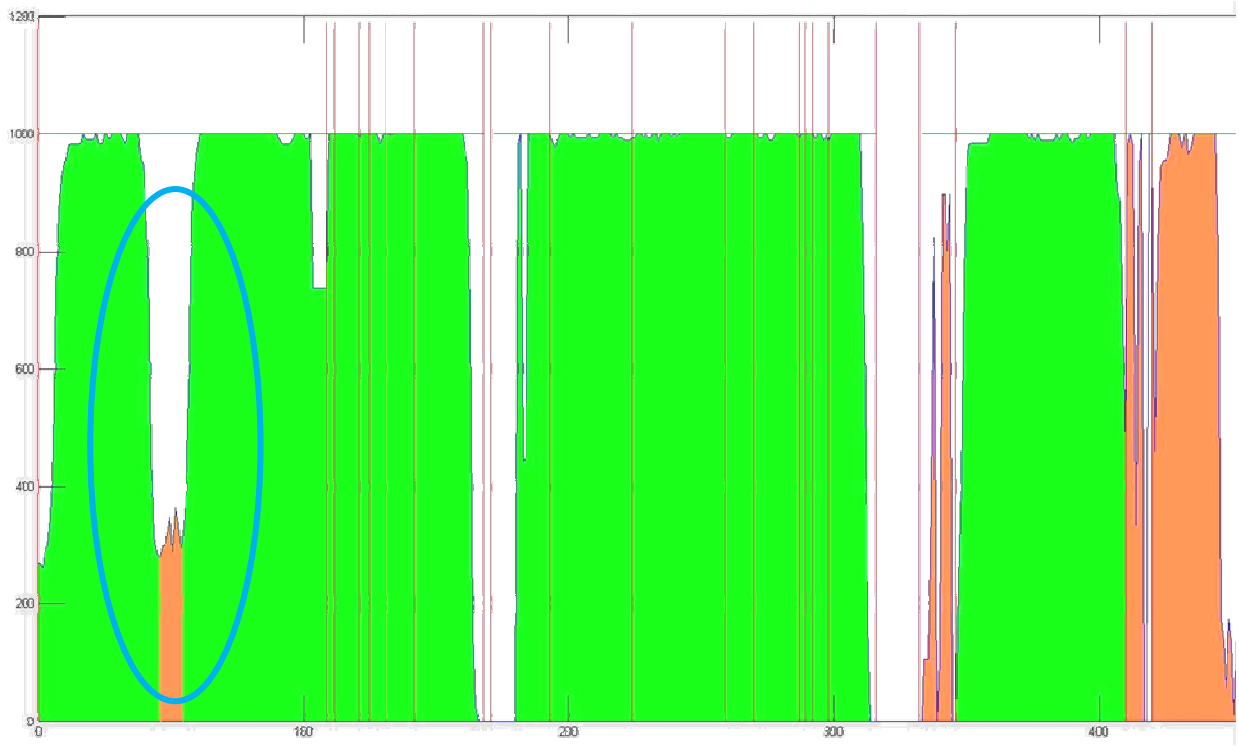
El resultado del bloque es la obtención de un vector en el que tendremos de forma ordenada los frames de mayor a menor puntuación según el valor de confianza (coincidencia) de la presencia de texto.

8.2.1 Ejemplos

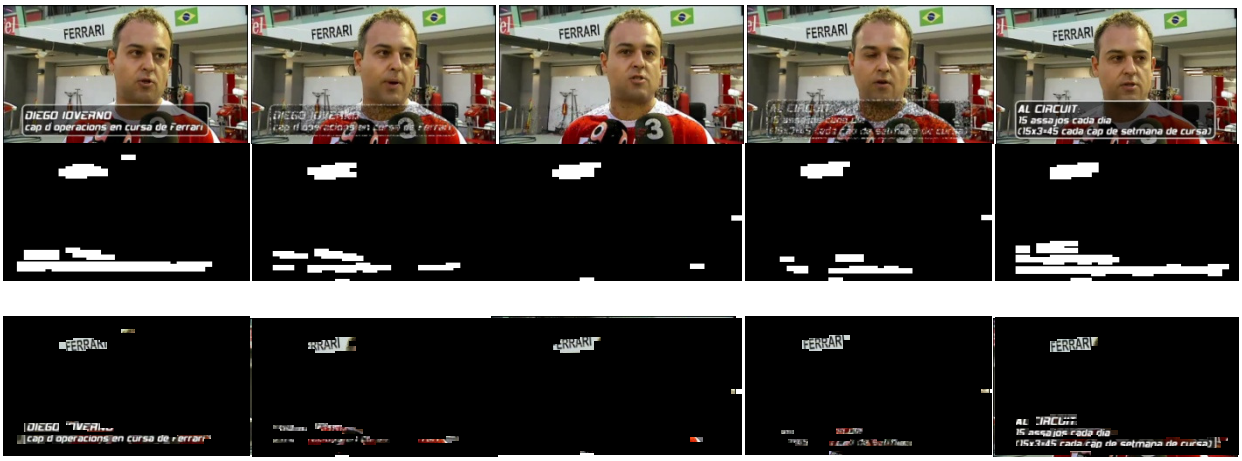
En los ejemplos que se detallan a continuación, se muestra un gráfico con el valor de confianza de la presencia de texto para cada frame. Este valor está normalizado a 1000, que representa el valor máximo. De esta forma se pueden apreciar tanto escenas con mucho como con poco texto.

En verde podremos ver frames en los que realmente hay texto, mientras que en naranja, frames normlamente con falsos positivos. La importancia reside en detectar qué frames contienen más texto, es decir, valores cercanos a 1000.

8.2.1.1. Secuencia "Pit Stop"



En la zona destacada, como ya habíamos visto en el ejemplo de blurring, el texto se desvanece y vuelve a aparecer. En este caso sigue habiendo texto, la palabra "Ferrari", por lo que no se trata de un falso positivo, como podemos ver en las imágenes superpuestas de abajo.



8.2.1.2. Secuencia "Motor a fons"

En esta secuencia se pueden observar dos ejemplos. Uno, en azul, marca una escena destacada en color rojo que corresponde a texto no detectado. El otro, en violeta, muestra el detalle de esos curiosos escalones.

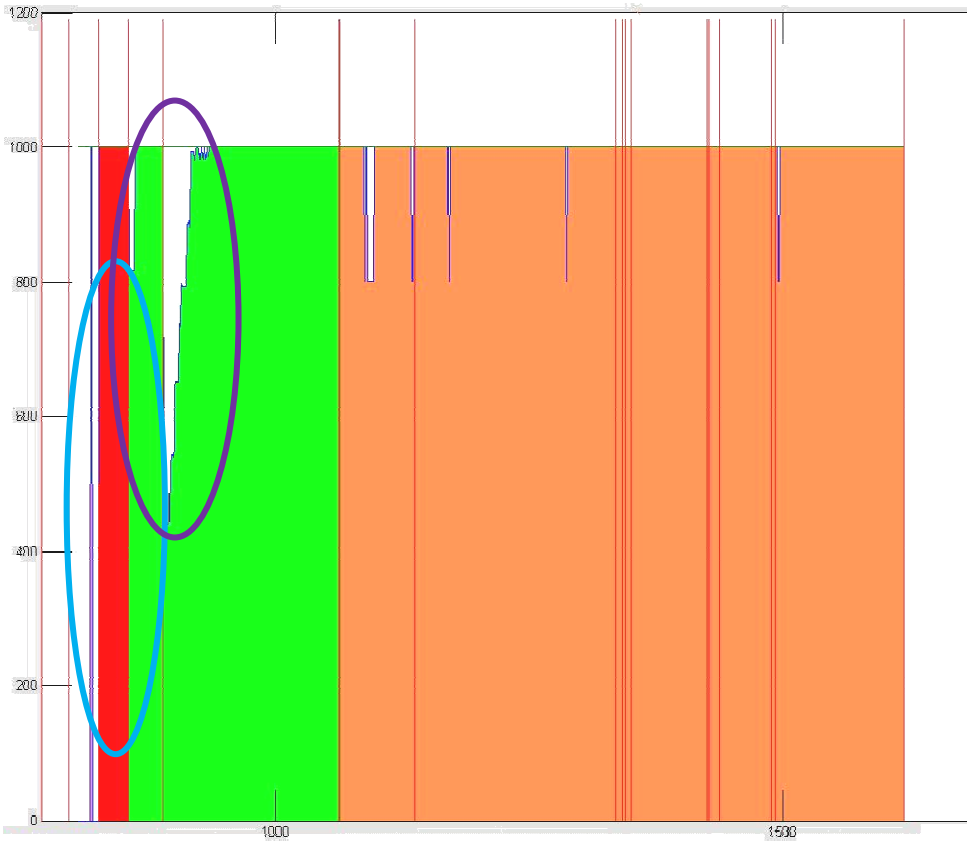
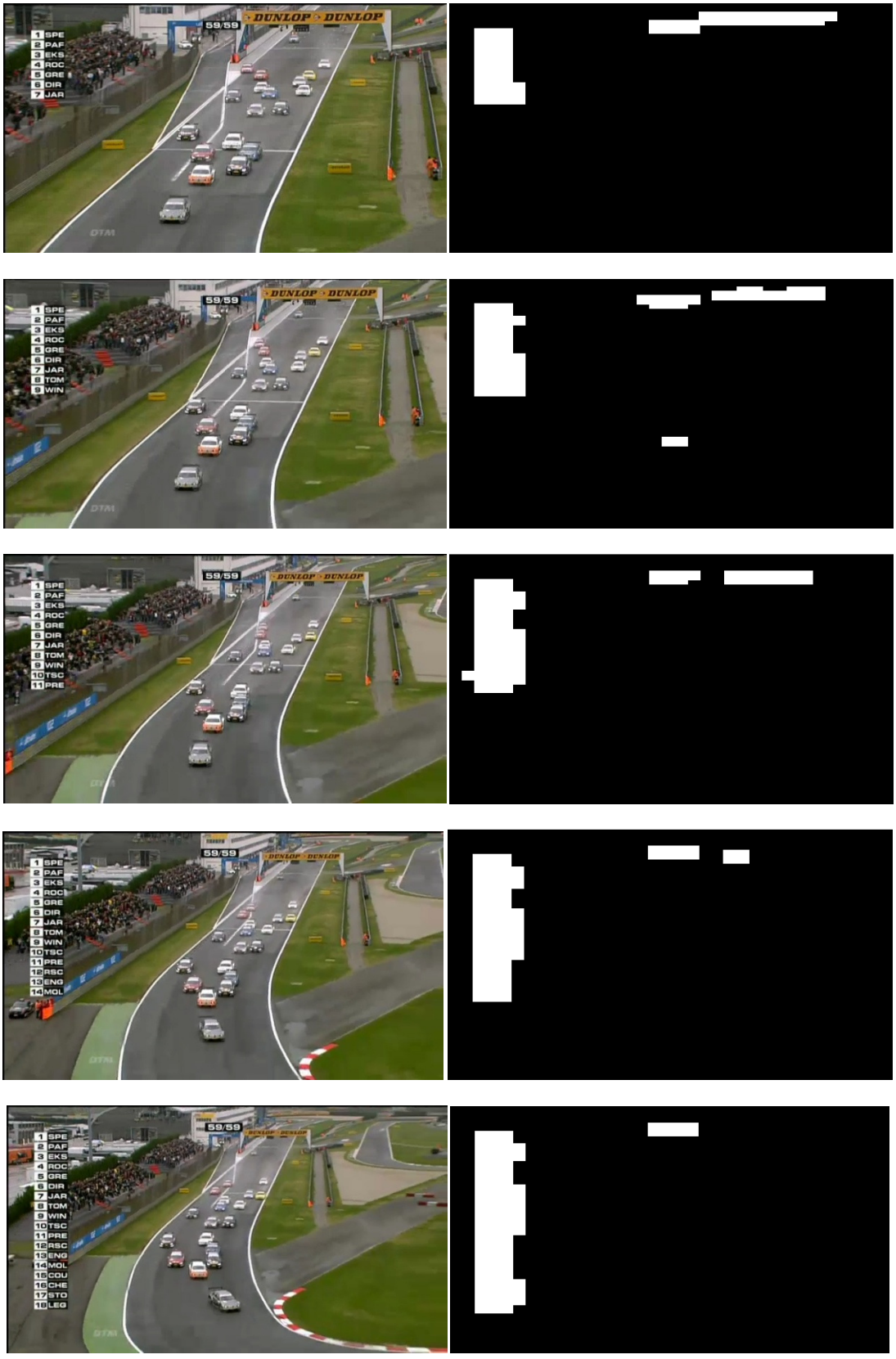


Figura 21: Evolución del tamaño de area de texto.

El caso del texto no detectado es discutible. No es un texto en pastillas o con un fondo bien diferenciado del texto. Se trata de una especie de logotipo de un tamaño considerable. Sin embargo parece que es suficientemente relevante como para ser detectado.

Figura 22: Ejemplo de texto no detectado. A la izquierda, imagen original. A la derecha, máscara de texto.

Por último, en este ejemplo se destaca la aparición de esos “escalones” en el gráfico de texto. Se deben al indicador con las posiciones que aparecen a la izquierda. A medida que los coches pasan por la meta, aparecen sus nombres en el marcador, por lo que se detecta una mayor cantidad de texto. Como siempre, nos interesa aquella imagen que contenga la máxima información posible. Ésta es la que contiene el nombre de la mayor parte de los participantes.



8.2.1.3. Secuencia "Telenotícies"

Esta secuencia pertenece a imágenes de noticiario informativo. En ellas es típico ver texto que aparece de un lado a otro de la pantalla. En el gráfico se puede ver aumentos en forma de rampa que provienen de este hecho.

