Anexo A

Flujo óptico de Horn [1]

Intuitivamente 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.

Una imagen en el instante 't' la podemos parametrizar como:

$$I(x, y, t)$$

Donde 'x' e 'y' son las coordenadas de textura y 't' es el tiempo o fotograma equivalente.

En un instante siguiente el pixel correspondiente $I(x, y, t)$ sale movido un valor (dx, dy, dt), por tanto el pixel en la nueva imagen será:

$$I(x + dx, y + dy, t + dt)$$

Estos dos instantes los podemos correlacionar gracias a la expansión de Taylor de primer orden:

$$I(x + dx, y + dy, t + dt) = I(x, y, t) + \frac{\partial I}{\partial x}\ dx + \frac{\partial I}{\partial y}\ dy + \frac{\partial I}{\partial t}\ dt$$

La idea consiste en que el objeto que estamos buscando tendrá la misma energía en un instante como en el otro, es decir, la misma luminosidad, el valor de cada pixel se mantendrá ya que sólo cambiará su posición, que es lo que queremos calcular, la cantidad de píxeles que se ha movido, por tanto podemos igualar $I(x + dx, y + dy, t + dt) = I(x, y, t)$ y el término restante deberá ser igual a cero.

$$\frac{\partial I}{\partial x}\ dx + \frac{\partial I}{\partial y}\ dy + \frac{\partial I}{\partial t}\ dt = 0$$

Donde:

$$\frac{\partial I}{\partial x} = I_x\ ;\ \frac{\partial I}{\partial y} = I_y\ ;\ \frac{\partial I}{\partial t} = I_t$$

$$I_x\ dx + I_y\ dy + I_t\ dt = 0$$

Si dividimos para 'dt':

$$I_x\ u + I_y\ v + I_t\ = 0$$

$$u = \frac{dx}{dt}\ ;\ v = \frac{dy}{dt}$$

Esa será nuestra primera ecuación, pero como vemos tenemos dos incógnitas, 'u' y 'v', que serán el flujo óptico horizontal y vertical respectivamente por lo que necesitaremos una segunda ecuación.

Esta segunda limitación o ecuación viene del hecho de que el objeto está sujeto a un movimiento como solido rígido o a una pequeña deformación, lo que provoca un ligero cambio en estos flujos y por tanto la minimización del cambio de estos en su entorno supondrá esta información requerida:

$$minimización \rightarrow \frac{\partial \mathrm{u}^2}{\partial \mathrm{x}} + \frac{\partial \mathrm{u}^2}{\partial \mathrm{y}} + \frac{\partial \mathrm{v}^2}{\partial \mathrm{x}} + \frac{\partial \mathrm{v}^2}{\partial \mathrm{y}} = \mathrm{u}_x{}^2 + \mathrm{u}_y{}^2 + \mathrm{v}_x{}^2 + \mathrm{v}_y{}^2$$

De esta manera podemos reducir nuestro problema a un problema de minimización en ambas ecuaciones, ya que la primera debe de ser cero y la segunda lo más pequeña posible. Utilizando α como un parámetro de minimización que relaciona ambos funcionales a minimizar, llegamos al siguiente funcional a minimizar:

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

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

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

Podemos ver que tenemos un funcional a minimizar del estilo de las ecuaciones de Euler Lagrange y por tanto tenemos su solución, podemos ver como se llega a ella en el Anexo J. De este modo, dado el siguiente funcional:

$$S(q) = \int_{\mathrm{a}}^{b} L\big(t, q(t), q'(t)\big) dt$$

Su solución es:

$$\frac{\partial \mathrm{L}\left(\mathrm{t}, \mathrm{q}(\mathrm{t}), \mathrm{q}'(\mathrm{t})\right)}{\partial \mathrm{x}_i} - \frac{\mathrm{d}}{\partial \mathrm{t}} \frac{\partial \mathrm{L}\left(\mathrm{t}, \mathrm{q}(\mathrm{t}), \mathrm{q}'(\mathrm{t})\right)}{\partial \mathrm{v}_i} = 0 \ \text{ para i} = 1, \dots, \mathrm{n}.$$

Donde:

$$\mathrm{n} = \text{número de dimensiones del espacio.}$$

$$\mathrm{x} = \mathrm{q}(\mathrm{t}); \ \ v = q'(t)$$

En nuestro caso tenemos un sistema de ecuaciones de Euler de dos funciones a minimizar, $(f_1, f_2) = (u, v)$ y dos dimensiones para la variable respecto a las cuales debemos minimizar, $x_i = (x, y)$, por tanto:

$$\frac{\partial \mathrm{L}}{\partial \mathrm{f}_1} - \sum_{i=1}^{n} \frac{\partial}{\partial \mathrm{x}_i} \frac{\partial \mathrm{L}}{\partial \mathrm{f}_{1,i}} = 0$$

$$\frac{\partial \mathrm{L}}{\partial \mathrm{f}_2} - \sum_{i=1}^{n} \frac{\partial}{\partial \mathrm{x}_i} \frac{\partial \mathrm{L}}{\partial \mathrm{f}_{2,i}} = 0$$

Donde:

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

$$x_i = (x, y)$$

$$(f_1, f_2) = (u, v)$$

$$(f_{1,i}, f_{2,i}) = (f_{1,x}, f_{1,y}; f_{2,x}, f_{2,y}) = \left(\mathrm{u}_x, \mathrm{u}_y; \mathrm{v}_x, \mathrm{v}_y\right)$$

Introduciendo estas variables en las ecuaciones (1) y (2) se llega a la siguiente solución:

$$\mathrm{I}_x{}^2 u + \mathrm{I}_x \mathrm{I}_y v = \alpha\nabla^2 u - \mathrm{I}_x \mathrm{I}_t$$
$$\mathrm{I}_x \mathrm{I}_y u + \mathrm{I}_y{}^2 v = \alpha\nabla^2 u - \mathrm{I}_y \mathrm{I}_t$$

Para poder resolverla discretamente se siguen 3 aproximaciones vistas en [1]. Como vemos la media se calcular dándole más peso a los píxeles más cercanos que a los que están en la proyección diagonal y el laplaciano se calcula haciendo el gradiente entre el pixel que estamos calculando y la media aproximada de los píxeles de alrededor.

$$\nabla^2 u = 3(\bar{u} - u)$$
$$\nabla^2 v = 3(\bar{v} - v)$$
$$\bar{u}_{i,j,k} = \frac{1}{6}\left\{u_{i-1,j,k} + u_{i,j+1,k} + u_{i+1,j,k} + u_{i,j-1,k}\right\} + \frac{1}{12}\left\{u_{i-1,j-1,k} + u_{i-1,j+1,k} + u_{i+1,j+1,k} + u_{i+1,j-1,k}\right\}$$
$$\bar{v}_{i,j,k} = \frac{1}{6}\left\{v_{i-1,j,k} + v_{i,j+1,k} + v_{i+1,j,k} + v_{i,j-1,k}\right\} + \frac{1}{12}\left\{v_{i-1,j-1,k} + v_{i-1,j+1,k} + v_{i+1,j+1,k} + v_{i+1,j-1,k}\right\}$$

El gradiente para las imágenes se calcula como la media ponderada entre los 4 gradientes calculados para dos píxeles cada dos fotogramas consecutivos:

$$I_x = 0.25\,(I_{i+1,j,k} - I_{i,j,k} + I_{i+1,j+1,k} - I_{i+1,j,k} + I_{i+1j,k+1} - I_{i,j,k+1} + I_{i+1,j+1,k+1} - I_{i,j+1,k+1})$$

$$I_y = 0.25\,(I_{i,j+1,k} - I_{i+1,j+1,k} + I_{i+1,j,k} - I_{i,j+1,k+1} + I_{i,j,k+1} - I_{i+1,j+1,k+1} + I_{i+1,j,k+1} - I_{i,j,k})$$

$$I_t = 0.25\,(I_{i,j,k+1} - I_{i,j,k} + I_{i+1,j,k+1} - I_{i+1,j,k} + I_{i,j+1,k} + 1 - I_{i,j+1,k} + I_{i+1,j+1,k+1} - I_{i+1,j+1,k})$$

Introducimos estas 3 aproximaciones en nuestras ecuaciones:

$$\mathrm{I}_x{}^2 u + \mathrm{I}_x \mathrm{I}_y v = \alpha\nabla^2 u - \mathrm{I}_x \mathrm{I}_t$$
$$\mathrm{I}_x \mathrm{I}_y u + \mathrm{I}_y{}^2 v = \alpha\nabla^2 u - \mathrm{I}_y \mathrm{I}_t$$

Y operando llegamos a:

$$u = \frac{\left(3\alpha^2 + \mathrm{I}_y{}^2\right)\bar{u} - \mathrm{I}_x\mathrm{I}_y\bar{v} - \mathrm{I}_x\mathrm{I}_t}{3\alpha^2 + \mathrm{I}_x{}^2 + \mathrm{I}_y{}^2}$$

$$v = \frac{\left(3\alpha^2 + \mathrm{I}_x{}^2\right)\bar{v} - \mathrm{I}_x\mathrm{I}_y\bar{u} - \mathrm{I}_y\mathrm{I}_t}{3\alpha^2 + \mathrm{I}_x{}^2 + \mathrm{I}_y{}^2}$$

Donde si hacemos un cambio de variables $\mathrm{u} = \mathrm{u}^{\mathrm{k+1}}$ y $\bar{\mathrm{u}} = \bar{\mathrm{u}}^{\,\mathrm{k}}$ y operamos llegamos finalmente a la solución para el cálculo del flujo óptico, k será el número de iteraciones que queremos utilizar para llegar a la minimización:

$$u^{k+1} = \bar{u} - \frac{\mathrm{I}_x\left(\mathrm{I}_x\bar{u}^k + \mathrm{I}_y\bar{v}^k + \mathrm{I}_t\right)}{3\alpha^2 + \mathrm{I}_x{}^2 + \mathrm{I}_y{}^2}$$

$$v^{k+1} = \bar{v} - \frac{\mathrm{I}_y\left(\mathrm{I}_x\bar{u}^k + \mathrm{I}_y\bar{v}^k + \mathrm{I}_t\right)}{3\alpha^2 + \mathrm{I}_x{}^2 + \mathrm{I}_y{}^2}$$

Anexo B

Flujo óptico de Roth y Black [2]

Partimos de la función a minimizar, similar a la explicada en el Anexo A pero con la principal diferencia de la con la adición de funciones de penalización ρD y ρS :

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

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

Consiste en encontrar los valores del flujo óptico 'u' y 'v' que minimizan la función anterior, 'u' y 'v' son los 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 termino).

Las funciones de penalización tienen como función el formalizar los errores que además devuelve siempre un número positivo y no surge el fenómeno de cancelación de errores. Funciones de penalización utilizadas:

-Cuadrática o HS. $\rho(x) = x^2$

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

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

El ruido es un grave problema para nuestro algoritmo ya que puede ser flujo y pensar que hay flujo donde no lo hay, por tanto se sigue el método visto en [3] para limpiar nuestras imágenes y conseguir el efecto de la figura 25:

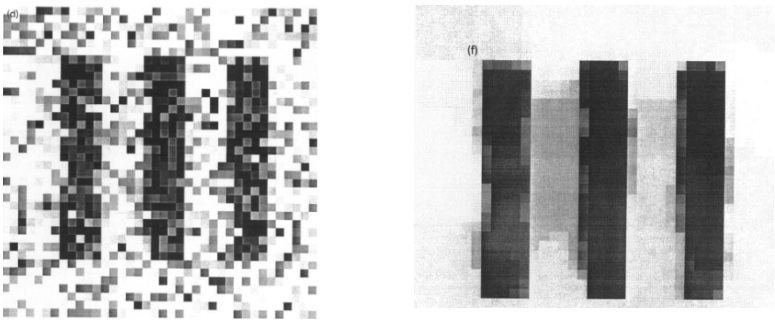

Figura 25. Comparación de una imagen con ruido y la misma con el ruido eliminado.

