UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉNCNICA SUPERIOR

DETECCIÓN, LOCALIZACIÓN E IDENTIFICACIÓN BIOMÉTRICA DE CARAS EN IMÁGENES: MÉTODOS Y EVALUACIÓN EN EL MARCO NIST-MBGG

Proyecto Fin de Carrera

Ingeniería de Telecomunicación

Andrei Bogdan Dragolici Dragolici

Junio 2010

DETECCIÓN, LOCALIZACIÓN E IDENTIFICACIÓN BIOMÉTRICA DE CARAS EN IMÁGENES: MÉTODOS Y EVALUACIÓN EN EL MARCO NIST-MBGG

AUTOR: Andrei Bogdan Dragolici
TUTOR: Pedro Tomé Gonzalez

Área de Tratamiento de Voz y Señales
Dpto. de Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Junio 2010

Resumen

A lo largo de este proyecto se estudia, implementa y evalúa un detector facial y un detector de ojos en imágenes. Dicho sistema ha sido implementado basándonos en el estado del arte y eligiendo un modelo de implementación considerado adecuado, el de Viola y Jones [1].

Como punto de partida se realiza un estudio y evaluación de los sistemas biométricos y un estudio del estado del arte de los sistemas de detección facial desarrollados hasta el momento, desarrollando finalmente el mecanismo seleccionado.

El objetivo principal del proyecto es la implementación de un algoritmo capaz de detectar objetos (caras) sobre imágenes además de poder ser evaluado en diferentes condiciones y entornos de trabajo.

Para llevar a cabo dicha evaluación se han utilizado las imágenes de las bases de datos CMU y NIST comparando el sistema implementado con sistemas de libre distribución que ya han mostrado un rendimiento competitivo y que son una referencia en el ámbito de la detección facial.

Palabras Clave

Detección Facial, biometría, detección de ojos, procesamiento de imágenes, reconocimiento de patrones, Viola, Jones, OpenCV.

Abstract

This Master of Science Thesis studies, implements and tests a face detection and an eye detection framework on images. This system has been implemented basing on the State of the Art, and by selecting an implementation model, the Viola and Jones one.

The thesis starts by studying and evaluating biometric systems in general and then, the face detection frameworks implemented until now. It is followed by the development of the selected mechanism.

The main target of this project is the implementation of an algorithm which can detect objects (faces) on images and which can be tested on different situations and environments.

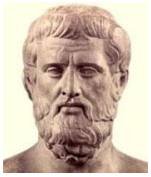
In order to accomplish the testing process, CMU and NIST images were used and the developed system was compared with open source systems which have shown a competitive performance and which are a reference among facial detection frameworks.

Key words

Face detection, biometrics, eye detection, image processing, pattern recognition, Viola, Jones, OpenCV

Agradecimientos

"El que sabe corresponder a un favor recibido es un amigo que no tiene precio". ***Sófocles***



Llegar al final de un largo camino da la posibilidad única de mirar hacia atrás y poder reflexionar sobre el camino recorrido, recordando los buenos y malos momentos, las alegrías y las tristezas, y sobre todo agradecer los apoyos recibidos.

En primer lugar quiero agradecer a mi ponente, Javier Ortega, que me haya abierto las puertas y me haya recibido en la ATVS para realizar un proyecto en un tema que me gusta y apasiona.

Sin embargo, siempre es necesaria una luz en medio de la oscuridad, y esa luz ha sido mi tutor, Pedro Tomé, al que no puedo dejar de agradecerle el apoyo y guía a lo largo de estos meses. Gracias. Gracias también a todos aquellos compañeros que me han acompañado durante estos años tanto en España como en mi año de estudios en Finlandia. Nada hubiese sido igual sin vuestra luz.

Por último quiero agradecer el apoyo de mi familia y amigos desde el comienzo de mis estudios y durante toda mi carrera universitaria.

Gracias.

Andrei Bogdan Dragolici
Junio 2010

El trabajo de investigación que ha dado lugar a este Proyecto de Fin de Carrera fue desarrollado en el *Área de Tratamiento de Voz y Señales*, del Departamento de Ingeniería Informática de la Escuela Politécnica Superior en la Universidad Autónoma de Madrid.

CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1

INTRODUCCIÓN

1.1 Motivación del proyecto

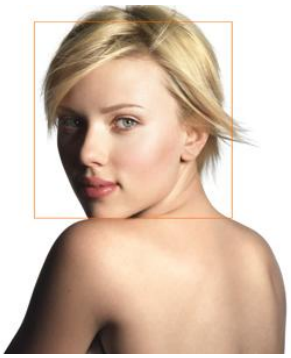
En la sociedad actual, dónde las tecnologías de la información (TI) ocupan un papel cada vez más importante, los requerimientos de seguridad son cada vez mayores y más demandados. Esto ha dado lugar a un gran desarrollo de los sistemas automáticos de identificación personal y de las técnicas biométricas como un método muy útil a la hora de implementar estos sistemas. Actualmente estos métodos tienen cada vez mayor relevancia ya que la biometría es una forma muy sencilla y en general incluso única para identificar a las personas. Los rasgos biométricos suelen ser más difíciles de duplicar o falsificar frente a otros métodos anteriormente utilizados como por ejemplo un PIN, una llave, etc. Esto se debe a que estos rasgos utilizan características propias de cada individuo y únicas de modo que pueden diferenciarlo de otras personas que intenten falsificar su identidad.

Los rasgos biométricos pueden ser divididos en rasgos fisiológicos, como pueden ser las huellas dactilares, el iris, el ADN, etc. o rasgos del comportamiento, como la firma, el modo de andar, la voz... Mientras que los primeros tienen una menor variabilidad a lo largo del tiempo, son también mucho más invasivos y requieren de la cooperación del individuo para su obtención.

Dentro de los rasgos biométricos el reconocimiento facial se ha convertido en un área de investigación bastante popular. Debido a la naturaleza del problema, no sólo los investigadores del ámbito de la ingeniería están interesados en él, sino además otros del ámbito de la neurociencia y la psicología.

El primer paso para el reconocimiento facial es la detección de caras dentro de imágenes. A partir de la detección es cuando podemos crear cualquier sistema que analice la información contenida en las caras: identidad, género, expresión... La detección facial se encarga de determinar si hay o no alguna cara en una imagen dada y, en caso de que exista, de extraer la localización y el contenido de dicha cara. En

general, este es un problema muy complejo ya que los objetos a detectar pueden ser de diversos colores, expresiones, poses, en diferentes situaciones de iluminación, etc.



La detección de caras lleva desarrollándose desde hace algún tiempo. En la década de los 70 surgieron los primeros algoritmos pero las investigaciones se abandonaron debido a la falta de utilidad debida a la falta de desarrollo de la técnica. Sin embargo, en la actualidad esta línea de investigación tiene numerosas aplicaciones tanto en el ámbito comercial con las cámaras de fotos, como en el de la seguridad y la investigación criminal, el control a zonas restringidas, el control de fronteras, etc. Por otro lado, la mayoría de los algoritmos de detección facial pueden extenderse para el procesamiento de otros objetos como coches, peatones, etc., o para la localización de otras áreas faciales.

1.2 Objetivos y enfoque

El presente proyecto se centrará en el desarrollo e implementación de un detector facial. A partir de este punto, se extenderá hacia la detección de ojos. La importancia de este detector se halla en el hecho de que forma el primer paso para otras investigaciones en las que la localización de una cara en una imagen, en caso de existir, nos da pie a una posible identificación biométrica, una extracción de otras áreas faciales, etc.

El objetivo propuesto es, en primer lugar, estudiar, desarrollar, implantar y documentar un algoritmo que permita la detección facial en base al estado del arte actual. A partir de ahí se procederá a una extracción de ojos y a la evaluación de ambos sistemas sobre las bases de datos CMU y NIST.

1.3 Metodología y plan de trabajo

Con el fin de cumplir los objetivos de este Proyecto de Fin de Carrera, se ha seguido una metodología de trabajo que se esquematiza a continuación.

El plan de trabajo se divide en 4 líneas paralelas con diferente inicio temporal:

- **Formación:** Es la primera parte y la fundamental de este proyecto ya que a partir de ella se adquieren los conocimientos necesarios para obtener los mejores resultados en el trabajo. El primer paso consistió en la lectura de publicaciones científicas en el tema en cuestión para poder conocer y entender el estado del arte en los sistemas de detección facial. Posteriormente

hubo una formación más específica en cuanto a la detección de otras partes de la cara.

- **Investigación:** Ha sido un punto importante en el desarrollo del proyecto. Comenzó tras el punto anterior y consiste en el entendimiento de las técnicas utilizadas y el mejoramiento a la hora de la implementación.

- **Desarrollo:** El objetivo final es obtener una implementación práctica de un sistema de detección. En un primer paso se llevó a cabo el desarrollo de un entrenamiento del detector, seguido de la detección y de una evaluación de su calidad mediante distintos experimentos y bases de datos.

- **Escritura:** Este ha sido el último paso del proyecto pero que se ha llevado a cabo durante toda la duración del mismo.

Por último, en la se incluye el plan de trabajo seguido para la realización de este proyecto, en el que de forma global se detallan las distintas partes en que ha consistido la evolución de las mismas a lo largo del periodo de trabajo.

	Mar09	Abr09	May09	Jun09	Jul09	Ago09	Sep09	Oct09
1ª Fase: Lectura y documentación								
Estudio del estado del arte en rec. Biométrico.								
Estudio del estado del arte en detección facial.								
Familiarización con el procesado de imágenes.								
Obtención de bases de datos de imágenes de caras.								
Familiarización con los archivos MEX.								
2ª Fase: Diseño y desarrollo								
Implementación en Matlab.								
Implementación con archivos MEX.								
3ª Fase: Realización de pruebas								
Evaluación de los resultados y conclusiones								
Documentación de los experimentos realizados								
4ª Fase: Documentación del PFC								
Escritura del Proyecto de Fin de Carrera								
Revisión y fe de erratas								
5ª Fase: Preparación de la presentación								
Defensa del Proyecto de Fin de Carrera								

	Nov09	Dic09	Ene10	Feb10	Mar10	Abr10	May10	
1ª Fase: Lectura y documentación								
Estudio del estado del arte en rec. Biométrico.								
Estudio del estado del arte en detección facial.								
Familiarización con el procesado de imágenes.								
Obtención de bases de datos de imágenes de caras.								
Familiarización con los archivos MEX.								
2ª Fase: Diseño y desarrollo								
Implementación en Matlab.								
Implementación con archivos MEX.								
3ª Fase: Realización de pruebas								
Evaluación de los resultados y conclusiones								
Documentación de los experimentos realizados								
4ª Fase: Documentación del PFC								
Escritura del Proyecto de Fin de Carrera								
Revisión y fe de erratas								
5ª Fase: Preparación de la presentación								
Defensa del Proyecto de Fin de Carrera								

Figura 1.1 Plan de trabajo

2

INTRODUCCIÓN A LA BIOMETRÍA

Establecer la identidad de una persona se está convirtiendo cada vez más en un asunto crítico dentro de nuestra sociedad. Preguntas como: *"¿Es una persona realmente quien dice ser?"* están siendo propuestas cada vez más y en una gran variedad de escenarios, desde el acceso a un edificio hasta la entrada en un país. La necesidad de técnicas de autenticación fiables ha aumentado en los últimos tiempos debido al incremento de la concienciación sobre los temas de seguridad y el rápido avance en temas de redes, comunicaciones y movilidad.

La biometría, descrita como la ciencia orientada al reconocimiento de un individuo basándose en sus rasgos físicos o de comportamiento, está convirtiéndose en un método cada vez más aceptado a la hora de determinar la identidad de un individuo [2].

2.1 Características de los rasgos biométricos

Al modernizarse nuestra forma de vida, también han aumentado los requerimientos en materia de seguridad, así como las posibilidades que se nos presentan en este campo. Las personas siempre han utilizado características corporales como la cara, la voz, etc., para reconocerse mutuamente. Fue Alphonse Bertillon, jefe de la división de identificación criminal del departamento de policía de París, quien desarrolló y puso en práctica la idea de utilizar medidas corporales para identificar a los criminales en el siglo XIX [3]. Esto dio paso más tarde a la creación de bases de datos en las que se guardasen las huellas dactilares de los criminales.

Aunque los avances en biometría fuesen en un primer momento ideados con fines forenses para la identificación de delincuentes, en la actualidad se utiliza en aplicaciones civiles muy variadas abriéndose así a un mercado mucho mayor.

El aumento de este campo y su uso en situaciones donde la fiabilidad es obligada, hace necesario establecer unos requisitos básicos a la hora de considerar un rasgo biométrico como tal [4] [5]:

- **Universalidad:** Cada individuo debe tener dicha característica.
- **Unicidad:** La característica debe ser lo suficientemente diferenciable entre los individuos sobre los que se aplica.
- **Permanencia:** La característica debe ser lo suficientemente estable para no cambiar significativamente con el tiempo o en distintos medios.
- **Mensurabilidad:** Debe ser posible adquirir y digitalizar la señal biométrica utilizando aparatos adecuados que no causen inconveniencias al individuo; y dicha señal debe ser medible.
- **Rendimiento:** La precisión de la identificación debe corresponderse a la requerida por la aplicación.
- **Aceptabilidad:** Indica el grado de tolerancia de la sociedad.
- **Fiabilidad o fraude:** Se refiere a la facilidad para burlar el sistema.

2.2 Rasgos biométricos

No se espera que un único rasgo biométrico cumpla todos los requerimientos (por ejemplo, precisión, coste, etc.) impuestos por todas las aplicaciones (Digital Rights Management (DRM), control de accesos...). En otras palabras, ningún rasgo biométrico es *ideal*, pero un gran número son *admisibles*. A continuación se da una breve explicación de un gran número de estos rasgos y en la tabla 2.1 se muestra el grado de cumplimiento de las características anteriores para algunos de ellos.

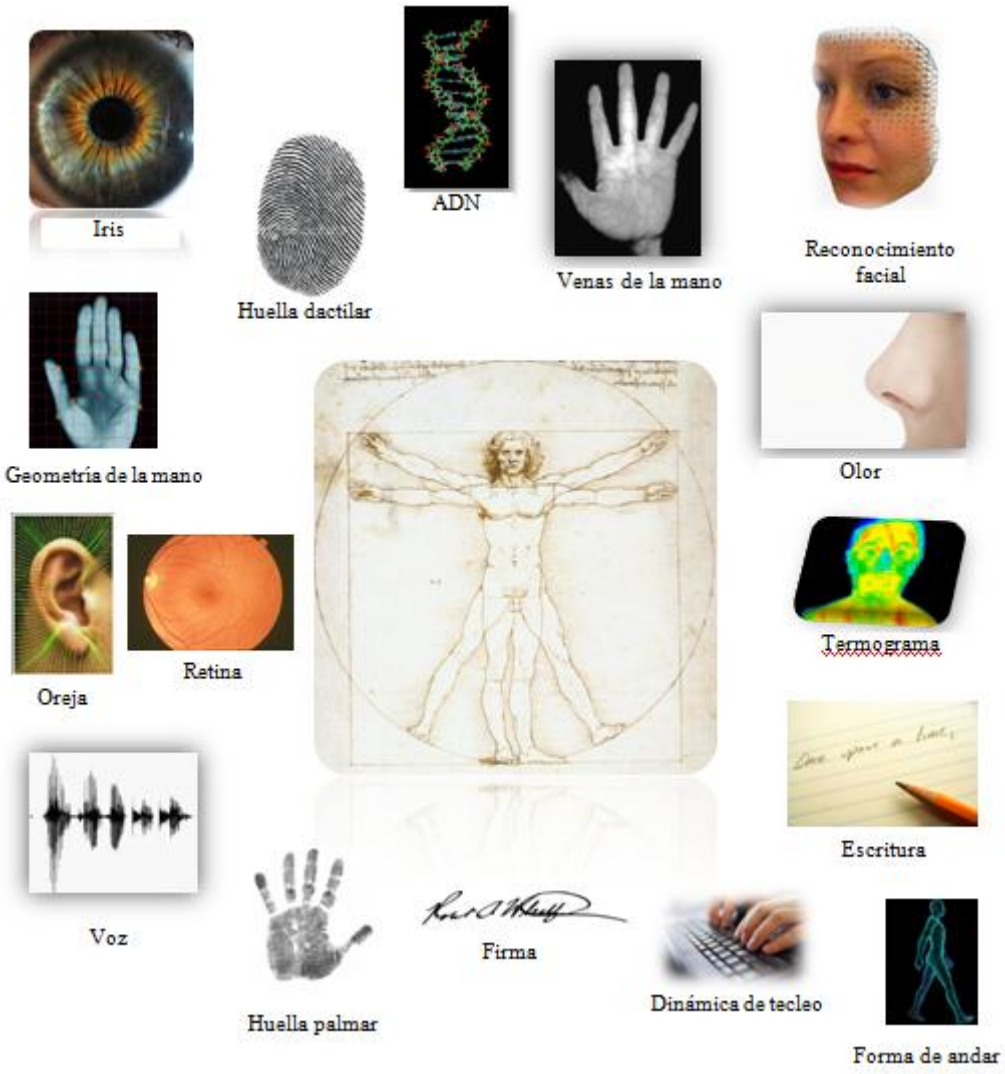
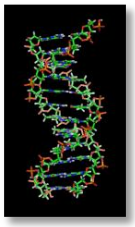


Figura 2.1 Rasgos biométricos en humanos.



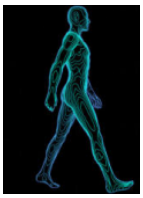
ADN. El ADN (ácido desoxirribonucleico) es un código unidimensional único para cada persona, excepto por el hecho de que los gemelos idénticos tienen patrones ADN idénticos. A pesar de esto, es utilizado en la mayoría de los contextos de aplicaciones forenses para el reconocimiento de personas. Tres hechos limitan el uso de estas técnicas para otras aplicaciones: contaminación y sensibilidad, es fácil de robar; problemas de reconocimiento automático en tiempo real; y problemas de

privacidad, ya que la información sobre ciertas enfermedades puede ser obtenida del ADN [3].

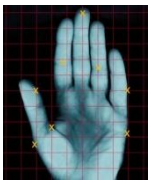
Dinámica de tecleo. Se cree que cada persona teclea de una manera característica. Este rasgo biométrico no se cree que sea único para cada individuo pero ofrece suficiente información discriminatoria para permitir una verificación de identidad [3] [4].

Escritura. La escritura es un rasgo biométrico de comportamiento, por lo que es bastante usual que no se mantenga invariable a lo largo del tiempo. Estos cambios en la escritura hacen que, a pesar de que sea un método poco invasivo, no sea un rasgo completamente discriminatorio. A pesar de que un experto pueda identificarla, en la misma medida, puede ser copiado por otro [6].

Firma. La identificación por la manera de firmar es una manera relacionada con la identificación por escritura. En este caso hay una diferencia ya que el texto es siempre el mismo. La manera en que una persona firma es característica para cada persona. A pesar de que la firma requiere contacto físico con un instrumento y la cooperación del individuo, ha sido ampliamente aceptado en transacciones gubernamentales, legales y comerciales como un método de autenticación [3] [4].



Forma de andar. La forma de andar no es un método muy distintivo, pero es suficientemente discriminatorio para permitir la verificación en algunas aplicaciones de bajos requerimientos de seguridad. Sin embargo, la forma de andar es un rasgo biométrico de comportamiento con lo que puede no mantenerse invariable a lo largo de un largo periodo de tiempo. Ya que los sistemas basados en la manera de andar utilizan secuencias de video de la longitud de los pies de una persona andando para medir diferentes movimientos de cada articulación, es computacionalmente cara [3].

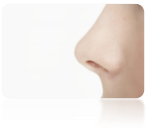


Geometría de la mano. Los sistemas de reconocimiento por la geometría de la mano están basados en un gran número de medidas tomadas a partir de la mano humana, incluyendo su forma, tamaño de la palma y longitud y anchura de los dedos. La técnica es muy simple, relativamente fácil de usar y no muy cara. Factores medioambientales como un clima seco o anomalías individuales como piel seca no parecen tener ningún efecto negativo sobre la precisión de estos sistemas. Adicionalmente, la geometría de la mano no es muy distintiva y estos sistemas no pueden desempeñar tareas de identificación de un individuo entre una gran población. Sin embargo, comercialmente ha tenido éxito y ha sido instalado en cientos de lugares.

Huella dactilar. Las huellas dactilares son surcos que se encuentran en los dedos de los seres humanos [5]. Los seres humanos han utilizado las huellas dactilares para realizar una identificación personal desde hace siglos y la precisión demostrada ha sido muy alta [7]. Las huellas dactilares de gemelos son diferentes. Así como las de diferentes de dos de una misma persona. La precisión de los sistemas de reconocimiento por huella dactilar es adecuada para sistemas de verificación y sistemas de identificación de tamaño medio con unos cuantos cientos de individuos. Múltiples huellas dactilares de una persona proveen información adicional para permitir el reconocimiento en sistemas a gran escala con varios millones de identidades. Un problema que presenta esta técnica es que requiere grandes cantidades de recursos de computación, especialmente en el modo de identificación [3]. Por otro lado, cabe destacar que estos sistemas son cada vez más accesibles para el gran público y algunos aparatos electrónicos los traen incluidos (PDAs, portátiles, etc.).

Huella palmar. La huella palmar, al igual que la dactilar, se considera una de las maneras más eficaces para identificar a un individuo, debido a su estabilidad y unicidad. El área de la palma de la mano es mucho más grande que el de un dedo, y es por ello que la huella palmar parece ser incluso más útil a la hora de distinguir individuos que la huella dactilar [3].

