Proyecto final de máster
Reconocimiento facial

A. Bruno Rodríguez Rodríguez
Director: Gregorio Robles

Máster oficial en Software libre
Universitat oberta de Catalunya

4 de junio de 2011

1. Introducción
   - Marco de trabajo
   - Entorno de trabajo
2. Funcionamiento
   - Visión global del sistema
   - Preprocesado
   - Localización de región de interés
   - Extracción de características
   - Comparación entre signatures
3. Resultados
   - Eficiencia
   - Tiempos

Objetivos

El objetivo principal del proyecto es la implementación de un sistema de reconocimiento biométrico a partir de los rasgos faciales. Se hace con las siguientes premisas:

- **Sencillez**: La cara normalmente suele ser una región del cuerpo expuesta a la vista.
- **Economía de componentes/uso**: El sistema puede desarrollarse sobre plataformas económicas.
- **Extensibilidad**: Se ha buscado modularidad y extensibilidad durante el desarrollo.

Biometría

La biometría consiste en el reconocimiento de individuos a partir de una o varias características físicas de su cuerpo. La eleccción del rostro como medida biométrica tiene ventajas e inconvenientes.

Ventajas:

- Sencilla de captar con hardware económico
- Comodidad de uso por parte del usuario

Inconvenientes:

- Alta variabilidad de la cara de un mismo individuo
- Posible repetición (hermanos gemelos)
- Fácil de falsificar (fotografías)

Software empleado y licencias

El software implementado para el proyecto tiene por licencia GPLv3

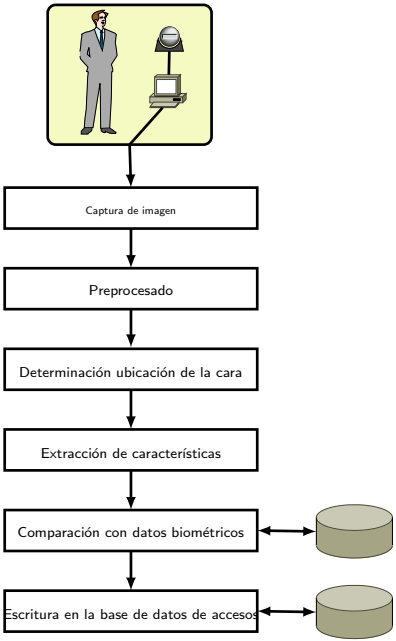
La documentación (incluída esta presentación) se libera bajo licencia Creative Commons - By - Shared Alike

Principalmente se ha utilizado el siguiente software:

- **Sistema Operativo**: GNU/Linux (GPL)
- **Lenguaje de programación**: Python 2.6 (compatible GPL)
- **Librerías empleadas**: OpenCV (BSD), GTK (GPL)
- **Otros**: LaTeX (GPL), GIMP (GPL), gnuplot (compatible GPL)...

En cuanto al Hardware empleado ha sido un ordenador doméstico y una webcam económica.

Esquema de funcionamiento



Paso a escala de grises

Se trabaja con la imagen en escala de grises para mejores resultados y mayor velocidad. Para cada punto en color RGB se aplica la siguiente función (siendo R,G,B las componentes roja, verde y azul y A la imagen resultado):

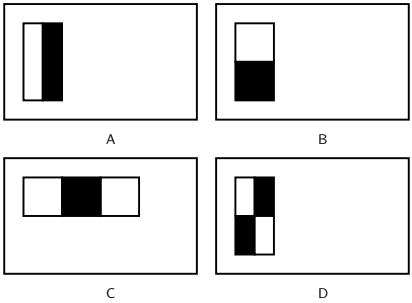
$$A_{x,y} = 0{,}3 * R_{x,y} + 0{,}59 * G_{x,y} + 0{,}11 * B_{x,y}$$

Ecualización del histograma

La ecualización del histograma es una transformación lineal de I a I' donde máx$(I') = 255$ y mín$(I') = 0$ para valores $I_{x,y} \in [0 \ldots 255]$. A la derecha, la imagen original y su histograma debajo, a la izquierda, imagen e histograma ecualizados:

Clasificadores de Haar

- Para localizar la cara en una imagen se han utilizado clasificadores de Haar.
- Se basan en el reconocimiento de patrones sobre la integral por zonas de una imagen.
- Requieren entrenamiento reconocimiento del mismo patrón en suficientes (> 1000) imágenes.
- Varios ejemplos de patrones son los siguientes: A reconocería bordes verticales, B bordes horizontales, C líneas verticales y D líneas diagonales.



