Universidad de la República Oriental del Uruguay

Facultad de Ingeniería

SISTEMA PCP

PROYECTO DE FIN DE CARRERA

Montevideo Uruguay

Agosto de 2003

Nicolás Scaglione
Martín Thomas
Pablo Varela

Tutor, Ingeniero Electricista Álvaro Pardo

ÍNDICE

Capítulo I
Prototipo de Software para Auditar la Difusión de Pautas Publicitarias: el PCP.....1

Capítulo II
Fundamentos Teóricos y Prácticos para la Implementación de PCP.....6

Capítulo III
Umbrales de Decisión para el Algoritmo de Detección de Cambios Abruptos de Escena.....19

Capítulo IV
Pruebas del Sistema PCP.....25

Capítulo V
Reporte de gestión de proyecto.....27

Apéndice A
Manual de usuario.....31

Apéndice B
Archivos Bitmap.....36

Apéndice C
Análisis y Diseño en UML.....38

Apéndice D
Plan Inicial del Proyecto Entregado Previo a su Inicialización.....47

CAPÍTULO I

Prototipo de Software para Auditar la Difusión de Pautas Publicitarias:

El PCP

Nicolás Scaglione, Martín Thomas, Pablo Varela.

***Abstract*- Los interesados en publicitar bienes y servicios en TV, generalmente empresas, invierten grandes sumas de dinero en publicidad televisiva. Esta situación, crea una demanda para disponer de un contralor efectivo del cumplimiento por parte de los canales de TV de sus obligaciones contractuales, en lo que respecta a la cantidad de apariciones, duración y horarios acordados para la difusión de las pautas publicitarias. Se describe a continuación un sistema, basado en un software, que responde a esta demanda registrando las apariciones de una pauta publicitaria preseleccionada en una secuencia de video.**

Índice de términos

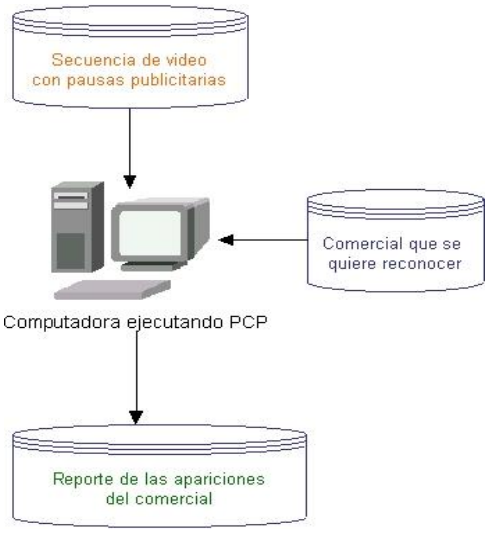
PCP –Prototipo de Control de Pautas, es el nombre del software.
Shot - Análogo a Escena.
Traza - Información sobre el comercial 'aprendido'.
Histograma - método cuantitativo para obtener la distribución de los colores sobre los píxeles de la imagen.
CAE – Cambios Abruptos de Escena.
CGE – Cambios Graduales de Escena.

I. INTRODUCCIÓN

El PCP, prototipo de control de pautas, esta orientado a reconocer comerciales predeterminados por el usuario con un alto grado de confiabilidad. Esta herramienta permite auditar el cumplimiento de los contratos con los medios de difusión televisivos, en lo que refiere a la cantidad de apariciones, duración y horarios de los comerciales. Se beneficia entonces a todos los actores involucrados directa o indirectamente en la difusión de pautas publicitarias. Los anunciantes, pueden tener un control veloz y veraz de la difusión de la publicidad que contratan. Las agencias de publicidad y los intermediarios entre los anunciantes y los medios de difusión televisivos pueden automatizar el control de sus pautas y ofrecer a sus clientes las garantías necesarias. Por último, los medios de difusión televisivos poseen una herramienta para respaldar la calidad de su servicio. Eventualmente se podrán hacer usos misceláneos del sistema con otro tipo de objetivos, citando un ejemplo, en estudios de mercado, el PCP facilitaría analizar las características con que se difunden ciertos comerciales en las pausas publicitarias.

El sistema consiste en un software que procesa archivos de video que contienen pausas publicitarias. (Fig. 1)

Primero, el usuario selecciona un comercial y se lo presenta al



sistema, para que este lo 'aprenda'.

Segundo, el sistema procesa un video, reconociendo las apariciones del comercial previamente 'aprendido'.

Tercero, el sistema genera un reporte de las apariciones del comercial en este archivo de video.

La aplicación se distribuye en tres módulos que realizan las tareas anteriormente descriptas. (Fig.2)

Módulo de aprendizaje: Sirve para que un usuario le enseñe al sistema el comercial que debe buscar en las pausas publicitarias.

Módulo de reconocimiento: Busca el comercial aprendido en las pausas publicitarias.

Módulo de reporte: Genera un reporte indicando apariciones del comercial.

II. TÉCNICAS UTILIZADAS

A. Escenas y cambios de Escenas

Se comienza definiendo algunos términos que se utilizan. Esto es necesario para la comprensión de las operaciones que realiza el sistema.

Desde el enfoque de este estudio, un video esta dividido en escenas. Se considera como escena aquel segmento de video en que se percibe continuidad en la secuencia de cuadros. Entonces se considera un cambio abrupto de Escena cuando entre dos cuadros consecutivos, se produce un cambio en el contenido de la imagen de uno a otro. Esos cambios, se deben fundamentalmente a variaciones de contexto del video, o a cambios en los ángulos de visualización del mismo contexto.
Los cambios de escena se clasifican en 2 clases: Cambios de Escena Abruptos y Graduales, CAE y CGE.

B. Módulos del Sistema

El sistema consta de 3 módulos, Aprendizaje, Reconocimiento y Reporte. Cada uno asociado a un modo de funcionamiento.

A continuación se describe la operativa normal del sistema en cada modo.

Modo Aprendizaje: El usuario selecciona un archivo que contiene el comercial de interés, el sistema lo segmenta en escenas automáticamente y lo despliega a través de una interfaz gráfica. El usuario selecciona aquellas escenas que componen el comercial y la aplicación genera automáticamente un archivo con información sobre el comercial seleccionado que identifica a este anuncio. A este archivo se le llama Traza. (Fig.3)

Modo Reconocimiento: El usuario selecciona un archivo de video, que contiene las pausas publicitarias donde se buscan las apariciones del comercial. Acto seguido inicia la búsqueda y la aplicación comienza el procesamiento.

El sistema procesa el archivo, indicando a través de su interfaz, las apariciones del Comercial. Finalmente genera

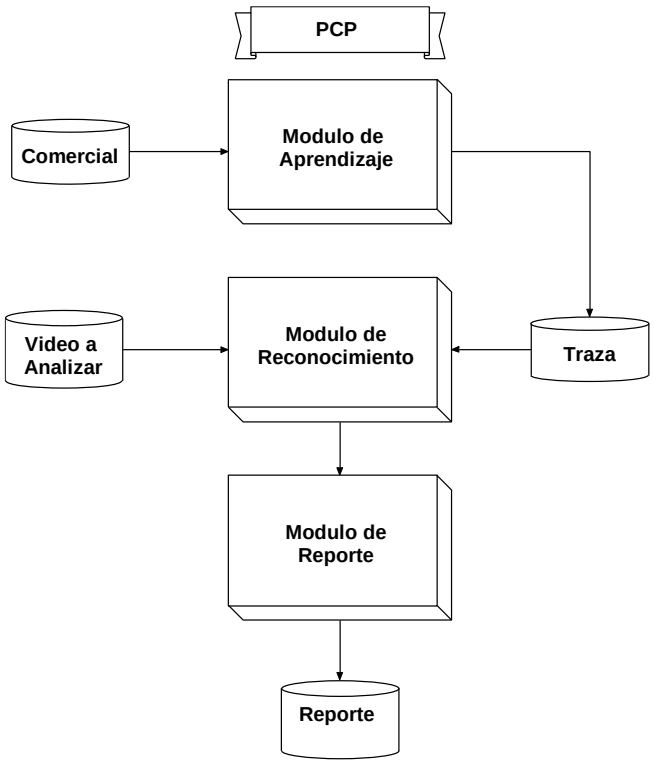


Fig.2 Diagrama mostrando los módulos que utiliza el PCP

un archivo con información sobre el resultado de la búsqueda. (Fig. 4)

Modulo de Reporte: En base a la información registrada por el módulo de reconocimiento, genera un archivo con información de

interés sobre el resultado de la búsqueda en un formato difundido y permite su cómoda visualización.

Para tener una visualización de las interfaces, dirigirse al manual de usuario en el *apéndice A.*

C. El algoritmo de detección de CAE

Se implementa en el módulo de Aprendizaje y el de Reconocimiento. Analiza el video midiendo y haciendo comparaciones entre ciertas características de imagen de los cuadros. Estas son: Distribución espacial de la luminancia en una imagen, distribución de los colores sobre los píxeles de la imagen (histogramas), movimiento de un cuadro a otro a través del estudio de variación de bordes, y estadística sobre escenas estáticas y dinámicas según variación de bordes en el contexto.

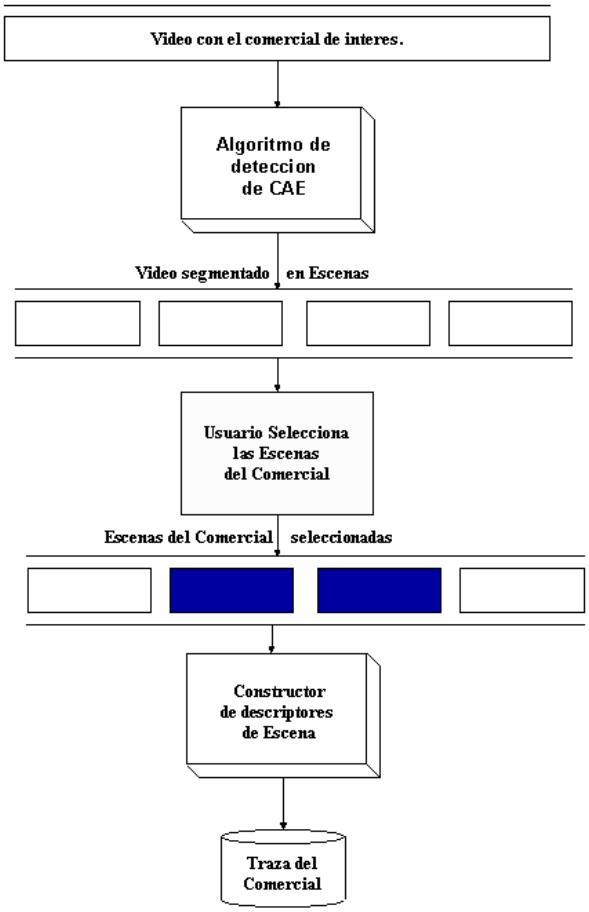


Fig.3 Descripción esquemática del proceso de Aprendizaje

D. Descriptores de escena

Es una técnica que se implementa en el módulo de Aprendizaje. Los descriptores de escenas están compuestos de histogramas especiales y otros datos que caracterizan cada escena. Estos histogramas contienen un resumen de la información de la escena y cumplen con una condición importante, permiten que el módulo de Reconocimiento realice su tarea velozmente en base a ellos.

E. Traza

A partir de las técnicas anteriores se genera la traza, que como concepto es la información total que se extrae del comercial y que el sistema almacena en un archivo. La traza contiene un descriptor de escena por cada escena del comercial y es la fuente de información para el reconocimiento.

F. Algoritmo de Reconocimiento

Se implementa en el módulo de Reconocimiento, encuentra las apariciones del comercial aprendido en un video. Utiliza también el algoritmo de detección de CAE, segmentando en escenas el video donde se hace la inspección. Cada vez que el algoritmo detecta un cambio de escena, empieza a comparar la escena que finaliza con ese cambio con los descriptores de escena correspondientes contenidos en la traza. De esta manera, va registrando si hay coincidencias hasta llegar al final del video inspeccionado.

La lógica y técnicas de comparación son parte medular del algoritmo, y se destaca su flexibilidad como un atributo importante para modificar la robustez del algoritmo en función inversa de la velocidad.

Los detalles técnicos y matemáticos de estas técnicas y algoritmos se pueden consultar el *capitulo II.*

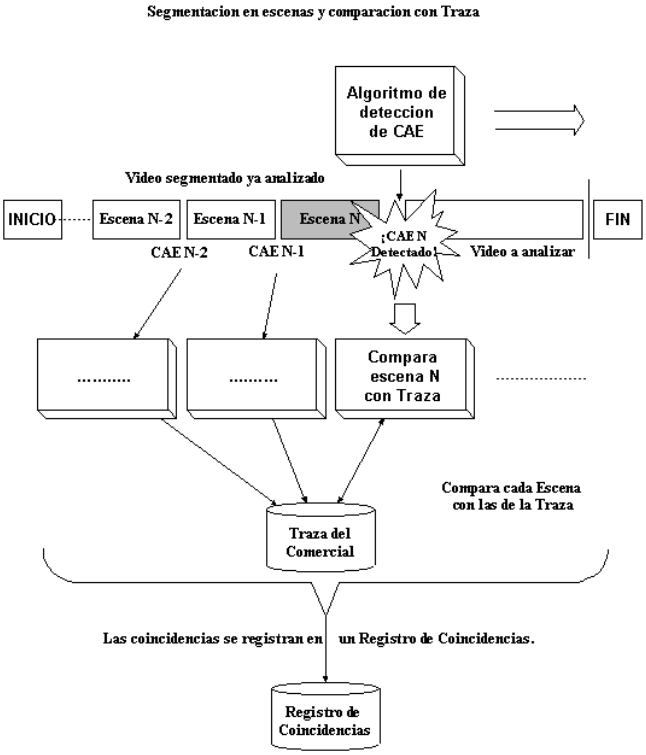


Fig. 4 Descripción Esquemática del proceso de Reconocimiento.

III. MÉTODO Y TECNOLOGÍAS UTILIZADAS

A. Análisis y Diseño

Para el estudio del problema a resolver y el diseño del sistema se realizó un análisis orientado por la metodología de diseño del lenguaje UML. La sigla significa en inglés 'Unified Modeling Language', y se traduce literalmente como Lenguaje Unificado de Modelado. Es un estándar para planificar, analizar, diseñar y especificar sistemas implementados en software.

En la primera etapa, sin disponer de soluciones a los problemas planteados, el UML permitió conceptualizar el problema, dividirlo en módulos y orientar sobre las clases necesarias para la implementación. En el *apéndice C*, se muestra el detalle de los desarrollos realizados.

B. Técnicas de tratamiento de Imágenes

Los algoritmos utilizados, y los descriptores de escena que conforman la traza, se basan en técnicas de tratamiento de imágenes. Estas técnicas, fueron desarrolladas y probadas durante el transcurso del proyecto.

El algoritmo de detección de CAE fue el presentado en [1]. El artículo describe el algoritmo y muestra sus resultados. Para su implementación se le introdujeron modificaciones que lo adaptaron a nuestras necesidades y resolvieron la problemática específica de nuestro caso de estudio. Se trabajó con una ventana adaptativa, que no estaba prevista por el algoritmo y que fue un aporte importante para estabilizar y dar coherencia a la detección de 'escenas' tanto en modo Aprendizaje como en modo Reconocimiento. Además permitió resolver el problema de los cuadros negros entre comerciales, técnica usada por algunos medios de difusión de TV. Se hicieron implementaciones propias para la detección de bordes y su dilatación (máscaras de Sobel), con el fin de evaluar movimientos en las escenas. Vale aclarar que en el documento no se hace referencia a los valores numéricos que se toman para implementar el algoritmo, por lo que fue una tarea importante determinar los umbrales que requiere el mismo. Para consultar las técnicas utilizadas para la determinación de los umbrales, ver *capítulo III*.

Para la elección de los descriptores de escena, se estudiaron los artículos[2],[3] y se adoptaron criterios propios. Estos artículos por si solos no abordaban la completitud del problema en cuestión, por lo que se reunió una serie de técnicas de cada uno además del aporte de nuestra parte. En este punto se abordó una solución de compromiso entre efectividad versus velocidad.

El algoritmo de reconocimiento fue diseñado e implementado por nosotros, es muy flexible a modificaciones que aumenten su velocidad o mejoren su eficiencia. En el capítulo X se detallan las implementaciones y los fundamentos teóricos de los descriptores de escena, de los algoritmos de detección de CAE y de Reconocimiento.

C. Decisiones de Implementación

1) Lenguaje de Programación

Se eligió C++ como lenguaje de programación. ¿Por qué orientado a objetos?, las características y dimensión del problema, sumado al mayor conocimiento de nuestra parte definieron este aspecto. Por razones de performance en materia de tiempos de ejecución, se prefirió trabajar en C++ frente a otros lenguajes orientados a objetos como Java. Se tuvo en cuenta la exigencia en este sentido, ya que el sistema realiza procesamiento de imágenes.

El entorno de desarrollo utilizado fue el Microsoft Visual C++ 6.0 porque se disponía del mismo.

2) Sistema Operativo

Se optó por el sistema operativo Windows en su versión 98 o superiores. La razón fundamental de esta elección, fue la experiencia que tenía el grupo sobre Windows. Otra opción de interés como Linux, sobre el que se tenía escasa experiencia, hubiera impuesto tiempos mayores para desarrollos equivalentes. Se prefirió economizar tiempos en este aspecto. Otra razón importante es el destino final del software a desarrollar. En este sentido se pensó en una aplicación con fines fuertemente comerciales, entonces, dada la mayor difusión de Windows en este tipo de empresas, la aplicación sería más simple de manejar para un operador PC medio, en general ya familiarizado con Windows. Se tomó en cuenta también la utilidad potencial de la MFC (Microsoft Fundation Class) que ofrece el uso de librerías para manejo de archivos de video.

3) Tecnología de video digital utilizada

Debido a que la información que maneja el sistema es video digital, se hizo un estudio de los formatos existentes. Sobre esto se identifican dos elementos diferentes, el formato del archivo, y el formato del video.

El formato del archivo elegido fue AVI ('Audio Video Interleave') por ser el mas difundido para audio y video sobre Windows. Se utiliza la librería vfw32 disponible en el Microsoft Visual Studio, para resolver la accesesibilidad a los archivos de videos .avi y otras facilidades relacionadas.

La elección del formato de video se tomo en segunda instancia y quedo condicionada al formato AVI ya elegido. Ya se estaban utilizando previamente funciones que disponían de algunos *Codecs* para versiones de los mecanismos de compresión *Cinepak* e *Indeo*. Se consideró de especial interés que el sistema soportara video comprimido, analizándose usar el formato MPEG-2. MPEG-2 es un formato muy difundido (es utilizado en TV digital y DVD), hay mucha información disponible y se tenían conocimientos sobre su funcionamiento. Utilizar formato comprimido con MPEG-2 fracasó por la incompatibilidad de tener un avi con un video comprimido en MPEG 2 en su interior. En los casos de *Indeo* y *Cinepak*, se hicieron las pruebas necesarias y se logró un buen desempeño.
Se optó entonces por desarrollar la aplicación para trabajar con video comprimido en *Indeo*, *Cinepak* y descomprimido. De todas formas, el bajo acoplamiento interno entre las clases de software de la aplicación, permitiría en el futuro utilizar otro tipo de archivos de video solo con agregar las clases adecuadas para su manejo.

Problemas en la programación: Para la implementación en C++ existieron varias dificultades. Poca experiencia de los integrantes del grupo en programación y escasa información para manejar archivos de video. No se encontró bibliografía recomendable, e Internet fue la principal fuente de información. La MSDN ('Microsoft Developer Network'– Documentación de Microsoft) contiene información sobre muchas librerías. Es de gran ayuda si se tiene la estructura de lo que se va a implementar, o se esta entendiendo código ya hecho. Pero ayuda poco cuando se tiene que comenzar de cero con un tema específico.

Relacionado con AVI posee información sobre la librería VFW32 que es la que contiene todo los métodos usados para el manejo de archivos AVI.

D. Pruebas y Ensayos

Para pruebas de las técnicas seleccionadas e implementación de los algoritmos utilizados, se encontró necesario trabajar sobre un entorno de desarrollo preliminar. La implementación de los algoritmos y su prueba sobre C++ requiere mucho tiempo, por lo que no es una herramienta útil para experimentar. Se utilizó entonces el software matemático *Matlab 6.0* con sus librerías para procesamiento de imágenes. La elección de este entorno estuvo basada en tres razones fundamentales: conocimiento de la herramienta, facilidad y rapidez para implementar pruebas y facilidad de manejo de videos digitales. Esto permitió centrar los esfuerzos en la matemática y lógica del algoritmo. Los algoritmos fueron desarrollados e implementados en versiones preliminares, sobre *Matlab*. Fue importante probar y corregir los algoritmos, antes de comenzar a implementar versiones definitivas en C++.

Las pruebas de rendimiento del PCP se realizaron en el entorno de desarrollo *Matlab* y en la versión definitiva en C++. Es de destaque la notable mayor velocidad de funcionamiento de la algoritmia en C++ sobre *Matlab*. Para consultar sobre los ensayos

y pruebas de rendimiento realizadas, se pueden ver los procedimientos y resultados en el Capitulo 4.

IV. Resultados

En las pruebas realizadas el comercial aprendido es reconocido un 100% de las veces y no se reportaron falsos positivos.

La velocidad de ejecución se midió en un PC Pentium III Intel, de 800 Mhz, con 128 MB de memoria RAM, con sistema operativo Windows 2000.

V. Discusión

El sistema muestra un grado de precisión que hace confiable su uso, esto es debido a la robustez de información utilizada para generar los descriptores de escena y un exigente algoritmo de reconocimiento. Tanto en el contenido de los descriptores como en el algoritmo de reconocimiento se puede ser menos conservador y obtener buenos resultados mejorando la velocidad de ejecución y disminuyendo la utilización de recursos del sistema. Se mencionan enriquecimientos importantes al sistema: trabajar sobre video adquirido en tiempo real desde un dispositivo de captura (tarjeta digitalizadora), incluir la posibilidad de reconocer varios comerciales a la vez, trabajar con una base de datos de anuncios aprendidos para mayor eficiencia en su funcionamiento, y mejorar las interfaces gráficas y de reporte.

Resueltos los puntos anteriores, se logra un producto comercial que tiene amplios usos en el mercado.

Por último, la aplicación podría tener una interfaz de red que hiciera el sistema accesible vía Web, para obtener, por parte de los clientes, los reportes de apariciones de sus comerciales en línea.

A nivel de técnicas, el sistema podría incorporar un algoritmo para detectar cambios graduales de escena, esto ampliaría su uso, no solo a reconocer comerciales, sino escenas cualesquiera que podrían no ser un comercial sino parte de una película.

VI. Conclusión

El PCP orienta y demuestra la viabilidad de usar sistemas automáticos de este tipo con fines comerciales y legales. Las técnicas y tecnologías para esto, están disponibles.

Referencias

[1] Chung-Ling Huang and Bing-Yao Liao, “A Robust Scene-Change Detection Method for Video Segmentation” *IEEE Trans. Circuits Syst. Video Technol.,* vol. 11, pp. 1281-1288, 2001.

[2] A Mufit Ferman, A. Murat Tekalp, Rajiv Mehrotra “Robust Color Histogram Descriptors for Video Segment Retrieval and Identification” *IEEE Transactions on image procesing*, Vol. 11, NO 5, pp. 497-508, mayo 2002.

[3] Juan Maria Sanchez and Xavier Binefa “Audicom : A Video Analysis System for Auditing Commercial Broadcast,” *IEEE International Conference on Multimedia Computing and Systems*, Vol. 2, pp. 272-275, junio 7 -11 1999.

CAPÍTULO II

Fundamentos Teóricos y Prácticos para la Implementación de PCP

Nicolás Scaglione, Martín Thomas, Pablo Varela.

***Abstract*-Se explican y analizan los algoritmos y conceptos desarrollados y utilizados para la implementación del sistema. Comprende un bloque de detección de cambios abruptos de escena, un bloque de aprendizaje y generación de los descriptores de escena de los comerciales, y un bloque de reconocimiento de comerciales.**

I. INTRODUCCIÓN

Este documento tiene como propósito dar una explicación de los algoritmos utilizados en el desarrollo del sistema, para un lector con formación matemática, sin conocimientos específicos de procesamiento de imágenes. Se va a hacer referencia a los siguientes tópicos.