Iris. El iris es la región anular del ojo comprendida entre la pupila y el blanco del ojo. La textura visual del iris se forma durante el desarrollo fetal y se estabiliza durante los dos primeros años de vida. La exactitud y velocidad de los actuales sistemas de reconocimiento basados en el iris son muy prometedoras y hacen pensar en sistemas de identificación a gran escala. Se cree que cada iris es diferente, incluso en gemelos idénticos. A pesar de que estos sistemas necesitasen una considerable participación del usuario y fuesen caros, los nuevos sistemas se han vuelto más fáciles de usar y baratos [2].

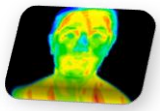


Olor. Es bien sabido que cada objeto emite un olor que es característico por su composición química y que podría ser utilizado para distinguir varios objetos. Un componente del olor emitido por una persona (o animal) es distintivo para un individuo en particular. Sin embargo, no está claro si la invariabilidad del olor del cuerpo podría ser detectada a pesar de los olores del desodorante o la variante composición química del medio [3].

Oreja. Se ha sugerido que la forma de la oreja y la estructura del tejido cartilaginoso del pabellón auditivo son características de cada persona. El reconocimiento se basa en verificar la distancia de los puntos salientes del pabellón de la oreja con respecto a una referencia común del interior de la misma [3].

Reconocimiento facial. El reconocimiento facial es un método no intrusivo y las imágenes faciales son probablemente las características biométricas más utilizadas por los seres humanos para hacer un reconocimiento personal. Los métodos utilizados más populares son: 1) la localización y forma de los atributos faciales, como ojos, cejas, nariz, labios y mentón y la distancia entre estos o 2) el análisis global de la imagen de la cara que representa una cara como una combinación ponderada de un cierto número de caras canónicas. A pesar de que estos sistemas están actualmente disponibles en el mercado, algunos requieren unas condiciones especiales de pose, fondo o iluminación. Para que un sistema de reconocimiento facial trabaje bien en la práctica, debería automáticamente: 1) detectar si hay una cara presente en la imagen; 2) localizar la cara, en caso de que haya una; y reconocer la cara desde un punto de vista general, es decir, desde cualquier pose [2].

Retina. El escáner de retina muestra una estructura vascular abundante que se supone que es única en cada individuo y ojo. Se dice que es el rasgo biométrico más seguro ya que no es fácil de cambiar o copiar. La adquisición necesita bastante colaboración y un contacto con el aparato además de un esfuerzo por parte del individuo ya que se adquiere cuando un individuo mira dentro de un aparato y fija su mirada en un punto dentro del campo visual para que una parte predeterminada de la estructura vascular sea extraída [3].



Termograma. El patrón de calor radiado por el cuerpo humano es característico de cada individuo y puede ser capturado por una cámara de infrarrojos de un modo parecido a una fotografía. Esta tecnología puede usarse para un reconocimiento encubierto. El sistema no requiere contacto y es no invasivo, pero la adquisición de la imagen es difícil en medios incontrolados donde haya elementos que emanen calor en las proximidades del cuerpo humano. Una tecnología similar se utiliza para escanear la parte trasera de la mano con el puño cerrado para determinar la estructura de las venas de la mano. Sin embargo, estos escáneres son tremendamente caros y esto hace que la implantación de esta tecnología no sea muy amplia [3].

Voz. La voz es una combinación de rasgos físicos y de comportamiento. Las características físicas se mantienen invariantes a lo largo del tiempo, pero las de comportamiento varían debido a la edad, problemas de salud, estados de ánimo, etc. [2].

Cada uno de los métodos propuestos anteriormente presenta tanto ventajas como inconvenientes. Unos son más o menos fiables, otros más o menos intrusivos, presentan diferentes niveles de resistencia ante ataques, etc. Para vencer los problemas de un determinado método lo que podemos hacer es utilizar distintas fuentes de información biométrica, dando así lugar a la multibiometría. Esta puede ser vista como distintas muestras de un mismo rasgo, múltiples rasgos, distintos algoritmos de reconocimiento...

Tabla 2.1 Comparación de varias tecnologías biométricas. H, M y L denotan Alta, Media y Baja, respectivamente [3].

Biometric identifier	Universality	Distinctiveness	Permanence	Collectability	Performance	Acceptability	Circumvention
DNA	H	H	H	L	H	L	L
Ear	M	M	H	M	M	H	M
Face	H	L	M	H	L	H	H
Facial thermogram	H	H	L	H	M	H	L
Fingerprint	M	H	H	M	H	M	M
Gait	M	L	L	H	L	H	M
Hand geometry	M	M	M	H	M	M	M
Hand vein	M	M	M	M	M	M	L
Iris	H	H	H	M	H	L	L
Keystroke	L	L	L	M	L	M	M
Odor	H	H	H	L	L	M	L
Palmprint	M	H	H	M	H	M	M
Retina	H	H	M	L	H	L	L
Signature	L	L	L	H	L	H	H
Voice	M	L	L	M	L	H	H

2.3 Variación biométrica

Vamos a examinar seguidamente como puede afectar la variación de los rasgos biométricos al uso de los mismos, a su fiabilidad y al cumplimiento de su función.

En el caso de un sistema de autenticación por medio de *password* estos problemas no aparecen, ya que el código requerido es siempre el mismo y tiene que coincidir completamente con el almacenado. Por otro lado las señales biométricas y su representación pueden variar considerablemente dependiendo de varios factores: método de adquisición, medio en el que se produce dicha adquisición, tipo de interacción con el aparato que realiza dicha adquisición y en algunos casos también varían algunos de los rasgos que se utilizan. Algunos de las razones más comunes para la variación de la señal o representación biométrica son:

I. **Presentación inconsistente.** La señal capturada por el sensor depende tanto de las características del aparato, como del rasgo en sí y de la interacción del individuo con dicho aparato. Por ejemplo, la forma tridimensional de un dedo se plasma sobre la superficie bidimensional de un sensor. Con lo cual, al no ser el dedo un objeto rígido y al no colocarlo exactamente igual sobre dicho sensor en cada interacción, tenemos varias representaciones de un mismo dedo. Extendiendo esto a la extracción de datos de la cara nos encontraríamos con un problema mucho más grande.

II. **Presentación irreproducible.** Los rasgos biométricos pueden ser objeto de cambios y modificaciones. Los trabajos manuales o los accidentes pueden afectar a las huellas dactilares, un catarro puede afectar a la voz, el vello facial a la cara, etc.

III. **Adquisición de señal y/o representación imperfecta.** Las condiciones de adquisición de un determinado rasgo biométrico no son perfectas en la práctica y causan extrañas variaciones en la señal adquirida. Por ejemplo, la falta de contacto uniforme con el sensor puede causar una mala extracción de la huella dactilar, distintas iluminaciones pueden provocar significantes variaciones en la extracción facial que conllevarían un posterior mal análisis, el ancho de banda del canal puede variar la señal de la voz.

Además los algoritmos de extracción son imperfectos e introducen errores de medición. Un rasgo biométrico perteneciente a dos individuos diferentes puede ser muy similar por la falta de información distintiva entre ambas muestras o debido a una representación inadecuada.

2.4 Biometría: aceptación y privacidad

Como en todo, el éxito o fracaso de una determinada tecnología depende de la aceptación social que tenga. La aceptación social conlleva renunciar a ciertos privilegios que los ciudadanos han tenido asegurados, como la posesión exclusiva de la información biológica de cada uno. A parte de esto, la facilidad al interactuar con un sistema biométrico contribuye enormemente a su aceptación. Cuanto menor sea la necesidad de un individuo de interactuar con un sistema biométrico para que este obtenga los datos necesarios, mayor será la aceptación de dicho sistema. Sin embargo, esto nos lleva a otro problema: la privacidad. Si podemos ser objeto de extracción de información biométrica sin nuestra colaboración, sin siquiera percibirlo, hasta qué punto lo seremos y para qué fin se utilizará esa información.

En este tema podemos encontrar tanto adeptos fieles como detractores feroces de una tecnología que cada vez se encuentra más en nuestra vida. Actualmente es un gran negocio para las empresas de alta tecnología y un tema muy importante para las agencias gubernamentales. En algunos países ya se ha implantado mucho más que en España, véase por ejemplo el caso de Bélgica y Alemania donde se tiene un pasaporte biométrico.

Los defensores de esta tecnología creen que la educación y la creación de una política de uso adecuada pueden eliminar todos los recelos. Sin embargo, hay un debate candente entre expertos y legisladores sobre qué políticas de uso podrían ser consideradas *adecuadas*. Según la biometría va apareciendo más y más en nuestras vidas, los temas relativos al potencial uso malintencionado de nuestros datos se hacen también más presentes.

3

SISTEMAS BIOMÉTRICOS

3.1 Funcionamiento general

En general, todos los sistemas biométricos poseen un funcionamiento similar. Un sistema biométrico es esencialmente un sistema de reconocimiento de patrones que adquiere datos biométricos de un individuo, extrae un conjunto de características de los datos, compara este conjunto con otro/s guardados en una base de datos y ejecuta una acción en función de la respuesta obtenida a partir de la comparación [4].

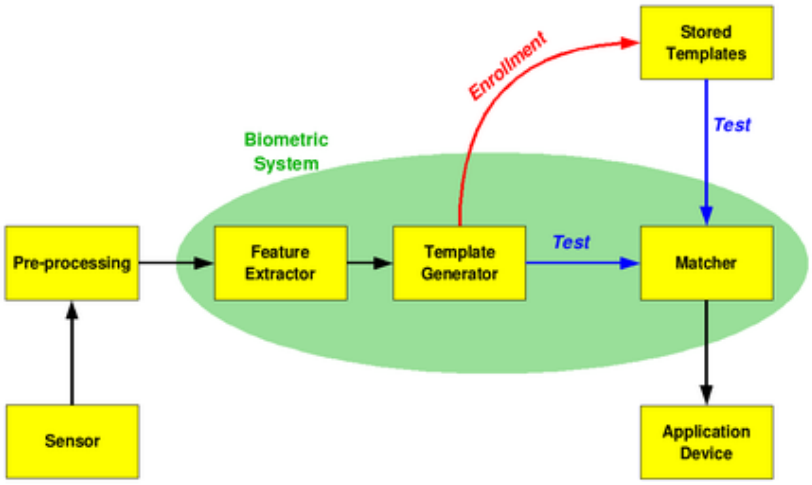


Figura 3.1 Estructura general de un sistema biométrico.

Vamos a proceder a la descripción del diagrama anterior para un mejor entendimiento de la cada uno de los pasos mostrados y del sistema en su conjunto.

- **Adquisición de datos (*sensor*).** En esta fase se necesita un sensor mediante el cual se puedan obtener los datos que nos interesan de un individuo. El tipo de sensor define el tipo de interacción entre el individuo y la máquina y es fundamental a la hora de definir el funcionamiento del sistema.

- **Preprocesado (*pre-processing*).** En ciertos casos es necesario adaptar la señal que hemos obtenido del sensor al modo de operación y el algoritmo que utilizamos, o simplemente utilizamos este paso para eliminar ruido, normalizar la señal o mejorarla.

- **Extracción de características (*feature extractor*).** Una vez obtenidos los datos en el formato adecuado, estos deben ser procesados para obtener un conjunto de características discriminatorias que representen el rasgo biométrico que nos interesa.

- **Generación de patrones (*template generator*).** Para poder realizar la etapa de registro dentro del sistema (*enrollment*), el conjunto de características se guarda dentro de la base de datos después de haberse generado una plantilla o patrón que será utilizada posteriormente para realizar las comparaciones necesarias para el funcionamiento del sistema. Estas plantillas se generan siempre que nuevos datos lleguen al sistema.

- **Comparación de patrones y toma de decisión (*matcher*).** Los patrones generados a partir de las nuevas muestras que han entrado en el sistema se comparan con aquellos guardados en la base de datos. Esta comparación nos proporciona unos valores (*match scores*) que se utilizan a la hora de tomar una decisión en el sistema al confrontarlos con un umbral. Moviendo estos umbrales es como se consigue modificar el rendimiento del sistema.

La plantilla de un usuario puede ser extraída de una o varias muestras biométricas. Algunos sistemas guardan múltiples muestras de un mismo usuario para tomar en cuenta las variaciones que puede sufrir un rasgo biométrico a lo largo del tiempo. Los sistemas de reconocimiento facial, por ejemplo, pueden guardar múltiples plantillas de un mismo individuo con diferentes poses con respecto a la cámara. Así se obtienen más posibilidades de obtener un correcto funcionamiento del sistema, pues tiene más datos con los que trabajar, aunque aumentan las necesidades computacionales.

Dependiendo de la aplicación, la plantilla puede ser guardada en la base de datos del sistema biométrico o en un dispositivo portátil que posea el individuo, como una tarjeta inteligente.

3.2 Tipos de operación

Vamos a tener en consideración tres tipos de operación: registro, verificación e identificación; y cada uno de ellos será explicado seguidamente. Por otro lado, la Figura 3.2 muestra un esquema orientativo de dichos modos.

En primer lugar nos referiremos al modo de ***registro***, ya que es fundamental a la hora de tener un sistema biométrico operativo. Ninguno de los otros tipos de operación

puede darse si no se ha producido este previamente. El objetivo en este punto es añadir un nuevo usuario a la base de datos con lo que es fundamental para tener una base de datos fiable y obtener un funcionamiento del sistema adecuado.

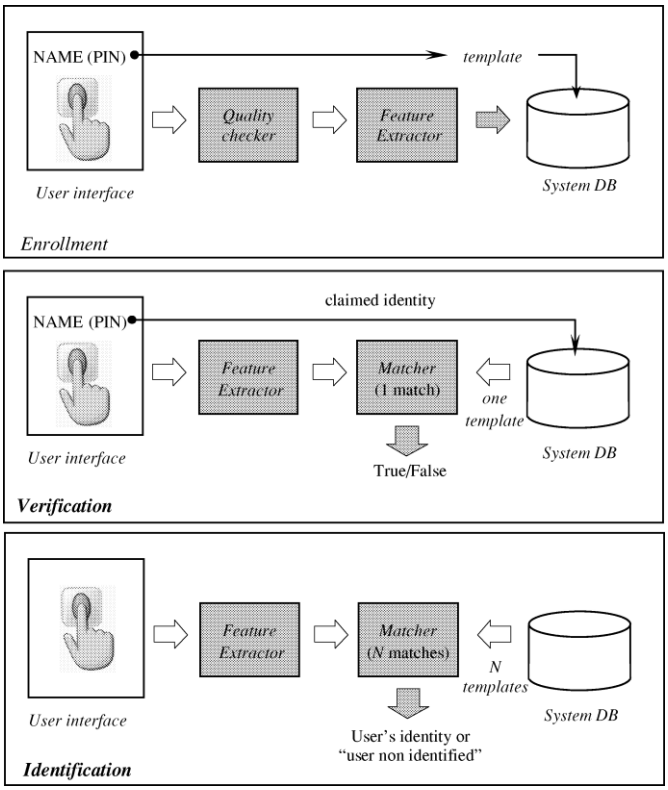


Figura 3.2 Diagrama de bloques de los modos de registro, verificación e identificación utilizando [3].

En el modo de ***verificación***, el sistema valida la identidad de una persona al comparar los datos biométricos capturados con la plantilla que tiene guardada en su base de datos. En este tipo de sistemas un individuo dice ser una determinada persona, normalmente haciendo uso de un *PIN*, un *nombre de usuario* o una *tarjeta inteligente*, y el sistema realiza una comparación entre los datos adquiridos y los guardados de dicha persona para comprobar que realmente el individuo es quien dice ser. La verificación suele utilizarse cuando se quiere evitar que varias personas usen la misma identidad.

Matemáticamente podemos formular el problema de la verificación del siguiente modo:

$$(I, X_Q) \in \begin{cases} \omega_1 \; if \; S(X_Q, X_I) \geq \eta \\ \omega_2 \qquad otherwise \end{cases},$$

donde se plantea la solicitud de una identidad I con un conjunto de características de entrada X_Q, y se debe determinar si (I, X_Q) pertenece a ω_1 o ω_2, donde ω_1 significa que la solicitud es verdadera y ω_2 que es falsa. X_I se refiere a la plantilla del usuario guardada en el sistema. $S(X_Q, X_I)$ es una función encargada de calcular la similitud

entre los datos de entrada y los guardados y el valor obtenido se confronta contra un umbral para determinar si hay coincidencia o no.

Por último, en el modo de ***identificación***, el sistema reconoce un individuo al buscar a través de todas las plantillas de todos los usuarios que tiene guardados en su base de datos. En este caso no tenemos un usuario queriendo identificarse como un individuo determinado, sino una identidad desconocida que queremos pase a ser conocida. La identificación es un componente crítico en las aplicaciones de *reconocimiento negativo*, donde el sistema establece si una persona es quien niega ser. El objetivo del reconocimiento negativo es evitar que una persona pueda utilizar múltiples identidades. La identificación también puede ser utilizada en el *reconocimiento positivo* para una mayor comodidad del usuario, ya que a este no se le requeriría identificarse sino que se haría de manera automática.

Matemáticamente el problema de la identificación puede ser referido del siguiente modo: Dada una entrada X_Q, determina la identidad I_K, $K \in \{1, 2, \dots, N, N+1\}$, donde $I_1, I_2, \dots, I_N$ son las identidades registradas en el sistema y I_{N+1} se refiere al caso en el que ninguna identidad de la base de datos se relaciona con la presentada [3].

$$X_Q \in \begin{cases} I_K & if\ K = \arg max_k \left\{ S\left(X_Q, X_{I_k}\right) \right\} \geq t, k = 1,2, \dots, N \\ I_{N+1} & otherwise \end{cases}$$

donde X_{I_k} es la plantilla biométrica correspondiente a la identidad I_K y t es un umbral predefinido.

3.3 Rendimiento y errores

¿Está nuestro sistema funcionando al máximo de sus posibilidades? ¿Estamos obteniendo los resultados adecuados o suficientes para nuestra aplicación? ¿Hay algo que podamos mejorar? Estas y otras preguntas surgen cuando hacemos uso de un sistema biométrico, ya que una vez establecidos los parámetros necesarios para su funcionamiento nos interesa saber si realmente funciona bien o no y en qué podríamos mejorarlo.

Los errores en los sistemas biométricos pueden deberse a diversos cambios en los seres humanos y diversos factores ambientales. Dos muestras de un mismo rasgo biométrico de una persona no son exactamente iguales debido a diferentes condiciones de toma de imágenes, cambios psicológicos o de comportamiento del individuo, condiciones ambientales, etc. Es por esto que la salida de un sistema biométrico es una comparación cuantificada por un número (*score*) que mide la similitud entre la entrada al sistema y la plantilla del usuario: $S\left(X_Q, X_I\right)$. Para tomar una decisión nos atenemos a un valor umbral que indica si los pares de muestras pertenecen a la misma persona o no.

En un sistema biométrico nos podemos encontrar con dos tipos de errores:

1) Identificar dos muestras de dos individuos diferentes como si perteneciesen al mismo: falsa aceptación (FM, *false match*).

2) Identificar dos muestras de una misma persona como pertenecientes a una persona diferente: falso rechazo (FR, *false rejection*).

La estos dos errores dan lugar a una falsa tasa de aceptación (FMR, *false match rate*) y a una falsa tasa de rechazo (FRR, *false rejection rate* o FNMR, *false non match rate*). Estos dos índices nos ayudan a evaluar el funcionamiento de nuestro sistema biométrico. Ambas tasas están relacionadas con el umbral puesto en el sistema: si el umbral aumenta conseguimos un sistema más seguro con menor FMR pero sin embargo aumenta la FRR; por otro lado una disminución del umbral conlleva la situación contraria, una disminución de la FRR y un sistema menos seguro por el aumento de la FMR. Debemos encontrar los umbrales adecuados para nuestro sistema teniendo en cuenta el tipo de aplicación que utilicemos y las necesidades de la misma. Es posible que no queramos un sistema excesivamente seguro, pero sin embargo no queramos importunar a los usuarios con las molestias causadas por un falso rechazo. Sin embargo, otras aplicaciones como las de control de accesos en áreas restringidas, donde la seguridad es un factor esencial, necesitarían una mayor seguridad donde la tasa de falsa aceptación sea lo menor posible.

Al evaluar un sistema real obtenemos gráficas similares a la Figura 3.3 obteniendo así la información necesaria sobre el rendimiento de nuestro sistema y una idea del umbral más adecuado a tener en cuenta.

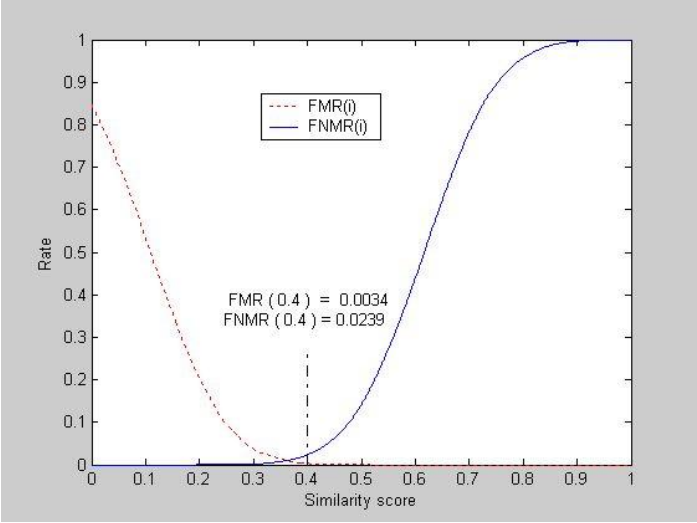


Figura 3.3 Ejemplo de tasas FMR y FNMR

Además de las tasas de error anteriores, también la tasa de fallos en captura (FTC, *failure to capture rate*) y la tasa de fallos en registro (FTE, *the failure to enroll*) son utilizadas para evaluar el rendimiento de un sistema. La tasa FTC sólo se aplica a sistemas biométricos que tengan una funcionalidad de captura automática implementada y muestra el porcentaje de las veces en que el sistema falla al capturar una muestra

cuando se presenta la característica biométrica. Por otro lado, la tasa FTE, denota muestra el porcentaje de las veces que los usuarios no son capaces de llevar a cabo un registro en el sistema de reconocimiento.