Clasificadores de Haar

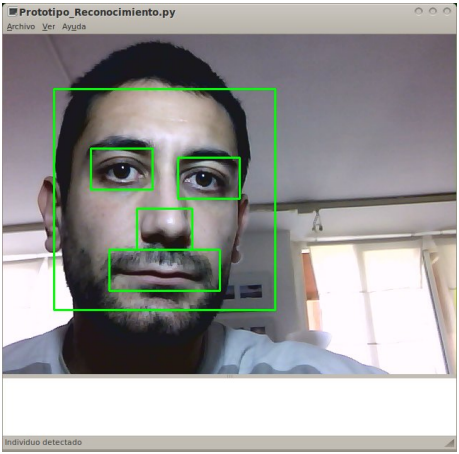


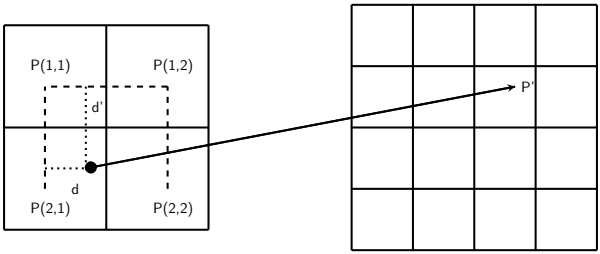
Figura: Ejemplo de detección con clasificadores de Haar

Escalado de imagen

Se redimensiona la imagen en escala de grises para

- Trabajar con imágenes de tamaño homogéneo, en este caso, 128x128 píxeles.
- Tener un control sobre el tamaño de las muestras

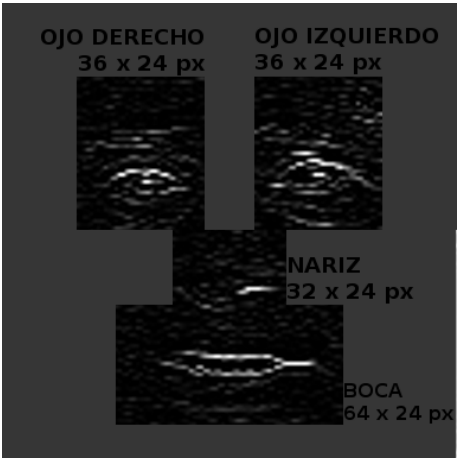
Se utiliza el algoritmo de escalado bilineal, en el que cada píxel toma un valor interpolado de sus tres vecinos.



Localización de rasgos

Para encontrar los rasgos (ojo derecho, ojo izquierdo, nariz y boca) de la cara:

- Aplicamos un Sobel (segunda derivada) en sentido Y sobre la imagen
- Se busca la ventana de valor máximo para cada rasgo según la siguiente imagen.



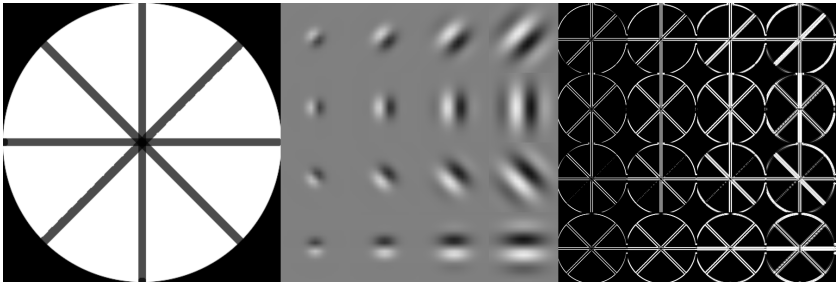
Localización de rasgos

Figura: Imagen con la cara detectada. A la derecha, de arriba a abajo, escalado, sobel, resultado de detección y encuadre sobre la imagen escalada.

