Universidad Zaragoza

Proyecto Fin de Carrera

DESCRIPTORES GLOBALES PARA VÍDEO

Alejo Concha Belenguer

Director: Javier Civera Sancho

Ingeniería Industrial
Automatización Industrial y Robótica

Departamento de Informática e Ingeniería de Sistemas
Escuela Universitaria de Ingeniería y Arquitectura
Universidad de Zaragoza

Julio 2012

Descriptores globales para vídeo

RESUMEN

En internet se almacenan grandes cantidades de imágenes y vídeos. Las dos principales plataformas web que tratan con este tipo de información son Google y YouTube respectivamente. El número de imágenes y vídeos que poseen es tan grande que se requieren técnicas para clasificarlos según su contenido, es decir se necesitan técnicas para predecir, ante un nuevo vídeo o imagen, a que categoría pertenece. En imágenes el problema ha sido muy estudiado. En vídeos ha habido mucho menos trabajo. Los vídeos necesitan ser clasificados en categorías como por ejemplo política, deportes, música, etc. Es importante que esta clasificación sea de acuerdo con su contenido. YouTube actualmente clasifica sus vídeos según las descripciones que el responsable que los ha subido ha escrito para identificarlo. De esta manera hay muchos vídeos clasificados erróneamente. La clasificación de vídeos también puede ser interesante en reconocimiento de personas u objetos (como carteles escritos) en ambientes complicados grabados por cámaras corrientes o cámaras de vigilancia. También pueden aportar aplicaciones para dispositivos móviles.

Los aspectos más relevantes para la clasificación de vídeos son los descriptores y las máquinas de aprendizaje. Los descriptores de vídeo se encargan de describir el vídeo según su contenido. Las máquinas de aprendizaje toman estas descripciones de cada vídeo para aprender a que tipo de vídeo pertenece cada descripción y así ante la aparición de un vídeo sin clasificar poder determinar a que categoría pertenece. En este proyecto final de carrera se han usado ambos aspectos y principalmente se han estudiado los descriptores de vídeos. Los descriptores que hay en la actualidad son de dos tipos, globales y locales. Los globales describen el vídeo de forma global y los locales describen sólo zonas salientes del vídeo. Estas zonas salientes del vídeo son localizadas mediante detectores. Se ha propuesto un descriptor global para vídeos y un detector de zonas salientes para describirlas localmente. El descriptor global esta basado en el cálculo de gradientes en las tres dimensiones del vídeo. El detector local aplicado es la extensión a 3 dimensiones del detector SIFT 2D que es el mayor usado en imágenes. Ambas propuestas se han implementado en Matlab y se han evaluado de manera extensiva en bases de datos públicas y con implementaciones actuales.

Además serán utilizadas estas bases de datos de vídeos para realizar un afinamiento de los parámetros de los descriptores y deducir que parámetros son los mejores. Se estudiará el porqué de los resultados aparecidos. Los resultados que se han obtenido han mejorado los resultados en bases de datos de vídeos extensos y complejos. En el caso del detector local propuesto, no se ha apreciado mejora con respecto a los detectores propuestos en la literatura actual.

Índice

1. Introducción

1.1. Clasificación automática de vídeos

La visión por computador es la disciplina cuyo objetivo se puede expresar intuitivamente como "que las máquinas puedan ver". Pese a la simplicidad de la frase, la visión artificial ó automática es una tarea compleja y lejos de estar resuelta en el caso general. La visión artificial es una disciplina muy fragmentada que abarca campos como la visión 3D, el reconocimiento de objetos y de escenas o el seguimiento de objetos móviles. De entre ellos, uno de los aspectos que puede resultar de gran interés en un futuro próximo es el reconocimiento automático de vídeos por su contenido.

Para entender por qué este problema de clasificación es interesante podemos fijarnos en un ejemplo con imágenes. Google dispone de un eficiente algoritmo de reconocimiento de patrones en imágenes gracias al cual aporta dos aplicaciones muy conocidas en cuanto a uso aunque no en cuanto a los algoritmos en los que se basan:

- Google imágenes: Es la versión en imágenes del buscador Google. Dada una palabra o frase introducida en la barra de búsqueda, Google imágenes devuelve las imágenes relacionadas con esa búsqueda. Se puede apreciar su funcionamiento en la figura 1 tras la búsqueda de la palabra 'iglesia'.
  Anteriormente, al introducir una búsqueda en Google imágenes, Google buscaba imágenes donde la palabra buscada se encontrara en un texto cercano o en su mismo título. Actualmente Google ha mejorado este algoritmo, incorpora un sistema de reconocimiento de imágenes para que además de buscar las imágenes con la palabra correspondiente busca también imágenes por reconocimiento. De esta manera se evitan muchos falsos positivos, es decir imágenes que contienen la palabra buscada pero no se corresponden en contenido con lo que se está buscando.

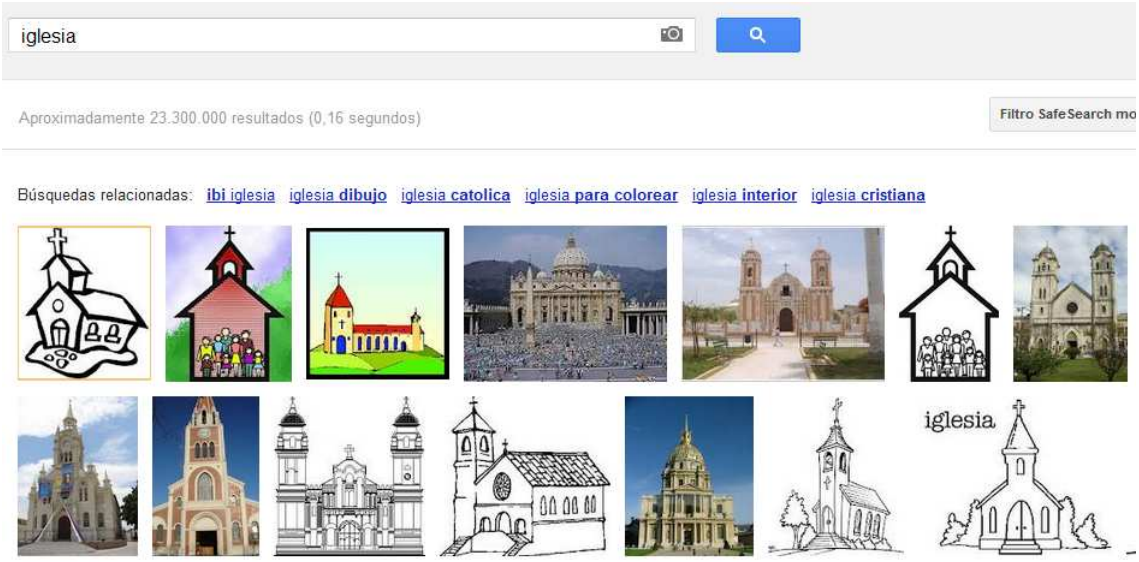

Figura 1. Búsqueda de 'iglesia' en Google, se buscan imágenes relacionadas con la palabra iglesia y además fotos reconocidas como una iglesia gracias al algoritmo utilizado por Google.

- Google goggles: Aplicación para móviles de Google. Consiste en el reconocimiento de imágenes al realizar una foto. Se realiza una foto a una iglesia y Google goggles es capaz de reconocerla gracias al mencionado algoritmo de reconocimiento, decirnos que iglesia es, donde se encuentra y la información que necesitemos. Nótese que a diferencia del caso

anterior con Google imágenes la clasificación es realizada únicamente con información de la propia imagen, puesto que es la única información que tenemos. Si hacemos una fotografía a la carátula de un libro o un videojuego también es capaz de reconocerlo y de aconsejarnos una web para adquirirlo. En la figura 2 se muestra un ejemplo en el cual se ha fotografiado un cuadro famoso y Google googles lo identifica y proporciona información sobre él vía web.

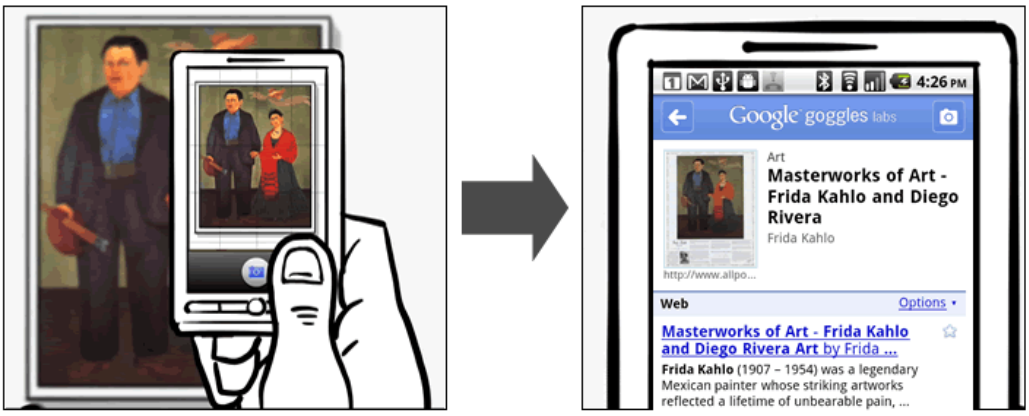

Figura 2. Imagen ejemplo de Google donde se toma una foto de un cuadro y Google consigue clasificarlo y darnos su información vía internet.

Estas dos aplicaciones que están siendo utilizadas por Google sería interesante que pudieran ser extendidas a vídeos. YouTube es un sistema de almacenamiento de vídeos que a día de hoy utiliza un sistema de busqueda solamente relacionado con la palabra/frase introducida y la correspondencia de esta con las 'tags' o palabras clave que contiene el vídeo para ser encontrado, como ejemplo ver figura 3.

Figura 3. Búsqueda de la palabra ciclismo en YouTube.

A diferencia de Google, YouTube no reconoce los vídeos, sólo busca según la palabra introducida y su correspondencia con las palabras clave que el responsable del vídeo ha utilizado antes de subirlo a la web. Debido a esto youtube devuelve muchos falsos positivos en

la busqueda introducida, en la misma figura 3 se aprecia como los dos primeros vídeos que devuelve YouTube no están relacionados con la palabra buscada.

Con el uso de un algoritmo de reconocimiento de vídeos, YouTube sería capaz de reconocerlos de una manera más eficaz. Extrapolando la aplicación de Google goggles, una posible aplicación podría ser que al realizar un vídeo de una película o algo previamente grabado como podría ser un acto histórico, el sistema de reconocimiento sería capaz de reconocerlo y decirnos de que vídeo se trata. En vídeos tendremos más información que en imágenes ya que en un vídeo disponemos de muchas imágenes, pero en contrapatida la tarea de clasificación es más compleja ya que es más dificil y costoso procesar toda esta información.

1.2 Objetivos y resumen

Una de las partes más importantes de los sistemas de reconocimiento son los descriptores. Los descriptores deben de describir el vídeo de la mejor forma posible de manera que extraigan características típicas de cada tipo de vídeo y que a su vez estas caraterísticas sirvan para distinguirlo de los demás. Por ejemplo, en imágenes, si describimos una imagen de forma que encontremos dos ventanas, una puerta y un tejado el sistema de reconocimiento debería de saber reconocer esa imagen como la imagen de un casa. En concreto en este proyecto final de carrera se van a estudiar descriptores globales y descriptores locales. Ambos descriptores utilizan un algoritmo similar de descripción, la principal diferencia es que los descriptores locales aplican este descriptor a puntos o zonas característicos del vídeo mientras que los globales lo aplican a todo el volumen del vídeo. De esta manera otra diferencia entre ambos es que los descriptores locales necesitan de la localización de los puntos característicos o salientes que se realizará con un detector.