3.4 Ataques y limitaciones

Los sistemas biométricos son utilizados principalmente en aplicaciones relativas al ámbito de la seguridad o forense por lo que muchas personas podrían tener intereses en atacarlos y explotar al máximo sus vulnerabilidades. Las vulnerabilidades de un sistema pueden ser divididas principalmente en dos grupos principales:

3.4.1 Ataques directos

La posibilidad de crear muestras biométricas sintéticas se puede definir como la primera vulnerabilidad de un sistema biométrico. Estos ataques se realizan a nivel de sensor y no es necesario poseer un conocimiento específico sobre las operaciones que realiza el sistema. Es más, el ataque se lleva a cabo en el dominio analógico sin entrar en la parte digital del sistema, por lo que los mecanismos de protección digital no pueden ser usados ante este tipo de ataques.

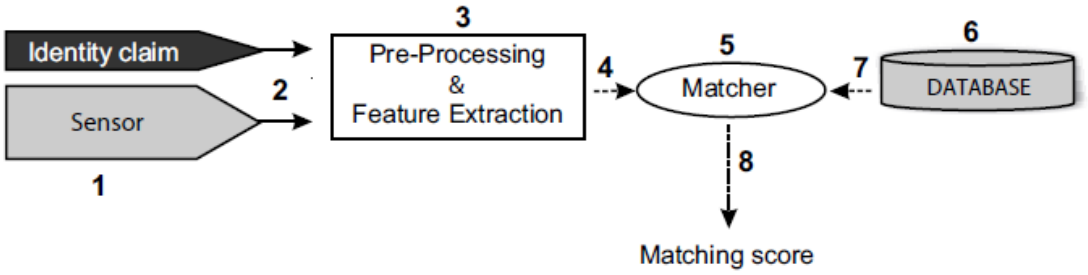


Figura 3.4 Arquitectura de un sistema biométrico de verificación automático. Posibles puntos de ataque numerados de 1 a 8 [8].

3.4.2 Ataques indirectos

Según [9] en este grupo se incluyen el resto de ataques, de 2 a 8: 2) volver a presentar una señal biométrica ya guardada, 3) sustituir el extractor de características por un troyano que produzca conjuntos de características predeterminados, 4) sustituir conjuntos legítimos de características con conjuntos sintéticos, 5) sustituir el comparador (*matcher*) por un troyano, 6) modificar o eliminar las plantillas guardadas en la base de datos, 7) alterar los datos en el canal de comunicaciones entre algunos módulos del sistema, 8) invalidar la salida del sistema.

A diferencia de los ataques directos, en este caso el intruso necesita tener alguna información sobre el funcionamiento interno del sistema biométrico y, en la mayoría de casos necesita tener acceso a algunos de los componentes de la aplicación.

3.4.3 Ataques sin esfuerzo

Los rasgos biométricos de un intruso pueden ser lo suficientemente similares a los de un individuo legítimamente registrado dando lugar a una falsa aceptación y a una vulnerabilidad del sistema. Esta cuestión nos lleva a pensar en la individualidad de los rasgos biométricos.

3.4.4 Limitaciones

Algunas de las limitaciones y desafíos más destacados en los sistemas biométricos son explicadas en lo sucesivo:

I. **Ruido en los datos de entrada.** Una imagen de una huella dactilar con una marca o señal, o una muestra de voz alterada por un resfriado son ejemplos de datos afectados por ruido. Estos datos pueden surgir también por un mal uso o mantenimiento de los sensores o malas condiciones ambientales (por ejemplo, mala iluminación en un sistema de reconocimiento facial).

II. **Variaciones intra-clase.** Un individuo puede interactuar de manera incorrecta con el sensor o estar sometido a algún cambio en sus características biométricas. Estos cambios pueden ser compensados guardando múltiples plantillas de un mismo usuario y actualizándolas a lo largo del tiempo.

III. **Unicidad.** Solapamiento entre características de varios usuarios. La similitud entre usuarios causa un aumento de la tasa de falsa aceptación.

IV. **No-universalidad.** Es posible que algún grupo de usuarios no posean un determinado rasgo biométrico, lo cual haría imposible su uso de manera fiable en un sistema.

V. **Problemas de interoperabilidad.** En la mayoría de los sistemas biométricos se asume que los datos que se adquieren como los guardados en la base de datos del sistema son tomados con los mismos sensores.

VI. **Ataques Spoof.** Un individuo puede intentar modificar deliberadamente alguno de sus rasgos biométricos para evitar el reconocimiento o para ser identificada como otra persona determinada.

3.5 Sistemas biométricos multimodales

Como hemos visto los sistemas biométricos presentan limitaciones y vulnerabilidades que pueden comprometer su funcionamiento. Algunas de las limitaciones que presentan los sistemas unimodales anteriores pueden solventarse mediante el uso de sistemas multimodales que integren varias fuentes de información.

Estos sistemas permiten capturar múltiples muestras de un rasgo biométrico o muestras de múltiples rasgos.

Según [10] estos sistemas se categorizan en 3 arquitecturas diferentes según la estrategia empleada para combinar los rasgos:

- **Fusión a nivel de extracción:** La información extraída de diferentes sensores se combina en un vector de características único que se compara con una plantilla, Figura 3.5.

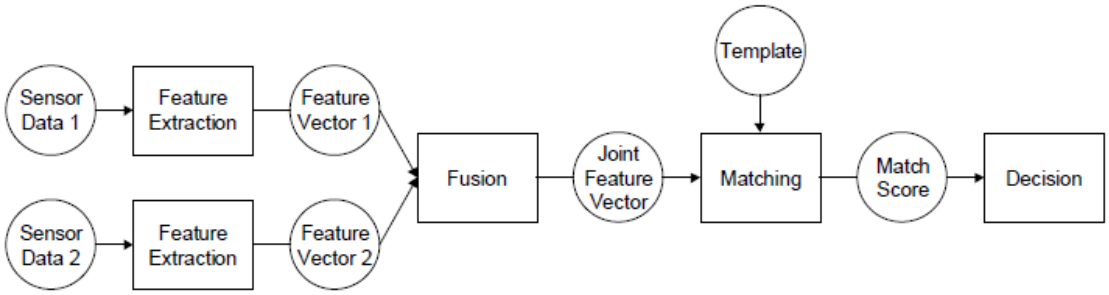


Figura 3.5 Fusión a nivel de extracción de características [11].

- **Fusión a nivel de comparación:** Los vectores de características se crean independientemente y se comparan con las plantillas guardadas en el sistema que son guardadas separadamente para cada rasgo biométrico, Figura 3.6.

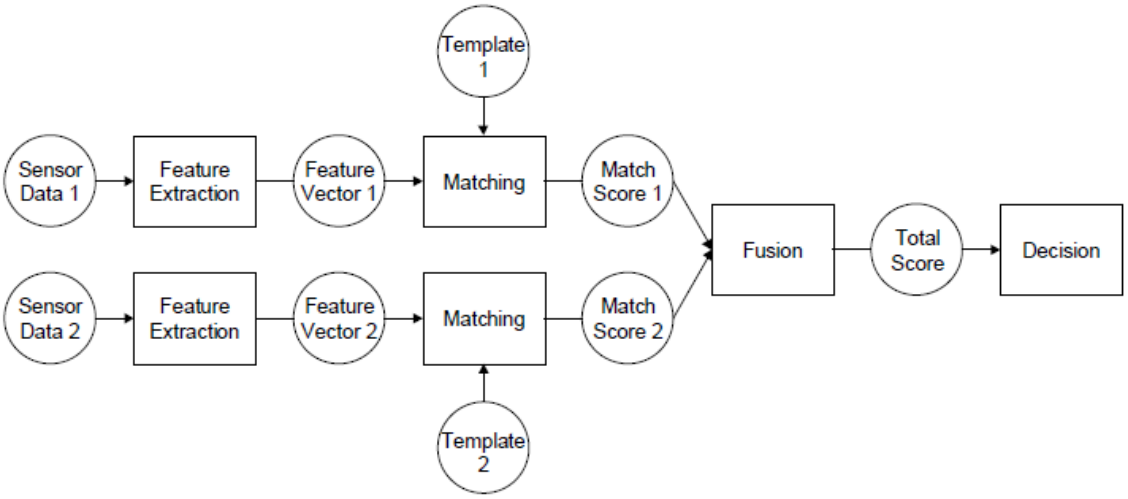


Figura 3.6 Fusión a nivel de comparación [11].

- **Fusión a nivel de decisión:** Se toma una decisión por cada rasgo biométrico y estas decisiones se combinan para obtener una decisión final, Figura 3.7.

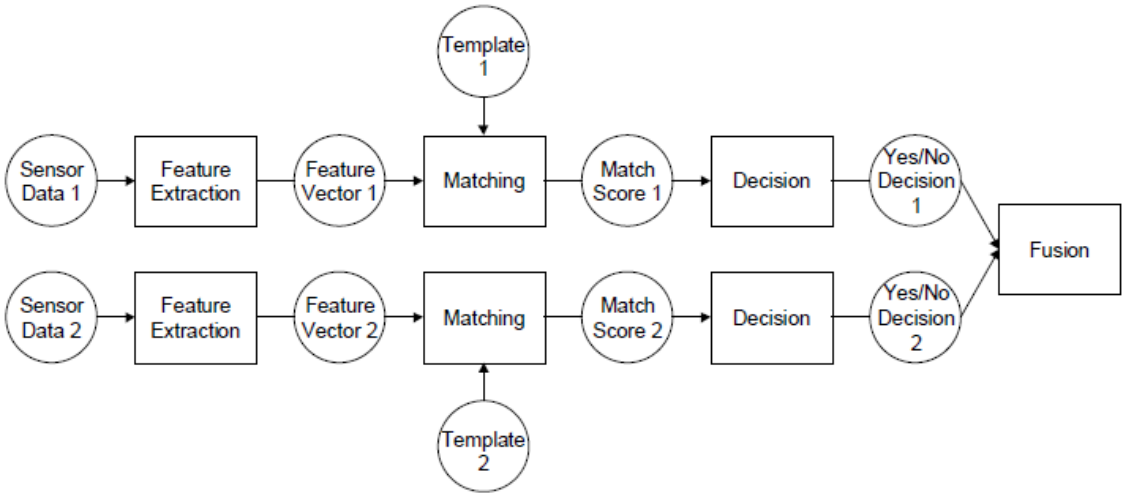


Figura 3.7 Fusión a nivel de decisión [11].

Un sistema multibiométrico puede operar en los siguientes modos:

- **Modo serie:** La salida producida por un rasgo biométrico es usada como entrada para el próximo rasgo biométrico. De este modo se reduce el número de posibles coincidencias dentro de un sistema de identificación. Por ejemplo, un sistema que use tanto reconocimiento facial como por huella dactilar podría emplear en primer lugar la información facial para extraer algunas coincidencias y, en segundo lugar la información de las huellas dactilares para encontrar la identidad que nos interesa.

- **Modo paralelo:** La información obtenida de la aplicación de múltiples rasgos biométricos se usa de modo simultáneo para producir la identificación requerida.

- **Modo jerárquico:** Las informaciones de los diferentes rasgos biométricos no tienen por qué ser adquiridas de manera simultánea y además una decisión podría tomarse sin necesidad de adquirir la información de todos los rasgos. Los clasificadores individuales se combinan en una estructura de árbol.

3.6 Aplicaciones

Establecer la identidad de una persona es una necesidad crítica en varias aplicaciones, por lo que la biometría se incorpora a cada vez más aplicaciones. Estas aplicaciones pueden ser divididas en tres grandes grupos principales:

- **Comerciales:** Comprenden situaciones tales como autenticación en red, la seguridad de datos electrónicos, el comercio electrónico, acceso a Internet, tarjetas de crédito, etc.

- **Aplicaciones gubernamentales:** DNI, pasaporte, carnet de conducir, seguridad social, control de fronteras, etc.

- **Aplicaciones forenses:** identificación de cadáveres, investigación criminal, determinación de relaciones de parentesco, etc.

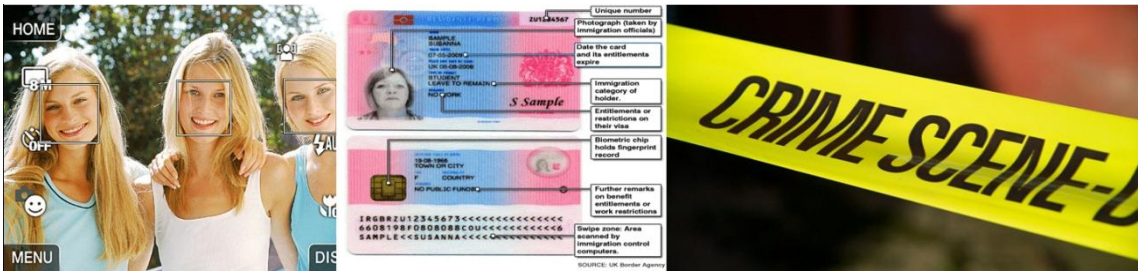


Figura 3.8 Aplicaciones biométricas: cámaras fotográficas (izq.), DNI electrónico (medio), forenses (dcha.).

Algunos ejemplos más determinados pueden ser los siguientes:

- El aeropuerto Schiphol de Ámsterdam está utilizando con éxito un sistema de seguridad biométrico basado en el escaneo del iris que evita a los pasajeros los controles de pasaporte, en Reino Unido, la policía del sur de Gales ha implantado un sistema de Visionics de reconocimiento facial para detectar hooligans y en Alemania existe un pasaporte digital.

- Algunas instituciones en Japón han instalado sistemas de autenticación utilizando las venas de las palmas de las manos en sus cajeros para ayudar a validar la identidad de un cliente.

- La cadena estadounidense de supermercados Kroger emplea escáneres de huella dactilar en alguna de sus tiendas para ayudar a los clientes a realizar sus pagos. Los interesados pueden registrar sus huellas dactilares junto a sus datos bancarios.

4

DETECCIÓN FACIAL. ESTADO DEL ARTE

4.1 Introducción

Las imágenes que contienen caras son esenciales en muchos sistemas inteligentes de visión y muchos esfuerzos en el procesado facial incluyen el reconocimiento facial, el seguimiento facial, la estimación de la pose y el reconocimiento de expresiones. Sin embargo, muchos métodos asumen que las caras de una imagen se encuentran identificadas y localizadas. Para construir sistemas completamente automáticos, que analicen la información contenida en imágenes de caras, se requieren sistemas de detección facial robustos y eficientes.

Dada una imagen el objetivo del sistema de detección es identificar todas las regiones que contienen una cara sin importar su pose, orientación o condiciones de luz. Este problema presenta una gran dificultad ya que las caras son objetos diferentes según el sujeto y variables en tamaño, forma, color y textura.

- **Pose.** El ángulo de la cámara al capturar las imágenes puede variar rotando una cara hasta 180º desde una posición frontal dando lugar a poses frontales, de perfil, rotadas 45º, tomadas desde arriba o desde abajo… La orientación de la imagen también puede producir oclusiones en algunas características faciales como la nariz o los ojos.

- **Presencia de componentes estructurales.** La presencia de características faciales como barba, bigote y gafas influyen en el rendimiento del sistema y presentan un grado de variabilidad bastante alto.

- **Expresión facial.** La apariencia de las caras está directamente afectada por la expresión facial de la persona. En algunas aplicaciones biométricas se rechazan imágenes que muestren alguna expresión.

- **Oclusión.** Parte o la totalidad de una cara puede verse comprometida al ser tapada por diversos objetos o por la pose del usuario. En una imagen de un grupo de personas algunas caras pueden estar parcialmente tapadas por otras.

- **Orientación.** La rotación de una imagen afecta la posible localización de caras.

- **Condiciones de captura.** Cuando una imagen es capturada, factores como la iluminación y las características de la cámara afectarán la respuesta del sistema en el posterior procesado.

4.2 Métodos de detección de caras

Solamente un proceso de reconocimiento basado en una cara compuesta de características puras puede ser denominado reconocimiento facial [12]. La localización de dichas características, el proceso de detección facial, puede ser llevada a cabo por muchos métodos que clasificaré en cuatro categorías, desde métodos basados en el desarrollo de reglas basadas en conocimientos previos de características, a métodos de búsqueda de características faciales invariantes.

I. **Métodos basados en conocimiento.** Las reglas en estos métodos están basadas en el conocimiento humano sobre las características que definen una cara. La mayoría de las reglas utilizan la relación entre características. Estos métodos están designados principalmente para la localización facial.

II. **Aproximaciones de características invariantes.** Estos algoritmos tienen por objetivo encontrar características estructurales que existen incluso cuando varían la pose, el punto de vista, las condiciones de luz, etc. Estos métodos están designados principalmente para la localización facial.

III. **Métodos de coincidencia de plantillas.** A partir de un conjunto de muestras se construye un patrón facial estándar. La relación entre la imagen de muestra y el patrón definido es observado y utilizado para hacer una deducción. Estos métodos están designados tanto para la localización como para la detección facial.

IV. **Métodos basados en apariencia.** Son similares a los anteriores, pero a diferencia de ellos las plantillas son obtenidas a partir de un conjunto de imágenes de entrenamiento que deberían capturar la variabilidad representativa de la apariencia de una cara. Estos métodos se usan principalmente para la detección facial.

Tabla 4.1. Métodos descritos en este proyecto para la Detección Facial en una Imagen.

Tipo de método	Trabajos descritos
Métodos basados en conocimiento.	a) Yang y Huang [13]. b) Tang, Kawato, Ohya y Nakatsu [14]. c) Peer y Solina [15].
Métodos basados en características invariantes.	a) Lienhart y Maydt [16]. b) Garcia y Tziritas [16]. c) Menser y Müller [17].
Métodos basados en coincidencia de plantillas.	a) Saber y Tekalp [17]. b) Yuille, Hallinan y Cohen [18]. c) Sinha [19].
Métodos basados en apariencia.	a) Pham, Worring y Smeulders [20]. b) Rowley, Kanade y Balaujua [21]. c) Roth, Yang y Ahuja [22]. d) Scheiderman y Kanade [23].

4.3 Métodos basados en el conocimiento

El desarrollo de un sistema de detección facial basado en estos métodos implica que una serie de reglas son definidas antes de la implementación del sistema. Estas reglas se derivan del conocimiento del investigador sobre las caras humanas. En general estos métodos carecen de todo entrenamiento. El sistema se resume a lo que ha sido definido por el investigador que actúa como un límite sobre la efectividad de la detección.

Es fácil descubrir reglas que describan las características de una cara. Por ejemplo, una cara suele aparecer en una imagen con dos ojos simétricos entre sí, una nariz, una boca… Las relaciones entre características pueden ser representadas por sus distancias relativas y posiciones.

Uno de los problemas de estos métodos es la dificultad que se presenta al intentar traducir el conocimiento humano a reglas bien definidas. Existen dos tipos de reglas: estrictas, que dan lugar a una baja tasa de detección; o laxas, que producen una alta tasa de falsas detecciones.

4.3.1 Yang y Huang

Yang y Huang utilizaron un método basado en conocimiento jerárquico para detectar caras [13]. Su sistema consiste en 3 niveles de reglas. En el nivel más alto todas las posibles caras son encontradas mediante un escaneado con una sub-ventana sobre la imagen y la aplicación de un conjunto de reglas en cada localización. En este primer nivel las reglas son descripciones generales de lo que es una cara y según nos movemos a otros niveles las reglas son más estrictas. Se crea una jerarquía de imágenes de multi-resolución mediante el cálculo de promedios y sub-muestreo como se ve en la Figura 4.1.

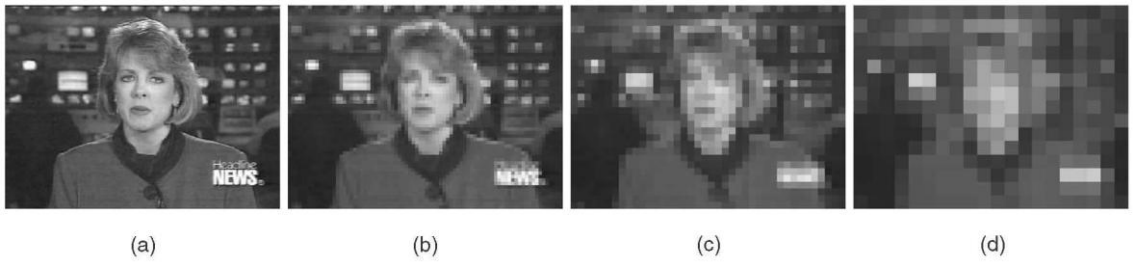


Figura 4.1 Imagen original (a), y sus correspondientes a menor resolución (b-d) [24].

Algunos ejemplos de las reglas utilizadas para localizar las caras en las imágenes con menor resolución son: la parte central de la cara tiene cuatro celdas con una intensidad básicamente uniforme (ver Figura 4.1 y Figura 4.2); la parte más externa alrededor de la anterior (más claramente coloreada en la Figura 4.2) tiene una intensidad básicamente uniforme y la diferencia entre la media de la zona gris más oscura y la más clara es significante.