Transformada de Gabor

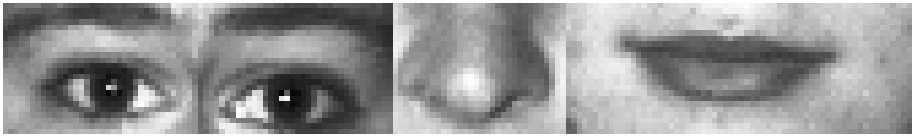
Un banco de filtros de Gabor es una herramienta de análisis frecuencial para:

- Emula el comportamiento de las células del córtex cerebral que se encargan de la visión
- Es un banco de kernels resultantes de convolucionar una gaussiana y un plano senoidal orientado con un ángulo ψ
- Filtra la respuesta frecuencial convolucionando los kernels anteriores con la imagen



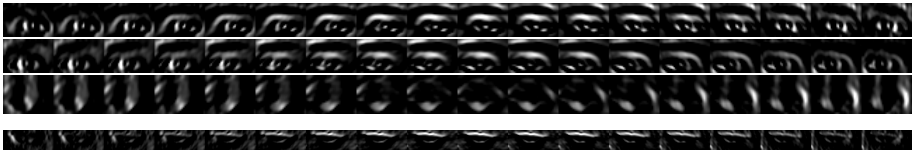
Transformada de Gabor

- Se ha aplicado una transformada de Gabor con kernel de tamaño 9 e inclinaciones $\psi \in 0 \ldots 180 \mod 10 = 0$
- Tras la aplicación, existen dos maneras de almacenar el resultado de los bancos de filtros: Concatenados y superpuestos.
- Se usarán los siguientes rasgos para mostrar un ejemplo de cada uno:



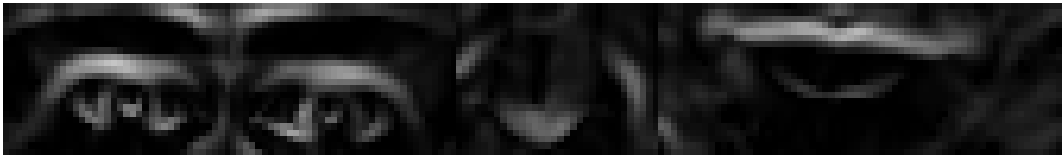
Concatenación de filtrados

- El resultado de cada filtrado se concatena al lado del anterior
- Mejores resultados en comparación
- Incrementa en $|\psi| * r$ (donde r es el ancho del rasgo) el tamaño de la signature y el tiempo necesario para compararla
- Tamaño total de signature: $18 * \sum_{\forall r} W_r$ para cada rasgo r
- Descartada por tamaño e incompatibilidad en caso de cambio de número de orientaciones



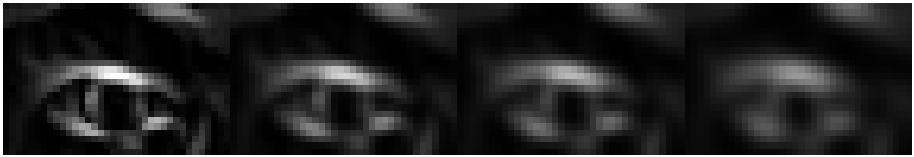
Superposición de filtrados

- El resultado de cada filtrado se suma ponderadamente
- Resultados ligeramente peores en comparación
- Para cada rasgo r el tamaño de la signature es el mismo que el del rasgo del que se obtiene
- Tamaño total de signature: $\sum_{\forall r} W_r$ para cada rasgo r
- Elegida por la velocidad de comparación, tamaño y relación fiabilidad/tiempos de cómputo



Suavizado

- El resultado de la superposición de signatures se ha suavizado para mejor respuesta ante oscilaciones (movimientos y similares)
- Suavizar una imagen consiste en convolucionar con una gaussiana. En este caso, el kernel con el que se ha convolucionado la imagen ha sido de tamaño 3x3.
- En la imagen se puede comprobar a la izquierda, la signature original, y de izquierda a derecha, la suavización con una gaussiana de 3x3, 5x5 y 7x7, respectivamente