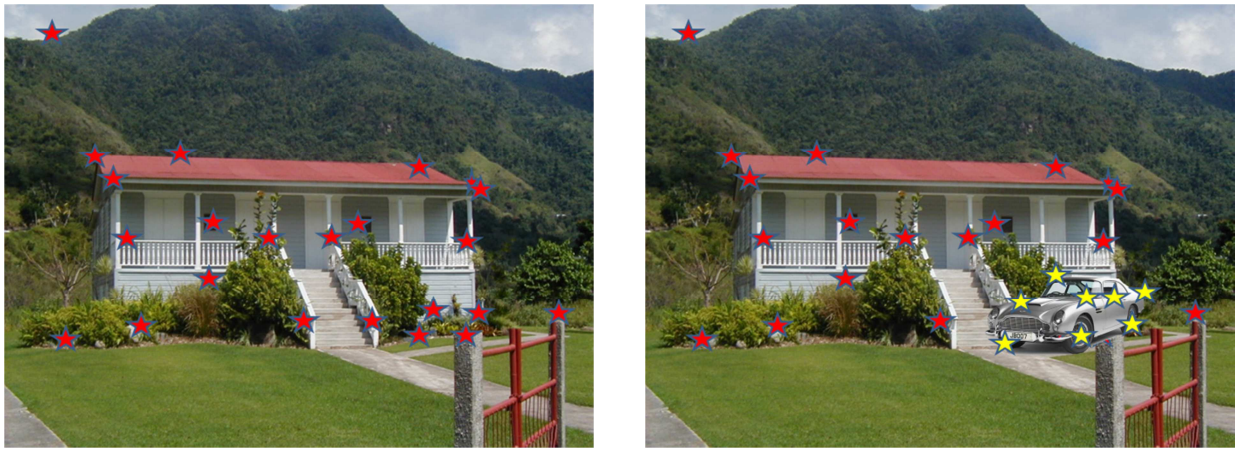

Figura 4. El descriptor local describe las zonas alrededor de puntos característicos, la entrada del coche en la imagen hace que se pierdan algunos puntos clave pero no todos

Gracias a ambos descriptores (descriptor global por un lado y detector y descriptor local por el otro) ya tenemos el vídeo descrito, con esta descripción hay que enseñarle a una máquina de aprendizaje de la misma manera de la que aprendemos los seres humanos, por repetición. Se le envía a esta máquina una serie de descripciones de vídeos y la clase de vídeo a la que pertenecen. Con esta información es capaz de aprender que tipo de descripciones en cada clase de vídeo se relacionan con cada clase. Ante la entrada de un nuevo vídeo, buscará entre las clases de vídeos de los que ya ha aprendido previamente cual se parece más y así conseguir su reconocimiento.

En este proyecto se propone un descriptor global no utilizado antes y un detector de puntos característicos local tampoco utilizado antes. Se define descriptor como el cálculo de gradientes de color en la imagen / vídeo. Un descriptor global calcularía los gradientes a lo largo y ancho de las imágenes que se ven en la figura 4. Como se puede apreciar estos gradientes saldrían diferentes para cada imagen y por tanto el descriptor cambiaría, ya que sólo lo aplicamos una vez y a toda la imagen. En cambio para el caso de un detector y descriptor local se calculan estos gradientes sólo en una región alrededor del punto clave determinado por el detector. Por esta razón el descriptor local es más robusto ya que muchos de los puntos clave se mantienen al introducir otro objeto en la imagen (el coche en este caso) aunque aparecen otros nuevos debidos al nuevo objeto.

El **objetivo** de este proyecto de fin de carrera es la **evaluación de descriptores globales y locales para clasificación automática de vídeos**. El **trabajo desarrollado** se puede resumir en:

1. Propuesta de un **descriptor global de vídeos** (apartado 2).
2. Propuesta de un **detector local para vídeos** (apartado 3).
3. **Implementación** de ambos (detector y descriptor) en Matlab.
4. Selección y uso de **librerías para aprendizaje automático**.
5. **Selección de datasets** para evaluación con el estado del arte en clasificación automática de vídeos.
6. Sintonización de parámetros de los descriptores globales para optimizar los resultados de clasificación.
7. Sintonización de parámetros de los descriptores y detectores locales para optimizar los de resultados clasificación.
8. **Evaluación** de ambos descriptores y comparación con el estado del arte.

Como adelanto de las conclusiones, los descriptores globales funcionan mejor en datasets más amplios, con mucha información y con muchas clases a determinar, además de en datasets donde no solo la acción es necesaria si no también el fondo donde se desarrolla. Dicho esto los descriptores globales pueden ser una buena opción ya que YouTube dispone de una cantidad inmensa de vídeos y de clases de vídeos que necesitarían de su reconocimiento.

El resto del documento se organiza de la siguiente manera: la sección 2 da los detalles del descriptor global propuesto. La sección 3 desarrolla el detector local propuesto. La sección 4 describe la maquina de aprendizaje utilizada (clasificador). Las secciones 5 y 6 comparan los descriptores utilizados. La sección 7 muestra un dataset propio. La sección 8 muestra los experimentos realizados. En la sección 9 se encuentran las conclusiones, la 10 propone el posible trabajo futuro y la sección 11 aporta los anexos.

2. Descriptores globales de vídeo

El descriptor es un vector de números (cada número es una característica) que tiene la función de dar un significado computacional a cada vídeo, deberá de proporcionar descripciones similares para vídeos de una clase y descripciones diferentes para vídeos de clases diferentes. Basamos nuestro descriptor en el histograma de los gradientes orientados utilizado en imágenes (HOG [13,14]) y que aplicaremos tanto en vídeos (apartado 2.1) como en flujo óptico (apartado 2.2). También expondremos otros tipos de descriptor (apartados 2.3 a 2.7).

2.1 Descriptor propuesto. Histogramas globales de gradientes orientados en 3D (G-HOG3D)

Explicamos el concepto en 2D [12,13] en el apartado 2.2.1 y en el apartado 2.2.2 lo extendemos a la tercera dimensión para explicar el algoritmo completo en 2.2.3.

2.1.1 Histogramas de gradientes orientados en 2D

Dada una imagen, se obtiene la matriz que contiene los valores de cada uno de sus píxeles para cada uno de los 3 colores (rojo, verde y azul) a lo largo y ancho de su superficie (matriz de 'n' filas, 'm' columnas y 3 canales para cada calor). Se calculan las derivadas de la textura de la imagen. En matrices tenemos datos discretos, por tanto no podemos derivar explícitamente. Para derivar en discreto se restan los valores de los píxeles que están en la dirección respecto la que queremos derivar. Esto lo conseguimos gracias al filtrado de esta matriz con máscaras derivativas, sustituyen cada píxel por el valor de la derivada buscada. Se puede realizar para cada uno de los tres colores o pasando la imagen a escala de grises y derivando en ella.

Existen varios máscaras para realizar estas derivadas, la que planteamos a continuación está demostrado que es el que mejor funciona según [12,13]. Consiste en calcular los gradientes en 'xi y en 'y' mediante la resta de los píxeles contiguos al píxel y situados en la dirección de la derivada, tanto horizontalmente para los gradientes en 'x', como verticalmente para los gradientes en 'y'. Se trata de máscaras centradas.

Con estas máscaras se calculan los gradientes horizontales (G_x) y verticales (G_y):

$$G_x = I \otimes m_x$$

$$G_y = I \otimes m_y$$

$$m_X = [-1,0,1]$$

$$m_Y = -[-1,0,1]^T$$

I es la imagen a tratar. G_x y G_y tienen tres dimensiones debidas a cada uno de los tres colores de la imagen (rojo, verde y azul). Se toma el gradiente máximo de los tres colores.

Se calcula el módulo y ángulo para cada píxel:

$$G = \sqrt{{G_x}^2 + {G_y}^2}$$

$$\theta = atan2(G_y, G_x)$$

Se divide la imagen en celdillas espaciales, y cada una de las celdillas se divide en 'bins' (ver figura 5) que es como llamaremos a las regiones angulares que determinan los histogramas angulares. Se toma cada píxel dentro de la celdilla y se suma su módulo (G) al 'bin' al que corresponda si su módulo es mayor que un umbral a imponer (este umbral será la media o la mediana de los módulos dentro de cada celdilla).

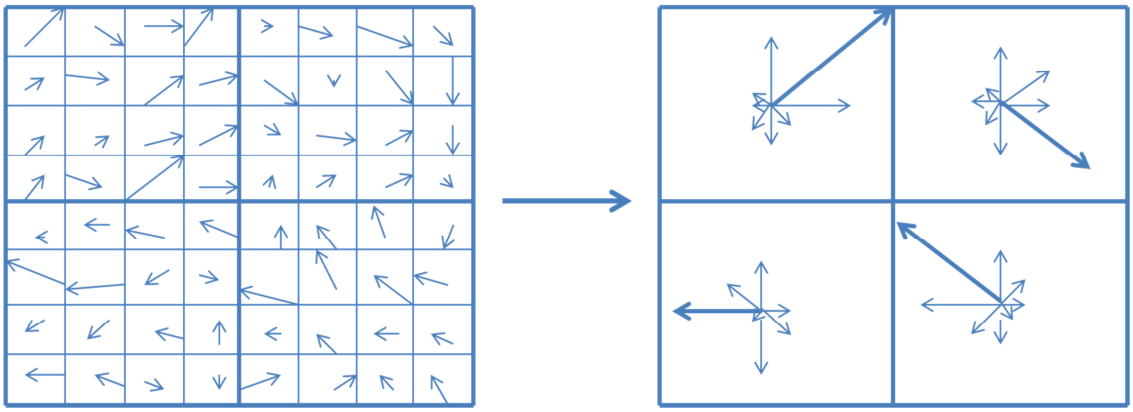

Figura 5. División de la imagen en 4 celdillas, para cada una se suman los gradientes en el histograma angular que en este caso se ha tomado de 8 regiones angulares.

2.1.2 Extensión a 3D

La formulación anterior es la utilizada para el procesamiento de imágenes, en nuestro caso tenemos un problema en una dimensión más debido al añadido del tiempo en el vídeo, por lo tanto se ha añadido la variable temporal a estas ecuaciones y en vez de tener una imagen bidimensional tenemos un vídeo en 3D. Nuestras variables serán 'x', 'y', y el tiempo 't'. En este caso se calculan dos ángulos (θ y ψ) para podes describir las dos dimensiones. Nótese esta extensión en la figura 6. θ se corresponde con el ángulo espacial igual que en 2D y ψ se corresponde al ángulo temporal.

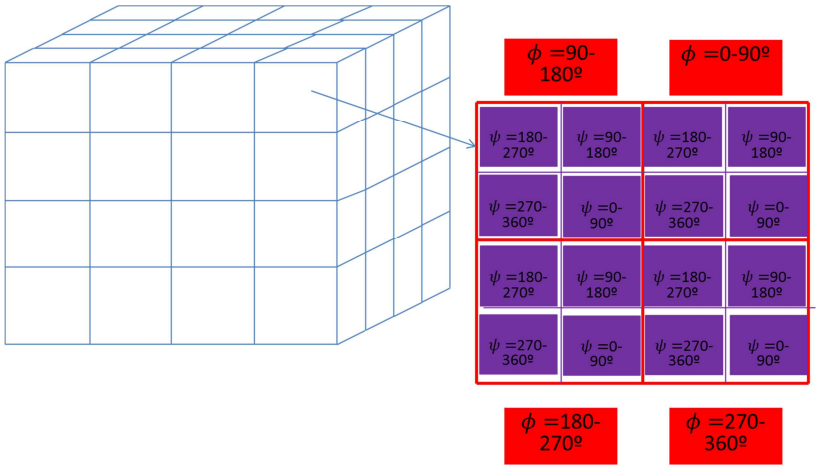

Figura 6. División del vídeo en celdillas, en este caso 64 celdillas, para cada una se suman los gradientes en el histograma angular que en este caso se ha tomado de 8 regiones angulares para cada uno de los dos ángulos, es decir 64 combinaciones angulares para cada una de las 64 combinaciones espacio-temporales de las celdillas.