Primero se descompone la imagen en la imagen deseada y el ruido indeseado que hay en ella:

$$u_0(x,y) = u(x,y) + n(x,y)$$

$$u_0 = imagen$$

$$u = imagen \sin ruido$$

$$n = ruido$$

La idea principal se fundamenta en que un pixel rodeado de 4 píxeles iguales entre si y diferentes del pixel central será un espurio. Para detectarlo matemáticamente:

La derivada en el dominio horizontal se define como: hx = [-1, 0, 1]. Si realizamos la derivada de la derivada obtenemos: hxx = [-1, 0, 2, 0, -1] y $hyy=hxx^T$. Si componemos hyy con hxx surge una máscara en forma de cruz respecto al pixel que analizamos, a este pixel se le da valor 4 y a los píxeles que están a su alrededor se les da peso -1. Se multiplica cada valor por su peso y se hace la suma, obtenemos un valor que de ser alto significará que estamos ante un espurio. Por tanto el objetivo será minimizar valores altos tras la aplicación de esta mascara por toda la imagen, por tanto:

$$minimizar \int_\Omega \sqrt{(u_{xx} + u_{yy})^2}\, dxdy$$

Sujeto a 2 restricciones, 1 que dice que la media de los valores de todos los píxeles en ambas imágenes debe mantenerse y otra que aproxima la variación con una sigma:

La primera restricción nos habla de que la media del ruido será cero y que su desviación es sigma y es dada.

$$\int_\Omega u = \int_\Omega u_0$$

$$\int_{\Omega} (u - u_0)^2 = \sigma^2$$

Sin embargo esto no reportó resultados buenos y se cambió la función a minimizar, con la nueva en vez de fijarnos en las derivadas segundas nos fijamos en las primeras, la mascara pasa a ser cero el pixel central, 1 en los píxeles de abajo y a la derecha y -1 en los píxeles de arriba y a la izquierda, con esto se mide si hay un cambio de textura horizontal y vertical, lo que produce que si ambos se cumplen el cambio de textura sea sólo en el punto ya que un punto (el espurio) es la unión de dos líneas (líneas de cambio de textura horizontal y vertical, si solo tenemos una de las dos no estamos ante un espurio aunque podría darse el caso de tener una línea de espurios). Por tanto la minimización se llevará a cabo en:

$$minimizar \int_{\Omega} \sqrt{{u_x}^2 + {u_y}^2}\, dxdy$$

Tenemos un funcional a minimizar que según el método de los multiplicadores de Lagrange es:

$$L = \sqrt{{u_x}^2 + {u_y}^2} + \lambda_1(u - u_0) + \lambda_2 \frac{1}{2}(u - u_0)^2$$

Para su resolución se utilizan la ecuación de LaGrange (explicada en el anexo J):

$$\frac{\partial L}{\partial u} - \frac{\partial\left(\frac{\partial L}{\partial u_x}\right)}{\partial x} - \frac{\partial\left(\frac{\partial L}{\partial u_y}\right)}{\partial y} = 0$$

Con ellas llegamos a:

$$\frac{\partial\left(\frac{u_x}{\sqrt{{u_x}^2 + {u_y}^2}}\right)}{\partial x} + \frac{\partial\left(\frac{u_y}{\sqrt{{u_x}^2 + {u_y}^2}}\right)}{\partial y} - \lambda_1 - \lambda_2(u - u_0) = 0$$

La ecuación anterior no se cumplirá en un primer calculo por lo que la igualamos a una variable que denotará su valor y se realizarán iteraciones hasta disminuirla al máximo posible. Se utilizará el método del gradiente descendente, esta ecuación se iguala a 'u_t' y convergerá a cero con el tiempo. 'u_t' será el cambio temporal entre imágenes.

$$u_t = \frac{\partial\left(\frac{u_x}{\sqrt{{u_x}^2 + {u_y}^2}}\right)}{\partial x} + \frac{\partial\left(\frac{u_y}{\sqrt{{u_x}^2 + {u_y}^2}}\right)}{\partial y} - \lambda(u - u_0)$$

Restricción $\frac{\partial u}{\partial n} = 0 \quad en\ la\ frontera$

$$\mathrm{u(x, y, 0)}\ es\ dado$$

Nótese la no utilización de la primera restricción debido a que esta incluida en nuestro procedimiento si la media de u(x, y, 0) es la misma que la de u_0.

Gracias a esta convergencia a cero podemos calcular λ, simplemente multiplicando a ambos lados por $(u-u_0)$, integrando en ambos lados utilizando la integración por partes y las restricciones impuestas y sabiendo que u_t converge a 0:

$$\lambda(t)(u-u_0)^2 = (u-u_0) * \left[\frac{\partial \left(\frac{u_x}{\sqrt{{u_x}^2 + {u_y}^2}} \right)}{\partial x} + \frac{\partial \left(\frac{u_y}{\sqrt{{u_x}^2 + {u_y}^2}} \right)}{\partial y} \right]$$

$$\int \lambda(t)(u-u_0)^2 \, dxdy = \int (u-u_0) * \left[\frac{\partial \left(\frac{u_x}{\sqrt{{u_x}^2 + {u_y}^2}} \right)}{\partial x} + \frac{\partial \left(\frac{u_y}{\sqrt{{u_x}^2 + {u_y}^2}} \right)}{\partial y} \right] dxdy$$

$$\lambda(t) = \frac{1}{2\sigma^2} \int (u-u_0) * \left[\frac{\partial \left(\frac{u_x}{\sqrt{{u_x}^2 + {u_y}^2}} \right)}{\partial x} + \frac{\partial \left(\frac{u_y}{\sqrt{{u_x}^2 + {u_y}^2}} \right)}{\partial y} \right] dxdy$$

Para realizar la integral restante se integra por partes como hemos dicho utilizando la siguiente fórmula, el término debido a la superficie (teorema de gauss) se elimina por razones físicas:

$$\int_\Omega \frac{\partial u}{\partial x_i} v dx = -\int_\Omega \frac{\partial v}{\partial x_i} u dx$$

$$\lambda(t) = -\frac{1}{2\sigma^2} \int \left[\sqrt{{u_x}^2 + {u_y}^2} - \frac{u_x {u_0}_x}{\sqrt{{u_x}^2 + {u_y}^2}} - \frac{u_y {u_0}_y}{\sqrt{{u_x}^2 + {u_y}^2}} \right] dxdy$$

Discretizamos el modelo en las dos direcciones:

$$x_i = ih; y_i = jh; \;\; j = 0,1 \ldots, N \quad con\ Nh = 1$$

$$t_n = n\Delta t; \;\; n = 0,1 \ldots.$$

$$u_{ij}^n = u(x_i, y_j, t_n)$$

$$u_{ij}^{0} = u_0\left(x_i, y_j, t_n\right) + \varphi(ih, jh)\sigma$$

Donde φ es la aproximación de la diferencia entre una imagen y su homóloga sin ruido, por tanto tiene media cero para cumplir con la restricción 1.

El cambio temporal entre dos imágenes será el calculado 'u_t' debido a la convergencia de este desde la imagen original a la reparada.

$$u_{ij}^{n+1} = u_{ij}^{n} + \frac{\Delta t}{h}\left[\Delta_{-}^{x}\left(\frac{\Delta_{-}^{x}u_{ij}^{n}}{\left(\left(\Delta_{-}^{x}u_{ij}^{n}\right)^2 + \left(m\left(\Delta_{-}^{y}u_{ij}^{n}, \Delta_{+}^{y}u_{ij}^{n}\right)\right)^2\right)^{1/2}}\right) + \Delta_{-}^{y}\left(\frac{\Delta_{-}^{y}u_{ij}^{n}}{\left(\left(\Delta_{-}^{y}u_{ij}^{n}\right)^2 + \left(m\left(\Delta_{-}^{x}u_{ij}^{n}, \Delta_{+}^{x}u_{ij}^{n}\right)\right)^2\right)^{1/2}}\right)\right] - \Delta t\lambda^{n}\left(u_{ij}^{n} - u_0(ih, jh)\right) \qquad para\ \ i, j = 1,2, \dots, N$$

$$u_{0j}^{n} = u_{1j}^{n}\ ;\ u_{Nj}^{n} = u_{N-1j}^{n}\ ;\ u_{i0}^{n} = u_{i1}^{n}\ ;\ u_{iN}^{n} = u_{iN-1}^{n}\ \ debidas\ a\ las\ condiciones\ de\ frontera$$

Para entender la nomenclatura:

$$\Delta_{-}^{x}u_{ij} = -\left(u_{i-1,j} - u_{ij}\right)$$

$$\Delta_{+}^{x}u_{ij} = +\left(u_{i+1,j} - u_{ij}\right)$$

$$Lo\ mismo\ para\ \Delta_{-}^{y}u_{ij}\ y\ \Delta_{+}^{y}u_{ij}$$

$$m(\,a\,, b\,) = mínimo\ modulo\ entre\ a\ y\ b$$

Finalmente se pasa λ al dominio discreto:

$$\lambda^{n} = -\frac{h}{2\sigma^2}\left[\sqrt{(\Delta_{+}^{x}u_{ij})^2 + (\Delta_{+}^{y}u_{ij})^2} - \frac{(\Delta_{+}^{x}u_{ij}^{0})(\Delta_{+}^{x}u_{ij}^{n})}{\sqrt{(\Delta_{+}^{x}u_{ij})^2 + (\Delta_{+}^{y}u_{ij})^2}} - \frac{(\Delta_{+}^{y}u_{ij}^{0})(\Delta_{+}^{y}u_{ij}^{n})}{\sqrt{(\Delta_{+}^{x}u_{ij})^2 + (\Delta_{+}^{y}u_{ij})^2}}\right]$$

Se impone una restricción para cada paso por estabilidad:

$$\frac{\Delta t}{h^2} \leq c$$

Una vez el ruido ya no es un problema, uno de los principales inconvenientes surge cuando queremos calcular flujo que se desplaza más de un pixel entre fotogramas. Para abordar este problema se parte de la solución mostrada por [4].

En primer lugar explicamos como se lleva a cabo la minimización, se utilizará un modelo similar al de los gradientes descendentes de Newton Rapson con la adición de los parámetros 'w' y T(s). Este modelo se utilizará igual para calcular 'v', 'u' y los coeficientes 'a' (que en realidad son , como veremos más abajo, los coeficientes que aproximarán el flujo óptico con una función lineal) con la diferencia de que para el cálculo del gradiente del funcional (E) se aplicará sólo la primera parte de éste (E_D) para los valores 'a' ya que es donde están estos coeficientes implícitos y el funcional entero (E) para el flujo óptico en si ('u' y 'v'):

$$u^{n+1} = u^n - w(u)\frac{1}{T(u)}\frac{\partial E}{\partial u} \quad Para\ todo\ u$$

$$v^{n+1} = v^n - w(v)\frac{1}{T(v)}\frac{\partial E}{\partial v}\ Para\ todo\ v$$

$$a_i^{n+1} = a_i^n - w\frac{1}{T(a_i)}\frac{\partial E_D}{\partial a_i}\ \ Para\ todo\ a$$

Donde 0 < w < 2 es un parámetro de sobrerrelajación para corregir en demasía el calculo de u^{n+1}desde u^n. Con estos valores el sistema se ha mostrado que converge aunque el ratio exacto de convergencia dependerá de su valor exacto de 'w'. El cálculo de w(u) y w(v) se explicará más adelante.

T(x) es el límite superior de la derivada segunda de la función objetivo a minimizar, con esto nos aseguramos de que si hay un cambio muy grande en la derivada de la función a minimizar el cambio en está será pequeño para no calcular valores de u^n muy distintos entre uno paso y el otro, si ocurriera lo contrario pudiera pasar que el sistema no convergiera al pasarnos del valor que minimice la función por utilizar un paso demasiado elevado:

$$T(u) = \frac{\partial^2 E}{\partial u^2};\ T(v) = \frac{\partial^2 E}{\partial v^2}\ ;\ T(a) = \frac{\partial^2 E_D}{\partial a^2}$$

Se comienza la iteración con flujo u=0 y el primer cambio 'du' en el flujo es computado. El nuevo flujo dado por la suma de estos es proyectado en el siguiente nivel de una hipotética pirámide. Se realiza el mismo proceso con la imagen original y la que acabamos de calcular gracias al flujo y se computa un nuevo 'du' en este nivel, el proceso se repetirá hasta que el flujo ha sido computado en toda la resolución.