Binarización

- La binarización se define como que por cada píxel de la imagen I, con un umbral T y un valor máximo M :

$$thr(I,T,M) = \begin{cases} 0, & \text{si } I_{x,y} < T \\ M, & \text{si } I_{x,y} \geq T \end{cases}$$

- Esto hace más cómoda la comparación y nos permitiría almacenar la signature únicamente con dos valores binarios (no se ha podido realizar). El valor de umbral aplicado ha sido $T = 16$ y el máximo $M = 255$
- En la imagen se ve el resultado de binarizar la imagen anterior con valores $T = (0, 30, 60, 90)$ en la primera fila y $T = (120, 150, 180, 210)$ en la segunda

Distancia Euclídea

- Es una medida de disimilitud (a mayor valor, menor similitud) entre signatures basada en la distancia euclídea ponderada.
- Se calcula según la siguiente fórmula para cada píxel

$$D_{euc}(m,n) = \frac{\sum_{i,j=0}^{X,Y} \parallel m_{i,j} - n_{i,j} \parallel}{XY}$$

  donde $\parallel a-b \parallel$ representa el valor absoluto de la resta $a-b$ y X, Y son el ancho y el alto de la signature, respectivamente
- $D_{euc}(m,n) \in \mathbb{R}_{0,255}$, es decir, el resultado de aplicarla es un número real entre 0 y 255, truncable a un valor natural $D_{euc}(m,n) \in \mathbb{N}_{0,255}$
- Sencilla y rápida de calcular. Se ha escogido como medida de distancia entre signatures

Distancia XOR/OR

- Es una medida de disimilitud entre signatures (a mayor valor, menor similitud) calculada a partir de la división de los sumatorios del resultado de la XOR y la OR de las signatures.
- Se calcula según la siguiente fórmula para cada píxel

$$D_{xor}(m,n) = \frac{\sum_{x,y=0}^{X,Y}(m_{x,y} \oplus n_{x,y})}{\sum_{x,y=0}^{X,Y}(m_{x,y} \vee n_{x,y})}$$

  donde X, Y es el tamaño de la imagen, $\oplus$ simboliza la XOR bit a bit y $\vee$ simboliza la OR bit a bit de cada píxel de la imagen
- El resultado que da está en el rango $D_{xor}(m,n) \in \mathbb{R}_{0,1}$
- Funcionamiento correcto, de hecho similar o mejor para filtrados concatenados que la D_{euc}

Distancia AND/OR

- Es una medida de similitud entre signatures (a menor valor, mayor similitud) calculada a partir de la división de los sumatorios del resultado de la AND y la OR de las signatures.
- Se calcula según la siguiente fórmula para cada píxel

$$D_{and}(m,n) = \frac{\sum_{x,y=0}^{X,Y}(m_{x,y} \wedge n_{x,y})}{\sum_{x,y=0}^{X,Y}(m_{x,y} \vee n_{x,y})}$$

  donde X, Y es el tamaño de la imagen, $\wedge$ simboliza la AND bit a bit y $\vee$ simboliza la OR bit a bit de cada píxel de la imagen
- El resultado que da pertenece al rango $D_{and}(m,n) \in \mathbb{R}_{0,1}$
- Funcionamiento inusualmente correcto, sobretodo para los filtrados superpuestos

Gráficas TFA/TFE

- La fiabilidad del sistema se mide en gráficas TFA/TFE.
- TFA: Tasa de falso acierto (probabilidad de confundir un individuo con otro)
- TFE: Tasa de falso error (probabilidad de no reconocer a un individuo dado de alta)