2.1.3 Desarrollo del algoritmo

Se divide cada video en sus fotogramas. La primera operación que se hace en cada imagen es escalarla un factor a determinar (hacemos la imagen más pequeña) para conseguir que los pequeños detalles queden difuminados y no produzcan gradientes innecesarios, un ejemplo podría ser el reconocimiento de un portátil donde el contenido de la pantalla es innecesario para nosotros y sólo nos interesa los gradientes entre los bordes de este y el ambiente.

Respecto al formato de la imagen no cambiamos de color (rojo, verde, azul) a blanco y negro ya que está comprobado en [13] que el primero da mejores resultados en 2D como ya se ha comentado en el apartado 2.1.1. Tendremos 3 matrices por imagen (una para el color rojo, otra para el verde y otra para el azul).

Se utiliza la compresión gamma (γ), con ello conseguimos un efecto parecido al escalado de imagen, consiste en elevar cada pixel de la matriz a un factor igual a 0.5 con el objetivo de disminuir los gradientes, el resultado será que los grandes gradientes disminuirán pero seguirán teniendo su efecto y los pequeños gradientes tenderán a desaparecer y no 'molestar' a nuestro descriptor. Después de la compresión gamma se divide por el máximo valor de la imagen y se multiplica por 255 para situar todos los valores de la imagen entre toda la escala de valores desde 0 (negro) a 255 (blanco):

$$I = 255 * \left(\frac{I^{\gamma}}{\max(I^{\gamma})}\right)$$

Se realiza el cálculo de los gradientes respecto a cada una de las tres direcciones. Utilizamos máscaras derivativas centradas ([-1, 0,1]) igual que en el caso de 2D (apartado 2.2.1):

Tenemos las siguientes máscaras derivativas:

$$m_x = [-1,0,1]$$

$$m_y = -m_x{}^T$$

$$m_t = m_x$$

Las aplicamos al vídeo y calculamos el módulo y gradiente de cada píxel. Nótese que ahora tenemos dos ángulos, el ángulo θ igual que en imágenes y el ángulo ψ debido al cambio temporal.

$$G_{x,y,t} = I_{x,y,t} \otimes m_{\,x,y,t}$$

$G_{X,Y,t}$ tiene tres dimensiones debidas a cada uno de los tres colores de la imagen (rojo, verde y azul). Se toma el gradiente máximo de los tres colores.

El módulo y ángulo se calculan como:

$$G = \sqrt{G_x{}^2 + G_y{}^2 + G_t{}^2}$$

$$\theta = atan2(G_y, G_x)$$

$$\psi = atan2(G_x, G_t)$$

Calculamos dos ángulos ya que es la información necesaria para definir un vector en 3 dimensiones. A continuación se detalla la manera en la que se forma nuestro descriptor a partir de celdillas espaciales y temporales y 'bins' espaciales y temporales.

Un solo pixel no es representativo por lo que se juntan varios píxeles en celdillas para ser procesados conjuntamente y no uno a uno. Por ello se divide el vídeo en celdillas volumétricas (tendrán tres dimensiones, 2 espaciales y una temporal). Las celdillas o bloques del descriptor pueden tener tanto forma cúbica como estérica, en nuestro caso optaremos por la forma cúbica. En cada celdilla se forma un histograma de combinaciones angulares para cada uno de los dos ángulos. Para cada región angular se suma el módulo (G) de cada píxel si este módulo es mayor que un umbral (como en 2D este umbral será la media o mediana de los módulos de las celdillas).

Para clarificar, en el apartado resultados (apartado 8), se utilizará la nomenclatura ''4 celdillas espaciales'', ''5 celdillas temporales'' que significa que el vídeo se ha dividido en 4 partes para las dimensiones espaciales (horizontal y vertical) y en cinco partes para las dimensión temporal. Por ejemplo, en la figura 6 hay 4 celdillas en las dimensiones espaciales (dimensión horizontal y vertical) y 4 en las temporales. En total hay 4*4*4 = 64 celdillas. Además en este caso para cada celdilla tiene 8 regiones angulares o 'bins' para cada uno de los dos ángulos, es decir 8*8=64 combinaciones de regiones angulares en cada celdilla.

Una vez procesados todos los módulos de los píxeles de una celdilla se procede a su normalización para tener todas las celdillas en una escala fija y común.

Existen 4 tipos de normalizaciones, vamos a utilizar la segunda de la lista que sigue a este párrafo. Esta normalización también recorta los valores del descriptor a un máximo de 0.2 para evitar darle mas importancia a gradientes debidos a cambios no importantes como por ejemplo cambios fuertes debidos a cambios en la luminosidad de un cuerpo donde por un lado le da la sombra y en el otro le refleja la luz del sol. Probamos esta normalización tanto de forma global en el descriptor completo, como de forma singular en cada una de las celdillas y dio mejor resultado la segunda que es la que se utilizará.

Norma L2 $$f = \frac{v}{\sqrt{||v||_2^2 + \varepsilon^2}}$$

Norma L2.2: $$f = \frac{v}{\sqrt{||v||_2^2 + \varepsilon^2}}$$

Misma normalización que en caso anterior con el recorte de los valores a un máximo de 0.2

Norma L1: $$f = \frac{v}{\left(||v||_1 + \varepsilon\right)}$$

L1-raíz: $$f = \sqrt{\frac{v}{\left(||v||_1 + \varepsilon\right)}}$$

Las celdillas y 'bins' pueden tener superposición. Consiste en dividir el vídeo en celdillas y 'bins' de manera que las fronteras contiguas se solapen unas con otras y así hacer la división de una manera más suave. Para ver un ejemplo remito a la figura 7 donde se realiza este método en una imagen en 2D.

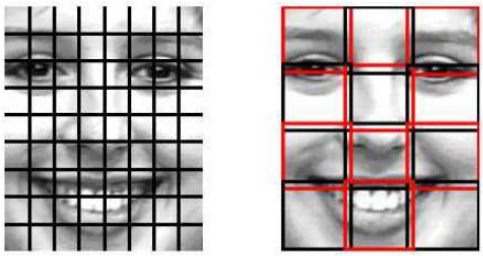

Figura 7. Ejemplo de celdillas ordinarias en la 1ª imagen y celdillas con superposición en la 2ª imagen. Figura tomada de [7]

Se forma el descriptor concatenando cada uno de los histogramas de combinaciones angulares de cada una de las celdillas. Por ejemplo si tenemos 4 celdillas espaciales (en las dos direcciones), 4 celdillas temporales y 8 'bins' para cada región angular el descriptor tendrá una longitud de 4096 = 4*4*4*8*8. Finalmente se realiza una normalización unitaria al descriptor completo.

2.2 Descriptor propuesto. Histogramas globales de gradientes de flujo orientados en 3D (G-HOF3D).

El flujo óptico es el movimiento que ha sufrido un píxel de una imagen a otra. Si tenemos una cámara estática el flujo óptico del fondo será nulo mientras que el fondo óptico de la acción que estemos grabando será distinto de cero.

El flujo óptico se centra en movimientos y no en textura. Si la textura está muy relacionada con la acción el flujo óptico perderá esa información del fondo. Se han probado dos modelos de cálculo de flujo entre dos imágenes, el modelo de Horn de 1981 [1] y el modelo de Roth y Black de 2010 [2]. Explicamos ambos en los apartados 2.2.1 y 2.2.2 y demostramos en el apartado 2.2.3 con una comparación de cálculo de flujo entre dos imágenes que el modelo explicado en [2], es mucho más preciso y lo utilizaremos para nuestros experimentos.

El descriptor G-HOF3D es el mismo que el descriptor G-HOG3D del apartado 2.1 pero en vez de analizar fotogramas se analizarán el flujo óptico de estos fotogramas. Es decir en vez de tener un video de imágenes tenemos un vídeo de flujo óptico. La principal diferencia es que el vídeo de flujo óptico contiene 'imágenes del movimiento'. Otra diferencia es que antes por cada imagen teníamos 3 matrices debidas a cada uno de los 3 colores y ahora tendremos dos matrices una debida al flujo horizontal y otra debida al flujo horizontal.

2.2.1 Flujo óptico [Horn 1981]

El flujo óptico es la idea de que al mirar un objeto en movimiento este lo vemos sin prestar demasiada atención al resto de la escena. Es la diferencia entre nuestro objetivo a seguir y el resto de la escena. A continuación se explica brevemente la solución de Horn, para una explicación detallada remito al lector al anexo A.

El flujo óptico de Horn se basa en la minimización del siguiente funcional:

$$E^2 = \int_{\Omega} [\alpha^2 {E_c}^2 + {E_b}^2] d\Omega$$

$${E_b}^2 = (\mathrm{I}_x\ \mathrm{u} + \mathrm{I}_y\ \mathrm{v} + \mathrm{I}_t)^2$$

$${E_c}^2 = {\mathrm{u}_x}^2 + {\mathrm{u}_y}^2 + {\mathrm{v}_x}^2 + {\mathrm{v}_y}^2$$

Donde I se corresponde con el volumen del vídeo, u y v son el flujo óptico horizontal y vertical. El subíndice 'x' se corresponde con derivada en la dirección horizontal y el subíndice 'y' se corresponde con derivada en la dirección vertical.
Queremos minimizar el anterior funcional ya que:

- E_C es la ecuación debida a que el objeto está sujeto a un movimiento como solido rígido o a una pequeña deformación. Esto quiere decir que el cambio en el flujo debe de ser muy pequeño y por tanto se requiere de la minimización del gradiente del flujo, es decir de E_C.
- E_b representa el cambio que ha sufrido un píxel de un fotograma a otro, este cambio también lo queremos minimizar ya que debe de ser cero.

2.2.2 Flujo óptico [Roth y Black 2010]

Se ha comprobado el algoritmo del aparado anterior para la detección de flujo por la imagen y no ha dado los resultados esperados (como veremos en el apartado 2.2.3). Se expone a continuación un algoritmo más preciso visto en [2].

Explico brevemente el flujo óptico de Roth y Black, para una explicación detallada remito al lector al anexo B.

Se parte de la función a minimizar, similar a la expuesta en el apartado 2.2.1 pero con la principal diferencia de la con la adición de funciones de penalización $\rho D\ y\ \rho S$:

$$E(u,v) = \sum_{i,j} \{\lambda_D \rho D(I_1(i,j) - I_2(i + u_{i,j}, j + v_{i,j})) + \lambda_S [\rho S(u_{i,j} - u_{i+1,j)} + \rho S(u_{i,j} - u_{i,j+1)} + \rho S(v_{i,j} - v_{i+,j)} + \rho S\big(v_{i,j} - v_{i,j+1)}\big]\}$$

$$E_D = \sum_{i,j} \{\lambda_D \rho D(I_1(i,j) - I_2(i + u_{i,j}, j + v_{i,j}))\}$$

Consiste en encontrar los valores del flujo óptico u y v que minimizan la función anterior, u y v son el número de píxeles de movimiento entre dos fotogramas consecutivos. El primer término se corresponde con la diferencia entre un punto de la imagen primera y su homólogo en el nuevo fotograma, que en un caso ideal sería cero. El segundo término consiste de la consideración de que estamos trabajando con un sólido rígido y se dan pequeñas deformaciones por lo que el flujo no puede tener grandes variaciones. λ es un parámetro de regularización mientras que ρ son las funciones de penalización de la información dada (primer término) y penalización espacial debida a las diferencias en flujo (segundo término).