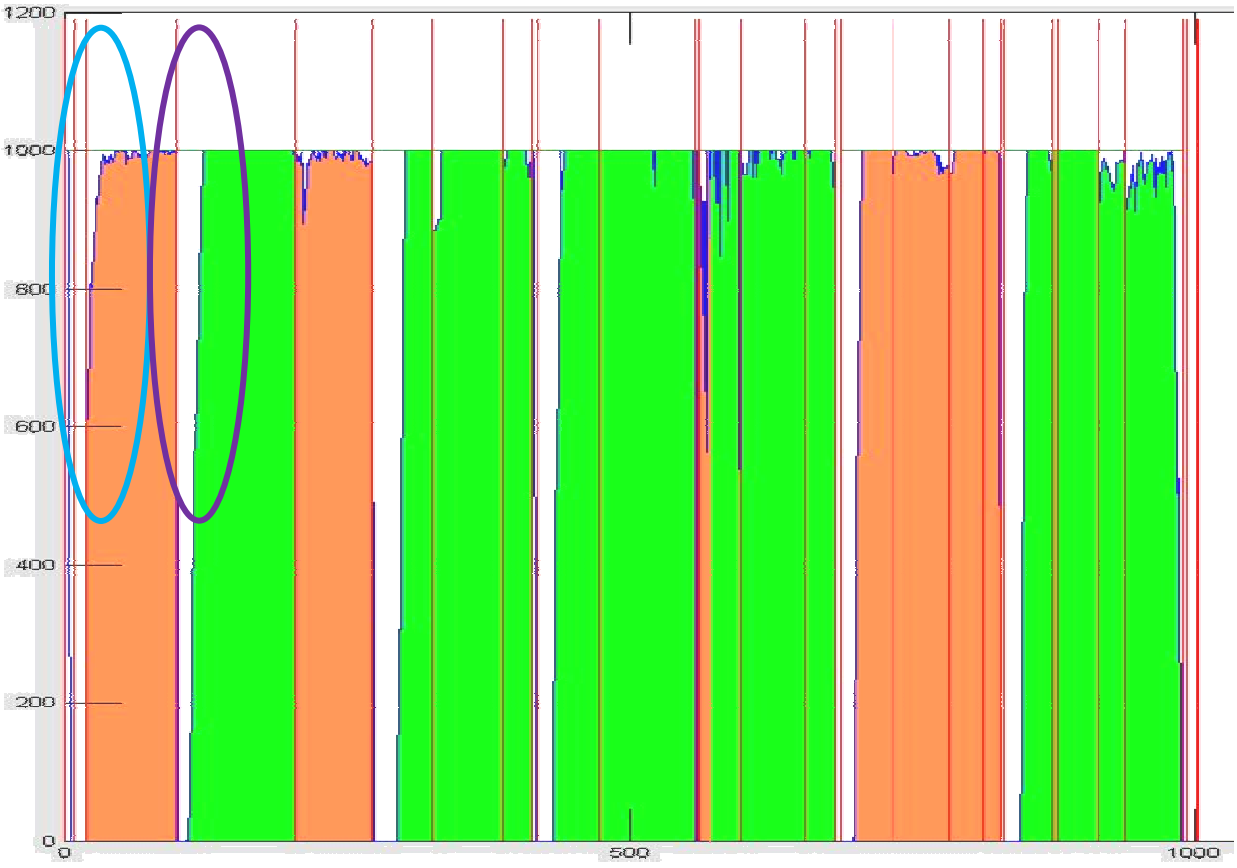
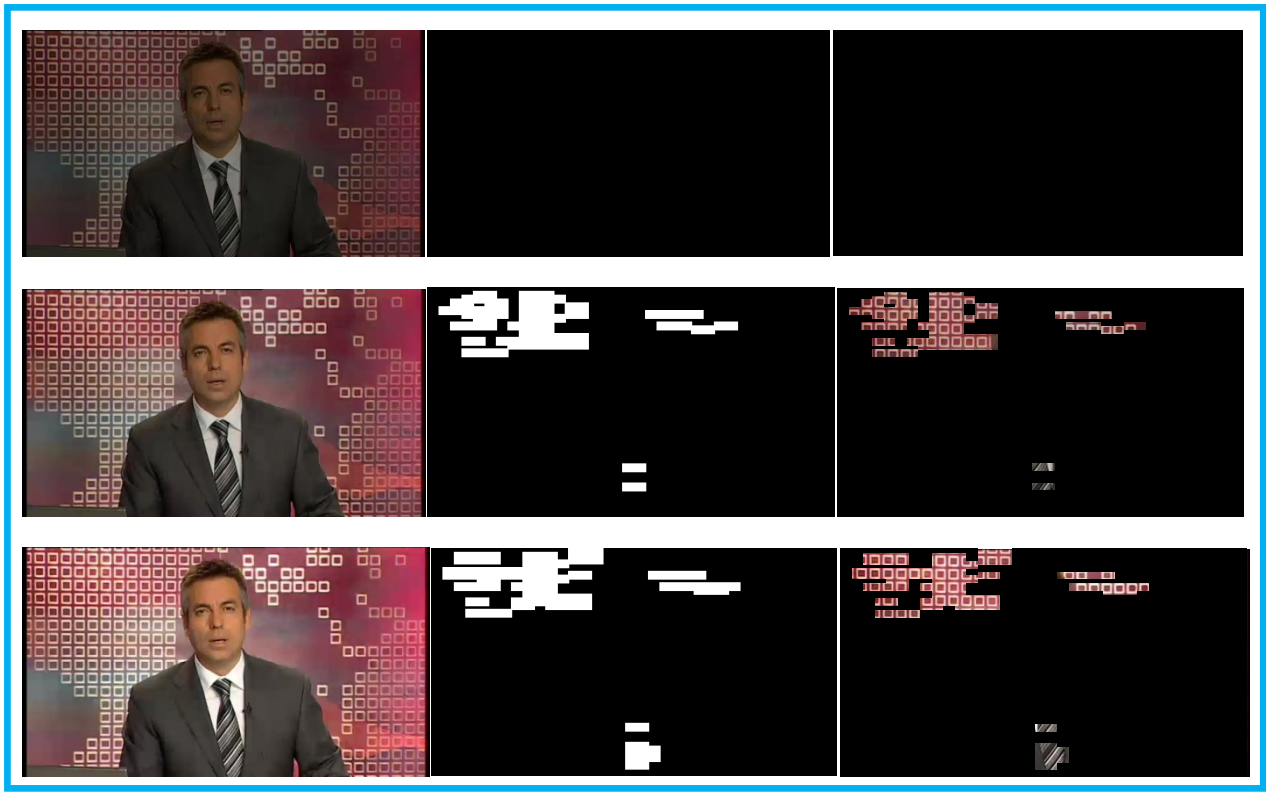


Figura 23: Evolución del tamaño de area de texto.

Un ejemplo de lo que comentamos:

En esta secuencia los falsos positivos (zonas en naranja) corresponden al peculiar fondo que poseen los estudios de TV3.



8.3 Puntuación basada en la detección de caras.

El método de Viola&Jones como hemos visto en el apartado 7.1, nos proporciona una salida "binaria" en cuanto a la detección de caras. Es decir, no tenemos un valor que nos represente una probabilidad o confianza de que una región sea una cara. Esto provoca además que sea difícil descartar falsos positivos.

Tampoco podemos utilizar la misma técnica aplicada en el apartado anterior ya que una cara frontal, que son las que queremos detectar, no es tan estática como el texto. Pensemos por ejemplo un debate o una asamblea, en los que es habitual enfocar a la persona que tiene la palabra y ésta no mira frontalmente a la cámara como si fuese un informativo, sino que baja la cabeza para mirar el papel, mira a los asistentes, etc.

Esto condiciona nuestro sistema final ya que nos obliga a ser más drásticos en la selección o descarte de frames.

No obstante, tenemos que tener en cuenta que la detección de caras se producirá con mayor probabilidad en imágenes más claras, es decir, con menos blurring y por tanto, también detectará mejor texto si hay. Parece entonces justificado que si un frame contiene una o varias caras sea seleccionado junto al resto de frames de la escena que también indican la presencia de caras como candidatos a keyframe.

8.3.1 Ejemplos

A continuación se muestran diferentes resultados de la detección de caras.

Figura 24: Detección correcta y falso positivo.

Figura 25: Las caras giradas no son detectadas.

Figura 26: Caras detectadas a pesar de presentar oclusiones parciales. Nótese que el giro de las caras es de tipo “yaw”, que son detectadas mejor que las de tipo de giro “roll”.

Figura 27: Falso positivo

8.4 Sistema final

Nuestro sistema extractor de KeyFrames tiene que ser capaz de seleccionar, para cada escena, el mejor frame. La cuestión es: ¿cúal es el mejor?. Hasta ahora tenemos diferentes bloques que toman un único criterio cuantificable para seleccionar un frame:

- Tenemos un bloque de blurring que nos especifica la energía de cada frame. Por tanto, podemos establecer un orden de prioridad siendo los de mayor energía más prioritarios.
- El bloque de texto selecciona aquellos frames que contienen más texto discriminando zonas que probablemente son falsos positivos. Podemos ordenar también, de mayor a menor, los frames con mayor área de texto.
- Nuestro detector de caras indica para cada frame cuántas caras hay, sin tener un valor de confianza.

¿De qué forma consideraremos que un frame es mejor que otro en base a los tres criterios a la vez?

Cada bloque es independiente y capaz de establecer un orden de preferencia. Sin embargo, la unión de los tres no es algo trivial. El caso ideal sería que un mismo frame fuese el de máxima energía de contornos, el de máxima área de texto y el que más caras contiene. No obstante, teniendo en cuenta que cada escena puede tener centenares de frames, es poco probable.

Es necesario establecer un cierto criterio de selección:

1. Como nuestro detector de caras tiene una salida binaria (hay caras / no hay caras), será nuestro primer filtro. Consideramos entonces que son más importantes las imágenes que contienen caras y además, frontales.
   Durante una misma escena, es habitual que o toda contenga caras o no contenga ninguna, por lo que en realidad estamos seleccionando las caras que presentan características más claras. De esta forma podemos utilizarlo en el futuro para un posterior reconocimiento de caras.

2. Para las imágenes que contienen caras (o todas, si no se han detectado en ningún frame), se busca las que mayor contenido de texto tengan.
   Debido a la estaticidad del texto, es probable que se encuentren numerosos frames con un valor similar de area de texto, por lo que acabaremos de decidirnos en el siguiente filtro.

3. Dentro del conjunto de frames con valor similar del tamaño del área de texto (válido también para cuando no se ha detectado texto en ninguna imagen), la última palabra en el proceso de elección la tendrá el bloque de blurring. Escogeremos aquella que contenga mayor energía de contornos.
   Como ya hemos visto, una imagen con mayor energía nos proporcionará un frame más nítido. Además, si éste contiene caras, texto o ambos, estaremos escogiendo el que probablemente nos pueda dar mejores resultados para un posterior reconocimiento de caras o texto.

La ventaja de este modelo jerárquico es que podemos suprimir cualquier bloque que no nos interese analizar, por ejemplo si no queremos detectar texto, sin que se altere la estructura.

Además, cualquier mejora en la detección tanto de caras como de texto o contornos, repercutirá en una mejor elección de KeyFrames sin modificar ningún parámetro.

9. Resultados

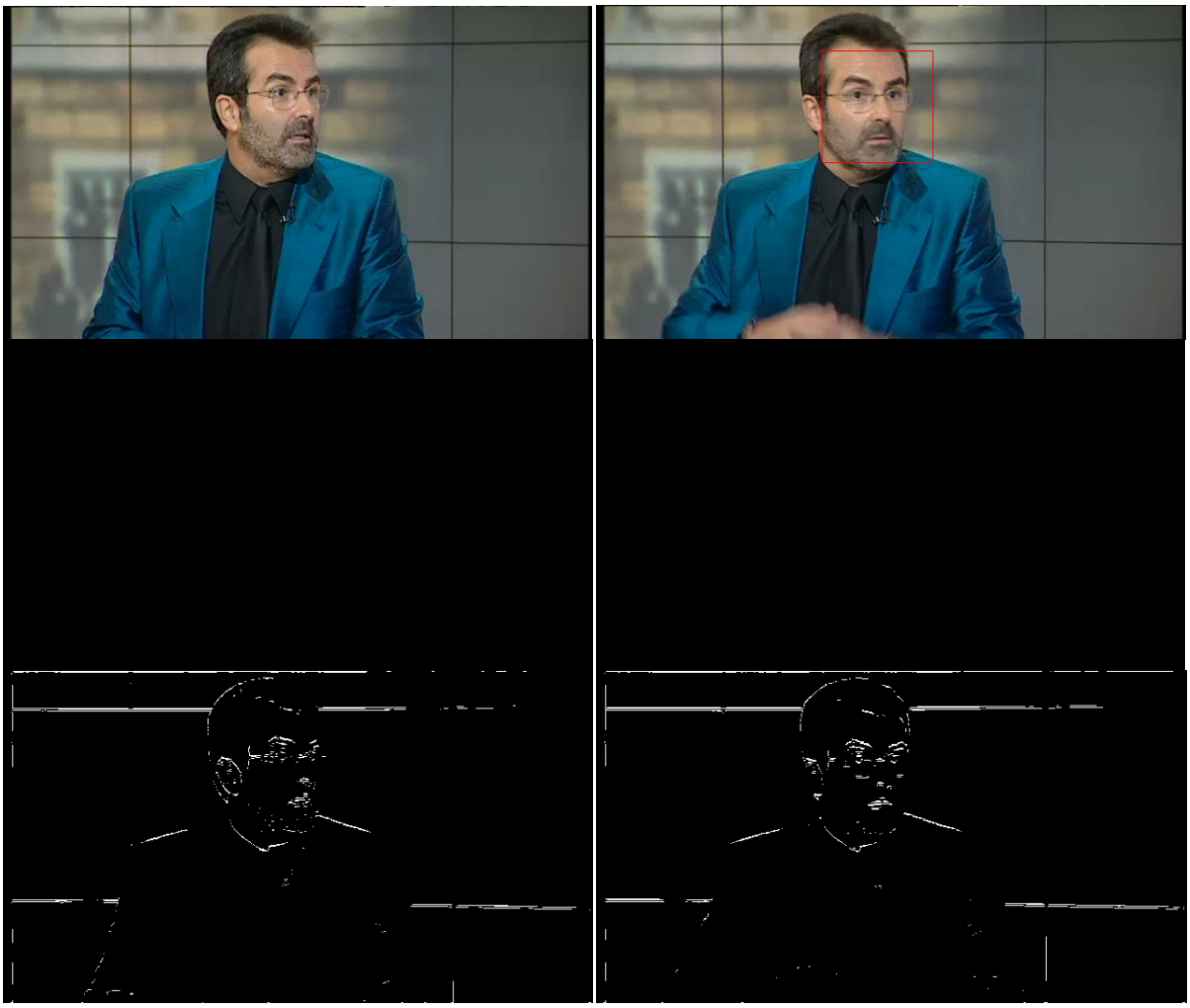
En este apartado se muestran diferentes ejemplos en los que comparamos el sistema que se estaba utilizando (un detector de cambios de escena) con nuestro sistema de extracción de KeyFrame.

Para cada ejemplo veremos: a la izquierda el resultado actual y a la derecha el resultado del extractor de KeyFrames. Además, debajo de las imágenes originales, podremos ver la máscara de texto y la detección de contornos de cada imagen.

9.1 Secuencia "Divendres"

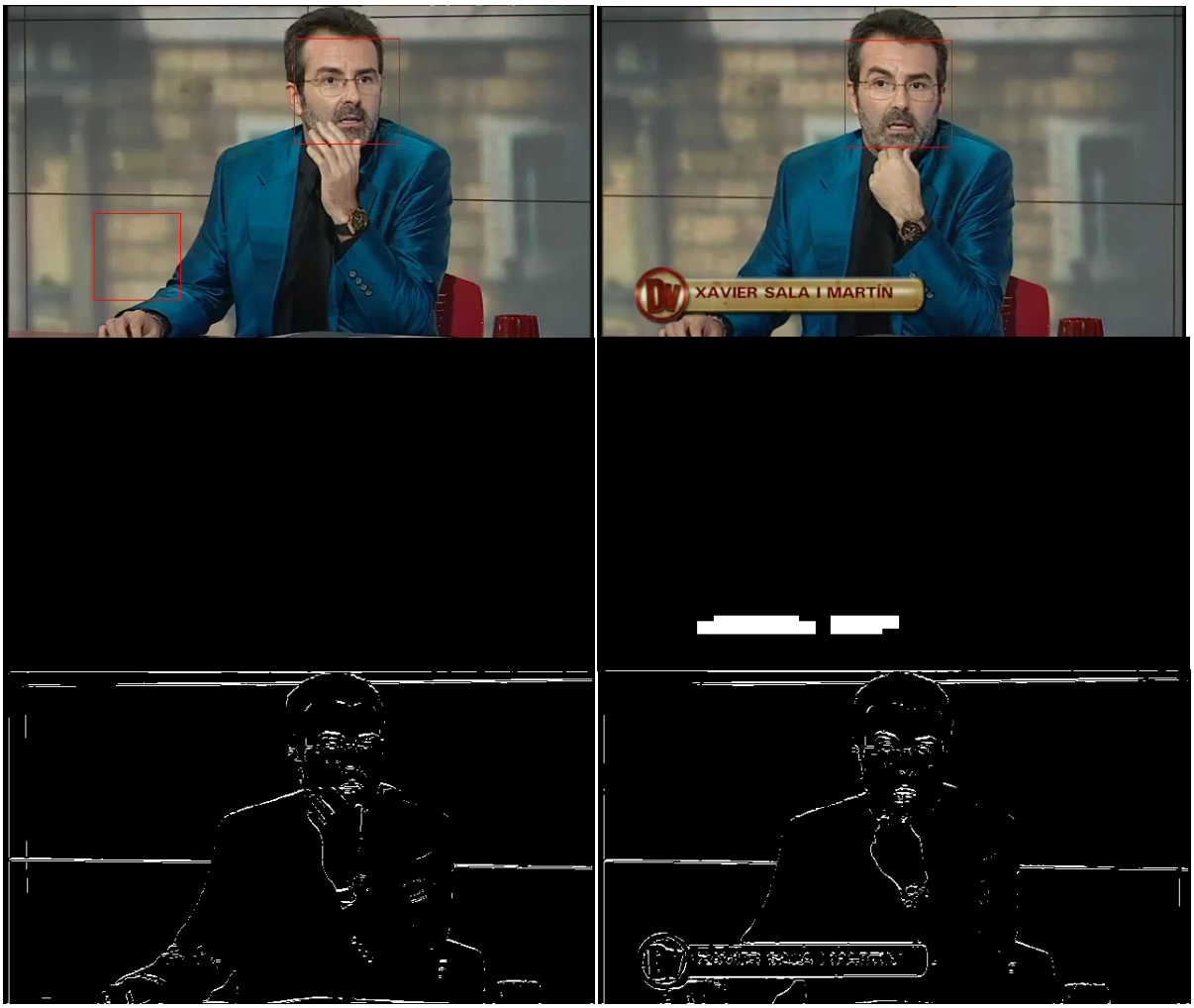
Esta secuencia contiene un programa de tertulia en la que intervienen diferentes personajes.