- Imágenes digitales.
- Formatos utilizados.
- Cambios de escena, definición y clasificación.
- Utilización de los cambios de escena en el sistema.
- Métricas para medir distancias entre cuadros. Diferencias de píxeles, de histogramas y variación de bordes. Clasificación de transiciones de escena orientada a la detección de CAE. Implementaciones.
- Algoritmo de detección de *cambios abruptos de escena*, diseño e implementación.
- La *traza* y los descriptores de escena. Histogramas del cuadro más representativo, descriptores de escenas basados en histogramas promedio, medios, de promedio ponderado por alfa e histogramas intersección.
- Técnica de compensación de corrimientos. Histograma intersección con corrimientos.
- Algoritmo de reconocimiento.

II. FUNDAMENTO TEÓRICO

Las imágenes digitales, pueden sintetizarse como un mapa, en forma de cuadricula de puntos o elementos llamados píxeles. Se pueden dividir en dos clases, imágenes vectoriales o imágenes de mapa de bits. Las imágenes de mapa de bits o bitmap, tal como sugiere su nombre se construyen describiendo cada uno de los puntos que componen la imagen y llevan, por tanto, información acerca de la posición absoluta y el color de cada uno de ellos, es decir que a cada píxel se le asigna un valor tonal, negro, blanco, matices de gris o color. Estos valores están representados en un código binario que típicamente es almacenado en una computadora. La ventaja que presenta este formato es la posibilidad de recoger una amplísima gama tonal, por lo que es el tipo adecuado para representar imágenes captadas de la realidad. A diferencia de las imágenes vectoriales, la variación de tamaño

supondrá modificaciones en la calidad, ya que el número de celdas que forman la imagen permanece invariable, por lo que un aumento del tamaño hace que el único recurso posible sea ampliar el tamaño de cada una de ellas. Frecuentemente, usando técnicas de compresión, se reduce el espacio en memoria requerido por estas imágenes. El proceso inverso se llama descodificación, y el conjunto de algoritmos que hacen los dos procesos se conocen como *Codec*. La ventaja de esta técnica es ahorrar memoria de almacenamiento o velocidad de transmisión para las imágenes. La desventaja es el trabajo y tiempo exigido a los procesadores para comprimir y descomprimir la información.

La resolución de una imagen digital se traduce en la capacidad de distinguir los detalles espaciales finos. Por lo general la frecuencia espacial a la que se muestrea la imagen digital, es un indicador de resolución, este es el motivo por el que los píxeles por unidad de longitud son términos que expresan la resolución de las imágenes digitales.

Las dimensiones en píxeles, son las medidas horizontales y verticales de la imagen, expresadas en píxeles.
La profundidad de bits es determinada por la cantidad de bits utilizados para definir el valor tonal de cada píxel. Cuando mayor sea la profundidad de bits, tanto mayor será la cantidad de tonos que se pueden representar. Las imágenes digitales se pueden producir en blanco y negro, a escala de grises o a color.
Si es bitonal esta representada por píxeles que constan de 1 bit cada uno, que pueden representar dos tonos, utilizando los valores 0 para el negro y 1 para el blanco o al revés.
Una imagen a escala de grises esta compuesta por píxeles representados por múltiples bits de información, que varían según su resolución.
Una imagen a color está representada por una profundidad de bits entre 4 y 32. En una imagen con 24 bits de profundidad, los bits se dividen en tres canales: 8 bits para el rojo, 8 bits para el verde y 8 bits para el azul, este formato es llamado RGB 24. Para representar los colores se utilizan las combinaciones de estos. En total se tienen 16,7 millones de valores de color diferentes. Por detalles sobre el formato bitmap, ver apéndice B.
Un video digital esta compuesto por una secuencia de imágenes digitales, llamadas cuadros. La adquisición y la reproducción de estas imágenes se realiza a una cierta tasa, que se expresa en cuadros por segundo.

III. FORMATOS UTILIZADOS

Formato de video digital utilizado: Sobre esto se identifican dos elementos diferentes, el formato del archivo, y el formato del video.
Como formato de archivo de video el sistema utiliza AVI("Audio Video Interleave") por ser el mas difundido para audio y video sobre Windows. Se utiliza la librería vfw32 disponible en el Microsoft Visual Studio, para resolver la accesesibilidad a los archivos de video .avi y otras facilidades relacionadas.
La elección del formato de video se tomo en segunda instancia y quedo condicionada al formato AVI ya elegido. Se consideró de especial interés que el sistema soportara video comprimido, analizándose usar el formato MPEG 2.
MPEG es un formato muy difundido y además es utilizado en TV digital y formato DVD. La factibilidad de usar formato comprimido con MPEG fracasó por la incompatibilidad de tener un avi con un video comprimido en MPEG 2 en su interior.

Reconociendo la importancia de trabajar con video comprimido, teniendo en cuenta que el orígen de todos los videos sería un disco de almacenamiento, y que los videos requieren de gran cantidad de espacio en bytes, se testearon otros codecs, logrando resultados positivos de desempeño con Indeo y Cinepak.

Se optó entonces por desarrollar la aplicación para trabajar con video comprimido en Indeo, Cinepak y descomprimido. De todas formas, el bajo acoplamiento interno entre las clases de software de la aplicación, permitiría en el futuro utilizar otro tipo de archivos de video.

Formatos de imagen digital utilizados: imágenes de mapa de bits, de 160x120 píxeles de dimensión. Ver apéndice B.

El sistema utiliza RGB con una profundidad de color de 24 bits. La tasa de muestreo es de 25 cuadros por segundo. Con estas características se considero que los tiempos de ejecución del sistema iban a ser razonables y cercanas al tiempo real.

Cuadro 1 Cuadro 2 Cuadro 3

Cuadro 4 Cuadro 5 Cuadro 6

Fig.2 Ejemplo de un *cambio gradual de escena*, en particular una disolución. Observar los cuadros 3 y 4 especialmente.

IV. Cambios de Escena, Definición y Clasificación

Para obtener mejores formas de indexación y procesamiento es necesario estructurar el video, descomponiéndolo en unidades de menor tamaño. A esas unidades se le denominan, desde un nivel semántico mas alto, escenas, y desde un nivel semántico mas bajo, shot.
Un shot es un conjunto de cuadros temporalmente vecinos y espacialmente conectados. Un shot representa una acción continua en el espacio, es decir, que se percibe continuidad en la secuencia de cuadros.
Existen varias formas de transición de un shot a otro, dependientes de la técnica de edición empleada. Las transiciones mas comunes en TV son:

- Cambios Abruptos de Escena (CAE), ver figura 1.
- Cambios Graduales de Escena (CGE), ver figura 2.

Dentro de los CGE, se distinguen fade-in, fade-out y las disoluciones. Un fade es una transición gradual entre una imagen constante y una escena (fade in) o entre una escena y una imagen constante (fade out), ver figura 3. Una disolución es una transición gradual entre una escena y otra, ver figura 3.

Fig.1 Ejemplo de un *cambio abrupto de escena*.

V. Utilización de los Cambios de Escena en el Sistema

A. *¿Por qué interesa distinguir los Cambios Abruptos de Escena?*

Segmentar el video en escenas o shots es necesario como punto de partida para analizar el contenido de un video, en particular para reconocer comerciales.
En una primera instancia, el problema de reconocer un cierto comercial dentro de una secuencia de comerciales, supone desde este enfoque, poder particionar la secuencia de video en todos los comerciales que la componen. A partir de la partición es posible analizar cada comercial por separado.
Se observa que las transiciones entre comerciales *(inter comerciales)* son CAE, lo cual es lógico ya que un CGE supondría alterar los cuadros finales e iniciales de dos comerciales adyacentes, lo que sería inadmisible para los anunciantes. Además, gran cantidad de transiciones dentro de los comerciales *(intra comerciales)* son CAE, lo que permite a su vez, segmentar cada

Fig.3 Ejemplo de un fade de escena, en particular un fade-out. El caso tiene la particularidad, de que parte de la imagen final ya se hallaba presente superpuesta a la imagen inicial.

comercial en escenas o shots. Esto es, al tener cada comercial segmentado a su vez en escenas, se generan *descriptores de escena*, información mas valiosa de cada comercial y herramienta fundamental para el posterior reconocimiento.
La obtención de estos *descriptores*, que se lleva a cabo durante el aprendizaje, busca dar características representativas de un *grupo de cuadros* dados, GdC.
Si el GdC se corresponde con un *shot*, es de esperar uniformidad en sus características, entonces se pueden extraer valores descriptivos generales del GdC que guarden una fuerte relación con cada uno de los cuadros del GdC.
Si el GdC no se corresponde con un *shot*, sino que es la concatenación de mas de uno, los valores descriptivos generales del GdC serán una solución de compromiso entre cuadros que pertenecen a diferentes escenas. Esto debilita la similitud entre los descriptores del GdC y los cuadros de este.
Este último análisis es una fuerte razón para detectar los CAE.

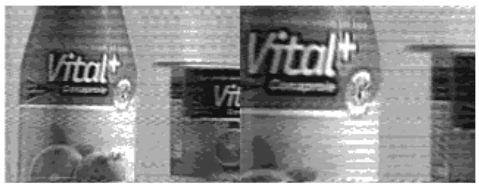
De los dos tipos de transiciones de escenas, el sistema distingue los CAE y no los CGA. Por lo expuesto anteriormente, la segmentación en CAE es necesaria y suficiente para los fines del PCP.

Para la segmentación en CAE, se tomo como base un algoritmo de detección de cambios abruptos de escena desarrollado en [1], al que se le introdujeron modificaciones para adaptarlo a los requerimientos de PCP. Este punto se aborda mas adelante en base al trabajo publicado por sus autores.

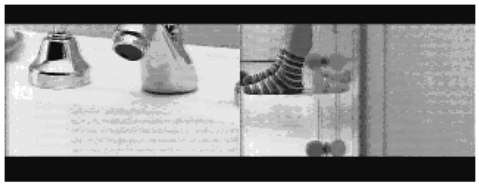
VI. Métricas para Medir Distancias entre Cuadros

Para segmentar una secuencia de video en *shots* se debe utilizar una medida de lo distinto que es un cuadro del siguiente. Esta medida debe presentar un valor alto, solo cuando los 2 cuadros pertenecen a 2 *shots* diferentes.

Se utilizan 3 medidas entre cuadros: Diferencia entre píxeles, diferencia entre histogramas y variación de bordes.



Cuadro 1 Cuadro 2



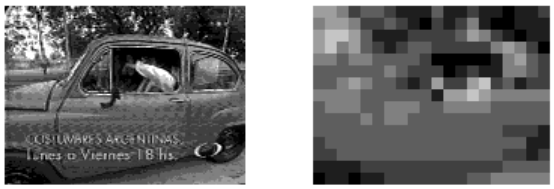
Cuadro 3 Cuadro 4

Fig.4. Entre los cuadros 1 y 2, en los que hay un acercamiento de cámara, se obtiene una diferencia de píxeles mayor que entre los cuadros 3 y 4 en los que hay un CAE. Esto muestra la debilidad de las métricas basadas en píxeles para detectar CAE.

A. Medición de diferencia entre píxeles

La diferencia basada en píxeles, surge de restar el valor codificado de cada píxel de un cuadro, con su equivalente en posición del cuadro siguiente. Luego se suman todas las diferencias y se obtiene la medida.
Dos cuadros se consideran pertenecientes al mismo *shot*, si su diferencia no supera cierto umbral. La diferencia entre píxeles es muy sensible al movimiento, ya sea de objetos o de cámara. Esto ocasiona que se puedan generar diferencias grandes donde no hay CAE, ver figura 4.
¿Qué se puede hacer para mejorar la sensibilidad al movimiento?
Una técnica útil es realizar difusión de cuadros. La difusión puede ser hecha en las imágenes antes de medir la diferencia de píxeles. La idea es subdividir las imágenes en bloques de 8x8 píxeles, y para cada bloque promediar los valores de píxeles. Con este procedimiento se disminuye la sensibilidad a pequeños movimientos. La imagen, así tratada es llamada *Imagen DC*, ver figura 5.
Los valores promedio de cada bloque vienen dados por $C_X(m,n)$ y $C_Y(m,n)$. Se consideran las funciones $f_X(k,l)$ y $f_Y(k,l)$, que representan para los cuadros *X* e *Y*, los valores de píxel para la posición (k,l). Para cada bloque de 8x8 se calcula,

Cuadro 1 Cuadro 2



Cuadro 3 Cuadro 4

Fig. 5. Ejemplo de difusión de imágenes para distintas dimensiones en píxeles. El cuadro 1 es una imagen de 384x288 píxeles, en el cuadro 2 se muestra la imagen difundida en bloques de 8x8 píxeles. El cuadro 3 es una imagen de 160x120 píxeles, mostrándose en el cuadro 4 el resultado de la difusión en bloques de 8x8 píxeles. A partir del cuadro 4 es difícil identificar el cuadro 3, dado que resulta en una matriz de solo 20x15 píxeles.

$$C_X(m,n) = \frac{1}{64} \cdot \sum_{i=0}^{i=7} \sum_{j=0}^{7} f_X(8m+i, 8n+j). \qquad (1)$$

De esta forma se obtienen las *Imágenes DC*. Luego para calcular la diferencia de píxeles se procede a hacer las restas, y a sumar sus valores absolutos tal que,

$$Dcvalor(X,Y) = \frac{\sum_m \sum_n |C_X(m,n) - C_Y(m,n)|}{k} \quad (2)$$

con *k* la cantidad de bloques de las imágenes DC.
En la implementación se le llama a dicha medida *DCvalor*.

B. Implementación de imagenDC

Para la implementación de *Dcvalor* fue necesario implementar un bloque de código que construyera la *imagenDC*. Para construir la *imagenDC*, se contempló el hecho de que la librería que se utilizamó estructura los bytes del cuadro extraído del stream de video en un vector.
Antes de comenzar a resolver el problema de construir la *imagenDC* se cuestionó si era necesario construir una matriz que portara los bytes respetando la geometría de la imagen. Construir esta matriz facilita la construcción de la *imagenDC* ya que esta matriz tiene una representación numérica de la imagen, a cada elemento de la matriz le corresponde el valor de luminancia de un píxel. La idea que surgió fue construir la *imagenDC* a partir del vector que contiene los bytes de la imagen original, con esto se evita perder tiempo en construir matrices que contengan los bytes de la imagen. Hay que resaltar que es de vital importancia optimizar esta operación ya que se usa un número de veces del orden de la cantidad de cuadros que se procesan. No es así el caso de las funciones que implementan la *prueba de escena estática* o *HValor* como se verá cuando se estudie el *algoritmo de detección de CAE*.

El precio que se paga con esta implementación es en la complejidad del diseño de esta parte del software.
Antes de proceder a la codificación de este bloque se simuló en matlab. Se corrió un programa que construye la *imagenDC* desde un vector que contiene los bytes de la imagen. Por otro lado se corrió un programa que construye la *imagenDC* a partir de una matriz con los bytes de la imagen de dos cuadros. Este último programa incluía la construcción de la matriz. Se comparo el desempeño de ambos códigos midiendo el tiempo que demoraban en construir *imagenDC* para el mismo cuadro. Se logró el doble de velocidad con el que construye la *imagenDC* a partir de el vector.

C. Medición de diferencia entre histogramas de color

Para tomar en cuenta las variaciones de color se usan diferencias entre los histogramas de cuadros consecutivos. Un histograma se puede tratar como un vector, *H*, de longitud *K*, siendo *K* el número total de colores posibles dados por la profundidad de bits para cada canal. Los valores *H(j)* son el número de píxeles de color *j* en el cuadro, con *j* entre 1 y *K*.
Esta es una representación en el dominio cromático, sin tener en cuenta su distribución espacial en el cuadro. Esto hace esta métrica más inmune a los movimientos rápidos, ya sean de objetos o de cámara. Si 2 cuadros tienen una diferencia entre histogramas menor a cierto umbral, se declara que pertenecen al mismo *shot*. Hay varios tipos de métricas para medir distancia entre histogramas. En nuestro caso se empleara,

$$Hvalor(X,Y) = \sum_{j=1}^{K} \frac{(H_X(j) - H_Y(j))^2}{\max(H_X(j), H_Y(j))},$$

si (3)

$$(H_X(j) \neq 0) \cup (H_Y(j) \neq 0)$$

o $Hvalor(X,Y) = 0$ *,en cualquier otro caso.*

$H_X(j)$ es el valor del j-ésimo elemento del histograma del cuadro *X* , y *K* es el número total de bins, esto es el número de tonalidades por canal posibles.

En la implementación se le llama a dicha medida *Hvalor*.
Esta medida es más robusta frente al movimiento que *DCvalor*, pues la medida de *Hvalor* no varía con la distribución espacial de los píxeles en los cuadros. Sin embargo un cambio de shot pero dentro del mismo contexto cromático puede engañar a esta medida. *Hvalor* es más sensible a variaciones en la luminosidad.

D. Medición de diferencias basadas en bordes

Una escena se dice estática si los movimientos que presenta son suaves y lentos respecto al tamaño de los cuadros, ya sean por los objetos contenidos en esta, como por movimientos de cámara. En una escena estática se produce una mínima variación de la información espacial y temporal entre cuadros consecutivos. Por el contrario en una escena dinámica existen movimientos rápidos, producidos por los objetos o por la cámara, pudiendo ser estos últimos, además de longitudinales o transversales, acercamientos o alejamientos.

Se observa, que durante una escena estática, las transiciones o bordes de las imágenes, se conservan cercanas a la ubicación espacial inicial. Este hecho sugiere una poderosa herramienta para detectar cuando una escena es estática. Si en dos cuadros consecutivos los bordes están lo suficientemente cerca, y esto se cumple durante toda una escena, ello significa que la escena es estática.

Para desarrollar el concepto anterior, se implementa la *prueba de escena estática*, que consiste en dos operaciones, detección de bordes y dilatación de bordes, ver figura 6. Esta prueba da como resultado un valor llamado radio de cobertura, *s*, indicativo de cuanto coinciden los bordes entre los cuadros. Para la detección de

Cuadro 1

Cuadro 2

Cuadro 3

Fig. 6 Al cuadro 1 se le aplica la operación de detección de bordes y el resultado puede verse en el cuadro 2. El cuadro 3 es el resultado de aplicar la operación de dilatación de bordes al cuadro 2, tomando r = 3.

bordes, se utilizan operadores de gradientes como las máscaras de Sobel. Para la operación de dilatación de bordes, se toma cada píxel de los bordes, y se dilata agregando un diamante blanco de altura y ancho (2 x r) + 1 píxeles, centrado en el píxel en cuestión, ver figura 6, cuadro 3. El parámetro *r* se denomina radio de dilatación.

Desde un enfoque matemático, la *prueba de escena estática* aplicada a dos cuadros, se describe de la siguiente manera.
Sean F1 y F2 los cuadros a medir. Primero se calculan los bordes de ambos cuadros. Luego se le aplica a los bordes de F2 la operación de dilatación de bordes tomando un *r* adecuado. Si los bordes calculados para F1 se movieron menos que el radio de dilatación *r*, entonces son cubiertos por los bordes dilatados de F2. Si se movieron mas que *r*, quedan fuera de los bordes dilatados de F2 y habrá menos cobertura.
El radio de cobertura s se define de esta forma:

$$s = 1 - \frac{\sum_{X,Y} E\{F_1(x,y)\}.\overline{E\{F_2(x,y)\}}}{\sum_{x,y} E\{F_1(x,y)\}} \qquad (4)$$

Donde $E\{\cdot\}$ es la operación de detección de bordes y el techo la dilatación de bordes.
Un bajo radio de cobertura significa que los cuadros son similares desde el punto de vista de los bordes. El radio *r* de dilatación impone cuanto movimiento relativo se tolera para considerar que una escena es estática. La transición entre 2 cuadros consecutivos con radio de cobertura mayor que cierto umbral se considera como una prueba de que la escena no es estática.

E. Implementación de Sobel

Para calcular una matriz con información de los bordes de una imagen utilizamos el algoritmo de *Sobel*. No hicimos hincapié en su optimización numérica ya que no es crítico el número de veces que se utiliza en el algoritmo de CAE.

El operador de Sobel implementa una medida del gradiente espacial en 2D sobre una imagen, con el objeto de enfatizar las zonas de la imagen donde hay bordes o transiciones. El operador consta de dos matrices, ambas mostradas en (5) .

$$G_X = \begin{pmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{pmatrix} \quad G_Y = \begin{pmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} \qquad (5)$$

Para mostrar como opera, supongamos que *G* es una matriz que representa la imagen que grafica los bordes y G^I la matriz a la cual se le quiere hallar los bordes. Así el operador de Sobel relaciona estas matrices según (6).

$$G = Sobel(G^I) \qquad (6)$$

La matriz *G* se obtiene según (7), siendo $G_{i,j}$ los elementos de *G* y $G_{i,j}^I$ los elementos de G^I.

$$G_{ij} = \left|G^I_{i-1,j+1} + 2G^I_{i,j+1} + G^I_{i+1,j} - G^I_{i-1,j-1} - 2G^I_{i,j-1} - G^I_{i+1,j-1}\right| + \left|G^I_{i-1,j-1} + 2G^I_{i-1,j} + G^I_{i-1,j+1} - G^I_{i+1,j-1} - 2G^I_{i+1,j} - G^I_{i+1,j+1}\right| \qquad (7)$$

Esto equivale a decir que se pasan dos máscaras, una rotada 90º con respecto a la otra, y luego se suman los valores absultos del resultado de pasar cada mascara.

Una forma de obtener una imagen con los bordes, es construir una imagen solamente con negros y blancos para que los bordes queden bien distinguidos. Esta imagen se construye a partir de los $G_{i,j}$, si $G_{i,j}$ supera cierto umbral, entonces se codifica un blanco en la matriz de bordes, de lo contrario se codifica un negro.

En la figura 7.b se muestra el resultado de aplicar sobel a la imagen mostrada en 7.a.

(7.a) (7.b)

Fig.7

F. Clasificación de transiciones de escena

Otra técnica utilizada para determinar si existe un CAE, consiste en la clasificación de la escena anterior y posterior al posible CAE. Aplicando la *prueba de escena estática* algunos cuadros antes y después donde se esta estudiando un potencial CAE, se pueden clasificar las transiciones en tres tipos:

- Escena estática a estática.
- Escena dinámica a estática o viceversa.
- Escena dinámica a dinámica.

A la cantidad de cuadros antes y después elegida se le denomina *alejamiento*.
Basado en datos experimentales y observaciones se resuelve que cuando se trata de transiciones dinámica-dinámica no se declara CAE, y en caso contrario sí. Comúnmente una transición dinámica – dinámica indica continuo movimiento de objetos o cámara. Esta técnica es utilizada para tomar decisiones en casos donde las métricas mencionadas anteriormente no arrojan valores concluyentes.

VII. ALGORITMO DE DETECCIÓN DE CAE

A. Presentación del algoritmo

La elección de este algoritmo surge de la búsqueda y estudio de diferentes técnicas para detectar cambios de escena. Se analizaron con este fin una serie de transaction del IEEE, [1], [3], [4].
Los criterios para elegir la técnica adecuada se basaron en:

- Simplicidad para la implementación.
- Adecuación a nuestras necesidades.

Estos criterios se fundamentan en que objetivo del proyecto no era desarrollar un algoritmo de detección de cambios de escena sino utilizarlo como herramienta para el reconocimiento de comerciales.

El algoritmo que se describe a continuación, fue incorporado y analizado de [1]. El algoritmo se basa en la detección de CAE locales a una ventana de cuadros deslizante que recorre la secuencia de video. El diagrama de flujo del algoritmo se muestra en Fig. 8.

Se declaran 2 tipos de CAE, genuino y ambiguo, con el fin de relajar el problema de los umbrales. El sistema consta de dos fases, la decisión final para los CAE declarados ambiguos se toma en la segunda fase.