La región del flujo se describe de manera lineal con los coeficientes a como:

$$u(x, y, a) = \begin{bmatrix} u(x,y) \\ v(x,y) \end{bmatrix} = \begin{bmatrix} a_0 + a_1 x + a_2 y \\ a_3 + a_4 x + a_5 y \end{bmatrix}$$

En una primera iteración se parte como ya se ha comentado con flujo nulo, por tanto se compara la segunda imagen con las mismas posiciones en la primera imagen y se calcula el flujo du y dv, con este flujo en la siguiente iteración comparemos las posiciones de la segunda imagen (x, y) con las posiciones donde este flujo nos dice que está la primera imagen ($\hat{x}, \hat{y}$). Para relacionar unas con otras basta con restar el flujo óptico a la segunda imagen para tener las posiciones de ese mismo pixel en la primera imagen:

$$(\hat{x}, \hat{y}) = (x, y) - flujo(u, v)$$

$$(\hat{x}, \hat{y}) = (x, y) - (a_0, a_3) - A * (\hat{x} - Cx, \hat{y} - Cy)$$

$$A = \begin{bmatrix} a_1 & a_2 \\ a_4 & a_5 \end{bmatrix}$$

Donde Cx y Cy se corresponden con el centro de la región.

Calculamos ahora las derivadas de los funcionales para poder llevar a cabo el método de minimización, para ello utilizaremos la función de penalización $\rho(x, \sigma)$ de Geman-McClure por facilidad pese a no ser ninguna de las mencionadas anteriormente, el cálculo sería el mismo con cualquiera de ellas:

$$\rho(x, \sigma) = \frac{x^2}{\sigma^2 + x^2}; \quad \psi(x, \sigma) = \frac{\partial \rho}{\partial x} = \frac{2x\sigma}{(\sigma^2 + x^2)^2}$$

$$\frac{\partial E_D}{\partial a_0} = \sum I_X \lambda_D \, \psi(I_1(i,j) - I_2\big(i + u_{i,j}, j + v_{i,j}\big), \sigma_D)$$

$$\frac{\partial E_D}{\partial a_1} = \sum I_X x \, \lambda_D \psi(I_1(i,j) - I_2\big(i + u_{i,j}, j + v_{i,j}\big), \sigma_D)$$

$$\frac{\partial E_D}{\partial a_2} = \sum I_X \, y \lambda_D \psi(I_1(i,j) - I_2\big(i + u_{i,j}, j + v_{i,j}\big), \sigma_D)$$

$$\frac{\partial E_D}{\partial a_3} = \sum I_y \lambda_D \, \psi(I_1(i,j) - I_2\big(i + u_{i,j}, j + v_{i,j}\big), \sigma_D)$$

$$\frac{\partial E_D}{\partial a_4} = \sum I_y \, x \lambda_D \psi(I_1(i,j) - I_2\big(i + u_{i,j}, j + v_{i,j}\big), \sigma_D)$$

$$\frac{\partial E_D}{\partial a_5} = \sum I_y \, y \lambda_D \psi(I_1(i,j) - I_2\big(i + u_{i,j}, j + v_{i,j}\big), \sigma_D)$$

Mientras que los coeficientes T(x) se calculan como:

$$Ta_0 = \sum \lambda_D {I_X}^2 \, \rho''(x) max$$

$$Ta_1 = \sum \lambda_D {I_X}^2 x^2 \, \rho''(x) max$$

$$Ta_2 = \sum \lambda_D {I_X}^2 \, y^2 \rho''(x) max$$

$$Ta_3 = \sum \lambda_D {I_y}^2 \, \rho''(x) max$$

$$Ta_4 = \sum \lambda_D {I_y}^2 \, x^2 \rho''(x) max$$

$$Ta_5 = \sum \lambda_D {I_y}^2 \, y^2 \rho''(x) max$$

$$\rho''(x)max = \frac{2}{{\sigma_D}^2}$$

Se busca el valor máximo de $\rho''(x)$ para cumplir con la restricción de que T(x) debe de ser máxima, como se ha comentado anteriormente.

Para el cálculo de T(u), T(v) y las derivadas parciales del funcional (en este caso el funcional total, ya que el flujo óptico está definido en su totalidad en él) se calculan de un modo similar donde los resultados son:

$$T(u) = \frac{\lambda_D {I_X}^2}{{\sigma_D}^2} + \frac{4\lambda_s}{{\sigma_s}^2}$$

$$T(v) = \frac{\lambda_D {I_y}^2}{{\sigma_D}^2} + \frac{4\lambda_s}{{\sigma_s}^2}$$

El parámetro de sobrerrelajación 'w' en este caso no se calcula de forma experimental y se establece entre 0 y 2 si no que se aproxima como el valor propio máximo de la matriz Jacobiana, donde la aproximación para el valor óptimo resulta:

$$w_{opt} = \frac{2\left(1 - \sqrt{1 - \mu_{max}^2}\right)}{\mu_{max}^2}$$

$$\mu_{max} = \cos h\pi$$

$$h = \frac{1}{n+1}$$

$$n = dimensión\ matriz$$

Se demuestra que un filtrado de mediana en cada iteración de tamaño 5 en ambas direcciones da mejores resultados que no llevarlo a cabo o que realizar un filtrado de tamaño 3, 7 o un filtrado de tamaño 5 dos veces seguidas

Por ello se modifica la función objetivo para incluir este filtrado de mediana:

$$\begin{aligned} E(u,v) = \sum_{i,j} \{ & \lambda_D \rho D(I_1(i,j) - I_2(i + u_{i,j}, j + v_{i,j})) + \lambda_s [\rho S(u_{i,j} - u_{i+1,j)} + \rho S(u_{i,j} - u_{i,j+1)} \\ & + \rho S(v_{i,j} - v_{i+,j}) + \rho S(v_{i,j} - v_{i,j+1})]\} + \lambda_2 \left(\left|| \boldsymbol{u} - \hat{\boldsymbol{u}} \right||^2 + \left|| \boldsymbol{v} - \hat{\boldsymbol{v}} \right||^2 \right) \\ & + \sum_{i,j} \sum_{i',j' \epsilon\, N(i,j)} \lambda_3 (\left| v_{i,j} - \hat{v}_{i',j'} \right| + \left| u_{i,j} - \hat{u}_{i',j'} \right|) \end{aligned}$$

Se corresponde con el mismo del caso anterior con la adición de los dos últimos términos, pesados con λ_2 y λ_3. Los términos $\hat{v}\ y\ \hat{u}$ se corresponden con un nuevo flujo de campo, Nij es la región de vecinos utilizada para estos terminos. El porqué de que está adición esté relacionado con un filtro de mediana esta demostrado en [5]. En él se expone que esta minimización esta relacionada con una computación de la mediana aunque un tanto diferente:

$$\hat{u}^{k+1}{}_{i,j} = median(Vecinos^k \cup Data)$$

Donde

$$Vecinos^k = \{\hat{u}^{k+1}{}_{i',j'}\}\ para\ i', j' \in N(i,j)$$

$$\hat{u}^0 = 0$$

$$\text{Data} = \left\{u_{i,j}, u_{i,j} \pm \frac{\lambda_3}{\lambda_2}, u_{i,j} \pm 2\frac{\lambda_3}{\lambda_2}, \dots, u_{i,j} \pm |N_{i,j}|\frac{\lambda_3}{2\lambda_2}\right\}$$

$$|N_{i,j}| = n\acute{u}mero\ de\ vecinos\ de\ (i,j)$$

Nótese que conforme la relación $\frac{\lambda_3}{\lambda_2}$ aumenta la ecuación de $\hat{u}^{k+1}{}_{i,j}$ se aproxima a la mediana en la primera iteración.

Una optimización de este modelo se consigue con la minimización alternativamente de:

$$E_0(u,v) = \sum_{i,j}\{\lambda_D \rho D(I_1(i,j) - I_2(i+u_{i,j}, j+v_{i,j})) + \lambda_s[\rho S(u_{i,j} - u_{i+1,j)} + \rho S(u_{i,j} - u_{i,j+1)} + \rho S(v_{i,j} - v_{i+,j}) + \rho S(v_{i,j} - v_{i,j+1})]\} + \lambda_2\left(||\boldsymbol{u} - \hat{\boldsymbol{u}}||^2 + ||\boldsymbol{v} - \hat{\boldsymbol{v}}||^2\right)$$

y

$$E_1(u,v) = \lambda_2\left(||\boldsymbol{u} - \hat{\boldsymbol{u}}||^2 + ||\boldsymbol{v} - \hat{\boldsymbol{v}}||^2\right) + \sum_{i,j}\sum_{i',j' \in N(i,j)} \lambda_3(|v_{i,j} - \hat{v}_{i',j'}| + |u_{i,j} - \hat{u}_{i',j'}|)$$

En primer lugar se minimiza $E_0(u,v)$ respecto a u y v tomando $\hat{v}$ y $\hat{u}$ como fijos y posteriormente se hace la operación inversa con $E_1(\hat{u},\hat{v})$

Finalmente se desarrolla una mejora del modelo, en vez de describir el termino no local como está descrito actualmente se hace con una función que representa como de probable es para un pixel i' j' estar en la misma superficie que i j.

$$\sum_{i,j}\sum_{i',j' \in N(i,j)} w_{i,j,i',j'}(|v_{i,j} - \hat{v}_{i',j'}| + |u_{i,j} - \hat{u}_{i',j'}|)$$

Esta función estará por tanto relacionada con la distancia entre píxeles, la diferencia entre los valores de la imagen en esas posiciones y la oclusión. Como es lógico la función debería incrementar con distancias largas entre píxeles o elevada oclusión como podemos apreciar:

$$w_{i,j,i',j'} = exp\left\{-\frac{|i-i'|^2 + |j-j'|^2}{2\sigma_1^2} - \frac{|I(i,j) - I(i',j')|}{2\sigma_2^2}\right\}\frac{o(i',j')}{o(i,j)}$$

La oclusión, como vemos en [6] se calculará la divergencia del flujo óptico que distingue entre distintos tipos de movimiento:

$$o(i,j) = \frac{\partial}{\partial x}u(i,j) + \frac{\partial}{\partial y}v(i,j)$$

Anexo C

Descriptor global basado en líneas

Se utiliza una técnica llamada detector de Canny [8] para detectar las líneas de una imagen. En primer lugar se pasa un filtro gaussiano con una sigma que nos da un suavizado de la imagen para eliminar el ruido de esta y evitar la detección de líneas que no existen debidas a espurios. En segundo lugar se calculan los gradientes en la dirección vertical y horizontal:

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

$$\theta = \text{atan}\left(\frac{G_y}{G_x}\right)$$

El ángulo theta se redondea a un ángulo múltiplo de 45 grados. Con este ángulo sabemos la dirección de la supuesta línea. Se establece un intervalo donde debemos de introducir un límite superior para ver cuando un punto será candidato a formará una línea y cuando no y un limite inferior. El límite inferior estará relacionado con lo permisivos que somos al forma la línea una vez se ha detectado el punto inicial de ésta.

Se mirará el valor del gradiente de los pixel posterior y anterior en la dirección del ángulo del pixel que estamos comprobando, si el valor de este pixel es superior a los otros dos entonces pasará a ser un candidato de línea y guardaremos su valor, en caso contrario será rechazado (ver figura 26).

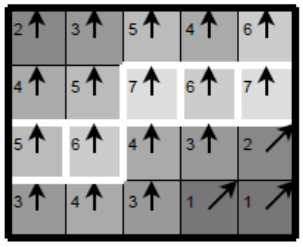

Figura 26. Gradientes y valores en una imagen. Figura tomado de [8]

A partir de aquí debemos establecer con que puntos nos quedamos para formar parte de una línea. El valor máximo del intervalo debe ser alto porque mide como de restrictivos vamos a ser en nuestra selección de las líneas. Al escoger un valor alto nos aseguraremos que ese borde formará una línea. El valor bajo del intervalo servirá para calcular los bordes que acompañaran al borde calculado con el intervalo alto, por tanto este valor deberá ser bastante pequeño para aseguramos de que formamos la línea que empieza en este valor alto. Una vez tenemos que píxeles van a formar líneas y cuales no se codifica la imagen en binario, si hay línea se iguala el pixel a 1 y si no a 0. Finalmente se procede a contar el número de píxeles que votan por una línea u otra. Sabemos que una línea esta parametrizada por 'ro' y 'theta' (ver figura 27), por tanto:

Con la transformada de Hough, que es un simple procedimiento matemático disponible en Matlab, se calcula la posición 'ro' y 'theta' de cada pixel con valor uno en la figura 27. 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.

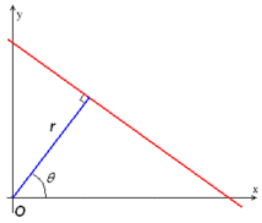

Figura 27 Línea parametrizada por r y θ . Figura tomada de [8].