En el primer ejemplo vemos que la escena contiene el primer plano de una persona. No obstante, contiene imágenes en las que aparece de forma frontal, las cuales son más interesantes. Además, no hay presencia de texto, por lo que la decisión final se tomará en función de la nitidez de la imagen.



En este ejemplo podemos ver cómo las caras de perfil no son detectadas pero sí la cara que aparece en una pantalla al fondo. Ambas imágenes la contienen por lo que la presencia o no de texto y el posterior análisis de borrosidad genera una elección de KeyFrame diferente. Como detalle, se pueden ver diversas zonas de falsos positivos de texto.

A continuación, ambas imágenes contienen caras, incluso con un falso positivo. Sin embargo, el frame escogido contiene texto, por lo que es más relevante.

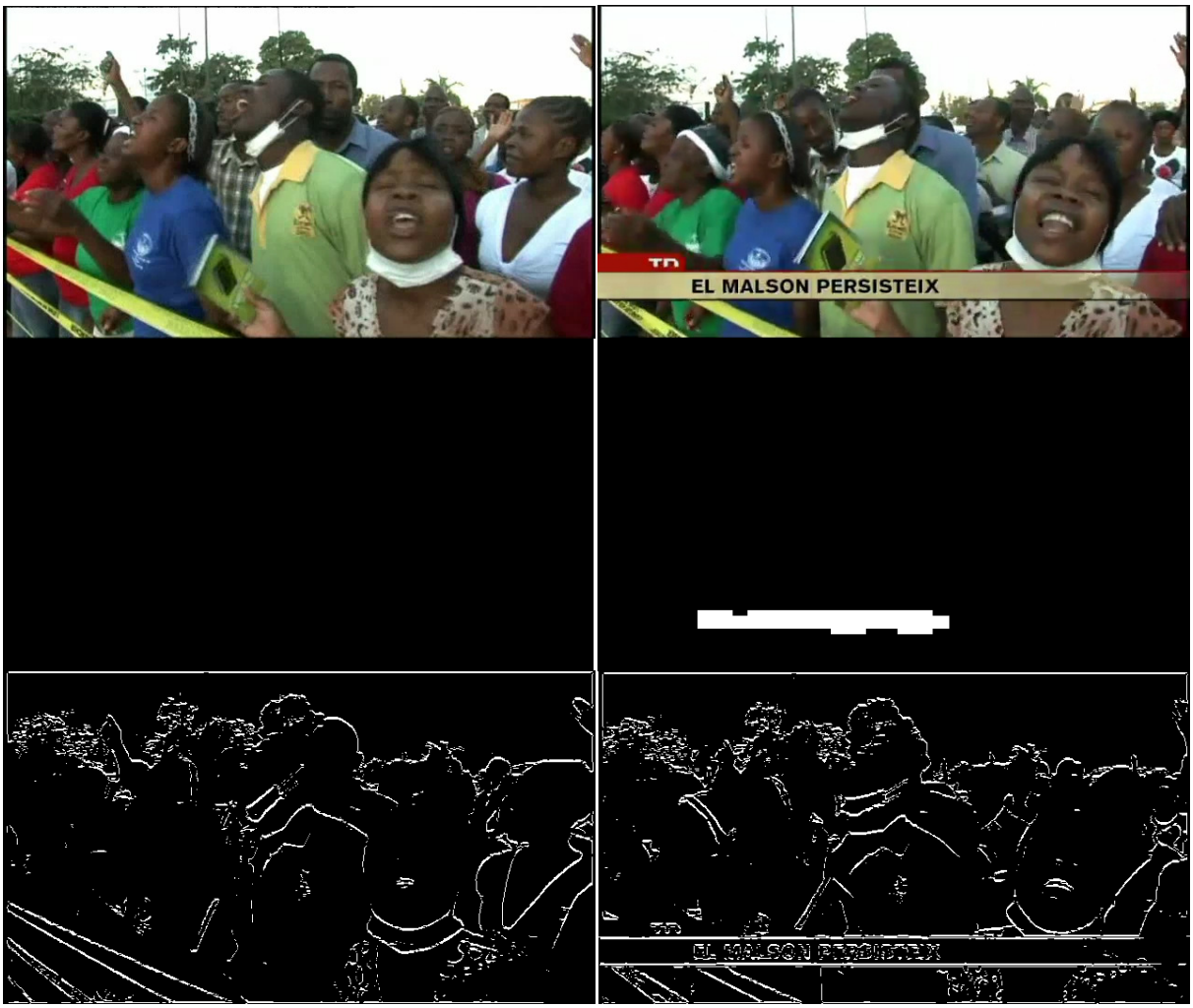


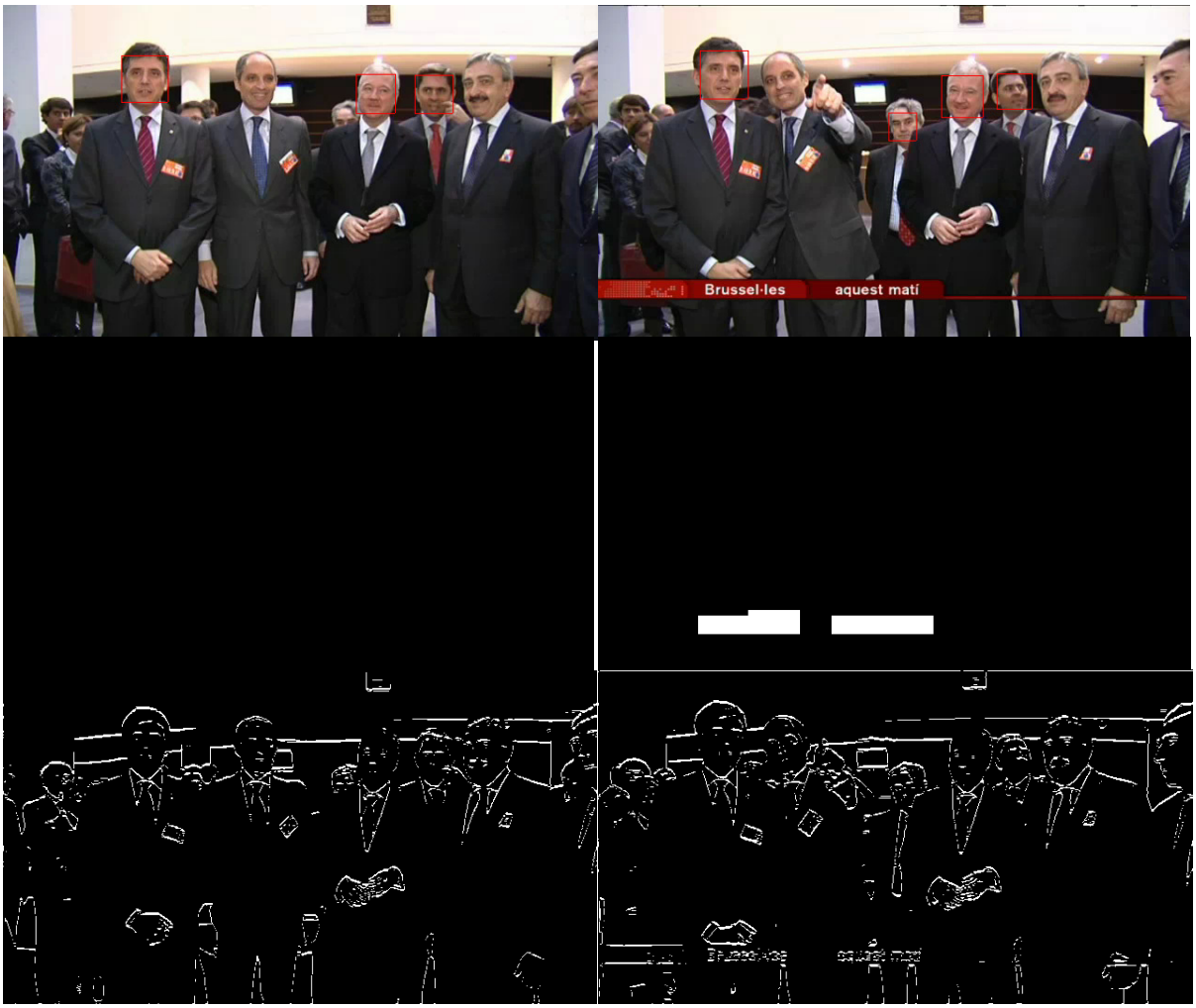
Como último ejemplo de la secuencia, en la imagen de la izquierda no se han detectado caras, mientras que hemos seleccionado una en que sí. Nótese la cantidad de falsos positivos de texto en ambas imágenes y que permanecen durante toda la escena, fácilmente confundible con texto.

9.2 Secuencia “Telenotícies”

Esta secuencia muestra el típico avance informativo que se emite al principio de un noticiario. Muestra un resumen de las noticias a tratar.

Por tanto, lo habitual es que aparezca un texto con el titular mientras van mostrando diferentes imágenes. La elección en este caso será normalmente por la presencia de este rótulo.


EL MALSON PERSISTEIX


Brussel·les
aquest matí

VII. Detector de carátulas de agencia

1. Motivación

Una gran fuente de información, por la cantidad de contenido que se genera y más aún por la relevancia de su contenido, son las secuencias de noticias de agencia.

Una agencia de información o agencia de noticias es una organización que recoge noticias de sus corresponsales en distintos lugares de su área de actividad y las transmiten inmediatamente a la central, donde, después de tratar la información, la envían, lo más rápido posible, a sus clientes (radios, diarios, revistas, televisoras o portales), conocidos en el argot periodístico como abonados. Estos abonados, como sería el caso de *Televisió de Catalunya*, se subscriben al tipo de información que desean (nacional, internacional, formato texto, formato vídeo, etc…) pudiendo recibir este contenido. En el caso de estudio en este documento es el de la recepción de secuencias de video.

Estas secuencias de video suelen tener la misma estructura: portada, índice de las noticias que se van a mostrar y a continuación, por cada noticia, una portada de la noticia que se va a mostrar y el video en sí. Estas portadas e índices aparecen como un texto sobre un fondo característico de cada agencia, que llamaremos carátula.

Figura 28: Ejemplo de índice de la agencia SNTV. Indica cual es la siguiente noticia marcada con un asterisco y su duración.

Es fácil de ver por tanto, que estas portadas contienen una información muy valiosa para una posterior indexación. No en vano, la información mostrada nos está indicando con mayor o menor detalle qué contiene exactamente esa secuencia, cosa que nos irá muy bien en el futuro para la búsqueda de contenidos en una base de datos.

Dada la importancia de dicha información, es necesaria una herramienta capaz de detectar estas carátulas y de extraer el texto que nos interesa para enviarlo a un sistema de reconocimiento óptico de caracteres (OCR). Como resultado, tendremos automáticamente el contenido de una secuencia de video en forma de texto.

2. Estado del arte

El sistema implementado en la CCMA por la compañía Visual Century Research S.L., de igual forma que con el detector de cambios de escena, se compone de un módulo (plugin) o filtro DirectShow llamado SatDetector. Este filtro es capaz de detectar la aparición de carátulas en el vídeo.

El plugin detecta el instante en que aparecen y desaparecen ciertas imágenes que han sido aprendidas previamente. Cuando una de las imágenes aprendidas aparece en el vídeo, el plugin notifica el inicio de la carátula, junto con el identificador del elemento detectado. Cuando la imagen cambia se notifica el fin de la carátula. El plugin genera escenas cada vez que detecta una nueva carátula. También implementa la detección de cuts, que funciona igual que el plugin detector de cuts.

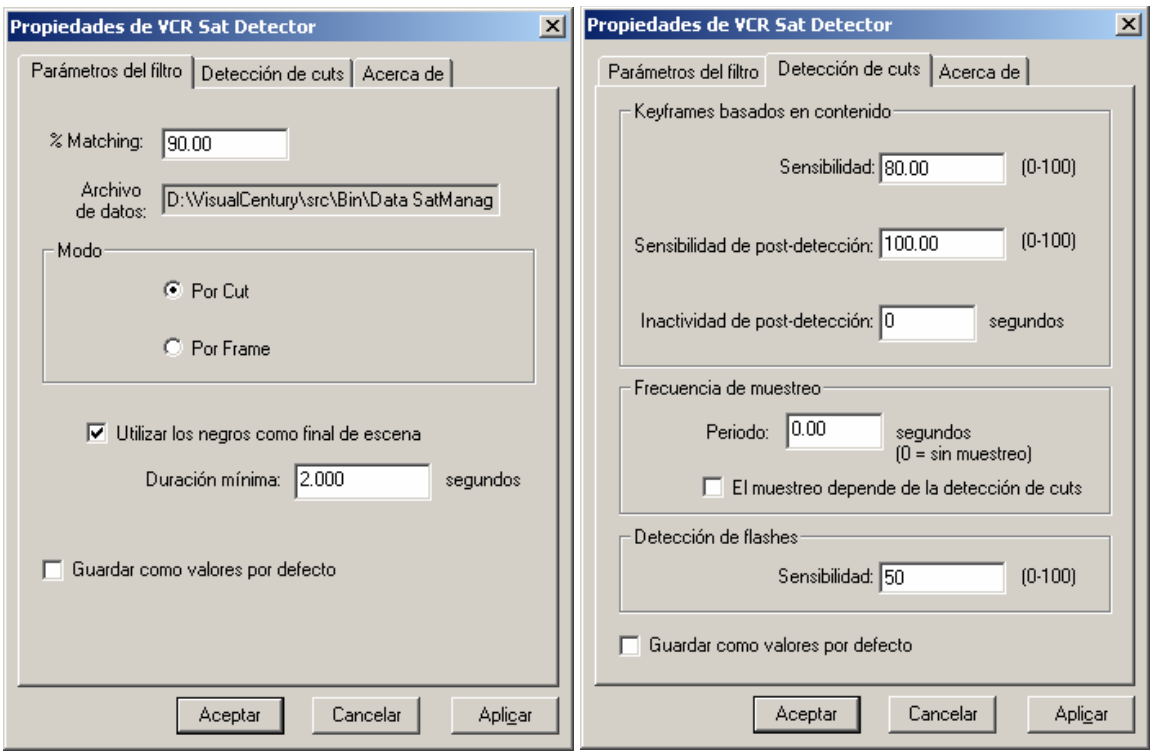


Figura 29: Interfaz del plugin detector de carátulas.

Para poder detectar carátulas, el primer paso consiste en aprender las imágenes que hay que detectar. Para ello se usa la aplicación SatManager con la que se gestionan los elementos a detectar, organizándolos en archivos .dat. Una vez se han aprendido las imágenes, se registra el archivo de datos mediante una opción de la aplicación.