Las funciones de penalización tienen como función el formalizar los errores, devuelven siempre un número positivo y no surge el fenómeno de cancelación de errores. Las posibles funciones de penalización que se utilizan en este algoritmo son:

-Cuadrática o HS. $\rho(x) = x^2$

-Charbonier o Classic-C. $\rho(x) = \sqrt{x^2 + \varepsilon^2}$

-Lorentzian o Classic-L. $\rho(x) = \log(1 + \frac{x^2}{2\sigma^2})$

2.2.3 Comparación entre ambos

Se utilizará el flujo óptico de Roth y Black [2] ya que como podemos apreciar en la imagen de abajo a la derecha en la figura 8 el flujo de estas personas en movimiento es constante en sus cuerpos y no se crean gradientes debidos a cambios en textura no importantes como los pliegues de su ropa que si se dan en el flujo de Horn (abajo a la izquierda en la figura 8).

Figura 8. Arriba dos fotogramas consecutivos, abajo flujo óptico para ambos. El flujo óptico calculado abajo a la izquierda corresponde con el algoritmo de Horn y el de abajo a la derecha con el algoritmo de Roth y Black.

2.3 Detección del movimiento. Ventana de seguimiento de la acción.

Nuestro objetivo es describir el movimiento que sucede en un vídeo. Vamos a utilizar el descriptor global en este experimento pero vamos a acotar la zona del fotograma a aplicarlo, sólo se aplicará en la zona que éste que está en movimiento. Nos centramos en la zona en movimiento y de este modo no importa en que zona del vídeo se encuentre la acción. En contraposición haciendo esto nos olvidamos de la escena que puede ser tan importante como la acción.

La idea principal es buscar el movimiento de la acción y separarla del fondo. Cuando las cámaras están en movimiento resulta muy complicado ya que se produce movimiento por toda la imagen. Se intentó separar el movimiento del fondo de la persona con cámaras móviles pero no fue posible ya que el movimiento del fondo no es constante. Su flujo óptico varía ya que no todas las zonas a las que apunta la cámara están a la misma distancia y por

tanto su velocidad será diferente y se producirán muchos flujos distintos de los cuales será difícil predecir cual es nuestro objetivo.

En cambio este problema no surge si la cámara es estática como ocurre en el dataset KTH [15], realmente a veces no es estática del todo pero debido a la planitud del fondo se podrá diferenciar el fondo de la acción con bastante exactitud.

En primer lugar se pasa un filtro de mediana por todas las imágenes para eliminar los píxeles espurios. Se toman los fotogramas de 3 en 3 y se calcula el gradiente temporal del fotograma central:

$$G_t = I_t \otimes m_t$$

Una vez tenemos este gradiente temporal se calcula las gradientes en el eje horizontal y en el eje vertical de este gradiente temporal para calcular los bordes donde se ha producido el movimiento. Finalmente se calcula el módulo de estos dos gradientes verticales y horizontales:

$$G_{tx} = G_t \otimes m_x$$

$$G_{ty} = G_t \otimes m_y$$

$$G = \sqrt{{G_{tx}}^2 + {G_{tx}}^2}$$

Los valores altos (más altos que un umbral que ahora determinaremos) de 'G' determinarán si hay movimiento o no en la imagen. Se pasa un filtro gaussiano por 'G' para suavizar los gradientes de la imagen ya que en ocasiones surgen en lugares de la imagen donde no hay nada en movimiento y son debidos a cambios de luminosidad, al ser tan escasos se corrigen en su mayoría con este filtro gaussiano.

Se calcula el histograma de este módulo (figura 9), se ve claramente como hay una zona inicial muy grande debida a todos los píxeles que no se mueven y una zona más extensa pero mucho más lineal y con una repetición mucho más baja debida a la acción en movimiento.

La zona del histograma (figura 9) debida al movimiento se corresponde con valores bajos y constantes entre si. Se procederá de la siguiente manera (figura 9): una vez se detecten 5 valores seguidos en el histograma menores que un valor límite se tomará el valor mínimo de esos valores como el valor mínimo que tendrá movimiento.

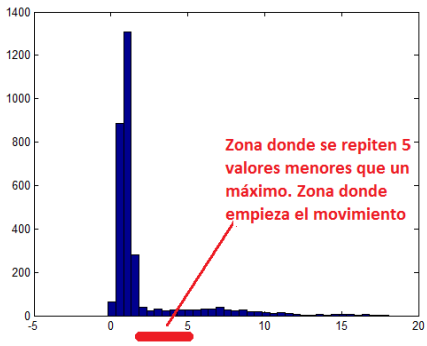

Figura 9. Histograma de los gradientes en textura de los gradientes temporales.

En el módulo (G) de cada píxel se buscan valores mayores que el dado por el histograma anterior y así tendremos los píxeles que se corresponden con acción en movimiento. Se tomará la ventana que englobe a todos los píxeles que estén en movimiento no sin antes incrementar la ventana un 30% en ambas direcciones para dar margen.

Con la forma de cálculo anterior surge el problema de que en un fotograma la ventana de movimiento detecta por ejemplo sólo las manos de la persona y en el siguiente detecta la persona entera, por tanto estamos mezclando las manos con otras zonas del cuerpo, como puede ser la cabeza y esto no es del todo correcto. Por ello se realiza un sistema de votaciones, cada 10 fotogramas se calcula el tamaño dado por la mediana de los tamaños. La posición de la imagen se calcula del mismo modo, así se tiene un tamaño estándar cada diez fotogramas. Para pasar de este tamaño al dado por los siguientes diez fotogramas se realizan de forma lineal y no de forma brusca.

Finalmente la tercera modificación viene dada ya que la modificación anterior no arregla del todo el problema dado en la primera aproximación así que en este caso se toman todas las ventanas que describen el vídeo y se forma una ventana fija y global que englobe a todas las ventanas calculadas en cada fotograma. Esta ventana fija describe bien la zona de movimiento a lo largo de todo el vídeo como vemos en la figura 10.

Figura 10 Fotograma completo y ventana de movimiento.

2.4 Descriptor global basado en líneas

Para una explicación detallada remito al lector al anexo C.

Se utiliza una técnica llamada detector de Canny [8] para detectar las líneas de una imagen. Las líneas tendrán valor uno y los demás píxeles tendrán valor 0. Con la transformada de Hough, que es una función de Matlab, se calcula la posición 'ro' y 'theta' de cada pixel con valor uno en la imagen (ver figura 11). De esta manera hacemos un sistema de votaciones para estas dos variables. Esta votación no la realizamos en toda el volumen del vídeo si no que también empleamos celdillas espaciales y temporales, en cada una de ellas realizamos la detección de líneas y su respectivo sistema de votaciones, así formamos el descriptor que entrenará a la maquina de aprendizaje.

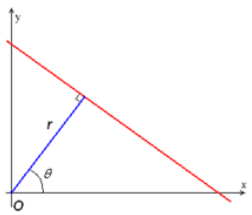

Figura 11 Línea parametrizada por 'r' y θ . Figura tomada de [8].

2.5 Descriptor global basado en espacio de colores (L-a-b)

Este espacio de colores sirve para procesar los colores que puede percibir el ojo humano. 'L' representa la luminosidad (0 = blanco, 100 = negro), 'a' representa su posición entre magenta y verde, valores negativos representan más cerca de magenta y positivos de verde y lo mismo ocurre para 'b', donde representa la posición entre amarillo y azul.

Se realiza un histograma para 'L', 'a' y 'b' en 4, 14 y 14 regiones respectivamente para formar un descriptor global de 14*14*4 = 784 dimensiones.

2.6 Descriptor global basado en patrones locales binarios

Para una explicación completa remito al lector al anexo D.

Se utiliza el código público [7] para su implementación.

Los patrones binarios se determinan mediante un número binario que surge de la comparación de un píxel central con los píxeles de alrededor. Si un píxel tiene un valor en la escala de grises más alto que el píxel central entonces se le atribuirá un 1, si no un 0. Se sitúan los 0´s y 1´s en un determinado orden para formar un número binario que se pasará a decimal. La frecuencia normalizada de estos números determinará nuestro descriptor. Se puede aplicar el algoritmo a todo el vídeo o se puede discretizar en celdillas volumétricas para después concatenarlas como se ha hecho en los demás descriptores.

2.7 Descriptor global basado en restas entre vídeos

Método basado en la idea vista en [16] sobre reconocimiento de imágenes a partir de su parecido con un dataset grande de imágenes. En este artículo se toman 80 millones de imágenes en escala reducida y se tienen como un dataset de entrenamiento. Dada una nueva imagen para reconocer se compara con las imágenes de entrenamiento y la imagen a la que más se parezca le atribuirá la clase que tenga. En este proyecto realizamos esta misma idea pero en vídeos.

Dado un vídeo a clasificar se calcula la distancia de este respecto a los vídeos de entrenamiento y el que de la distancia mínima determinará la clase de vídeo a la que pertenece.

Se escalan todos los vídeos y se pasan a escala de grises. Los hacemos pequeños en tamaño para eliminar los detalles de las imágenes, no queremos detalles como puedan ser pliegues de la ropa, colores, gorras, etc. Sólo queremos la esencia de lo que esta ocurriendo.

Además de reducirlos, hay que hacerlo a un tamaño común ya que la comparación será la norma de la resta entre volúmenes de vídeos. La norma mínima será el vídeo al que más se parece.

$$d(i) = norma(I(i) - I)$$

Donde 'I' es el vídeo a clasificar, 'd (i)' es la distancia de este vídeo al vídeo de entrenamiento e 'I (i)' es cada uno de los vídeos de entrenamiento. La mínima 'd (i)' dará la clase del vídeo a clasificar.

3. Descriptores locales de vídeo

Los descriptores locales de vídeo operan en tres pasos, primero se calculan puntos de interés del vídeo (apartado 3.1), con estos puntos se calculan sus descripciones (apartado 3.2) y finalmente con la descripción de todos los puntos se crea un diccionario donde estos descriptores se agrupan en 'K' grupos o palabras (apartado 3.3). Al conjunto de palabras se le denomina bolsa de palabras. Cada punto clave pertenecerá a una palabra y la frecuencia de cada palabra dentro del vídeo determinará el descriptor de nuestro vídeo. Dado un nuevo vídeo a clasificar se calcularán sus puntos clave y descriptores para clasificarlos con las palabras surgidas en el entrenamiento, le frecuencia de estas palabras determinará el descriptor a comparar con los descriptores de entrenamiento.

3.1 Localización de puntos de interés

Uno de los métodos más utilizados para la detección de puntos de interés en los últimos años han sido los detectores de Harris en 3D (apartado 3.1.1) como se puede ver en [17], además ha habido otros tipo de selecciones de puntos clave como es la clasificación densa, es decir, escoger los puntos clave de una forma distribuida y prefijada, y también otro tipo de selección surge de escoger estos puntos clave al azar. Existen otras formas de detección pero en este proyecto final de carrera se presenta una manera no utilizada todavía, la detección de estos puntos clave con un detector SIFT en 3D (apartado 3.1.2) que surge de extender el detector SIFT en 2D [18] a la tercera dimensión temporal.

3.1.1 Detector de Harris en 3D

Para una explicación detallada remito al lector al anexo E.