La ventana es de vital importancia ya que determina el número de cuadros que se procesan por paso del algoritmo. Permite estudiar las diferencias entre cuadros, inmersas en un determinado contexto. Es decir, en un paso dado del algoritmo, con una ventana de tamaño N, se toman los N-1 *DCvalor* entre cuadros consecutivos. A partir de estos *Dcvalor* se eligen los dos mayores. Luego se obtiene *nratio* a partir del cociente de las mismas.
La ventana da un carácter de selectividad a esta etapa del algoritmo, filtrando aquellas diferencias entre cuadros que no son

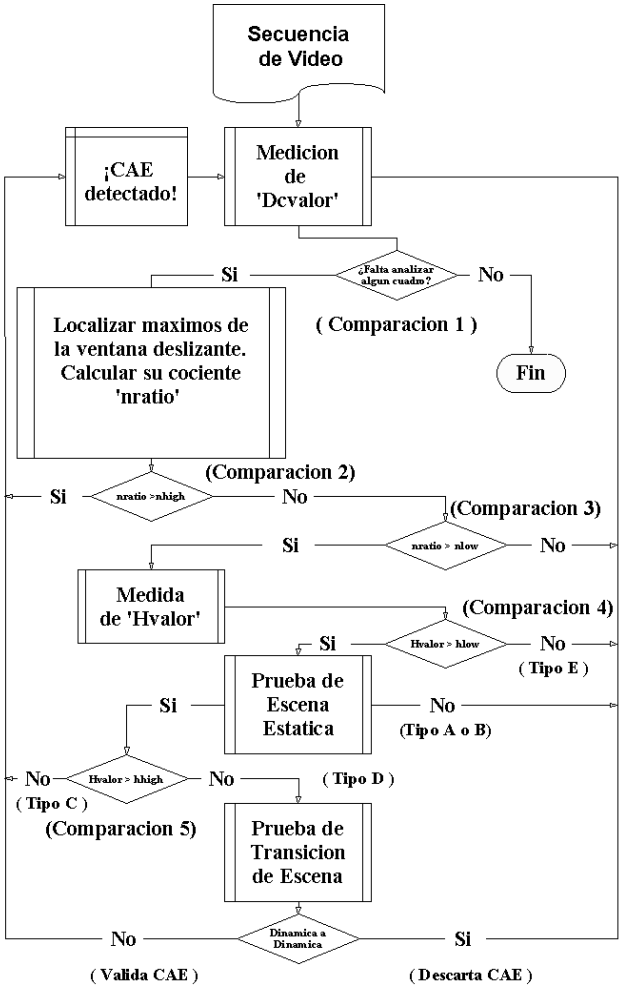


Fig.8 Diagrama de flujo del algoritmo con las medidas realizadas, las comparaciones con umbrales y los tipos de transiciones

relevantes.

Dos umbrales son usados para evaluar *nratio*, *nhigh*, y *nlow*. Si *nratio* supera *nhigh*, se declara un CAE genuino (Comparación 1

de Fig. 8). El parámetro *nhigh* es lo suficientemente alto como para no ser superado por rápidos movimientos de cámara, esto es

para evitar falsas alarmas. Si el *nratio* es muy pequeño, se asume que no hay ninguna transición, tal es el caso en que *nratio*< *nlow* (Comparación 2 de la Fig.8) Para CAE ambiguos con *nratio* entre *nhigh* y *nlow*, se pasa a la segunda fase.

En la segunda fase, se mide *Hvalor*, la diferencia entre histogramas entre los cuadros donde se registro el máximo de la primera fase.
Para *Hvalor* se usan 2 umbrales de comparación, *hhigh* y *hlow*, aunque no se usan a la vez como en el caso de *nhigh* y *nlow*. Si *Hvalor* es menor que *hlow* entonces se considera que no hubo CAE, (Comparación 3 de la Fig.7), que *Hvalor* no supere *hlow* es indicativo de una escasa variación de las características de color de la escena, este tipo de transición es clasificada como tipo E.
Si *Hvalor* es mayor a *hlow* se pasa a hacer la *medida de escena estática*.

El resultado de esta prueba, que es el radio de cobertura *s*, se lo compara contra un umbral *shigh*, si es menor que este, el resultado implica que la escena es estática. En caso que sea estática se declara que no hay CAE. Si resulta que la escena no lo es, entonces se pasa a comparar con el *hhigh* (Comparación 5 de la Fig. 8). Si resulta que *Hvalor* es mayor que *hhigh*, se declara un CAE. Se está ante un cambio significativo de colores entre cuadros, y no han coincidido los bordes.
En caso que *Hvalor* sea menor que *hhigh* nos encontramos con una escena con un *nratio* y *hvalor* posicionados en valores intermedios respecto a los umbrales correspondientes (*nlow* < *nratio* < *nhigh* y *hlow* < *Hvalor* < *hhigh*), y que no es estática. Para tomar una decisión, el algoritmo aplica la *clasificación de transición de escena*, que consiste en hacer 2 *mediciones de escena estática*, una antes de los cuadros del máximo, y otra después. Como ya se menciono los posibles resultados son: estática – estática, dinámica – estática o viceversa, o dinámica – dinámica. El algoritmo decide según esta regla, si la transición es dinámica – dinámica, entonces se declara que no hay CAE, en todos los demás casos se valida como CAE.

En resumen, el algoritmo tiene una primera fase que efectúa una medida basada en diferencias de píxeles, *DCvalor*. En caso de que el resultado sea ambiguo, pasa a la segunda fase.
La segunda fase se puede ver como un detector por métrica de diferencias de histogramas (Comparación 5), pero que se protege de falsas alarmas provocadas por cambios en la luminancia, usando la *medida de escena estática*. Si la escena resulta aún ambigua respecto de esta métrica usa como último criterio la *clasificación de transición de escena*.

B. Diseño e Implementación

En la descripción anterior del algoritmo, quedan algunos aspectos importantes sin especificar. Para cumplir los requerimientos de implementación, fue necesario incorporar soluciones de diseño que no estaban tratadas en [1].

A causa de la importancia de su utilización tanto en el modo de *aprendizaje* como de *reconocimiento*, los CAE detectados en uno,

como en otro modo, deberían tener absoluta coherencia. En este punto trascendente para la resolución global del problema, se encontraron algunas dificultades. El algoritmo como estaba pensado, era vulnerable en ciertos casos críticos para los intereses específicos de esta aplicación. La dificultad principal se centra en

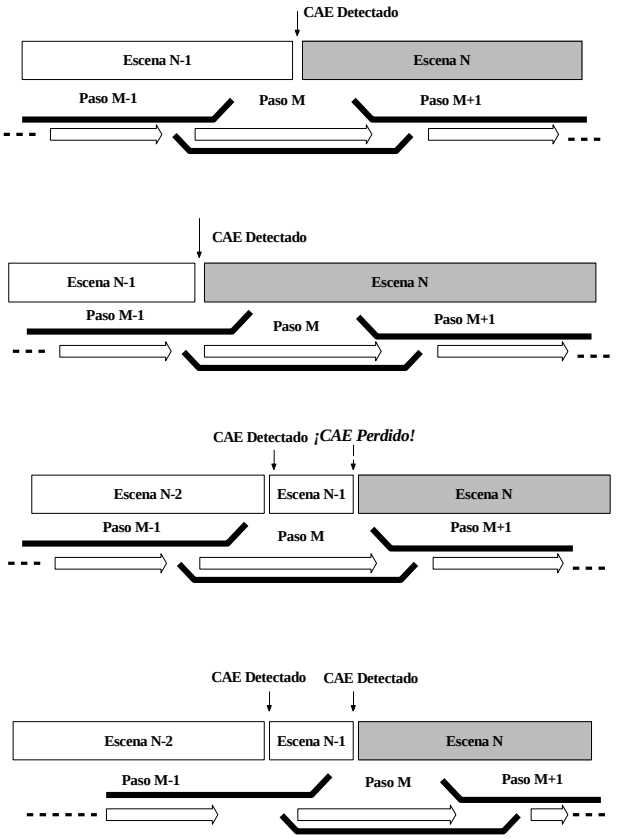


Fig. 9 a) Dentro de una ventana se detecta un solo CAE. b) Un CAE queda dentro de dos ventanas deslizantes diferentes, debido al solapamiento. El CAE es detectado dos veces, esto no es grave, simplemente se descartan la repeticiones. c) Se pierde la detección de un CAE por haber dos dentro de una misma ventana. d) Para las mismas escenas de c), si las ventanas vienen en posiciones relativas diferentes, es posible que se detecten los dos CAE. Esto ilustra el efecto aleatorio del avance de la ventana deslizante.

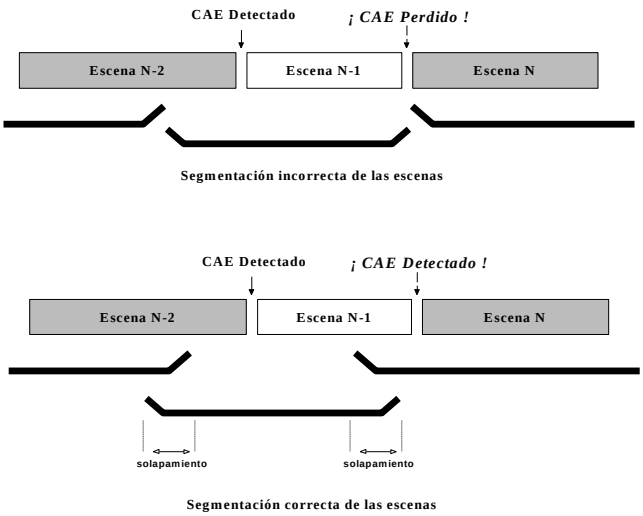


Fig. 10 En el primer caso se muestra el avance de la ventana sin solapamiento. En segundo se agrega el solapamiento.

la forma que la ventana recorre el video en estudio, ver figura 9. El algoritmo, tal cual en [1], no especifica una regla para hacer avanzar la ventana.

La primera consideración que se tuvo está relacionada con el solapamiento entre ventanas adyacentes. Se observó que si entre una ventana y la siguiente no existen cuadros en común, y hay un CAE entre ellas, este no es detectado, ver fig 10. Para solucionar este problema se considera que ventanas adyacentes deben tener cuadros en común. A el número de cuadros que pertenecen a ambas ventanas se le denomina *solapamiento*.

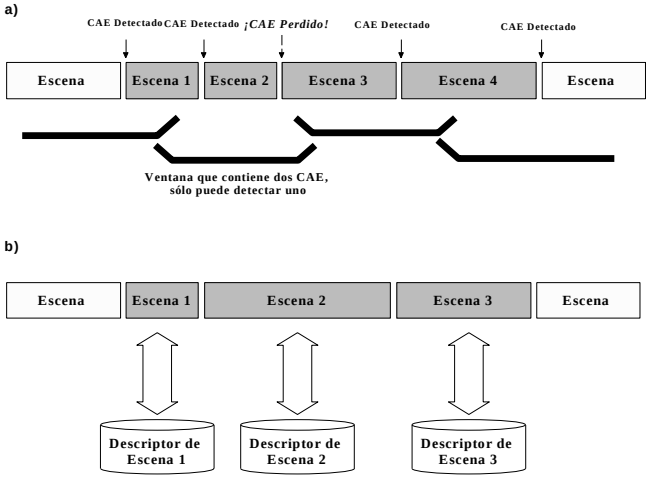


Fig. 11 En a) la ventana recorriendo el video en modo aprendizaje, las escenas del comercial de interés están en color gris.. En b) los descriptores extraídos de la segmentación.

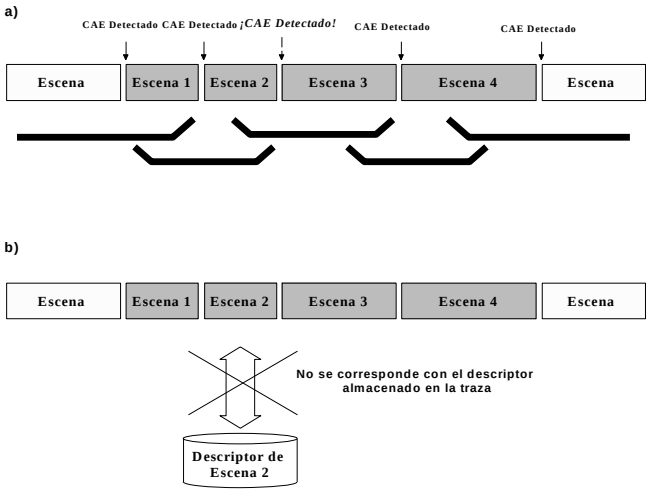


Fig. 12 En a) la ventana recorriendo el video en modo reconocimiento, las escenas del comercial de interés están en color gris. En b) se esquematiza la inconsistencia de la escena 2 con el descriptor almacenado.

La segunda consideración esta vinculada al hecho de que este algoritmo se utiliza tanto en el modo *aprendizaje* como de *reconocimiento.* Esto impone que exista absoluta coherencia entre los CAE detectados en ambos modos.

Se citan ejemplos para resaltar la importancia de este punto:

- En el modo Aprendizaje la ventana deslizante puede comenzar a recorrer el comercial de interés desde una posición relativa al primer CAE distinta a la que lo empiece a recorrer en el modo Reconocimiento. Lo anterior es natural que suceda por el orígen distintos de los videos de Aprendizaje y Reconocimiento. Esta situación ocasiona que se puedan detectar distintos CAE en Aprendizaje que en Reconocimiento. En general lo que sucede es que se pierda algún CAE en alguno de los modos, ver figuras 11 y 12. Esta inestabilidad provoca que el comercial no sea reconocido por la dependencia del sistema de reconocimiento con las escenas.

- En los casos en que existen escenas muy cortas, de algunos pocos cuadros, o que los canales de TV insertan cuadros negros entre comerciales, también se producen inestabilidades, ver figura 13. Nuevamente la posición relativa de la ventana a los CAE, en un modo u otro puede ser fatal. Si la ventana contiene dos CAE, podrá detectar uno solo de ellos o ninguno. Los cuadros negros, provocan además que se pueda perder el comienzo exacto del comercial o su final, arruinando el normal funcionamiento.

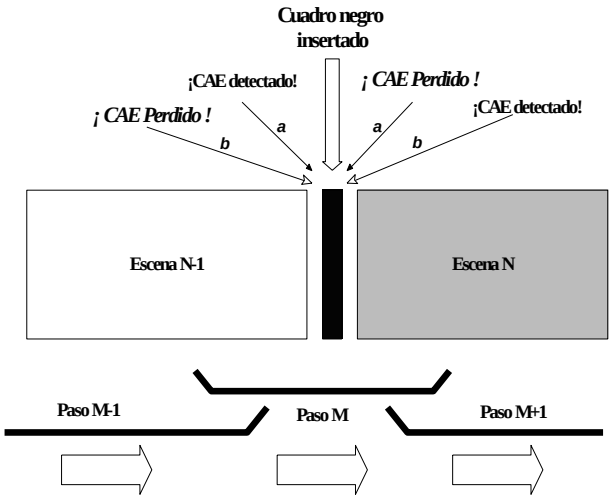


Fig. 13 La inserción de un cuadro negro entre comerciales también genera inestabilidades. El resultado puede ser a o b dependiendo de la escena N-1 y la escena N.

El problema crítico no es que no se detecte un CAE , sino que no

sean los mismos CAE detectados en *aprendizaje* que en *reconocimiento*.

Para solucionar los problemas anteriores, se focalizó la atención en la forma de asegurarse de que el comercial a reconocer, sea recorrido de la misma forma por la ventana deslizante, tanto en modo *aprendizaje* como en *reconocimiento*. Para lograr esto, se pensó en sincronizar la ventana deslizante con los CAE. Cuando en la ventana no se detecten CAE, avanza con el criterio inicial tomando en cuenta el *solapamiento*. Cuando se detecta un CAE, se ajusta la ventana a ese CAE y se empieza desde allí sin tomar en cuenta el solapamiento, que no aporta nada en este caso.
Esto asegura que la ventana recorra el comercial aprendido, de igual forma que cuando lo pretende reconocer, solucionando el problema antes mencionado, ver figura 14.

El problema de las escenas muy cortas, queda resuelto en el sentido que el sistema va a detectar tanto en el *aprendizaje* como en el *reconocimiento* las mismas escenas.

Para los cuadros negros insertados, la ventana sincronizada contribuye de manera importante a la solución, pero es necesario tomar ciertas precauciones en el algoritmo de reconocimiento para

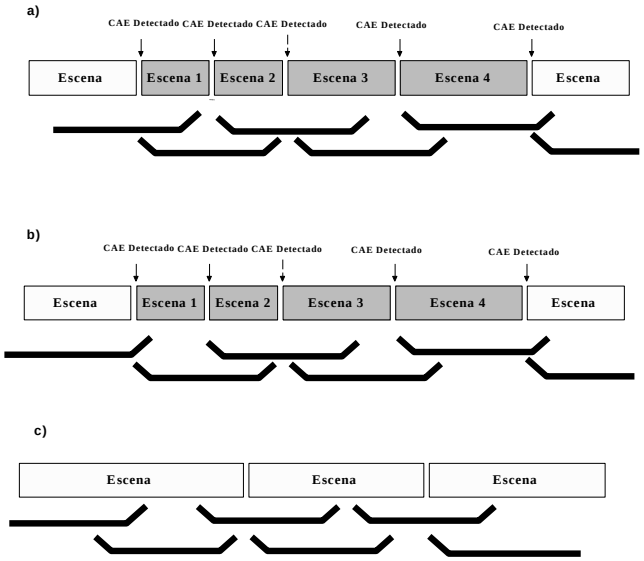


Fig. 14 En a) la ventana recorriendo un video. Observar la posición relativa de la primer ventana al comercial (escenas grises) de interés. En b) se muestra la ventana recorriendo otro video, que contiene el mismo comercial. Observar que pese a que la posición relativa de la primer ventana respecto al comercial es distinta que en a), todas las demás ventanas que lo recorren tienen la misma posición relativa que en a), esto permite visualizar el efecto sincronizar la ventana a los CAE. c) Se muestra que mientras la venta no contiene CAE, se respeta el solapamiento para su avance.

resolver definitivamente el problema, ver figura 15.

Para la implementación del algoritmo se contemplaron todos los detalles del diseño. Se priorizaron los aspectos de eficiencia orientados a la velocidad y administración de recursos, teniendo en cuenta que este algoritmo era parte fundamental en los modos de funcionamiento *aprendizaje* y *reconocimiento*.

VIII. La Traza y los Descriptores de Escenas

Para trabajar en un nivel semántico más alto, se refiere a la *traza* como un concepto que engloba la información total que se extrae del comercial y que el sistema almacena en un archivo como un conjunto de bytes. La *traza* es generada durante el aprendizaje y contiene un descriptor de escena por cada escena del comercial. Además es la fuente de información esencial para el reconocimiento.
Los *descriptores de escena*, por ser parte medular de la *traza* toman un rol relevante en la escencia del sistema. Se le exigen tres características fundamentales que se trasladan también a la *traza*:

a. Ser lo suficientemente fiel y representativo de la escena como para no perder apariciones del comercial y no generar falsos reconocimientos.

b. Permitir al sistema realizar una comparación ágil y veloz para descartar coincidencias. Estadísticamente es la situación mas frecuente que se va a presentar e impactaría fuertemente en la performance temporal del sistema. Fundamental para trabajar con varias *trazas* a la vez.

c. Tolerancia a distorsiones que pueden existir entre el video del cual se generó la *traza* y sobre el que se busca. Esto apunta a las variaciones de colores causadas por diferencias de ajuste de color en los emisores o receptores de TV (según sean PAL o NTSC).

Para satisfacer estas tres propiedades se utilizaron tres tipos de descriptores, descriptores de GdC basados en histogramas ponderados por alfa, histogramas intersección[i] e intersección con corrimientos, y cantidad de cuadros por escena.

Se busca obtener alguna información que sea representativa de un *shot* o Grupo de Cuadros GdC, existen varios métodos para esto. Se empleará el término GdC pues los descriptores en si trabajan sobre un GdC y no necesariamente sobre un *shot*. Si bien ya se explico que a efectos del funcionamiento óptimo del sistema es deseable que los GdC de los que se construyan descriptores, sean en efecto *shots*. Esa es la tarea del algoritmo detector de CAE, particionar la secuencia de video en *shots*, para que posteriormente el sistema genere la *traza*.

A. Histogramas del cuadro representativo

El histograma del cuadro representativo [2] es el histograma de colores que se considera el más representativo de un GdC. La descripción de color dada por este histograma varía significativamente con el criterio tomado para elegir el cuadro representativo. Una forma de evitar dicho problema es elegir el cuadro representativo de la siguiente forma. Para un cuadro cualquiera l en un GdC, con histograma H_l, se define el error promedio E_{H_i} como

$$E_{H_i} = \frac{1}{M}\sum_i \|H_i - H_l\|, i = 1..., M$$

(8)

Donde M es el número de cuadros en el GdC. El histograma H_r que minimiza (8) es el histograma óptimo representativo del GdC para la medida analizada.
Este método requiere de muchas operaciones de cómputo y una búsqueda exhaustiva entre los cuadros para encontrar aquel que minimiza el error. Para un GdC de duración M, un total de Mx(M-1) comparaciones de histogramas son necesarias para llevar a cabo el cálculo. Dicho valor puede volverse muy grande con mucha rapidez si aumenta M. En este procedimiento siempre se encuentra un único cuadro para representar todo el GdC, por lo que exige para su efectividad que el GdC tenga contenido uniforme.
Por su peso en materia de cálculo y su dependencia de que el GdC sea un shot, se decidió no incluir este descriptor en la implementación del sistema. [2]

B. Método de histogramas promedio ponderados por alfa

Un descriptor más conveniente que el anterior es uno que represente la información *acumulada* de color de todos los cuadros dentro de un GdC [2]. El método más directo para lograr esto es acumular todos los valores de color de los píxeles de todos los cuadros en un mismo histograma. Tras aplicar una normalización adecuada se obtiene el histograma promedio. Cada bin j en el histograma promedio para el k-esimo GdC es definido como.

$$Histprom_k(j) = \frac{1}{M}\sum_{i=b_k}^{e_k} H_i(j), j = 1,..., B$$

(9)

Donde H_i es el histograma del i-esimo cuadro, b_k y e_k son los cuadros inicial y final del GdC, $M = (e_k - b_k + 1)$ el número de cuadros del GdC y *B* es el número total de bins de los histogramas. Un problema potencial con usar promedios para calcular el valor de cada bin es que si se presenta una distribución atípica en algún o algunos cuadros del GdC, con un color no utilizado en el resto de los cuadros, o con un color distribuido en más píxeles, el bin correspondiente a este color quedara con un valor que no es representativo de todo el GdC, sino que está introducido por un cuadro que presenta desviación respecto al promedio de los cuadros. Un ejemplo podría ser una variación rápida de la luminancia, el pasaje rápido de un objeto o un acercamiento y alejamiento veloz de cámara.
Un modo de obtener un descriptor de color robusto frente a estas desviaciones es reemplazar el promedio, con la media de los histogramas. Esto es una forma eficiente de eliminar los efectos de

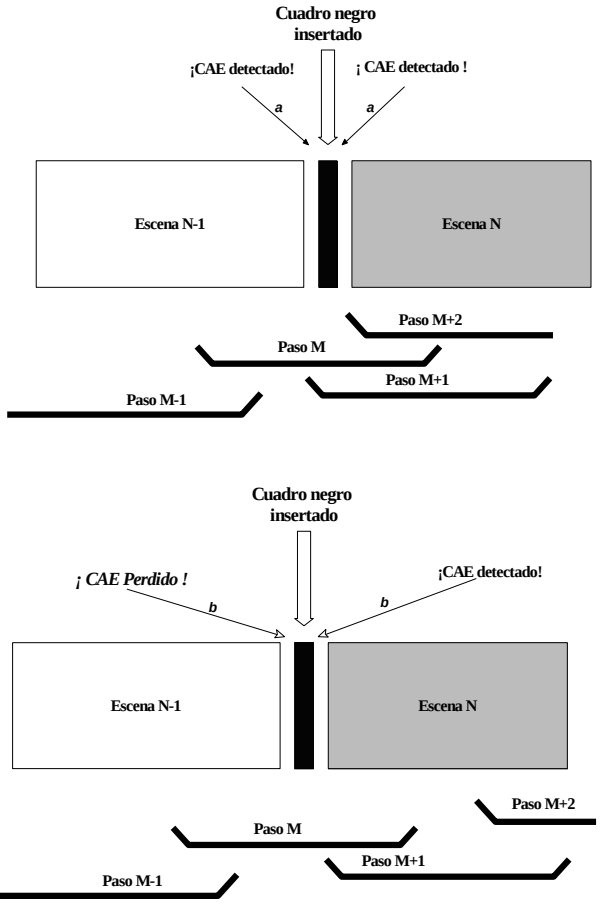


Fig. 15 En el primer caso, el sincronismo de ventana permite que se detecten los dos CAE adyacentes al cuadro negro, esta situación se indica con la letra en las correspondientes flechas. En el paso M se detecta el primer CAE y en el paso M+1 se detecta el segundo gracias al sincronismo.
En el segundo caso, sucede que en el paso M se detecta el segundo CAE, en lugar del primero, por lo tanto, aunque la ventana se sincronice, el primer CAE ya no se puede detectar.

los cuadros atípicos dentro del GdC. El histograma medio es dado por

$$Histmed(j) = mediana\{H_{b_k}(j), H_{b_k+1}(j), ..H_{e_k}(j)\} \quad (10)$$

Para calcular el j-esimo bin en *Histmed* se calculan una lista ascendente de los valores presentes en el j-esimo bin de los M cuadros del GdC. Si M es impar, se toma la mediana de esta lista como el resultado, si M es par se calcula el promedio entre los dos valores centrales de la lista.