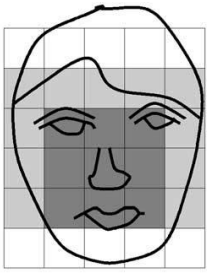


Figura 4.2 Cara típica utilizada en métodos basados en conocimiento.

4.3.2 Tang, Kawato, Ohya y Nakatsu

La técnica presentada por estos investigadores [14] basa la detección en dos factores clave: distribuciones Gaussianas del color de la piel y el pelo, y diferenciación entre datos de piel irrelevantes utilizando una "curva de decisión" que localice la línea del pelo en una cara humana. En general, el método de detección puede dividirse en dos estados separados, cada uno de los cuales genera imágenes intermedias y que producirá una curva de decisión en el estado final.

I. Generación de una distribución del color de la piel produciendo una imagen binaria.

II. Generación de una distribución del color del pelo produciendo otra imagen binaria.

III. Cálculo de una curva de decisión ofrecida por las imágenes binarias previas.

El primer estado se basa en un modelo de color de piel que se estima a partir de una distribución bidimensional Gaussiana cuyo propósito es la diferenciación respecto a un "fondo complejo". El método se basa en la regla de que el color de piel está limitado a un rango de posibles colores RGB. En el primer estado dentro de la generación de la distribución de colores, se produce una imagen que contiene solamente los pixeles de la piel pintados. El cálculo de la curva de detección se produce en el siguiente estado del método, y se basa en un modelo de pelo, basado de nuevo en una distribución Gaussiana en 2D. La aplicación del modelo de pelo a la imagen originar también va a proveer una imagen secundaria que contenga pintados solamente los pixeles de la región del pelo.

Antes del cálculo de una curva de decisión las imágenes de muestra de los modelos de piel se observan y se realiza un pre-procesado para eliminar los elementos de la piel que no son representativos en una cara. Esto se hace mediante un escaneo de la imagen tanto en horizontal como en vertical y un reajuste de carreras de pixeles coloreados a cero si están por debajo de un umbral predeterminado. Esto producirá otra imagen y se estimará una curva de decisión a partir de ella.

Esta técnica es un buen ejemplo de un sistema basado en conocimiento, cuyo método se basa tanto en las distribuciones de color del pelo como de la piel y la presencia de una línea de pelo. Aunque esta técnica es bastante estricta, su aplicación ha demostrado que es bastante efectiva en detección facial de imágenes frontales, y también si la orientación facial ha sido alterada. La restricción obvia de este método es que el individuo debe tener pelo.

4.3.3 Peer y Solina

En la publicación presentada por Peer y Solina [15] se plantea una segmentación de las áreas de piel de una imagen a color y se aplican filtros y algoritmos de detección de bordes para localizar la presencia de ojos humanos en las muestras de piel. El proceso de detección es simple en esta técnica y ha sido diseñado para maximizar la velocidad del sistema. La aplicación que realiza conjuntos de reglas geométricas para la detección de ojos es un factor limitador, pero se compensa con el hecho de usar algunas reglas heurísticas.

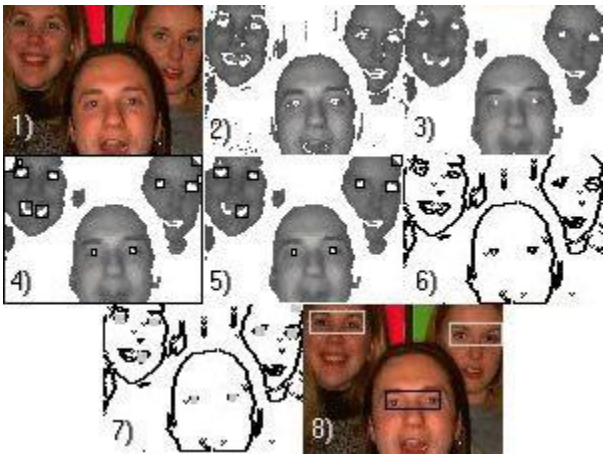


Figura 4.3 Diferentes estados del método de Peer y Solina [15].

El método tiene varios estados (ver Figura 4.3). En los tres primeros la imagen pasa por un proceso de filtrado donde los segmentos de piel son identificados y se convierte al color blanco. Posteriormente la imagen se convierte a escala de grises y se aplica un filtro, utilizando la imagen transformada, que borra las regiones blancas que sean significativamente pequeñas o las pequeñas variaciones en el color de la piel de las caras de muestra. Por lo tanto, al finalizar este estado se ha completado la detección de piel basada en la distribución del color.

Figura 4.4 Decisiones de la localización de los ojos [15].

En los tres estados posteriores se identifican y segmentan las regiones blancas a través del uso de un algoritmo, se eliminan las regiones que sean demasiado pequeñas para representar una cara y se trazan los bordes donde haya un color gris significante. A partir de ahora es cuando comienza la deducción acerca de la existencia de caras en la imagen. Mediante el uso de transformadas de Hough se seleccionan las áreas más probables como posibles contenedoras de ojos (Figura 4.4) y en cada área se busca la situación de los ojos. Utilizando características faciales geométricas el algoritmo encuentra círculos parejos que representen un par de ojos. La deducción se basa en la información de los ojos y el color de piel prediciendo la probabilidad de que haya una cara. Esto es mejorado con un algoritmo basado en reglas heurísticas que ayuda a eliminar los falsos positivos.

4.4 Métodos basados en características invariantes

Localizar en una imagen características útiles para la clasificación es una tarea compleja. A diferencia de los métodos basados en conocimiento, en este caso se buscan características invariantes para la detección facial. Se ha observado que los seres humanos pueden detectar caras y objetos sin esfuerzo en diferentes poses y condiciones de luz y por eso deben existir propiedades o características que sean invariantes sobre los conjuntos que se nos presentan. A partir de esta presunción numerosos métodos han sido propuestos para detectar características faciales y deducir si se da o no la presencia de una cara. Las características faciales (Figura 4.5) como cejas, ojos, nariz, boca, línea de pelo son extraídas comúnmente utilizando detectores de borde. Basándose en las características extraídas se construye un modelo estadístico que describa sus relaciones y verifique la existencia de una cara. Una limitación con estos algoritmos surge de la modificación de las características debido a la luminosidad, ruido, oclusiones, etc.

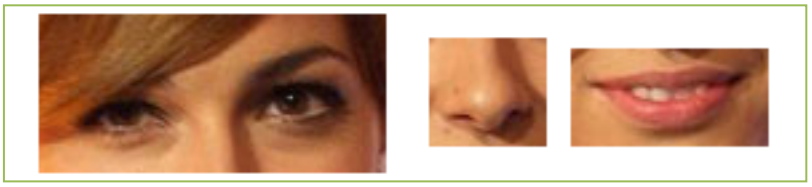


Figura 4.5 Algunas características faciales.

4.4.1 Lienhart y Maydt

Lienhart y Maydt [16] propusieron una mejora al algoritmo de Viola-Jones, introduciendo una ampliación del conjunto de características de Haar utilizadas. La técnica propuesta da lugar a resultados de detección más efectivos y un error de clasificación menor a través del uso de un conjunto de características modificado, permutaciones del conjunto de Haar y la adopción del algoritmo "Gentle Adaboost" para la selección de características.

Lienhart y Maydt mencionaron que los sistemas basados en características, al contrario que los basados en pixeles, reducen la variabilidad intra-clase. También establecieron que al combinar las características con un método de selección como Gentle Adaboost la capacidad de selección puede aumentar. Las características utilizadas por Viola-Jones se modificaron rotando cada una de ellas 45º tanto en sentido positivo como negativo (ver Figura 4.6) y algunas fueron elegidas para la detección. Se vio que se produjo un descenso de aproximadamente el 10% en las tasas de falsa alarma frente a lo presentado por Viola-Jones.

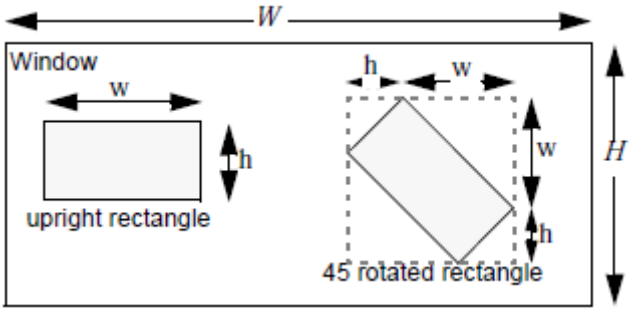


Figura 4.6 Ejemplo de un rectángulo rotado 45º [16].

La técnica también evaluó una gran cantidad de algoritmos de *boosting* e implemento el Gentle Adaboost, que en comparación a los otros se concentra menos en los ejemplos difíciles de definir. Esto es bastante diferente al proceso de Adaboost presentado por Freund y Schapire [25].

4.4.2 Garcia y Tziritas

El color y la textura también son considerados como un conjunto importante en una cara. Garcia y Tziritas presentaron una técnica que utiliza regiones de color de piel cuantificadas y un análisis de paquetes de *wavelets* [26]. Su método tiene dos etapas principales: la detección del color de piel y el análisis efectivo de la textura utilizando una aplicación de análisis mediante filtros de paquetes *wavelets*.

Figura 4.7 Ejemplo de imagen de 24 bits (dcha.) e imagen reducida a 8 colores.

En el primer estado se realiza una cuantificación del color de la imagen y posteriormente se detectan las regiones con color de piel. La cuantificación ayuda a mejorar la segmentación de los colores de piel homogeneizando las regiones de la imagen. Esta cuantificación se consigue a través de la cuantificación de vector y reduce los colores que aparecen en la imagen a un conjunto menor (ver Figura 4.7). Acto seguido se produce la segmentación del color de la piel en la imagen obtenida. El método utiliza dos modelos para la representación del color: YCbCr y HSV.

La detección de posibles regiones faciales se consigue mediante la unión de regiones homogéneas de color de piel. El algoritmo de unión calcula un grafico de adyacencia entre regiones, donde cada nodo es una región de color de piel de la imagen cuantificada, y conecta o no las regiones basándose en la similitud de los colores.

En la última etapa de esta técnica se produce la clasificación haciendo uso del análisis de paquetes de *wavelets* de aquellas regiones que sean posibles caras. Eligieron el uso de *wavelets* debido a su flexibilidad de aplicación y velocidad de cálculo. Se produce una descomposición por paquetes de *wavelets* en el plano de la intensidad de la imagen y se produce un vector de características de coeficientes de *wavelets*. La deducción se realiza aplicando un cálculo de distancias sobre los vectores de características extraídos de las áreas de color de piel segmentadas. Estos vectores caracterizan la textura facial.

4.4.3 Menser y Müller

Otra aplicación basada en la distribución del color es la de Menser y Brunig [27]. Se utiliza el análisis de color y la aplicación de un mapa de distancias así como el análisis de componentes principales (PCA, *principle component analysis*) para reducir la gran dimensión del espacio de la imagen en una muestra. La deducción se lleva a cabo a través de una comparación entre la energía de una sub-ventana con un umbral predeterminado.

La técnica construye una tabla donde se representa la probabilidad de cada pixel por separado de pertenecer a una región de piel. Esto se llama una "imagen de probabilidad de piel" (ver Figura 4.8). El hecho de que la imagen original y la imagen de probabilidad de piel tienen distribuciones de niveles de gris similares en las regiones faciales, se utiliza para detectar dichas zonas.

Figura 4.8 Imagen en escala de grises (izq.) e imagen de probabilidad de piel (dcha.) [27].

El análisis de componentes principales es el proceso mediante el cual se generan eigenvectores. Menser y Brunig utilizan la transformada de Karhunen-Loève para esto. Estos valores se utilizan entonces para construir una matriz de proyección. Se calcula acto seguido una media sobre el conjunto de imágenes de test obteniendo así una imagen que representa la media de las caras. La deducción se produce a través del cálculo de la distancia entre esta imagen de media de las caras y la matriz de proyección calculada previamente.

Este método de detección facial combina el análisis del color de la piel con una aproximación al cálculo de eigenvectores. La incorporación de la información del color

reduce la influencia del fondo de las imágenes y mejora la tarea de la detección especialmente en imágenes donde el tamaño de las caras es relativamente pequeño [27].

4.5 Métodos de coincidencia de plantillas

En los métodos de coincidencia de plantillas un patrón facial estándar (normalmente de una cara en pose frontal) se predefine manualmente o con parámetros mediante una función. Dada una imagen de entrada, se calculan unos valores de correlación utilizando los patrones estándar para el contorno facial, los ojos, la nariz y la boca independientemente. La existencia de una cara se determina en base a los valores de correlación. Estos métodos tienen como ventaja una implementación sencilla, sin embargo se ha probado que son ineficaces para la tarea de la detección facial ya que no pueden tratar efectivamente la variación en escala, pose y forma de la cara.

La aplicación de una plantilla es similar a los métodos basados en conocimiento, ya que el conocimiento de las características faciales o bien puede ser aprendido y construir así una plantilla dinámica, o bien puede ser predefinido. La aplicación de una plantilla podría realizarse a diferentes escalas debido a la simpleza de muchas plantillas que se relacionan simplemente con la simetría de las características faciales y con las distancias relativas. Sin embargo, como se ha dicho, utilizar solamente estos métodos no es muy efectivo con lo que a menudo suelen utilizarse junto a otras técnicas basadas en características invariantes. Los métodos descritos a continuación se centran sólo en aquellas técnicas donde la coincidencia de plantillas es la base para la detección.

4.5.1 Saber y Tekalp

Sabert y Tekalp [17] utilizaron una función de coste basada en la simetría y un módulo de clasificación de formas basado en la deducción sobre un conjunto de muestra. Mediante el uso de un proceso de segmentación de piel esta técnica es capaz de trabajar basándose en una plantilla predefinida, que es usada junto a un modelo elíptico y un algoritmo de búsqueda de características faciales.

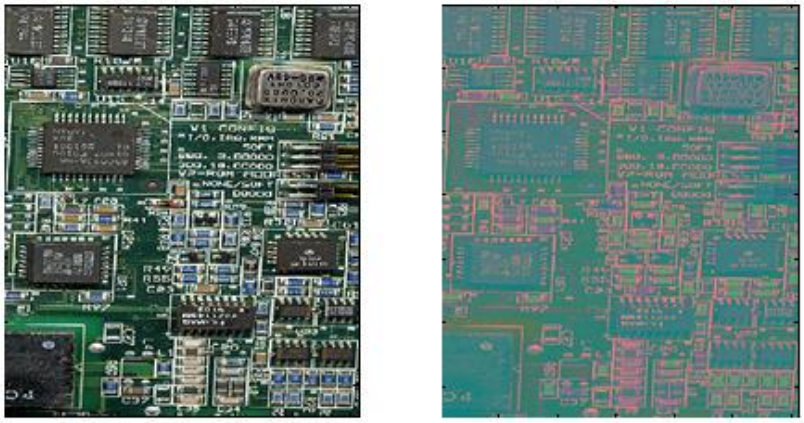


Figura 4.9 Figura board.tif obtenida en Matlab en formato RGB (izq.) y su correspondiente en el espacio YCbCr (dcha.)

En la primera etapa se realiza una segmentación del color de la imagen. Esto no es muy diferente a las técnicas que usan características invariantes basadas en el color presentadas anteriormente [26] [27]. Sin embargo, hay una diferencia y es la aplicación del espacio YCbCr en vez del RGB, que es menos sensible a las variaciones de la intensidad (ver Figura 4.9). Se realiza a continuación una clasificación del color de la piel, que se deriva de una distribución obtenida a partir de un conjunto de pruebas. Para agrupar las zonas de piel se utiliza un filtrado GRF (Gibbs Random Field). El uso del color como clasificador es muy eficiente computacionalmente y se usa como un proceso preliminar a la coincidencia de plantillas.

Una vez identificadas las regiones de piel debemos decidir si contienen una cara o no. Esto se realiza principalmente mediante el uso de un modelo de clasificación de forma. El modelo es una plantilla de cara basada en la naturaleza elíptica de la cara humana. El centro de la elipse se determina mediante el cálculo de eigenvectores que se derivan de las coordenadas espaciales de los pixeles de piel identificados en una región continua y se calcula su tamaño basándose en la distancia mínima de Hausdorff calculada a partir de los píxeles de la zona de piel agrupados alrededor del centro de la elipse. La plantilla se representa como una elipse y una función de coste de simetrías para las regiones de los ojos, la nariz y la boca (ver Figura 4.10).

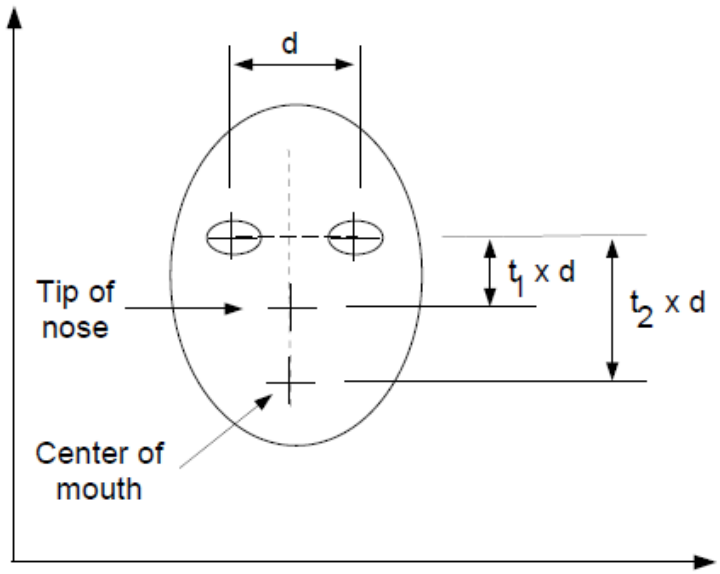


Figura 4.10 Localización de ojos y la boca y previsión para la situación de la nariz [17].

La función se basa en:

I. Los ojos están colocados en una línea paralela con el eje menor de la elipse.

II. Los ojos son simétricos con respecto al eje mayor de la elipse, representado mediante la dirección del eigenvector asociado con el mayor eigenvalor.

III. Los ojos son equidistantes respecto al eje menor representado por la dirección del eiegenvector asociado al menor eigenvalor.

IV. Los ojos son generalmente los huecos más próximos en la segmentación de la piel.

V. Los ojos están colocados sobre el eje menor de la elipse.

Utilizando la función de coste se realiza la decisión acerca de la existencia de una cara en las zonas de piel identificadas en la primera etapa de la técnica. La aplicación de la segmentación del color es el paso clave sin el cual el modelo elíptico de la cara no podría formarse ni se podrían observar las características localizadas. Por lo tanto resulta clara la importancia de que exista algún tipo de mecanismo que identifique dónde debería colocarse la plantilla. En este caso se hace mediante la segmentación del color de piel.

4.5.2 Yuille, Hallinan, Cohen

En este método se propone detectar y describir características faciales utilizando plantillas deformables que modelen esas características y que encajen en un modelo elástico *a priori* (por ejemplo los ojos) [18]. En esta técnica las características faciales se describen mediante plantillas, que han sido especificadas utilizando parámetros que posibilitan el conocimiento previo sobre la forma esperada de las características para guiar el proceso de detección. Estas plantillas son lo suficientemente flexibles como para poder cambiar su tamaño, y otros parámetros, para ajustarse a los datos. Se define una función de energía para unir bordes, picos y valles en la imagen de entrada con los correspondientes parámetros en la plantilla. El mejor ajuste del modelo elástico a la imagen se realiza minimizando la función de energía de los parámetros.

Una vez hecho esto los parámetros de la plantilla se actualizan mediante descenso por gradiente. Cambiar estos parámetros se corresponde con una alteración en la posición, orientación, tamaño y otras propiedades de la plantilla.

Aunque sus resultados experimentales hayan demostrado un buen rendimiento en el rastreo de características no rígidas, una desventaja de este método es que la plantilla debe ser inicializada en las cercanías del objeto de interés.

4.5.3 Sinha

Esta técnica [19] utiliza un pequeño conjunto de características espaciales invariantes en una imagen para describir el espacio de los patrones faciales. Para designar lo invariante se basa en que, mientras que las variaciones en iluminación cambian el brillo individual de diferentes partes faciales (ojos, mejillas, frente) el brillo relativo de estas partes permanece aproximadamente invariable. Determinar estos ratios en unas cuantas regiones y guardar solamente su dirección (por ej. ¿Es una región más brillante que otra?) proporciona una característica invariante bastante robusta.

Por lo tanto, las regularidades en el brillo facial se codifican como una plantilla de ratios que es una plantilla espacial de una cara con subregiones que se pueden

identificar con características faciales clave como ojos, mejillas y frente. Las relaciones en el brillo entre distintas zonas faciales se capturan mediante relaciones entre subregiones. Se dice que se ha localizado una cara si una imagen satisface todas las relaciones establecidas anteriormente.

La idea de utilizar diferencias de intensidad entre regiones adyacentes ha sido extendida posteriormente a representaciones basadas en wavelets para peatones, coches, y detección facial.

