Universidad Zaragoza
1542

Trabajo Fin de Grado

Definición y extracción de características en programas audiovisuales para el reconocimiento automático de temática

Autor:

Pablo Pérez Zarazaga

Director:

Emiliano Bernués del Río

EINA – Escuela de Ingeniería y Arquitectura

2015

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

D./Dª. Pablo Pérez Zarazaga,

con nº de DNI 76972686N en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Grado, (Título del Trabajo) Definición y extracción de características en programas audiovisuales para el reconocimiento automático de temática,

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 25/06/2015

Fdo: Pablo Pérez Zarazaga

Resumen del proyecto

La clasificación de contenidos audiovisuales en las cadenas de televisión hoy en día es una tarea dura dada la gran cantidad de vídeos que se producen cada día, además de todo el contenido almacenado que todavía se encuentra sin clasificar. La automatización de esta tarea ahorraría mucho tiempo y trabajo en las emisoras de contenidos audiovisuales, que podrían tener un programa informático que se ejecutase durante las 24 horas del día bajo una supervisión mínima.

Desarrollar un software de este estilo no es una tarea sencilla, ya que los videos pueden tener una calidad u otra en función de cómo han sido grabados y editados para su muestra al público, teniendo en cuenta que la máxima calidad posible será la de televisión estándar (*SDTV*). Además, también se depende de la gran variabilidad que aparece en el contenido de estos vídeos y de cómo se han organizado éstos en su realización.

El objetivo de este trabajo de fin de grado (TFG) es el desarrollo de una aplicación que permita clasificar distintos programas de televisión en función de su temática. Esta se desarrollará en el entorno de desarrollo Visual Studio 2013 y estará programada en lenguaje C++, diseñado como una extensión de C, manteniendo su sintaxis pero añadiendo las funcionalidades de la programación orientada a objetos.

Esta aplicación recibirá como entrada un vídeo cualquiera procedente de la televisión sin más información que el propio video. La aplicación se encargará de realizar un análisis en el que buscará las características más destacables del vídeo y se las enviará a un clasificador que decidirá a qué categoría pertenece.

Como resultado esta aplicación devolverá la categoría dentro de la que se encuentra el programa. Esto permite centrar futuros análisis en características más específicas en función del tipo de programa del que se trate, facilitando así la extracción de información del vídeo.

Índice de contenidos

Índice de figuras

1. Introducción

1.1 Motivación

La clasificación de contenido audiovisual en una cadena de televisión supone hoy en día un gran coste de tiempo y recursos, ya que no existe ninguna herramienta que facilite el proceso, sino que tiene que hacerse a mano por una persona. La automatización mediante un programa informático del proceso de análisis de los videos y de su posterior clasificación serviría para ahorrar una gran cantidad de tiempo, y por consiguiente de dinero. Este ahorro se vería reflejado tanto en el propio proceso de clasificación como en el posterior uso de estos datos, ya sea para para encontrar un vídeo clasificado dentro de una categoría como para un posterior análisis más profundo de las características del vídeo que necesite conocer previamente a qué categoría pertenece el video analizado.

1.2 Objetivos

El objetivo de este trabajo de fin de grado (TFG) es el desarrollo de un sistema de clasificación de contenido audiovisual basado en la temática de diferentes programas de televisión. Se van a investigar y probar diferentes algoritmos para la extracción de información de varios programas de televisión, de manera que se pueda implementar una aplicación que se encargue de analizar esa información y clasificar los vídeos conforme a las distintas temáticas que se definan.

La obtención de esta información se realizará a partir de vídeos tal y como son emitidos en televisión. Esto supone que la calidad del video será SDTV (Standard-Definition TV: 720x576) con una tasa de 25 imágenes por segundo. La aplicación recibirá como entrada uno de estos vídeos y, a partir del vídeo únicamente, extraerá las características principales y las analizará de tal manera que sea capaz de diferenciar entre unas categorías y otras. Como salida de la aplicación tendremos la categoría a la que pertenece el vídeo, la cual se encontrará entre las que definiremos más adelante.

Además del funcionamiento automático de la aplicación, se intentará también que el tiempo de procesado que requieran los algoritmos utilizados no sea muy grande, ya que uno de los principales problemas a solucionar es el ahorro de tiempo en el proceso de la aplicación. Esto se conseguirá tanto buscando algoritmos eficientes para el análisis de video como haciendo un uso más selectivo de los algoritmos más costosos, de manera que solamente se utilicen en las circunstancias en que sean imprescindibles.

1.2 Herramientas

Para la implementación de estos algoritmos y el desarrollo de la aplicación, se va a utilizar la herramienta Microsoft Visual Studio 2013, el cual nos proporciona un sencillo entorno de desarrollo para probar estos algoritmos. El lenguaje de programación en el que se programará será C++, ya que todas las librerías externas que vamos a utilizar se encuentran programadas en este lenguaje y proporciona una mayor eficiencia con respecto a otros lenguajes que podríamos utilizar, como C#. Además se va a aprovechar la librería de procesado de imagen OpenCV, la cual se encuentra programada también en C++ y nos proporciona algunos métodos que facilitan y aceleran el trabajo con matrices de gran tamaño utilizando el procesado mediante GPU.

2. Estudio del estado del arte

El principal reto al que nos enfrentamos en este TFG es que, tras realizar una búsqueda relacionada con la clasificación de video, no encontramos prácticamente información concerniente a clasificación en un contexto tan general como el que se plantea en este trabajo. La clasificación que se pretende desarrollar en este proyecto engloba todas las temáticas que se retransmiten por televisión, las cuales mezclan contenidos muy variados y que, a veces, ni siquiera guardan relación entre ellos.

El poco trabajo realizado en este campo de clasificación tan general supone que tendremos que realizar un análisis previo más exhaustivo para definir las características con las que trabajaremos. Además, gran cantidad de los algoritmos de clasificación desarrollados actualmente no servirán bien para nuestro propósito, ya sea porque están centrados en temáticas muy específicas o el propio método de análisis no se adapta a nuestras necesidades.

Actualmente, los métodos existentes se centran mayoritariamente en la temática de deportes. A continuación, vamos a comentar algunos de estos métodos actuales. Estos métodos, sin embargo, no nos serán de mucha utilidad en nuestro trabajo por varias razones. En primer lugar, no son aplicables para trabajar con otras temáticas. Otro inconveniente es que estos algoritmos se centran en analizar la información que contiene una única imagen. Sin embargo, estos métodos no se pueden generalizar al análisis de videos debido a que es muy costoso en tiempo afrontar el análisis de un vídeo como una sucesión de imágenes estáticas.

2.1 Métodos de clasificación de deporte

En [1] podemos observar un método de clasificación de deporte basado en el reconocimiento de objetos dentro de la imagen y la definición de eventos para clasificar las escenas analizadas. Este método analiza la imagen para reconocer los objetos que aparecen en ella y establecer una serie de tags que servirán tanto para definir el evento que se está desarrollando como para clasificar la escena. En la figura 1 podemos ver un ejemplo de este algoritmo aplicado a una regata en un lago.

Este método, sin embargo, no es útil para nosotros porque el reconocimiento de objetos es un proceso lento y costoso que afectaría al progreso del programa. Además, para utilizar un reconocedor de objetos necesitamos tener una base de datos previa y que la calidad de la imagen sea lo suficientemente alta.

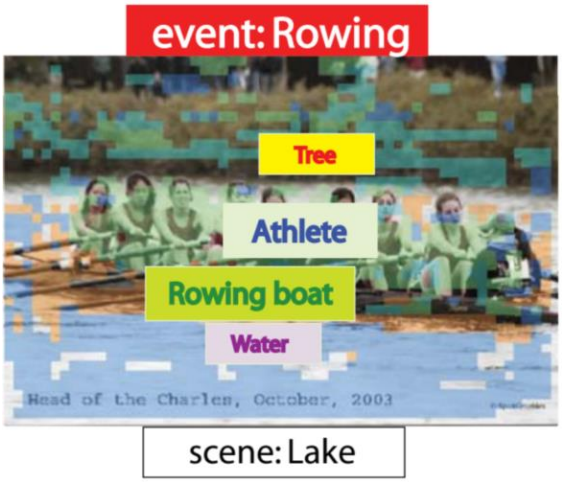

Figura 1: Ejemplo del método explicado en [1] con tags sobre una imagen

Otro método utilizado para clasificación de deporte es el explicado en [2]. Este método tiene dos pasos, en primer lugar analiza el color de la imagen para extraer datos del campo de juego y delimitar la zona de análisis y después extrae las siluetas de los jugadores para estudiar las posturas de éstos. Una vez tiene los datos del campo y definidas las posturas de los jugadores utiliza redes neuronales para decidir qué tipo de deporte está analizando.

Este método, aunque utiliza algoritmos bastante sencillos cuya ejecución no conllevaría mucho trabajo, necesita una imagen de buena calidad donde se distingan claramente los jugadores para extraer sus posturas. Como veremos más adelante, las imágenes correspondientes a deportes con las que contaremos nosotros tienen mucho movimiento que difumina las siluetas de los jugadores o están tomadas desde demasiado lejos como para definir a éstos de una manera fiable.

En [4, 5] encontramos más métodos que analizan una imagen de un evento deportivo y extraen sus características para clasificarlas aplicando distintos algoritmos, sin embargo, plantean problemas similares a los que tenemos con los dos algoritmos anteriores y no nos serán de utilidad en este TFG.

Dado que pretendemos analizar un vídeo completo y no solamente una imagen, centraremos nuestro análisis en estudiar la evolución del flujo de imágenes a lo largo de un periodo de tiempo. Por ello, dos características muy útiles que podremos extraer serán el movimiento que se produce dentro de la imagen entre dos frames consecutivos y el fondo estático de una escena, el cual nos servirá para retirarlo del análisis de la imagen y aislar así los objetos en movimiento.