Anexo D

Descriptor global basado en patrones locales binarios [7]

Descripción de un vídeo a partir de la repetición de patrones binarios en su volumen. Dado un punto del volumen del vídeo, se toman los puntos a su alrededor como vemos en la figura 28, se toman los fotogramas anterior y posterior, se describe un circulo alrededor del pixel en la imagen central y alrededor del pixel homólogo en las otros dos imágenes. Con estos tres círculos más los dos píxeles centrales exteriores tenemos definidos un conjunto de puntos que describirán al punto en cuestión.

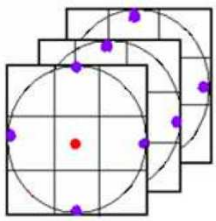

Figura 28 Volumen de puntos tomados para el cálculo de los patrones binarios respecto de cada punto. Figura tomada de [7]

El primer paso consiste en substraer el valor del pixel central a todos los demás encerrados en su volumen. Cada fotograma contendrá P puntos y disponemos de 3 fotogramas para cada cálculo del patrón binario, el central y el posterior y anterior.

Para conseguir invariancia en la escala de grises nos olvidamos de estas diferencias y simplemente codificamos en binario su resultado dependiendo de su signo, si este es positivo cambiaremos su valor por +1 y si es negativo lo cambiaremos por 0, esto lo representamos con la función 's':

$$s(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

Se forma una ristra de 0´s y 1´s (siempre en un mismo orden dado) con estos valores y se calcula su número decimal, surgen 2^(3P+2) (3P píxeles que rodean al central debido a que tomamos 3 fotogramas y 2 píxeles debidos al pixel central correspondiente a los fotogramas posterior y anterior) números decimales diferentes, cada vez que aparezca un número será contado y así se formara nuestro descriptor. Este cálculo se realiza para cada pixel en el vídeo dentro de unos límites, debe de estar dentro de estos límites ya que el círculo no puede ser cortado en los casos de píxeles cerca de las fronteras verticales, horizontales o temporales. En el fotograma último y primero por tanto no se podrá calcular, así como los píxeles más cercanos a la frontera que la distancia del radio.

Es necesaria también invariancia respecto a la rotación, para ello se toma un vector de cinco componentes, en la primera y última componente se emplaza el número binario correspondiente a los píxeles centrales de los fotogramas posterior y anterior. En las tres componentes centrales se hace lo mismo para los 4 píxeles de cada una de las tres imágenes, es decir en la segunda componente se colocan en orden los 4 píxeles de la primera imagen, en la tercera lo mismo con la imagen central y en la cuarta componente con la imagen posterior.

Una vez hecho esto por ejemplo nos queda un vector de este estilo:

(1, 1010, 1101, 1100, 1)

Cada una de las componentes del vector se rota el mismo número de veces hasta conseguir el número menor en el conjunto global del número binario. Esta rotación no se hace respecto de un punto y sí de este modo ya que no hay un solo eje de rotación como ocurre si calculásemos los LBP solamente en textura textura [9].

Tras la invariancia el vector queda:

(1, 0101, 1011, 1001, 1)

Para P=4 tendremos un clasificador de 16384 valores, es demasiado grande.

Por ello se hace la siguiente reducción del problema:

Se toman tres planos ortogonales respecto al punto de referencia, uno será el plano XY, otro el plano XT y por último el plano YT. Los tres planos en este caso tendrán el mismo punto central, se calculará para cada uno de ellos los LBP igual que en el caso anterior pero con la diferencia de que se toma cada plano independiente del otro y no se calcula el número binario de la composición de los tres, si no que se calcula el número binario para cada uno y posteriormente se concatenan como vemos en la siguiente imagen (figura 29).

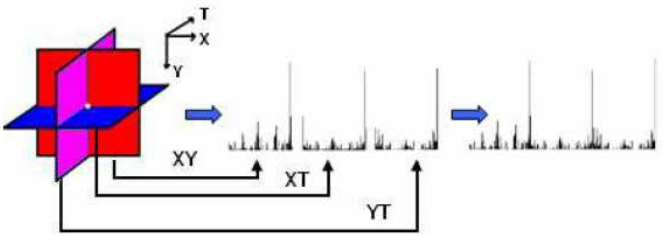

Figura 29 Concatenación de tres planos donde a cada uno se aplica los LBP y es independiente de los demás. Figura tomada de [7]

El histograma se define como una matriz $H_{i,j}$ donde j representa a que plano nos estamos refiriendo y el subíndice i representa el sumatorio de las veces que aparece el nivel 'i' para todos los planos de su tipo, es decir todos los planos XY, XT o YT. Por nivel 'i' entiéndase el valor que tomar el número binario al pasarlo a decimal.

$$H_{i,j} = \sum_{x,y,t} I\{f_j(x,y,t) = i\}$$

Donde la función 'I' devolverá 1 si su ecuación interior 'A'=true y 0 si 'A'=false:

Además se hace una normalización ya que estos planos pueden ser de diferentes tamaños en sus dos dimensiones, ya sean espaciales o espaciales y temporales, por ello:

$$N_{i,j} = \frac{H_{i,j}}{\sum_{k=0}^{n_{j-1}} H_{k,j}}$$

Anexo E

Detector de Harris en 3D

La idea consiste en calcular la correlación de un píxel con sus alrededores, se realiza con un suavizado previo del vídeo y la correlación se calcula como la diferencia entre un píxel y sus vecinos:

$$S(x,y,t) == \sum_u \sum_v \sum_t w(u,v,t)(I(u,v,t) - I(u+x, v+y, z+t))^2$$

Donde $w(u,v,t)$ *es el suavizado*

Al igual que se hacia en el cálculo del flujo óptico se quiere aproximar la diferencia $I(u,v,t) - I(u+x, v+y, z+t)$ por el desarrollo en serie de Taylor. La diferencia es que en este caso buscaremos máximos en esta diferencia y no mínimos ya que queremos detectar puntos salientes en las tres dimensiones.

$$\mathrm{I(u+x, v+y, z+t)} = \mathrm{I(u,v,t)} + \frac{\partial \mathrm{I}}{\partial \mathrm{x}}\ \mathrm{dx} + \frac{\partial \mathrm{I}}{\partial \mathrm{y}}\ \mathrm{dy} + \frac{\partial \mathrm{I}}{\partial \mathrm{t}}\ \mathrm{dt}$$

De esta manera la auto-correlación se puede aproximar como:

$$S(x,y,t) = \sum_u \sum_v \sum_t w(u,v,t) \left(\frac{\partial \mathrm{I}}{\partial \mathrm{x}} \ \mathrm{dx} + \frac{\partial \mathrm{I}}{\partial \mathrm{y}} \ \mathrm{dy} + \frac{\partial \mathrm{I}}{\partial \mathrm{t}} \ \mathrm{dt} \right)^2$$

Expresándolo en forma matricial:

$$S(x,y,t) = (x,y,t)A \begin{pmatrix} x \\ y \\ t \end{pmatrix}$$

Donde A es la matriz de Harris en cada pixel, en este caso queremos que los valores de los valores propios sean grandes los tres para tener los puntos salientes en las tres direcciones:

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

Anexo F

Clasificador

Se utiliza una maquina de aprendizaje supervisada (Support Vector Machine 'SVM'), es supervisada ya que conocemos la clase de cada video. Se transmite a la maquina la información de los descriptores y la clase de video al que pertenece cada uno de ellos. Ésta recopila la información y calcula las fronteras entre cada clase para una vez introducimos un nuevo video ver dentro de que frontera se encuentra y ser capaz de saber con la máxima posible exactitud a que clase pertenece.

El SVM es un algoritmo muy complejo y disponemos de sus librerías [10]. En estas líneas vamos a explicar como opera exactamente el SVM (ver figura 30). Lo vamos a hacer de una manera meramente intuitiva, ya que vamos a utilizar solamente dos características y dos clases que clasificaremos de forma lineal.

Figura 30 Ejemplo ilustrativo de como funciona el SVM, 3 clases y 2 características. Tomado de http://www.csie.ntu.edu.tw/~cjlin/libsvm/

Al tener solo dos características por clase podemos ver fácilmente en el plano X-Y de la figura 31 como funciona el SVM. En el eje X tendremos la característica X_1 y en el eje Y tendremos la característica X_2. El valor de ambas nos da una posición en el plano y su valor será '1' en el caso de ser una clase de video u '-1' en el caso de ser la otra clase. Recordemos que el sistema real será capaz de distinguir entre cuantos vídeos queramos y no entre solamente 2, en ese caso en vez de dar valores '-1' y '1' se les dará valores a las clases mediante numeración de estas.

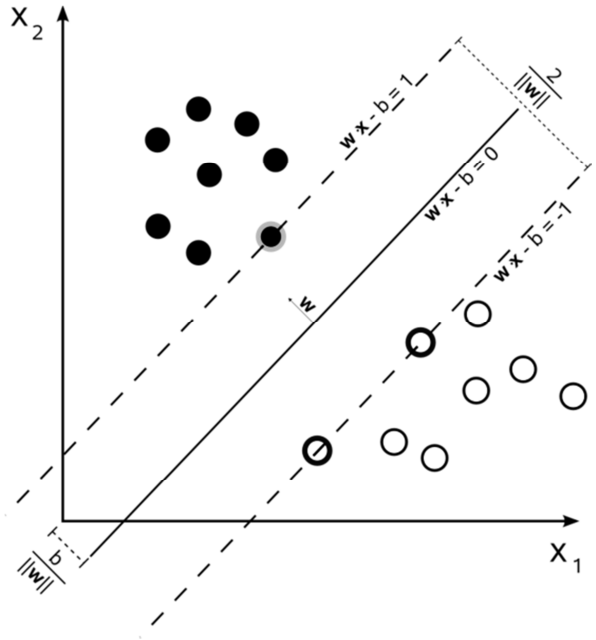

Figura 31 Ejemplo de división que realiza la máquina de aprendizaje.

Como podemos ver calculamos 2 hiperplanos que separan nuestras dos clases y entre ambos se calcula el hiperplano medio. Sabemos que un hiperplano esta definido como:

$$a_1X_1 + a_2X_2 - b = 0$$

Donde $[a_1, a_2]$ es el vector normal 'w' y el '0' de la parte de la derecha puede tomar cualquier valor siempre y cuando variemos el valor de b para dejar la ecuación intacta.

El hiperplano limitante de una clase lo forzamos a ser igual a 1 en el borde (mayor que uno cuando nos alejemos del borde hacia el exterior) y el hiperplano de la otra clase se fuerza a ser igual a -1 (menor que -1 cuando nos alejemos del borde hacia el exterior).

De esta forma tenemos 2 variables 'w' y 'b' pero podemos ver que las soluciones son infinitas ya que hay infinitos planos paralelos que cumplan estar entre ambas clases debido a las diferentes inclinaciones que pueden tomar, por tanto la idea es que los planos separadores se encuentren a la máxima distancia posible y así minimizar la posibilidad de fallo.

Matemáticamente:

$$\boldsymbol{wx} - b \geq 1;\ y \geq 1$$
$$\boldsymbol{wx} - b \leq -1;\ y \leq -1$$

Por tanto:

$$\boldsymbol{y}(\boldsymbol{wx} + b) \geq 1$$

De tal forma que se maximice la distancia entre planos que podemos calcular como:

$$\boldsymbol{wx_1} - b = 1$$

$$\boldsymbol{wx_2} - b = -1$$

$$\boldsymbol{w}(\boldsymbol{x_1} - \boldsymbol{x_2}) = 2$$

$$\frac{\boldsymbol{w}}{||\boldsymbol{w}||}(\boldsymbol{x_1} - \boldsymbol{x_2}) = \frac{2}{||\boldsymbol{w}||} = distancia$$

Conseguiremos maximizar la distancia minimizando $||w||$. El problema es que la norma nos obliga a trabajar con raíces cuadradas que no es bueno para el coste computacional por lo que pasamos a minimizar $\frac{1}{2}||w||^2$. El factor ½ es simplemente por conveniencia matemática, se irá al realizar la derivada en la minimización.

Concepto de los multiplicadores de Lagrange:

Concepto importante y necesario en nuestro desarrollo. Introducimos el problema de maximizar (minimizar) una o varias funciones ‘ f(x, y, z,…) ‘ sujetas a una o varias condiciones ‘g(x, y, z,…) = c’ (ver figura 32).