Un enfoque alternativo para calcular el histograma del GdC es definir una familia de *histogramas de promedio ponderado*, que es generada a partir de un operador de promedio ponderado. Un operador de promedio ponderado se obtiene ordenando el arreglo de valores de cada bin en orden ascendente, y promediando solo los elementos centrales del arreglo ordenado. Para un número par de cuadros cada j-esimo bin en el histograma promedio es calculado de esta forma

$$\alpha \Pr ompond_k(j,\alpha) = \frac{1}{M - 2\cdot\lfloor aM \rfloor} \sum_{m=\lfloor \alpha M \rfloor + 1}^{M-\lfloor \alpha M \rfloor} \hat{h}_j(m) \quad (11)$$

Donde $\lfloor \alpha M \rfloor$ es el mayor entero no mayor a αM ,y $\hat{h}$ es el arreglo ordenado de los j-esimos valores de bin. El parámetro de ponderación $\alpha, 0 \le \alpha \le 0.5$, controla el número de datos que son excluidos del cálculo del promedio.

El operador de promedio ponderado reduce la contribución de datos atípicos, descartando iguales cantidades de datos a cada extremo de los arreglos ordenados. Nótese que cuando $\alpha = 0$, el resultado es igual al histograma promedio, y cuando $\alpha = 0.5$ es como calcular el histograma medio[1]. Generalmente si no hay un fuerte efecto de cambios de luminancia y/o crominancia el histograma promedio es representativo de las características generales de color del GdC, insumiendo pocas operaciones de cálculo. Si se presentan variaciones mas importantes de las características de color, es necesario tomar valores de $\alpha \ne 0.$

Los promedios ponderados de histogramas proveen una estimación apropiada de las características de color de todos los cuadros de un GdC, un modo de determinar la precisión de dichos descriptores es realizar una medida del error promedio $E_{H_{GdC}}$. Para un GdC de duración *M* cuadros, $E_{H_{GdC}}$ se define como el promedio por cuadro de las distancias acumuladas entre los histogramas de cada cuadro y el histograma de promedios ponderados.

$$E_{H_{GdC}} = \frac{1}{M} \sum_i \left\| H_i - H_{GdC}(j) \right\| \quad (12)$$

Se pueden usar varias normas para calcular la distancia entre el histograma del cuadro dado y el histograma de promedios ponderados.

El error absoluto promedio es

$$\left\| H_i - H_{(\cdot)}(j) \right\| = \sum_j \left| H_i(j) - H_{(\cdot)}(j) \right| \quad (13)$$

El error cuadrático promedio es

$$\left\| H_i - H_{(\cdot)}(j) \right\| = \sum_j \left| H_i(j) - H_{(\cdot)}(j) \right|^2 \quad (14)$$

De cálculos estadísticos surge que la métrica que minimiza el error para el caso del histograma medio es el error absoluto promedio, y si se usa el histograma promedio, entonces es más conveniente el uso del error cuadrático promedio. En el sistema corresponde la primera, ya que el $\alpha \ne 0$. El descriptor de promedio ponderado por alfa, es implementado por el sistema. Si bien requiere de mucho procesamiento, a raíz entre otros elementos de los ordenamientos ascendentes, su atributo principal es la fidelidad y representatividad de la escena, como para no perder apariciones del comercial y no generar falsos reconocimientos. [2]

C. Histograma intersección

El *histograma intersección* [2] es calculado de la siguiente forma

$$Hist\,\mathrm{int}_k(j) = \min_i \{H_i(j)\} \quad (15)$$

El valor de cada bin en *Histint* representa el número de píxeles de un cierto color que aparece en todos los cuadros de un GdC.

Este tipo de histogramas representa el mínimo común que tienen entre si los cuadros de un GdC, en vez de ser un histograma representativo de las características de color del GdC. Esta propiedad del histograma intersección lo hace apropiado para el rápido descarte de un GdC en una búsqueda dada de este dentro de una secuencia de video.

Dado cualquier GdC, *k* y su *histograma intersección*, *Histint* se deduce de su definición que,

$$H_i^k(j) \ge Histin_k(j), \forall i \in GdC_k,\ j = 1,..., B \quad (16)$$

donde H_i^k es el histograma del i-esimo cuadro en el k-esimo GdC. Esta propiedad puede ser directamente usada para determinar si un cuadro *puede* pertenecer a un GdC específico. Dado un cuadro *f*, es posible descartar si pertenece a un GdC rápidamente, ya que las diferencias $H_f(j) - Histin_k(j)$ deben ser siempre positivas si $f \in$ k en el video. Surge entonces un veloz método para el descarte de GdC, dada una búsqueda de uno en particular. Lo que sigue, sugiere otros usos de este descriptor y lo desarrollamos a modo ilustrativo ya que no fue utilizado en la implementación.

$$D = H_f(j) - Histin_k(j),\ j = 1,..., B \quad (17)$$

Si *D* < 0 para algún bin, entonces el GdC es rechazado como candidato a satisfacer la búsqueda. De lo contrario se define un coeficiente de correspondencia,

$$C_{f,k} = 1 - \frac{1}{N_f} \sum_j D \quad (18)$$

[1] La formula esta pensada para valores de M impares, si M par, la media se calcula como el promedio de los 2 valores centrales.

Donde N_f es el número total de píxeles en el cuadro *f* . Si *f* pertenece al k-esimo GdC, entonces la $\sum_j D$ debe ser pequeña. De aquí que un valor mayor de $C_{f,k}$ implica una mayor probabilidad de que $f \in GdC_k$. Los $C_{f,k}$ pueden ser ordenados para confeccionar una lista de GdC a los que *f* puede pertenecer.

Este *histograma intersección* es utilizado en el sistema como descriptor de los *shot* que integran el comercial aprendido. Su atributo principal es la rapidez con que se puede determinar si una cuadro dado no pertenece al shot. Este descriptor cumple fundamentalmente con el objetivo de economizar operaciones y ganar velocidad. [2]

D. Cantidad de cuadros en la escena

Este valor también se incorpora como un descriptor secundario o descriptivo de cada escena. Su función es permitir en la etapa de reconocimiento, gran velocidad en el descarte de las escenas. Su atributo está en su sencillez y la facilidad para su implementación. Naturalmente se toman márgenes de seguridad en cuanto a la cantidad de cuadros, para que no se pierdan reconocimientos a causa de este descriptor. Es una especie de filtro bruto para el proceso de reconocimiento.

IX. TÉCNICA DE COMPENSACIÓN DE CORRIMIENTOS

Satisface el objetivo c. de los descriptores, le llamamos *histograma intersección con corrimientos*. [5]

Los *histogramas intersección*, son una técnica poderosa para descartar GdC que no coinciden con los *shots* del comercial que se busca reconocer. En efecto, el no cumplimiento para algún cuadro del GdC, de la condición de la ecuación (16) implica que podemos descartar ese GdC, y no dedicar más operaciones a su procesamiento.
Sin embargo, los *histogramas intersección* son vulnerables a diferencias de color entre el video donde se aprendió el comercial y donde se está buscando. Estas diferencias se refieren a diferentes ajustes de color que pudieran tener las emisoras de broadcast televisivo. Por ejemplo, si se aprendiera un comercial cuyo video tiene como origen una emisora, y luego se quisiera reconocer monitoreando la señal de TV de una emisora distinta, los colores podrían tener variaciones que engañaran al *histograma intersección*. El problema anterior existe tanto en el sistema PAL, donde la emisora ajusta el color y el receptor no, como en NTSC donde el receptor también puede ajustar el color.
Las variaciones en los histogramas de color de cada cuadro, producidas por los diferentes ajustes de color, pueden ser modeladas como corrimientos en sus bins [5]. Se desarrollo un método para que el *histograma intersección* que se calcula en el sistema tenga cierta tolerancia a estos errores.
Supongamos que se tiene un *GdC* "a" que constituye el comercial del cual se halla su *histograma intersección* $H_A(j)$. Supongamos que en una secuencia de video tenemos presente el mismo *GdC*, pero por las razones ya explicadas hay una diferencia en sus colores. El *histograma intersección* del *GdC* en esta última secuencia, se corresponde según el modelo de corrimientos, con una de las siguientes funciones,

$$\overline{H}_A(j) = \begin{cases} H_A(j-c) & j < c \\ 0 & j \le c \end{cases} \quad (19)$$

$$\overline{H}_A(j) = \begin{cases} H_A(j+c) & j < B-c \\ 0 & j \le B-c \end{cases} \quad (20)$$

Esto corresponde a traslaciones horizontales del histograma, la primera hacia la izquierda y la segunda a la derecha. Si en estas condiciones queremos verificar (16), es plausible que en ciertos casos esta no se verifique, y que por tanto el GdC sea considerado como no perteneciente al comercial.
Para no descartar el GdC en situaciones como estas, es que se ideó un método que relaja la imposición dada por el h*istograma intersección*, aplicándole lo que se denomina una *compensación de corrimientos* al *histograma intersección*. La idea es que en vez de guardar el *histograma intersección* $H_A(j)$ como descriptor de la escena, se guarda un vector $L(j)$, vector de *compensación de corrimientos* del histograma que cumple

$$L(j) \le H_A(j \pm a) \qquad \forall a / |a| \le corr \quad (21)$$

Donde *corr* es el máximo corrimiento esperado por errores de color en los histogramas. Tenemos un descriptor que cumple la condición

$$H_A(j \pm a) \ge L(j), \forall j \in GdC_k,\ j = 1,...,B \quad (22)$$

y sustituyendo por (19) o (20)

$$\overline{H}_A(j) \ge L(j), \forall j \in GdC_k,\ j = 1,...,B \quad (23)$$

La condición anterior significa que si usamos *L* como descriptor, los histogramas que presenten un corrimiento menor a *corr* serian validados, teniéndose una tolerancia a distorsiones de color. Este descriptor, *histograma de intersección con corrimientos* no es

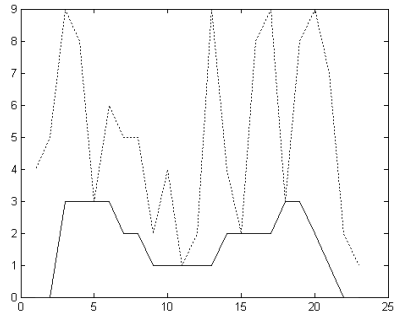


Fig. 16 La operación de compensación de corrimientos. La línea punteada es el *histograma intersección* original, en vez de guardar este como descriptor, se guarda el representado por la línea continua que soporta la comparación con el original, aun con corrimientos laterales. Adicionalmente se puede agregar un corrimiento vertical hacia abajo.

implementado por el sistema en la versión final del proyecto, pero se propone como solución al problema explicado. Se hicieron pruebas utilizándolo en lugar del *histograma intersección*, con resultados positivos, ver figura 16.

X. ALGORITMO DE RECONOCIMIENTO

A. Diseño

El desarrollo de este algoritmo está íntimamente relacionado con las características impuestas a la traza y sus descriptores. A partir de estas se busco el procedimiento mas eficaz de reconocimiento, tomando como objetivo los siguientes puntos:

a. Descartar lo antes posible y con el menor uso de recursos las escenas no contenidas en el comercial a reconocer.

b. Frente a una escena potencialmente contenida en el comercial buscado, tener un recurso exigente para poder *descartarla* con la certeza de que no es la buscada.

c. Si no se puede descartar, someterla a una comparación que permita asegurar que es la buscada.

Este orden y filosofía de búsqueda, está inspirado en el hecho estadístico de que la mayoría de las veces, el sistema se enfrenta a escenas que no son las buscadas, por eso se comienza por intentar *descartarla*. Tener criterios de descarte veloces aumentan la eficiencia del algoritmo, mejorando fundamentalmente su performance en velocidad y requerimientos de recursos de hardware. La velocidad es un atributo esencial para el reconocimiento, ya que se entiende que la aplicación debería poder procesar videos en tiempo real. Por último, para evitar falsos reconocimientos es necesario tener una prueba mas exhaustiva, que es deseable ejecutarla la menor cantidad de veces.

Se tomaron tres descriptores de escena, la *cantidad de cuadros por escena*, el *histograma intersección* y el *histograma promedio ponderado por alfa*.

El diagrama de flujo del algoritmo se muestra en Fig. 17.

El algoritmo procesa el video segmentándolo en escenas. Durante la segmentación realiza la búsqueda. Cada vez que se produce un CAE, se empieza a procesar la escena que finaliza con ese CAE. El procesamiento consiste en:

1) Se chequea primero si esa escena tiene la misma cantidad de cuadros que la correspondiente buscada. Si no se supera esta instancia se sale del algoritmo, este paso busca satisfacer el objetivo a.

2) Si coincide el número de cuadros, se empieza a comparar el histograma de cada canal de cada cuadro de la escena, con un *único histograma intersección* por canal. Estos tres *histogramas intersección* son descriptores del shot o escena original y están registrados en la *traza*. El primer histograma que se encuentre que no supera la prueba, es decir que tiene un bin de valor inferior a su correspondiente en el *histograma intersección*, se sale del algoritmo. Esta prueba cumple con el objetivo b.

3) Si se supera la prueba anterior, aún no es suficiente para asegurar que se ha reconocido la escena, porque lo anterior permite descartar pero no confirmar. Podría suceder que se supere la prueba anterior pero los histogramas sean diferentes a los originales. Entonces, para cumplir con el objetivo c, se chequea que cada histograma de cada canal, tenga "poca diferencia" con el *histograma medio ponderado por alfa* correspondiente contenido en la *traza*. Se aclara que hay tres *histogramas medios ponderados por alfa* por cada shot, almacenados en la *traza*, uno por cada canal. En este paso del algoritmo se decide definitivamente si es la misma escena.

El algoritmo determina que un comercial ha sido reconocido, cuando se reconocieron todas sus escenas.

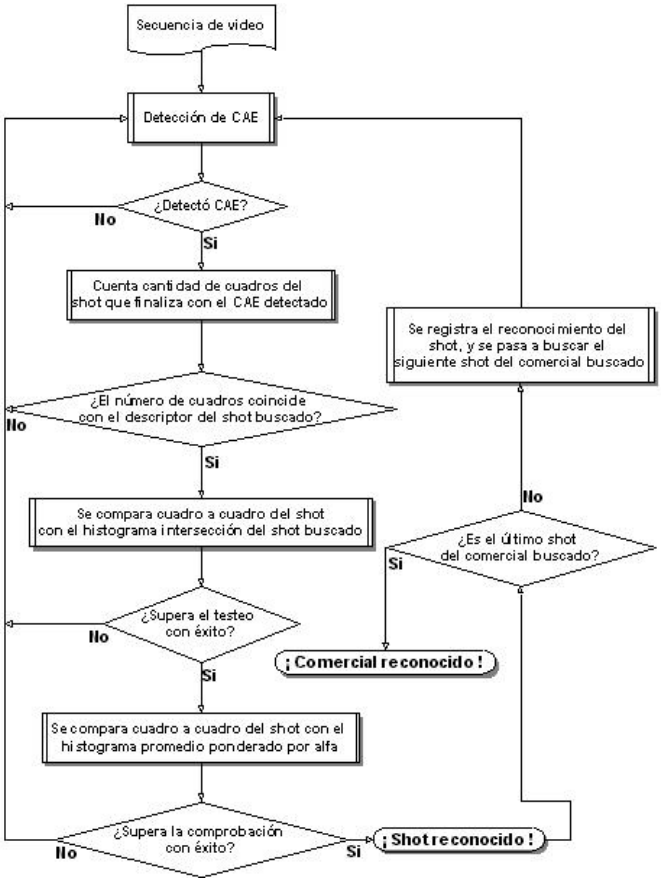


Fig.17 Diagrama de flujo del algoritmo de reconocimiento.

B. Implementación

A los efectos de la implementación, se respetan las características de diseño. Se toman tolerancias para las comparaciones con los descriptores. Se permite mas menos 1 cuadro con la *cantidad de cuadros por escena* y hasta 2 ocurrencias más por bin con el *histograma intersección*. La *traza*, es un archivo asociado a cada comercial aprendido que es levantado antes comenzar a correr el algoritmo, para cargar los descriptores asociados al mismo. Se resalta la flexibilidad en la implementación del algoritmmo, lo que permite modificaciones. En particular durante las pruebas se utilizó también el descriptor *histograma intersección con corrimientos* sustituyendo al *histograma intersección*, sin modificar en absoluto la estructura del algoritmo.

Algunos canales de TV insertan cuadros negros entre comerciales. Esto genera fallas al esquema de reconocimiento, que fueron corregidas desde el algoritmo de detección de CAE, a través del sincronismo de ventana, y en el algoritmo de reconocimiento, no chequeando el último cuadro del último shot de cada comercial, ver figura 15 y distinguir que si no se toma en cuenta el último cuadro, que es eventualmente el negro se corrige el problema.

XI. COMENTARIOS Y REQUERIMIENTOS DEL SISTEMA

La administración de la memoria *RAM* disponible, requirió especial atención. En un primer momento del desarrollo este elemento no fue tenido en cuenta. Luego, empezaron a aparecer problemas porque se ocupaba toda la memoria *RAM* disponible y el sistema no podía seguir con sus funciones. Se hizo primeramente una tarea exhaustiva para controlar la administración de la memoria, con la ayuda del entorno de desarrollo. Sin embargo, el sistema seguía requiriendo grandes cantidades de *RAM* para cumplir sus funciones, en particular del orden de 80 MB. Se encontró el causante de este problema, y estaba relacionado con una estructura que se utiliza para extraer y descomprimir los cuadros desde un stream de AVI. Esta estructura no se estaba cerrando adecuadamente. El sistema requiere finalmente, 4 MB de memoria *RAM* promedialmente para su correcto funcionamiento, sin tener oscilaciones importantes de la misma.

En cuanto a la velocidad de procesador se requiere un procesador de 600 Mhz o superior. La memoria necesaria para la instalación del programa es 13 MB en disco duro. No se requieren prestaciones especiales para la tarjeta de video.

XII. DISCUSIÓN

Si bien el sistema funciona correctamente, se pueden hacer mejoras en la algoritmia de reconocimiento, orientadas a una mejor performance en tiempo de ejecución. Los test actuales son muy exigentes ya que reconocen todas las escenas. Una posible mejora sería no examinar todas las escenas sino algunas, esto incrementaría la velocidad de desempeño poniendo en compromiso la fidelidad del reporte. Se recomienda la sustitución del histograma intersección por el histograma intersección con corrimientos por las razones expuestas. Las medidas de tiempo de ejecución realizadas en el modo de reconocimiento son muy cercanas al tiempo de reproducción del video. Teniendo en cuenta la posibilidad de hacerle optimizaciones tanto al código como a los ejecutables del programa se piensa que se estaría trabajando con tiempos inferiores a los de reproducción. Esto facilitaría una futura implementación capturando la señal el video desde una tarjeta adquisidora y no desde disco. Las técnicas analizadas y expuestas en ese capítulo abren un abanico de posibilidades para futuras implementaciones.

REFERENCIAS

[1] Chung-Ling Huang and Bing-Yao Liao, "A Robust Scene-Change Detection Method for Video Segmentation," *IEEE* Trans. Circuits Syst. Video Technol., vol. 11, pp. 1281-1288, diciembre 2001.

[2] A. Mufit Ferman y A. Murat Tekalp, "Robust Color Histogram Descriptors for vides Retrieval and Identification," *IEEE* Transactions on Image Processing, vol 11, No. 5, Mayo 2002.

[3] Hyung Sung Chang, Sanghoon Sull, and Sang Uk Lee, "Efficient Video Indexing Scheme for Content-Based Retrieval," IEEE Trans. Circuits Syst. Video Technol., vol. 9, pp. 1269-1279, diciembre 1999.

[4] Z. Cernekova, C. Nikou, and I. Pitas, "Shot Detection in Video Sequences Using Entropy-Based Metrics," *IEEE ICIP*, pp. 421-424, 2002.

[5] Juan Maria Sanchez and Xavier Binefa "Audicom : A Video Analysis System for Auditing Commercial Broadcast," *IEEE International Conference on Multimedia Computing and Systems*, Vol. 2, pp. 272-275, junio 7 -11 1999.

CAPÍTULO III

Umbrales de Decisión para el Algoritmo de Detección de Cambios Abruptos de Escena

Nicolás Scaglione, Martín Thomas, Pablo Varela.

Abstract-**Se analizan los valores numéricos típicos que tienen las distintas distancias que los algoritmos miden en las secuencias de video, para determinar el ajuste de los valores umbrales de los algoritmos de detección de cambios abruptos de escena. Se muestran los resultados obtenidos.**

Índice de términos —*DCvalor*, *Hvalor*, *StaticSceneTest*

I. INTRODUCCIÓN

El propósito de este estudio, es determinar los umbrales numéricos que el algoritmo de detección de CAE usa para sus decisiones, de acuerdo a tres objetivos de referencia:

a. El algoritmo debe reconocer la mayor cantidad de CAE, en particular todos los *intercomerciales*.
b. El número de falsas alarmas debe ser el menor posible. Una falsa alarma se produce cuando se detecta un CAE, en donde no lo hay.
c. Economizar operaciones, para ganar velocidad.

II. FUNDAMENTO TEÓRICO

Usamos dos tasas que miden el funcionamiento del algoritmo:

$$\text{Detección} = (\text{Ncae}) / (\text{Ncae} + \text{Nper}); \quad (1)$$

$$\text{Precisión} = (\text{Ncae}) / (\text{Ncae} + \text{Nfa}); \quad (2)$$

- Donde Ncae el numero de CAE detectados, Nper el numero de CAE perdidos y Nfa, el número de falsas alarmas [1]. Para cada uno de los pares de cuadros correspondientes a CAE *intercomerciales* o CAE *intracomerciales*, se calcularon las distancias *nratio*, *h* y *s*. También se calcularon estas distancias para los cuadros restantes donde no hay CAE. Se tiene así un conjunto compuesto por ternas (*nratio*, *h* y *s*), que está particionado en tres subconjuntos. Estos son, los compuestos por las ternas asociadas a CAE entre comerciales, a CAE dentro de un comercial y al resto.

El algoritmo utilizado mide tres distancias, que miden distintas propiedades entre cuadros consecutivos. Se compara cada una con uno o dos umbrales. Por detalles sobre estas medidas y sobre el funcionamiento del algoritmo remitirse al *capitulo II.*
Un diagrama de flujo del algoritmo se muestra en la figura 1.

Medida	Umbrales	Mide
Nratio	*nlow , nhigh*	Cambio relativo de las características generales entre cuadros.
hvalor	*hhigh, hlow*	Cambios en el contenido cromático entre cuadros.
Prueba de escena estática	*shigh*	Cambio en los bordes de la imagen.
Validación Cruzada	*sshigh*	Clasifica el tipo de transición de escena

Tabla 1. Resumen de los tipos de distancias utilizadas, propiedades que miden y umbrales de comparación asociados a cada uno.

Con el fin de determinar los umbrales de decisión se efectuaron diferentes medidas y pruebas. El material de video que se utilizó fueron secuencias de video digitalizadas provenientes de señales de TV abierta. Como plataforma de pruebas se utilizo *Matlab*, con la utilería de tratamiento de imágenes que este incorpora para hallar y graficar los valores de cada distancia.

III. MÉTODO

Se detallan a continuación los pasos seguidos:

- Se midieron y graficaron los valores de *nratio*, *h* y *s* (Ver Figs. 2, 3 y 4). A partir de estas gráficas se estimaron valores para los umbrales. Se registraron los valores máximos, mínimos y medios de las distancias en presencia de CAE. También se registraron los valores medios de estas cuando no había CAE. Tomando en cuenta los valores registrados, y basándose en los objetivos trazados, se fijaron los umbrales iniciales.

- Se segmentó la secuencia en *shots* utilizando el algoritmo con los umbrales anteriores.

- Se hizo un control visual de los cuadros en que el algoritmo declaró presencia de CAE. Se eliminaron falsas alarmas y se agregaron CAE perdidos. Con esto obtuvimos un registro de los cuadros donde existe CAE en la secuencia.

- Los CAE fueron diferenciados en dos tipos. Aquellos que limitan dos comerciales consecutivos, denominados CAE *intercomerciales*, y los que limitan escenas dentro de un mismo comercial, denominados CAE *intracomerciales*.

Nicolás Scaglione, Martín Thomas, Pablo Varela, estudiantes de grado, Facultad de Ingeniería de la Universidad de la Republica, Montevideo, Uruguay.