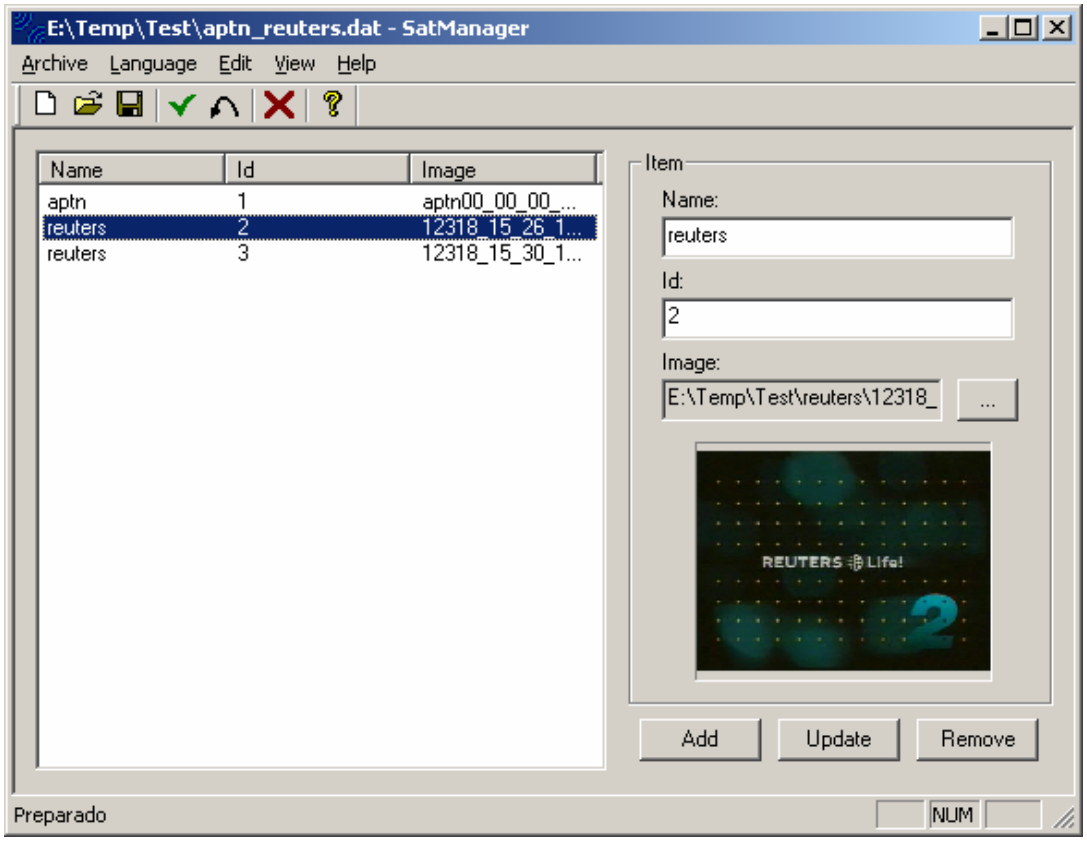


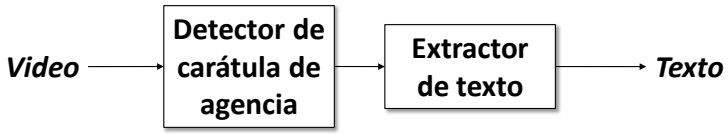
Figura 30: Interfaz del SatManager.

El método de detección se basa en el descrito en el bloque VI, apartado 4: “Detector de cambio de escena”. En este caso, además de detectar un cambio de escena, comprobamos si la nueva escena es una carátula comparándola con las almacenadas mediante la aplicación SatManager.

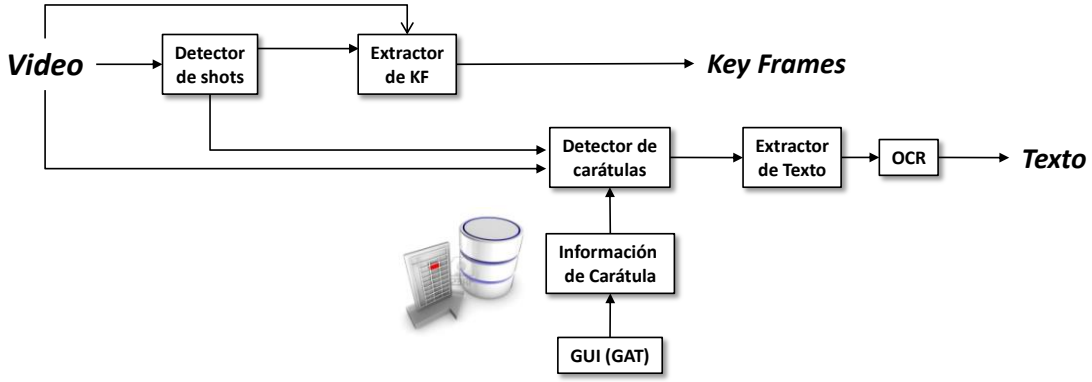
Por tanto, el primer paso para el desarrollo de nuestro sistema de extracción, tal y como se hizo con él detector de cambios de escena, será replicar el filtro DirectShow dentro de la librería desarrollada por el Grupo de Procesado de Imagen de la UPC, Image+.

3. Esquema

Habiendo establecido las necesidades de nuestro sistema se propone un esquema en el cual tenemos como entradas una secuencia de video y obtenemos a la salida texto plano. Este texto es el que contienen de forma gráfica las carátulas de video.



Este diagrama genérico contiene a su vez una serie de bloques que detallaremos a continuación:



Detector de carátulas: detecta la aparición de una carátula de agencia de noticias previamente almacenada en una base de datos.

Información de carátula: extrae y almacena en la base de datos la información relevante para la identificación de carátulas.

GUI: interfaz gráfica para la gestión de las carátulas que se quieren detectar.

Extractor de texto: extrae el texto presente en una carátula de agencia.

A continuación se describen las principales características y funcionalidades de cada bloque así como las razones por las que se utilizan.

3.1 Detector de Carátulas

La técnica utilizada para la detección de carátulas es exactamente la misma que la utilizada para detectar cambios de plano, el método propuesto por Swain-Ballard [1]. De hecho, el método se presenta como una técnica para identificar objetos en una base de datos, por sus aplicaciones en el campo de la visión por computador y robótica.

En este caso el criterio de comparación cambia. En vez de comparar el frame actual con el anterior, compararemos el frame actual con los presentes en una base de datos, siendo más estrictos en la comparación. Es decir, buscaremos cuál de los histogramas de color encaja mejor con el histograma de entrada. Además, tiene que parecerse lo suficiente como para juzgar que se trata de la misma carátula.

Los resultados experimentales de Swain-Ballard muestran que [1]:

- El Histograma Intersección puede distinguir objetos de una base de datos con un centenar de elementos. Algo apropiado para nuestra aplicación, ya que no se suele trabajar con más de una decena de agencias de noticias de las cuales, no suelen tener más de dos o tres tipos de carátula.
- Del abanico de colores que hay en la naturaleza tan solo se necesita dividirlo en 200 colores discretos diferentes para distinguir entre un gran número de imágenes.
- El resultado de la búsqueda es suficientemente insensible a la rotación y cambios en la distancia del plano.
- La identificación se puede realizar incluso cuando una parte significante de la imagen presenta una oclusión.
- La precisión del resultado es insensible a la resolución del histograma utilizado.
- Tendremos que tener en cuenta la sensibilidad que demuestra el método frente a flashes o cualquier tipo de destello que nos modifica bruscamente la iluminación.

El detector de carátulas puede funcionar de dos formas:

1. Para cada imagen de entrada, se compara con la base de datos y se busca si hay alguna carátula que se asemeje. Esto implica un coste computacional alto: por cada frame de entrada, hacemos N comparaciones de histogramas, donde N es el número de carátulas almacenadas.
2. Utilizar el detector de cambios de escena: la aparición de una carátula la podemos considerar como una escena nueva y así la detecta nuestra herramienta. Por tanto, sólo analizamos la primera imagen de la escena. Es obvio que en comparación, el coste computacional se reduce tanto como el número de frames que tiene la escena por el número de carátulas almacenadas.

Parece que utilizar la segunda opción es lo más apropiado. Sin embargo presenta un problema: las transiciones entre carátulas de la misma agencia no son detectadas por el detector de cambio de

escena, ya que son muy parecidas (tan sólo cambia el texto normalmente), por lo que podríamos perder información importante.

Por ejemplo, como comentaremos en el siguiente apartado, es habitual que se muestre una tabla de contenidos de noticias, el cual puede durar unos 10 segundos. A continuación, sin modificar el fondo, aparece el texto de la primera noticia a modo de portada. El cambio es tan ligero a nivel de histograma que el detector de cambios de escena lo considerará la misma y de hecho, según como lo hemos definido, lo es.

a) b)

Figura 31: a) Índice de noticias de un boletín de la agencia AP. b) Primera noticia del boletín, escena siguiente al índice.

Para solventar este problema, lo que hacemos es alternar los dos métodos:

- Mientras no se ha detectado ninguna carátula, sólo se analizan las procedentes del detector de cambios de escena.
- Cuando una carátula ha sido detectada, se analizan todos los frames de entrada, buscando cambios considerables (aunque menores que los que se producen en un cambio de escena) y se analiza cada nueva portada que aparezca. Una vez el detector de cambio de escena nos indique el comienzo de una nueva, volveremos al primer método.

De esta forma aseguraremos no perder ningún contenido.

3.2 Carátulas patrón

Una pieza clave del sistema es la creación de una base de datos con los modelos de las carátulas de las agencia de noticias que se van a utilizar. Las agencias a las que está suscrita y que utiliza habitualmente Televisió de Catalunya son las siguientes:

Reuters

Agencia estadounidense con sede en Nueva York y con más de 55.000 empleados en todo el mundo.

Associated Press Television News (APTN)

AP Television News es la división internacional de televisión The Associated Press (www.ap.org), de origen británico-canadiense y la organización más antigua de recopilación de noticias.

EFE

La primera agencia de noticias en español y la cuarta del mundo por detrás de Reuters, APTN y la francesa Agence France-Presse (AFP), con más de setenta años de trayectoria.

Sports News Television (SNTV)

Líder a nivel mundial en agencias de noticias deportivas.

Omnisport

Creada en Febrero del 2008, fue la primera en realizar un boletín diario de noticias deportivas diseñado para el mundo digital.

Estas agencias emiten constantemente contenido mediante streaming a un canal al cual el cliente tiene acceso. En general, este contenido puede ser de varios tipos:

- Boletín periódico
  A unas ciertas horas predefinidas se reproduce un boletín informativo con las noticias destacadas de la jornada hasta ese momento. Viene precedido por una secuencia corporativa y una tabla de contenidos. Para cada noticia suele contener una portada con cierta información textual descriptiva.
- Última hora
  En situaciones en las que ocurre un suceso trascendental se emiten instantáneamente las imágenes que se han captado de la noticia. Son por tanto emisiones impredecibles en cuanto a la hora de emisión. Suele aparecer como una simple portada con el título de la noticia.
- Vacío
  Durante el tiempo en el que no se está emitiendo nada de lo anterior, hay agencias que o bien no emiten nada (fondo negro), un bucle con su logotipo, un temporizador hasta su próxima emisión, etc.

a) b) c)

Figura 32: Ejemplos de contenido vacío: a) Agencia EFE b) Agencia AP c) Agencia EFE cuando hay un boletín previsto.

Además, la infraestructura establecida en Televisió de Catalunya hace que el flujo de video de las diferentes agencias se unifique en uno sólo, por lo tanto el sistema se tiene que plantear como que la entrada puede provenir de cualquier agencia.

Sin embargo, algo que sí es común es la presencia de las portadas antes de la noticia, con la carátula propia de la agencia como fondo. El primer paso para detectar que nos está entrando contenido nuevo relevante y de paso, de qué agencia proviene, será detectar esta portada. Para ello debemos indicarle al sistema cuáles son las carátulas que nos interesa detectar.

La primera opción que nos podríamos plantear sería conseguir las carátulas propiamente dichas, es decir, el fondo sin ningún tipo de información en forma de texto para que fuese lo más genérica posible. Sin embargo para hacer esto nos deberíamos poner en contacto con los grafistas de cada agencia para que nos proporcionasen este fondo, lo que resultaría una labor complicada por no decir que tendríamos que repetirla cada vez que se varía la imagen corporativa de la agencia.

La otra opción será utilizar cualquier portada capturada manualmente, conteniendo la carátula y texto de una noticia cualquiera, tarea que cualquier usuario puede realizar. Por tanto cada vez que nos interese capturar un nuevo tipo de portada, tan sólo tenemos que buscar una del mismo tipo y añadirla a la base de datos.

La base de datos estará entonces compuesta por una serie de metadatos que identifiquen a la imagen y la información necesaria para la identificación. Como hemos visto en la sección anterior, lo que necesitamos es el Histograma de Color que caracteriza a esa imagen, por lo que cuando añadimos una imagen nueva, tan sólo tenemos que calcular su Histograma de Color y guardar sus valores.

3.3 Segmentación de la zona de interés

En una primera propuesta se estudió la posibilidad de autogenerar la carátula a partir de N ejemplos de portadas que el sistema fuese encontrando. Es decir, que el sistema fuese aprendiendo cómo es esa carátula y fuese creando un modelo de ella. No obstante, en las pruebas que se han realizado se ha visto que la luminancia y la crominancia de las muestras variaban frecuentemente, con lo que complicaba bastante la reconstrucción. Resultaba imposible además en carátulas que contenían imágenes en movimiento.

No obstante se puede aprovechar la situación en la que es un usuario cualquiera el que gestiona lo que se añade a la base de datos y que cada agencia usa apenas dos o tres tipos de portada (una con el logotipo de presentación, tabla de contenidos y portada de la noticia). Si el usuario añade una de cada tipo o de las que quiera extraer información, también puede delimitar de qué zona es de la que quiere extraer esa información. Es decir, un primer paso hacia la segmentación para la localización del texto.

El usuario tan solo tiene que delimitar, mediante uno o varios rectángulos, las zonas que considera de interés. Con ello no sólo ahorramos en tiempo computacional, sino que se facilita pasos intermedios de análisis, extracción del texto y la posterior indexación. Por ejemplo, sabiendo que una región concreta contiene el título o la localidad de la noticia, el sistema puede automáticamente indexar esa noticia con el texto extraído. Además, no sólo se ahorra tiempo porque se reduce el espacio, sino por todo el texto que no se tiene que analizar como el propio de un logotipo. En definitiva, sabiendo qué tipo de carátula se ha detectado, conocemos su layout y por tanto podemos centrarnos en la región que nos interesa.

Para facilitar esta tarea al usuario, es necesaria la creación de una interfaz gráfica sencilla que permita estas opciones:

Añadir/Eliminar imágenes
Definir y gestionar regiones
Importar/Exportar datos

Afortunadamente, una interfaz que permite éstas y muchas otras opciones ya estaba creada y en continuo desarrollo por el Grupo de Procesado de Imagen. Esta interfaz se denomina Graphic Annotation Tool (GAT) y además ya se utilizaba en la propia TVC como herramienta para la anotación manual de contenidos audiovisuales, por lo que ya estaban habituados a utilizarla.

3.3.1 Interfaz gráfica

Esta herramienta de anotación manual ha sido desarrollada por el GPI de la UPC en el marco de la tarea A2T3 del Proyecto i3media. El GAT es una aplicación implementada en Java y su ejecución sólo está condicionada a la instalación de Java en el sistema operativo, que podrá ser MacOSX, Microsoft Windows o GNU/Linux.

GAT inicialmente tiene como objetivo proporcionar una interfaz gráfica de anotación de keyframes que facilite la posterior creación automática de modelos de clase semántica. Se ha diseñado e implementado para ofrecer al usuario un medio intuitivo e interactivo a través del teclado y el ratón.