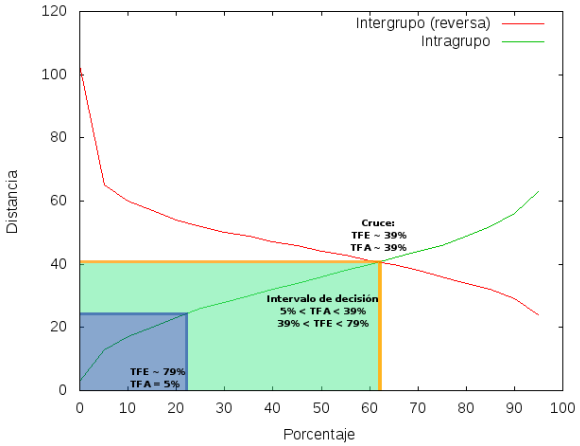


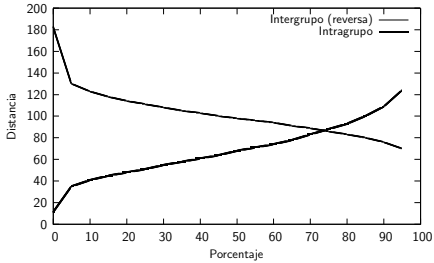
Tabla de eficiencia

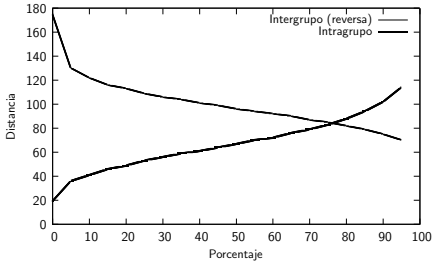
Tras el cálculo de la eficiencia para las variantes aplicadas, se llegó a la conclusión que la mejor opción era el uso de la distancia euclídea con filtrados superpuestos. El valor de umbral 16 es el que da mejores resultados:

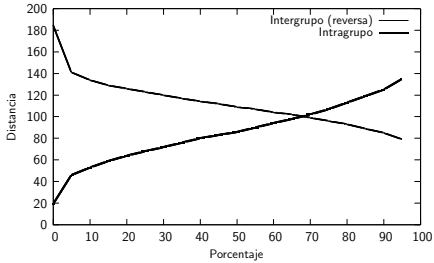
Distancia	Rasgo	Umbral	Punto de corte	%TFA	%TFE
Euclídea con filtrados superpuestos	Ojo derecho	8	71.00	$\approx 25{,}00$	$\approx 25{,}00$
		16	87.00	$\approx 25{,}00$	$\approx 25{,}00$
		32	86.50	$\approx 25{,}00$	$\approx 25{,}00$
		64	58.50	$\approx 30{,}00$	$\approx 25{,}00$
	Ojo izquierdo	8	70.50	$\approx 25{,}00$	$\approx 25{,}00$
		16	85.00	$\approx 20{,}00$	$\approx 20{,}00$
		32	83.00	$\approx 25{,}00$	$\approx 20{,}00$
		64	55.00	$\approx 25{,}00$	$\approx 25{,}00$
	Nariz	8	68.50	$\approx 35{,}00$	$\approx 35{,}00$
		16	100.50	$\approx 30{,}00$	$\approx 30{,}00$
		32	100.50	$\approx 30{,}00$	$\approx 30{,}00$
		64	52.00	$\approx 35{,}00$	$\approx 35{,}00$
	Boca	8	94.00	$\approx 35{,}00$	$\approx 35{,}00$
		16	112.50	$\approx 35{,}00$	$\approx 35{,}00$
		32	94.50	$\approx 35{,}00$	$\approx 35{,}00$
		64	48.00	$\approx 40{,}00$	$\approx 40{,}00$

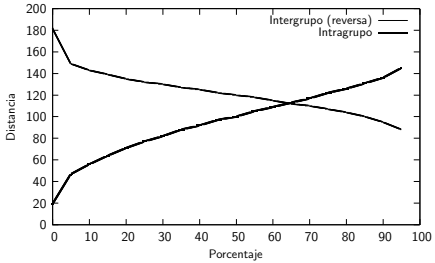
Gráficas de eficiencia

Gráficas de eficiencia para filtrados superpuestos, umbral 16 y distancia euclídea. Rasgos: ojo derecho, ojo izquierdo, nariz y boca









Tiempos

Tiempos de extracción de la face signature en un AMD Athlon(tm) 64 X2 6000+ con 3800340 kB de RAM y kernel Linux 2.6.31-22 x86_64 (Ubuntu SMP)