El detector de Harris calcula puntos de interés que tengan elevados gradientes en las tres direcciones (x, y, t). Para ello calcula la siguiente matriz 'A' en cada píxel y determina como puntos característicos aquellos que tienen los valores propios altos en las tres direcciones.

$$A = \sum_u \sum_v \sum_t w(u,v,t) \begin{bmatrix} I_x^2 & I_x I_y & I_x I_t \\ I_x I_y & I_y^2 & I_y I_t \\ I_x I_t & I_y I_t & I_t^2 \end{bmatrix}$$

Donde 'w' es el suavizado e $'I_{x,y,t}'$ son los gradientes de el vídeo en cada una de las tres direcciones.

3.1.2 Detector propuesto. Detector SIFT en 3D

Tomamos la idea de [19] donde aplica un detector SIFT 3D para detectar puntos en imágenes en 3D. Adaptamos este algoritmo al nuestro cambiando su tercera dimensión 'z' por nuestra tercera dimensión temporal 't' para aplicarlo en vídeos y no en imágenes en 3 dimensiones. Es un algoritmo más avanzado que el Harris 3D para la detección de puntos característicos ya que con este algoritmo no solamente detectamos puntos salientes en las 3 dimensiones sino que también nos aseguramos de que serán puntos invariantes a la escala. Se aplica a diferentes tamaños del vídeo con el objetivo de detectar diferentes puntos clave, en un video ampliado detectaremos muchos puntos y en un vídeo reducido detectaremos muchos menos (ver figura 12).

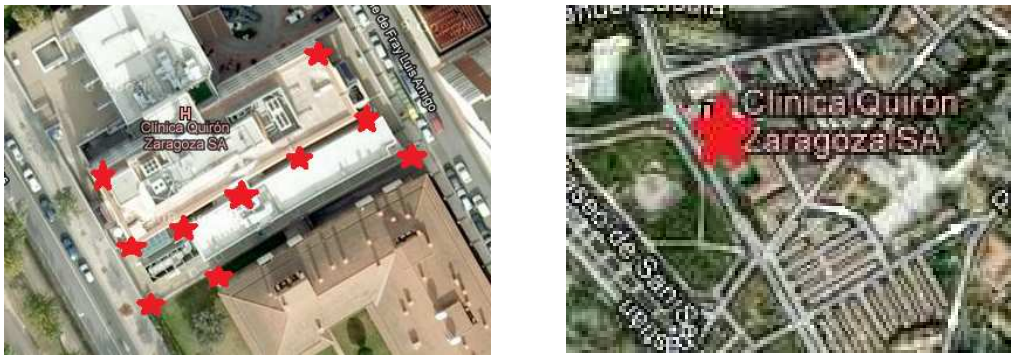

Fig.12 Comparación entre los puntos clave detectados a una escala cercana (1ª imagen) y una escala lejana (2ª imagen).

Los puntos detectados a escalas diferentes no serán los mismos. En un vídeo de tamaño grande podemos detectar muchos puntos en una casa pertenecientes a las ventanas, puertas, etc. En el mismo vídeo reducido sólo tendremos un punto clave perteneciente a toda la casa como vemos en la figura 12. Además, como explicaremos en el siguiente párrafo, esta búsqueda de puntos clave también se realiza buscando máximos y mínimos respecto a distintos suavizados en la imagen, que es la principal característica de los detectores SIFT.

En primer lugar se adapta el vídeo a una escala oportuna (una escala menor para que el coste computacional sea menor), si es la escala la original mejor porque se detectarán más puntos clave como se ve en la figura 12. Se pasa el vídeo a escala de grises y se normaliza el valor de cada pixel a escala [0,1] para poder realizar las restricciones que se impondrán una vez detectados los puntos característicos para eliminar los puntos erróneos. El siguiente paso consiste en formar las octavas, la octava consiste en formar una pirámide de vídeos suavizados cada vez en mayor medida.

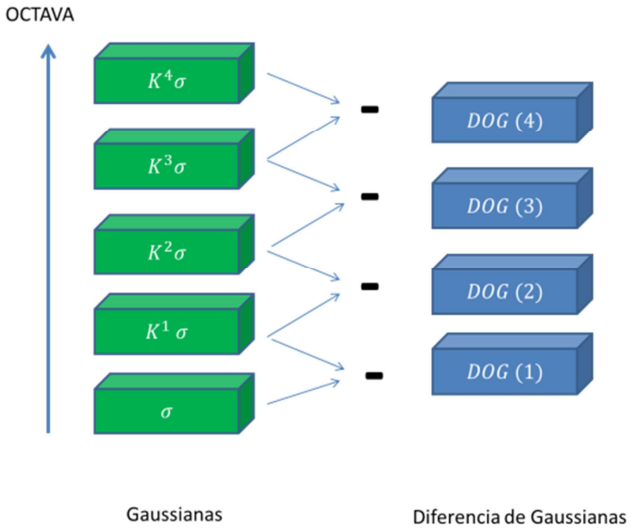

Fig.13 Formación de octavas. En cada octava tenemos una pirámide de vídeos de un mismo tamaño. Cada vídeo de la pirámide se suaviza y conforme estamos en un valor más alto de la pirámide el suavizado es mayor. Por cada par de vídeos suavizados con estas gaussianas (vídeos de color verde) se realiza una extracción que resulta en los vídeos diferencia de gaussianas (DOG, vídeos de color azul) y finalmente se procede a la búsqueda de máximos y mínimos locales (figura 14)

El primer vídeo se suaviza con un valor $'\sigma'$ que irá incrementando su valor al multiplicarlo por un factor 'k' constante como podemos ver en la figura 13. Una vez tenemos construida esta

pirámide llamada octava se sustraen 2 a 2 estos vídeos suavizados como podemos ver en la citada figura.

Obtenemos los vídeos diferencia de gaussianas (DOG). La siguiente octava se formara tomando el vídeo suavizado número 4 de la pirámide verde de la figura 13 y se situará en el primer nivel de la siguiente pirámide para volver a repetir el proceso. Una vez tenemos todas las diferencias de gaussianas se procede a la búsqueda de los puntos salientes. Para ello, como se puede apreciar en la figura 14, se toma cada pixel de las diferencias de gaussianas (DOG (2) y DOG (3)) y se busca si es un máximo o mínimo local respecto de los 26 píxeles que tiene alrededor y respecto de los 27 píxeles que tiene en las 'DOG' posterior y anterior a él, si este píxel resulta ser un máximo o mínimo local respecto de estos 80 píxeles será un punto clave.

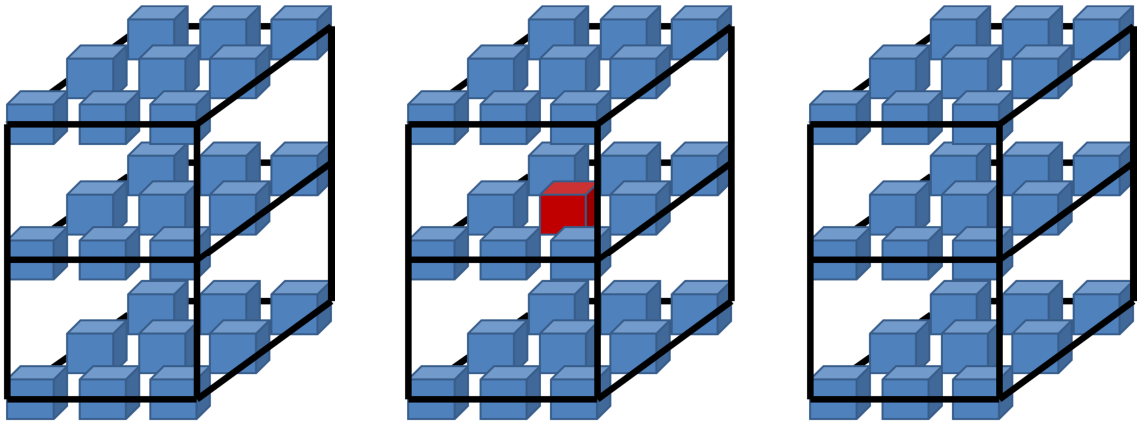

Figura 14. Búsqueda de máximos y mínimos locales. Para cada píxel de DOG(2) Y DOG(3) se comprueba si es un máximo/mínimo local comparándolo con los 26 píxeles que tiene alrededor en su mismo volumen y con los 27 pixeles de las DOG(1) Y DOG(3) en el caso de DOG(2) y DOG(2) Y DOG(4) en el caso de DOG(3) correspondientes a la misma zona volumétrica. Si el píxel rojo resulta ser un máximo o mínimo entre estos 80 píxeles será un punto característico potencial. Este proceso se repite para cada escala del vídeo.

Se imponen dos restricciones para eliminar puntos característicos mal calculados:

- Restricción puntos pobremente localizados en un borde:
  Además de querer puntos máximos y mínimos locales también queremos que cumplan que tengan una gran variación en las tres direcciones en las que está definido. Para ello se forma la matriz hessiana a partir de las derivadas segundas de las diferencias de gaussianas 'DOG':

$$H = \begin{bmatrix} DOG_{XX} & DOG_{XY} & DOG_{XT} \\ DOG_{XY} & DOG_{YY} & DOG_{YT} \\ DOG_{XT} & DOG_{YT} & DOG_{TT} \end{bmatrix}$$

  Está demostrada la validez de la siguiente ecuación para rechazar estos puntos:

$$Eliminamos\ punto\ si:$$

$$\frac{traza^{\,3}}{determinante\ (\,H\,)} < \frac{(2r+1)^3}{r^2}$$

$$r = 40$$

$$traza = \ DOG_{XX} + \ DOG_{YY} + \ DOG_{TT}$$

- Restricción de poca densidad.
  Si la diferencia de gaussianas es menor que un cierto umbral (0.03 en nuestro caso) no tomamos ese punto como punto clave, la razón es que si DOG < 0.03 significa que no hay un cambio de contraste necesario para considerar un punto clave en el tiempo, debe de haber un alto contraste tanto en el cambio de textura como en el cambio temporal.

3.2 Descriptores locales

3.2.1 Histogramas locales de gradientes orientados en 3D (S-HOG3D)

Mismo descriptor que el utilizado para analizar los vídeos de manera global (apartado 2.1), en este caso en vez de utilizar este descriptor sobre todo el vídeo se utilizará sobre una región alrededor del punto clave. Esta región se determinara como ¼ de la anchura del vídeo, ¼ de la altura del vídeo y ¼ de la dimensión temporal del vídeo.

3.2.2 Descriptor SIFT en 3D

Se expande el descriptor anterior donde se introducen 3 innovaciones vistas en [20]:

- Invariancia a rotación

  Se realiza una reorientación de los dos ángulos, se calcula dos histogramas de las magnitudes de los gradientes, dividiendo el vídeo en 36 zonas angulares para el ángulo θ y otro para el ángulo ψ. En estos histogramas se toma la zona angular donde la suma de las magnitudes es mayor y se coloca este ángulo como referencia (θ=0 y ψ = 0) para así hacer nuestro sistema invariante a la rotación

- Ángulo sólido.

  Se dividen las magnitudes de los gradientes por el ángulo sólido (w) para hacerlas independientes del ángulo que tienen. $G = \frac{G}{w}$

- Ventana gaussiana aplicada al vídeo

  Se aplica una ventana gaussiana de 3D a la zona local de vídeo a describir con el propósito de que los gradientes centrales tengan más importancia que los gradientes en los extremos. Es decir los gradientes más cercanos al punto clave serán tomados más en cuenta que los más lejanos.