Los objetos semánticos que se quieren anotar y las partes que los componen están identificados y estructurados en una o varias ontologías de clases semánticas. Los objetos y sus partes están unidos por la relación “parte de”. La relación “parte de” entre clases semánticas se indica durante el proceso de anotación. Cada clase semántica se caracteriza por una etiqueta de texto y un identificador numérico. La versión actual permite la lectura de ontologías definidas en un archivo MPEG-7/XML o TXT, o importadas desde una URL, es decir, de una página Web. Permite importar ontologías en formato OWL (Ontology Web Language). La interfaz también incluye un editor de ontologías en formato MPEG-7/XML.

La herramienta permite que la semántica contenida en una imagen se pueda anotar a dos escalas: imagen o píxel. En el caso imagen, el área de soporte es la imagen completa, mientras que en el caso píxel, el área de soporte es un subconjunto de píxeles de la imagen, una región. La selección del área de soporte por parte del usuario dependerá de la naturaleza de la clase semántica. Mientras algunos conceptos abstractos como bosque o evento deportivo se querrán representar con la totalidad de píxeles de la imagen, otros conceptos como coche o jugador de fútbol se querrán representar con un subconjunto específico de píxeles de la imagen.

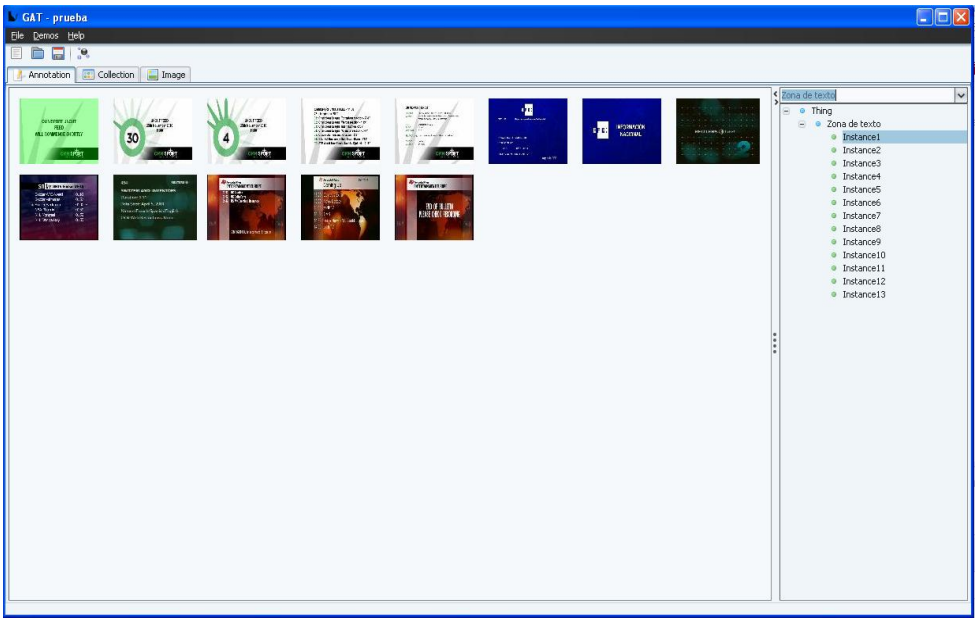


Figura 33: Imagen del GAT con algunas carátulas insertadas.

El GAT permite hacer anotaciones masivas de imágenes a nivel imagen, es decir, en lugar de anotar imagen por imagen el usuario puede anotar una tira de imágenes a la vez, resultando una instancia atómica por cada una de les imágenes.

El archivo de salida de GAT corresponde a una sesión de anotación y también se expresa en formato MPEG-7/XML. El archivo describe, por cada imagen anotada, qué área de soporte (ya sea toda la imagen, regiones, puntos, etc.) representan las instancias de les clases semánticas definidas en una ontología.

Nuestro uso del GAT será mucho más básico ya que sólo debemos indicar a cada imagen un conjunto de rectángulos que a priori contendrán texto. La salida será un archivo XML con la información en uno de sus nodos del nombre de la imagen y la sucesión de coordenadas que representan dos vértices opuestos de cada rectángulo en otro nodo del archivo. Al iniciar nuestro sistema, buscará este archivo XML y recorrerá los diferentes nodos para buscar qué imágenes (carátulas) se van a utilizar, calculará su Histograma de color y lo almacenará junto con los diferentes rectángulos de cada imagen, los cuales llamaremos Bounding Box.

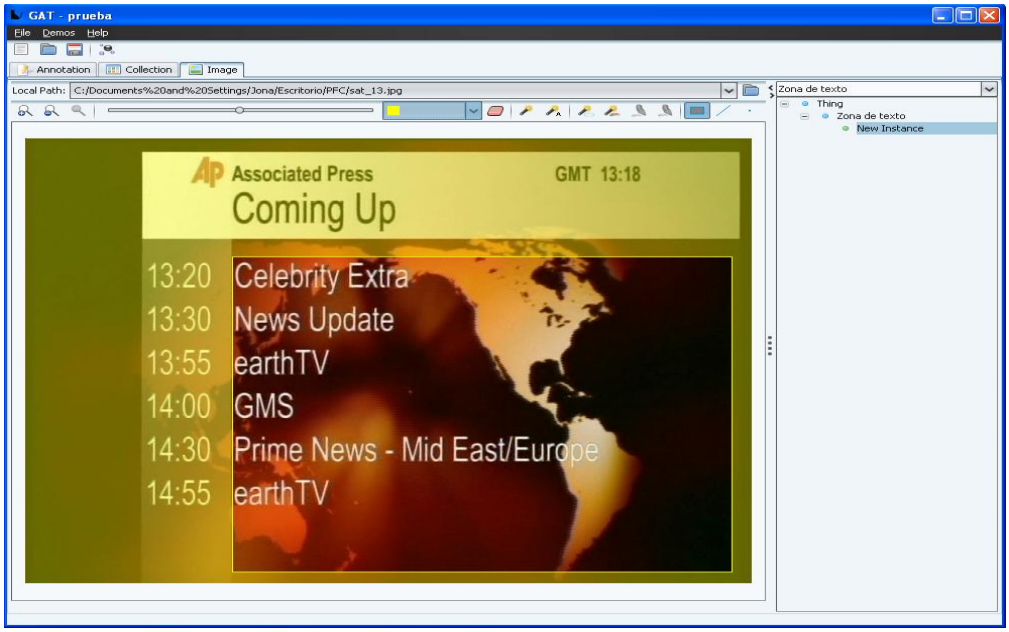


Figura 34: Detalle del GAT con la selección de la zona de interés de un tipo de carátula (agencia AP).

Nuestras carátulas patrón quedan definidas entonces por su histograma de color y sus bounding boxes. Cuando una carátula es detectada en una secuencia, se identifica con la almacenada en la base de datos y se mira cuántas regiones o Bounding Boxes contiene. El siguiente paso entonces es buscar, para cada Bounding Box, qué conjuntos de píxeles contienen texto.

4. Algoritmo

4.1 Detección de texto

Para identificar las regiones que contienen texto usaremos el mismo método que hemos utilizado para la detección de texto en el extractor de keyframes (bloque VI, apartado 6). En este caso la detección de texto se efectúa para realizar una extracción mediante una binarización y un reconocimiento de caracteres.

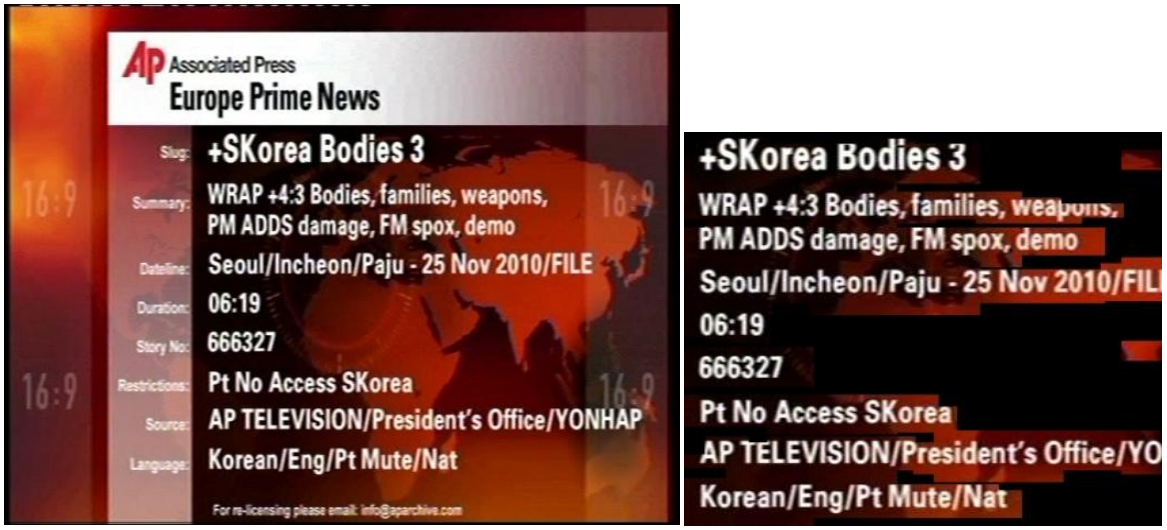


Figura 35: A la izquierda, carátula detectada de la agencia AP. A la derecha, región de interés seleccionada y resultado del detector de texto.

4.2 Binarización

El objetivo del proceso de binarización en la extracción de texto, es obtener una imagen en la que únicamente aparezca texto. Esta imagen será la entrada de un sistema OCR, con lo cual presumiblemente obtendremos un mejor resultado que con la imagen original. Por tanto, todos los elementos que no correspondan a caracteres, como texturas de fondo, deberán eliminarse.

4.2.1 Binarización convencional

El método propuesto en *Region-Based Caption Text Extraction* [5] analiza segmentos horizontales equidistantes para cada región candidata a contener texto.

Para cada segmento, se calcula la media y varianza de los valores de luminancia que contiene. Una línea con una alta varianza indica presencia de texto mientras que una con baja varianza asume que se corresponde únicamente con el fondo. En este último caso, el valor de la media puede utilizarse para caracterizar su PDF, que podemos asumir como Gaussiana.

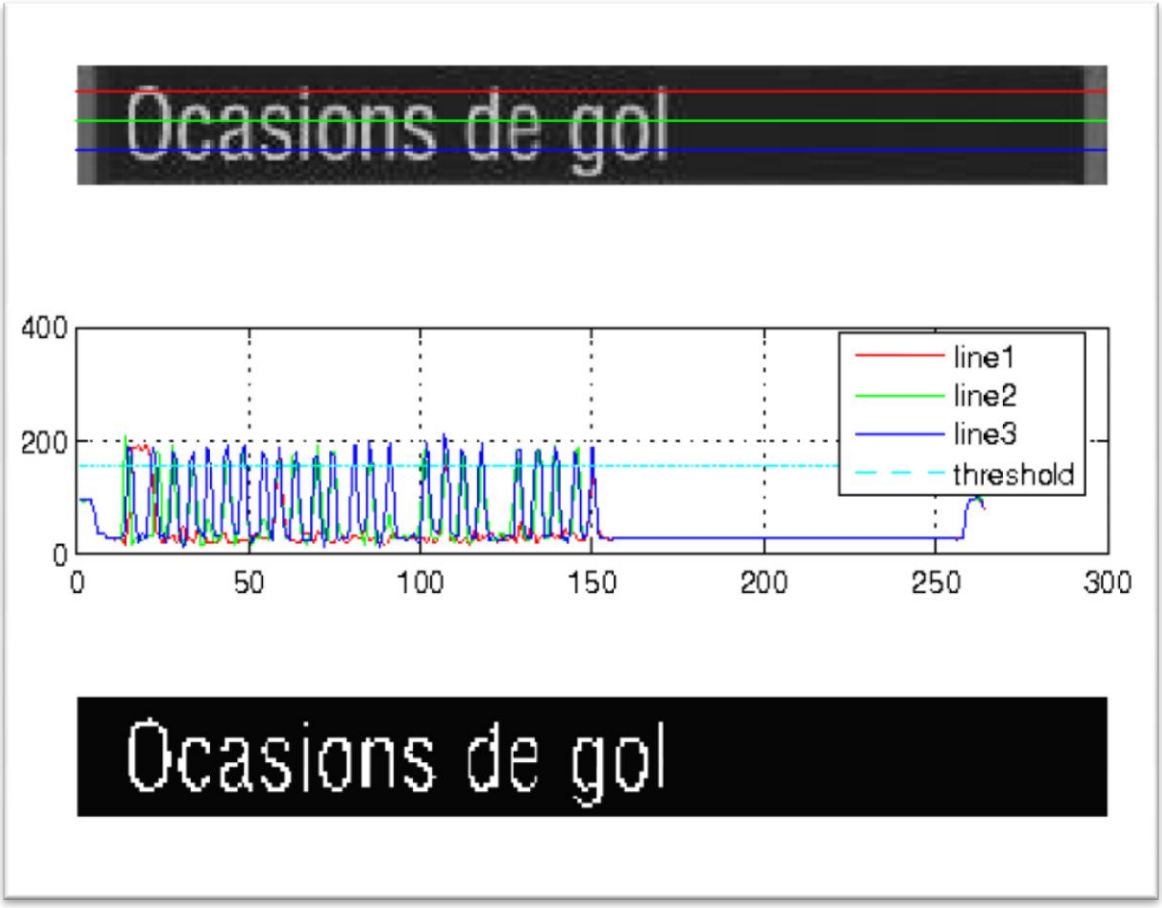


Figura 36: Resultados método propuesto en [5]

Este método ofrece buenos resultados incluso en imágenes poco contrastadas y en las que los caracteres estén bien definidos y separados.

En imágenes procedentes de una secuencia de video, debido a los procesos de codificación y compresión que sufren, es habitual la pérdida de definición. Esto provoca en algunos casos un deterioro sustancial tanto del texto como del fondo, por lo que complica su extracción. Esto se agrava en los textos más pequeños en términos de dimensión del carácter y cuyo grosor es equiparable al de la separación entre caracteres.

Figura 37: Fotograma de una secuencia de video

El sistema visual humano es muy sofisticado e imposible de reproducir a día de hoy. Desde la captación de una imagen en nuestro ojo, hasta la interpretación de nuestro cerebro existe todo un sistema complejo de procesos que tienen como resultado el sentido de la vista. En la figura 37, a pesar del tamaño podemos leer en la parte superior: “Título: Bajas temperaturas en Valladolid”.

Al diseñar algoritmos de compresión de video se tienen en cuenta estos procesos y se trata de que el sistema visual humano siga interpretando lo mismo a pesar de tener una imagen de peor calidad. Con ello se consiguen secuencias visualmente iguales representadas con un número menor de bits.

El problema se presenta cuando intentamos reconocer estas imágenes mediante sistemas digitales. El deterioro de los contornos produce un efecto de unión entre los caracteres.

Como se puede observar en la figura 38, la degradación de la imagen provoca que los contornos queden poco definidos, el color no sea homogéneo entre diferentes caracteres ni siquiera en sí mismos y sobretodo en el difuminado que une los caracteres.

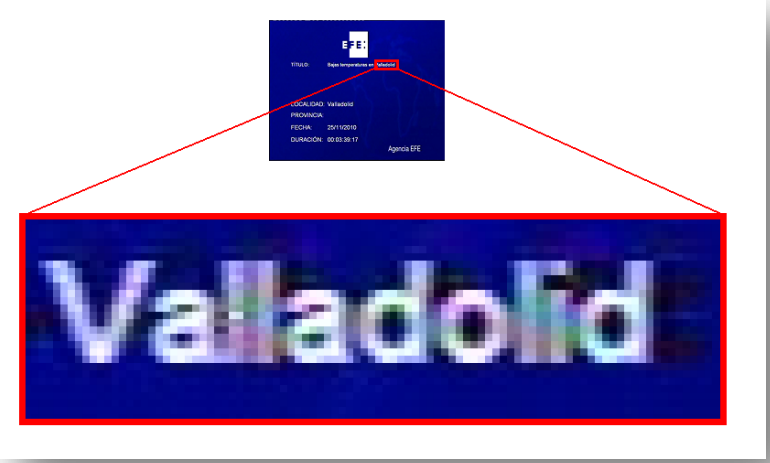


Figura 38: Efecto de la compresión de datos en video.