2.2 Extracción de movimiento de la imagen

Para estudiar el flujo óptico, es decir, el movimiento a lo largo de una secuencia de frames dentro de un vídeo, existen múltiples algoritmos. Algunos necesitan establecer primero unas zonas de interés, a partir de las cuales intentarán predecir la próxima posición que tendrán dentro de la imagen. Otros algoritmos simplemente analizan la imagen entera extrayendo el movimiento de cada uno de sus píxeles. Este es el caso del algoritmo de Farnebäck.

2.2.1 Algoritmo de Farnebäck

Como podemos ver en [5], el algoritmo desarrollado por Gunnar Farnebäck busca estimar el movimiento de cada punto de la imagen aproximando su evolución mediante expresiones polinómicas. Este algoritmo forma parte de los estimadores de flujo óptico de carácter denso, es decir, que calcula el movimiento de todos los píxeles de la imagen. Estos algoritmos suelen ser bastante lentos porque tienen que trabajar con una gran cantidad de puntos. Sin embargo, el algoritmo de Farnebäck, al utilizar expresiones polinómicas simplifica mucho la carga computacional y consigue unos tiempos de procesado mucho menores que otros algoritmos similares. Este ahorro de tiempo supone que el algoritmo no sea perfecto, sin embargo el compromiso calidad/tiempo es muy bueno.

Al final se acaba obteniendo una matriz de puntos que representa el movimiento de cada pixel desde su posición inicial hasta su posición en la última imagen. Estos datos son lo que conocemos como vectores de movimiento y en la figura 2 podemos ver una representación de éstos.

Figura 2: Ejemplo del algoritmo de Farneback aplicado a un partido de tenis

2.3 Extracción del fondo

Para ayudar a algunos algoritmos que utilizaremos en la clasificación, emplearemos en la mayoría de los casos un proceso de segmentación que consistirá en separar la parte de la imagen que forme parte del fondo

(background) de la parte de la imagen compuesta por los elementos en movimiento. En los artículos estudiados se emplean tanto la estimación de un modelo de fondo como la detección y substracción del color dominante de la imagen.

2.3.1 Estimación del fondo de una secuencia de imágenes

En este método se obtiene, a partir de una secuencia de imágenes adquiridas, una función de densidad de probabilidad tridimensional en la que cada dimensión corresponde a una de las componentes de color, para cada uno de los píxeles de la imagen. De esta forma, aunque haya elementos en movimiento dentro de la imagen, al realizar la estimación a lo largo del tiempo, será más común el color del fondo que el de los objetos que afecten al histograma de cada pixel.

A pesar de ser el método de estimación de fondo que parece más robusto, esta técnica solamente funcionará correctamente en situaciones en que no se produzcan movimientos de cámara, tanto físicos como ópticos.

2.3.2 Detección y substracción de color dominante

Éste es el algoritmo más utilizado en la clasificación de deportes, ya que el color del campo de juego no es uniforme en la mayoría de los deportes, y varía según las condiciones climáticas o la iluminación.

Este método, que se encuentra explicado en [6], aprovecha la uniformidad del campo de juego en el deporte, de manera que el fondo puede caracterizarse únicamente con un color dominante, el cual puede variar a lo largo del tiempo en función de las condiciones de iluminación de la imagen.

Cada pixel de la imagen se clasifica en función de la distancia, ya sea euclídea (en el caso del espacio de color RGB) o cilíndrica (en el caso del espacio de color HSV), entre el color dominante en la imagen y el mismo píxel. Así se puede obtener una máscara en la que se pueden observar únicamente las siluetas de los jugadores.

Esta máscara, la cual podemos observar en la figura 3, nos permitirá aislar los elementos que no se consideren parte del fondo de manera que el resto de la imagen (fondo) quedará en negro tras aplicar la máscara sobre la imagen.

Como hemos nombrado en el segundo algoritmo de clasificación de deportes, los métodos de análisis del color para extracción del campo de juego en deportes no necesitan mucho tiempo de procesado y pueden ser muy útiles para aislar los elementos en movimiento dentro de la imagen, por lo que en este caso sí que aprovecharemos el algoritmo en nuestro proyecto.

Figura 3: Máscara obtenida de aplicar substracción del color dominante en una imagen de un partido de fútbol

3. Desarrollo del proyecto

3.1 introducción

Como ya hemos mencionado anteriormente el objetivo del proyecto es desarrollar una aplicación que extraiga las características de un vídeo y las analice para automatizar el proceso de clasificación de estos videos en función de su temática. El entorno de desarrollo utilizado ha sido Visual Studio 2013 y el lenguaje en el que se ha programado es C++.

En este apartado se va a explicar detalladamente todos los pasos que se han seguido en el desarrollo de este TFG. Para ello seguiremos el orden con el que se ha desarrollado la aplicación, que es el orden habitual de un análisis de video.

En primer lugar, dado que el objetivo es discriminar los vídeos entre distintas categorías, debemos estudiar qué categorías son estas, analizando para cada una cuáles son sus características más importantes. Después analizaremos los algoritmos que se pueden utilizar para extraer estas características del video y elegiremos los más apropiados en cada caso. Por último, tenemos que definir un decisor que se encargue de recopilar estas características y asignar al vídeo analizado una de las categorías posibles.

3.2 Definición de categorías

En primer lugar, antes de centrarnos en el análisis del vídeo, debemos definir qué tipos de programas analizaremos, de manera que podamos decidir qué características obtener del vídeo para que la decisión sea lo más fiable posible. A continuación se explicarán los diferentes tipos de programas definidos así como algunas de sus principales características.

Informativos

Dentro de la categoría de informativos encontramos programas no solo como el telediario de cada cadena, sino que podemos encontrar gran cantidad de variantes que engloban esta categoría:

- Flash (Avances informativos)
- Telediario
- Reportajes de actualidad (Informe semanal)
- Entrevistas

Estos programas suponen un problema si los analizamos desde el punto de vista de contenido, ya que hay una gran variedad en lo que se muestra en cada uno de ellos. Sin embargo, hay una cosa que todos estos programas tienen en común y que los diferencia de los demás programas, y es una marcada estructura.

Figura 4: Ejemplos de presentador en dos de los programas de la categoría de informativos

Las dos escenas más importantes de este tipo de programas son los primeros planos en plató en los que aparecen los presentadores presentando las noticias como en la figura 4 y los reportajes correspondientes a cada noticia. Estos dos tipos de escenas se van intercalando manteniendo una duración prácticamente constante y esta característica puede ayudarnos a diferenciarlos de los demás tipos de programas.

Deportes

Dentro de esta categoría encontramos una gran cantidad de tipos distintos de deportes, sin embargo nos centraremos en los tres más televisados, fútbol, baloncesto y tenis.

Figura 5: Imágenes características de un partido de fútbol y uno de baloncesto

Los deportes televisados se caracterizan, como podemos ver en la figura 5, porque la cámara no suele tener mucho movimiento, mientras que los jugadores se mueven repartidos por el campo de juego de una manera u otra dependiendo del deporte. Este fondo estático con los jugadores en movimiento nos servirá para decidir si las escenas que estamos analizando pertenecen a un deporte o a otro tipo de programa.

Entretenimiento

En esta categoría reuniremos programas como galas, concursos, talk shows o magazines. Esta es la categoría que más variedad de contenido tiene y

en la que englobaremos la mayoría de los programas que no podamos clasificar en ninguna de las demás categorías

Culturales/Divulgativos

Aquí encontramos programas como documentales, musicales o especializados en distintos temas culturales. La variedad de contenido que encontramos aquí dentro es similar a los programas de entretenimiento e incluso algunos programas podrían ocupar ambas categorías.

3.3 Extracción de características

Antes de comenzar con el análisis, tenemos que tener en cuenta que se va a trabajar con un vídeo que va a contener distintas escenas. Para poder trabajar con el vídeo y aplicar algunos algoritmos desarrollados, necesitamos, en primer lugar, separar cada una de las escenas del vídeo. Para ello vamos a desarrollar un programa auxiliar que realice esta tarea de forma automática y nos devuelva los datos en un formato sencillo.

Después de desarrollar el programa de separación de escenas, nos vamos a centrar ya en la extracción de características para cada una de las categorías nombradas anteriormente. La elección de los algoritmos de extracción de características se verá influenciada por el decisor que se utilizará de forma definitiva. Este decisor, que explicaremos más adelante, será un árbol de decisiones binario que irá determinando las categorías de los programas en orden.

Basándonos en esto, analizaremos en primer lugar los algoritmos utilizados para diferenciar los programas de informativos del resto, los cuales consisten en un análisis de la estructura del programa. Después del análisis de estructura pasaremos a los algoritmos utilizados para el reconocimiento y clasificación de deportes, para los que recurriremos a un análisis del color de las imágenes y otro del movimiento que hay dentro de ellas.

3.3.1 Separación de escenas

Como hemos comentado, antes de centrarnos en extraer las características del video tendremos que separar éste por escenas definiendo donde se producen los cambios de plano para poder analizar cada una de estas por separado.

El programa auxiliar que hemos desarrollado se encarga de recorrer el vídeo estudiando cómo evolucionan las imágenes a lo largo del video y, cuando detecta que se ha producido un cambio de plano, escribe el frame correspondiente en el que se ha producido en un fichero ASCII. El

funcionamiento completo del programa auxiliar se explica detalladamente en el Anexo 1.

El fichero ASCII que obtendremos como salida tendrá el mismo nombre que el vídeo analizado y se encontrará en la misma carpeta que éste. De esta manera escribiendo solamente la ruta del vídeo podremos acceder fácilmente a él. Además, el formato del fichero auxiliar está definido de tal manera que cada cambio de plano se encuentra escrito en una línea distinta, así que para leer los cambios de plano solo es necesario leer línea a línea el fichero ASCII.