3.3 Bolsas de palabras (BOW)

Se utiliza la bolsa de palabras vista en [21]. La bolsa de palabras es un diccionario donde se almacenan un número de palabras a determinar a partir de los descriptores calculados en cada punto clave. Los vídeos de entrenamiento nos darán un conjunto de 'n' descriptores que agruparemos en 'K' grupos que formarán la bolsa de palabras. Cada grupo tendrá un centroide y dado un vídeo de test calcularemos la distancia de sus puntos clave a cada uno de los

centroides de cada palabra y el que de una distancia menor nos dará la clase del punto clave. Para agrupar los descriptores de los puntos clave en grupos se opera como se indica en la figura 15. Para la agrupación se ha utilizado el algoritmo 'k-means'.

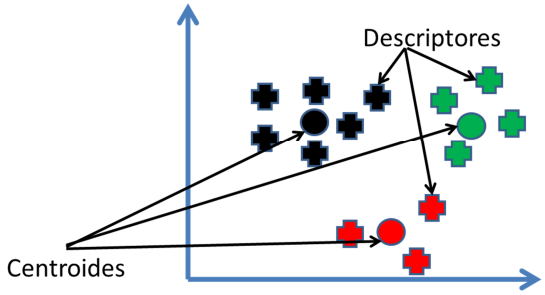

Figura 15. Formación de grupos a partir descriptores caracterizados por sus características, en este caso se forman 3 grupos a partir de 13 descriptores y 2 características por descriptor. Los 13 descriptores de la imagen no tienen ninguna clase atribuida en un principio. Al utilizar la función k-means con k=3 grupos se forman los tres grupos (negro, verde y rojo en este caso) con sus respectivos centroides.

Una vez tenemos un vídeo descrito con un conjunto de puntos clave y la clase a la que pertenece cada punto clave se procede a la formación del descriptor. Para ello se divide el vídeo en zonas o contextos como vemos en la figura 16, para cada una se suma la frecuencia de cada punto clave que aparece en el contexto. Esto se realiza con una función probabilística (Kernel gaussiana) la cual le da más peso a los puntos clave cercanos al centro del contexto que a los puntos clave de los extremos. La probabilidad de que una palabra/punto clave $\boldsymbol{x}_i$ exista dado que estamos en un contexto C_j se calcula como:

$$p(\boldsymbol{x}_i|C_j) = \sum_{x_i \epsilon C_j} K(\boldsymbol{x}_i, \boldsymbol{x})$$

$$K(\boldsymbol{x}_i, \boldsymbol{x}) = exp\left(-\left(\frac{(x-x_i)^2}{\delta_x^2} + \frac{(y-y_i)^2}{\delta_y^2} + \frac{(t-t_i)^2}{\delta_t^2}\right)\right)$$

Donde 'K' es la función Kernel Guassiana, 'x' es la localización del centro del contexto, 'x_i'es la localización del punto clave/palabra y '$\delta_{x,y,t}$' es la anchura del contexto en cada una de las tres direcciones.

Después de calcular la probabilidad de cada palabra se normaliza respecto a las demás:

$$b(x_i|C_j) = p(x_i|C_j) \,/ \sum_{x \epsilon C_j} p(x|C_j)$$

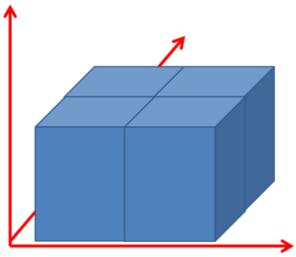

Figura 16. División del vídeo en contextos, en este caso dos contextos en la variable horizontal, dos en la variable temporal y uno en la variable vertical.

Finalmente se concatena el vector de características de cada contexto (que serán las frecuencias normalizadas de cada punto clave en el contexto). El descriptor queda:

$$d = [b(x_1|C_1), \dots, b(x_n|C_1), \dots, b(x_1|C_2), \dots, b(x_n|C_2), \dots, b(x_1|C_m), \dots, b(x_n|C_m)]$$

Donde 'n' es el número de palabras y 'm' el número de contextos.

4. Clasificador

Para una explicación más detallada del algoritmo remito al lector al Anexo F.

Se utiliza una maquina de aprendizaje supervisada, el adjetivo supervisada viene de que la máquina recibe vídeos de entrenamiento y también recibe las clases de este vídeo, si no conociera las clases sería no supervisada. Esta máquina recibe de cada vídeo la información de su descripción y la clase de video a la que pertenece. Con esta información la máquina calcula las fronteras entre distintas clases para una vez queremos clasificar un nuevo vídeo calcular dentro de que frontera se sitúa su descriptor y así tratar de adivinar a que clase pertenece.

Utilizamos las librerías que se han desarrollado [10] en los últimos años. Una forma intuitiva para entender como funciona esta librería podemos verla en la figura 17. En ella tenemos un espacio bidimensional, es decir cada clase esta compuesta por dos características. Hay un total de 3 clases, la morada, azul y amarilla. La máquina calcula las fronteras como vemos en la imagen de la derecha de la figura 17. Dado un nuevo vídeo se comprueba en que zona se encuentra de las 3 para determinar su clase.

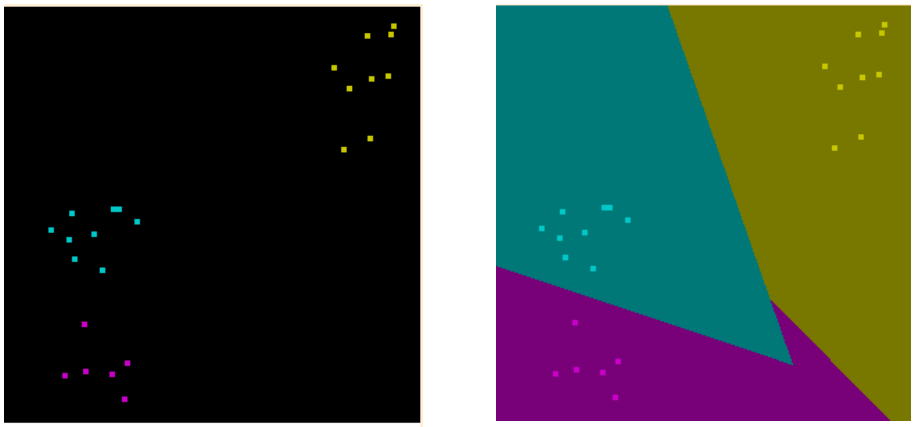

Figura 17. Ejemplo ilustrativo de como funciona el SVM, 3 clases y 2 características. Tomado de http://www.csie.ntu.edu.tw/~cjlin/libsvm/

5. Comparación entre descriptores locales y globales.

Como se puede apreciar en la figura 18, en el primer caso se extraen puntos característicos alrededor de los cuales se utiliza el descriptor HOG3D, con todas estas descripciones se forma una bolsa de palabras donde la frecuencia de cada uno de ellos da lugar al descriptor buscado. Algo parecido surge en segundo caso, la única diferencia es que los puntos característicos los determinamos nosotros de forma que el descriptor analice todo el vídeo de forma continua. En el último caso tenemos nuestro descriptor donde se aplica el G-HOG3D (apartado 2.1) a todo el vídeo y se forma así el descriptor.

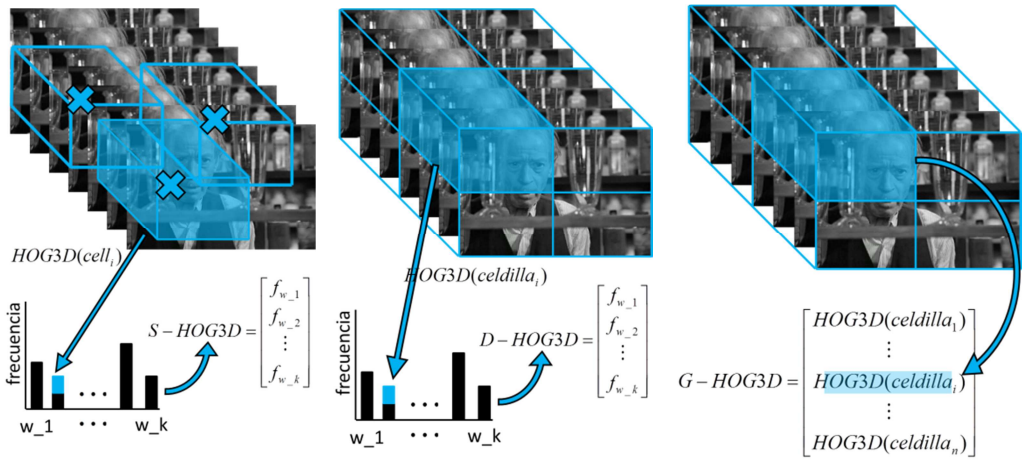

Figura 18. Comparación entre descriptores locales (1º y 2º) y globales (3º). Figura extraída de [11]

6. Comparación entre G-HOG3D y G-HOF3D

La principal diferencia entre ambos es que G-HOF3D analiza la acción en los vídeos, el movimiento, no analiza la escena. Esto se puede apreciar en las magnitudes de sus vectores en la tercera imagen de la figura 19 donde en una se aprecia la escena perfectamente y en la cuarta imagen sólo se aprecia la deportista con el balón sin incluir la cancha de baloncesto.

Figura 19. Comparación entre gradientes de textura (3ª imagen) y de flujo (4ª imagen). Figura extraída de [11]

7. Dataset

Se ha elaborado un dataset para testear los descriptores (véase dos clases en la figura 20). Está formado por 8 clases de vídeos, en la figura 20 se puede ver dos clases de este dataset. Para ver el resto de clases remito al Anexo K. La prueba del descriptor en este dataset propio resulto satisfactoria pero se tuvo que buscar otros dataset disponibles en la web para poder hacer una comparación realista con los demás descriptores. En el apartado 8 se introducen estos descriptores y los resultados obtenidos.

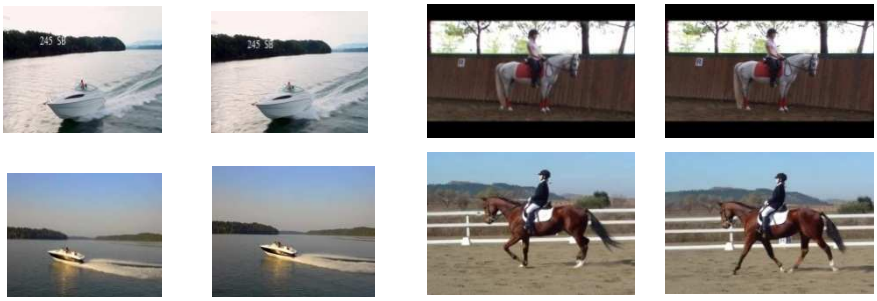

Figura 20. 2 clases del dataset realizado. La primer es la clase lancha motora y la segunda la clase caballo y jinete.

8. Experimentos

8.1. Experimentos para el descriptor global

8.1.1 Distancia entre descriptores

Se comprueba la validez de nuestro descriptor calculando las distancias entre histogramas de distintos vídeos. Por ejemplo el vídeo avión 1 aterrizando tiene la mínima distancia con el vídeo avión 2 aterrizando (0.25) mientras que las distancias con los otros vídeos que no se corresponden con la clase avión aterrizando son entre 0.35 y 0.52. Para visualizar los resultados completos remito al lector al anexo G

8.1.2 Experimentos realizados con 'dataset Hollywood2' [14]. Sintonización de parámetros

Para una explicación más detallada del trabajo realizado y para visualizar todas las clases de este dataset remito al lector al Anexo H.

Se utiliza el dataset elaborado por Ivan Laptev [14] para afinamiento de los parámetros de nuestro descriptor. Véase en la figura 21, 2 de las 12 clases utilizadas en este dataset.

Figura 21. 2 clases en el 'dataset Hollywood2'. Conducir (arriba) y levantarse (debajo).