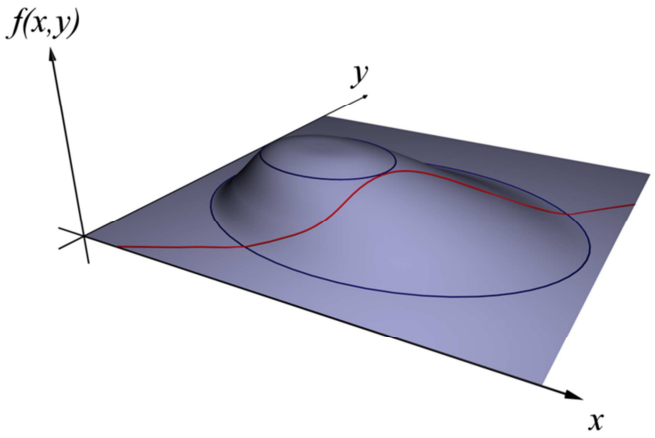

Figura 32. Representación de un funcional a minimizar f(x, y) sujeto a runa restricción g(x, y)=c. Imagen tomada de el articulo de los multiplicadores de LaGrange de Wikipedia.

Para resolverlo, podemos pensar que ambas funciones ‘f’ y ‘g’ deben cortarse para cumplirse a la vez. Además, debemos de buscar un máximo (mínimo) de ‘f’ por lo que el valor de esta no debe variar respecto a ‘g’ en el punto encontrado [x, y].

Por tanto no nos vale un punto donde 'f' y 'g' se crucen, si no que necesitamos que en ese punto sus tangentes sean iguales, así habremos encontrado el máximo de 'f' cumpliendo la restricción 'g'.

Matemáticamente:

$$\nabla_{\mathrm{x,y}}\mathrm{f} = -\lambda\nabla_{\mathrm{x,y}}g$$

El valor de lambda es debido a que las derivadas aunque iguales en dirección pueden tener distinta dimensión.

Le ecuación anterior y $g(x, y) = c$. las podemos unir como una nueva función:

$$\nabla_{x,y}f = -\lambda\nabla_{x,y}g$$

$$\Lambda(x, y, \lambda) = f(x, y) + \lambda(\mathrm{g(x,y)} - \mathrm{c})$$

Esto es así ya que si hacemos el gradiente respecto a 'x' e 'y' obtenemos la primera ecuación y si lo efectuamos respecto a lambda obtenemos la segunda $g(x, y) = c$. Así nuestra solución es:

$$\nabla_{x,y\lambda}\Lambda(x, y, \lambda) = 0$$

El gradiente igualado a cero de esta nueva función viene a ser lo mismo que una maximización (minimización) de:

$$\Lambda(x, y, \lambda) = f(x, y) + \lambda(\mathrm{g(x,y)} - \mathrm{c})$$

Por tanto parece claro que siguiendo con nuestro problema podemos plantearlo de esta misma manera a continuación, donde si hacemos este cambio de variables:

$$\alpha = -\lambda$$

$$\mathrm{f(x,y)} = \frac{1}{2}\left|\left|w\right|\right|^2$$
$$\mathrm{y(wx + b)} - 1 = g(x, y) - c$$

Tenemos:

$$min_{w,b,\alpha}\left\{\frac{1}{2}\left|\left|w\right|\right|^2 - \sum_{i=1}^{n}\alpha_i[\,\mathrm{y}_i(\mathrm{wx}_i - \mathrm{b}) - 1\right\}$$

Recordemos que la minimización es el gradiente igualado a cero, por ello la forma de expresarlo sigue siendo la misma. El problema que surge es que podríamos minimizarlo llevando los valores de α al infinito lo que no resulta correcto. Para resolverlo, maximizamos primero respecto a α y posteriormente continuamos con la minimización:

$$min_{w,b}max_{\alpha}\left\{\frac{1}{2}\left|\left|w\right|\right|^2 - \sum_{i=1}^{n}\alpha_i[\,\mathrm{y}_i(\mathrm{wx}_i - \mathrm{b}) - 1]\right\}$$

Surge otro inconveniente, el factor $\boldsymbol{y}(\boldsymbol{wx}+b)-1$ puede hacerse muy grande, por lo que igualaremos los α_i a cero en esos casos y será distinto de cero sólo en la frontera, en esos casos los $\boldsymbol{x_i}$ **serán nuestros vectores de apoyo (svm)**

Si derivamos respecto a 'w' e igualamos a 0 vemos que se tiene que cumplir la siguiente relación:

$$w = \sum_{i=1}^{n} \alpha_i \, \mathrm{y}_i \mathrm{x}_i$$

Lo mismo respecto a 'b':

$$\sum_{i=1}^{n} \alpha_i \, \mathrm{y}_i = 0$$

Solo algunos α_i serán mayores que cero, los correspondientes $\boldsymbol{x_i}$ serán los vectores de apoyo:

$$\mathrm{y}_i(\mathrm{wx}_i - \mathrm{b}) = 1 \Rightarrow \mathrm{wx}_i - \mathrm{b} = \frac{1}{\mathrm{y}_i} \Leftrightarrow b = \mathrm{wx}_i - \mathrm{y}_i$$

$$b = \frac{1}{N_{SV}} \sum_{i=1}^{N_{SV}} \mathrm{wx}_i - \mathrm{y}_i$$

Se realiza un cambio en la forma de expresarlo, se pasa a la forma dual:

Sustituimos estas 3 ecuaciones en la ecuación original a minimizar:

$$w = \sum_{i=1}^{n} \alpha_i \, \mathrm{y}_i \mathrm{x}_i$$

$$\sum_{i=1}^{n} \alpha_i \, \mathrm{y}_i = 0$$

$$\left|\left|w\right|\right|^2 = \mathrm{w} \cdot \mathrm{w}$$

Operando, llegamos a:

$$L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \, \alpha_j y_i y_j x^T{}_i x_j = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \, \alpha_j y_i y_j k(x^T{}_i, x_j)$$

Donde está solamente expresada respecto a α_i, por tanto se debe maximizar respecto a esta. Vemos que aparece una nueva función en vez de $x_i^T x_j$, aparece la función de Kernel:

$$k\left(x^T{}_i, x_j\right) = \varphi(x_i) \cdot \varphi(x_j)$$

La función de Kernel es utilizada ya que no siempre queremos separar nuestras clases de forma lineal, es por eso que en vez de $\boldsymbol{y}(\boldsymbol{wx} - b) \geq 1$ prefiramos otros tipos de separación, por ello, de forma general se puede expresar:

$$\boldsymbol{y}(\boldsymbol{w}\boldsymbol{\varphi}(\boldsymbol{x}) - b) \geq 1$$

Donde:

$$w = \sum_{i=1}^{n} \alpha_i \, \mathrm{y}_i \varphi(x_i)$$

$$w \cdot \boldsymbol{\varphi}(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i \, \mathrm{y}_i k(x_i, x)$$

La función de Kernel puede tener distintas composiciones, las mejores han sido estudiadas:

- Polinomio homogéneo: $k\left(x_i, x_j\right) = (x_i \cdot x_j)^d$
- Polinomio no homogéneo $k\left(x_i, x_j\right) = (x_i \cdot x_j + 1)^d$
- Función en base radial Gaussiana : $k\left(x_i, x_j\right) = exp\left(-\gamma \left|\left|x_i - x_j\right|\right|^2\right)$ para $\gamma > 0$
- Tangente hiperbólica: $k\left(x_i, x_j\right) = \tanh(kx_i \cdot x_j + c)$

¿Qué hacemos si la información no es separable? Si tenemos puntos clasificados en el otro dominio y viceversa (ver figura 33).

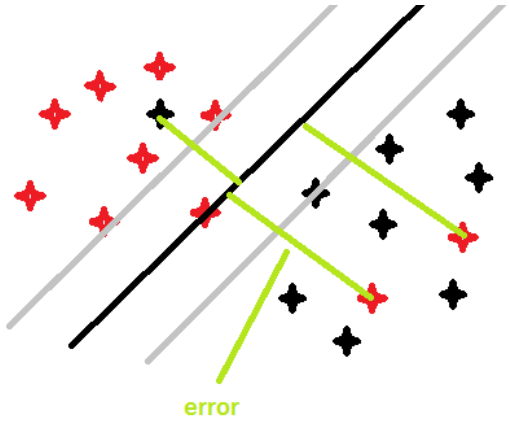

Figura 33. Información no separable

En este caso en vez de minimizar solamente la distancia 'w' también vamos a minimizar la suma de los errores con un peso 'C' a determinar para cada uno de ellos:

$$Min[\frac{1}{2}ww^T + C\Sigma\varepsilon_j]$$

$$\text{error } = \varepsilon_j$$

Nuestra función a cumplir cambiará por tanto para estos puntos que están fuera del margen, en estos casos:
$\boldsymbol{y_i(w\varphi(x_i) - b)} \geq 1\text{-}\varepsilon_i$

Procediendo de la misma manera que en el caso anterior:

$$min_{w,b}max_{\alpha\beta}\left\{\frac{1}{2}||w||^2 - \sum_{i=1}^{n}\alpha_i[\,\mathrm{y}_i(\mathrm{wx}_i - \mathrm{b}) - 1 + \varepsilon_i] - \sum_{i=1}^{n}\varepsilon_i\beta_i\right\}$$

Nuestra forma dual pasará a ser la misma, con la única restricción de que nuestros valores 'C' deben de ser mayores que 'α'. Maximizando respecto a 'α'':

$$L(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\,\alpha_j y_i y_j k({x^T}_i, x_j)$$

$$0 \leq \alpha_i \leq C$$

Para resolver el hecho de que nuestro problema tiene tantas clases como queramos y no sólo 2 lo reducimos de un problema de multi-clase a varios problemas de clasificación binarios donde podremos utilizar la teoría explicada, tenemos dos opciones:

1) 1 clase comparada con el resto. 1 vs 1
2) 1 clase comparada con todas las demás. 1 vs todos.

Para nuestro problema probamos el Kernel Gaussiano, debemos de maximizar la precisión de nuestro SVM respecto a las dos contantes a determinar, 'C' que mide el peso dado al error y 'gamma' que proviene del propio Kernel Gaussiano. Está demostrado que hay un rango de valores para ambos que maximizan esta eficiencia y los cuales probamos con un doble bucle para encontrar la mayor.

$$\mathrm{C}\,\epsilon\,\{2^{-5}, 2^{-3}, \dots, 2^{13}, 2^{15}\};$$

$$\gamma\,\epsilon\,\{2^{-15}, 2^{-13}, \dots, 2^{1}, 2^{3}\}$$

Realizamos un bucle recorriendo todos los valores de C y gamma y cogemos los que nos den una mayor precisión respecto a un conjunto de validaciones. Una vez lo tenemos, comprobamos que realmente funciona con un conjunto de tests.

Anexo G

Distancia entre descriptores

Se comprueba la validez de nuestro descriptor calculando las distancias entre histogramas de distintos vídeos. Cuanto menor sea esta distancia querrá decir que más similares son los dos histogramas de los vídeos que hemos analizado y por tanto si entre vídeos de la misma clase esta distancia es visiblemente inferior que entre vídeos de otras clases tendremos que nuestro método describe bien el vídeo.

Como se aprecia en la tabla 13 las respectivas distancias entre clases 'caballo andando' y 'caballo corriendo' son menores que la distancia entre cualquiera de estas dos clases y el resto. Así mismo el descriptor empareja antes a los caballos por el fondo que por la acción que realizan (en este caso correr o andar), es decir la distancia es menor de 'caballo 1 corriendo' a 'caballo 1 andando' que de 'caballo 1 corriendo' a 'caballo 2 corriendo'. En el caso de la clase 'speech' la distancia entre ambos vídeos es la mínima posible, por lo tanto describe bien personas hablando. Lo mismo ocurre con avión despejando y aterrizando y la clase ping pong, también es un buen descriptor para ellos. El descriptor sólo falla en la clase pájaros ya que la empareja con la clase ping pong, aunque no falla del todo porque la 2ª, 3ª y 4ª distancias mas cortas a pájaro 2 son pájaro 1, avión 2 y avión 1 respectivamente, por tanto parece ser un resultado lógico. Llegamos a la conclusión de que es un descriptor aceptable para nuestro objetivo y a partir del siguiente experimento utilizaremos la máquina de aprendizaje explicada en el apartado 3 en vez de mínimas distancias para clasificar los vídeos.