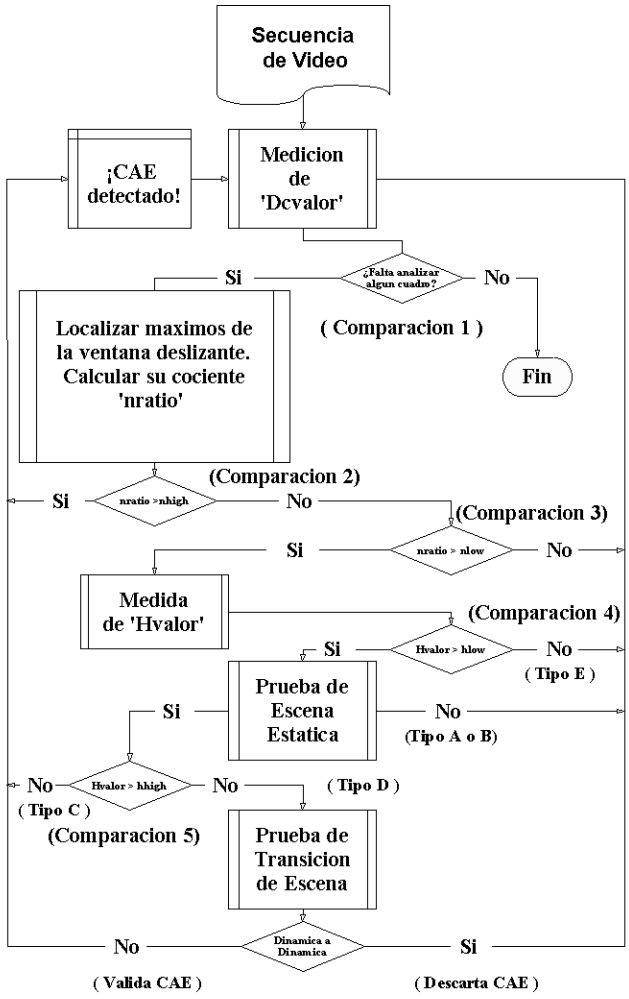


Fig.1 Diagrama de flujo del algoritmo de detección de CAE.

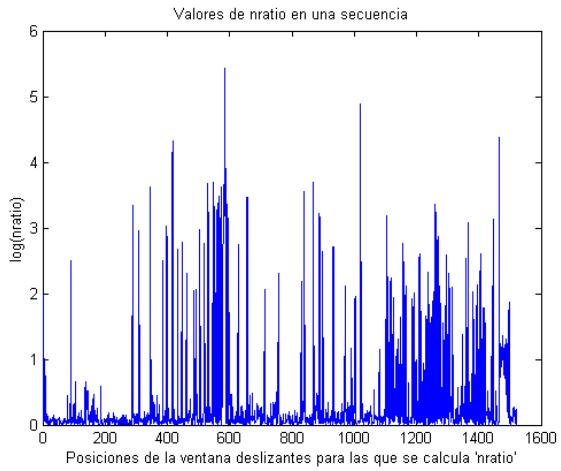


Fig.2 Gráfica con los valores de *nratio* obtenidos del análisis de una secuencia de video, se representa el logaritmo de *nratio* por razones de claridad. Los valores muy cercanos a cero, corresponden a valores muy cercanos a 1. Los valores altos aislados pertenecen a CAE, pero también escenas con movimiento aumentar su valor, ver cuadros 1100 al 1500.

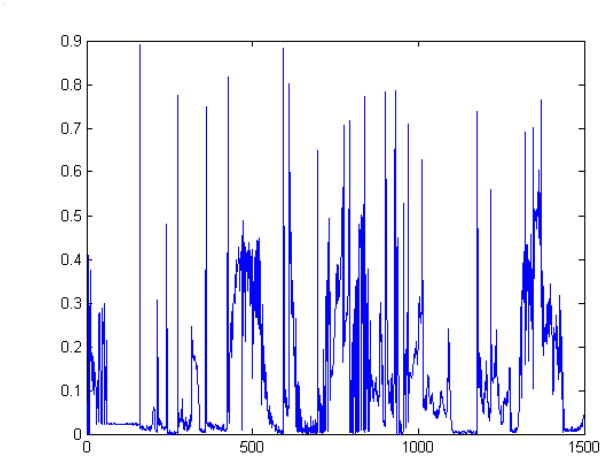


Fig.3. Gráfica con valores *s* de una *prueba de escena estática*. Se observan picos aislados correspondientes a CAE. También se observan series de medidas consecutivas que arrojan valores mas elevados, indicando escenas con mayor movimiento.

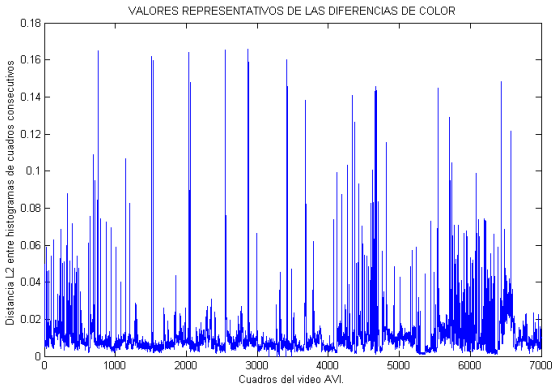


Fig. 4 Gráfica con valores de *hvalor* para un video con comerciales. Se aprecian las escenas con mayor movimiento, en el entorno del cuadro 6000. Entre 1000 y 4000 en cambio, se destacan los CAE con sus valores mayores a la media.

- Se preparó un programa que calcula las tasas de detección y precisión del algoritmo, para diferentes valores de los umbrales. Luego, se hallaron aquellos que maximizan la detección y la precisión. Como la detección y la precisión no son variables independientes, se maximizó una suma ponderada de ellas.

IV. DETERMINACIÓN DE LOS *nhigh*, *shigh* Y *hhigh*

El algoritmo recorre el video usando una ventana deslizante. Para cada posición de esta ventana, se dan tres situaciones distintas y excluyentes: transiciones sin CAE, CAE *intracomercial* o CAE *intercomercial*. Para cada posición de la ventana, se extrajo la terna de valores de distancias *h*, *s* y *nratio*, entre los dos cuadros de la ventana en donde existe un potencial CAE. Cada terna fue clasificada según a cual de los tres conjuntos de situaciones pertenece.

Estos valores, pueden ser graficados en tres dimensiones para visualizar su distribución en el espacio, como se muestra en la

figura 5. Cada terna de medidas entre cuadros, es asociada así a un punto en el espacio (*h,s,nratio*).

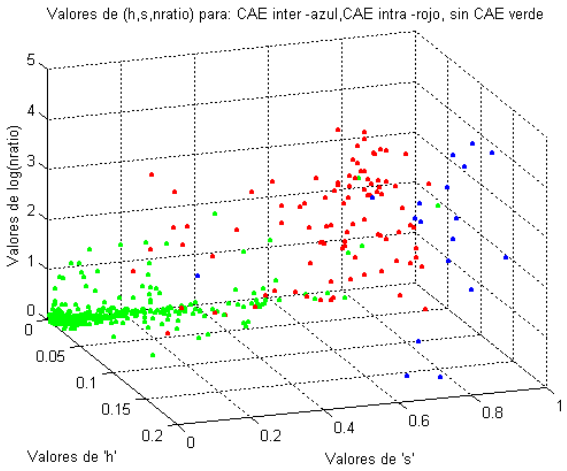


Fig.5 Gráfica mostrando la distribución de las distancias (*h*, *s*, *nratio*) medidas por el algoritmo para distintas situaciones. En azul los CAE inter, en rojo los CAE intra, y en verde el resto de las transiciones. Se grafica el logaritmo de *nratio* solo por razones de claridad.

En las comparaciones 2, 5 y de *prueba de escena estática* del algoritmo, que se ven en la figura 1, son usados estos umbrales para detectar un CAE.
Se asume como supuesto que la mayoría de los CAE deben ser detectados en estas comparaciones. Este supuesto se fundamenta en la idea de que la *prueba de transición de escena* es aplicada por diseño, a casos ambiguos y poco frecuentes en el tipo de secuencias a analizar. Son casos en que ninguna de las cinco comparaciones previas que realiza el algoritmo logran clasificar la transición.

Dado un par de cuadros entre los que hay un CAE, para ser detectado en las comparaciones 2 y 5, deben cumplirse alguna de las dos condiciones siguientes:

$$nratio > nhigh \quad (3)$$

o

$$s > shigh \quad \text{y} \quad h > hhigh \quad (4)$$

La región que contiene los puntos que cumplen las condiciones (3) y (4) es denominada *volumen de detección* y se puede visualizar en la figura 6. Los puntos contenidos en ella son los CAE que declara el algoritmo en las comparaciones 2 y 5.
Este volumen tiene nueve caras, que son: A-E-J-D, A-B-C-D, A-E-F-B, B-C-L-K-G-F, D-C-L-M-I-J, M-J-K-L, I-J-E-F-G-H, H-G-K-J, I-H-J-M. De estas, seis están determinadas por valores máximos y mínimos de los datos. Pero tres de ellas varían según los umbrales de detección, I-H-J-M, H-G-K-J y I-J-E-F-G-H, ver figura 7.

El problema de determinar los umbrales *nhigh*, *hhigh* y *shigh* se puede visualizar como el de ubicar estas caras de forma que el volumen contenga la mayor cantidad de puntos correspondientes a CAE, y la menor cantidad de no CAE.
El objetivo a) es mayor detección, y el b) mayor precisión posible, sin embargo, detección y precisión no necesariamente aumentan a la vez. Si ampliáramos el volumen de detección hasta abarcar todos los puntos correspondientes a CAE, se logra una detección de 100% a costa de una baja precisión (solución trivial). Como es de interés maximizar la detección y precisión a la vez, se busco maximizar la suma ponderada, *M*, de ambas tasas. Esta se define de la siguiente forma:

$$M = D * peso_d + P * peso_p$$

Donde detección (D) y precisión (P) son las tasas definidas en (1) y (2), ponderadas por dos coeficientes $peso_d$ y $peso_p$.

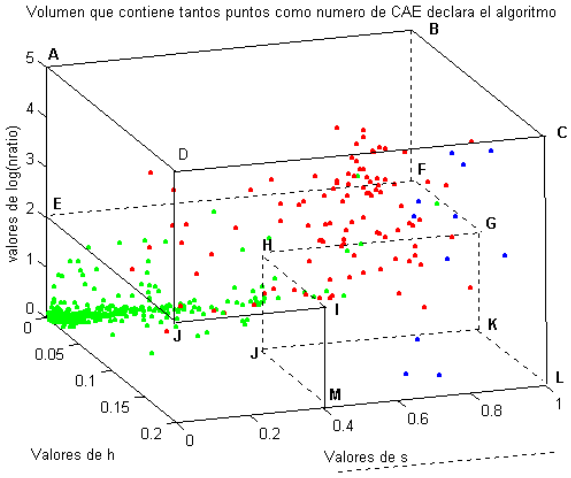


Fig. 6 Se muestra el volumen de detección, con los puntos *nratio*, *h* y *s*. Los que caen en su interior son declarados como CAE por el algoritmo.

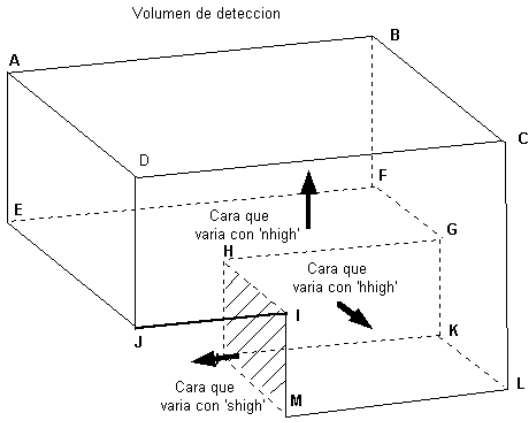


Fig.7 El volumen de detección mostrando las 3 caras que varían con los umbrales *nhigh*, *hhigh* y *shigh*.

Se procedió a variar los parámetros *nhigh*, *hhigh* y *shigh* dentro de ciertos rangos que surgieron de los de valores máximos y mínimos de estas distancias previamente calculadas. Para cada combinación de umbrales, se contabilizó cuantos puntos pertenecen al volumen de detección y cuantos no. Con esta información se calcularon las tasas de detección y precisión respectivas para mas de 10^5 combinaciones diferentes de umbrales. Obtenidos estos datos se hallaron las ternas que maximizan M para diferentes valores de los pesos $peso_d$ y $peso_p$.
Se calcularon las tasas de detección, precisión y precisión corregida, esta última surge de eliminar falsas declaraciones de CAE, originadas por cambios graduales de escena. Esto se hizo para comparar las tasas obtenidas con las reportadas por los creadores del algoritmo. Estos utilizan el algoritmo de detección

de CAE junto con un algoritmo de detección de cambios graduales. Por lo tanto, pueden diferenciar si en la secuencia hay un cambio gradual, y eventualmente descartar declararlo como un CAE.

Se obtuvieron los resultados que se muestran en la tabla número 2.

pesod	pesop	Detec	Prec.	Prec.C.	nratio	hhigh	Shigh
0,50	1,5	90,18%	96,19%	96,19%	8,7	0,017	0,55
0,60	1,4	90,18%	96,19%	96,19%	8,7	0,017	0,55
0,70	1,3	90,18%	96,19%	96,19%	8,7	0,017	0,55
0,80	1,2	90,18%	96,19%	96,19%	8,7	0,017	0,55
0,90	1,1	90,18%	96,19%	96,19%	8,7	0,017	0,55
1,00	1	91,07%	95,33%	96,30%	7,3	0,017	0,55
1,10	0,9	91,07%	95,33%	96,30%	7,3	0,017	0,55
1,11	0,8	96,43%	88,52%	95,65%	7,3	0,019	0,31
1,12	0,7	96,43%	88,52%	95,65%	7,3	0,019	0,31
1,13	0,6	96,43%	88,52%	95,65%	7,3	0,019	0,31
1,14	0,5	99,11%	82,22%	88,28%	8,7	0,019	0,19

Tabla 2. Resultados de detección, precisión y precisión corregida, en esta última no se incluyen las falsas detecciones que en realidad provienen de cambios graduales de escena. Datos obtenidos para distintos valores de los pesos pesod y pesop.

pesod	pesop	Detec	Prec.	Prec.Corregida.	nratio	hhigh	shigh
0,5-0,9	1,5	90,18%	96,19%	96,19%	8,7	0,017	0,55
1-1,10	1-0,9	91,07%	95,33%	96,30%	7,3	0,017	0,55
1,11-1,13	0,8-0,6	96,43%	88,52%	95,65%	7,3	0,019	0,31
1,14	0,5	99,11%	82,22%	88,28%	8,7	0,019	0,19

Tabla 3. Resumen de las 4 combinaciones obtenidas, representativas de soluciones de compromiso entre detección y precisión.

Tomando los coeficientes de la tercera fila del cuadro de la tabla 3, se logran tasas tentativas de 96.43% de detección y 95.65% de precisión. Son tentativas porque el volumen de detección no incluye los puntos que son analizados por la *prueba de transición de escena*. Es así que se decidió tomar estos valores:

$$\underline{nhigh = 7.3}$$

$$\underline{hhigh = 0.019}$$

$$\underline{shigh = 0.31}$$

V. Determinación de *nlow* y *hlow*

Estos 2 umbrales cumplen la función de que para valores muy bajos de *nratio* y *h*, el algoritmo descarte la posibilidad de que halla un CAE, ver comparaciones 3 y 4 de la figura 1.

Llamamos paso del algoritmo a todas las operaciones de medida, comparación y declaración que hace el algoritmo para una posición de la ventana deslizante dada.

Para *nlow*, se encuentran valores correspondientes a CAE muy cercanos a 1 (1.0125). Para no arriesgar perder detección, se fijo un valor menor a 1.0125 y se tomó un valor intermedio entre 1.0125 y 1 (el mínimo valor posible de *nratio*), $\underline{nlow = 1{,}006.}$ Con la adición de 0.006 a 1, se logra en la práctica un descarte por tener *nratio* < *nlow* en el 6% de los pasos del algoritmo, es decir que de todos los *nratios* calculados, el 6% es menor que *nlow* y el algoritmo decreta no CAE. Se vio que acercar mas este valor a 1.0125 no producía un aumento significativo de este porcentaje de descarte. Por su controlada incidencia en el problema, no se amplio mas el estudio sobre este umbral.

El *hlow* se lo tomó por debajo de los valores mínimos de *h* encontrados en CAE (0.0175), pero lo suficientemente por encima del valor *h* promedio cuando no hay CAE, que es 0.010, se vio que por encima de 0.015, acercar mas este valor a 1.0175 no producía un aumento significativo de este porcentaje de descarte. Este valor se fijo en $\underline{hlow = 0.015.}$
Se tiene un margen por debajo de 0.0175, lo que cubre de perder detección, y se logra un descarte de CAE del 69% de los pasos, por resultar *hvalor* < *hlow*.

Los dos umbrales juntos deciden el descarte de CAE en el 75% de los pasos. En los datos analizados, un 79% de los pasos del algoritmo corresponden a transiciones que no corresponden a CAE. El porcentaje de descartes de CAE que queda sin resolver después de pasar por *nlow* y *hlow*, es algo mayor a (79% - 75%) = 4%. Este porcentaje sin resolver, corresponde a los descartes de CAE que se efectuarán mas adelante por la *medida de escena estática*, la *prueba de transición de escena* y los falsos positivos.

VI. Determinación de *sshigh* y *r1*

Para determinar a partir de que valores de *ss* una escena se puede considerar dinámica, se implemento un reproductor de archivos de video. Este reproduce solo los cuadros entre los que la *medida de escena estática* es mayor a un umbral *sshigh*. El *sshigh* se tomo inicialmente en valores cercanos a 0, y luego se fue aumentando. Así se llego a un *sshigh* tal que el reproductor solo mostraba escenas con un grado apreciable de movimiento. De estas pruebas surgió que una escena dinámica arroja típicamente un *ss* mayor a 0.1 entre cuadros consecutivos. Se ajusto *sshigh* a este valor pero se observo que ciertos CAE eran considerados como falsa alarma a través de la *prueba de transición de escena*. Se busco disminuir la posibilidad de esto, reduciendo la posibilidad de detectar transición dinámica– dinámica, para ello se probo cambiar el radio de cobertura de *r1* = 3 a *r1* = 10 que resolvió las falsas alarmas anteriores. Aumentar el valor del radio es aumentar el grado de movimiento necesario para declarar una escena como dinámica.

Existe otro parámetro, que determina cuantos cuadros atrás y adelante de los cuadros donde hay un posible CAE, se realizan las *medidas de escena estática* usadas por la *prueba de transición de escena*. Este parámetro llamado 'alejamiento' se fijo en 1, es decir se mide un cuadro antes y un cuadro después de los cuadros con CAE potenciales. No hay razones para fijarlo en un valor mayor, y fijarlo en uno es medir la característica de la escena, en el instante de tiempo mas cercano a la transición que se dispone.

VII. Tamaño de la Ventana Deslizante

Es la cantidad de cuadros con que se recorre el video, y además son los cuadros que se consideran para realizar el cómputo del primer y segundo máximo de las diferencias *dcvalor*. Cuanto

mayor sea esta, menos pasos se realizan, y más rápido corre el algoritmo. Pero se corre el riesgo, que siendo mayor que una escena, se pierda alguna. Si se achica demasiado aparecen algunos problemas, se demora mucho en recorrer el video, y pierde sentido la medida *nratio* que está pensada para varios *DCvalor*, generando bajas en las tasas de precisión.
El valor de la ventana se fijó en 10 cuadros, permitiendo un buen desempeño del algoritmo.

Además, está el parámetro *solapamiento* que determina un solapamiento entre las posiciones consecutivas de la ventana, para evitar perder un CAE entre 2 bordes de la ventana. El solapamiento se fijo en 3, ya que con hacerlo mayor se perdía velocidad sin ganar nada, y para hacerlo menor está la *prueba de transición de escena* como limitante, a raíz del *alejamiento*.

VIII. Umbrales Seleccionados

Los umbrales del algoritmo que se determinaron fueron,

Umbral o parámetro	Valor
Ventana deslizante	10 cuadros
solapamiento	3
nhigh	7,3
nlow	1,007
hhigh	0,019
hlow	0,015
shigh	0,31
sshigh	0,1
alejamiento	1
r	3
r1	10

IX. Resultados

En las pruebas realizadas, que se hicieron sobre archivos con comerciales de canales de TV abierta, se encontraron las siguientes tasas de calidad del algoritmo para los umbrales determinados:

Precisión = 91%

Detección = 97 %

Detección de CAE *intercomerciales* = 100%

No se encontró ningún CAE perdido que correspondiera a una transición entre dos comerciales distintos.

X. Discusión

Los resultados obtenidos pueden no ser una solución óptima al problema, pero si suficiente para la aplicación que se da al algoritmo en el desarrollo en cuestión. El algoritmo de detección de CAE utilizado, es un algoritmo que por su concepción pretende relajar el problema de los umbrales [2].

Se llegó a tasas cercanas a los preconizadas por los autores del algoritmo, que son detección = 97.7%, y precisión = 96.8%. En las pruebas realizadas por estos sobre comerciales, los valores son de 96% y 97% respectivamente. En comparación con nuestros resultados, la precisión es la que esta mas alejada. No se dispone de información numérica, sobre los valores y rangos en las mediciones realizadas por los autores del algoritmo. Esto nos impide evaluar en profundidad el porque de esta diferencia.

El algoritmo no es siempre inmune a acercamientos de cámara, ya que si se presenta un cambio significativo en la distribución de los colores, con la variación correspondiente de *Hvalor*, se puede declarar un falso CAE.

Se podría haber realizado una selección mayor del *nlow*, ya que con la selección realizada se entra en la segunda fase del algoritmo alrededor del 94% de las veces. Esto impone una sobrecarga de procesamiento, pero el mayor tiempo de cálculo está en la determinación de los *DCvalor*, paso que por ser el primero, siempre es realizado.

Los autores del algoritmo [2] en sus resultados para CAE, registran dos causas de perdida de CAE, una debida a que el *nratio* no supera el *nlow*, y otra debida a transiciones dinámica – dinámica detectadas en la *prueba de transición de escena*. La primera causa está ausente en nuestra implementación, debido al criterio elegido para *nlow*. Las causas de pérdida obtenidas en las pruebas fueron mayormente debidas a valores de *s* y *h* mayores a los umbrales *shigh* y *hhigh*, y escenas que no eran clasificadas debidamente por la *prueba de transición de escena*. Así mismo se observo que se pueden generar falsos CAE con esta medida. La medición de sólo 2 diferencias antes y después de la transición, no siempre caracteriza bien las escenas. No esta garantizado que dentro de una escena dinámica, entre cualquier par de cuadros la *prueba de escena estática* arroje un *s* mayor que *sshigh*. Esto hace algo inexacta la caracterización del tipo de escenas por este método.
Por último la decisión de declarar que no hay CAE si la transición es dinámica-dinámica, es una decisión estadística, e introduce una imperfección inherente al algoritmo mismo.

Los falsos se dan típicamente en escenas con movimiento y cambios de color, en especial aquellas que tienen efectos especiales de edición.

En todas las pruebas siempre se detectaron los CAE existentes entre comerciales, siendo esto lo mas importante a efectos de la utilidad del algoritmo en este trabajo. En efecto, no reconocer un CAE entre dos comerciales tiene dos efectos no deseados:

- Evita la segmentación automática del comercial en escenas para su correcto *aprendizaje.*

- Produce en la fase de reconocimiento, que escenas del comercial tengan que ser analizadas junto con una escena ajena a este. Esto hace virtualmente imposible de reconocer la escena y por tanto el comercial. Lo anterior está vinculado a la robustez de los *descriptores de escena*, la unión de una escena con otras origina que el algoritmo de reconocimiento la descarte.

La precisión menor al 100%, no hace más que introducir una segmentación redundante de los comerciales, haciendo algo mas compleja la traza, pero no compromete el reconocimiento.

Es importante aclarar que estos resultados son válidos para video capturado a 25 cuadros por segundo. Se hicieron pruebas con video capturado a 15 cuadros por segundo y disminuían notoriamente las tasas de detección y precisión. Esto se explica, porque las variaciones de las métricas entre cuadros adyacentes, son mayores en 15 cuadros por segundo que en 25 cuadros por segundo.