Se trata de un 'dataset' de películas de Hollywood. Es utilizado por diversos autores que han investigado este problema para así poder hacer una comparación con el estado del arte. Este 'dataset' se divide en 12 acciones (descolgar el teléfono, beso, abrazo, levantarse, sentarse…) sacadas de diversas películas de Hollywood, unas se utilizan para realizar el entrenamiento y otras distintas para testear nuestro clasificador.

Debido a su gran extensión se ha llevado a cabo en sólo una parte de los vídeos. Se han tomado 20 vídeos de entrenamiento por clase, 10 vídeos por clase para su validación y 20 vídeos por clase para el test final.

Se ha medido la calidad del descriptor en relación a la precisión de la media, que es un parámetro resultado del cociente entre los falsos positivos por clase y la suma de los positivos totales (falsos y verdaderos).

$$precisión = \frac{\text{verdaderos positivos}}{\text{verdaderos positivos} + \text{falsos positivos}}$$

Como adelanto de los resultados la configuración que mejor resultados ha dado ha sido **con 6 celdillas espaciales, 4 celdillas temporales, 6 bins angulares, 48 píxeles en altura para el tamaño del vídeos y el uso de la mediana de la celdilla en la aceptación de un módulo como significante**. Estos resultados han sido de un **22.3 %** de precisión. No se han obtenido resultados muy altos en relación a otros 'datasets' ya que se han tomado muy pocos vídeos para así favorecer la rapidez computacional y poder optimizar los parámetros más rápidamente.

Por otra parte se han obtenido unos resultados bastante independientes a la variación de parámetros, podemos decir que nuestro descriptor es bastante robusto a pequeños cambios y no cambia mucho al darse pequeñas variaciones en los parámetros, lo contrario no hubiera sido un buen resultado ya que depender de un descriptor que cambia en demasía con la variación de parámetros no aporta demasiada confianza cuando se intenta probar un conjunto nuevo de vídeos ya que los valores que optimicen a estos nuevos vídeos pueden ser muy diferentes que en experimentos anteriores. Esto haría que el sistema fuera más caro computacionalmente hablando, en cualquier caso hay unas pequeñas diferencias en los resultados que procedemos a comentar:

- En una prueba inicial se obtienen unos resultados de un 11%, 4% superiores a los obtenidos al azar. Nos damos cuenta de un problema que tiene este dataset y es que en cada vídeo se dan muchas tomas que no tienen nada que ver con la clase a reconocer. Por ello se realiza un filtrado de los vídeos para seleccionar sólo las tomas buenas y los resultados mejoran el doble. Dando un 20.2% respecto al 7% que da el azar.
- En este dataset el uso de la mediana de la celdilla como límite inferior (apartado 2.1.3) para la aceptación de un gradiente funciona mejor que la media.
- El aumento del número de celdillas y 'bins' mejora los resultados hasta llegar a un punto de inflexión donde empeoran. En este datase ese punto es 6 celdillas espaciales, 4 celdillas temporales y 6 'bins' para ambos ángulos.
- El tamaño de la imagen es mejor que sea pequeño para eliminar detalles irrelevantes. Se prueban los tamaños 24, 36, 48 ,72 y 96. Un tamaño de 36 y 48 proporciona los mejores resultados. Un tamaño de 24 es demasiado pequeño y a partir de 64 demasiado grande.

8.1.3 Experimentos realizados con 'dataset KTH' [15]. Sintonización de parámetros

Para una explicación más detallada del trabajo realizado y para visualizar todas las clases de este dataset remito al lector al Anexo I.

Se trata, como vemos en la figura 22, de un dataset más sencillo a la hora del reconocimiento debido a la sencillez de sus acciones y el emplazamiento del sujeto en un fondo constante donde la única zona móvil es la persona que realiza la acción.

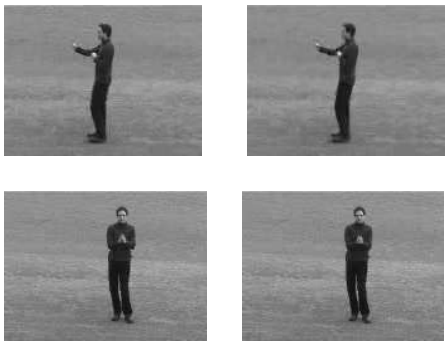

Figura 22. Clases boxear y dar palmas en dataset KTH

Se sigue sintonizando los parámetros de nuestro descriptor. Para visualizar el desarrollo completo de estas pruebas remito al lector al anexo, a continuación expongo los resultados más relevantes:

- Darle un mayor peso a los gradientes temporales respecto a los espaciales empeora los resultados
- En este dataset el uso de la media de la celdilla como límite inferior (apartado 2.1.3) para la aceptación de un píxel funciona mejor que la mediana.
- El suavizado llevado a cabo por la compresión gamma (apartado 2.1.3) da peores resultados al suavizar y eliminar parte de los gradientes.
- Una vez normalizadas las celdillas se les impone un límite superior para acotar gradientes muy altos debidos a errores en el cálculo de gradientes como puede ser la presencia de luces y sombras. Este límite es beneficioso y el valor óptimo para este dataset es de 0.4 aunque es muy parecido al resultado que da el límite de 0.2 ordinario.
- En este caso un aumento de bins de 6 a 8 no empeora los resultados.
- Finalmente la detección del movimiento presentada en 2.3 produce un incremento de resultados del 4% dando la mejor tasa de reconocimiento obtenida en este apartado para este dataset siendo esta del **89.25%** como se ve en la tabla 1.

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
G-HOG3D	**1**	**0.9022**	0.7007	0.7938	**0.9277**	0.8276	0.8587
G-HOG3D con detección de movimiento	0.9868	0.8812	**0.8807**	**0.8734**	0.8269	**0.9057**	**0.8925**

Tabla 1 Comparativa entre G-HOG3D con y sin la ventana de detección de movimiento.

8.1.4 Experimentos con otros descriptores

En la tabla 2 se recogen los resultados de otros descriptores descritos en la sección descriptores tales como: descriptor basado en 'líneas' (apartado 2.4) , descriptor de colores 'LAB' (apartado 2.5), 'Local Binary Patterns ' (apartado 2.6) o descriptor basado en la resta

entre vídeos escalados a una escala pequeña (apartado 2.7). En el caso de los 'Local Binary Patterns' se realiza utilizando celdillas y sin utilizarlas, es decir, se calculan los patrones locales binarios para todo el vídeo o para pequeñas porciones del vídeo resultado de la división de este en celdillas temporales y espaciales. Podemos apreciar que LBP funciona bastante bien en este dataset y el descriptor de colores LAB no ya que todos los vídeos tienen una distribución de colores bastante parecida y se hace difícil la diferenciación entre vídeos gracias sólo a esta descripción. El descriptor de líneas y el descriptor basado en semejanza de vídeos tienen unos resultados parecidos y bastante lejanos de los mejores del descriptor G-HOG3D. El volumen para los vídeos en el descriptor de semejanza de vídeos será de 12 píxeles horizontales, 16 verticales y 12 temporales.

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
LBP usando celdillas	0.7963	0.8427	0.9560	0.6092	0.6796	0.8763	0.7934
LBP sin usar celdillas	0.7903	0.8161	0.6947	0.7361	0.6562	0.8632	0.7594
Descriptor de colores LAB	0.2857	0.2769	0.3919	0	0	0	0.1591
Líneas	0,5385	0,5051	0,6204	0,4375	0,4815	0,6438	0,5378
G-HOG3D	0.9322	**0.9167**	0.7619	0.8587	**0.8830**	0.8889	0.8736
G-HOG3D con detección de acción	**0.9868**	0.8812	**0.8807**	**0.8734**	0.8269	**0.9057**	**0.8925**
Resta de vídeos en escala pequeña	0.6600	0.4100	0.4580	0.4792	0.3300	0.4580	0.4600

Tabla 2 Comparativa entre G-HOG3D con los demás descriptores utilizados.

8.1.5 Experimentos con adición del flujo óptico en el 'KTH dataset'

En este experimento se ha utilizado la aproximación del flujo óptico descrita en el apartado 2.2.2 y se han calculado los gradientes de este flujo del mismo modo que hacemos con los histogramas de gradientes orientados (apartado 2.2). Se han calculado por tanto los descriptores G-HOF3D, G-HOG3D y además se ha formado otro descriptor concatenando los vectores de ambos descriptores. Además se ha utilizado también la superposición como se ha comentado en el apartado 2.1.3, en vez de calcular celdillas una detrás de otra se han colocado de forma que las celdillas tengan unas zonas común de intersección. La configuración de las siguientes pruebas será: 4 celdillas espaciales y temporales, 8 bins, uso de la mediana como límite para la aceptación de los módulos de cada píxel y 48 píxeles de tamaño en altura. El KTH dataset no tiene fondo por lo que en este caso los resultados del G-HOF3D y G-HOG3D son muy parecidos ya que sólo importa la acción y no el escenario. El hecho de aumentar la información del descriptor concatenando el G-HOF3D al G-HOG3D resulta en una mejora de los resultados y el factor de superposición no los mejora.

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
G-HOG3D	**0,9371**	0,8125	0,7778	0,7708	0,6250	0,8889	0,8020
G-HOF3D	0,8042	0,7093	0,7500	0,6944	0,5208	0,8403	0,7198
G-HOG3D Superposición	0,8951	0,7708	0,8542	0,6806	**0,7292**	0,7778	0,7847
G-HOF3D Superposición	0,8604	0,6597	0,7361	0,7222	0,5764	0,8819	0,7394
G-HOG3D G-HOF	0,9301	**0,8750**	0,7917	0,7917	0,5972	**0,9306**	**0,8193**
G-HOG3D G-HOF3D Superposición	0,9231	0,7431	**0,8681**	**0,8056**	0,6319	**0,9306**	0,8170

Tabla 3 Comparativa entre G-HOG3D y G-HOG3D con el uso de superposición

Se hace un nuevo experimento con el añadido de nuevos vídeos de entrenamiento, los vídeos añadidos serán los mismos de entrenamiento actuales pero se les da la vuelta para estar de forma inversa, es decir, si antes teníamos a una persona andando de derecha a izquierda ahora la tendremos andando de izquierda a derecha pero el vídeo sigue siendo el mismo. Así disponemos de un dataset de entrenamiento el doble de amplio lo que debería de dar mejores resultados pese a lo que podemos ver en la siguiente tabla 4 en comparación con la anterior. Con esto se consigue que la concatenación de ambos descriptores no sea beneficiosa si no perjudicial.

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
G-HOG3D	0,8462	**0,8264**	0,7708	**0,8125**	0,6181	**0,9028**	**0,7961**
G-HOF3D	0,7278	0,6755	**0,8780**	0,5464	**0,7327**	0,8231	0,7305
G-HOG3D Superposición	**0,9161**	0,7986	0,7916	0,6111	0,6250	0,7847	0,7545
G-HOF3D Superposición	0,8881	0,6736	0,7431	0,7153	0,5764	0,8958	0,7487
G-HOG3D G-HOF	0,7832	0,6736	0,7361	0,7361	0,5417	0,8403	0,7185
G-HOG3D G-HOF3D Superposición	0,8462	0,6667	0,7431	0,7361	0,6181	0,8611	0,7452

Tabla 4 Comparativa entre G-HOG3D y G-HOG3D con el uso de superposición y con el añadido de vídeos invertidos en el entrenamiento.