4.6 Métodos basados en apariencia

Anteriormente los métodos de coincidencia de plantillas se basaban en una definición de las plantillas por parte del investigador. Sin embargo, en los métodos basados en apariencia aprenderemos de ejemplos en imágenes. En general estos métodos se basan en técnicas de análisis estadístico y aprendizaje automático para encontrar características relevantes de imágenes faciales o de imágenes que no contengan caras. Las características aprendidas se expresan como modelos de distribución o funciones discriminantes que son utilizadas para la detección facial. Muchas son las técnicas utilizadas: *eigenfaces*, métodos basados en distribuciones, redes neuronales, máquinas de vectores de soporte, clasificadores bayesianos, modelos de Markov, etc. Algunos ejemplos pasan a explicarse a continuación.

4.6.1 Pham, Worring y Smeulders

En este método se presenta una red Bayesiana con una estructura en forma de bosque para resolver un problema de clasificación de una sola clase [20]. Se eligen los clasificadores Bayesianos debido a la gran velocidad que estos pueden alcanzar. La técnica también hace uso de *bagging* para generar un clasificador agregado ya que proporciona una manera natural de resolver los problemas de clasificación referidos a una sola clase.

El método de detección facial aquí referido se construye sobre una base de resolución de 20 pixeles que se escala con un factor de 1.2 hasta que el tamaño de la imagen sea menor que el de la ventana escalada. La resolución de 20 pixeles fue elegida al considerarse por parte de los autores más que adecuada para contener las características de una cara humana y minimizar la relación entre la resolución de y el tiempo de clasificación. En el conjunto de datos la técnica presenta variaciones en iluminación, expresión, orientación y presencia o falta de componentes faciales como barba o gafas. La imagen también se somete a un pre-procesamiento lo cual implica una normalización del gradiente de iluminación y una ecualización de histograma. Esto reduce el efecto de las sombras y suaviza el contraste de la imagen.

Un clasificador agregado sirve como núcleo de la decisión del sistema. Está compuesto por tres clasificadores de redes Bayesianas elegidos a través de un proceso

de *bagging*. La selección se basa en las soluciones observadas y los compromisos que interese que se tomen. Por ejemplo, cuando el número de clasificadores aumenta la tasa de detección disminuye así como también lo hace la tasa de verdaderos positivos. Después de que la imagen fuese examinada y sus regiones etiquetadas como candidatas a ser caras o no, hay una etapa de post-procesamiento en la cual se seleccionan las regiones con los mayores valores de probabilidad y se marcan como caras.

4.6.2 Rowley, Kanade y Balaujua

En esta técnica se aplica un método basado en redes neuronales para proporcionar medios efectivos para la decisión [21]. Se presenta como una red neuronal que se entrena para reconocer caras frontales. También combina algunas de estas redes para mejorar el rendimiento total del sistema. Los datos usados para entrenar cada red varían, específicamente con la frecuencia de imágenes de no caras.

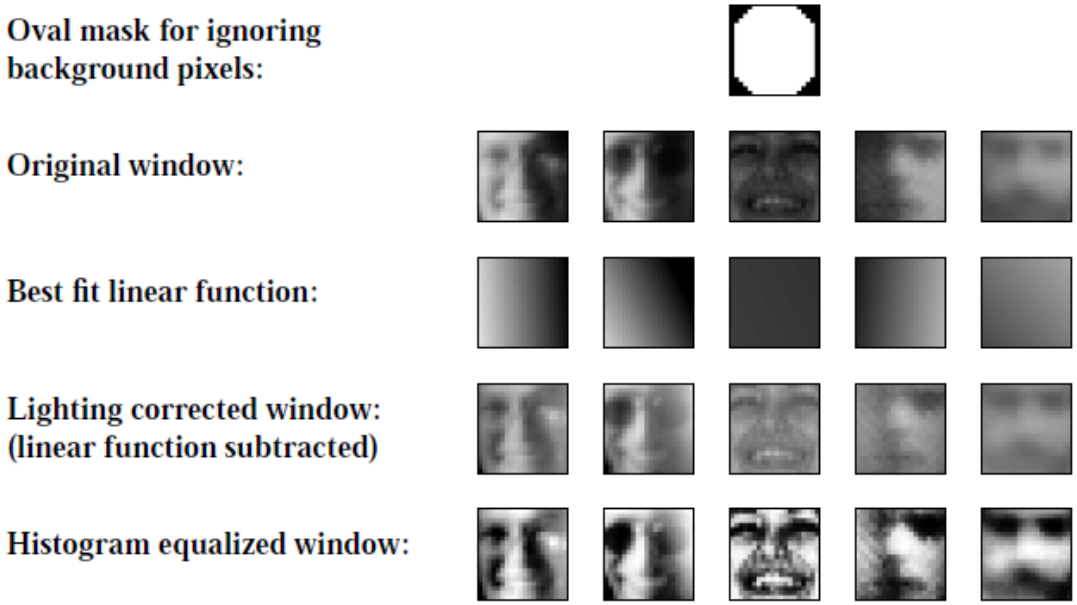


Figura 4.11 Pasos en el pre-procesado de una ventana [21].

Cada red se entrena con imágenes con una resolución de 20 por 20 píxeles, donde cada ventana es pre-procesada para ecualizar la intensidad. La intensidad se normaliza con una función que también tiene en cuenta un modelo elíptico simple, el cual oculta los bordes de la ventana de entrenamiento en un intento de ocultar píxeles que potencialmente puedan pertenecer al fondo de la imagen (ver Figura 4.11). Las redes neuronales tienen conexiones de retina del mismo modo que sus capas de entrada individuales, y un número real de valor único como salida que define la presencia de una cara. El algoritmo básico para la detección facial se presenta en la Figura 4.12.

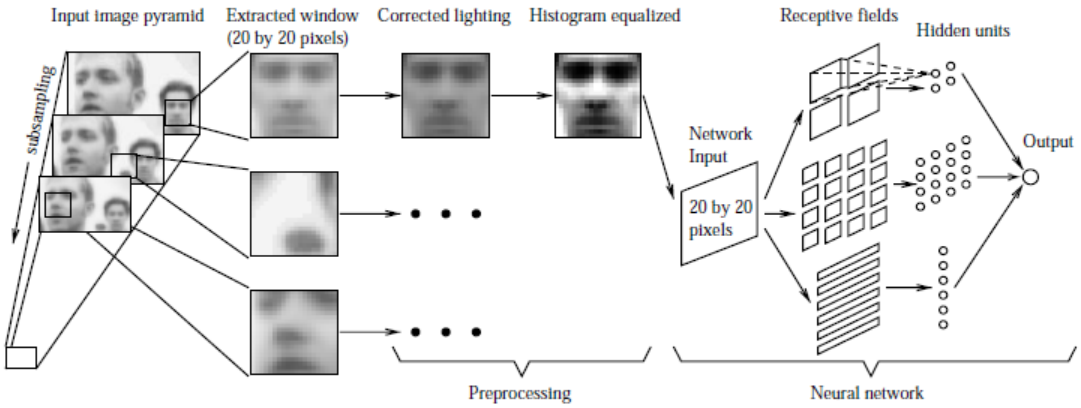


Figura 4.12 Algoritmo básico usado para la detección facial [21].

Uno de los beneficios más importantes de este método es su habilidad para utilizar múltiples redes entrenadas de manera diferente y por lo tanto proporcionar una decisión más exacta a la hora de localizar caras. Esto sólo es posible gracias a la naturaleza del método utilizado, simplemente entrenando redes individuales con pequeñas diferencias en los datos de entrenamiento obtenemos resultados completamente diferentes.

4.6.3 Roth, Yang y Ahuja

Roth, Yang y Ahuja [22] proponen una técnica diferente denominada SNOW (*Sparse Network of Winnows*) la cual utiliza una red formada por unidades lineares para definir el espacio de las características aprendidas. Al utilizar su método se da una relevancia particular a las tareas de clasificación que tengan grandes conjuntos de características. Dicho método muestra buenos resultados en la fase de aprendizaje en dichas circunstancias. Mediante el uso de características booleanas primitivas la técnica codifica tanto la intensidad, como los datos de la posición de los píxeles en la imagen de muestra.

Los nodos en la capa de entrada de la red representan relaciones simples entre las entradas y se utilizan como características de entrada. Cada unidad lineal se llama *nodo objetivo* y representa las relaciones de interés entre los ejemplos de entrada. La decisión en este sistema se hace mediante la unión de los *nodos objetivo*, en este caso sólo dos: cara y no cara. Dado un conjunto de relaciones, es decir tipos de características, que puedan existir en una imagen, cada imagen se asocia a uno de esos conjuntos de características presentes y SNOW propagará estas relaciones a un *nodo objetivo* y se tomará una decisión.

La aplicación de SNOW da lugar a detectar caras en múltiples poses, expresiones y diferentes niveles de iluminación. La aplicación del algoritmo *Winnow* en la red proporciona una manera de aprendizaje que se ajusta a dominios donde el espacio de características es grande y desconocido. El uso de la técnica se ha probado como una de las mejores implementaciones. Esto puede deberse a la aplicación del algoritmo

SNOW y la red de decisión que se produce. La decisión es bastante robusta debido al uso del algoritmo *Winnow*.

4.6.4 Scheiderman y Kanade

Una técnica diferente se presenta en el trabajo de Scheiderman y Kanade [23]. En este método se utiliza una vista en 2-dimensiones para representar la geometría de un objeto (cara, coche) en 3-dimensiones. A la vista en 2D se le aplican varias reglas que definirán si un objeto está presente en la vista con respecto a su orientación y pose frente a la cámara. Mediante la aplicación de una transformada de wavelets (ver Figura 4.14), los datos se evalúan estadísticamente y se producen histogramas para representar modelos de variaciones.

Figura 4.13 Ejemplos de imágenes de entrenamiento para cada orientación facial [23].

Para poder hacer frente a las variaciones en las imágenes se sigue una estrategia que consta de dos partes. Para las variaciones en pose, se utilizan múltiples detectores estando cada uno de ellos concentrado en una orientación específica (ver Figura 4.13). Para el resto de variaciones, se utiliza un modelo estadístico en cada uno de los detectores. La dificultad de configurar estos modelos estadísticos reside en que desconocemos sus características, y para representarlos han optado por utilizar un producto de histogramas que describa la variación en apariencia de los modelos faciales del resto del mundo físico. Cada histograma se refiere a un atributo visual diferente.

<table>
<tr><td>L1
LL</td><td>L1
HL</td><td rowspan="2">Level 2
HL</td><td rowspan="2">Level 3
HL</td></tr>
<tr><td>L1
LH</td><td>L1
HH</td></tr>
<tr><td colspan="2">Level 2
LH</td><td>Level 2
HH</td><td>Level 3
HH</td></tr>
<tr><td colspan="3">Level 3
LH</td><td>Level 3
HH</td></tr>
</table>

Figura 4.14 Representación de una imagen mediante wavelets [23].

En el proceso de decisión, un gran conjunto de atributos se utilizan para minimizar el error y para utilizar la mayor cantidad de información posible. Esta técnica trata de modelar conjuntamente la información visual localizada en el espacio, la frecuencia y la orientación. Para ello descompone la apariencia visual a lo largo de estas tres dimensiones. Con el fin de crear los atributos visuales mencionados anteriormente, hace falta seleccionar la información que está localizada a lo largo de estas dimensiones, aplicando una transformada de *wavelets* a la imagen.

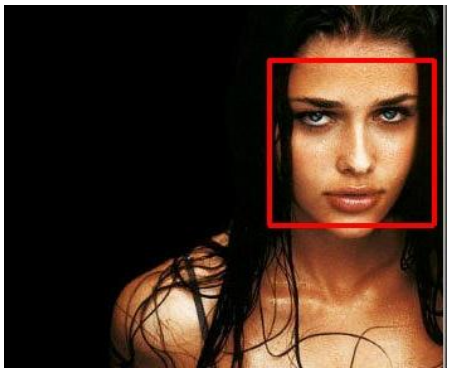
La decisión se basa en un escaneado de la imagen de entrada que lleva a la evaluación de atributos de bajo nivel, concentrándose en regiones donde los atributos parezcan prometedores a la hora de encontrar una cara. La clasificación final es un producto de los histogramas que se producen en la etapa de entrenamiento. En este método la principal meta es tratar de utilizar la mayor cantidad de información posible para la decisión.

5

ESTUDIO DEL DISEÑO E IMPLEMENTACIÓN DEL SISTEMA

5.1 Introducción

A lo largo de este capítulo se describe la estructura y el funcionamiento del detector facial de Viola y Jones [1] [28]. Esta estructura es la que se ha implementado dentro de este proyecto.



Uno de los aportes más interesantes planteados por Viola y Jones es un método que se distingue de los anteriores por su velocidad. Según los autores, operando sobre imágenes de 384 por 288 píxeles se detectarían caras a una velocidad de 15 frames por segundo utilizando un ordenador Pentium III de 700 MHz. Además de esto podemos diferenciar tres grandes aportes de esta técnica: *i)* la imagen integral, que nos posibilita realizar una rápida evaluación de las características; *ii)* un clasificador eficiente que utiliza el método de Adaboost para seleccionar grupos de características, que serán las que se utilicen para llevar a cabo la detección; *iii)* y un método de combinación de los clasificadores de manera eficiente tanto para la detección como para el tiempo que tome la misma.

En la sección 5.2 de este capítulo se describen las características utilizadas para la detección y la manera de extraerlas utilizando la imagen integral. En la sección 5.3 se muestra la manera de clasificar las imágenes en caras o no caras utilizando el algoritmo Adaboost. La sección 5.4 explica el método propuesto por Viola-Jones para realizar el detector final, haciendo uso de clasificadores en cascada, y cómo se extienden estas clasificaciones a imágenes de diversas escalas y localizaciones. Por último, en la

sección 5.5 se muestran los resultados planteados por Viola y Jones [1] para tener una idea del funcionamiento y rendimiento general del detector.

5.2 Características e imagen integral

El sistema de detección de Viola-Jones utiliza grupos de características simples para llevar a cabo la detección. El uso de estas características posibilita una velocidad mucho mayor a la hora de detectar caras frente a un sistema basado en píxeles y beneficia la codificación en el dominio *ad-hoc* [28]. Las características utilizadas son parecidas a aquellas del tipo Haar relacionadas con las propuestas por Papageorgiou et al. [29] en 1998.

Viola y Jones utilizan 4 tipos de características en su sistema de detección: dos tipos basados en dos rectángulos, un tipo basado en tres rectángulos y un tipo basado en cuatro rectángulos (ver Figura 5.1).

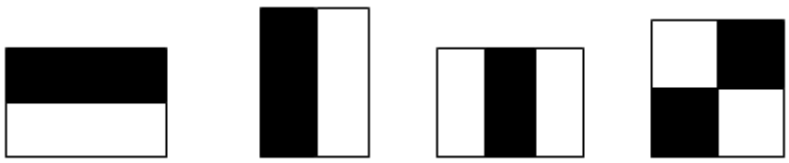


Figura 5.1 Características utilizadas en el detector de Viola-Jones.
De izq. a dcha. nombradas como tipo 1, 2, 3 y 4 respectivamente.

De manera simplificada, las características pueden ser vistas como evaluaciones de la intensidad de conjuntos de píxeles. La suma de la luminancia de los píxeles en la región blanca se resta de la suma de los píxeles en la región oscura. El valor obtenido mediante la diferencia es el valor de la característica y puede combinarse con otros formando hipótesis en regiones de una imagen.

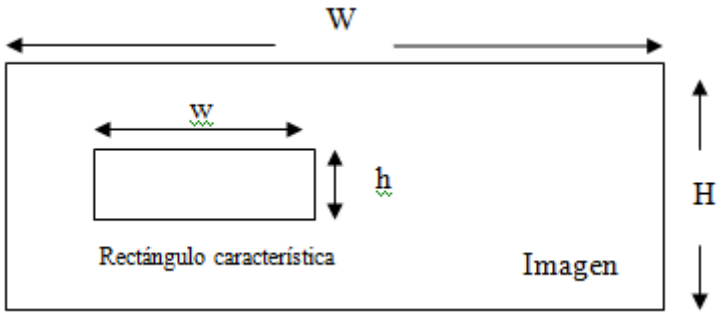


Figura 5.2 Representación de una imagen genérica conteniendo un modelo de característica.

Un hecho de particular relevancia, y no muy desarrollado en las publicaciones de Viola y Jones, es la selección del conjunto de características a partir de las imágenes de entrenamiento, en su caso de 24x24 píxeles. El número de características de cada tipo para cada imagen es muy grande y de ellas se eligen las más adecuadas para la

detección. Un método para calcular el número de características de cada tipo es el propuesto a continuación extraído de la implementación de Lienhart y Maydit [16]. Para realizar estos cálculos, se hace uso de la Figura 5.2 que nos sirve como modelo para identificar los parámetros utilizados en la ecuación de Lienhart y Maydit:

$$F = X \cdot Y \left(W + 1 - w\frac{X+1}{2}\right)\left(H + 1 - h\frac{Y+1}{2}\right)$$

En la ecuación podemos identificar: $\left\lfloor X = \frac{W}{w} \right\rfloor$, $\left\lfloor Y = \frac{H}{h} \right\rfloor$, siendo X e Y los valores máximos de escalado de la característica, W y H la anchura y altura máximas de la imagen respectivamente, w y h la anchura y altura máximas de la característica y F el número de características que se pueden extraer de dicha imagen. Aplicando la fórmula a las características utilizadas por Viola y Jones (descritas en la Figura 5.1) y basándonos en los datos resumidos en la Tabla 5.1 obtenemos el número de características por cada tipo.

Tabla 5.1 Resumen del cálculo del número de características utilizadas por cada imagen de 24x24 píxeles.

Tipo característica	w/h	X/Y	Núm. caract.
Tipo 1	1/2	24/12	43200
Tipo 2	2/1	12/24	43200
Tipo 3	3/1	8/24	27600
Tipo 4	2/2	12/12	20736
Total			134736

5.2.1 Imagen integral

A la hora de crear un sistema de detección facial resulta crucial encontrar un compromiso entre velocidad y eficiencia. Mediante el uso de una nueva representación de las imágenes, llamada imagen integral, Viola y Jones describen un método de evaluación de características de manera efectiva y a mayor velocidad.

El concepto de la imagen integral es fácilmente comprensible. Esta estructura se construye tomando la suma de los valores de luminancia de los píxeles que se encuentran por encima y a la izquierda de un cierto punto en la imagen. Viola y Jones presentan la imagen integral como la integral doble de una imagen primero a lo largo de

las filas y después a lo largo de las columnas. La imagen integral en el punto *(x,y)* viene dada por [1]:

$$ii(x,y) = \sum_{a \leq x, b \leq y} i(a,b)$$

Donde *ii(x,y)* es la imagen integral y *i(x,y)* es el valor de la imagen en unas coordenadas especificas (ver Figura 5.3).

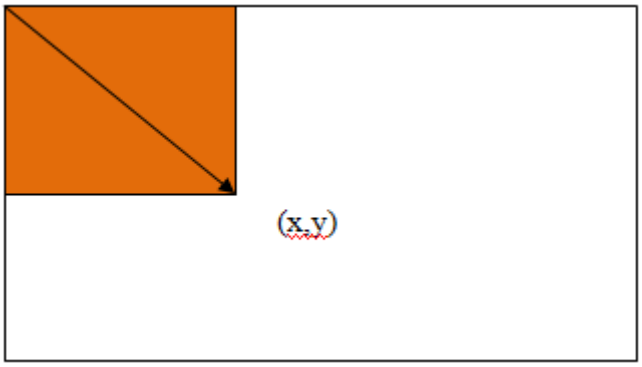


Figura 5.3 El valor de la imagen integral en un punto (x,y) es la suma de los píxeles de arriba a la izquierda.

La importancia de la imagen integral se basa en la capacidad de calcular rápidamente la suma de píxeles dentro de un área determinado de la imagen. Cualquier suma dentro de un área de la imagen puede calcularse utilizando cuatro referencias (ver Figura 5.4). Por lo tanto la diferencia entre dos regiones puede calcularse utilizando 8 referencias dentro de la imagen. Sin embargo, teniendo en cuenta que, por ejemplo los dos primeros tipos de características utilizan dos regiones rectangulares adyacentes la diferencia puede realizarse utilizando 6 referencias, en el caso del tercer tipo se utilizarían 8 referencias y en el cuarto tipo 9 referencias.

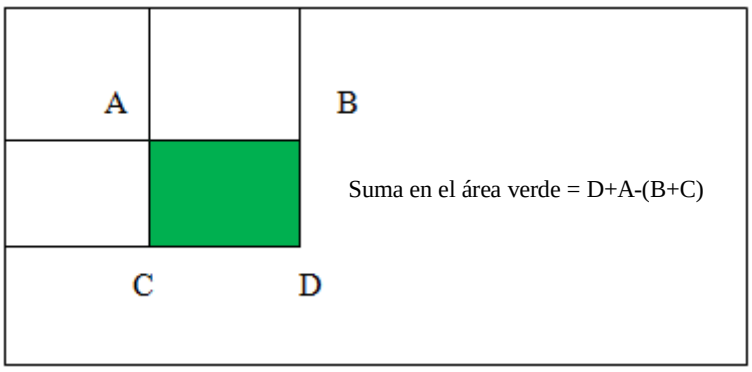


Figura 5.4 Cálculo del valor de la suma de los píxeles dentro de un área.

Al implementarse este sistema utilizando Matlab, el algoritmo para calcular la imagen integral puede ser simplificado utilizando la función *"cumsum()"* de manera repetida: `ii=cumsum(cumsum(double(image)), 2);`

5.3 El clasificador

Una vez encontradas las características dentro de una imagen, más bien dentro de un conjunto de imágenes de entrenamiento, el objetivo del sistema es encontrar aquellas características que mejor definan una cara y nos ayuden a localizarla en una imagen. La hipótesis planteada en las publicaciones de Viola y Jones establece que un pequeño número de esas características pueden combinarse para formar un clasificador. Para ello establecen una variante del algoritmo AdaBoost de Freund y Shaphire [30]. El algoritmo se utiliza para mejorar el rendimiento de un algoritmo de aprendizaje simple. Este algoritmo simple se llama clasificador simple o *weak learner*.