3.3.2 Definición de primeros planos

La primera categoría en la que nos centraremos son los informativos, en los que la característica predominante es la marcada estructura que tienen. Esta estructura la analizaremos estudiando la duración de los primeros planos en los que aparece el presentador del programa así como la de los reportajes que tienen normalmente una duración muy poco variante. Para realizar este análisis definiremos en primer lugar qué consideraremos nosotros como primer plano.

La escena que buscamos clasificar como primer plano es una en la que se vea claramente la cara del presentador de frente y con un movimiento mínimo, tal y como sucede en un plató de televisión. Para encontrar estas escenas necesitaremos implementar, en primer lugar un detector de caras y, en segundo lugar una función que se encargue de medir el movimiento total de la imagen.

Búsqueda de caras

En primer lugar, para identificar los primeros planos dentro de una imagen necesitamos buscar qué caras se encuentran en este plano y si cumplen los requisitos necesarios para que se considere un primer plano de un presentador de un programa. Para esta búsqueda recurriremos a un clasificador en cascada.

Un clasificador en cascada es un algoritmo muy usado en la búsqueda de caras que se basa en determinar si ciertas zonas de la imagen se corresponden o no con las características de una cara. Dado que normalmente, la mayoría de las zonas que se investigarán no contendrán una cara y no contendrán la característica que estemos analizando, el algoritmo se encargará de descartar las zonas de la imagen donde se verifique que no se encuentra la característica buscada. De esta manera, si analizamos una serie de características, solamente quedarán consideradas como caras esas zonas de la imagen que hayan pasado todos los filtros. Este algoritmo consigue una gran eficiencia porque las primeras capas de filtrado utilizan características más generales que suponen un análisis muy rápido, dejando una zona bastante reducida para las capas de análisis que hay después y que cuentan con características más complejas.

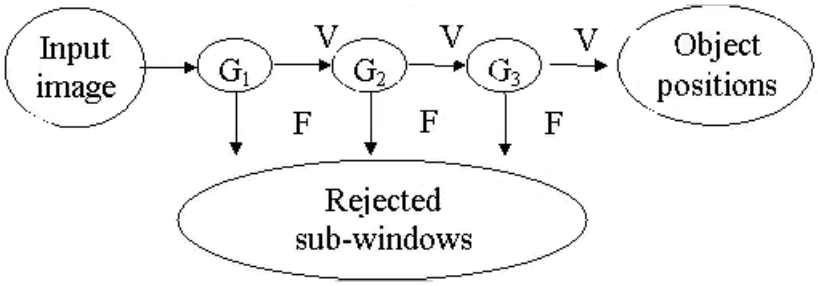

Figura 6: Esquema de un clasificador en cascada

Este clasificador, cuyo esquema se puede ver en la figura 6, no solo sirve para detección de caras, sino que se puede utilizar para cualquier tipo de objeto una vez definidas las características de las capas del clasificador. Para ello, es necesario entrenar el clasificador para que se adapte al objeto que nosotros vayamos a buscar en la imagen. Afortunadamente, la librería OpenCV nos proporciona varios ficheros en los que se definen estas características de manera que sea fácil entrenarlo. Para nuestro caso elegiremos el fichero entrenado para caras que se encuentran de frente mirando a la cámara ("haar_cascade_frontalface.xml").

Figura 7: Ejemplo del detector de caras con clasificador en cascada aplicado sobre una imagen de informativos

Una vez hemos obtenido las caras que hay en la imagen, representadas como un rectángulo que contiene la cara, tal y como podemos ver en la figura 7, no podemos quedarnos con todas ellas. Para determinar si de verdad se trata de un primer plano tendremos que buscar las caras que tengan un tamaño mayor que un umbral que podemos controlar. Cuando tengamos las caras que cumplen los requisitos de tamaño, tendremos que comprobar que no hay muchas de ellas en la imagen. En todos los programas de informativos que se encuentran dentro de la categoría hay solamente un presentador, y en algunos casos como algún telediario podemos encontrar dos presentadores. Sin embargo nunca habrá más de dos o tres caras que cumplan las condiciones de tamaño en un mismo plano, así que desecharemos todas las imágenes que contengan más de tres caras en el plano principal.

Medición de movimiento

Como explicamos anteriormente, una vez hemos detectado las caras que aparecen en la imagen, tenemos que comprobar que no se produce un gran movimiento dentro de la imagen.

Para medir en movimiento en este caso no necesitaremos recurrir al algoritmo de Farnebäck, sino que bastará con evaluar dos o tres imágenes consecutivas pixel a pixel y obtener el valor de la variación de cada pixel a lo largo de esta sucesión de imágenes. Si la variación de un pixel es superior a un umbral consideraremos que ese pixel se ha movido y lo marcaremos para analizarlo después.

Cuando ya tengamos todos los pixeles evaluados por el umbral sumaremos todos los que han sido marcados como pixel en movimiento y tendremos una medida aproximada del movimiento total de la imagen. Si esta cantidad de movimiento no supera el 10% de la cantidad de puntos total de la imagen, consideraremos que la imagen analizada es un primer plano del presentador del informativo.

3.3.3 Análisis de la estructura

Ahora que ya tenemos la manera de definir qué planos son los primeros planos que nos interesan y cuáles no, iremos etiquetando cada uno de manera que podamos hacer un recuento y clasificar cada tipo de programa.

Para evitar tener que analizar todos los frames del video consideraremos que la escena mantiene sus características a lo largo de toda su duración, así solamente tendremos que analizar las primeras imágenes de cada plano ahorrando así tiempo de cómputo y aumentando la eficiencia del programa.

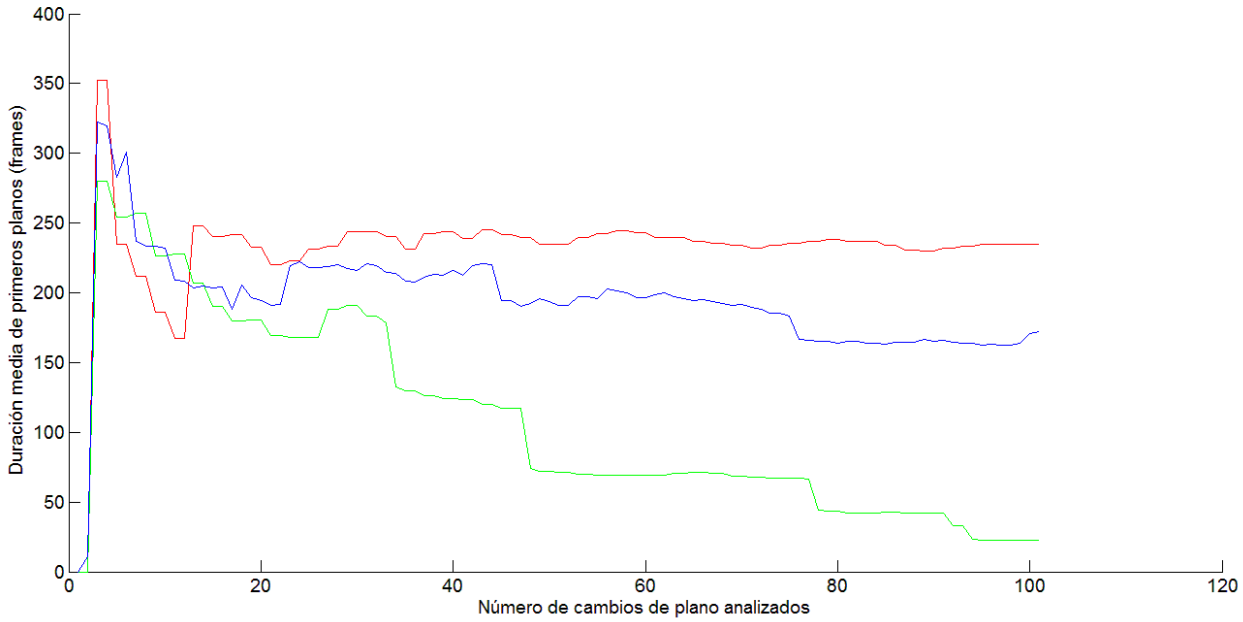

Figura 8: Gráfica de la evolución de la duración de los cambios de plano en vídeos de las tres categorías

En la figura 8 podemos ver una gráfica que representa los valores medios que toma a lo largo del tiempo la duración de los primeros planos en frames. Para realizar la medida hemos utilizado diez vídeos de cada categoría, almacenando valores hasta que se observa un cierto estado estacionario en las dos categorías que nos interesan.

Al principio de la gráfica podemos observar como no se pueden distinguir ninguno de los tres tipos de programas debido a que no se han evaluado suficientes valores como para que se estabilice la medida. Sin embargo, a partir de 20 cambios de plano y según se van analizando más vemos que los valores adoptan cierta tendencia que se mantiene hasta el final.

La medida que nos interesa a nosotros es la que podemos obtener al final del análisis, es decir, una vez se han analizado los 100 cambios de plano y el resultado se mantiene estable. A partir de ahora, cada vez que nos refiramos a los valores de la gráfica nos estaremos refiriendo al valor final de la medida.

En verde podemos ver la duración media de los primeros planos en videos de deportes. Como podemos observar, está muy por debajo de las otras dos categorías por lo que no habría ninguna dificultad a la hora de discriminarla de las otras dos.

En rojo representamos la categoría de informativos, mientras que en azul una mezcla de todos los demás programas que se encuentran fuera de las dos categorías. Podemos observar que la duración de los primeros planos en los informativos ronda alrededor de los 240 frames, bastante por encima de los 130 en los que se encuentran los demás programas. Estableciendo un umbral en la duración de los primeros planos de 200 frames podremos distinguir la mayoría de los programas de los informativos. De esta manera, si la duración media de los primeros planos se encuentra por encima de este umbral consideraremos que estamos analizando un informativo.