En una situación como esta, con un fondo muy homogéneo y un texto muy contrastado respecto a éste, no es difícil establecer un umbral para binarizar texto y fondo mediante el método explicado anteriormente.

Si representamos los valores de luminancia de un segmento horizontal en la zona central de la zona de texto capturada, podemos ver un gráfico como el siguiente:

Figura 39.1: Luminancia en el segmento indicado de color rojo, con el umbral calculado en color violeta y resultado tras la binarización.

En la figura 39.2 se puede apreciar la problemática del método de binarización comentado. Con este resultado a la entrada de un OCR es poco probable obtener la misma palabra que nuestro sistema visual nos dice que hay. Sería necesario un diccionario y un entrenamiento muy específicos.

Figura 39.2: Detalle del resultado.

Para resolverlo se necesita un algoritmo que no se base tan sólo en la eliminación del fondo, sino también en la separación de caracteres. Por ello se plantea el siguiente método de binarización por máximos y mínimos.

4.2.2 Binarización por máximos y mínimos

Como se ha estado diciendo hasta ahora, los métodos convencionales de binarización se basaban en la idea de extracción de texto y eliminación del fondo. No obstante, vemos que para imágenes con texto relativamente pequeño, afectadas por técnicas de compresión u otros deterioros, presenta algunos resultados menos precisos.

El método propuesto cambia esta idea y lo que queremos eliminar no es únicamente el fondo, sino que debemos buscar las transiciones entre caracteres para poder generar regiones inconexas que serían estos mismos caracteres.

Para la estimación del fondo, aplicaremos el método convencional.

Una vez dicho esto, es habitual pensar en los operadores morfológicos de apertura/cierre. No obstante, no es trivial definir un elemento estructurante que nos sirva para cualquier tipo de imagen y que, a la vez, asegure separar los caracteres sin hacer desaparecer elementos propios: puntos, comas, huecos internos, etc…

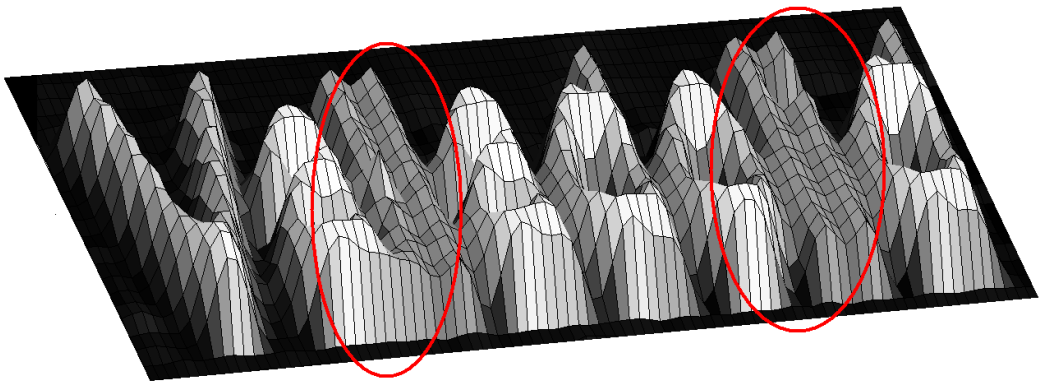


Figura 40: Vista en tres dimensiones destacando la unión de caracteres.

El problema principal se puede ver en la figura 40, donde representamos la luminancia como altura en la palabra “Valladolid”. Lo que observamos es que, como ya habíamos anticipado, los caracteres tienen un decaimiento que se podría modelar como gaussiano. Esto provoca en algunos de ellos que estos decaimientos intersecciones, por lo que puede ocurrir que no haya un contraste suficiente como el que hay con el fondo. Un ejemplo clarísimo de esto lo podemos apreciar en la cadena de caracteres “ll” y “li” de la palabra “Valladolid” o bien en menor medida en la unión de “al” e “id”.

Veamos este fenómeno en el corte que estábamos analizando anteriormente:

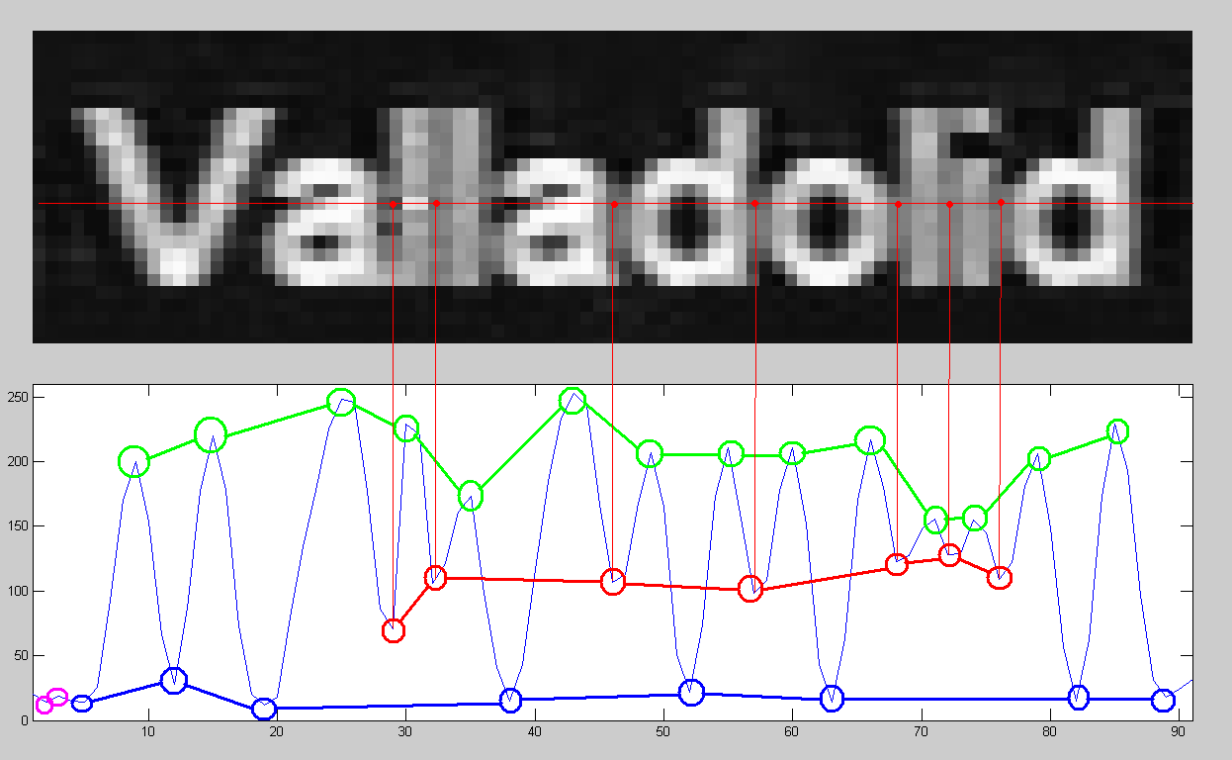


Figura 41 Luminancia a lo largo del segmento y clasificación de los puntos de interés: en verde, máximos; en rojo, mínimos debidos a intersecciones; en azul, mínimos debidos al fondo y en violeta, máximos y mínimos debidos al ruido.

En la figura 41 se observa claramente y destacado en color rojo estas intersecciones entre caracteres que se presentan en forma de mínimos locales. Para cada palabra estos mínimos tienen unos valores similares, por lo que podríamos acotarlos en un rango.

Por otro lado, la linea verde nos muestra la envolvente del texto, que no es más que la unión de máximos de cada parte del carácter. Por tanto, si tuviésemos que definir un umbral para aplicar una binarización, éste debería estar entre las lineas verde y roja sin llegar a tocarlas. En este caso concreto, como no llegan a cruzarse, no sería dificil establecer este umbral.

Para finalizar, hay que tener en cuenta la presencia de máximos y mínimos debidos al ruido, que aparecen en color violeta, para tratar de diferenciarlos con los máximos y mínimos que realmente nos interesan. En este caso, por suerte, aparecen únicamente en valores de fondo, aunque podrían aparecer en cualquiera de los otros dos niveles: el de mínimos por intersección o el de máximos de texto.

El método propuesto se basa en encontrar estos puntos que nos permitirán realizar una binarización más eficiente. Para ello podemos utilizar diversas técnicas las cuales explicaremos en detalle:

- Detección de máximos

  Detectando los máximos locales podemos representar el esqueleto de los diferentes caracteres para una posterior reconstrucción.
- Detección y clasificación de mínimos

  En este tipo de imágenes, podemos encontrar hasta 3 tipos de mínimos locales: debidos al ruido, debidos a las intersecciones y debidos al fondo. Detectando correctamente los debidos a las intersecciones, establecemos un umbral por encima del valor de luminancia de éstas para posteriormente binarizar.
- Detección de máximos y mínimos

  Juntando los dos conceptos, podemos llegar a establecer una binarización mucho mas eficiente.

4.2.2.1. Detección de máximos

El método basado en la detección de máximos tiene como objetivo destacar o ensalzar los puntos de la cresta que forman los caracteres en la figura 40.

Para ello se extraen fila a fila los máximos locales mediante la estimación o aproximación de derivadas. Una vez extraidos, los clasificaremos en dos grupos: máximos debidos a texto, que aparecen con valores de luminancia típicamente altos y máximos debido a ruido, que normalmente aparecen en valores del fondo. Como algoritmo de clasificación se utiliza K-means con 2 centroides.

Una vez clasificados, se realiza un toggle-map aplicado sobre los máximos, es decir, los máximos clasificados como texto, se les da el mayor valor de luminancia posible. A continuación, binarizamos estableciendo el umbral inmediatamente inferior a este valor máximo.

Fig 42 Resultado de la binarización por máximos

Con esta técnica obtenemos un esqueleto bastante completo de los diferentes caracteres que forman la palabra. Los máximos que no han sido realzados han sido clasificados como máximos de fondo. Esto nos hace ver la pequeña parte de aleatoriedad durante la etapa de la clasificación: si no encuentra máximos de ruido, divide igualmente los máximos en dos grupos, con lo que perdemos candidatos buenos. No obstante, mediante esta técnica aseguramos que aquellos máximos realzados deben aparecer en la binarización.

4.2.2.2. Detección y clasificación de mínimos

La siguiente técnica trata de encontrar específicamente los mínimos debidos a intersecciones. Para ello utilizaremos la variable lambda-contrast (contraste lambda) que se define como la mínima distancia entre un mínimo y el máximo que lo precede o le sigue. Por tanto, se trata de la diferencia entre el máximo mas cercano (luminancia más baja) y el mínimo en sí. Con ello obtenemos para cada mínimo su relación de contraste respecto a los máximos que le envuelven. La idea es que los mínimos debidos a ruido tendrán una lambda menor, los mínimos debidos al fondo tendrán una lambda mayor y, por último, los mínimos debido a intersecciones tendrán una lambda con un valor intermedio.

Aplicando una vez más el algoritmo de clasificación K-means, con 3 centroides en esta ocasión, trataremos de encontrar estos tres tipos de lambda: debidas al ruido, al fondo o a intersecciones.

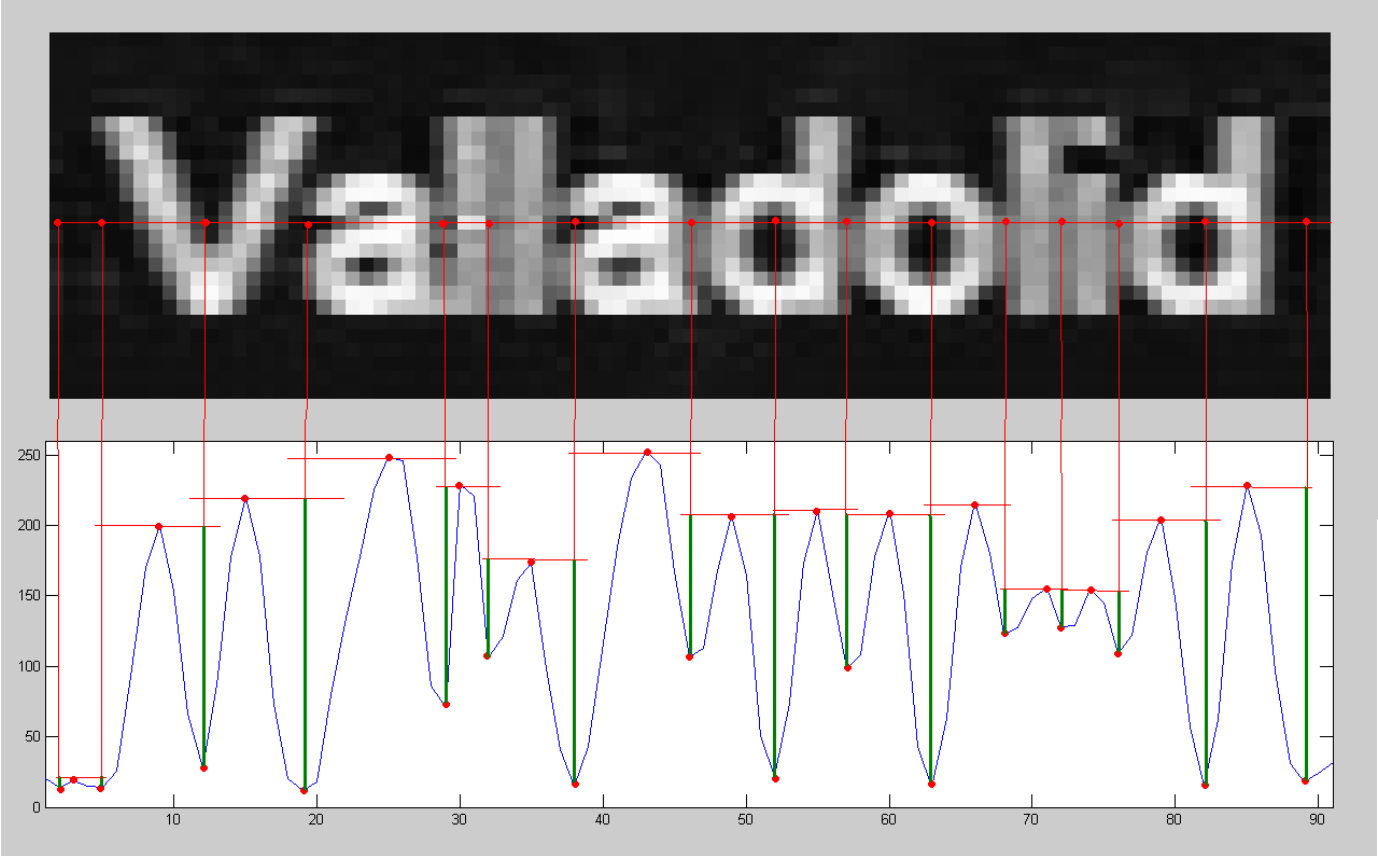


Fig 43. Visualización de la distancia entre mínimos y su máximo más cercano (lambda-contrast, en verde)

Con la clasificación hecha, buscamos aquellos mínimos catalogados como mínimos por intersección (medios) y situamos el umbral inmediatamente por encima del mayor de todos ellos. Obtendríamos un resultado como el representado en la siguiente figura.

Figura 44: Resultado de la binarización a partir del umbral definido por el mínimo de intersección de mayor luminancia.

A partir del resultado obtenido, nos planteamos utilizar la técnica de toggle-map como habíamos realizado en la binarización por máximos. Así pues, para los mínimos con lamdas medias y grandes, es decir, todos los mínimos que no sean debidos al ruido, les asignamos el mínimo valor posible, normalmente 0. Con ello nos aseguramos separar todas las intersecciones sea cual sea el umbral.