5.3.1 Clasificador simple

La base del sistema de Viola y Jones es la combinación de clasificadores simples o *weak learners* para obtener una clasificación eficiente de los datos. En relación a esto, un *weak lerner*, h_j, es una estructura simple que contiene una característica *f*, un umbral θ y una paridad *p*. La salida del clasificador es binaria y depende de si el valor de una característica se encuentra por encima o debajo de un umbral.

$$h(x, f, p, \theta) = \begin{cases} 1 & si, \quad pf(x) < p\theta \\ 0 & en\ caso\ contrario \end{cases}$$

Los clasificadores simples que se utilizan pueden verse como nodos de decisión en estructuras en árbol. Teniendo en cuenta que hay una gran cantidad de características en una imagen de 24x24 píxeles, el algoritmo AdaBoost debe seleccionar las características que mejor diferencien entre caras y no caras dentro de un conjunto de imágenes, y en él recala la mayor parte del trabajo del entrenador.

La clasificación es un componente muy importante en los sistemas inteligentes. El problema que se plantea a la hora de realizar una clasificación consiste en decidir a qué *"clase"* pertenece un objeto. Sobre dicho objeto realizamos varias mediciones, que son las llamadas *"características"*, en las que nos basamos a lo largo de este proyecto. Por ello, lo que nos interesa es aprender y encontrar una relación entre dichas características y las diversas clases a las que nos enfrentamos. Para implementar los *weak learners*, en cada uno de los clasificadores fuertes o *strong classifiers* tenemos varias opciones utilizando diversos sistemas de aprendizaje: *SVM, perceptron, MLP, Naive Bayesian Classifiers...*

La selección de características se implementa del siguiente modo: Para cada ronda de *boosting*:

1. Evaluar cada *feature*, aplicando cada filtro rectangular, sobre cada imagen de ejemplo.

2. Ordenar las imágenes según los valores obtenidos en el paso anterior.

3. Seleccionar el mejor umbral para cada característica.

4. Seleccionar el mejor filtro/umbral, es decir, la mejor feature.

5. Actualizar los pesos.

El método puede verse contextualizado dentro del algoritmo Adaboost descrito en el siguiente apartado. La descripción de este método será más exhaustiva y pormenorizada en el capítulo referente a la implementación del detector.

5.3.2 Algoritmo Adaboost

El algoritmo Adaboost, descrito en la Tabla 5.2, se utiliza para seleccionar los mejores clasificadores simples. Teniendo en cuenta que existe un clasificador simple por cada característica tenemos que evaluar un total de $K \cdot N$ clasificadores siendo K el número de características por imagen y N el número de imágenes que contenga nuestro set de entrenamiento.

La ventaja principal de AdaBoost es su velocidad de aprendizaje. En este algoritmo los pesos se asignan de manera que favorezcan la clasificación de las caras, consiguiendo que estos ejemplos tengan mayores pesos o importancia.

Las mejores características se eligen basándose en el error ponderado que se produce. Este error ponderado es una función que utiliza los errores pertenecientes a los ejemplos de entrenamiento. El peso de un ejemplo clasificado correctamente se modifica mientras que el peso de un ejemplo mal clasificado se mantiene constante. Con esto se consigue que sea más difícil que la segunda característica clasifique erróneamente un ejemplo que haya sido clasificado erróneamente por la primera característica frente a un ejemplo clasificado correctamente por esa primera característica. Otra manera de ver esto sería que la segunda característica, y las sucesivas, se ven forzadas a tomar más en cuenta los ejemplos clasificados erróneamente por características anteriores.

Tabla 5.2 Algoritmo AdaBoost utilizado en el sistema [1].

- Dadas imágenes de ejemplo $(x_1,y_1),\ldots,(x_n,y_n)$ donde y_i= -1; 1 para ejemplos negativos o positivos de caras respectivamente, y T el número de hipótesis a construir.
- Inicializa los pesos $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ para y_i= -1; 1 respectivamente, donde m y l son el número de ejemplos negativos y positivos respectivamente.
- For t = 1, … , T
    1. Normalizar los pesos, $w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$
    2. Seleccionar el mejor clasificador simple respecto al error ponderado
    $$\epsilon_t = min_{f,p,\theta} \sum_i w_i |h(x_i, f, p, \theta) - y_i|$$
    3. Definir $h_t(x) = h(x, f_t, p_t, \theta_t)$, donde f_t, p_t, θ_t son los valores que minimizan ϵ_t.
    4. Actualizar los pesos:
    $$w_{t+1,i} = w_{t,i}\beta_t^{1-e_i}$$
    donde $e_i = 0$, si la imagen x_i esta correctamente clasificada, o $e_i = 1$ en caso contrario; y $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$
- El clasificador final sería:
$$C(x) = \begin{cases} 1 & si \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2}\sum_{t=1}^{T} \alpha_t \\ 0 & en\ caso\ contrario \end{cases}$$
donde $\alpha_t = log\frac{1}{\beta_t}$

Después de sucesivas iteraciones del algoritmo el resultado es un conjunto de hipótesis ponderado que conforma un clasificador fuerte. La hipótesis final encontrada después de T iteraciones se obtiene al concluir el algoritmo, donde la clasificación binaria (cara Vs. no-cara) se realiza en función de los pesos individuales α (representando los umbrales) y la suma del producto de cada peso y la clasificación de cada ejemplo en particular $\alpha_t h_t(x)$.

5.4 El detector de clasificadores en cascada

El principio básico del algoritmo de detección facial de Viola y Jones consiste en escanear el detector muchas veces a través de una misma imagen, en diferentes posiciones y a distintas escalas. Incluso si una imagen contiene muchas caras está claro que la mayoría de las sub-ventanas que se escaneen no contendrán ninguna cara. Esto lleva a una nueva manera de ver el problema: *En vez de encontrar caras, el algoritmo debería descartar 'no-caras'*.

La idea subyacente se basa en que es más fácil descartar una imagen que no contenga una cara que encontrar una cara en una imagen. Con esto en la cabeza, parece ineficiente construir un detector que contenga un solo clasificador fuerte ya que el tiempo de clasificación será constante sin importarnos la entrada del clasificador, ya que el clasificador tiene que evaluar todas las características que lo forman. Aumentar la velocidad de clasificación generalmente implica que el error de clasificación aumentará

inevitablemente, ya que para disminuir el tiempo de clasificación se debería disminuir el número de clasificadores simples que se utilizan. Para evitar esto Viola y Jones proponen un método para reducir el tiempo de clasificación manteniendo los requerimientos de rendimiento del clasificador.

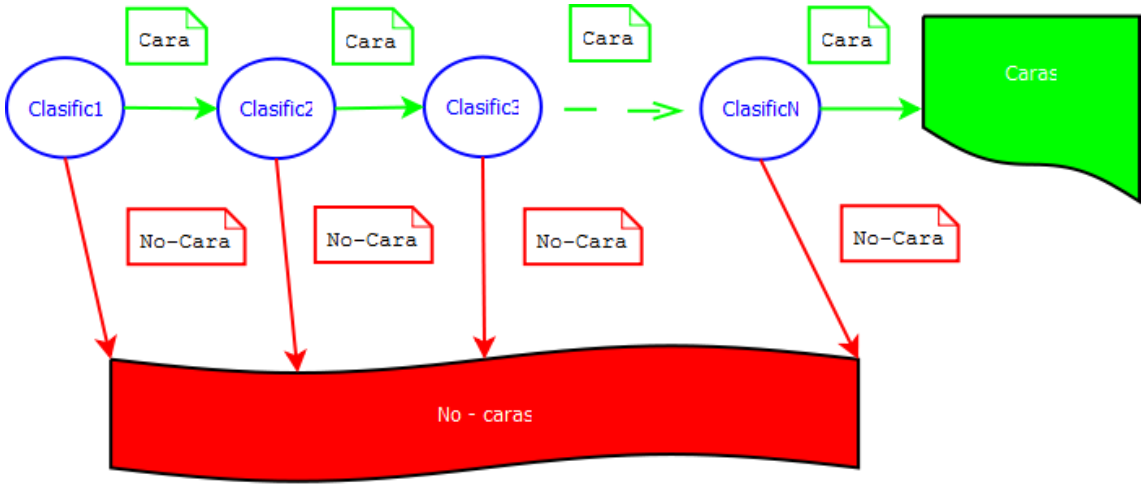


Figura 5.5 Estructura de clasificadores en cascada.

Este método consiste en el uso de una cascada de clasificadores fuertes. El trabajo en cada etapa del clasificador consiste en determinar si la sub-ventana que se analiza es definitivamente una no-cara o podría ser una cara. Cuando una sub-ventana es clasificada como no cara en alguna de las etapas del detector, se descarta inmediatamente. En caso contrario, si se clasifica como una posible cara, pasa a la siguiente etapa del clasificador. Se identificará una sub-ventana como contenedora de una cara si y sólo si pasa a través de todas las etapas del detector de forma positiva (ver Figura 5.5).

Si se utilizase un clasificador que tuviese un único estado, normalmente habría que aceptar los falsos negativos para reducir la tasa de falsos positivos. Sin embargo, para las primeras etapas del clasificador en cascada se acepta una alta tasa de falsos positivos esperando que las etapas posteriores puedan encargarse de reducir esta tasa mediante clasificadores más especializados. Con esto se pretende también reducir la tasa de falsos negativos en el clasificador final, ya que una sub-ventana será clasificada como cara sólo en el caso de que haya pasado por todas las etapas del clasificador.

5.4.1 Entrenamiento de la cascada

El proceso de diseño de la cascada empleado por Viola y Jones se basa en objetivos de detección y rendimiento similares a lo que se hizo hasta el momento. Los sistemas anteriores obtenían tasas de detección de entre el 85% y el 95% y tasas de falsos positivos extremadamente bajas del orden de 10^{-5} [1]. Es por ello que el sistema debería tener un número de etapas suficientes para obtener resultados similares.

Dada una cascada de clasificadores, la tasa de falsos positivos se calcularía como sigue:

$$F = \prod_{i=1}^{K} f_i$$

Donde F es la tasa de falsos positivos del detector, K es el número de clasificadores, y f_i es la tasa de falsos positivos calculada para el clasificador i. Por otro lado la tasa de detección de la cascada se calcularía según:

$$D = \prod_{i=1}^{K} d_i$$

Donde D es la tasa de detección del detector, K es el número de clasificadores, y d_i es la tasa de detección calculada para el clasificador-i. Una vez establecido esto, vemos como por ejemplo un detector consistente en 10 etapas con una tasa de detección del 99% por etapa y una tasa de falsos positivos del 30% por clasificador nos llevaría a una tasa de detección global del $0.99^{10} \approx 0.9$ y una tasa de falsos positivos global del $0.3^{10} \approx 6x10^{-6}$.

El proceso de entrenamiento debe considerar las limitaciones a las que se ve sometido. En la mayoría de los casos, los clasificadores construidos utilizando más características tendrán una mayor tasa de detección y una menor tasa de falsos positivos. Al mismo tiempo, los clasificadores con más características necesitarán más tiempo para determinar si una sub-ventana contiene o no una cara. A la hora de entrenar el clasificador uno debe optimizar el número de etapas del clasificador, el número de características y el umbral de cada etapa. Sin embargo, realizar esta optimización es un trabajo tremendamente costoso.

Para simplificar este problema los autores han realizado un algoritmo para poder entrenar de manera efectiva la cascada de clasificadores (ver Tabla 5.3). En este caso el usuario elige las tasas de falsos positivos y de detección de cada etapa así como la tasa de falsos positivos referente a todo el detector. Cada etapa del detector se entrena utilizando AdaBoost del modo descrito en la Tabla 5.2, incrementando el número de características de la etapa hasta que se obtengan las tasas de falsos positivos y de detección deseadas. Dichas tasas se determinan confrontando el detector con un set de validación. Si la tasa de falsos positivos global no es la que interesa se añade otra etapa al clasificador.

Tabla 5.3 Algoritmo de entrenamiento para construir un detector en cascada [1].

- El usuario selecciona el valor de *f*, máx. tasa de falsos positivos aceptable por etapa, y de *d*, mín. tasa de detección aceptable por etapa.
- El usuario selecciona la tasa de falsos positivos global para el detector, F_{target}.
- *P* = conjunto de ejemplos positivos.
- *N* = Conjunto de ejemplos negativos.
- $F_0 = 1.0$; $D_0 = 1.0$
- $i = 0$
- mientras $F_i > F_{target}$
  - $i \leftarrow i + 1$
  - $n_i = 0$; $F_i = F_i\text{-}1$
  - mientras $F_i > f \times F_i\text{-}1$
    - * $n_i \leftarrow n_i + 1$
    - * Utilizar *P* y *N* para entrenar un clasificador con ni características utilizando AdaBoost.
    - * Evaluar el clasificador en cascada actual sobre el set de validación para determinar F_i y D_i.
    - * Disminuir el umbral para el clasificador-*i* hasta que el actual clasificador en cascada tenga una tasa de detección de al menos $d \times D_i$ -1 (esto también afecta a Fi).
  - N ← Ø
  - Si $F_i > F_{target}$ evaluar el detector en cascada actual utilizando el set de no-caras y poner todos los falsos positivos en el set N.

Por otro lado, con el objetivo de demostrar que el clasificador en cascada es más veloz frente a un solo clasificador fuerte y, además, no pierde calidad en la detección, se entrenó un clasificador con una etapa de 200 características por un lado, y otro con 10 etapas de 20 características cada una. En la Figura 5.6 se puede ver la curva ROC combinando ambos clasificadores. A partir de la misma podemos ver una pequeña diferencia en términos de calidad, sin embargo sabemos que la diferencia en términos de velocidad es mucho mayor.

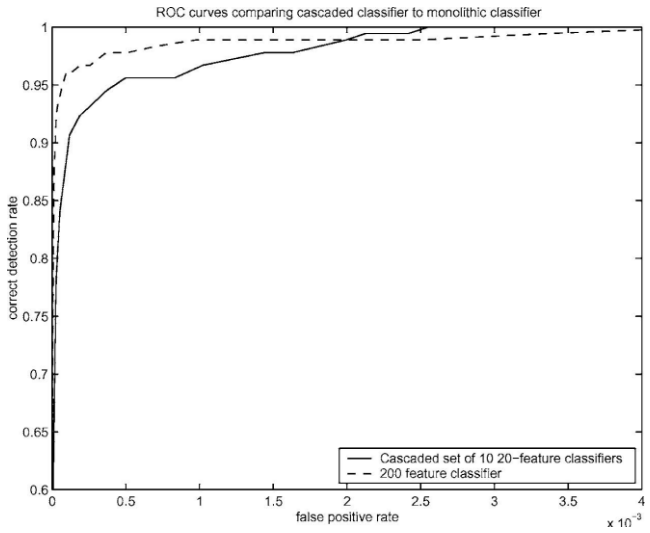


Figura 5.6 Curvas ROC comparando un clasificador de 200 características con uno en cascada de 10 etapas de 20 características cada una. El rendimiento no es muy diferente mientras que la velocidad del detector en cascada es casi 10 veces mayor [1].

5.5 Resultados

En esta sección se dará una breve visión de los resultados obtenidos por Viola y Jones según sus publicaciones [1]. A lo largo de este apartado se comentará el conjunto de entrenamiento utilizado, la estructura del detector que conseguido y el procesado de imágenes de diferentes tamaños, donde haya que integrar múltiples detecciones en diferentes localizaciones y a diferentes escalas.

5.5.1 Conjunto de entrenamiento

El conjunto de entrenamiento consiste en 4916 caras con una resolución de 24x24 píxeles y etiquetadas a mano. Algunas de esas imágenes se muestran en la Figura 5.7. Viola y Jones remarcan la diferencia entre su conjunto de entrenamiento y el utilizado por Rowley et al. [21] el cual usaba imágenes de 16x16 píxeles. Presumiblemente con los ejemplos de Viola y Jones se obtienen mejores resultados y la información contenida en estas imágenes puede ser utilizada para descartar no-caras en etapas previas de la cascada.

Figura 5.7 Ejemplo de imágenes frontales utilizadas por Viola y Jones [1].

5.5.2 Estructura del detector

El detector final obtenido por Viola y Jones contiene una cascada de 38 etapas con un total de 6060 características. El primer clasificador en cascada utiliza dos características y rechaza aproximadamente el 50% de las no-caras mientras que detecta correctamente casi el 100% de las caras. El siguiente clasificador tiene 10 características y rechaza el 80% de las no-caras detectando casi el 100% de las caras. Las siguientes dos etapas tienen clasificadores de 25 características y las siguientes 3 etapas tienen clasificadores de 50 características y así sucesivamente.

Es interesante remarcar que Viola y Jones eligieron de forma manual el número de características de los 7 primeros estados para reducir el tiempo de entrenamiento del detector, lo que nos hace ver que a pesar de la velocidad que pueda tener el detector a la

hora de trabajar, su entrenamiento es tremendamente costoso, con lo que los mismos autores reconocen haber modificado ligeramente el algoritmo presentado en la Tabla 5.3 para facilitar la tarea computacional [1]. El tiempo de entrenamiento utilizado para el detector con sus 38 etapas en una máquina 466MHz AlphaStation XP900 fue del orden de semanas.

5.5.3 Procesado de imágenes

Todos los ejemplos utilizados para el entrenamiento eran de imágenes normalizadas con el objetivo de minimizar el efecto de las diferentes condiciones de iluminación. Por lo tanto, parece lógico entender que la normalización es también necesaria para la detección.

Por otro lado el detector se aplica sobre imágenes de diversos tamaños y las caras que aparecen en las mismas contienen también distintos tamaños. Por ello el sistema debe ser capaz de extraer sub-ventanas de una imagen y analizarlas. Según los autores se obtienen buenos resultados escalando el tamaño de estas sub-ventanas utilizando un factor de 1.25. El detector también escanea una imagen en diversas localizaciones, con lo que las sub-ventanas se mueven a lo largo de la imagen un número determinado de píxeles, Δ. La elección de este número afectará tanto a la velocidad del detector como a su rendimiento. En la publicación de Viola y Jones se muestran resultados utilizando tanto $\Delta = 1$, como $\Delta = 1.5$, utilizando escalas de 1 y 1.25 respectivamente (ver Figura 5.8).

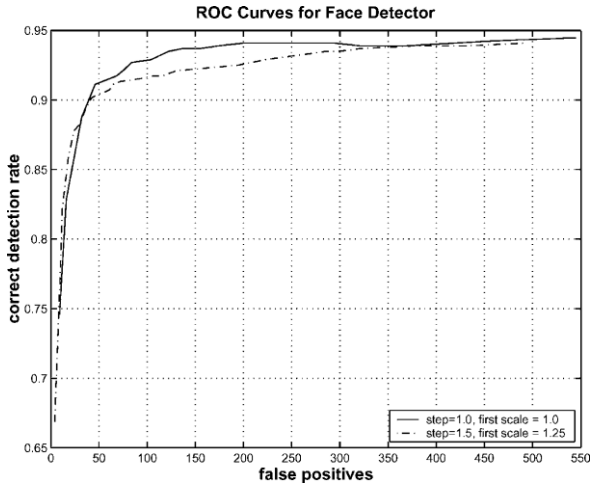


Figura 5.8 Curvas ROC para el detector de Viola y Jones sobre la base de datos MIT+CMU [1].

Debido a que normalmente se producirán múltiples detecciones alrededor de una cara dentro de una imagen, se deben combinar estas detecciones en una sola. Una manera de hacerlo sería realizar una media de la posición de las esquinas de los recuadros de todas las detecciones sobre una misma cara.

5.6 Conclusiones del trabajo de Viola y Jones

En este capítulo se ha presentado el detector facial de Viola y Jones que servirá de base al trabajo realizado en este proyecto. Este detector se ha escogido debido a las contribuciones realizadas, enumeradas en la introducción del capítulo, así como por su velocidad a la hora de detectar caras en imágenes, sin olvidarnos de la popularidad que ha adquirido gracias a la implementación OpenCV [33].

6

IMPLEMENTACIÓN DEL SISTEMA

El objetivo del proyecto consiste en la construcción de un sistema de detección y localización facial. Una vez construido dicho sistema se podría hacer extensible a la detección de otro tipo de objetos, como por ejemplo la detección de ojos que también ha sido implementada. Para implementar dicho sistema se ha pasado por varias etapas que son descritas a lo largo de este capítulo.

6.1 Etapas de la implementación

En la tabla inferior (Tabla 6.1) se adjunta la descripción de las sucesivas etapas en las que consistió la implementación.

Tabla 6.1 Etapas de la implementación

Etapa	Duración temporal	Tarea
Documentación general.	Febrero 2009 – Mayo 2009	Recopilación de información acerca del funcionamiento del detector de Viola & Jones.
Matlab	Junio 2009 –Noviembre 2009	Implementación exclusiva en Matlab del detector.
Documentación implementación Matlab + C	Noviembre 2009 – Diciembre 2010	Recopilación de información acerca del funcionamiento de las implementaciones en Matlab y C.
Matlab + C	Enero 2010 – Abril 2010	Implementación del detector en Matlab + C.

En los sucesivos puntos del capítulo se analizará el trabajo y los resultados obtenidos en cada una de las etapas anteriormente indicadas.