Sin embargo, tenemos que tener en cuenta que la medida de la gráfica es únicamente una media de varias observaciones. Puede darse el caso en que dentro de la categoría de otros vídeos, dada la gran variedad de contenido que ésta engloba, se supere este umbral. Para solucionar esto también tenemos como dato la duración de un informativo en televisión, que es siempre un valor constante que está entre 45 minutos y 1 hora. Teniendo en cuenta este dato, solamente habrá que analizar como posibles informativos aquellos programas que se encuentren dentro de estos dos valores de duración. Estamos trabajando con video a 25 frames por segundo, por lo tanto los umbrales de duración total del programa serán 67500 frames en el límite inferior y 90000 frames en el límite superior.

Con este algoritmo seremos capaces de diferenciar los programas de la categoría de informativos. Ahora pasaremos a analizar los algoritmos necesarios para la clasificación de deportes, estudiando en primer lugar métodos de análisis del color y después el estudio del movimiento a lo largo de una escena.

3.3.4 Análisis del color

Centrándonos ahora en deporte, uno de los elementos más característicos que encontramos es el campo de juego. En este apartado estudiaremos como delimitar el campo de juego y extraerlo de la imagen dejando únicamente los objetos que hay sobre él. Este método nos ayudará en el posterior análisis de movimiento para aislar los distintos jugadores que hay por el campo.

Para desarrollar el análisis del color y la extracción del campo de juego necesitamos en primer lugar estudiar qué espacio de color nos favorece más para los algoritmos que vamos a utilizar.

Espacio de color HSV

La representación de color en RGB es una forma precisa e intuitiva de almacenar y trabajar con imágenes. Sin embargo, en nuestro caso estamos analizando imágenes donde la combinación de colores puede cambiar mucho de un pixel a otro y a simple vista seguir pareciendo una superficie uniforme. Además, tenemos que tener en cuenta que las imágenes que entrarán al programa pueden haber sido grabadas en cualquier momento del día. Esto hace que puedan existir sombras o cambios de brillo que afecten gravemente a la hora de analizar el color. Un ejemplo lo podemos ver en la figura 9, donde el partido de tenis fue jugado a una hora del día en la que el ángulo del sol producía una gran sombra que dividía el campo en dos.

Figura 9: Partido de tenis donde el campo ha dejado de parecer una superficie de color uniforme

Para evitar este efecto, tendremos que buscar un espacio de color que se aproxime más a la percepción humana que a una fácil representación de éste. Esto será capaz gracias a una transformación que haremos desde la representación en RGB de los puntos de la imagen al espacio HSV. Este espacio de color [7, 8] consta de tres componentes, H (HUE), S (saturación) y V (value), con las cuales podemos aislar la componente de color de los demás factores que pueden afectarle.

- HUE (color): Esta componente representa el color. El valor del color se representa con un ángulo entre 0º y 360º. Los ángulos comprendidos

entre 0º y 60º son los colores rojos, los que se encuentran entre 120º y 180º son los verdes y aquellos cuyo ángulo está entre 240º y 300º son azules. Los ángulos que se encuentran entre ellos son todos los colores mezcla de éstos.

- Saturación: La saturación indica la pureza del color con respecto a una referencia de color blanco. Esta componente toma valores entre 0 y 1, que también pueden tomarse como porcentajes. Un valor de 0% es un color gris o blanco, mientras que una saturación del 100% es un color primario.
- Value (brillo): El valor o brillo es la percepción de la luz que contiene el color. Al igual que la saturación, el brillo se representa mediante porcentajes. Cuando el brillo vale 0, el color que veremos será negro, y a medida que se incrementa el valor veremos el color más claro.

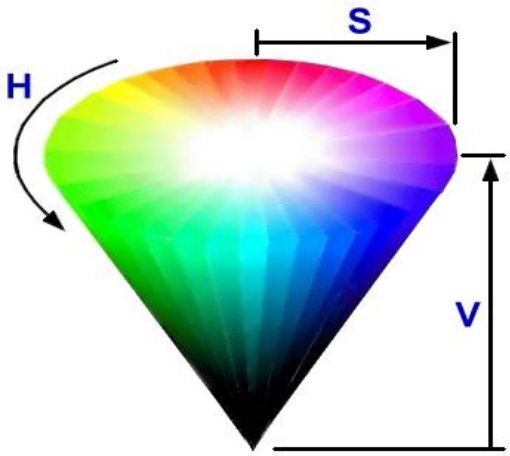

Figura 10: Representación esquemática del espacio de color HSV

La figura 10 muestra claramente la representación del espacio de color HSV como un cono en el que la base forma el círculo cromático recorriendo todos los valores de color, mientras que las componentes de saturación y brillo se representan como la distancia al centro del cono y a la base respectivamente.

Este espacio de color sirve muy bien para nuestro propósito, ya que, además de aislar los valores de color de la saturación y brillo que afectan a nuestra percepción, representa esta componente de color dentro de una sola componente, lo que facilita bastante su análisis. Gracias a este espacio de color, para buscar un color en especial dentro de la imagen solamente tenemos que buscar el sector circular, más ancho o más estrecho, dentro del círculo que forma la componente de color.

Extracción del campo de juego

A la hora de extraer el campo de juego, nos hemos planteado utilizar algunos algoritmos como el mencionado en [6]. Sin embargo, tenemos que tener en cuenta que vamos a analizar una gran cantidad de contenido que no va a tratarse de deporte, sino que habrá muchos programas distintos a analizar. Por

ello hemos decidido no recurrir a un algoritmo tan específico, sino simplemente aprovecharemos las características del espacio de color HSV para obtener la superficie de color uniforme que forma el campo buscando unos resultados similares a los de algoritmos más complejos.

Como hemos nombrado antes, al transformar la imagen de un espacio de color RGB al espacio HSV reunimos los valores de color dentro de una sola componente. Para extraer el campo de juego necesitaremos encontrar una gran superficie de un color uniforme dentro de la imagen, la cual consideraremos el campo si nos encontramos bajo ciertas condiciones.

En primer lugar, para buscar una gran superficie de color uniforme tendremos que buscar el color predominante dentro de la imagen. Una vez tengamos la imagen en HSV nos quedaremos solamente con la componente de color (HUE), y realizaremos un histograma de esta para ver en qué proporción se encuentran los distintos colores dentro de ésta. De esta manera obtendremos un vector de valores dentro del cual buscaremos cual es el valor máximo.

Con el valor máximo localizado establecemos, hablando en términos de la representación cónica del espacio de color, un segmento circular que comprenda cinco grados a cada lado del color mayoritario de la imagen. Este sector determina el fragmento del cono que contendrá los colores que consideraremos parte de la percepción de color uniforme. Con estos colores crearemos una máscara que nos indique qué colores pertenecen a nuestra selección y miraremos si éste color forma una superficie uniforme que ocupe más de cierta cantidad de la imagen. En nuestro caso elegiremos un 70%.

Figura 11: Partido de fútbol donde se ha extraído la superficie uniforme de color dominante

Como podemos ver en la figura 11, el algoritmo que hemos definido, a pesar de ser tan simple funciona muy bien. Si tenemos en cuenta que el campo de fútbol tiene zonas de hierba con distintos tonos de verde podemos verificar que la elección del espacio de color para desarrollar el algoritmo ha sido acertada.

Además de la cantidad de imagen que ocupa el color uniforme, guardaremos la máscara que nos servirá en apartados posteriores en los que esta superficie uniforme nos ayudará a distinguir qué elementos son jugadores y cuáles no. Ahora pasaremos a analizar el movimiento dentro de la imagen, que será la segunda parte del método para clasificar los deportes en su correspondiente categoría.

3.3.5 Análisis del movimiento

Como se menciona al definir la categoría de deportes, una de las características más destacables de los deportes, además del color del campo de juego, es la distribución del movimiento y su comportamiento a lo largo de varios frames.

Este análisis nos llevará a definir la posición de los jugadores dentro del campo de juego, lo que utilizaremos para diferenciar unos deportes de otros, y a su vez del resto de programas.

Este proceso se intentó realizar utilizando algoritmos centrados en reconocimiento de personas. Sin embargo, estos algoritmos tienen un coste en tiempo muy superior al análisis de movimiento que vamos a analizar, y además fallaban muy a menudo porque necesitan una imagen nítida donde se vea a la persona claramente y con el movimiento de las cámaras y la distancia a la que se graban los partidos en algunos casos perdíamos toda la nitidez necesaria para un reconocimiento claro de las personas presenten en la imagen.

Vectores de movimiento

A diferencia de algoritmos como los nombrados en [1] y en [2], en este trabajo no contamos con imágenes claras del deporte que estamos analizando, ya que en una retransmisión televisiva normalmente se da una visión de una gran parte del campo para que se pueda obtener una imagen general del juego. Esto impide que se puedan utilizar algoritmos de detección de objetos para la búsqueda de jugadores de una manera sencilla. En este caso, recurriremos a analizar los vectores de movimiento de la imagen para delimitar las zonas de la imagen que consideraremos jugadores.

Para empezar obtendremos los vectores de movimiento que forman el flujo óptico entre dos imágenes de un mismo plano. Para obtener el flujo óptico utilizaremos el algoritmo de Farnebäck [5], obteniendo así una matriz de coordenadas del mismo tamaño que la imagen, correspondiendo un punto de la imagen con cada elemento de la matriz. Estas coordenadas que contiene la matriz representan el nuevo punto donde se encuentra el píxel que se corresponde con este elemento dentro de la matriz. Con estos datos podemos formar para cada pixel de la imagen un vector que nos indique cuanto y en qué dirección se ha movido de una imagen a la siguiente.