Figura 45: Resultado tras aplicar toggle-map a los mínimos.

Este último resultado es muy esperanzador y certifica que es una buena técnica. Sin embargo, hay que tener en cuenta que dependemos mucho del clasificador, por lo que puede ocurrir que no establezcamos el umbral en un rango aceptable. Para ello es conveniente fusionar y perfeccionar ambas técnicas.

4.2.2.3. Detección de máximos y mínimos

Mediante la unión de estas dos técnicas, obtenemos dos umbrales distintos: umbral de máximos y el umbral de mínimos. Esto nos establece un rango a priori válido para establecer el umbral.

Podemos encontrar hasta cuatro casos:

I - Umbral de máximos >> Umbral de mínimos
II - Umbral de máximos > Umbral de mínimos
III - Umbral de máximos < Umbral de mínimos
IV - Umbral de máximos << Umbral de mínimos

El valor a escoger entre estos rangos dependerá del tipo de imágenes que utilizamos. Para las imágenes de agencia o texto en pastillas, tras diferentes pruebas, la mejor forma de elegir un candidato a umbral es ponderando los dos resultados según en cual de los cuatro casos nos encontremos quedando de esta forma:

I – 70% Umbral de máximos + 30% Umbral de mínimos
II – 60% Umbral de máximos + 40% Umbral de mínimos
III – 40% Umbral de máximos + 60% Umbral de mínimos
IV – 30% Umbral de máximos + 70% Umbral de mínimos

De esta forma, el umbral que escogemos queda siempre dentro del rango de los dos que hemos encontrado. Ademas, con el toggle-map realizado en cada etapa de detección de máximos y mínimos, nos aseguramos que se realcen máximos o desaparezcan mínimos que han quedado por debajo o por encima del umbral respectivamente, tras una mala clasificación.

4.2.3 Binarización palabra por palabra

Hasta ahora hemos asumido que el fondo y el texto que deseamos extraer tienen una función densidad de probabilidad que podríamos modelar como Gaussiana. Sin embargo, esto no ocurre cuando aparecen palabras de distinto color o el fondo contiene degradados o formas . Así pues, las condiciones cambian y ya no es válido el umbral que habíamos encontrado. Por ello es necesario afinar y acotar en espacio las zonas donde aplicar ese umbral.

El paso natural para dividir un texto en zonas es separarlo por palabras. Para encontrar las diferentes palabras podemos aplicar un detector de contornos y proyectar los contornos sobre una línea horizontal imaginaria.

Figura 46: De arriba abajo: Carátula original. Zona de interés. Detección de bordes y proyección de las líneas.

Esta línea contendrá discontinuidades, de las cuales algunas, las mas grandes, correspondrán a las separaciones entre palabras y el resto a las pequeñas separaciones que hay entre caracteres.

Una vez clasificadas estas discontinuidades, habremos delimitado las palabras que forman el texto, aunque no es necesario que sea de forma exacta.

Por otro lado, realizamos la misma operación en vertical. Proyectamos sobre una linea vertical imaginaria los contornos encontrados. Con ello delimitamos las lineas que contienen texto y su altura.

El tratamiento de las palabras es ahora independiente para cada una de ellas. Se aplica la detección de máximos y mínimos para cada una de las lineas que contienen texto, establecido por la linea vertical imaginaria. Aquellas líneas que no estén marcadas como texto se utilizarán para estimar el fondo. Sin embargo al reducir el espacio de análisis a cada palabra, el método de clasificación de puntos (K-means) que utilizamos en varios puntos del algoritmo puede que no consiga obtener claramente las clases que hemos supuesto.

Por ejemplo para la clasificación de mínimos esperamos la presencia de mínimos debidos al ruido. Si no aparecen en una palabra, habitualmente porque es muy corta, el algoritmo clasificará como mínimos al ruido a otros mínimos.

Para solventar este problema lo que hacemos es forzar la aparición de estos mínimos. Antes de efectuar la clasificación, se añaden al grupo N mínimos (habitualmente N=3) con el valor que nos interesa. En este caso, valores pequeños (lambda=1), por tanto éstos apareceran en el grupo de mínimos debidos al ruido por su bajo valor de lambda y posteriormente se eliminan.

Para la detección de máximos ocurre algo parecido. Imaginemos que nuestra línea muestra una perfecta sinusoide. Los únicos máximos que aparecerían serían los propios del texto, por lo que se clasificaría erroneamente con dos clases. Nos interesa que aparezcan unos máximos debidos al fondo, por lo que introducimos N máximos con el mismo valor que se ha estimado del promedio del fondo.

Por último, cuando se ha obtenido un umbral para cada linea, se calcula el promedio de éstos y se aplica a toda la región o palabra.

En resumen, el proceso resultante es el siguiente:

1. Imagen de entrada: pastilla de texto (zona de texto detectado)
2. Detección de bordes: filtro Canny
3. Proyección del resultado del filtro horizontal y vertical:
4. La proyección en vertical nos indicará la separación entre palabras y caracteres.
5. La proyección en horizontal nos indicará las líneas que contienen texto.
6. Se divide el texto en palabras clasificando las discontinuidades de la proyección vertical. Cada palabra es tratada independientemente de la anterior:

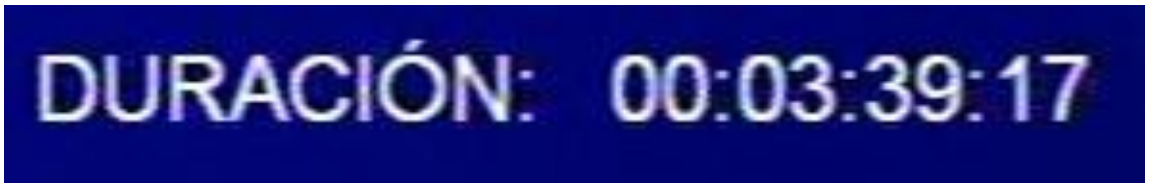
a. Las líneas que no contienen texto se utilizan para estimar el valor medio de luminancia del fondo.
b. A las líneas que contienen texto se les aplica la detección de máximos y mínimos y se establece un umbral por linea. Una vez procesadas todas las lineas, se calcula la media de los umbrales y se aplica a toda la palabra.

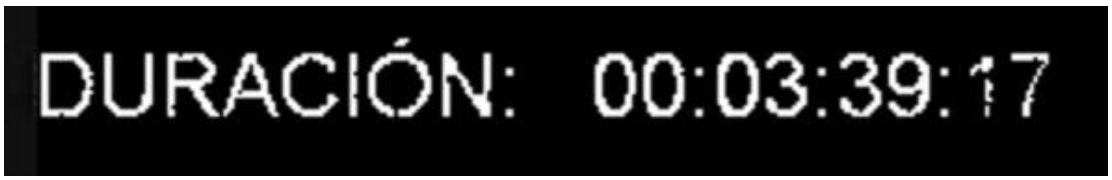
4.2.4 Ejemplos

A continuación se muestran diferentes carátulas que se han detectado. Se seleccionan zonas de texto para ver el resultado de la binarización.

4.2.4.1. Carátula SNTV:

4.2.4.2. Carátula EFE:





4.2.4.3. Carátula APTN

4.2.4.4. Carátula APTN (2)

El último ejemplo es uno de los que más complicación presentan: mientras que prácticamente todas las carátulas son estáticas, en este caso el globo terráqueo que se encuentra de fondo va dando vueltas y produciendo destellos luminosos. Para un humano no es más que un adorno, pero la extracción de texto aquí se complica enormemente por las constantes variaciones del fondo.

Cada frame contendrá diferente información visual y habrá zonas en las que se puede detectar correctamente y en otras no. Nos encontramos palabras en las que el color del fondo varía bruscamente, incluso de blanco a negro, por lo que establecer un umbral es complicado.


Celebrity Extra


Prime News - Mid East/Europe


Celebrity Extra


Prime News - Mid East/Euro

4.3 Sistema OCR: Tesseract

Tesseract es un motor OCR de código abierto desarrollado en HP entre 1984 y 1994. Apareció de la nada para la UNLV 1995 Annual Test of OCR Accuracy [1], con unos resultados destacables, y luego desapareció de nuevo en el mismo secreto en el que se había desarrollado.

Tras diez años sin ningún desarrollo, fue liberado como código abierto en el año 2005 por Hewlett Packard y la Universidad de Nevada, Las Vegas. Actualmente es desarrollado por Google y distribuido bajo la licencia Apache, versión 2.0.

Tesseract está considerado como uno de los motores OCR libres con mayor precisión disponibles actualmente. Desde su liberalización, los detalles de la arquitectura y algoritmos pudieron ser revelados.

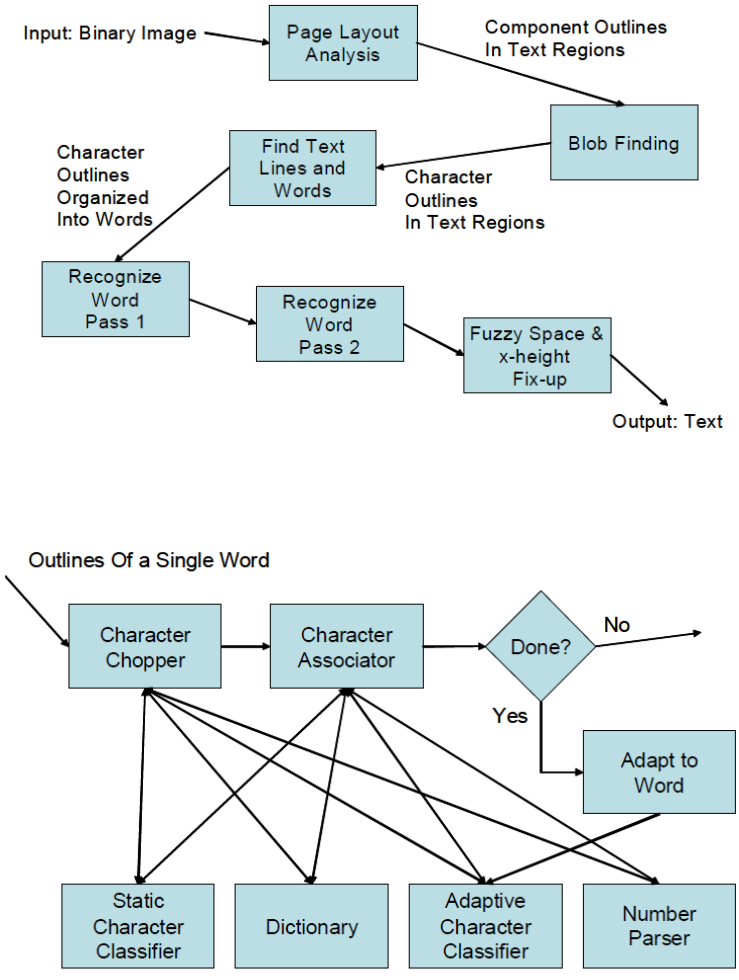
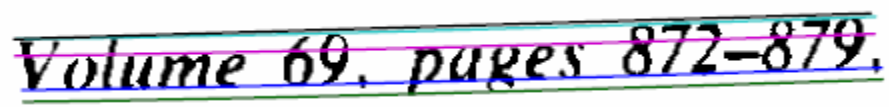


Figura 47: Esquemas del sistema Tesseract

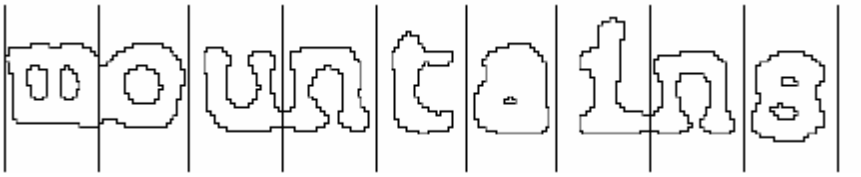
4.3.1 Arquitectura

El proceso sigue una tradicional pipeline paso a paso. El primer paso es un análisis de componentes conexas en el que se describen en forma de árbol las componentes que se almacenan. Este fue un diseño computacionalmente costoso en aquel momento, pero tenía una ventaja significativa: inspeccionando los nodos del esquema y el número de ramificaciones, era simple detectar texto y reconocer si es negro sobre blanco o al revés. Tesseract fue probablemente el primer motor OCR capaz de manejar texto blanco sobre negro tan trivialmente. En este punto, los nodos se unificaban en burbujas.

Las burbujas están organizadas en líneas de texto, y una vez han sido encontradas, se ajustan líneas de tendencia con aproximación cuadrática para seguir la trayectoria de la línea. Este fue otra de las innovaciones en un sistema OCR, ya que permitía a Tesseract analizar páginas con líneas de texto curvadas.



Una vez tenemos las líneas de texto, éstas son analizadas buscando pasos fijos (pitch). Cuando encuentra texto con unos pasos fijos, las líneas de texto se rompen en palabras de forma diferente de acuerdo con el tipo de espaciado de carácter. De la misma forma, se corta cada palabra por los diferentes caracteres aun cuando la proporción del texto se rompe. Para ello el sistema mide los espacios en un rango vertical limitado entre las líneas de tendencia y la línea central.



Partiendo de los valores de menor confianza dados por el clasificador de caracteres, se trocea la burbuja de cada carácter utilizando puntos en los vértices cóncavos de una aproximación poligonal. De esta forma, puede haber otro vértice cóncavo o un segmento en el lado opuesto. Se pueden necesitar hasta tres pares de puntos para separar correctamente caracteres de la tabla ASCII.

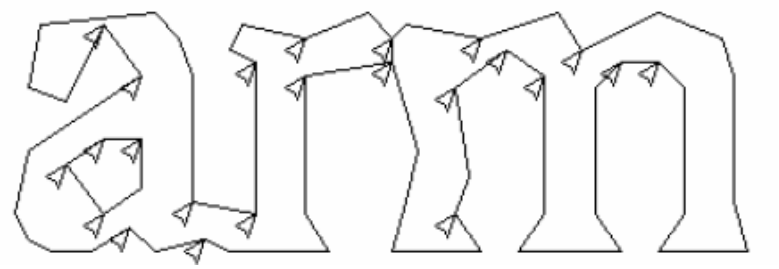
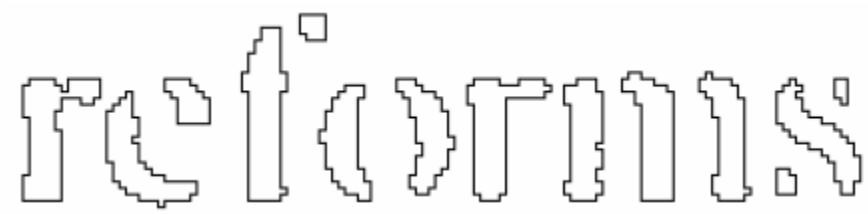


Figura 48: Conjunto de puntos candidatos y punto de rotura escogido a modo de segmento entre las letras ‘r’ y ‘m’

Las divisiones se producen en orden de prioridad. Si cualquier fragmentación falla en la mejora de la confianza, el resultado se deshace, pero no se descarta.

Cuando se agotan las posibles divisiones y una palabra no presenta aún buenos resultados, se pasa al bloque asociador. El bloque asociador busca los posibles caracteres próximos con mayor fragmentación y los une en una burbuja como candidato a carácter.



El reconocimiento entonces procede como un proceso de dos pasadas. En la primera pasada, se hace un intento de reconocer cada palabra. Cada palabra que se detecta satisfactoriamente se pasa a un clasificador adaptativo como datos de entrenamiento. El clasificador tiene entonces la oportunidad de hacer un reconocimiento más preciso del texto que tiene a continuación.

Como el clasificador puede haber aprendido algo útil demasiado tarde como para hacer una contribución buena en la primera parte del texto, se recorre una segunda vez

Una fase final resuelve espaciados poco claros y comprueba hipótesis alternativas para las variaciones de altura o letras de tamaño menor. Para ello utiliza una ratio que relaciona la línea de tendencia con la altura, mientras que el clasificador normaliza los caracteres utilizando el centroide (momento de primer orden) para la posición y los momentos de segundo orden para una normalización de tamaño anisotrópica. Esto hace más sencillo distinguir entre mayúsculas y minúsculas así como mejorar la inmunidad contra posibles artefactos.