6.2 Implementación en Matlab

Después de una primera etapa en la que se realizó un trabajo de documentación para entender la tecnología planteada, se procedió a comenzar la implementación del sistema de detección, la cual estuvo planeada para Matlab.

Utilizando este programa se implementó el cálculo individual de uno de los clasificadores que forman la cascada del detector de Viola&Jones. Dicho clasificador utilizaba en un principio 10 features extraídas de 111 imágenes de la base de datos utilizada por Viola y Jones [1]. Esta primera prueba sirvió para tener un primer contacto con la tecnología. El tiempo que tardaba en entrenar y evaluar clasificar el conjunto de imágenes del entrenamiento rondaba los 34 minutos. Para evaluar el pequeño detector se utilizaron las mismas imágenes. Por un lado el objetivo era implementar un evaluador y evaluar el funcionamiento del clasificador entrenado, y por otro ver cuanto tiempo tardaba en realizarse dicha evaluación.

Hay que tener en cuenta que los datos obtenidos en esta prueba no fueron más que orientativos, pues ni la base de datos utilizada para el entrenamiento, ni el clasificador entrenado podrían ser considerados como representativos. En la Figura 6.1 podemos ver la relación entre el número de weak learners o características utilizadas y la exactitud del detector al clasificar el conjunto de imágenes de entrenamiento.

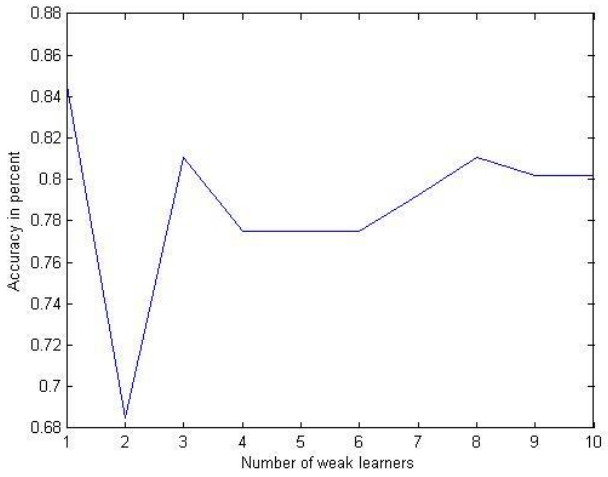


Figura 6.1 Primer entrenamiento del detector.

Al evaluar los resultados obtenidos en esta primera aproximación procedimos a escalar el clasificador con el objetivo de obtener una implementación que pudiese considerarse competitiva frente al resto de detectores existentes.

Para ello lo primero que tuvimos que hacer fue desarrollar el cálculo de todos los tipos de características que iban a ser utilizados (ver Figura 6.2).

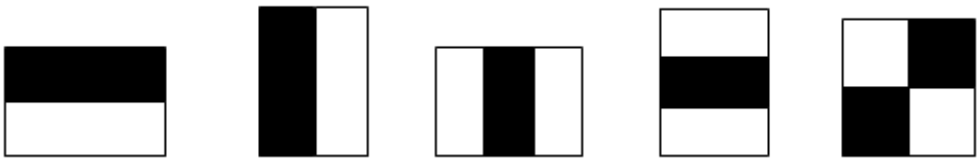


Figura 6.2 Características utilizadas en la implementación del detector. Añadida la imagen situada a la derecha. De izq. a dcha. nombradas como tipo 1, 2, 3, 4 y 5 respectivamente.

En la prueba anterior simplemente se utilizo una característica, la izquierda de la Figura 6.2, con lo cual se hacía necesaria la incorporación del resto de los tipos de características. En la Tabla 6.2 se muestra una descripción de las características utilizadas.

Tabla 6.2 Desglose de características utilizadas.

Tipo característica	w/h	X/Y	Núm. caract.
Tipo 1	1/2	24/12	43200
Tipo 2	2/1	12/24	43200
Tipo 3	3/1	8/24	27600
Tipo 4	1/3	24/8	27600
Tipo 5	2/2	12/12	20736
Total			162336

Una vez implementada esta funcionalidad se procedió a lanzar el entrenamiento del detector con las 5 características y utilizando la totalidad de las imágenes de Viola&Jones, es decir, 12788 imágenes de entre las cuales 4916 correspondían a caras y 7872 a no caras.

El entrenamiento se planteo de manera similar al propuesto por Viola y Jones pero utilizando la descripción realizada por Ming-Hsuan Yang en la *International Conference on Pattern Recognition* [34]. A partir de las publicaciones de Viola y Jones

sobre el tema, no queda perfectamente clara la composición del detector. Por tanto se utilizó la siguiente (Tabla 6.3):

Tabla 6.3 Descripción de las etapas del detector que se planteo al principio de la implementación.

Detector de 32 etapas	
Etapa 1	2 características
Etapa 2	5 características
Etapas 3-5	20 características
Etapas 6-7	50 características
Etapas 8-12	100 características
Etapas 13-32	200 características
Total características:	4667 características

Si comparamos este planteamiento con el ofrecido por Viola y Jones [1] vemos que frente a 6060 características elegidas por ellos, nosotros utilizaríamos sólo 4667. Esta decisión se tomo por una cuestión temporal ya que el entrenamiento tardaba demasiado y el objetivo de dicho proyecto no es conseguir un sistema completamente entrenado si no un algoritmo funcional que pueda ser escalado en un futuro haciendo básicamente una inversión en tiempo de entrenamiento.

Cabe destacar que el entrenamiento ideal propuesto por los autores en sus publicaciones [1] [28] fue ajustado por ellos mismos como bien explican en su propia publicación: *"El procedimiento utilizado para elegir el número de características por etapa fue guiado por la intervención humana (para las primeras 7 etapas) para reducir el tiempo de entrenamiento del detector"*. Por otro lado también comentan como el algoritmo que proponen fue modificado, sin indicar explícitamente cómo, especificando el número mínimo de características y eligiendo más de una característica cada vez. Esto ocurre porque el algoritmo que plantean (mostrado en la Tabla 5.3) emplearía un tiempo más que considerable en llevar a cabo el entrenamiento.

6.2.1 Resultados obtenidos

Los resultados obtenidos de esta implementación tuvieron que ser descartados debido a varios factores: excesivo tiempo de entrenamiento, excesivo tiempo de detección y calidad del detector.

El detector obtenido no llego a ajustarse ni a escalarse para ser utilizado con imágenes de varios tamaños, sino que fue entrenado y evaluado utilizando un subconjunto de las imágenes de Viola y Jones. Las pruebas propuestas fueron las siguientes (Tabla 6.4):

Tabla 6.4 Primeras pruebas planteadas

Pruebas	Test 1	Test2	Test3	Test4	Test32
Training images	2000	4000	6000	5000	12788
Faces	500	1000	3000	4000	4916
Non-faces	1500	3000	3000	1000	7872
Validation set	3000	7916	5916	6916	12788
Faces	500	3916	1916	916	4916
Nonfaces	2500	4000	4000	6000	7872

No todas las pruebas llegaron a ser realizadas ya que enseguida observamos la inviabilidad del sistema debido al tiempo de entrenamiento y posteriormente al de detección.

Extrapolando los resultados obtenidos, el entrenamiento de un detector competitivo en el que se utilizase un número de imágenes razonable (las 12788 de Viola) y se obtuviesen un número de características aceptable (las indicadas anteriormente en la Tabla 6.2) habría tardado meses.

Por otro lado el tiempo requerido para la detección facial en una imagen de tamaño 200x300, 400x500 o similar, duraría entre 30 minutos y una hora.

La conclusión extraída es que un detector de esta envergadura no puede llevarse a cabo de manera eficiente utilizando exclusivamente Matlab, debido a la lentitud del programa a la hora de trabajar, independientemente de la exactitud de los resultados obtenidos en la detección. Estos factores nos llevaron a descartar este detector y a buscar otras soluciones de implementación. Para una mayor eficiencia y velocidad se utilizó C/C++ junto a Matlab que será explicado en los capítulos sucesivos.

6.3 Implementación final del detector con archivos MEX

A lo largo de los capítulos anteriores se ha explicado tanto el funcionamiento y la estructura del detector de Viola y Jones que se pretende implementar, como la

estructura de programación que se pretende seguir. En este apartado se va a explicar la estructura final del detector.

6.3.1 Estructura general del entrenador

Para comenzar el entrenamiento del detector, partimos de un archivo .m en el que se implementó la extracción de los datos iniciales necesarios para la detección:

- 12788 imágenes de Viola y Jones de 24x24 píxeles (4916 caras frente a 7872 no caras) en escala de grises.

- Estructura de las características que se extraen para entrenar. Como ya se indicó se utilizan 5 tipos distintos de características (ver Tabla 6.2), pero cada característica está formada por 2, 3 o 4 rectángulos que deben ser definidos. Para ello se creó una variable que recogiese estos datos. Cada una de sus columnas representa cada uno de los rectángulos que estructura se desglosa en la :

Tabla 6.5 Desglose de la estructura de las *features* utilizadas.

Dato representado
Índice del patrón, tipo de característica: (1, 2, 3, 4, 5)
Ancho en píxeles de la característica
Alto en píxeles de la característica
Nº total de rectángulos que contiene dicha característica
Índice del rectángulo dentro de la característica
Coordenada *x* del rectángulo en el conjunto de la característica
Coordenada *y* del rectángulo en el conjunto de la característica
Ancho en píxeles del rectángulo en cuestión
Alto en píxeles del rectángulo en cuestión
Índice de la operación del valor que representa el rectángulo (±1)

A partir de esta base se procede a realizar el entrenamiento, el cual consta de las siguientes etapas:

1. Cálculo de la imagen integral. El cálculo es inmediato y no gasta un tiempo significativo (alrededor de 1.5 segundos). El método de cálculo ya fue explicado en el apartado 5.2 y no presenta mayor complicación.

2. Cálculo de las características de *haar* sobre el conjunto de imágenes de entrenamiento. Para ello se creó una función que haciendo uso de la variable en la que se definen los rectángulos de las características (anteriormente descrita) y las dimensiones de las imágenes de entrenamiento (24x24 píxeles en nuestro caso) definimos la localización de todas las características adaptadas para nuestro modelo de imágenes de entrenamiento (ver Tabla 6.6). Con esto se crea un código generalizado que deja abierta la posibilidad de introducir imágenes de distintas dimensiones como set de entrenamiento. Esto puede ser útil si lo

que queremos implementar es un detector de otro tipo de objetos. La estructura final de nuestra variable es de 6 filas x 162336 columnas.

Para conseguir que este cálculo se llevase a cabo de una manera más rápida se implementó utilizando archivos MEX (ver Anexo III Manual del programador).

Tabla 6.6 Estructura de la definición de las características dentro del conjunto de entrenamiento

Dato representado en cada columna
Índice del patrón, tipo de característica: (1, 2, 3, 4, 5)
Coordenada *x* de posicionamiento en la imagen
Coordenada *y* de posicionamiento en la imagen
Ancho en píxeles de la característica en cuestión
Alto en píxeles de la característica en cuestión
Índice de la característica en la variable de definición de las *features.*

3. Cálculo de la cascada de datos utilizando archivos MEX para aumentar la velocidad del cálculo (ver Anexo III Manual del programador) .

6.3.2 Cálculo de la cascada de clasificadores

El cálculo de la cascada siguió el modelo AdaBoost descrito por Viola y Jones, procediendo según lo indicado en sus publicaciones [1] [28] y en capítulos anteriores. La elección de los *weak classifiers* que componen cada clasificador se basa en los valores de las características a lo largo de todas las imágenes de entrenamiento y en la actualización de los pesos asociados (ver Figura 6.3).

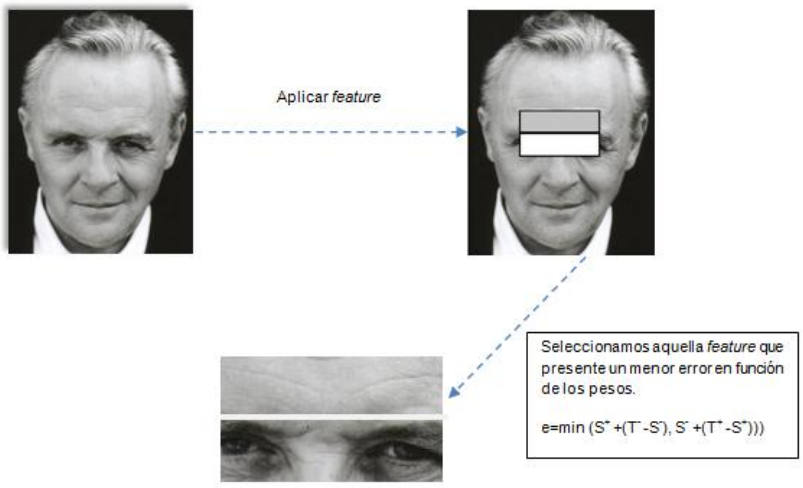


Figura 6.3 Extracción de la *best-feature*

Para elegir la característica adecuada, las imágenes se ordenan según el valor de la característica que se evalúa en cada momento. El umbral de decisión óptimo para la característica en cuestión se calcula utilizando cuatro valores: T^+, suma total de los pesos de los ejemplos de entrenamiento positivos, T^-, suma total de los pesos de los ejemplos negativos, S^+, suma de los pesos de los ejemplos positivos que se encuentran

por debajo del que se está evaluando, y S^- suma de los pesos de los ejemplos negativos que se encuentran por debajo del ejemplo que se está evaluando.

Al elegir una característica y aplicar un umbral de decisión, la evaluación del error que conlleva es fácilmente calculable aplicando la fórmula:

$$e = \min(S^+ + (T^- - S^-), S^- + (T^+ - S^+))$$

Con esta ecuación evaluamos el error de elegir entre clasificar todos los ejemplos por debajo del actual como negativos y los que se encuentran por encima positivos, y viceversa. Elegimos el error mínimo ya que al ser un decisor binario basta con cambiar el signo de la elección para conseguir el error mínimo de manera inmediata.

Con esta configuración se entrenó una cascada de 8 clasificadores y un total de 100 características distribuidas a lo largo de dichas etapas, para observar los resultados de la implementación y poder evaluar la viabilidad y exactitud del mismo. Los resultados obtenidos, que fueron positivos, se describen en el siguiente capítulo.

Una vez analizado el rendimiento del detector, se procedió a realizar la implementación del detector final que nos interesaba obtener. Analizando el tiempo de entrenamiento del detector, se observó que se había reducido ampliamente frente a la implementación llevada a cabo anteriormente en Matlab, con lo cual el objetivo inicial se había conseguido sin perder calidad a la hora de realizar la detección. Sin embargo, el tiempo de entrenamiento seguía siendo considerablemente alto y se optó por implementar un sistema con un menor número de características que pudiese dar unos resultados relativamente aceptables para poder evaluar el funcionamiento del detector. Se deja por tanto como trabajo futuro el entrenamiento de un detector compuesto de un mayor número de características. La única dificultad que plantearía dicho trabajo futuro sería el empleo de tiempo en la tarea del entrenamiento, pues la implementación y el correcto funcionamiento del detector quedan mostrados a lo largo de este proyecto.

Finalmente se ha implementado un detector de 11 clasificadores fuertes con un total de 532 *weak learners* cuya distribución a lo largo del sistema queda descrita en la Tabla 6.7. Se intentó mediante esta distribución de clasificadores mantenerse lo más fiel posible a la implementación propuesta por Ming-Hsuan Yang descrita anteriormente [34]. A pesar de que el tiempo de entrenamiento se vio drásticamente reducido, el total empleado por este detector fue de aprox. 7 días.

Tabla 6.7 Estructura del detector final implementado.

Etapa	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Nº de *weaks*	2	5	10	20	20	50	50	75	100	100	100

Los resultados fueron evaluados sobre la base de datos CMU y el detector final fue ajustado en relación a los resultados obtenidos sobre dicha base de datos. Siguiendo

las indicaciones de Viola y Jones [1] se planteó un detector que fuese más laxo a la hora de eliminar falsos positivos en las primeras etapas y más restrictivo en las últimas. Esto se consigue no sólo mediante el variable número de características ya mostrado en la Tabla 6.7, sino además modificando el umbral de decisión de los clasificadores que componen el detector. Así en las primeras etapas el umbral fue bajado reduciendo las restricciones impuestas a la hora de detectar las caras, y respectivamente el umbral se subió en las últimas etapas para evitar así la proliferación de falsos positivos. Con dichas decisiones, a pesar de que se obtiene una menor detección, también se obtiene un menor número de falsos positivos que pueden llegar a ser igual de inconvenientes según la aplicación a la que adapte el detector.

Una vez evaluado el sistema entrenado y observado sus resultados, el mismo entrenamiento se planteó con imágenes de ojos con el objetivo de conseguir un detector de ojos en imágenes. Por otro lado, después de crearse la estructura del sistem,a el entrenamiento de los ojos podría ser prácticamente inmediato y el trabajo realizado por el entrenador, sería el mismo que el realizado en el caso de las caras.

En este caso se utilizó la base de datos de BiosecurID [35] para el entrenamiento. Esta base de datos está formada por 12928 imágenes obtenidas de 404 usuarios en 4 sesiones diferentes en cada una de las cuales se obtuvo una imagen de ambos ojos por separado (ver Figura 6.4). El conjunto final estaba compuesto por 5000 ojos, 2500 derechos junto a 2500 izquierdos, frente a 7872 imágenes de *no ojos*. Estas últimas imágenes son las mismas utilizadas en el entrenamiento del detector facial. Con esto se pretendía repetir el entrenamiento ya realizado para la detección de caras intentando obtener resultados similares.

Figura 6.4 Ejemplo base de datos BioSecurID [35].

Los ojos utilizados fueron reajustados desde su tamaño original al de 24x24 píxeles para mantener la similitud con el conjunto de entrenamiento de Viola y Jones y poder así aprovechar su conjunto de imágenes negativas sin tener que crear otro conjunto.

El detector de ojos final fue implementado con la misma estructura que el facial en cuanto a la distribución de las etapas (ver Tabla 6.7). Por otro lado se mantuvieron los mismos ajustes en cuanto al número de *features* por cada etapa.

6.3.3 Implementación del detector

Una vez implementado el entrenamiento del detector, se hace necesaria la implementación de la detección. La tarea de la detección no debe ser solamente eficaz sino también rápida, ya que en ningún caso podría ser útil ningún sistema que emplease demasiado tiempo en realizar esta tarea. Por tanto, la implementación de esta parte del proyecto era necesario hacerla utilizando archivos MEX (ver Anexo III Manual del programador).

La estructura del detector recibe: la imagen sobre la que trabajará, la cascada, y los ajustes de escalado y movimiento de la sub-ventana a lo largo de la imagen.

En el capítulo 5 del presente trabajo ya se explicó la implementación y funcionamiento genérico del detector. Los ajustes realizados en esta implementación fueron hechos evaluando los resultados obtenidos sobre la base de datos CMU.

El detector se aplica a lo largo de la imagen en distintas posiciones y a diferente escala ya que se debe considerar la existencia de más de una cara y de diferentes tamaños. El escalado se consigue escalando el detector en vez de la imagen. Este proceso tiene sentido ya que las características pueden ser evaluadas a cualquier escala con el mismo coste. Viola y Jones aconsejan utilizar un factor de escala de 1.25 [1].

El detector también se aplica en diferentes posiciones a lo largo de la imagen. En nuestra implementación se eligieron 1.5 píxeles de desplazamiento inicial. Tanto el tamaño de la sub-ventana que se evalúa como el tamaño del desplazamiento de la misma se incrementan utilizando el factor de escala. Estos valores son los indicados por Viola y Jones.

Acto seguido, el detector debe eliminar tanto la mayor cantidad de falsos positivos, como de detecciones redundantes. Una imagen en la que se observen alrededor de una cara muchas detecciones es igual de molesta que una en la que haya muchos falsos positivos. Por tanto se implementó un algoritmo encargado de la fusión de las detecciones en el caso de que tengan el mismo tamaño y compartan al menos un 25% de su área.