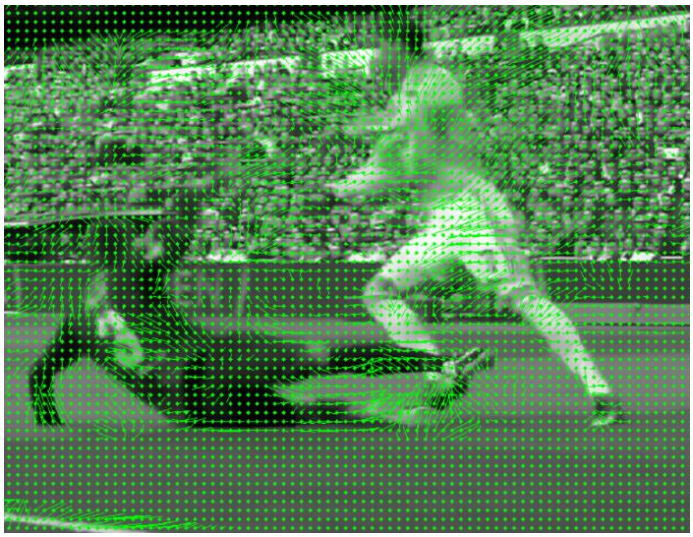

Figura 12: Imagen de un partido de fútbol sobre la que se observan los vectores de movimiento

En la figura 12, se ve una escena donde aparecen dos jugadores en el campo de juego. Como podemos apreciar, el algoritmo de Farnebäck funciona para cada pixel de la imagen, lo que nos deja una gran cantidad de puntos donde se produce movimiento que no necesitaremos. Además, dependiendo de la separación que haya entre las dos imágenes que utilicemos y de la velocidad a la que se produzca el movimiento puede que observemos movimientos fantasmas que empeorarán la calidad de los análisis posteriores.

Filtrado de color

El problema que tiene el algoritmo de Farnebäck es que, al tratarse de un algoritmo de extracción de flujo óptico de carácter denso, no solamente obtiene los vectores de movimiento de los jugadores, sino que obtiene pixel a pixel el movimiento de toda la imagen. Esto provoca que aparezcan movimientos residuales que es necesario filtrar para que no afecten al proceso de análisis posterior.

El filtrado de color que vamos a realizar tiene dos objetivos, eliminar el movimiento del fondo correspondiente a la cámara y establecer la zona en la que podemos detectar los jugadores.

Para realizar el filtrado de color recurriremos de nuevo a la máscara del color predominante, donde tenemos una gran superficie de color uniforme que será el campo de juego. Aplicando la máscara a la matriz de movimiento podremos quedarnos únicamente con las componentes de movimiento que se corresponden con el campo de juego.

Con los vectores de movimiento del campo aislados de los de los jugadores podremos realizar una media aritmética que nos dará como resultado

un vector que representará el movimiento general del campo de juego y que utilizaremos para corregir el movimiento del fondo provocado por la cámara.

El movimiento medio de la imagen lo podemos corregir restando el valor de ese vector a todos los elementos de la matriz de vectores. Así los vectores del campo se verán reducidos en gran medida, mientras que los movimientos de verdad no se verán afectados, ya sea porque su módulo es mayor que el de las componentes del fondo o porque tienen otra dirección en la que la resta del vector de movimiento medio no tiene gran repercusión.

Definición de los rectángulos de los jugadores

Una vez tengamos corregido el movimiento del fondo con respecto a la imagen podemos pasar a analizar las zonas de movimiento y buscar si cumplen las características que hemos comentado anteriormente para cada uno de los deportes.

En primer lugar, aplicaremos un umbral al módulo de los vectores de movimiento de toda la matriz para librarnos de posibles residuos de movimiento en el fondo y de elementos estáticos. Este umbral lo estableceremos en función del valor medio de movimiento que hemos obtenido de las componentes del campo de juego. Esto nos da una matriz binaria donde los ceros representan los pixeles que no superan el umbral mientras que los unos muestran las zonas de la imagen donde el módulo del movimiento es mayor que el del umbral.

Figura 13: Imagen de un partido de fútbol con su correspondiente matriz binaria de movimiento.

Como podemos ver en la figura 13, en la zona del campo no vemos nada de movimiento y la imagen obtenida se parece mucho a la original, donde se observan las siluetas de los jugadores. El problema lo encontramos en las líneas del campo, cuyo movimiento no ha podido ser filtrado con los algoritmos que hemos aplicado y que dificulta la percepción de los jugadores.

Para clasificar el movimiento de la imagen y solucionar el problema de las líneas del campo recurriremos, en primer lugar, a delimitar los objetos en movimiento, por lo que nos quedaremos únicamente con los bordes de cada una de las siluetas que aparecen en la imagen binaria.

Una vez tenemos los bordes de las siluetas tenemos que definir los contornos de cada uno y almacenarlos, para ello realizamos un recorrido de los bordes guardando las coordenadas de cada punto en un vector. Sin embargo, guardar todos los bordes como una secuencia de puntos supone un gasto considerable de memoria y resta sencillez a un posterior análisis de cada uno de los contornos.

Para solucionar esto, representaremos cada contorno como el rectángulo de menor área que contenga todo el contorno. De esta manera representaremos cada rectángulo únicamente como un vector que contiene le punto de su esquina superior derecha y las longitudes de cada lado. Esto nos ahorra espacio en memoria y permite trabajar con los contornos de una manera más intuitiva.

Estos rectángulos que hemos definido, contienen no solamente a los jugadores sino también los elementos de la imagen que no han conseguido ser filtrados, como por ejemplo en la figura 13 algunas líneas del campo.

Para solucionar este problema y quedarnos solamente con los rectángulos de los jugadores tendremos que aplicar otro filtrado a las dimensiones de cada uno de los rectángulos. Si tenemos en cuenta que los rectángulos que nos interesan deben contener a personas, lo primero que debemos buscar es que los rectángulos tengan dimensiones naturales para un ser humano que está corriendo. Esto significa que la relación de aspecto de los rectángulos tiene que encontrarse dentro de unos valores y además que estos valores solo se cumplan para rectángulos verticales.

De esta manera definiremos unos umbrales que eviten que tengamos rectángulos horizontales o que sean demasiado alargados para una persona normal:

- Altura > (1.5 * Anchura)
- Altura < (3 * Anchura)

A pesar de este filtro, puede darse el caso de que el contorno de una línea coincida con las condiciones de relación de aspecto que hemos establecido y se nos cuele en el vector definitivo. Para solucionar este caso vamos a suponer que la línea que detectamos es un objeto de un gran tamaño, ya que su contorno será parte de la delimitación del campo de juego. Teniendo en cuenta esto,

estableceremos unos límites para el área de cada rectángulo excluyendo de la selección los rectángulos más grandes y los más pequeños.

Los límites que pondremos para el área de los cuadrados serán en función de la resolución de la imagen, de manera que se adapten a la cantidad de puntos que esta contenga. Dado que las escenas que estaremos viendo serán cámaras que sacarán gran parte del campo de juego, los jugadores se verán bastante pequeños en comparación con la imagen total, por lo que como límite superior cogeremos la centésima parte de los puntos de la imagen. Además, también puede darse el caso de que aparezcan pequeñas zonas de movimiento que no se hayan eliminado correctamente, así que también definiremos un límite inferior que será de la milésima parte del área de la imagen.

Aplicando estos límites en una imagen con una resolución estándar de televisión digital (720x576) donde tenemos 414720 puntos en toda la imagen, obtendríamos unos rectángulos con áreas comprendidas entre 4147 y 414 puntos.

Unificación de las partes del cuerpo

Cuando los deportistas se encuentran en movimiento, es normal que unas partes de su cuerpo, como pueden ser las piernas o los brazos, tengan una velocidad distinta al resto del cuerpo del jugador. Esto puede producir distintas zonas de movimiento que el algoritmo considerará independientes y que tendremos que revisar para unir todas las partes de un mismo jugador en un solo rectángulo.

Gracias a la representación mediante rectángulos, tenemos zonas contenidas dentro del rectángulo que no coincidirán exactamente con el objeto en movimiento, sino que el rectángulo abarcará una zona ligeramente mayor. Este exceso proporcionado por el rectángulo nos ayuda para considerar partes de un mismo cuerpo aquellas cuyo rectángulo se superponga con otro.

Para estudiar la superposición de los rectángulos utilizaremos las coordenadas de los cuatro vértices de cada uno y observaremos si alguno de los puntos se encuentra dentro de los cuatro vértices del otro rectángulo. Si se produce una superposición, desecharemos los dos rectángulos y nos quedaremos con aquel de menor área que contenga a los dos.

Aplicando estos métodos en el orden que se han explicado sobre la imagen de la figura 13 obtenemos como resultado el conjunto de rectángulos que vemos representado en la figura 14. Se puede observar como algunos jugadores no se detectan, pero eso se debe a que no han sufrido un cambio muy relevante de posición entre las dos imágenes utilizadas para calcular el flujo óptico. Sin embargo el resultado obtenido es bastante prometedor en comparación a lo observado en las siluetas de la figura 13.

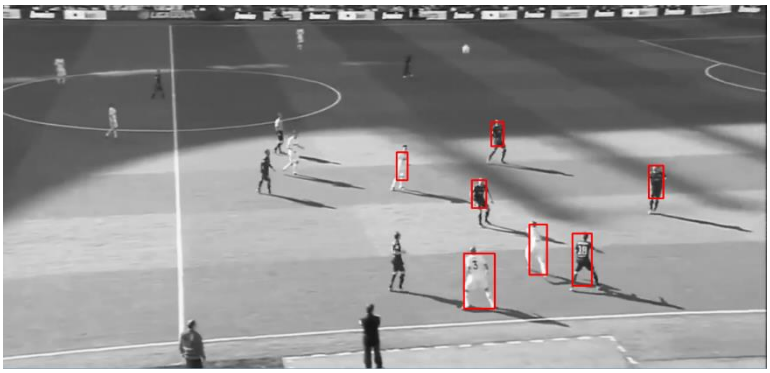

Figura 14: Representación de los rectángulos de los jugadores

3.4 Elección del decisor

Una vez hemos obtenido todas las características del vídeo, tenemos que hacer que éstas pasen por un decisor que se encargará de analizarlas y elegir a qué categoría pertenece el vídeo. Dentro de la visión por computador existen varios sistemas que se utilizan en el reconocimiento de patrones. En un principio se planteó utilizar una máquina de vectores de soporte para implementar el decisor, pero se acabó desechando la idea por una más sencilla y que se adaptaba mejor a los algoritmos y características que formaban parte del análisis.