	Caballo 1	Caballo 1 R	Caballo 2	Caballo 2 R	Discurso 1	Discurso 2	Avión 1 A
Caballo 1	0	**0.23**	**0.33**	0.32	0.53	0.53	0.48
Caballo 1 R	**0.23**	0	0.36	0.35	0.56	0.55	0.52
Caballo 2	0.33	0.36	0	**0.20**	0.46	0.44	0.42
Caballo 2 R	0.32	0.35	0.35	0	0.5	0.48	0.46
Discurso 1	0.53	0.56	0.56	0.5	0	**0.34**	0.43
Discurso 2	0.53	0.55	0.55	0.48	**0.34**	0	0.41
Avión 1 A	0.48	0.42	0.42	0.46	0.43	0.41	0
Avión 1 D	0.44	0.47	0.41	0.42	0.52	0.47	**0.36**
Avión 2 A	0.41	0.45	0.38	0.4	0.45	0.44	**0.25**
Avión 2 D	0.43	0.47	0.4	0.4	0.49	0.44	0.35
Pájaros 1	0.56	0.58	0.51	0.54	0.42	0.47	0.44
Pájaros 2	0.48	0.49	0.44	0.44	0.47	0.42	0.46
Ping Pong 1	0.47	0.50	0.39	0.4203	0.44	0.42	0.35
Ping Pong 2	0.45	0.47	0.41	0.43	0.46	0.44	0.38

Tabla 13(1). Distancia entre descriptores

	Avión 1 D	Avión 2 A	Avión 2D	Pájaros 1	Pájaros 2	Ping-Pong 1	Ping Pong 2
Caballo 1	0.44	0.41	0.43	0.56	0.48	0.47	0.44
Caballo 1 R	0.47	0.45	0.47	0.58	0.49	0.50	0.47
Caballo 2	0.41	0.38	0.4	0.51	0.44	0.39	0.41
Caballo 2 R	0.42	0.4	0.40	0.54	0.44	0.42	0.43
Discurso 1	0.52	0.45	0.49	0.42	0.47	0.44	0.46
Discurso 2	0.47	0.44	0.44	0.47	0.42	0.42	0.44
Avión 1 A	0.36	**0.25**	0.35	0.44	0.41	0.35	0.38
Avión 1 D	0	0.35	**0.18**	0.48	0.42	0.37	0.40
Avión 2 A	0.35	0	0.33	0.43	0.41	0.35	0.36
Avión 2 D	**0.18**	0.33	0	0.49	0.41	0.35	0.37
Pájaros 1	0.48	0.43	0.49	0	0.41	0.46	0.45
Pájaros 2	0.42	0.41	0.41	**0.41**	0	0.39	0.37
Ping Pong 1	0.37	0.35	0.35	0.46	0.396	0	**0.29**
Ping Pong 2	0.40	0.36	0.37	0.45	**0.37**	**0.29**	0

Tabla 13(2). Distancia entre descriptores

Anexo H

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

Se utiliza el dataset elaborado por Ivan Laptev [14] para afinamiento de los parámetros de nuestro descriptor. Véase desde la figura 34 a la figura 39 algunas de las distintas clases utilizadas en este dataset.

Figura 34 Conducir

Figura 35. Levantarse

Figura 36. Pelea

Figura 37. Correr

Figura 38 Levantarse

Figura 39. Ponerse de pie

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 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).

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

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 vídeos, 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 una ristra nueva 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:

	Aleatorio	G-HOG3D		Aleatorio	G-HOG3D
AnswerPhone	0,012	**0,026**	HugPerson	**0,077**	0,041
DriveCar	0,151	**0,183**	Kiss	0,089	**0,111**
Eat	0,120	**0,231**	Run	0,056	**0,109**
FightPerson	0,184	**0,325**	SitDown	**0,045**	0,000
GetOutCar	0,023	**0,077**	SitUp	0,038	**0,050**
HandShake	0,041	**0,138**	StandUp	0,023	**0,108**
mean	0.071	**0.112**	mean	0.071	**0.112**

Tabla 14. Resultados con el G-HOG3D y unos parámetros iniciales.

Se realiza un primer experimento. En la tabla 14 se ve como no salen resultados demasiado altos. Analizando los vídeos se ve que quizás el 'dataset' de Hollywood2 no sea muy interesante para verificar las mejoras que se realicen en nuestro descriptor ya que tiene diversos problemas: En algunas tomas ocurren dos mismas acciones a la vez (por ejemplo, ir en coche y comer, hablar por teléfono y comer, dar un beso y abrazo al mismo tiempo, etc…) además mezcla escenas 'buenas' donde aparece la acción especificada con diversas escenas donde no ocurre aparentemente nada o incluso ocurre otra acción distinta a la que realmente debería de ocurrir. Por esto, se hizo un filtrado del dataset, escogiendo sólo escenas buenas y esto dio lugar, como se ve en la tabla 15, a un incremento de casi el 100% en la clasificación de vídeos.

	Aleatorio	G-HOG3D	G-HOG3D FILTRADO
AnswerPhone	0,012	0,026	**0,100**
DriveCar	0,151	0,183	**0,370**
Eat	0,120	0,231	**0,391**
FightPerson	0,184	**0,325**	0,300
GetOutCar	0,023	0,077	**0,214**
HandShake	0,041	0,138	**0,161**
HugPerson	0,077	0,041	**0,200**
Kiss	0,089	0,111	**0,132**
Run	0,056	0,109	**0,222**
SitDown	0,045	0,000	**0,333**
SitUp	0,038	**0,050**	0,000
StandUp	0,023	**0,108**	0,000
mean	0.017	0.112	**0,202**

Tabla 15. Comparación entre el dataset y el dataset filtrado a sólo tomas buenas

Como se explicó en la sección descriptores en cada celdilla tenemos los valores de los módulos de los vectores direccionales y para fijar si el valor de este módulo es punible como

significativo se compara con los demás módulos de dos formas distintas, por un lado si este es superior a la media se acepta y por el otro si es superior a la mediana. Como se ve en la tabla se han hecho comparaciones y utilizando la mediana se obtienen resultados de entorno al 3% superiores. Por tanto, al menos para este dataset, conviene utilizar la mediana (ver tabla 16).

	media	mediana	media	mediana	media	mediana
AnswerPhone	0,100	**0,105**	**0,136**	0,100	0,059	**0,150**
DriveCar	0,345	**0,357**	0,303	**0,370**	0,323	**0,333**
Eat	0,300	**0,333**	0,353	**0,391**	**0,296**	0,280
FightPerson	**0,300**	**0,300**	**0,233**	**0,300**	**0,304**	**0,313**
GetOutCar	**0,129**	0,107	0,150	**0,214**	0,069	**0,077**
HandShake	0,238	**0,250**	**0,346**	0,161	0,208	**0,250**
HugPerson	0,143	**0,200**	0,067	**0,200**	0,182	**0,214**
Kiss	0,180	**0,238**	**0,143**	0,132	0,154	**0,233**
Run	0,240	**0,300**	0,211	**0,222**	**0,304**	0,318
SitDown	0,000	**0,143**	**0,400**	**0,333**	0,222	**0,333**
SitUp	0,000	0,000	0,000	0,000	0,000	0,000
StandUp	**0,143**	0,133	0,000	0,000	0,133	**0,177**
mean	0,176	**0,206**	0,195	**0,202**	0,188	**0,223**

Tabla 16. Comparación ente el uso de media y mediana.

Las celdillas son divisiones de nuestro vídeo tanto espaciales como temporales, para la división temporal se comprueba (ver tabla 17) que a mayor división en el tiempo mejor clasificación (hasta un punto donde empieza a bajar la precisión si seguimos incrementando la discretización temporal). El pensamiento es que en el caso de ser una acción continua y constante importará poco si discretizamos mucho o poco mientras que si la acción es variable con el tiempo mejorará la clasificación con diescretizaciones más altas, por ejemplo para describir un beso, si este se compone de mirada inicial, beso y despedida parece que funcionaría mejor una descripción temporal dividida en tres partes y no en una sola parte o uno dividido en más partes que tres.

	1 celdilla temporal	2 celdillas temporales	3 celdillas temporales	4 celdillas temporales	6 celdillas temporales
AnswerPhone	0,000	**0,188**	0,177	0,150	0,118
DriveCar	0,000	0,321	0,333	0,333	**0,355**
Eat	0,000	0,308	**0,318**	0,280	0,286
FightPerson	0,278	0,238	0,208	**0,313**	0,250
GetOutCar	0,000	0,036	0,077	0,077	**0,100**
HandShake	0,250	**0,280**	**0,280**	0,250	0,269
HugPerson	0,000	**0,250**	0,214	0,214	**0,235**
Kiss	0,092	0,088	0,135	**0,233**	0,156
Run	0,500	0,143	0,313	0,318	**0,500**
SitDown	0,000	0,000	0,167	**0,333**	0,200
SitUp	0,000	**0,222**	0,000	0,000	0,000
StandUp	0,167	0,111	0,125	**0,177**	0,067
mean	0,107	0,182	0,196	**0,223**	0,211

Tabla 17. Aumento de las celdillas temporales

Las imágenes como hemos explicado anteriormente han sido reducidas (ver figura 40), también hemos comprobado el efecto de estas reducciones en los resultados (tabla 18), entre 24 y 96 píxeles para la dimensión vertical de la imagen la mejor clasificación la da un valor intermedio de 36 / 48 píxeles. La explicación es que para imágenes grandes tenemos información con mucho detalle y se calcularán muchos gradientes innecesarios, para imágenes muy pequeñas ocurre que perdemos gradientes importantes conforme el pixelado es menor en la imagen.

Figura 40. Fotograma con distintos tamaños.

	Tamaño 24 en altura	Tamaño 36 en altura	Tamaño 48 en altura	Tamaño 72 en altura	Tamaño 96 en altura
AnswerPhone	0,125	0,136	0,118	**0,158**	0,133
DriveCar	0,300	0,313	**0,355**	0,324	0,231
Eat	0,304	0,250	0,286	**0,308**	0,296
FightPerson	0,280	0,273	**0,250**	0,188	0,125
GetOutCar	0,048	**0,143**	0,100	0,063	0,079
HandShake	0,194	**0,273**	0,269	0,238	0,214
HugPerson	**0,250**	0,182	0,235	0,200	0,071
Kiss	0,125	**0,177**	0,156	0,132	0,091
Run	0,167	0,313	**0,500**	0,333	0,250
SitDown	0,250	0,333	0,200	0,333	**0,429**
SitUp	0,000	**0,200**	0,000	0,000	0,000
StandUp	**0,143**	0,000	0,067	0,077	0,077
mean	0,182	0,216	0,211	0,196	0,166

Tabla 18. Aumento del tamaño en altura

Para la discretización en textura de las celdillas ocurre algo parecido que para la temporal, en este caso hemos analizado como responde ante la variación del número de celdillas verticales y horizontales y efectivamente mejoran los resultados a mayor número de celdillas (otra vez, hasta llegar a cierto punto, ver tabla 19). En este caso se ha dividido la anchura y altura entre un número par para no perder la perspectiva de que la persona es un sujeto simétrico.

	6 celdillas espaciales	8 celdillas espaciales		6 celdillas espaciales	8 celdillas espaciales
AnswerPhone	0,150	**0,210**	HugPerson	**0,214**	0,167
DriveCar	0,333	**0,400**	Kiss	**0,233**	0,143
Eat	0,280	**0,364**	Run	**0,318**	0,235
FightPerson	**0,313**	0,250	SitDown	**0,333**	0,286
GetOutCar	0,077	**0,095**	SitUp	0,000	0,000
HandShake	0,250	**0,300**	StandUp	**0,177**	0,077
mean	**0,223**	0,211	mean	**0,223**	0,211

Tabla19. Aumento de celdillas espaciales

Lo mismo ocurre para el número de bins (tabla 20), donde se han utilizado sobretodo números pares de configuraciones con el mismo objetivo de preservar la simetría del cual hemos habla de anteriormente y el sistema mejora hasta llegar a 6 divisiones por ángulo, 8 divisiones ya da lugar a un empeoramiento de los resultados.