XI. CONCLUSIÓN

Tomando en cuenta los resultados obtenidos, detección 97%, precisión 91%, y detección de CAE *intercomerciales* del 100%, se considera que la elección de estos umbrales, si bien puede no ser una solución óptima, ha sido acertada. Los umbrales elegidos, arrojan tasas que sustentan la utilización del algoritmo para la resolución del problema global de reconocimiento de comerciales.

REFERENCIAS

[1] Tieyan Liu, Xudong Zhang, Linwei Shan and Yingning Peng, "Constant false-alarm ratio processing for video cut detection," *IEEE ICIP* 2002, pp. 121-124.

[2] Chung-Ling Huang and Bing-Yao Liao, "A Robust Scene-Change Detection Method for Video Segmentation" *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, pp. 1281-1288, 2001.

CAPITULO IV

Pruebas del Sistema PCP

Nicolás Scaglione, Martín Thomas, Pablo Varela.

I. Sumario

En este documento se describe cuales fueron las pruebas sistemáticas realizadas al sistema, y que resultados arrojaron. Estas pruebas corroboran los resultados que se obtuvieron en pruebas previas, no documentadas, que se hicieron durante el desarrollo.

II. Descripción de las Pruebas Realizadas

Se probaron comerciales de 4 archivos *AVI* descomprimidos, provenientes de tres canales de televisión abierta de Montevideo. Se utilizaron los cuatro archivos que se describen en la tabla 1.

Características de los archivos analizados

Nombre	Tamaño	Cuadros	Duración
Canal 4-1	192MB	3465	2 min. 18 seg.
Canal 4-2	196MB	3539	2 min. 21 seg.
Canal 10 –1	677MB	12335	8 min. 13 seg.
Canal 12 - 1	563MB	10150	6 min. 46 seg.

Tabla 1. Características de los cuatro archivos evaluados. Dos de ellos son del canal 4 y los otros del canal 10 y 12 de la TV abierta de Montevideo.

Las pruebas consistieron en los siguientes pasos

En el modo *aprendizaje* se le ingresaron al sistema 28 comerciales procedentes de estos cuatro archivos.

Para cada uno de estos comerciales, en el modo *reconocimiento*, se puso el sistema a analizar cada uno de los cuatro archivos utilizados.

Se registro si el comercial era reconocido cuando correspondía, y que no hubiera falsos reportes de reconocimiento. En total se registraron los resultados para 116 operaciones de reconocimiento del sistema. Esto incluye el control de falsos reconocimientos.

III. Resultados

En la tabla 2 se ven los resultados obtenidos para los comerciales probados. Todos los comerciales fueron reconocidos y no se produjo ningún falso reconocimiento.

Resultados con comerciales

Comercial	Origen	Canal 4-1	Canal 4-2	Canal 10-1	Canal 12-1
Radio Montecarlo	4-1	OK			
Centro de Ojos	4-1	OK			
Lancome	4-2		OK		
Loreal	4-2		OK		
Resistiré	4-2		OK		
Secom	4-2		OK		
Resistiré	4-2		OK		
Canal	4-2		OK		
Coca Cola	4-2		OK		
Conaprole	4-2		OK		
El Chana	4-2		OK		
Costumbres Argentinas	4-2		OK		
Mesa	10-1			OK	
Tres en una mesa	10-1			OK	
Garbarino	10-1			OK	
TotalNewsAgency	10-1			OK	
Caldos Knork	12 -1				OK
Higienol	12-1				OK
Círculos de Creditel	12-1				OK
Coca Cola	12-1				OK
Loreal Age Perfect	12-1				OK
Margot	12-1				OK
Mueller	12-1				OK
Filarmónica Mont.	12-1				OK
P. Carretas Shopping	12-1				OK
Rosario Viñoly	12-1				OK
Siempre Libre	12-1				OK
Yerba Arminio	12-1				OK

Tabla 2. Resultados de las pruebas. En las columnas de izquierda a derecha se indica: Comercial evaluado, archivo al que corresponde dicho comercial, resultados para el archivo correspondiente. Dos de ellos son del canal 4 (4-1, 4-2) y los otros del canal 10 (10-1) y 12 (12 – 1) de la TV abierta de Montevideo.

IV. Discusion

Los resultados demuestran una absoluta fiabilidad del sistema, como es de esperar del diseño robusto de los descriptores de escena y los algoritmos. Sin embargo sería prudente para sacar

Nicolás Scagilone, Pablo Varela, Martín Thomas, estudiantes de Grado de la facultad de Ingeniería.

conclusiones definitivas hacer más pruebas. Se estaría en resultados del órden de los deseados para su utilización. Que el PCP no halla reportado falsos reconocimientos puede deberse a dos factores:

- No se analizaron suficientes videos como para que surja un caso

- Los descriptores de escena tienen gran información, y la algoritmia de reconocimiento es muy exigente.

VI. CONCLUSION

Los resultados obtenidos muestran que el sistema caracteriza con un grado de precisión los comerciales, que hacen viable su utilización. El hecho de que en las pruebas realizadas no se presentaran fallas, da idea de la factibilidad de usar el sistema para fines comerciales. Para esto debe tenerse en cuenta que se guarda un cuadro de cada comercial reconocido. Esto permitiría que un usuario humano corrija posibles falsos reconocimientos.

CAPÍTULO V

Reporte de gestión de proyecto

Nicolás Scaglione, Martín Thomas, Pablo Varela.

I. Sumario

En este documento se informan los recursos, tiempos planificados y tiempos de ejecución, e indicadores de gestión. Se realiza una comparación con lo planificado, y se indican causas de las desviaciones. Por último se hace un análisis retrospectivo de riesgos.

II. Resultados del Proyecto

Si bien se cumplió en materia de requerimientos técnicos, los plazos de entrega estipulados inicialmente tuvieron que ser reajustados en el transcurso del proyecto. Se realizaron las entregas en fechas posteriores a la planificación inicial.

III. Seguimiento de Proyecto

Al finalizar el curso de gestión de proyecto, se acordaron los objetivos, alcance y entregables del proyecto, ver apéndice D.

El objetivo general del proyecto según lo planificado consistía en desarrollar e implementar un software que sea capaz de:

- Permitir al usuario seleccionar dentro de una secuencia de video ciertos *shots* consecutivos a ser reconocidos.
- Reconocer los *shots* seleccionados dentro de una secuencia de video cualquiera.
- Generar un reporte indicando las coincidencias.
- El desarrollo podrá implementarse en un PC estándar. Se entiende por estándar el tipo de PC que se vende en plaza.

Para su ejecución se dividió en tres etapas principales, llamadas entregables.
Se describe a continuación los objetivos de cada uno.

Primer entregable

Es aquella parte del software que presenta una interfaz gráfica con el usuario y permite la manipulación de videos, incluyendo algunas funcionalidades específicas.

Segundo entregable

Se agrega a lo realizado en el primer entregable, las funcionalidades para aprender y reconocer comerciales. Para este entregable se trata de tener resuelta toda la problemática general que surja. No se realizan pruebas exhaustivas para verificar los porcentajes de error. Se deja para el último entregable la implementación de estas funcionalidades a través de interfaces gráficas adecuadas y optimizaciones necesarias en la algoritmia.

Tercer entregable

En este entregable se completará la realización del proyecto.
En esta etapa se optimizan los algoritmos de detección y reconocimiento implementados. Se realizan pruebas de performance y se analizan fallas específicas.
Sobre la base de esto daremos un índice que reflejara cuantitativamente cuan eficaz es nuestro sistema.

Como bloque final de nuestro sistema se elabora un módulo de software que genere un reporte sobre los comerciales reconocidos en una determinada secuencia de video.

El entregable consistirá en la presentación de un informe detallando la pruebas realizadas con sus resultados correspondientes y el programa completo con algoritmos optimizados.

Seguimiento

Se explican los diferentes indicadores utilizados.

Indicadores de gestión del trabajo realizado:

- WP: Horas presupuestadas para realizar el trabajo efectivamente realizado.
- AWP: Cantidad de horas efectivamente invertidas.
- WS: cantidad de horas presupuestadas para realizar el trabajo que se debería haber hecho a la fecha.
- *Variación de costo*: Mide la diferencia entre las horas presupuestadas y las horas realmente trabajadas. Indica calidad en la estimación de las horas necesarias para el trabajo.

Variación de costo: WP – AWP

- *Variación de cronograma*: Mide la diferencia entre el trabajo realizado y el trabajo que se debería haber hecho a la fecha. Indica calidad en la dedicación de la cantidad de horas de trabajo planificadas.

Variación de cronograma: WP - WS

Nicolás Scaglione, Pablo Varela, Martín Thomas, estudiantes de Grado de la Facultad de Ingeniería.

Informe sobre el primer entregable

Fecha planificada de entrega: 29 /10 /2002

Fecha de entrega: 18 / 12 / 2002

Tareas realizadas:

Estudio Formatos de Video

Estudio de papers

Estudio C++

Diseño UML del Sistema

Implementación de segmentación de AVI

Interfaz gráfica

A la fecha de la entrega los indicadores de gestión eran:

WP = 714

AWP = 625

WS = 714 + 313 = 1027

Variación de costo = WP – AWP = 714 – 625 = 89

Variación de cronograma: WP – WS = 714 - 1027 = -313

Estado del buffer de proyecto:

Días consumidos del buffer del proyecto: 50 días.

Debido al buffer consumido en el primer entregable se reformuló el cronograma. Se acordó como fecha de entrega del segundo entregable el 15 de abril. Para definir la fecha no se hicieron variaciones en el alcance del proyecto.

Para el control de riesgos se agregaron los siguientes:

- Incertidumbre respecto a los requerimientos de capacidad, velocidad y hardware que requiera la aplicación. Esto se relaciona con la capacidad necesaria para ejecutar los algoritmos de reconocimiento durante el despliegue del archivo de video.
- Dudas sobre el diseño actual que podrían llevar a tener que rediseñar parte del sistema. Esto re refiere a la implementación que se le dio a la interfaz gráfica, podría dar problemas para la realización de partes futuras.

Informe sobre el segundo entregable

Fecha planificada de entrega: 15 / 4 /2003

Fecha de entrega: 5 / 6 / 2003

Tareas realizadas:

Estudio de herramientas de tratamiento de imágenes.

Estudio de papers

Desarrollo e implementación de algoritmos en Matlab y Pruebas

Implementación de algoritmos en C++

Software

A la fecha de la entrega los indicadores de gestión eran:

WP = 728

AWP = 1000

WS = 728 + 216 = 944

Variación de costo = WP – AWP = 728 – 1000 = -272

Variación de cronograma: WP – WS = 728 – 944 = -216

Estado del buffer de proyecto:

Días consumidos del buffer del proyecto: 101 días.

Debido al buffer consumido en el segundo entregable se reformuló el cronograma. Se acordó como fecha de entrega del tercer entregable el 15 de julio. Para definir la fecha no se hicieron variaciones en el alcance del proyecto.

Para el control de riesgos se agregaron los siguientes:

- El riesgo respecto a los requerimientos de capacidad, velocidad y hardware requeridos, fue totalmente atenuado. En este entregable se lograron tiempos de ejecución razonables, y aunque se tenia un uso relativamente alto de memoria, los resultados mostraban la viabilidad del sistema.

Informe sobre el tercer y ultimo entregable

Fecha planificada de entrega: 15 / 7 /2003

Fecha de entrega: 18 / 8 / 2003

Tareas realizadas:

Documentación, depuración y detalles del software

Pruebas sobre versión final del desarrollo

Documentación General

A la fecha de la entrega los indicadores de gestión eran:

WP = 216

AWP = 360

WS = 216

Variación de costo = WP – AWP = 216 - 360

Variación de cronograma: WP – WS = 216-216 = 0

Estado del buffer de proyecto:

Días consumidos del buffer del proyecto: 134 días.

Informe del proyecto

A la fecha de la entrega los indicadores de gestión para la totalidad del proyecto eran:

WP = 1658

AWP = 1985

WS = 1658

Variación de costo = WP – AWP = 1658 – 1985 = -327

Variación de cronograma: WP – WS = 1658 - 1658 = 0

Días consumidos del buffer del proyecto: 134 días.

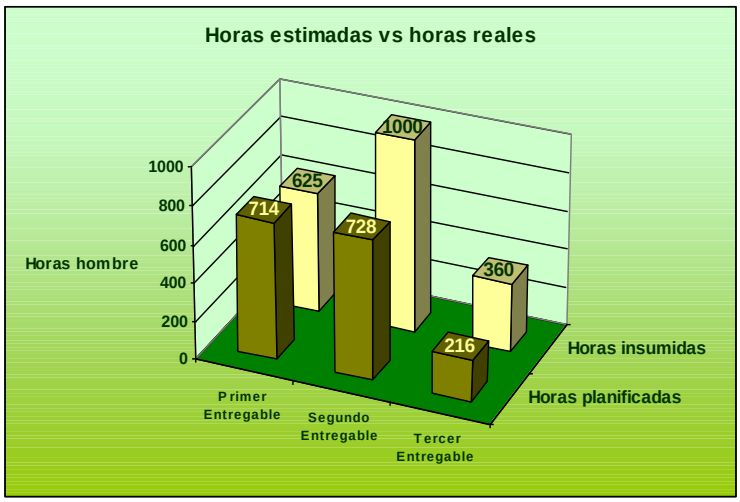


Gráfico indicando las horas planificadas e insumidas en los entregables.

Evaluación retrospectiva de riesgos:

Ninguno de los riesgos considerados tuvo impacto en el cumplimiento de los objetivos del proyecto. De esto se puede concluir que la consideración de riesgos fue adecuada.

IV. Discusión

El proyecto de PCP, implicó para sus tres participantes la participación en un desarrollo de software para una aplicación de procesamiento de imágenes y video. No existía experiencia previa para ninguno de los participantes en un proyecto de similar magnitud. Consideramos que es una materia cualitativamente diferente debido varias características:

- Concretar un sistema de magnitud que efectivamente funcione.
- Libertad y responsabilidad en las decisiones de desarrollo e implementación.
- Interactuar con problemas prácticos.
- Formalidad en las justificaciones, presentación y redacción de la documentación.
- Trabajo en grupo y organización.

Los puntos anteriores incrementaron las dificultades y satisfacciones comparadas con las otras materias de la carrera.

En particular mencionamos los problemas o dificultades de mayor relevancia.
En primer término, ninguno de los integrantes del grupo tenía experiencia en desarrollos de software aparte de los cursos curriculares. Este punto presentó una dificultad importante en materia de tiempo, debido a dos aspectos fundamentales, la elección de C++ como lenguaje de programación, y el requerimiento específico de manejo de archivos de video.

En segundo término, resaltamos la escasez de información sobre el manejo de archivos avi y de interfaces gráficas en C++.

En tercer lugar, no disponer de hardware adquisidor perjudicó el normal avance del proyecto. A causa de esto, se realizaron menos pruebas de las deseadas en cantidad y calidad. Se consiguieron los videos a través de contactos personales.

La variación de cronograma cero indica que todas las tareas por las que se presupuestaron horas fueron cumplidas.

La variación de costo de –327 horas es atribuible a la inexperiencia y mala planificación, en particular se detallan las tareas que llevaron mas horas de las previstas:

- Programación en C++
- Realización de la interfaz gráfica
- Errores y depuración
- Pruebas y determinación de umbrales
- Documentación

La razón que encontramos para explicar la utilización total del buffer se basa en dos factores:

- Pérdida de continuidad en las tareas, producto de interrupciones asociadas a actividades curriculares. Estas interrupciones no fueron previstas de modo adecuado.
- La variación de costo de –327 horas, tuvo su influencia en la utilización del buffer. Las razones ya fueron explicadas.

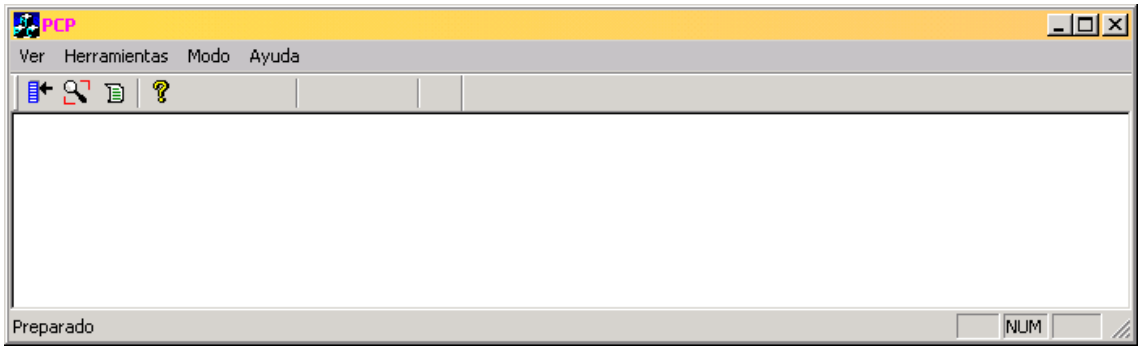
APÉNDICE A - Manual de usuario

Introducción

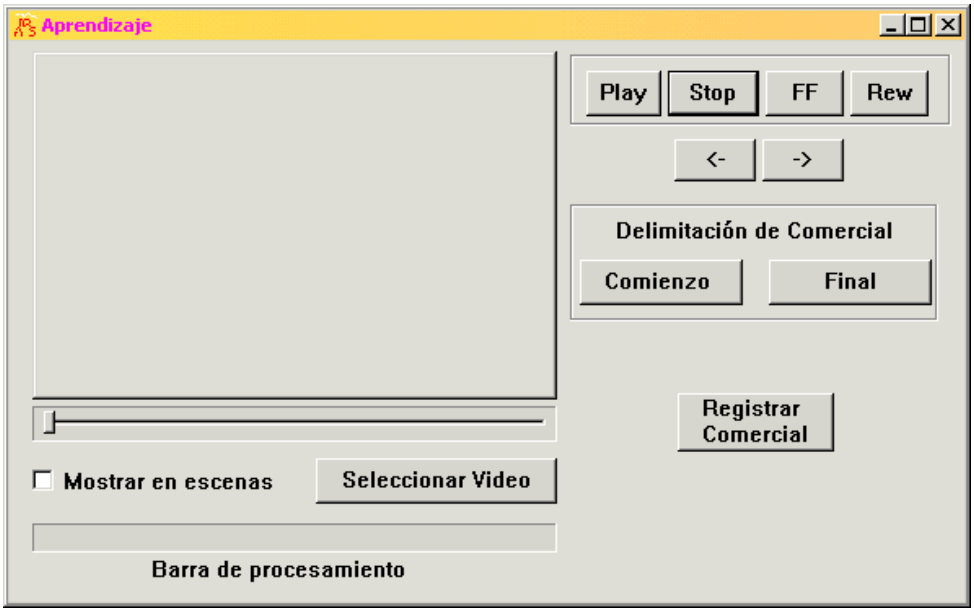
Este es el manual de usuario de el PCP. Aquí se pretende dar una conjunto de instrucciones de como usar el PCP con el propósito de detectar comerciales en un archivo de video.

Instrucciones de uso

1) Una vez que iniciamos la aplicación se despliega la siguiente ventana:



2) Antes de comenzar a reconocer comerciales el sistema necesita aprender el comercial para luego poder ser reconocido. Para que el sistema pueda comenzar a aprender debemos seleccionar el modo aprendizaje, para esto tenemos que seleccionar la opción **Modo** y luego hacer click sobre **Aprendizaje** para así poner en funcionamiento este modo. Otra opción para poder en funcionamiento este modo es hacer click sobre . Luego de esta operación tenemos la siguiete visualización:



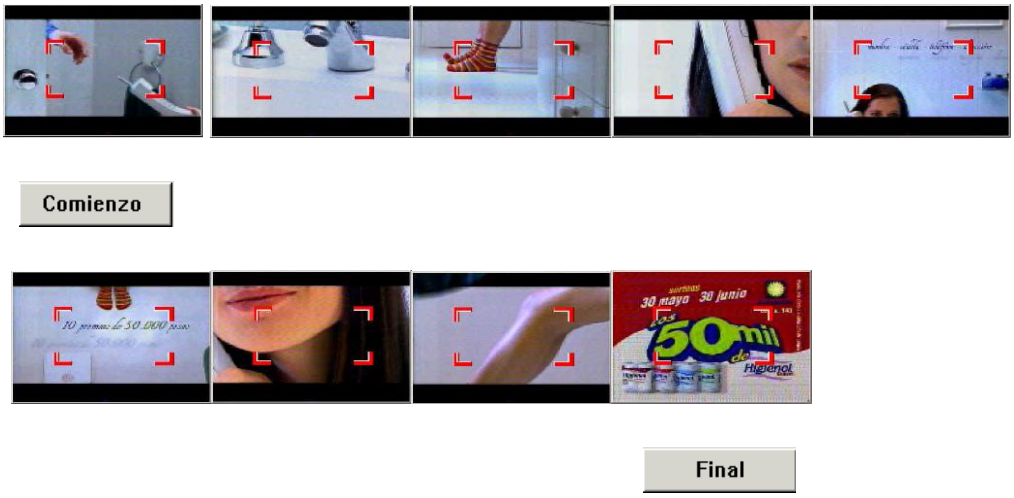
- Antes de seleccionar el video donde se encuentra el comercial a aprender debemos marcar **Mostrar en escenas**. Esto nos va a permitir ver el video seleccionado escena por escena, esto es necesario para poder hacer una delimitación correcta de el comercial.

- Hacemos click en **Seleccionar Video** para selecionar el archivo donde se encuentra el video que contiene el comercial a aprender.

  Una vez realizadas estas dos acciones el sistema comienza a procesar el archivo para mostrarlo en escenas. Si no se marca **Mostrar en escenas** el video seleccionado se va a mostrar cuadro por cuadro a la cadencia con que se halla grabado este video.

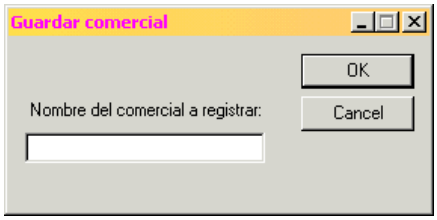
- Una vez que la barra de procesamiento llego a su fin, se puede recorrer todo el video escena por escena haciendo click sobre los siguientes botones:

- Cuando identificamos el comercial marcamos **Comienzo** y **Final** para mostrarle al sistema donde comienza y donde termina el comercial que queremos aprender. Si nos llega a surgir la duda, ¿cuales son las escenas que marcamos?. El sistema marca cuales son las escenas que el usuario marca, basta con retroceder usando el botón de retroceso (<-) y el sistema nos muestra todas las escenas comprendidas entre la escena en la que el usuario marcó **Comienzo** y la escena en la que marcó **Final.**



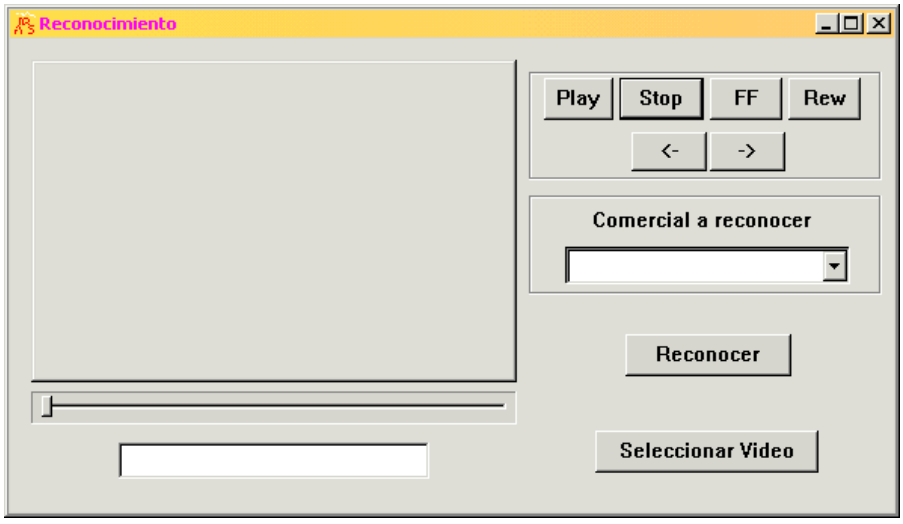
  La primer y último imagen representan la escena donde el usuario marcó **Comienzo** y **Final** respectivamente, luego si recorremos las escenas intermedias estas se van a ver marcadas como se muestra en el conjunto de imagenes anteriores. Si el conjunto de escenas esta incompleto se puede volver a repetir el procedimiento de marcado.

- Una vez que el usuario esta seguro de las escenas que marcó debe hacer click en el botón Registrar Comercial, con esta acción se despliega un cuadro de dialogo donde el usuario debe ingresar el nombre del comercial.