3.4.1 Máquina de vectores de soporte (SVM)

El decisor basado en máquina de vectores de soporte [9] consiste en crear un espacio multidimensional en el que cada característica de la imagen se ve representada como una dimensión de este espacio. Una vez entrenado el decisor se pueden observar los datos distribuidos dentro de todo el espacio que hayamos definido y el algoritmo se encarga de establecer los llamados hiperplanos, los cuales representan las fronteras óptimas para distinguir un tipo de dato de otro a lo largo de todas las dimensiones del espacio.

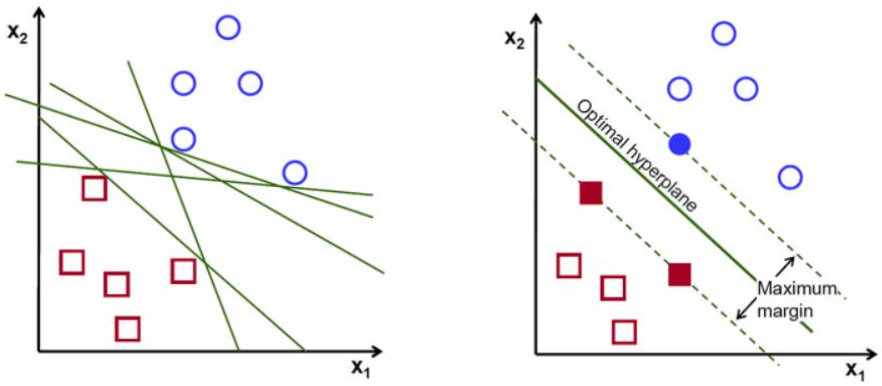

Figura 15: Representación básica de la definición de los hiperplanos óptimos en un espacio de características bidimensional

En la figura 15 podemos ver una representación bastante simplificada del funcionamiento del SVM, donde se crean dos espacios de muestras correspondientes cada una a una categoría del decisor. Cuando se tienen todas las características dentro del decisor, éste calcula el plano óptimo que se encuentra a la distancia máxima de las dos zonas de puntos. Ese plano servirá para decidir cuándo una muestra analizada pertenece a una categoría o a otra dependiendo de a qué lado se encuentre del plano.

Esta es la primera solución que se intentó implementar para el decisor, sin embargo surgieron varios problemas en el proceso. En primer lugar, para el correcto funcionamiento de este sistema es necesario entrenarlo con una gran cantidad de datos en la etapa previa a su uso. Sin embargo, para este trabajo contamos únicamente con el contenido que cada emisora de televisión pone a disposición del público y no con un espacio de muestras de entrenamiento.

El segundo problema al que nos enfrentamos es la necesidad de cuantificar todas las características que obtenemos de los vídeos para que puedan representarse como una coordenada del espacio del SVM. Este proceso no supondría mucha complicación, pero tras analizar las características que vamos a extraer de los vídeos se ha visto que la mejor forma de analizarlas es con una decisión binaria que indique si esa característica se encuentra o no en el vídeo.

Teniendo en cuenta estos inconvenientes, se ha decidido diseñar el decisor de manera que sea un árbol de decisión donde se vayan analizando las distintas características en orden y nos vayamos decantando por una o por otra solución.

La opción por la que se ha acabado optando es un árbol de decisiones que nos permita aplicar de manera secuencial los distintos elementos del análisis. Así podremos ir analizando desde características más generales hasta más específicas según se vaya bifurcando cada una de las ramas. Además, si en un futuro se intentan analizar más características para encontrar más categorías de programas, solamente es necesario encontrar a qué rama pertenece y continuar analizando a partir de ahí.

3.4.2 Decisor binario

Un árbol de decisión binario es una estructura de decisor que nos permite ir evaluando cada una de las características en orden y nos llevará a una solución o a otra en función de los resultados de las distintas condiciones. En la figura 16 vemos un esquema básico de un árbol de decisiones binario en el que se observa cómo se puede avanzar por una rama u otra en función de las condiciones que utilicemos.

La ventaja de este tipo de sistema es que tiene una escalabilidad muy sencilla, ya que si queremos añadir alguna condición más en alguno de los puntos solamente tenemos que continuar a partir del extremo de la rama correspondiente. Además, al ser un algoritmo que aplica las condiciones de manera secuencial, podemos ordenarlas de tal manera que utilicemos las

condiciones que requieran algoritmos más ligeros computacionalmente al principio y de esta manera que exista la posibilidad de evitar utilizar algoritmos más costosos si no se da el caso siguiendo la secuencia de decisiones correspondiente.

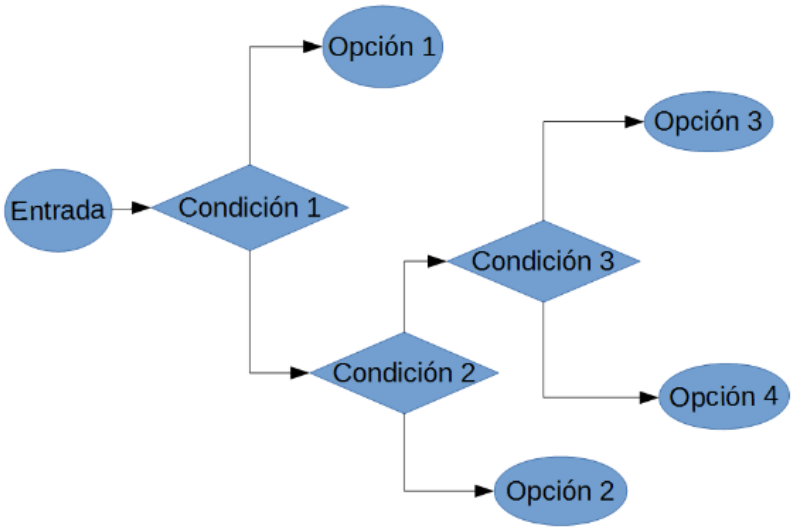

Figura 16: Esquema de un árbol de decisión binario

3.4.3 Árbol de decisiones definitivo

En el árbol de decisiones que hemos creado para este trabajo tenemos cinco clases a las que se puede llegar en función de las características del vídeo. Estas clases son Informativos, Fútbol, Baloncesto, Tenis y Otros. En este orden en el que las hemos definido será además el orden en el que se evaluarán las características.

Como hemos explicado al comienzo de esta sección, los algoritmos que hemos utilizado han sido adaptados de tal manera que funcionen bien con este tipo de decisor. Esto es, el análisis de estructura acabará devolviendo como resultado si el video analizado es de la categoría informativos o no y así con el resto de los análisis.

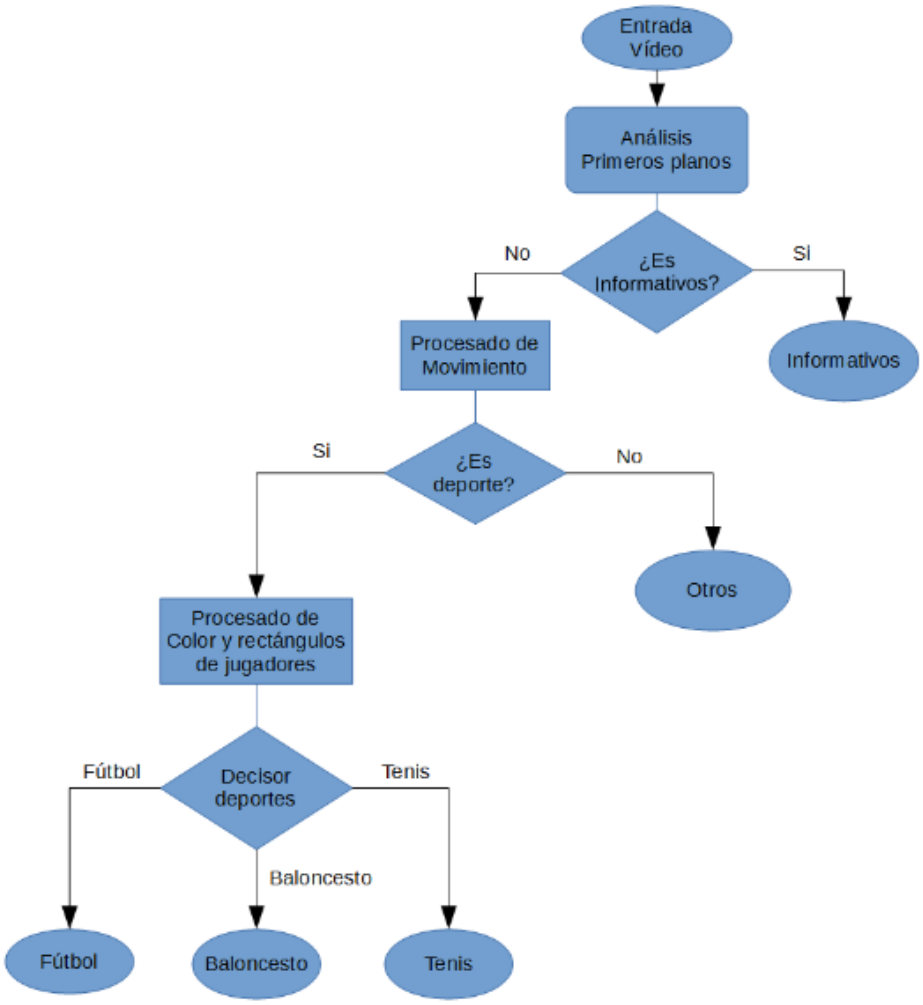

Figura 17: Diagrama del decisor binario implementado en la aplicación

En la figura 17 podemos ver el árbol de decisiones definitivo que se utilizará en la aplicación. En él vemos los distintos pasos que seguirá la aplicación para llegar desde que se ejecute y se le introduzca un vídeo hasta que decida una de las cinco categorías posibles.