	6 bins	8 bins	4 bins		6 bins	8 bins	4 bins
AnswerPhone	**0,150**	0,105	0,111	HugPerson	**0,214**	0,200	0,154
DriveCar	0,333	**0,357**	0,200	Kiss	0,233	**0,238**	0,200
Eat	0,280	**0,333**	0,143	Run	**0,318**	0,300	0,286
FightPerson	**0,313**	0,300	0,191	SitDown	**0,333**	0,143	0,000
GetOutCar	0,077	**0,107**	0,059	SitUp	**0,000**	**0,000**	**0,000**
HandShake	**0,250**	**0,250**	0,217	StandUp	**0,177**	0,133	0,133
mean	**0,223**	0,206	0,141	mean	**0,223**	0,206	0,141

Tabla 20. Aumento de 'bins'.

Se han probado otra configuraciones del descriptor, han dado peores resultados debido a que se ha utilizado información redundante en el sentido de que es repetida, esto es así ya que en vez de utilizar dos ángulos para el descriptor se han utilizado tres y realmente el vídeo es un volumen de imágenes y se puede describir con sólo dos ángulos, al igual que para describir un solo fotograma es suficiente un solo ángulo y no dos.

Anexo I

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

Se trata, como vemos desde la figura 41 a 46, 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.

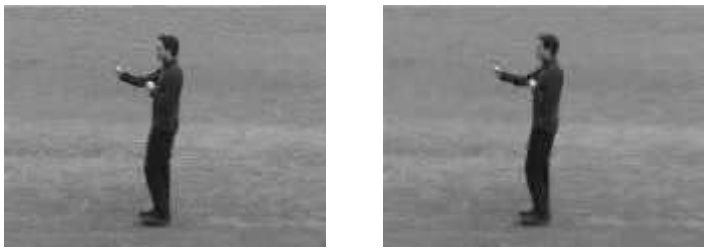

Figura 41. Boxear

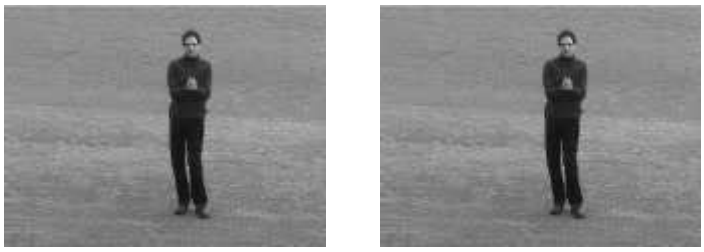

Figura 42. Dar palmas

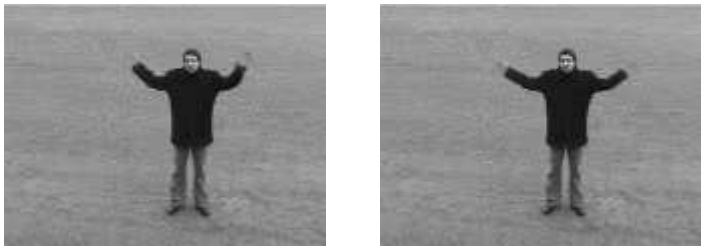

Figura 43. Movimiento de manos

Figura 44. Trotar

Figura 45. Correr

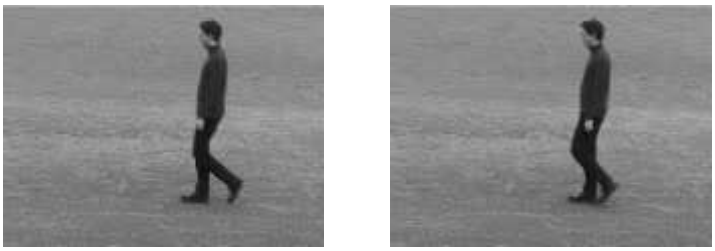

Figura 46. Caminar

Se sigue sintonizando los parámetros de nuestro descriptor. Este dataset de reconocimiento de acciones, también viene siendo muy utilizado por diversos autores, la razón por la que lo utilizamos, a parte de para poder comparar nuestro descriptor con el resto es porque se trata de un 'dataset' más sencillo y menos extenso lo que proporciona una visualización y comparación de resultados más fácil y rápida. Por etapas se hizo lo siguiente:

1) Se comienza utilizando el descriptor que mejor ha funcionado en el experimento anterior, se utiliza la misma configuración (tabla 21). Disponemos de 100 vídeos en total para cada una de las cuatro clases. Se otorgan 32 vídeos para hacer el entrenamiento de nuestra maquina de aprendizaje, 32 para la validación de la misma y 36 para comprobar los resultados de clasificación.

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
G-HOG3D	0.6531	0,9524	0,9140	0,5000	0,5700	0,5600	0.6531

Tabla 21. Prueba inicial para el G-HOG3D

2) En este descriptor se piensa en la idea de que es más importante el cambio en el tiempo de la acción que la acción en si misma. Por ello se piensa que pesando más los gradientes calculados en la dimensión temporal mejorará los resultados. Estos gradientes se calculan como hemos visto en la sección del descriptor HOG como diferencias en la dimensión temporal entre píxeles contiguos. Por este motivo se hace una prueba dándole un peso doble a los gradientes temporales respecto a los espaciales. Como vemos (tabla 22) los resultados no son bastante más bajos por lo que seguiremos pesando igualmente de ahora en adelante.

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
Peso gradiente temporal unitario	**0,6531**	**0,9524**	0,9140	**0,5000**	0,5700	**0,5600**	**0.6900**
Peso gradiente temporal doble	0,4500	0,7300	**1**	0,4400	**0,6190**	0,500	0.6200

Tabla 22. Aumento del peso en el gradiente temporal

3) Utilizamos el mismo descriptor que en la primera etapa, el cambio es que ahora no utilizamos los mismos vídeos que antes para entrenar y validar, pasamos 24 de los vídeos de validación a entrenamiento. Con esta modificación mejoran los resultados, algo que parece evidente ya que se utilizan los mismos vídeos para entrenar a nuestra SVM pero en este caso para el entrenamiento propiamente dicho se utilizan más videos y menos para la validación ya que con valores de entorno al 20% de vídeos para la validación respecto al entrenamiento es suficiente. Así, de ahora en adelante utilizaremos esta configuración para los vídeos. Ver tabla 23.

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
Vídeos de entrenamiento originales	**0,6531**	**0,9524**	0,9140	0,5000	0,5700	0,5600	0.6900
Mayor número de vídeos en entrenamiento	0,6182	0,9167	**1**	**0,6000**	**0,5800**	**0,7800**	**0,7491**

Tabla 23 Incremento del número de vídeos de entrenamiento

4) En esta nueva etapa variamos la forma de aceptación de un punto como significativo o no significativo. Como ya se ha explicado, para aceptar el gradiente total dentro de una celdilla lo compararemos con el resto mediante las funciones media y mediana. Si este valor es más grande que la media o mediana (depende de la función que utilicemos) lo tomaremos como significativo. Hemos utilizado la mediana ya que en el experimento de Hollywood otorgaba resultados un 3% superiores y además es más robusta a valores erróneos. En este caso vemos que la media funciona un 2% poco mejor y la continuaremos utilizando (ver tabla 24).

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
mediana	**0,6182**	0,9167	**1**	0,6000	0,5800	0,7800	0,7491
media	0,6111	**0,9545**	0,9667	0,6000	**0,6522**	**0,8235**	**0,7680**

Tabla 24.

5) Igual que en la etapa 4 pero sin el suavizado que se le hacía a cada imagen donde se elevaba el valor de cada pixel a 0,5 y se hacía una redistribución de los colores para utilizar toda la gama de estos, es decir desde 0 a 255. Se ve que el resultado mejora sin esta operación que sufrían las imágenes, esto es debido a que el suavizado provoca que algunos de gradientes se pierdan, esto es bueno si se pierden gradientes pequeños que no aportan la información necesaria, pero si se pierden los gradientes importantes es cuando los resultados empeoran (ver tabla 25).

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
Con suavizado	0,6111	0,9545	0,9667	0,6000	0,6522	**0,8235**	0,7680
Sin suavizado	**0,6800**	**0,9600**	**0,9688**	**0,6296**	**0,6818**	0,7895	**0,7849**

Tabla 25. Comparación entre suavizar y no suavizar.

6) Igual que en la etapa 5 pero en vez de calcular el ángulo que describen nuestro vídeo temporalmente mediante el gradiente en 'x' y en 't' lo haremos mediante los gradientes en 'y' y en 't'. Esto da un mejor resultado utilizar la relación de gradientes verticales respecto al tiempo antes que la utilización de gradientes horizontales respecto al tiempo (ver tabla 26). En un principio parecería lógico que los resultados fueran similares o iguales ya que para calcular la posición de un vector en tres dimensiones es suficiente con dos ángulos, sean los calculados hasta ahora o sea los probados en esta nueva etapa pero en cambio salen unos resultados un 4% superiores por lo que seguiremos utilizando esta configuración

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
Ángulo temporal (x-t)	0,6800	0,9600	0,9688	0,6296	0,6818	0,7895	0,7849
Ángulo temporal (y-t)	**0,7447**	**0,9655**	**1**	**0,6571**	**0,7632**	**0,8330**	**0,8272**

Tabla 26. Diferentes formas de cálculo del ángulo temporal.

7) Se calculan los descriptores utilizado tanto en la etapa 5 como en la etapa 6 (ver tabla 27) de forma independiente y se juntan uno detrás de otro en un descriptor global, se obtienen los mismos resultados que en la anterior etapa por tanto

desechamos esta idea ya que el descriptor es el doble en tamaño y no aporta mejores resultados.

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
Etapa 5	0,6800	0,9600	0,9688	0,6296	0,6818	0,7895	0,7849
Etapa 6	**0,7447**	**0,9655**	**1**	**0,6571**	**0,7632**	**0,8330**	**0,8272**
Etapa 5+6	**0,7447**	**0,9655**	**1**	**0,6571**	**0,7632**	**0,8330**	**0,8272**

Tabla 27. Unión etapa 5 y etapa 6

8) Una vez hemos calculado una celdilla al completo se procede a normalizarla como ya hemos visto, después se toman los valores mayores que 0.2 y se igual a 0.2 para evitar altos gradientes debidos por ejemplo a un cambio de luminosidad que no aporta mucha información. Se elimina esta restricción. Los resultados empeoran considerablemente al pensar que no íbamos a encontrar valores tan grandes como para estropear nuestro descriptor ya que valores grandes restan la aportación de los valores más pequeños (ver tabla 28).

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
Con límite de 0.2	**0,7447**	**0,9655**	**1**	**0,6571**	**0,7632**	**0,8330**	**0,8272**
Sin límite de 0.2	0,6863	0,8667	**1**	0,6098	0,7576	0,8286	0,7915

Tabla 28. Comparación entre limitación de los gradientes y no limitación

9) Igual que la etapa 6 pero se aumentan los bins. En este 'dataset' un aumento de los bins de 6 a 8 si que produce una mejora, aunque realmente poco significativa. Ver tabla 29

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
6 bins	**0,7447**	**0,9655**	**1**	0,6571	0,7632	**0,8330**	0,8272
8 bins	0,7292	0,9360	**1**	**0,7143**	**0,8056**	0,8158	**0,8334**

Tabla 29.

10) Igual que la etapa 9, el cambio aquí es muy parecido al cambio en 8 donde quitábamos la restricción de que si en una celdilla daba un valor mayor de 0.2 se igualaba a 0.2, en este caso somos un poco menos restrictivos y dejamos este valor en 0.4. Vemos que antes no había funcionado la idea de quitar la operación que elimina valores mayores de 0.2, en cambio si hacemos lo mismo pero con una restricción menor de hasta 0.4 los resultados mejoran aunque no es un gran cambio. Ver tabla 30.

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
Límite de 0.2	**0,7292**	**0,9360**	**1**	0,7143	0,8056	0,8158	0,8334
Límite de 0.4	0,7083	0,8710	**1**	**0,7879**	**0,8158**	**0,8611**	**0,8406**

Tabla 30. Distintas limitaciones para los valores altos del histograma.

11) Anteriormente se tomaba cada vídeo por completo y en este apartado se divide el vídeo en 4 partes ya que en cada uno transcurren 4 acciones idénticas, así las separamos gracias a un fichero de texto que identifica los intervalos donde se encuentra cada toma. Se obtienen resultados en torno al 3% superiores como podemos ver en la tabla. Se han tomado 12 bins para los ángulos y han ido variando las celdillas, nótese el incremento de la precisión al incrementar estas. Ver tabla 31.