- Ingresado el nombre se da el **Ok** y el sistema comienza a aprender el comercial. Cuando la barra de progreso ha llegado a su fin el sistema ya aprendió el comercial.

- En este punto el sistema ya aprendió el comercial y esta en condiciones de comernzar a reconocer este comercial en el video que el usuario seleccione. Para esto el usuario debe seleccionar el modo **Reconocimiento**. Una vez seleccionado el modo **Reconocimiento** se despliega la siguiente ventana:

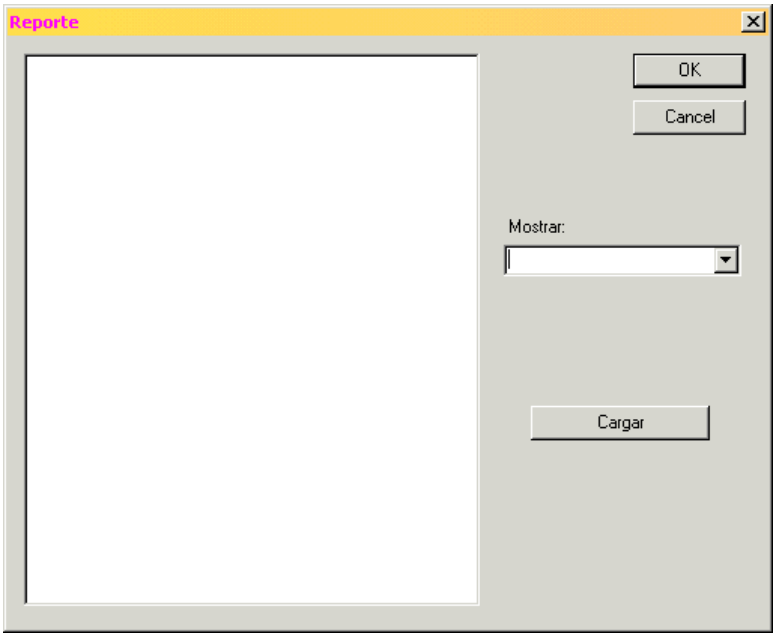


Este modo también puede ser llamado haciendo click en .

- En el combo **Comercial a reconocer** seleccionar el comercial aprendido por el sistema que se desea reconocer.
- Con el botón **Seleccionar Video** se despliega el cuadro de dialogo en el cual seleccionamos el video donde queremos reconocer el comercial aprendido.
- Haciendo click en **Reconocer** el sistema comienza a procesar el video seleccionado buscando a el comercial que se quiere reconocer. El sistema va reconociendo escena por escena el comercial seleccionado y sucesivamente nos va mostrando las escenas del comercial reconocidas. Por simplicidad mostramos la primera y la última escena reconocida por el sistema.

Luego de que el sistema termina de reconocer las escenas del comercial exhibe un mensaje diciendo que reconoció el comercial.

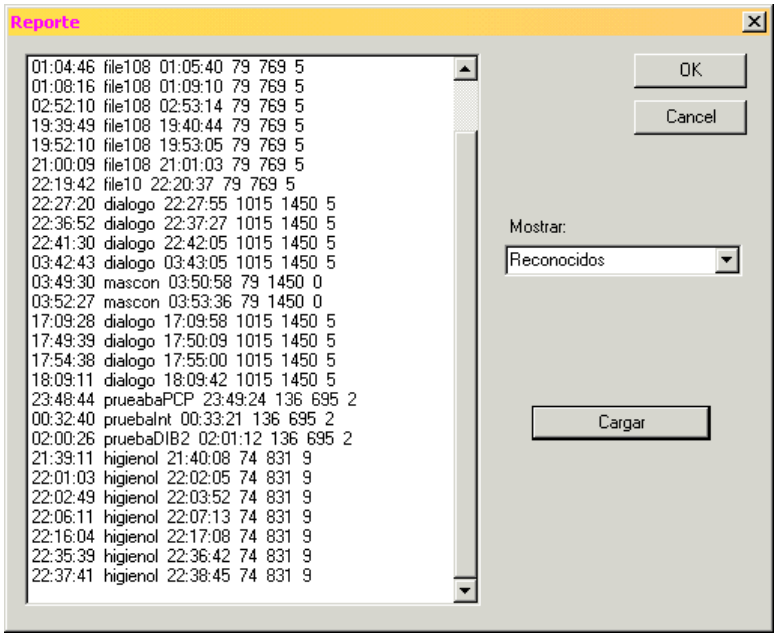
- Por último el usuario puede invocar a el modo reporte para poder ver la hora a la que se reconoció el comercial o en que cuadro apareció el comercial. Selecionar el modo **Reconcimiento,** se despliega la siguiente ventana:



También se puede seleccionar este modo haciendo click en .

- En el combo **Mostrar** se debe seleccionar el tipo de reporte que se desea hacer. En esta versión del programa hay dos opciones, Aprendidos o Reconocidos. Eligiendo estas opciones podemos ver los comerciales que el sistema tiene aprendidos y reconocidos respectivamente.

- Haciendo click en el botón **Cargar** se despliega el reporte seleccionado.



Los campos que se muestran en este reporte son, hora comienzo, nombre 0comercial, hora fin, cuadro comienzo, cuadro fin y escenas reconocidas.

APÉNDICE B - Archivos Bitmap

En este apéndice se pretende mostrar como es la estructura de un archivo bitmap, la comprensión de su estructura es vital para poder manipular este tipo de archivos, tanto para procesar imágenes en este formato como para extraerlas de un stream de video y editarlas en disco.

Archivos Bitmap

Los archivos bitmap en Windows son almacenados en un formato de mapa bits independientes del dispositivo(DIB Device Independent Bitmap) permitiéndole a Windows desplegar estos archivos independientemente del dispositivo que se utilice.
Cuando decimos "Independiente del dispositivo" significa que el bitmap especifica el color de los pixeles en una forma independiente de el método que usa el display para representar el color. La estructura de un archivo bitmap tiene la forma que se muestra en la figura1.

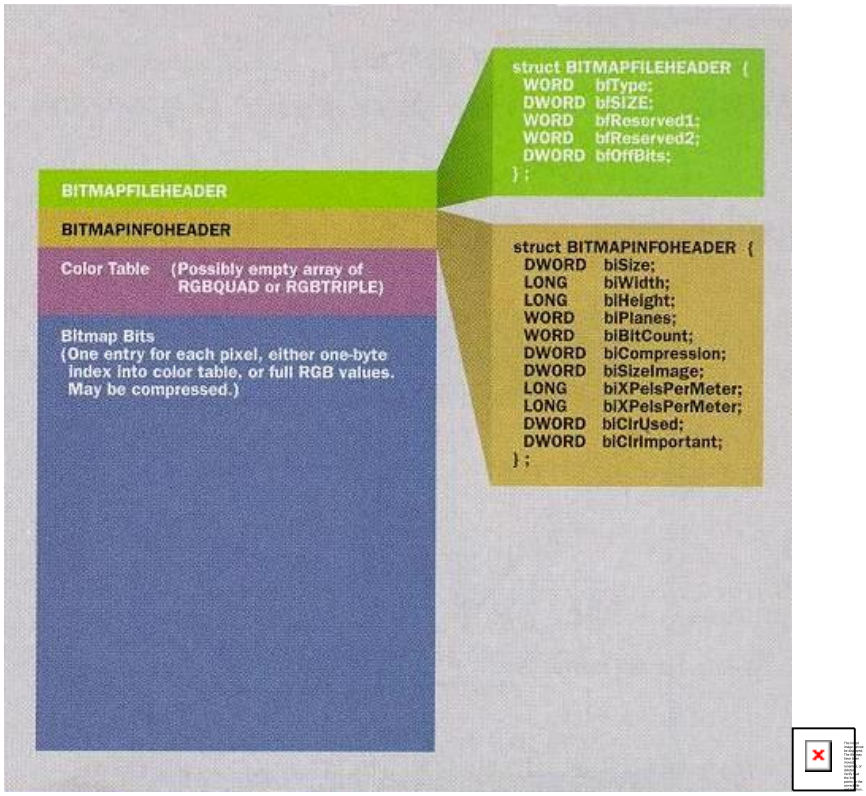


Fig. 1

Estructura de un archivo Bitmap

Cada archivo bitmap contiene una cabecera de archivo **BitMapFileHeader**,
una cabecera de información del bitmap llamada **BitMapInfoHeader**, una tabla de colores y un arreglo de bytes que contiene los bits del bitmap.

- **BitMapFileHeader**
  El BitMapFileHeader es una estructura de datos y contiene información del tipo, tamaño y la cantidad de bytes que ocupa la cabecera. El tamaño de esta estructura es de 14 bytes.

- **BitMapInfoHeader**
  El BitMapInfoHeader es una estructura que especifica propiedades de la imagen propiamente dicha como dimensiones, tipo de compresión y formato de color del bitmap. Esta cabecera conteniendo información de la imagen ocupa 40 bytes.

- **Tabla de colores**
  La tabla de colores se define como un arreglo de estructuras del tipo RGBQUAD y contiene tantos elementos como colores hay en el bitmap. Tal tabla de colores no esta presente para bitmaps que utilicen 24 bits de color ya que cada píxel de la imagen esta representado por 24 bits de rojo-verde-azul (RGB). Es de notar que si se usara una tabla de colores para 24 bits de color el archivo crecería en 2^24/8 bytes, esto da más o menos como 2MB extras de información.

- **Bits que componen la imagen**
  Lo que sigue inmediatamente después a la tabla de colores son los bits del bitmap, estos bits son un arreglo de bytes que representan filas consecutivas del bitmap. Cada fila representa los pixeles de una línea del bitmap ordenados de izquierda a derecha. Observar que el número de bytes por fila depende de el formato de color y de el acho del bitmap. Las líneas en un bitmap son almacenadas desde abajo hacia arriba.

Fue necesario comprender como estaba estructurado este tipo archivos ya que las funciones de la librería vfw que implementan la extracción de los cuadros, devuelven punteros a estructuras del tipo BITMAPINFOHEADER. Durante todo el desarrollo de la aplicación nos manejamos con esta estructura así como las que nombramos anteriormente.

APÉNDICE C

Análisis y Diseño en UML

Nicolás Scaglione, Martín Thomas, Pablo Varela.

Análisis del problema

Se describe el análisis del problema realizado de acuerdo a la metodología UML, en las fases que corresponden al planteamiento del problema, hasta la construcción de un diagrama de clases

Introducción

A efectos de realizar un estudio previo de ingeniería de software del problema, se comenzó haciendo un estudio general del problema a resolver, para este estudio se decidió aplicar la técnica de Análisis basado en patrones y el UML.
Esto permitirá tener desde el comienzo del desarrollo una idea consistente de los módulos, clases, e interfaces a desarrollar. Si bien es cierto que dada nuestra inexperiencia en desarrollo de aplicaciones de esta envergadura, muchas de las consideraciones iniciales serán sin duda diferentes al desarrollo final, esto es un hecho que el método de desarrollo basado en patrones prevé en las sucesivas iteraciones que lo componen.

Se cumplieron 3 etapas principales para esto que se describirán mas adelante.

1) Fase de planificación

-Conocimiento de los requerimientos
-Casos de uso

2) Fase de Análisis

-Construcción de un modelo conceptual
-Agregación de asociaciones
-Agregación de los atributos
-Registro de términos en un glosario
-Diagrama de secuencia del sistema
-Contratos

3) Fase de diseño

-Diagramas de colaboración
-Diagramas de clases del diseño
-Determinación de la visibilidad

Se utilizo el software *Together 6.0.1* como ayuda al desarrollo de los diagramas de clases, los diagramas de colaboración fueron realizados sin ningún software especial, y se utilizo el programa *Lan Flow* simplemente a efectos de dibujar los gráficos digitalmente.

Fase de planificación

El proyecto tiene por objeto la creación de un sistema de software capaz de reconocer en un video que es reproducido en un PC, desde algún medio de almacenamiento (por razones de tiempo vamos a trabajar con el disco duro del mismo PC), ciertas secuencias de interés que son previamente "aprendidas" por el sistema.

Clientes

Los tutores de proyecto de facultad. Eventuales usuarios que estén interesados en aplicaciones prácticas del software.

Metas

En términos generales la meta es automatizar el proceso de reconocer secuencias de video en un flujo de video cualquiera.
La meta incluye:

- Reconocimiento preciso de las secuencias a reconocer
- Reportes sencillos y exactos de las apariciones de esas secuencias

Funciones del sistema

Reproducir un archivo de video en pantalla

Capturar una secuencia de video elegida por el usuario (SAR)

Generar una Traza a partir de la SAR.

Almacenar Traza.

Registrar las Trazas almacenadas.

Mostrar las descripciones de las SAR capturadas y algunos cuadros como referencia.

Reconocer la aparición de una SAR, basándose en su Traza, cuando se reproduce una secuencia cualquiera de video (SED).

Generar aviso de reconocimiento cuando se detecta la aparición de una SAR.

Registrar información sobre las SAR reconocidas.

Almacenar algunos cuadros de las SAR reconocidas para chequear el reconocimiento.

Mostrar las descripciones de las SAR reconocidas y algunos cuadros como referencia.

Generar un Reporte con la información registrada de las SAR reconocidas.

Almacenar el Reporte.

Ofrecer un mecanismo de almacenamiento persistente.

Casos de uso

Se identificaron 3 casos de uso principales a saber:

a. Aprender las SAR.
b. Reconocimiento de las SAR.
c. Solicitar reportes.
d. Inicio del sistema.

De estos casos de uso nos concentramos en el a) por ser el mas importante para la realización de la primera fase del proyecto.
El caso de uso es útil, porque partiendo de una descripción verbal del funcionamiento del sistema, en una primera etapa esta concentrada en el curso normal de los eventos, es decir en el curso de acciones que se dan cuando la aplicación y el usuario hacen los pasos correctos.

Frontera del sistema

Se considera como sistema todo el hardware y software de la PC donde corre la aplicación, incluyendo algún dispositivo externo a la PC que se utilice como fuente de video.

Casos de uso en formato expandido

Caso de uso 1:	Enseñar las SAR o Aprender las SAR
Actores:	Usuario (iniciador).
Propósito:	Capturar la información necesaria sobre las SAR que el usuario elige.
Resumen:	Un Usuario selecciona el modo aprendizaje para enseñarle al sistema las SAR. El usuario elige el video que es reproducido, y selecciona las SAR que desea de este. Cuando el Usuario termina su selección, el sistema almacena la información necesaria sobre las SAR.
Tipo:	Primario y esencial.
Referencias Cruzadas:	*Funciones*: R1.1, R1.2, R1.3, R1.4, R1.5, R1.6, R1.7, R1.15

Curso normal de los eventos

Acción de los actores
U-Usuario
S-Sistema

U-1. Este caso de uso comienza cuando un Usuario, estando el programa abierto, selecciona el modo de aprendizaje.

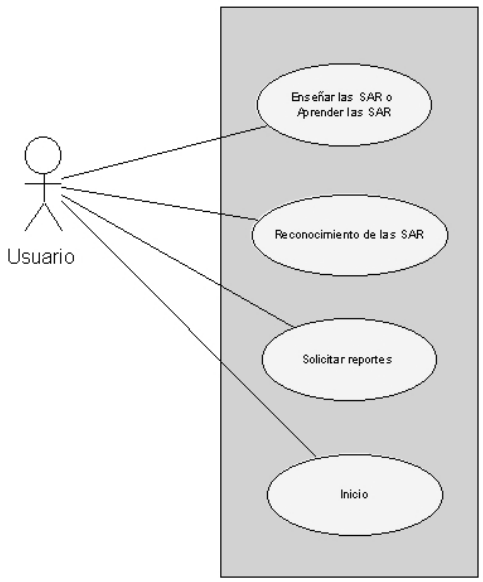
S-2. Acondiciona interfaz para modo de aprendizaje.

U-3. El Usuario selecciona el video a reproducir de donde elegirá las SAR.

U-4. El usuario "oprime play" para reproducir el video.

S-5. Reproduce el video en uno o varios *display* de la interfaz, en forma continua o mostrando cuadros representativos de las escenas o *shots*.
Se habilita mecanismo para que el Usuario pueda iniciar la captura.

Diagrama de los casos de uso



S-7. Determina el comienzo exacto de la SAR elegida.
Comienza a capturar la SAR.
Muestra que está capturando.

U-8. Cuando el Usuario distingue el final de la SAR, termina la captura.
Se habilita mecanismo para que el Usuario pueda terminar la captura.

S-9. Determina el final exacto de la SAR.
Termina de capturar la SAR.
Almacena información y algunos cuadros representativos de la SAR.
Muestra que finalizó la captura.

S-10. Genera la Traza a partir de la SAR.
Almacena la Traza.
Se habilita mecanismo para que el Usuario pueda iniciar la captura.

U-11. El usuario oprime *stop*.

S-12. Deja de reproducir el video.
Deshabilita mecanismo para que el Usuario pueda iniciar la captura.
Muestra opción para cambio de modo.

Cursos alternos

- Línea 11: el Usuario quiere enseñar otra SAR. Se pasa al evento 6.
- Línea 7: el Usuario oprime *stop*, el sistema lo toma como que finaliza la captura.
- Línea 7: finaliza el video, el sistema toma el último cuadro como fin de la captura. Se muestra mensaje de advertencia.

Fase de Análisis

Modelo Conceptual: Se identificaron los siguientes conceptos como relevantes en el dominio del problema.

Lista de Conceptos (Importantes para el Modelo Conceptual)

Objetos físicos o tangibles:

Computadora-
Disco duro
Pantalla
Tarjeta adquisidora

Especificaciones, diseño o descripciones de cosas:

Formato
Encabezado

Papel de las personas:

Usuario-

Contenedores de otras cosas:

Modo-
Directorio
Programa o ADRS (Aplicación de Reconocimiento de Secuencias)-
Fuente de video
Archivo
Video o SED -
Interfaz gráfica -
SAR -
Traza -
Buffer-

Cosas dentro de un contenedor:

SED o Video-
SAR-
Stream
Frame o Cuadro -
Visualizador de Imágenes
Botoneras de la Interfaz de usuario
Display
Keyframe

Concepto de nombres abstractos:

Linkeador -
Tijera o Captura -
Memoria temporal
Algoritmo de comparación
Condiciones
Lector
Extractor De Traza -
Especificación de SAR-

Eventos

Comparación
Adquisición
Lectura
Aprendizaje -
Reconocimiento positivo
Extracción de la SAR (como que lo abarca tijera o captura)
Lectura de la SED (pienso que se encarga visualizador)
Reporte de resultados

Manuales, libros

Manual de usuario

Conceptos esenciales del caso de uso Aprender la SAR.

Para elegir los conceptos iniciales se hizo lo siguiente:

- Una lista extensa de posibles conceptos usando la Lista de categoría de conceptos.
- Obtención de conceptos a partir del caso de uso en formato expandido.

Principales concepto

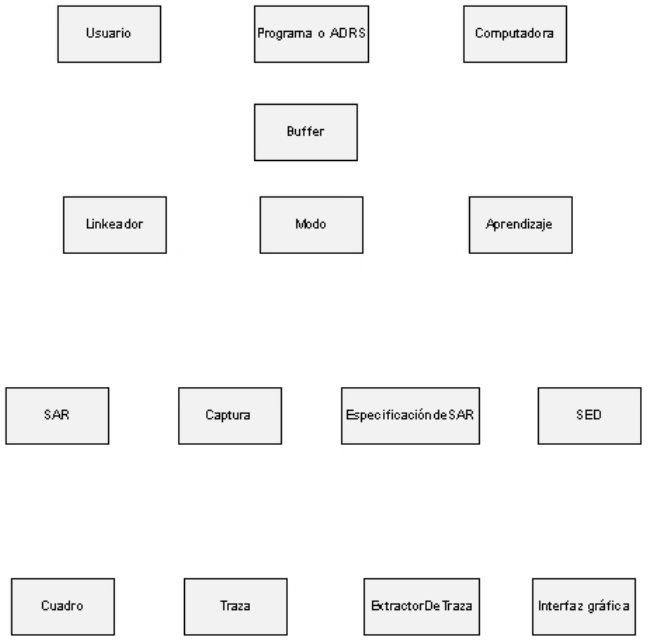
Entre ambas listas se hizo una depuración, en algunos casos intuitiva y en otros siguiendo recomendaciones (el cartógrafo por ejemplo).

La lista de conceptos es:

Usuario
Computadora
Programa o ADRS (Aplicación de Reconocimiento de Secuencias)
Aprendizaje
Modo
Linkeador
Interfaz gráfica
Video o SED
SAR
Especificación de SAR (ver especificar conceptos, y el caso de uso)
Captura o Tijera (ver cartógrafo)
Cuadro
Traza
Extractor De Traza
Buffer

De estos conceptos se hizo un diagrama conceptual que es complementado por relaciones entre los conceptos.
Luego de tener los conceptos definidos se agregan las relaciones que existen entre ellos por lo que al diagrama se le agregan asociaciones.

Modelo conceptual con relaciones y atributos (la lista no es extensiva)



Glosario de Términos

Aprendizaje Modo: Es el modo en que la aplicación esta lista para que el usuario le enseñe la SAR.
Reconocimiento Modo: Es el modo en que la aplicación esta buscando en la SED la o las SAR.
Reporte Modo: Es el modo en que el usuario extrae la información relativa a posibles reconocimientos de SAR en una SED ya analizada.
Linkeador: Aquella parte del software que almacena los directorios donde se almacenan los distintos archivos del sistema operativo.
Interfaz gráfica: Es la interfaz del programa con el usuario.
Video o SED : Secuencia de Datos, es el video que contiene la SAR.
SAR : Secuencia a Reconocer, es aquella parte de la SED, que contiene la parte que el usuario selecciono para ser reconocida.
Especificación de SAR: Es información relativa a la SAR que sirve para que el usuario la pueda identificar unívocamente.
Cuadro: Es un objeto que además de otras funciones a definir contiene información sobre una frame de video.
Traza: Es información que le es necesaria a la aplicación para poder hacer la búsqueda de la SAR en la SED.
Extractor de Traza: Aquella parte del software que utilizando la SAR genera la Traza.
Buffer: Es un Buffer para guardar información entrante al sistema.

Diagramas de Secuencia del sistema

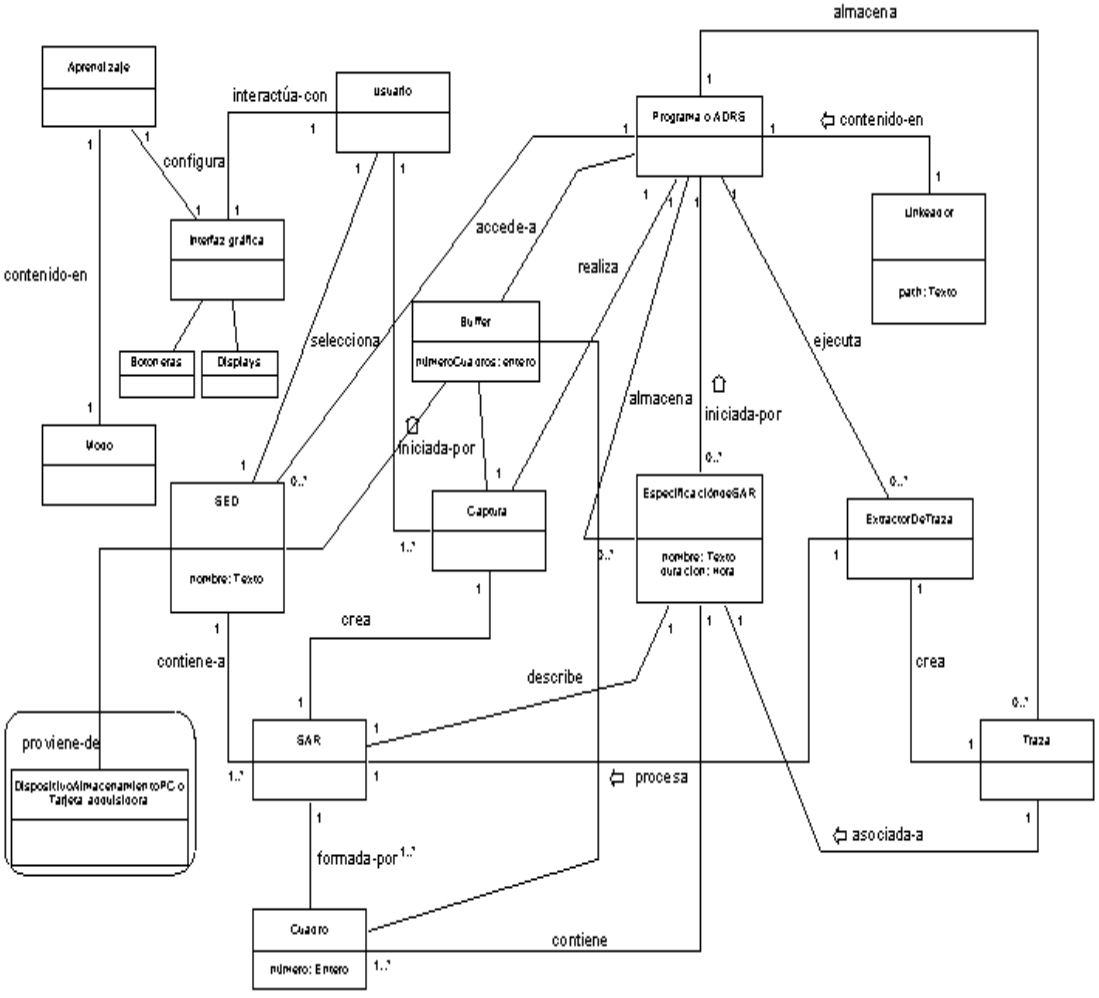


Diagrama de secuencia del caso de uso Aprendizaje.
Este diagrama ya permite ir esbozando el orden en que se realizan las funciones del sistema para el caso de uso dado.
Lo que hace cada función del sistema es definido en Contratos de las operaciones del sistema.