En primer lugar el programa realizará el análisis de estructura ya comentado, evaluando los primeros planos presentes en el vídeo hasta tomar una decisión. Si el decisor considera que el programa es un informativo no realizará ninguna operación más. Si el resultado es negativo, éste pasará a analizar el siguiente piso dentro del árbol.

El análisis de movimiento que utilizaremos para clasificar el deporte, a diferencia del de estructura aplicado anteriormente, no analiza el vídeo completo, sino que estudia cada escena por separado. Esto hace que dentro de ese análisis muchas de las escenas con las que se trabaje no serán de ayuda en la clasificación. Para solucionar esto diseñamos un sistema de votación que nos permita analizar un gran número de escenas y clasificarlas en la categoría a la

que correspondan. Una vez se hayan analizado todas las escenas que elijamos, haremos un recuento de cuántas pertenecen a cada categoría y así decidiremos cual es la categoría más probable en la que clasificaremos el vídeo.

Además, si tenemos en cuenta que queremos que el algoritmo sea rápido, no podemos hacer que el sistema de votación analice todas las escenas del vídeo. Por ello definiremos unas zonas de búsqueda que nos permitan desechar frames donde es poco probable que encontremos escenas como las que nos interesan para cada deporte. Estableceremos tres zonas de búsqueda basadas en cómo se estructuran los deportes y la duración de las jugadas típicas:

- En fútbol las jugadas no tiene límite de tiempo, pero suelen durar bastante, por lo que uno de los bloques de análisis contendrá los planos más largos del vídeo.
- En baloncesto la duración máxima de una posesión es de 24 segundos. Si tenemos en cuenta que dentro de un punto la posesión puede cambiar varias veces de equipo tendremos que establecer el segundo bloque de análisis en duraciones un poco superiores a 24 segundos. En nuestro caso elegiremos escenas cuya duración esté próxima a los 30 frames.
- En tenis los puntos no suelen durar tanto como en los otros deportes, y tras un pequeño estudio de varios partidos de tenis, se decide que el tercer bloque de análisis esté en torno a los 10 segundos.

Con estas tres zonas de búsqueda delimitadas existe una mayor probabilidad de que las escenas que analicemos cuenten con las características que necesitamos extraer para la clasificación de deportes.

4. Estructura del programa

En este apartado comentaremos las distintas clases que se han utilizado dentro del programa, ya sean propias de openCV o diseñadas por nosotros. También se explicará que relación guardan unas con otras y cómo interactúan dentro del programa.

4.1 Clases importantes

Para el programa hemos utilizado clases propias de openCV para trabajar con el fichero de vídeo e interpretar las imágenes como matrices para facilitar su interpretación y procesado. También se han utilizado clases básicas del API de C++ para trabajar con ficheros y poder leer y escribir en ellos de forma cómoda. Por último, se ha creado una clase propia que contiene todos los métodos que se utilizan en la aplicación así como los datos necesarios para que funcione.

VideoCapture

La clase VideoCapture es una clase que se encuentra en las librerías de openCV y que nos permite acceder a todos los datos de un fichero de vídeo codificado en H264 como se encuentran los vídeos que hemos analizado en este proyecto.

Esta clase está pensada para trabajar de manera sencilla con la clase Mat de openCV y cuenta con operadores sobrecargados que permiten volcar datos de manera fácil desde el fichero hasta la clase Mat dentro del programa.

La clase VideoCapture nos permite también acceder a los metadatos del fichero de vídeo para que sea fácil obtener la longitud en frames del fichero o la resolución en la que se encuentra. También nos permite, utilizando únicamente una función, acceder a un frame en particular como si estuviésemos moviéndonos a través de un array.

Mat

Esta clase de openCV es la más importante para nosotros, ya que es sobre la que vamos a trabajar en todos los apartados del programa. Esta clase implementa una matriz de las dimensiones que nosotros deseemos y nos permite realizar operaciones matriciales sencillas que se encuentran ya definidas sobrecargando los operadores comunes.

En el caso de las imágenes que extraeremos del vídeo, la clase VideoCapture las guarda como una matriz de enteros con tres canales. El formato de tres canales consiste en aprovechar que los enteros son de 32 bits en el compilador de Visual Studio 2013 para representar las tres componentes de color con un solo entero. Cada componente de color utiliza 8 bits para representar el valor que tiene en cada pixel, por lo que el formato de tres canales junta las componentes de color en bloques de 8 bits dentro de un entero. Esto hace que a la hora de acceder a las distintas componentes del espacio de color tengamos que trabajar a nivel de bit sobre el valor de cada pixel.

Ifstream y Ofstream

Las clases Ifstream (Input stream) y Ofstream (Output stream) son clases propias de C++ que nos permiten trabajar con cualquier tipo de fichero, ya que lo interpreta como un stream de datos. Estas clases nos permiten definir el formato que tendrá el fichero de datos que utilizaremos para guardar datos que necesitemos utilizar posteriormente y no se puedan guardar como variables del propio programa. Este es el caso de los ficheros donde guardaremos los datos de los cambios de plano de cada vídeo para no tener que repetir el proceso de extracción cada vez que utilicemos el programa.

DecisorProg

Por último, esta clase es una clase definida por nosotros que contiene todas las variables necesarias para la ejecución del programa, así como todos los métodos desarrollados para poder acceder a ellos de forma fácil pero que se pueda exportar a un programa más grande y se conserve la estructura definida para el decisor.

Esta clase contiene todo lo necesario para que, desde un programa externo se tenga que ejecutar únicamente un método para que se realice el procesado del vídeo hasta la obtención del resultado final del decisor.

Ahora vamos a analizar cómo se relacionan unas clases con otras dentro del programa:

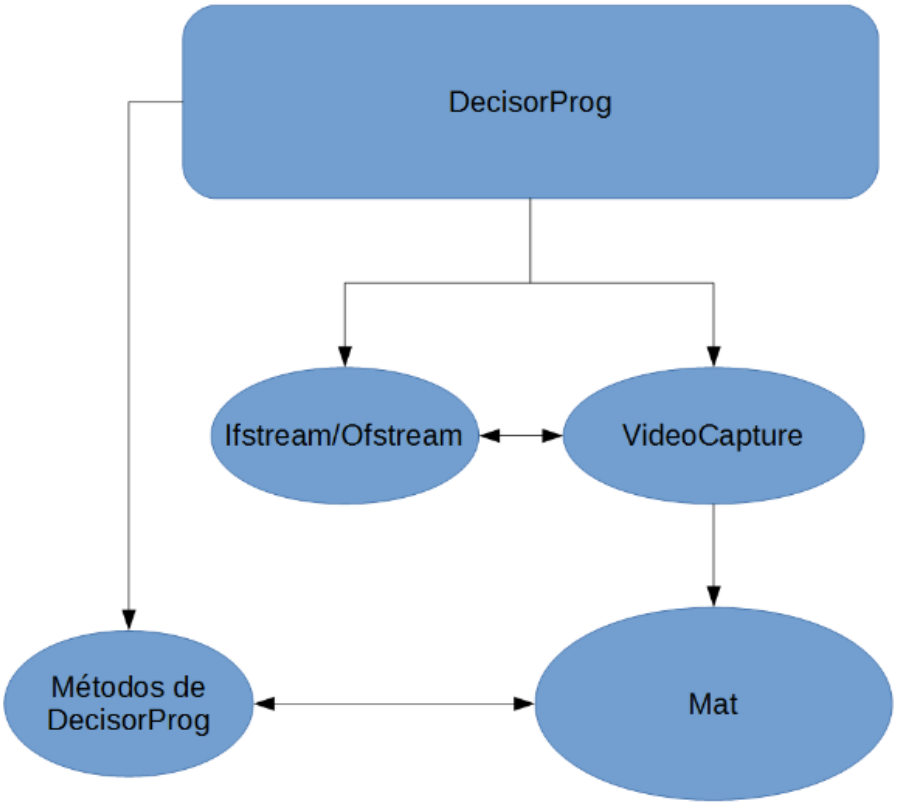

Figura 18: Relación entre las clases del programa

Como se observa en la figura 18, la clase DecisorProg que hemos creado nosotros se encarga de controlar todas las demás clases, ya sea directamente o mediante los métodos de procesado pertenecientes a esta clase.

Para empezar, la clase DecisorProg interactúa con la clase Ifstream para para leer el fichero ASCII en el que se encuentran los datos de los cambios de plano y acceder, a través de la clase VideoCapture, a los datos del vídeo que se necesiten utilizar en cada momento.

Utilizando métodos propios de la clase VideoCapture obtenemos la información de cada frame del vídeo y la guardamos como una matriz de la clase Mat. Cuando tenemos la matriz con los datos de la imagen ya podemos trabajar de manera sencilla sobre ella, así que podremos aplicar los métodos de la clase DecisorProg para modificar o extraer datos de la matriz.

Este es el proceso para extraer la información del vídeo, que según la vayamos obteniendo se irá almacenando en la clase DecisorProg que de manera interna se encargará de aplicar los algoritmos explicados en la anterior sección para clasificar el vídeo según dicta el objetivo de este TFG.