8.1.6 Comparación con el estado del arte [11]

Una vez se ha realizado la sintonización de parámetros ahora nos comparamos con la literatura actual. La configuración de parámetros utilizada será la de 4 celdillas espaciales y temporales, 8 'bins' angulares y tamaño de la imagen de 48 píxeles en altura. La comparación se hace respecto a los siguientes datasets (ver figura 23): KTF, UCF sports[22], Olympic Sports[22] y HMDB[23]. En las tablas 5 a 8 se ha representado los resultados del mejor descriptor local para cada uno de los datasets y los resultados de los descriptores globales (G-HOG3D y G-HOF3D). En el KTH dataset se obtienen los peores resultados. El KTH dataset es el más sencillo y pequeño de los dataset, los datasets medianos serían el UFC y Olympic sports donde en ellos se obtienen resultados similares a los descriptores locales. Finalmente en el dataset más extenso y complicado se obtienen unos resultados que casi doblan a los resultados provenientes de los descriptores locales.

Figura 23 Diferentes datasets para realizar la comparación. Figura extraída de [11]

	LOCAL	G-HOG3D	G-HOF3D
UCF dataset	**62.00**	60.51	43.05

Tabla 5. Resultados para el UCF dataset. Resultados extraídos de [11]

	LOCAL	G-HOG3D	G-HOF3D
KTH dataset	**90.00**	80.20	71.98

Tabla 6. Resultados para el KTH dataset. Resultados extraídos de [11]

	LOCAL	G-HOG3D	G-HOF3D
Olympic Sports Dataset	85.60	**86.00**	43.05

Tabla 7. Resultados para el Olympic sports dataset. .Resultados extraídos de [11].

	LOCAL	G-HOG3D	G-HOF3D
HMDB dataset	22.83	**42.01**	15.25

Tabla 8. Resultados para el HMDB dataset. Resultados extraídos de [11].

8.2. Experimentos para el detector local

8.2.1 Experimentos realizados con el 'KTH dataset'. Sintonización de parámetros

Se realiza un primer experimento (tabla 9) para ver la variación respecto al cambio en el número de palabras en el que dividimos nuestros puntos clave. Un aumento de palabras mejora los resultados hasta llegar a un punto donde se estabiliza. Este punto llega muy pronto respecto a lo esperado debido a que el dataset KTH es muy simple y no es posible dividirlo en más número de palabras.

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
50 palabras	0.0857	0.0278	0.0556	0.3889	0.4440	0.0833	0.1810
200 palabras	0.4571	0.6111	0.7500	0.4167	0.500	**0.8333**	0.5947
300 palabras	0.7330	0.6842	0.7297	0.4359	0.6552	0.6429	0.6469
500 palabras	**0.7812**	**0.7941**	0.7949	0.4651	0.7037	0.6750	**0.7023**
750 palabras	0.6667	0.6970	0.8056	0.4250	0.7143	0.5952	0.6506
1000 palabras	0.7568	0.7586	0.7800	0.4474	0.6923	0.6042	0.6738
2000 palabras	0.5745	0.7600	**0.8260**	**0.4872**	**0.7700**	0.6900	0.6864

Tabla 9. Resultados para el HMDB dataset. Resultados extraídos de [11]

Se realiza un experimento (tabla 10) para comprobar que un aumento del tamaño de la imagen aumenta los resultados al detectarse más puntos característicos

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
Tamaño 48	**0.7812**	0.7941	0.7949	**0.4651**	0.7037	0.6750	0.7023
Tamaño 64	0.7317	**0.8919**	**0.8919**	0.4615	**0.7273**	**0.7500**	**0.7432**

Tabla 10. Resultados para el HMDB dataset. Resultados extraídos de [11]

Se realiza un experimento (tabla 11) para comprobar si funciona mejor sumar palabras dentro de un contexto mediante probabilidades o sin ellas.

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
Con probabilidad	**0.4571**	**0.6111**	**0.7500**	**0.4167**	**0.5000**	**0.8333**	**0.5947**
Sin probabilidad	**0.4571**	**0.6111**	**0.7500**	0.3889	**0.5000**	0.8056	0.5800

Tabla 11. Resultados para el HMDB dataset. Resultados extraídos de [11]

Se realiza un experimento (tabla 12) para comprobar que el uso de contextos no mejora nuestros resultados en este dataset.

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
1 contexto	**0.7330**	**0.6842**	**0.7297**	**0.4359**	**0.6552**	**0.6429**	**0.6469**
8 contextos (2 en 'x', 2 en 'y', 2 en 't')	0.4857	0.4444	0.6667	0.3889	0.4440	0.6944	0.5208

Tabla 12. Resultados para el HMDB dataset. Resultados extraídos de [11]

9. Conclusiones

Los descriptores globales han resultado funcionar mejor en datasets grandes y de muchas clases. En el KTH dataset, que es el más sencillo y a la vez menos extenso, ha dado los peores resultados respecto a los descriptores locales aunque aun así de han alcanzado cotas de más del 80% de reconocimiento.

Las principales conclusiones que se extraen de nuestros experimentos sobre los descriptores globales son:

- El descriptor G-HOG-3D da mejores resultados que el descriptor G-HOF-3D.
- La unión de los descriptores G-HOG-3D (apartado 2.1) y G-HOF-3D (apartado 2.2) da los mejores resultados.
- Una mascara derivativa centrada da mejor resultado que una mascara no centrada. ([-1,0,1] funciona mejor que [-1,1])
- A mayor número de celdillas mejores resultados hasta llegar a saturar en torno a 6-8 celdillas.
- A mayor número de 'bins' mejores resultados hasta llegar a saturar (8-12 'bins'). La aparición de esta saturación se debe a que si por ejemplo nos vamos al extremo de que un celdilla = 1 pixel y el tamaño del 'bin' = 1º lo que ocurre es que analizamos el vídeo completamente pixel a pixel. De esta manera no agrupamos en celdillas y es imposible que dos vídeos sean parecidos.
- La normalización de celdillas aporta mejores resultados. Acotar los picos de los gradientes también mejora los resultados.
- Detectar la ventana de movimiento da mejores resultados en el KTH dataset. Este dataset tiene un fondo plano y apenas sin movimiento de cámara, quizás en datasets más complejos donde la cámara entra en acción y el fondo está relacionado con la acción esto no es así. Sería interesante analizar esto como trabajo futuro.
- La reducción de imágenes mejora los resultados al eliminar detalles innecesarios que provocan gradientes poco o nada relevantes.
- A mayor número de vídeos de entrenamiento mejores resultados sin saturar.

Las principales conclusiones que se extraen de nuestros experimentos sobre los descriptores locales son:

- A mayor número de palabras mejores resultados. Se llega a un punto donde aumentar palabras no mejora los resultados si no que los mantiene o los empeora. Esto es debido a que los vídeos de entrenamiento tienen un número finito de palabras que es imposible de saber a priori pero si se puede experimentalmente.
- Al contrario que en descriptores globales a mayor tamaño de las imágenes mejores resultados. Esto es debido a que se localizan más puntos característicos (ver figura 4) y la descripción es más completa.

10. Trabajo futuro

De este proyecto final de carrera se extraen algunas posibles áreas para una investigación futura:

- Comprobar los resultados que da el descriptor global en datasets todavía más amplios. Los resultados dados en el dataset HMDB que es el más amplio hasta fecha han sido muy buenos y puede ser que este descriptor sea una de las mejores soluciones en este problema.
- Comprobar los resultados que el descriptor global aporta en otras áreas de reconocimiento como el reconocimiento de escenas y objetos.
- Seguir testeando el detector y descriptor local en datasets más complicados y analizando el fotograma a tamaño completo en vez de a tamaño reducido. Previsiblemente, los resultados para descriptores locales mejorarán al extraer más puntos, pero permanecerán igual para descriptores globales.
- Testear el descriptor global con posibles mejoras como pueden ser las impuestas en el descriptor utilizado para el sift (descritas en el apartado 3.2.2) u otras diferentes.
- Creación de un dataset de entrenamiento de millones de vídeos de igual manera que se ha hecho en [16] con imágenes y realizar la clasificación de vídeos mediante la distancia de estos a cada uno de los vídeos del dataset de entrenamiento.

Bibliografía

[1] Horn, B.K.R, and Schunck, B.G. 1981. Determining optical flow, Artificial Intelligence 17: 185-204.-

[2] Secrets of optical flow estimation and their principles; Sun, D., Roth, S., and Black, M. J., IEEE Conf. on Computer Vision and Pattern Recog., CVPR, June 2010.

[3] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Phys. D*, 60(1-4):259–268, 1992.

[4] M. J. Black and P. Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *CVIU*, 63:75–104, 1996.

[5] Y. Li and S. Osher. A new median formula with applications to PDEmbased denoising. *Commun. Math. Sci.*, 7(3):741–753, 2009

[6] P. Sand and S. Teller. Particle video: Long-range motion estimation using point trajectories. *IJCV*, 80(1):72–91, 2008.

[7] Zhao G & Pietikäinen M (2007) Dynamic texture recognition using local binary patterns with an application to facial expressions. IEEE Transactions on Pattern Analysis and Machine Intelligence, 29(6):915-928

[8] Canny Edge Detection 09gr820 March 23, 2009.

[9]*A*honen T, Matas J, He C & Pietikäinen M (2009) Rotation invariant image description with local binary pattern histogram fourier features. Proc. 16th Scandinavian Conference on Image Analysis (SCIA 2009), Oslo, Norway.

[10] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. A practical guide to support vector classification.Technical report, Department of Computer Science and Information Engineering, National Taiwan University, Taipei, 2003. http://www.csie.ntu.edu.tw/_cjlin/libsvm/.

[11] J. Civera and Alejo Concha, Global descriptors for video, enviado a BMVC 2012. En proceso de revisión.

[12] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: Computer Vision and Pattern Recognition, San Diego, CA, June 20–25, 2005.

[13] N. Dalal, B. Triggs, C. Schmid, Human detection using oriented histograms of flow and appearance, in: European Conference on Computer Vision, Graz, Austria, May 7–13, 2006.

[14] Marcin Marsza and Ivan Laptev and Cordelia Schmid, "Actions in Context", "IEEE Conference on Computer Vision \& Pattern Recognition","2009

[15] C. Sch¨uldt, I. Laptev, and B. Caputo. Recognizing human actions: A local SVM approach. In ICPR, 2004.

[16] A. Torralba, R. Fergus, and W. Freeman. 80 million tiny images: Alarge data set for nonparametric object and scene recognition. PAMI,30(11):1958–1970, November 2008.

[17] Heng Wang, Muhammad Muneeb Ullah, Alexander Klaser, Ivan Laptev,Cordelia Schmid, Evaluation of local spatio-temporal features for action recognition, in: Proceedings of the British Machine Vision Conference(BMVC’ 09), London, United Kingdom

[18] D. G. Lowe. Distinctive image features from scale-invariantn keypoints. IJCV, 60(2):91.110, 2004.

[19] Flitton, G., Breckon, T., "Object Recognition using 3D SIFT in Complex CT Volumes". Proceedings of the British Machine Vision Conference. págs. 11.1-12. (2010)

[20] Paul Scovanner, Saad Ali, Mubarak Shah, A 3-dimensional SIFT descriptor and its application to action recognition, in: Proceedings of the International Conference on Multimedia (MultiMedia’ 07), Augsburg, Germany, September 2007, pp. 357‑360.

[21] A. Gilbert, J. Illingworth, R. Bowden, Fast realistic multi-action recognition using mined dense spatio-temporal features, in: Proc. Int. Conference on Computer Vision (ICCV), 2009.

[22] J. Niebles, C.W. Chen, and L. Fei-Fei. Modeling temporal structure of decomposable motion segments for activity classification. Computer Vision–ECCV 2010, pages 392–405, 2010.

[23] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. Hmdb: A large video database for human motion recognition. In 2011 IEEE International Conference on Computer Vision, pages 2556–2563. IEEE, 2011.