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
1 celdilla	0.7053	0.7624	**0.9643**	0.6463	0.6270	**0.8966**	0.7670
2 celdillas	0.8421	0.77826	0.7132	0.7912	0.8125	0.8727	0.8024
3 celdillas	0.9153	**0.9524**	0.7007	0.8068	0.8778	0.8205	0.8456
4 celdillas	1	0.9022	0.7007	0.7938	**0.9277**	0.8276	0.8587
5 celdillas	**0.9322**	0.9167	0.7619	**0.8587**	0.8830	0.8889	**0.8736**

Tabla 31. Incremento del número de celdillas

Experimento con detección de acción y substracción de esta

Se hacen tres pruebas relacionadas, la primera consiste en detectar la zona del fotograma que está en movimiento (explicado en sección descriptores) independientemente del movimiento de la cámara y analizarla igual que en casos anteriores, se llega a un resultado un uno por ciento superior al mejor caso de los anteriores y con la utilización de menos bins. Además se realiza el mismo experimento utilizando sólo el 40% de los vídeos para una mayor rapidez de en la próxima comparación. Ver tabla 32.

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
Con detección de acción	**1**	**0.9556**	**0.7680**	**0.9041**	**0.7788**	0.9029	**0.8849**
Con detección acción y utilizando el 40% de vídeos	**1**	0.9500	0.6531	0.8095	0.6512	**1**	0.8440

Tabla 32. Resultado tras la detección de acción

La segunda prueba consiste en hacer lo mismo que en el anterior caso pero se intenta mejorar la zona del fotograma relacionada con el movimiento en la acción, para ello como se ha explicado en la sección 3.5 se introduce un sistema de votaciones donde se selecciona una ventana que se ajuste lo mejor posible a 10 fotogramas consecutivos para suavizar el tamaño y posición de la ventana de movimiento, que no sea tan variable de un fotograma a otro y capte a la persona en su totalidad, además se hace un cambio constante y lineal del tamaño del fotograma para evitar cambios bruscos en los tamaños de estas ventanas. El resultado para este caso como vemos es un poco inferior que para el caso con la detección del movimiento sin suavizar. Ver tabla 33

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
Con detección acción y utilizando el 40% de vídeos	**1**	0.9500	**0.6531**	**0.8095**	0.6512	1	**0.8440**
Con detección acción suavizada y utilizando el 40% de vídeos	0.8065	**1**	0.6250	0.7500	**0.8485**	1	0.8383

Tabla 33. Comparación entre detección de acción suavizada y sin suavizar

Como tercera prueba y definitiva se hace que el tamaño de la ventana global que siempre sea igual de grande y en la misma posición para evitar la unión de gradientes de distintas zonas del cuerpo en una misma celdilla. Para ello se toma las 4 esquinas más alejadas de entre todas las ventanas calculadas en el caso anterior, como vemos los resultados mejoran un 4% respecto a la misma configuración pero sin substracción del movimiento. Ver tabla 34

	Boxing	Handclapping	Handwaving	Jogging	Running	Walking	mean
Sin detección acción	**1**	**0.9022**	0.7007	0.7938	**0.9277**	0.8276	0.8587
Con detección acción	0.9868	0.8812	**0.8807**	**0.8734**	0.8269	**0.9057**	**0.8925**

Tabla 34. Comparación entre detección de acción global y no detección de acción

Anexo J

Ecuación de Euler-Lagrange

La naturaleza ahorra energía potencial, siempre busca la posición de equilibrio hacia un estado de menos energía. Es por ello que se busca la minimización de la acción, es lo mismo que se busca en las minimizaciones de este proyecto como las que se utilizan en las maquinas de aprendizaje o en el cálculo del flujo óptico. En nuestro caso la minimización de 'S' está definida

como la integral de una función lagrangiana 'L' desconocida entre dos instantes de tiempo dados:

$$S(q) = \int_{\mathrm{a}}^{b} L\big(t, q(t), q'(t)\big)dt$$

Se desea considerar todas las posibles trayectorias que sigue la partícula durante el intervalo de tiempo analizado, podemos entonces parametrizarlo con el parámetro α:

$$q(t, \alpha) = q(t) + n(x)\alpha$$

Donde $n(a) = n(b) = 0$ para que $q(t, \alpha) = q(t)$ en los bordes temporales ya que eso no lo queremos cambiar. Sólo es interesante saber que trayectorias hacen que la acción se minimice pero la acción en los bordes debe ser la misma ya que todas las trayectorias deben de empezar y acabar en el mismo lugar. De este modo derivamos la acción respecto a α e igualamos a 0 para llegar al mínimo (o máximo):

$$\frac{\partial S(q, \alpha)}{\partial \alpha} = 0$$

Que es lo mismo que:

$$\int_{a}^{b} \frac{\mathrm{dL}}{\mathrm{d}\alpha}\mathrm{dt} = 0$$

Tras unas operaciones se llega a la buscada ecuación de LaGrange, en primer lugar aplicamos la regla de la cadena, descomponemos así la derivada total en sus parciales:

$$\frac{\mathrm{dL}}{\mathrm{d}\alpha} = \frac{\partial \mathrm{q}}{\partial \alpha}\frac{\partial \mathrm{L}}{\partial q} + \frac{\partial \dot{\mathrm{q}}}{\partial \alpha}\frac{\partial \mathrm{L}}{\partial \dot{\mathrm{q}}} + \frac{\partial \mathrm{t}}{\partial \alpha}\frac{\partial \mathrm{L}}{\partial t}$$

La derivada del tiempo respecto a α es cero porque no depende de ella:

$$\frac{\mathrm{dL}}{\mathrm{d}\alpha} = \frac{\partial \mathrm{q}}{\partial \alpha}\frac{\partial \mathrm{L}}{\partial q} + \frac{\partial \dot{\mathrm{q}}}{\partial \alpha}\frac{\partial \mathrm{L}}{\partial \dot{\mathrm{q}}}$$

El segundo termino de la parte de la derecha de la igualdad molesta por lo que se escribe en la forma:

$$\frac{\partial \dot{\mathrm{q}}}{\partial \alpha}\frac{\partial \mathrm{L}}{\mathrm{d}\dot{\mathrm{q}}} = \left(\frac{\partial \dot{\mathrm{q}}}{\partial \alpha}\frac{\partial \mathrm{L}}{\partial \dot{\mathrm{q}}}\right) - \frac{\partial \mathrm{q}}{\partial \alpha}\left(\frac{\partial \dot{\mathrm{L}}}{\partial \dot{\mathrm{q}}}\right)$$

Llegando a:

$$\frac{\mathrm{dL}}{\mathrm{d}\alpha} = \frac{\partial \mathrm{q}}{\partial \alpha}\frac{\partial \mathrm{L}}{\partial q} + \left(\frac{\partial \dot{\mathrm{q}}}{\partial \alpha}\frac{\partial \mathrm{L}}{\partial \dot{\mathrm{q}}}\right) - \frac{\partial \mathrm{q}}{\partial \alpha}\left(\frac{\partial \dot{\mathrm{L}}}{\partial \dot{\mathrm{q}}}\right)$$

Lo introducimos en la integral:

$$\int_{a}^{b} \left(\frac{\partial \mathrm{q}}{\partial \alpha}\frac{\partial \mathrm{L}}{\mathrm{d}q} + \left(\frac{\partial \dot{\mathrm{q}}}{\partial \alpha}\frac{\partial \mathrm{L}}{\mathrm{d}\dot{\mathrm{q}}}\right) - \frac{\partial \mathrm{q}}{\partial \alpha}\left(\frac{\partial \dot{\mathrm{L}}}{\partial \dot{\mathrm{q}}}\right)\right)\mathrm{dt} = 0$$

$$\int_a^b \left(\frac{\partial \mathrm{q}}{\partial \alpha}\frac{\partial \mathrm{L}}{\mathrm{d}q} - \frac{\partial \mathrm{q}}{\partial \alpha}\left(\frac{\partial \dot{\mathrm{L}}}{\partial \dot{\mathrm{q}}}\right)\right)\mathrm{dt} + \int_a^b \left(\frac{\partial \mathrm{q}}{\partial \alpha}\frac{\partial \dot{\mathrm{L}}}{\partial \dot{\mathrm{q}}}\right)\mathrm{dt} = 0$$

Sabiendo que $\frac{\partial \mathrm{q}}{\mathrm{d}\alpha}$ en los extremos porque en ellos todas trayectorias deben de tener el mismo valor podemos extrapolar que la segunda integral es nula:

$$\int_a^b \left(\frac{\partial \mathrm{q}}{\partial \alpha}\frac{\partial \mathrm{L}}{\partial q} - \frac{\partial \mathrm{q}}{\partial \alpha}\left(\frac{\partial \dot{\mathrm{L}}}{\partial \dot{\mathrm{q}}}\right)\right)\mathrm{dt} = 0$$

Que debe cumplirse para cualquier desviación $\frac{\partial \mathrm{q}}{\mathrm{d}\alpha}$:

$$\left(\frac{\partial \mathrm{L}}{\partial q} - \left(\frac{\partial \dot{\mathrm{L}}}{\partial \dot{\mathrm{q}}}\right)\right) = 0$$

$$\frac{\partial \mathrm{L}\left(\mathrm{t}, \mathrm{q}(\mathrm{t}), \mathrm{q}'(\mathrm{t})\right)}{\partial \mathrm{q}(\mathrm{t})} - \frac{\mathrm{d}}{\partial \mathrm{t}}\frac{\partial \mathrm{L}\left(\mathrm{t}, \mathrm{q}(\mathrm{t}), \mathrm{q}'(\mathrm{t})\right)}{\mathrm{q}'(\mathrm{t})} = 0$$

Anexo K

Dataset

A continuación se muestran las clases del 'dataset' creado (figuras 47 a 54).

Figura 47. Lancha motora

Figura 48. Caballo y Jinete

Figura 49. León caminando

Figura 50. Avión despegando

Figura 51. Persona hablando

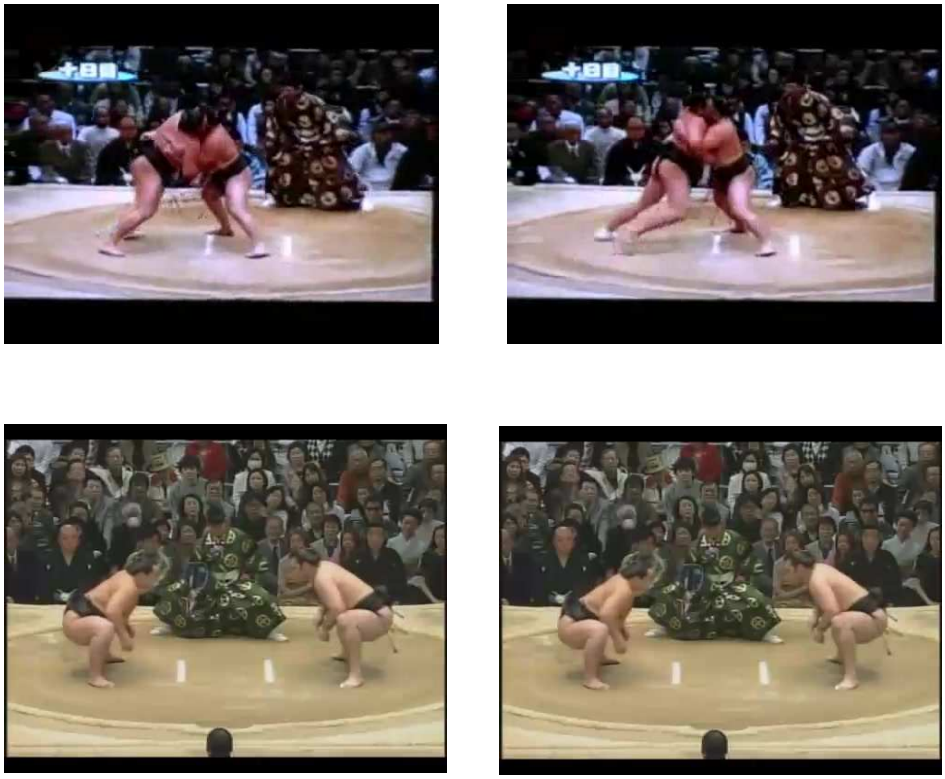

Figura 52. Pelea de sumo

Figura 53. Punto en tenis

Figura 54. Tren