Figura 49: Ejemplo de normalización para la palabra “win”

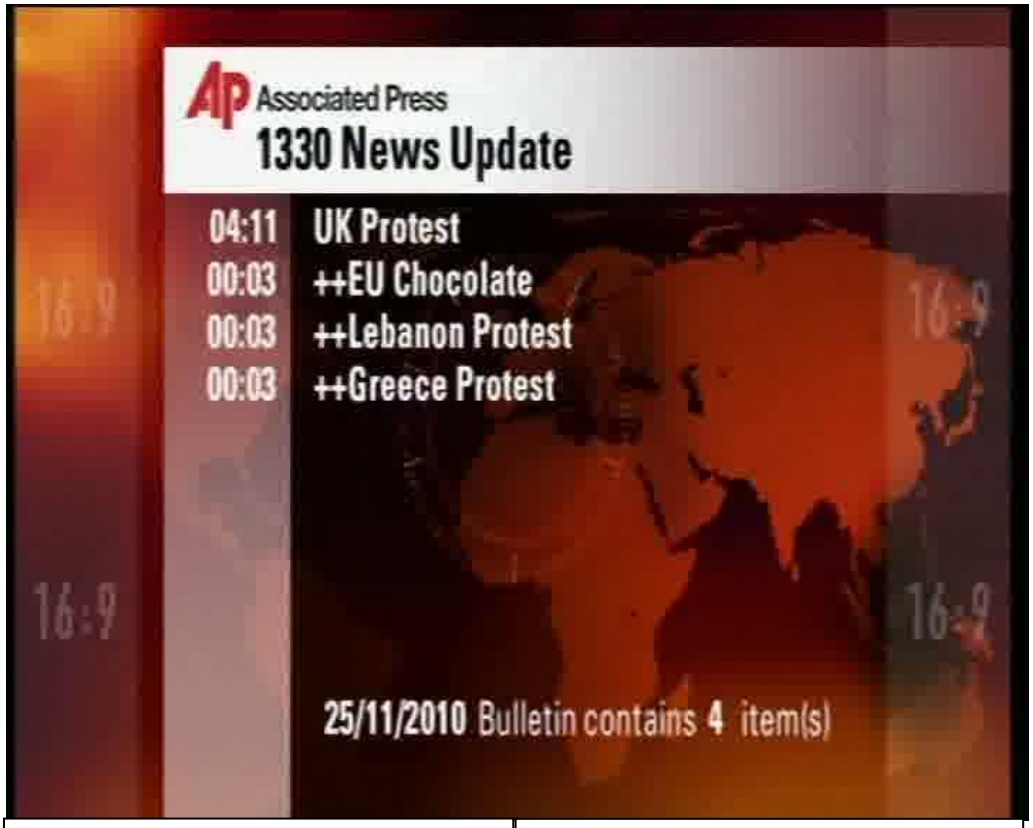
5. Resultados

La siguiente muestra de resultados contiene imágenes capturadas por el detector de carátulas. El sistema OCR final aplicado utiliza los parámetros por defecto, sin ningún entrenamiento específico y con un único paquete de idioma: el inglés.

Los resultados muestran la imagen capturada y la salida del OCR. A la izquierda, la salida que devuelve el sistema teniendo como entrada la imagen capturada. A la derecha, el resultado tras la segmentación y binarización de la imagen capturada.

5.1 Carátula APTN

En el primer resultado es un ejemplo de la mejora de prestaciones del sistema. Ambos resultados son inteligibles pero el segundo muestra una mejor comprensión si no vemos la imagen primero.



UKP|otest	UK **Pr**otest
+|~£U Chocolate 1	**++EU** Chocolate
-Hlehanon Protest	**++Leh**anon Protest
++Greece Protest	++Greece Protest
251112010 Bulletin contains I item1S1	25/11/2010 Bulletin contains -4 itemlsl

5.2 Carátula EFE

El siguiente ejemplo muestra cómo actúa el sistema ante un idioma para el cual no ha sido entrenado ya que a priori solo está preparado para el inglés. El resultado realmente está más condicionado por el tamaño de los caracteres, como hemos visto en la etapa de binarización, que por el propio idioma. Las etapas de preprocesado (localización y binarización del texto) quedan justificadas a la vista del resultado.

Es destacable el mal reconocimiento de los signos de puntuación, como el carácter ":", para el cual el sistema lo reconoce como un "2". Así como la palabra "temperaturas" reconocida como "temperaturas". Se trata de un claro ejemplo en el que un entrenamiento específico mejoraría los resultados.

TiTULO: Bias bempenlhlns enV&doid	TiTULO: **Baias temperatures en Valladolid**
LOCALIDADZ V3||3d0|iC|	LOCALIDAD2 **Valladolid**
PROVINCIN	PROVINCIA2
FECHN 25/11/2010	FECH**A**I 25/11/2010
DURACIONZ 00203239217	DURACION2 00203239217
Agencia EFE	Agencia EFE

5.3 Carátula SNTV

En este caso la imagen, siendo sintética al igual que el resto, presenta una mejor calidad visual. Esto se traduce en un buen reconocimiento en ambos resultados, aunque existen algunas mejoras cuando la entrada es preprocesada.

Soccer AFC Award 01:16	Soccer AFC Award 01:16
Soccer Palmeiras 01:57	Soccer Palmeiras 01:57
Soooer Beckham 01:00	**Soccer** Beckham 01:00
NBA Phoenix 01:59	NBA Phoenix 01:59
NHL Montreal 01:59	NHL Montreal 01:59
NHL **‘lem pa** Bay 01:55	NHL **Tenm pa** Bay 01:55

5.4 Carátula APTN (2)

La siguiente muestra ha sido seleccionada para demostrar lo que probablemente sea debido a la inteligencia del sistema OCR: a pesar de ser una imagen de calidad, tamaño y color de texto similar que la del resultado 1, la salida es casi perfecta sin necesidad de preprocesamiento. Esto posiblemente sea debido a que al haber mayor cantidad de texto, el sistema es capaz de aprender mejor sus características. Por otro lado, el resultado final acaba de corregir los errores, dando lugar a un resultado ideal.

UK Auction	UK Auction
REPLAY Original Darth Vader costume	REPLAY Original Darth Vader costume
to go under hammer	to go under hammer
London, 23 November 2010	London, 23 November 2010
02:57	02:57
666148	666148
Check script **ior** details	Check script **for** details
AP Television	AP Television
English/Nat	English/Nat

5.5 Carátula APTN (3)

Como hemos visto en el apartado de binarización, esta carátula que contiene fondo en movimiento dificulta el proceso de extracción.

Gracias a la binarización palabra por palabra podemos reducir este problema, ya que cada zona de texto la tratará independientemente del resto, es decir, un umbral para cada región. No obstante, palabras largas como “East/Europe” que contienen cambios bruscos de fondo inevitablemente van a ser mal binarizadas, como ya hemos visto, y por tanto mal reconocidas.

13:20 Ce1ebrj1y	13:20 **Celebrity** Extra”
13:30 News Update	13:30 News Update
13:55 earthTV	13:55 earthTV
14:00 GMS	14:00 GMS
14:30 Pfamemf- Mid	14:30 **Prime News** - Mid EastfEur
14:55 ear0aTV	14:55 earthTV

VIII. Conclusiones

En este Proyecto Final de Carrera se han presentado dos importantes herramientas para la indexación de video. El objetivo consistía en confeccionar un sistema que de forma automática seleccionase qué es relevante y qué se puede omitir en el proceso de almacenamiento de video. Los resultados aportados dan idea de la viabilidad de la senda seguida.

En primer lugar, el extractor de KeyFrames consigue agrupar en pocas instantáneas el contenido de una secuencia audiovisual. El detector de cambios de escena, aún no presentaba suficiente autonomía para decidir una imagen representativa pero sí para delimitar desde qué instante hasta cuál era necesario escoger uno. Sin embargo, con la adhesión de los diferentes bloques de detección (caras, texto, borrosidad) y de su capacidad para discernir entre una imagen u otra, se ha conseguido una herramienta que facilitará muchos procesos que suelen venir a continuación, entre ellos, el de almacenar y recuperar el contenido.

A partir de este punto, cualquier proceso que quiera recuperar un video a partir de una búsqueda textual o gráfica (búsqueda a partir de otra imagen o query by example) hecha por un usuario, podría actuar directamente sobre los KeyFrames. Esto reducirá tiempos de cálculo y por tanto, costes económicos. Además, el sistema modular permite que se le añadan nuevas funcionalidades, como por ejemplo un reconocimiento de caras, el cual puede tan sólo analizar los KeyFrames para saber qué personas aparecen en un video.

En segundo lugar y enlazando con lo anterior, el detector de carátulas de agencia y la posterior extracción de texto muestran que se puede archivar automáticamente una secuencia que proviene de una emisión en vivo. Mediante un entrenamiento específico del sistema OCR para contenidos e idiomas habituales, los resultados pueden mejorar sustancialmente. Al obtener la información en una portada de lo que se va a ver, podemos prever el contenido del video. Además, si lo unimos al extractor de KeyFrames, crearemos un conjunto de imágenes y texto con toda la información relevante. Esto acelera los procesos de producción y posterior emisión en el ámbito televisivo.

En ambos casos, conviene seguir trabajando en la mejora de cada detector, debemos tener en cuenta que el contenido audiovisual que se quiere analizar puede llegar a tener varias décadas. Este hándicap implica que la fuente del video puede ser de menor calidad, por lo que repercutirá en la detección de falsos positivos y falsos negativos y éstos condicionarán los resultados.

En definitiva, en el marco económico en el que nos encontramos y las constantes noticias sobre recortes en el sector público entre los cuales se encuentran los medios de comunicación (locales, autonómicos, nacionales…) públicos, este conjunto de herramientas supone un pequeño respiro por la reducción de tiempo y personal que genera. Permite entonces la destinación de recursos a otros procesos de la cadena, como puede ser la propia investigación para el desarrollo de nuevos módulos acoplables al sistema según las necesidades de la compañía que, a su vez, repercutirá en otra reducción de recursos destinados a estas tareas.

IX. Trabajo Futuro

Una vez analizados los resultados de ambas herramientas es necesario seguir trabajando en diferentes partes.

Por un lado, el extractor de KeyFrames, aunque presenta resultados muy satisfactorios en diferentes secuencias de video, es posible que en otros tipos no consiga extraer los mejores frames. Podría ser necesario la adhesión de un nuevo criterio de selección o la modificación del proceso de selección por uno menos jerárquico.

Cabe destacar que cualquier mejora en alguno de sus módulos repercutirá en una mejora de resultados. Esto es especialmente esencial en el detector de texto, ya que el sistema actual presenta un número considerable de falsos positivos. Si conseguimos mejorar esta detección o sustituirla por otra implementación, podremos ahorrarnos el paso por la confección de mapas de calor, que ralentizan el cálculo y obtendremos valores con mayor confianza de la presencia de texto.

Una alternativa a introducir en este caso, es la reciente incorporación a la librería Image+ de un método basado en la publicación de Microsoft Research *Detecting text in natural scenes with stroke width transform* [18]. El método, aun en fase de desarrollo por parte del Grupo de Procesado de Imagen de la UPC, presenta resultados prometedores en cuanto a la detección de texto, basándose en la estimación del grosor de los caracteres y su uniformidad.

En relación con lo anterior, la etapa de detección y extracción de texto en carátulas de agencia mejoraría sustancialmente. Implicaría una binarización de forma trivial, ya que se detectarían directamente los caracteres. Por tanto, la salida del sistema OCR podría obtener mejores resultados.

Por otro lado, debido a diferentes factores, se ha creado una oportunidad de negocio en el complicado sector audiovisual. El factor más importante es la desaparición de la empresa que daba soporte al código de la CCMA (Televisió de Catalunya) y la necesidad de establecer un vínculo entre CCMA y la UPC que diese soluciones en forma de productos. Tras el tiempo dedicado a la implementación de herramientas como éstas y la relación establecida tanto con el personal de la CCMA y el Grupo de Procesado de Imagen, Miquel Àngel Farré y un servidor decidimos aprovechar la oportunidad y establecer una nueva empresa que dé salida y soporte a nuevos productos realizados en la UPC.

X. Referencias

[1] M. Swain and D. Ballard, Color Indexing, International Journal of Computer Vision 7:1, 11-32 (1991)

[2]E.Weyuker, F.Vokolos, Experience with performance testing of software systems: issues, an approach, and case study, IEEE Transactions on Software Engineering, p.1147-1156, December (2000).

[3] J. Canny, A Computational Approach to Edge Detection, IEEE Trans. on Pattern Analysis and Machine Intelligence, 8(6), pp. 679-698 (1986).

[4] F. Escolano, O. Colomina, M.A.Cazorla. Visión Artificial: Extracción de Características I, 2006.

[5] Hill Green. Canny Edge Detection Tutorial. 2002.

[6] M. Leon, V. Vilaplana, A. Gasull, F. Marques, Region-based caption text extraction, Proceedings of WIAMIS 2010, 11th International Workshop on Image Analysis for Multimedia Application Services, Desenzano del Garda, Italy, 2010.

[7] V.Vilaplana, F.Marqués, P.Salembier, Binary Partition Trees for Object Detection. IEEE Transactions on Image Processing, vol. 17, no. 11, pp.2201–2216, November (2008)

[8] S.Li, A. Jain, Handbook of Face Recognition, Ed. Springer Science (2005)

[9] E.Hjelmas, B. Kee, Face Detection: a Survey, Computer Vision and Image Understanding Volume 83, Issue 3, p. 236-274, September 2001,

[10] G.Yang & T.S.Huang. Human Face Detection in Complex Background, Pattern Recognition, vol. 27, no. 1, pp. 53-63, 1994

[11] K.C.Yow & R.Cipolla. Feature-based human face detection, Image and Vision Computing, vol. 15, no. 9, pp. 713 – 735, 1997.

[12] A.L.Yuille, P.W.Hallinan & D.S.Cohen. Feature extraction from faces using deformable templates, International Journal of Computer Vision vol. 8, no. 2, 99–111, 1992.

[13] P.Juell & R.Marsh. A hierarchical neural network for human face detection, Pattern Recog. 29, 1996, 781–787.

[14] M.Schulze, K.Scheffler, & K.W.Omlin. Recognizing Facial Actions with Support Vector Machines, In Proc. PRASA, pp. 93-96, 2002.

[15] L.R.Rabiner. A tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, Proceedings IEEE, vol. 77, no. 2, Feb 1989.

[16] M.A.Turk & A.P.Pentland. Face recognition using eigenfaces, Proceedings of the IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, pp. 586-591, Maui, Hawaii 1991.

[17] B.Zarit, B.J.Super & F.Quek. Comparison of five color models in skin pixel classification, ICCV'99 Int'l Workshop on Recognition, Analysis and Tracking of Faces and Gestures in Real-Time Systems, pp. 58–63, Corfu, Greece, September 1999.

[18] P.Viola & M.Jones. Rapid Object Detection using a Boosted Cascade of Simple Features, Computer Vision and Pattern Recognition, 2001.

[19] C.H.Lee, J.S.Kim & K.H.Park. Automatic human face location in a complex background, Pattern Recognition Letters, 29, 1877–1889, 1996.

[20] R.J.Qian, M.I.Sezan & K.E.Matthews. A Robust Real-Time Face Tracking Algorithm, ICIP (1) 1998: 131-135.

[21] Yoav Freund and Robert E. Schapire. A short introduction to boosting. Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, pages 1401-1406. Morgan Kaufmann, 1999.

[22] B.Epshtein, E.Ofek and Y.Wexler, Detecting Text in Natural Scenes with Stroke Width Transform, Microsoft Corporation.