5. Resultados experimentales

5.1 Montaje experimental

Para probar el programa contamos con un conjunto de 100 videos obtenidos de las páginas de las televisiones a la carta donde se incluyen videos a disposición del público. Dentro de estos 100 videos contamos con programas de todas las categorías que hemos definido para el trabajo. En la categoría otros contamos con la mayoría de los videos, ya que necesitamos probar programas de todas las posibles categorías futuras y asegurarnos de que no dan fallos y se detectan como informativos o deportes. De la categoría de informativos contamos con 10 vídeos de distintas cadenas de manera que podamos asegurarnos de que el algoritmo se podrá utilizar independientemente del decorado con el que cuente cada emisora en el plató de televisión. Por último, en la categoría de deportes contamos con 25 videos que se distribuyen en fútbol, baloncesto y tenis en los tres tipos de campo posibles, hierba, tierra y cemento.

La máquina con la que contaremos para realizar las pruebas será un ordenador portátil Mountain Graphite con un procesador Intel core i7-4810MQ a 2,8 GHz y una tarjeta gráfica Nvidia GTX 970m. Dado que se trata de un equipo portátil las prestaciones no podrán compararse con las de un equipo dedicado específicamente para realizar esta tarea, sin embargo estaremos ante unos resultados bastante similares a los que se conseguiría con un PC de sobremesa normal.

5.2 Resultados

Tras probar el software con la mayoría de los videos con los que contamos, hemos observado que éste cuenta con una tasa de éxito superior al 80%. Los errores que aparecen se deben sin embargo a programas demasiado cortos como para que el sistema de votación reúna datos suficientes para la decisión. Esto podría arreglarse si estableciésemos un mínimo de cambios de plano necesario para realizar el recuento de votos.

A pesar de los errores, si analizamos las distintas categorías por separado se pueden observar estos resultados prometedores. Dentro de la categoría de informativos se han incluido todos los programas con los que contamos sin ningún falso positivo de algún programa de otra categoría. Los deportes también se han clasificado de manera correcta y, donde se han observado fallos es en la categoría otros, donde algunos de los videos los ha incluido dentro de los deportes. Sin embargo estos videos son únicamente 11 de los 69 con los que contamos en esta categoría.

Otro factor que hemos observado realizando las pruebas es que la clasificación de los vídeos se realiza en unos tiempos que, en el peor de los casos se encuentran alrededor de los 3 minutos. Teniendo en cuenta que este tiempo de análisis se produce para vídeos con una duración que ronda las 2

horas se puede considerar que el tiempo de procesado es muy reducido. Suponiendo que se ejecutase este algoritmo de manera continua en una máquina, tendríamos una tasa de 480 vídeos clasificados cada 24 horas.

Tabla 1: Representación de los resultados finales tras utilizar la aplicación sobre los programas con los que contamos.

Categoría	Elementos analizados	Elementos correctos	Elementos erróneos	Tiempo de ejecución (minutos)
Informativos	10	10	0	1
Deportes	21	19	2	3
Fútbol	7	7	0	3
Baloncesto	7	7	0	3
Tenis	7	5	2	3
Otros	69	58	11	2
Total	100	87	13	

En la tabla 1 podemos ver los resultados obtenidos para cada una de las categorías con el número de videos de los que se disponía para hacer las pruebas. Como hemos nombrado antes, podemos observar unos cuantos errores en la categoría de otros que se deben a que los programas son demasiado cortos, por lo que el sistema de votación que se encarga de distinguir los deportes del resto de programas no tiene suficientes muestras para tomar una decisión fiable.

En la categoría de deportes podemos observar dos errores en la subclase de tenis. Estos errores se han producido en dos partidos de tenis sobre tierra batida en los que el campo de juego tiene un color muy similar al de un campo de baloncesto y, al ser la retransmisión de un campeonato hay más personal por el campo aparte de los jugadores. Esto provoca que si los puntos son muy cortos, el movimiento de los recogepelotas y los cambios de escena entre punto y punto predominen sobre las jugadas en sí. Este problema podría solucionarse evaluando con mayor o menor peso cada una de las distintas categorías. Esta solución se está estudiando y podría implementarse en futuras versiones de la aplicación.

Otra opción que podría estudiarse para solucionar estos errores en la clasificación sería buscar nuevas características dentro del vídeo que se puedan analizar, sin embargo, viendo la eficacia de los métodos desarrollados hasta ahora no serían una alternativa a los algoritmos sino un añadido para hacer más robusto el sistema de decisión.

6. Conclusiones y líneas futuras de trabajo

El objetivo principal del trabajo era el desarrollo de una aplicación que se encargase de analizar y clasificar de manera automática vídeos procedentes de retransmisiones televisivas, la cual como entrada recibiría el vídeo correspondiente de cualquier tema posible y al final obtendríamos como respuesta una de las categorías definidas previamente.

A lo largo del transcurso de este trabajo hemos analizado e implementado distintos algoritmos y métodos que se pueden utilizar para la extracción de características de un vídeo, centrando nuestro análisis en analizar el flujo de vídeo y no únicamente imágenes aisladas como hacen la mayoría de los algoritmos actuales.

Para la realización del trabajo hemos seguido una estructura típica para el análisis de vídeo que consiste en analizar, en primer lugar, el contenido que con el que vamos a trabajar, diferenciando cada uno en distintas categorías y definiendo las principales características con las que cuentan. Después se han estudiado distintos algoritmos típicos de procesado de vídeo para analizar su posible aplicación dentro de este trabajo y como pueden combinarse para extraer las características de cada programa estudiado. Por último, hemos implementado un decisor que se adapte a los datos recibidos de la extracción de características.

Dado el tiempo reducido con el que se ha contado para realizar el proyecto y teniendo en cuenta la novedad del campo en el que se ha trabajado, no se ha conseguido cumplir con el alcance completo del trabajo. Sin embargo, se ha desarrollado una aplicación que consigue clasificar los vídeos de entrada con una tasa de éxito más que aceptable y que además consigue realizar esta clasificación en intervalos de tiempo muy reducidos.

A la vista de estos resultados podemos definir como líneas futuras el estudio completar el alcance que se fijó en principio para clasificar todos los programas en sus categoría correspondiente, además de buscar características en los programas que ya clasificamos para dotar de más robustez a la aplicación.

A pesar de no haber conseguido completar los objetivos propuestos al principio del proyecto, podemos considerar los resultados obtenidos muy prometedores, ya que las categorías que sí hemos podido clasificar consiguen una tasa de éxito muy alta en poco tiempo de análisis, que es lo que en principio teníamos marcado como objetivo.

Referencias

[1] Li-Jia Li and Li Fei-Fei. "What, where and who? Classifying events by scene and object recognition". *DOI: 10.1109/ICCV.2007.4408872 ICCV, 2007. pp. 1 – 8.*

[2] P. Panakarn, S. Phimoltares and C. Lursinsap. "Complex sport image classification using spatial color and posture context descriptors and neural classifiers". *DOI: 10.1109/ICMLC.2010.5580565 Conference: Machine Learning and Cybernetics (ICMLC), 2010 International Conference on, Volume: 2, Pages: 713 – 718.*

[3] Shenghua Gao, Liang-Tien Chia and Ivor Wai-Hung Tsang. "Multi-layer Group Sparse Coding for Concurrent Image Classification and Annotation". *DOI: 10.1109/CVPR.2011.5995454 IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2011. pp. 2809 – 2816.*

[4] Yasuo Ariki and Yoshiaki Sugiyama. "Classification of TV sports news by DCT Features using multiple subspace method". *DOI: 10.1109/ICPR.1998.711988 IEEE Internacional Conference on Pattern Recognition, vol. 2, pp. 1488 – 1491, 1998.*

[5] Gunnar Farnebäck. "Two-frame motion estimation based on polynomial expansion". *In Proceedings of the 13th Scandinavian Conference on Image Analysis, LNCS 2749. Gothenburg, Sweden, 2003; 363-370.*

[6] Ahmet Ekin and A. Murat Tekalp. "Robust dominant color region detection and color-based applications for sports video". *DOI: 10.1109/ICIP.2003.1246888 ICIP 2003, vol. 1, pp. I – 21 – 4.*

[7] HSV (HUE, saturation, value), "http://www.tech-faq.com/hsv.html", consultado 30/3/2015

[8] The HSV Colorspace, "http://ie.technion.ac.il/CC/Gimp/node51.html", consultado 30/3/2015

[9] Introduction to Support Vector Machines, "http://docs.opencv.org/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html", consultado 3/3/2015.

Bibliografía

[1] Georgios Th. Papadopoulos, Alexia Briassouli, Valiseios Mezaris, Ioannis Kompatsiaris and Michael G. Strintzis. "Statistical Motion Information Extraction and representation for semantic video analysis". *DOI: 10.1109/TCSVT.2009.2026932 IEEE Trans. Circuits and Systems for Video Technology 2009, vol. 19, pp. 1513 - 1528.*

[2] Rostom Kachouri, Khalifa Dhemal, Hichem Maaref, Dorra Sellami Masmoudi and Nabil Derbel. "Content description and classification for image recognition system". *DOI: 10.1109/ICTTA.2008.4529998 Information and Communication Technologies: From Theory to Applications, 2008. ICTTA 2008. 3rd International Conference on*, *Year: 2008, Pages: 1 – 4*

[3] Yuanbo Chen, Zhixuan Li, Xin Guo, Yanyun Zhao,"A Spatio-Temporal interest point detector based on vorticity for action recognition" Anni Cai1,2 DOI: 10.1109/ICMEW.2013.6618448, *Multimedia and Expo Workshops (ICMEW), 2013 IEEE International Conference on, Year: 2013, Pages: 1 - 6*

[4] OpenCV 2.4.11.0 documentation, "http://docs.opencv.org/index.html", consultado 1/3/2015