Contrato Inicio

Nombre: inicio ().

Responsabilidades: Inicializar el sistema.

Tipo: Sistema.

Referencias Cruzadas: Funciones: R1.2.
Caso de uso: inicio.

Notas:

Contrato

Nombre: seleccionarModo (aprendizaje: texto).

Responsabilidades: Registrar el modo y acondicionar la interfaz para modo de aprendizaje.

Tipo: Sistema.

Referencias Cruzadas: Funciones: R1.2.
Caso de uso: Aprender las SAR.

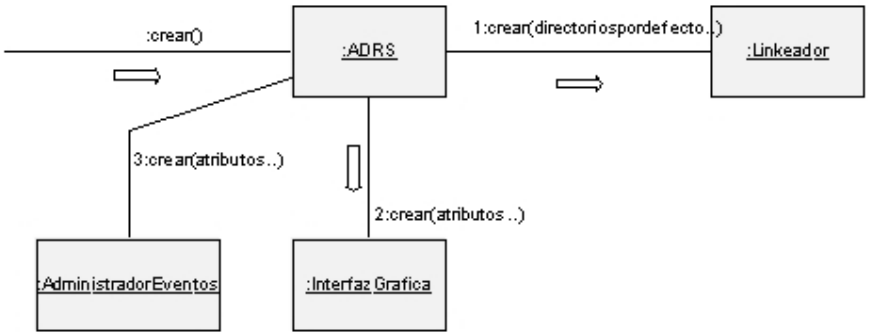
Notas:

Excepciones:

Salida:

Diagrama de colaboración de la operación Inicio

Caso de uso: comun al inicio de cualquiera si el sistema se inicia



Excepciones:

Salida:

Precondiciones:
Poscondiciones:

- Fue creada una instancia de ADRS, EspecificaciónDeSAR, *Linkeador* e Interfaz gráfica (creación de instancia).
- Una instancia de EspecificacióndeSAR fue asociada a ADRS (asociación formada).

Precondiciones:

Poscondiciones:

- Fue creada una instancia de Aprendizaje (creación de instancia).
- Una instancia de Aprendizaje fue asociada a Modo (asociación formada).
- Una instancia de Aprendizaje fue asociada a Interfaz Gráfica (asociación formada).
- Una instancia de Interfaz Gráfica fue asociada a Botoneras (asociación formada).
- Una instancia de Interfaz Gráfica fue asociada a Displays (asociación formada).
- Se definió para Botoneras.atributos los atributos correspondientes (modificación de atributo).
- Se definió para Displays.atributos los atributos correspondientes (modificación de atributo).

Comentario aparte: (Se creó una instancia de Interfaz Gráfica de Aprendizaje (creación de instancia), duda, si se define una interfaz

particular para aprendizaje o sino habría que ver como se hace, porque si no se tiene este concepto habría que modificar la botonera de la interfaz y el o los *display*, pero estos no me parece bien que se tomen como atributos del concepto interfaz.)

Contrato Seleccionar SED

Nombre:	seleccionarSED(path: Texto).
Responsabilidades:	Registrar el origen de donde proviene la secuencia de datos y acondicionar la interfaz gráfica para la reproducción.
Tipo:	Sistema.
Referencias Cruzadas:	Funciones: R1.2. Caso de uso: Aprender las SAR.
Notas:	
Excepciones:	Si la ruta no es válida, indique que se cometió un error.
Salida:	
Precondiciones:	La SED debe ser AVI y se permite que no este comprimida o ciertos formatos de compresión.

Poscondiciones:

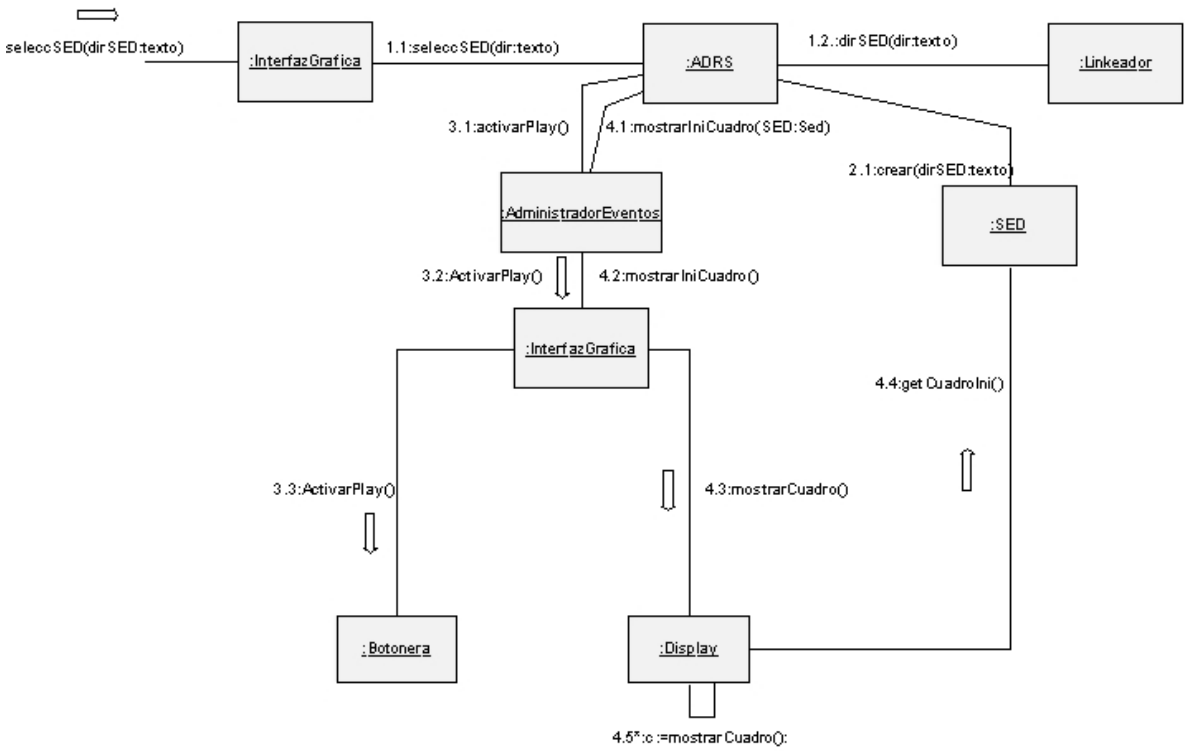
- Una instancia de ADRS fue asociada a 'Linkeador' (asociación formada).
- Se asignó 'path' a Linkeador.path (modificación de atributo).
- Fue creada una instancia de SED (creación de instancia).
- Se asignó nombre a SED.nombre (Lector) (modificación de atributo).
- Una instancia de SED fue asociada a ADRS (asociación formada).

Contrato

Nombre:	iniciarReproducciónSED().
Responsabilidades:	Desplegar en forma adecuada la SED en la interfaz gráfica. Pienso que faltan especificar mas responsabilidades.
Tipo:	Sistema.
Referencias Cruzadas:	Funciones: R1.1, R1.2. Caso de uso: Aprender las SAR.
Notas:	

Diagrama de colaboración de la operación seleccionar SED

Caso de uso: seleccionar SED



Excepciones: Si no se seleccionó la SED, indique error.

Salida:

Precondiciones: Se seleccionó SED de formato desplegable.

Poscondiciones:

- Fue creada una instancia de Buffer (creación de instancia).
- Una instancia de Buffer fue asociada a ADRS (asociación formada).
- Fueron creadas una o varias instancias de Cuadro (creación de instancia).
- Una instancia de Buffer fue asociada a una o varias instancias de Cuadro (asociación formada).
- Fueron asignados números a Cuadro.número (modificación de atributo). (-3,-2,-1,0,1,2,..)
- Una instancia de SED fue asociada a una o varias instancias de Cuadro (asociación formada).
- Fueron asignados números a Cuadro.número (modificación de atributo).
- Fueron eliminadas instancias de Cuadro (eliminación de instancias).

Observación: Este es un proceso continuo, dinámico, dejamos en claro que el sistema queda iterando las poscondiciones.

Contrato Iniciar Captura

Nombre: iniciarCaptura ().
Responsabilidades: Indicar a través de la interfaz gráfica que se está capturando. Determinar el comienzo exacto de la SAR.

Almacenar los cuadros de la SAR desde el inicio.

Tipo: Sistema.

Referencias Cruzadas: *Funciones:* R1.1, R1.2, R1.3. *Caso de uso: Aprender las SAR.*

Notas:

Excepciones:

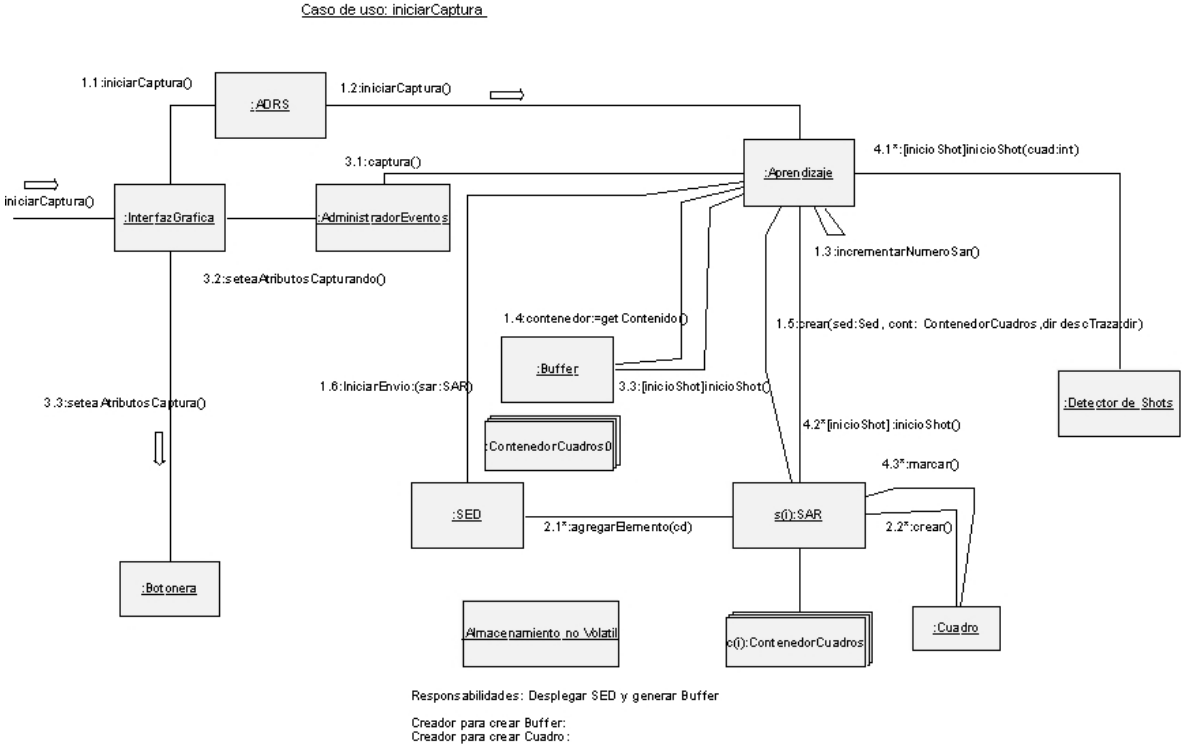
Salida:

Precondiciones: Se despliega una SED con el formato correcto.

Poscondiciones:

- Fue creada una instancia de Captura (creación de instancia).
- Una instancia de Captura fue asociada a ADRS (asociación formada).
- Una instancia de Captura fue asociada a Buffer (asociación formada).
- Una instancia de SED fue asociada a Buffer (asociación formada).
- Fue creada una instancia de SAR (creación de instancia).
- Una instancia de SAR fue asociada a Captura (asociación formada).
- Una instancia de SAR fue asociada a SED (asociación formada).
- Una o varias instancias de Cuadro fueron asociadas a SAR (asociación formada).
- Fueron asignados números a Cuadro.número (modificación de atributo).

Diagrama de colaboración de la operación iniciarCaptura



Contrato Finalizar Captura

Nombre: finalizarCaptura (numeroCuadro: entero).

Responsabilidades: Indicar a través de la interfaz gráfica que finalizó la captura. Determinar el final exacto de la SAR. Generar y almacenar la Traza. Almacenar información y algunos cuadros representativos de la SAR.

Tipo: Sistema.

Referencias Cruzadas: Funciones: R1.1, R1.2, R1.3, R1.4, R1.5, R1.6.
Caso de uso: Aprender las SAR.

Notas:

Excepciones: Si no se está capturando, indicar que ocurrió un error.

Salida:

Precondiciones:
Poscondiciones:

- Fue creada una instancia de ExtractorDeTraza (creación de instancia).
- Una instancia de ExtractorDeTraza fue asociada a ADRS (asociación formada).
- Una instancia de ExtractorDeTraza fue asociada a SAR (asociación formada).
- Fue creada una instancia de Traza (creación de instancia).
- Una instancia de Traza fue asociada a ExtractorDeTraza (asociación formada).
- Una instancia de SAR fue asociada a Especificaciónde SAR (asociación formada).
- Fue asignado nombre a Especificaciónde SAR.nombre (modificación de atributo).
- Fue asignada duración a Especificaciónde SAR.duración (modificación de atributo). (Podrían haber mas atributos que guardar)
- Fue asignado número de shots a Especificaciónde SAR.numshots (modificación de atributo).
- Una instancia de Traza fue asociada a Especificaciónde SAR (asociación formada).
- Una o varias instancias de Cuadro fueron asociadas a Especificaciónde SAR (asociación formada).
- Fue eliminada la instancia SAR.
- Fue eliminada la instancia ExtractorDeTraza.
- Una instancia de Traza fue asociada a ADRS (asociación formada).

- Fue eliminada la instancia Captura.
- Una instancia de ADRS fue asociada a Especificaciónde SAR (asociación formada).

Observación: Surge la idea de dos conceptos nuevos, Shot y DetectorShot. Proposición de Martín de lo que sería la traza. Sería una colección ordenada de subtrazas de Shots, es para dar la idea.

Contrato Finalizar reproducción SED

Nombre: finalizarReproduccionSED ().

Responsabilidades: Terminar de desplegar la SED.

Tipo: Sistema.

Referencias Cruzadas: *Funciones:* R1.1, R1.2.
Caso de uso: Aprender las SAR.

Notas:

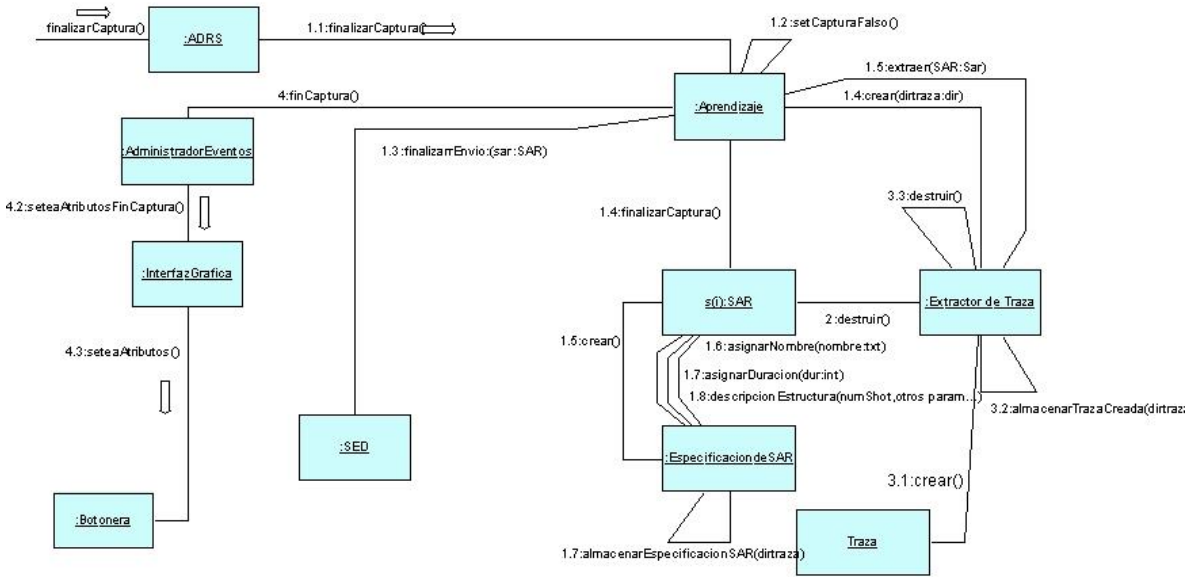
Excepciones:

Salida:

Precondiciones:

Diagrama de colaboración de la operación finalizarCaptura

Caso de uso: finalizarCaptura



Poscondiciones:

- Fue eliminada la instancia Buffer (eliminación de instancia).

Fue eliminada la instancia SED (eliminación de instancia).

De todos estos diagramas surge una idea de las clases y sus atributos, porque se concentran en algunas clases ciertas funcionalidades, son decisiones que fueron tomadas siguiendo los siguientes patrones GRASP

Experto
Control
Separación Modelo Vista
Bajo acoplamiento
Alta cohesión

El diagrama final de clases se muestra a continuación.

APÉNDICE D

Plan Inicial del Proyecto Entregado Previo a su Inicialización

Nicolás Scaglione, Martín Thomas, Pablo Varela.

Problemas o necesidad que genera el proyecto

Los anunciantes de publicidad gastan grandes sumas de dinero en anuncios. Estos se cobran por minuto y segundos en TV. Es de interés de anunciantes monitorear si realmente se pasan sus anuncios en el espacio contratado en un determinado medio de difusión audiovisual (TV abierta o TV cable) así como el número de veces, la hora específica y el tiempo contratado. Dicho control actualmente y solo en casos concretos se realiza por operarios contratados especialmente para eso.

Actores Involucrados en el Proyecto

Actores involucrados o interesados: Anunciantes, Agencias de publicidad, organismos de contralor (reguladora de comunicaciones, y otros), medios de difusión y demás empresas vinculadas a la publicidad.

Objetivo General

El objetivo general será desarrollar e implementar un software que sea capaz de:

a) Permitir al usuario seleccionar dentro de una secuencia de video ciertos *shots* consecutivos a ser reconocidos.

b) Reconocer los *shots* seleccionados dentro de una secuencia de video cualquiera.

c) Generar un reporte indicando las coincidencias.

d) El desarrollo podrá implementarse en un PC estándar. Se entiende por estándar el tipo de PC que se vende en plaza.

Definición de si el proyecto fue exitoso

El proyecto será exitoso si es capaz de:

a) Dada la secuencia buscada se genera un reporte con una probabilidad mayor que un cierto valor.

b) Dado un reporte existe una probabilidad mayor que un cierto valor, de que
corresponda a la secuencia buscada.

c) Cumplir con requerimientos de tiempo.

Estos ítems podrían variar según la aplicación que se le de al software.

Supuestos y restricciones

a) Contamos con hardware adquisidor de video cuyo uso nos sería facilitado por el instituto.

b) Las propiedades de color, luminancia y la relación señal a ruido de la secuencia a reconocer deben ser similares que la aparición de la misma en la secuencia donde se realizara la búsqueda.

c) Una restricción es que el proyecto va a estar dedicado a trabajar sobre la parte de imagen de la secuencia de video y no tomaremos en cuenta el audio.

d) Tiempo disponible para realizar el proyecto.

e) Pueden surgir restricciones si se decide procesar en un formato estándar determinado.

f) Un problema que nos planteamos es la posible dificultad que habría con el manejo de los archivos que contienen las secuencias de video con las que vamos a trabajar.

g) Otro problema inminente es la delimitación de las secuencias a manipular, o sea como decidir en que cuadro comienza nuestra secuencia a detectar y en que cual termina.

Estos puntos condicionan los entregables a realizar.

Objetivos Específicos

Realizar un cronograma y plan de trabajo de las actividades a desarrollar. Estas son:

a) Estudio de diferentes artículos relacionados con el tema.

b) Estudio del estándar de vídeo a utilizar, esto nos va a dar una pauta de cómo manejar la información a procesar.

c) Manejo de herramientas de manipulación de video.

d) Aprender lo necesario de herramientas de desarrollo de software como ser C y C++.

e) Hacer un modulo que lea el archivo que contiene una secuencia y sea capaz de reproducirla.

f) Modulo de delimitación de la secuencia de interés.

g) Modulo de procesamiento de la información necesaria para efectuar el reconocimiento.

h) Modulo de reconocimiento busca la secuencia de interés en el archivo de video en base a la información generada por el modulo de procesamiento. Nota: En estos módulos radica el núcleo de nuestro proyecto.

i) Pruebas para optimizar los algoritmos implementados.
j) Hacer que nuestro sistema pueda generar reportes de coincidencias y el momento de las coincidencias.

Principales Entregables

Primer entregable

Es aquella parte del software que presenta una interfaz gráfica con el usuario y realiza las funciones que se detallan a continuación.

Interfaz gráfica.

Consta de los siguientes elementos:

a) Una ventana con una pantalla para reproducción de un archivo de vídeo en el cual se seleccionara la Secuencia a Reconocer.

b) Botones para la reproducción, pausa, retroceso y avance del archivo de vídeo que se reproduce en la pantalla.

c) Botones necesarios para que el usuario pueda marcar en la secuencia de vídeo que se ve en la pantalla el principio y final de la Secuencia a Reconocer.

Funciones a realizar

Reproducir el archivo de vídeo en la pantalla.
Permitir al usuario elegir a su voluntad sobre la base de las escenas visualizadas en la pantalla, la Secuencia a Reconocer.
Generar un archivo que contiene la Secuencia a Reconocer elegida por el usuario.

Segundo entregable

Se detallan las tareas:

a) Software que realiza el procesamiento sobre la Secuencia a Reconocer previamente almacenada. Este procesamiento se refiere básicamente a la obtención de los keyframes de la secuencia por un método definido.

b) Software que captura el video (en principio de un archivo previamente almacenado) y compara el mismo con la Secuencia a Reconocer previamente procesada. Este software debe ser capaz de indicar de alguna forma cuando detecta la secuencia en cuestión. Para este entregable se trata de tener resuelta toda la problemática general que surja y no se realizan pruebas exhaustivas para verificar los porcentajes de error, dejando para el último entregable posibles optimizaciones de la algoritmia de comparación.

Tercer y último entregable

En este entregable se completará la realización del proyecto. Hasta aquí ya disponemos de buena parte del software de nuestro desarrollo.
En esta etapa estaríamos dedicados a optimizar los algoritmos de reconocimiento y comparación que implementamos, basándonos en pruebas que exigiremos a nuestro programa.
También nos dedicaríamos (una vez optimizado nuestro sistema) a verificar su eficacia. Esto implica analizar en que situaciones falla y en que situaciones no, o sea cuando detecta la secuencia y cuando no. Sobre la base de esto daremos un índice que reflejara cuantitativamente cuan eficaz es nuestro sistema.

Como bloque final de nuestro sistema elaboraremos un módulo de software que genere un archivo informando cuantas veces se encontró la secuencia buscada en una determinada secuencia de video , también podría devolver algún dato de a que hora fue la coincidencia y que duración tuvo.

El entregable consistirá en la presentación de un informe detallando la pruebas realizadas con sus resultados correspondientes y el programa completo con algoritmos optimizados.