6.4 Descripción general del sistema

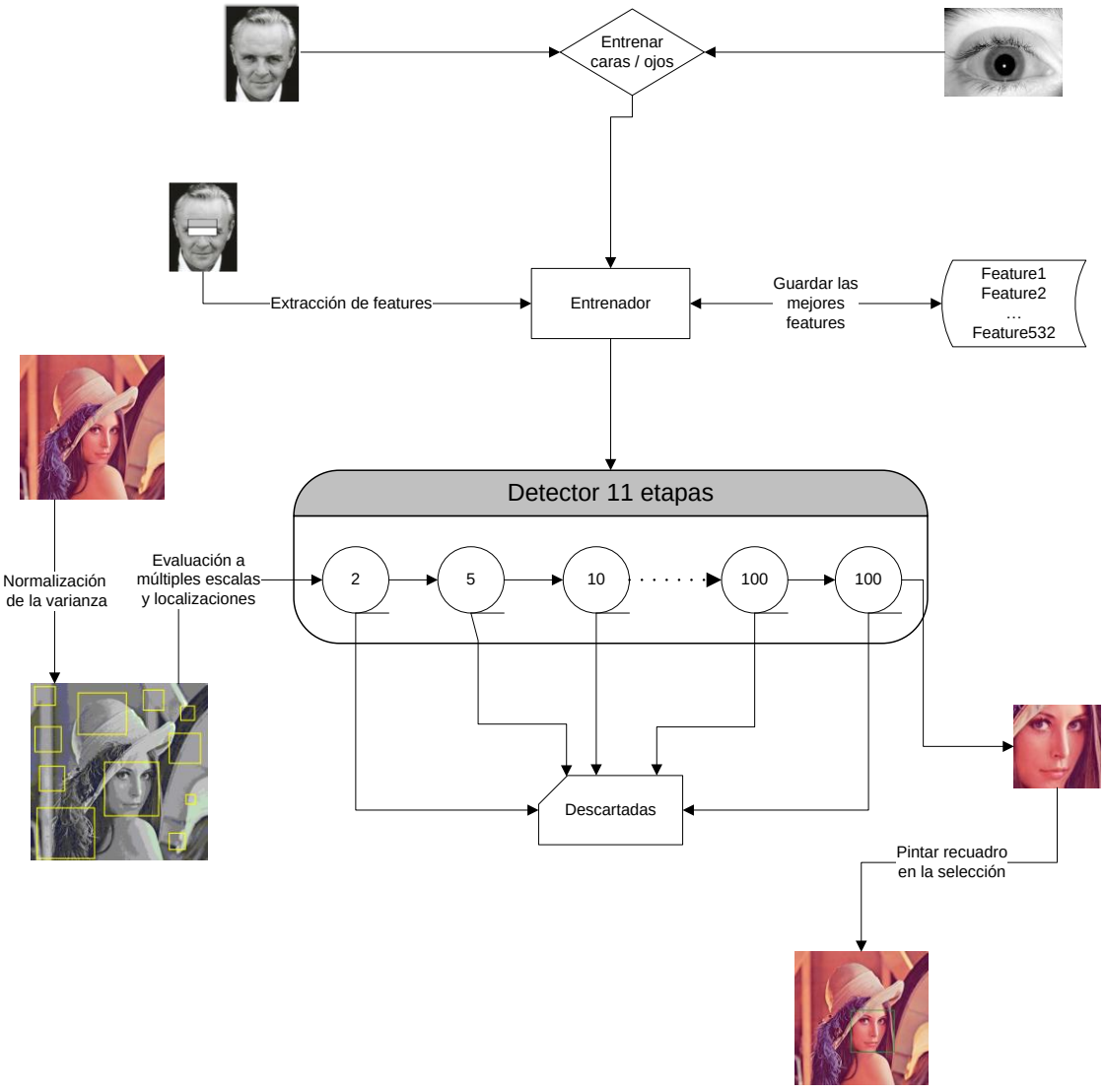


Figura 6.5 Descripción general del funcionamiento del sistema.

7

EVALUACIÓN Y RESULTADOS DEL DETECTOR

A lo largo de este capítulo se muestran los resultados obtenidos a partir de las diversas pruebas realizadas utilizando los detectores que hemos planteado. Para evaluar el funcionamiento del detector se describen los resultados obtenidos al aplicarlo sobre varias bases de datos, que se describen en el primer apartado, y se comparan con el funcionamiento del sistema OpenCV como referencia, utilizando los clasificadores genéricos OpenCV para caras y los clasificadores desarrollados por Castrillón-Santana [36] como referencia para los ojos.

7.1 Bases de datos utilizadas

A lo largo del proyecto varias han sido las bases de datos utilizadas para entrenar y para evaluar el sistema obtenido (ver Anexo IV Imágenes de ejemplo de las bases de datos) . Las imágenes utilizadas para el entrenamiento de las caras ya fueron descritas así como aquellas para los ojos:

- **Conjunto de entrenamiento de caras**: Conjunto de Viola y Jones [1] de 12788 imágenes: 4916 caras frente a 7872 no caras en escala de grises, normalizadas para minimizar el efecto de las diferentes condiciones de luminosidad.

- **Conjunto de entrenamiento de ojos**: Subconjunto de BioSecurID [35] de 12872 imágenes: 5000 ojos, 2500 derechos y 2500 izquierdos, frente a 7872 no ojos, los mismos del conjunto anterior, normalizadas para minimizar el efecto de las diferentes condiciones de luminosidad.

Junto a estos dos conjuntos hemos unido otros tres que sirven para ajustar y evaluar los resultados del detector:

- **Conjunto de imágenes Jensen**: Esta base de datos es muy similar a la base de datos utilizada para el entrenamiento. Está compuesta por 15000 imágenes en

blanco y negro, normalizadas del mismo modo que el conjunto de entrenamiento, de entre las cuales 5000 corresponden a imágenes de caras y 10000 a imágenes de no caras. Esta base de datos fue construida por Oleg Jensen como parte de su Tesis: *"Implementing the Viola-Jones Face Detection Algorithm"* [37]. Haciendo uso de estas imágenes se pudo evaluar el rendimiento del detector en imágenes del mismo tamaño y formato que las de entrenamiento. El objetivo de este trabajo es evaluar la precisión de las detecciones sin tener en cuenta las variaciones que puedan aparecer en sistemas con imágenes reales de diversos tamaños y efectos de luminosidad y poder así tener una primera aproximación. (Ver Figura 7.1)

Figura 7.1 Imágenes de la Base de datos de Jensen

- **Base de datos CMU**: La base de datos CMU de imágenes frontales es utilizada en el proyecto "*CMU Face Detection Project*" [38] de la universidad Carnegie Mellon. Esta base de datos se distribuye con el objetivo de evaluar algoritmos para la detección de imágenes faciales frontales. Está dividida en cuatro subconjuntos que han sido evaluados de manera separada: test, testlow, newtest y rotated.

- **Base de datos NIST MBGC v2.0 Face Visible Stills** [39]**:** La base de datos de NIST MBGC (National Institute of Standards and Technology Multi Biometrics Grand Challenge) está formada por 3482 imágenes frontales de caras en diferentes localizaciones y a diferente escala. En cada imagen solamente hay contenida una cara que puede encontrarse a una distancia cercana, media o lejana respecto a la cámara.

7.2 Sistemas de referencia

Para realizar un trabajo completo, además de llevar a cabo una evaluación del funcionamiento del sistema desarrollado, se ha realizado un trabajo de evaluación y comparación entre diversos sistemas de libre distribución. En el caso de la detección facial, se utilizó el sistema OpenCV con sus clasificadores genéricos; mientras que en el caso de la comparación de los ojos, se uso OpenCV con los clasificadores de Castrillón-Santana [36].

7.2.1 OpenCV

OpenCV es un conjunto de librerías de código libre de de visión artificial disponible en *http://SourceForge.net/projects/opencvlibrary*. La librería está escrita en C y C++ y funciona en Linux, Windows y Mac OS X.

OpenCV fue diseñado para conseguir una gran eficiencia computacional concentrándose activamente sobre las aplicaciones en tiempo real. OpenCV está escrito y optimizado en C y puede utilizar de manera eficiente los procesadores *multicore* [33]. Su librería posee más de 500 algoritmos optimizados (ver Figura 7.2) y se utiliza ampliamente alrededor del mundo. Es un sistema muy relevante y ampliamente utilizado ya que tiene más de 2 millones de descargas y más de 40000 personas en su grupo de usuarios [40].

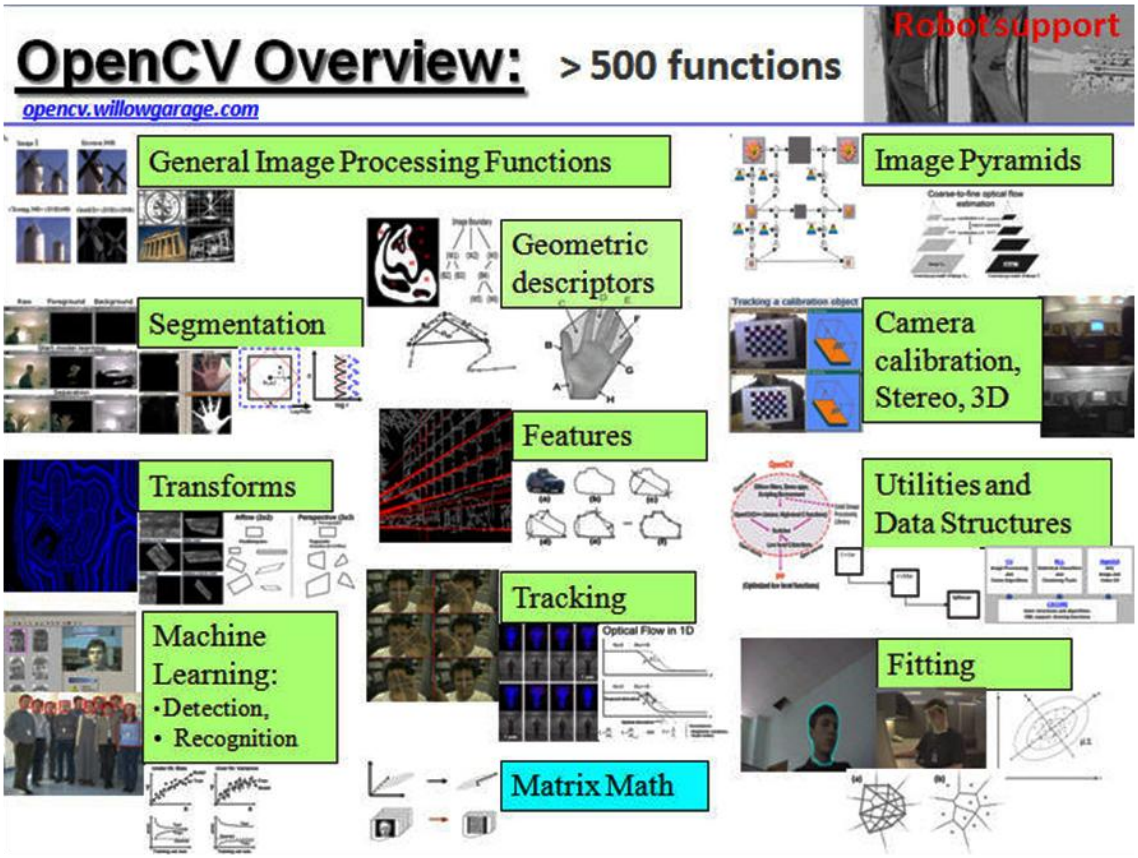


Figura 7.2 OpenCV, visión general [40].

La importancia de estas librerías se debe a que han sido ampliamente evaluadas y mejoradas por un amplio grupo de personas con lo que se ha obtenido un sistema sólido. De entre todas las funcionalidades que tiene, las que a nosotros nos interesan son las que se relacionan con la detección facial, que han sido extraídas de la versión 1.0 de OpenCV.

Los ficheros necesarios para la implementación de la detección facial son: *frontalface_default.xml* y *frontalface_alt2.xml*. Estos ficheros contienen 2 de las cascadas entrenadas para la aplicación de detección facial de OpenCV.

- ***Frontalface_default.xml***: La información del archivo se puede extraer de su propia cabecera. La cascada fue desarrollada por Rainer Lienhart utilizando el algoritmo Adaboost con imágenes de 24x24 y obteniendo 25 etapas en el detector. Por tanto este sistema tiene una filosofía de implementación similar a la propuesta en este proyecto.

- ***Frontalface_alt2.xml***: De igual modo este archivo fue también implementado por Rainer Lienhart pero utilizaba el algoritmo Gentle Adaboost, el cual minimiza la función exponencial de pérdidas de Adaboost mediante aproximaciones utilizando el método de Newton [41]. Este método se presenta como más eficiente con lo cual también nos pareció interesante evaluarlo, aunque sus imágenes de entrenamiento son de 20x20 píxeles y las etapas utilizadas son 20.

A diferencia del detector implementado para este proyecto, el presentado en OpenCV, implementado por Rainer Lienhart, utilizó un número diferente de características como se puede observar en la Figura 7.3 [16].

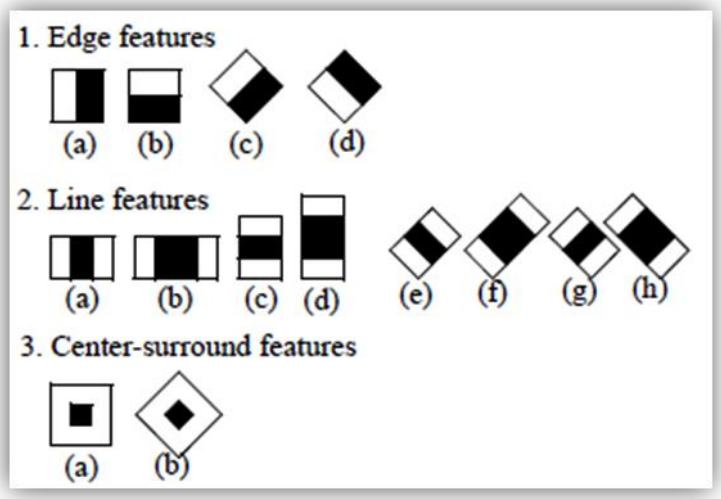


Figura 7.3 Características utilizadas por Rainer Lienhart [16].

La implementación de OpenCV utiliza 9 características, más que las planteadas en este proyecto y las presentadas por Viola y Jones, por lo que se pueden esperar resultados mucho más exactos en la detección. La mejora de los resultados se ve penalizada por un mayor tiempo de entrenamiento, sin embargo no se ve penalizada a la hora de evaluar el tiempo de detección. Esto se debe a que el número de etapas y características no varía, y el tiempo de computación de un tipo de característica frente a otra es prácticamente el mismo.

Según el autor, las características adicionales utilizadas mejoraron significativamente los resultados de la detección. El número de características utilizadas se pude observar en la Tabla 7.1:

Tabla 7.1 Número de características en una imagen de 24x24 [16].

Tipo de característica	w/h	X/Y	#
1a ; 1b	2/1 ; 1/2	12/24 ; 24/12	43.200
1c ; 1d	2/1 ; 1/2	8/8	8.464
2a ; 2c	3/1 ; 1/3	8/24 ; 24/8	27.600
2b ; 2d	4/1 ; 1/4	6/24 ; 24/6	20.736
2e ; 2g	3/1 ; 1/3	6/6	4.356
2f ; 2h	4/1 ; 1/4	4/4	3.600
3a	3/3	8/8	8.464
3b	3/3	3/3	1.521
Suma total			**225.897**

7.2.2 Conjunto de comparación de ojos

Para la comparación de las detecciones de ojos se han utilizado los clasificadores publicados por Castrillón-Santana [36]. Estos clasificadores son, según el autor, los que mostraron tener los mejores resultados. Dichos clasificadores se entrenaron con ojos izquierdos o derechos respectivamente, de tamaño 18x12 píxeles. En total se utilizaron 7000 ejemplos de entrenamiento positivos, aunque no se ha podido acceder a la información acerca del número de ejemplos negativos. Se utilizaron 18 etapas en el detector de ojos izquierdos y 16 etapas en el respectivo de ojos derechos.

Los resultados obtenidos comparando ambos detectores con los detectores de dominio público disponibles en el momento de la prueba, dan una mayor tasa de detección y en casi todos los casos una menor tasa de falsos positivos [36].

7.3 Escenarios de pruebas

Las pruebas evaluadas en este apartado tienen una función de evaluación del detector implementado, así como de comparación entre este y los demás que se encuentran desarrollados a gran escala. A partir de este trabajo se puede extraer también información acerca de la eficiencia de los otros detectores evaluados sobre dos grandes bases de datos: CMU y NIST.

7.3.1 Evaluación de la primera implementación del detector

- **Descripción del detector:** El detector se implementó utilizando archivos MEX (Anexo III Manual del programador). A partir la primera implementación se quería verificar el funcionamiento general del entrenador, por lo que el objetivo era evaluar la viabilidad de realizar el entrenamiento a una mayor escala, incluyendo más

características. Al ser este un proceso muy costoso, esta etapa es clave a la hora de continuar con la línea de trabajo planteada.

- **Metodología:** Se ha entrenado un detector de 8 etapas con 100 características distribuidas a lo largo de dichas etapas, según lo mostrado en la Tabla 7.2. Para este entrenamiento, así como para todos, se utilizó el conjunto de imágenes de Viola y Jones. La evaluación se realizó calculando las detecciones sobre el conjunto de Jensen [37] y también sobre el de Viola y Jones.

Tabla 7.2 Configuración del primer detector entrenado.

Etapa	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Nº de *weaks*	1	2	3	4	10	20	30	30

A lo largo del proyecto se comentó que el tiempo de detección se reducía considerablemente a la hora de utilizar una cascada de clasificadores frente a un detector con un solo clasificador fuerte que concentrase todas las características, ya que los falsos positivos más claros se eliminarían a lo largo de las primeras etapas de la cascada. Por tanto parece interesante evaluar el rendimiento del sistema tanto si funciona con una cascada, como con un único clasificador fuerte

- **Resultados y conclusiones:** Los resultados de estas pruebas se han dividido en 4 partes diferentes: análisis sobre las bases de Viola y Jensen respectivamente, y análisis utilizando una cascada o un solo clasificador fuerte, respectivamente.

Tabla 7.3 Resultados detector 100 *features* sobre las bases de datos de Jensen y Viola.

	Base de datos Jensen		**Base de datos de Viola**	
CASCADA	true positive rate	58,76%	true positive rate	47,57%
	false positive rate	0,02%	false positive rate	0%
	accuracy	86,24%	tasa de aciertos	79,84%
NO CASCADA	true positive rate	97%	true positive rate	95,52%
	false positive rate	2,62%	false positive rate	2,15%
	accuracy	97,25%	tasa de aciertos	96,96%

A partir de la Tabla 7.3 y la Figura 7.4 podemos observar los resultados obtenidos. Cabe destacar el rendimiento del sistema en cuanto a la tasa de detecciones. Por un lado, se observa que la tasa obtenida sobre la base de datos de Viola es menor que aquella obtenida sobre la base de datos de Jensen. Un resultado como este parece que no debería darse, sin embargo debemos contextualizar el clasificador construido a la situación en la que nos encontramos, es decir, un detector extremadamente débil que evalúa muy pocas características.

A pesar de esta debilidad inicial del detector, se obtiene una tasa de detecciones que llega hasta casi el 60% sobre la base de datos de Jensen, es decir, sobre imágenes distintas a aquellas sobre las que se entrenó.

Por otro lado, la tasa de falsos positivos es remarcablemente pequeña sobre ambas bases de datos utilizando la cascada, en torno a un 0%. Esto último nos proporciona una tasa de aciertos bastante grande, más de un 80%, para una primera prueba.

Si evaluamos los dos planteamientos de detector, cascada frente a clasificador fuerte, observamos que la tasa de detección es notablemente mayor si utilizamos un único clasificador, en torno al 95%. Sin embargo, no hay que olvidar que utilizamos extremadamente pocas características, con lo cual si a un planteamiento en el que se eliminan muchos falsos positivos en el comienzo de la cascada le unimos un detector facial débil, parece lógico deducir una gran diferencia entre ambos detectores.

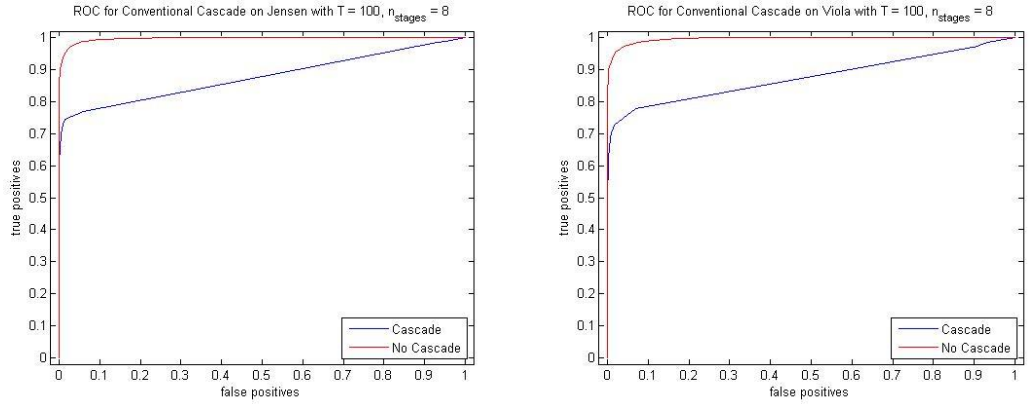


Figura 7.4 Gráfica ROC del detector de 100 *features* evaluado sobre el conjunto de imágenes de Jensen (izq.) y Viola (dcha.). En ambos casos se diferencia entre el detector en cascada y el formado por un único clasificador.

Finalmente hay que destacar que nos encontramos ante un clasificador que no sólo debe tener una buena tasa de detección sino además una baja tasa de falsos positivos, y por tanto una buena tasa de aciertos en general. No sólo es importante que el sistema acierte en el número de caras presentes, sino que además debe ser capaz de eliminar el mayor número de falsos positivos, porque no es útil un detector que al aplicarlo sobre una imagen con una única cara nos identifique un número elevado de zonas como cara. Por ello, obtener una tasa de falsos positivos tan pequeña es remarcable.

7.3.2 Evaluación del detector sobre las imágenes de Viola y Jensen.

- **Descripción del detector:** La implementación final del detector facial consta de 532 características distribuidas a lo largo de 11 etapas tal como se indicó en la Tabla 6.7. Como se comentó la cantidad de tiempo necesaria para el entrenamiento hizo obligó a reducir el número de características a implementar.

- **Metodología:** En esta primera evaluación del sistema se siguieron exactamente los mismos pasos que en el caso anterior. Con ello se pretende dar una continuidad a las pruebas y además buscar algún avance en el sistema desarrollado que nos pueda indicar

que vamos por el buen camino. El objetivo primordial es encontrar mejoras significativas con el sistema anterior.

Por tanto, la evaluación volvió a realizarse calculando las detecciones sobre el conjunto de Jensen y sobre el de Viola y Jones, así como diferenciando entre una implementación en cascada o mediante un único clasificador fuerte.

- **Resultados y conclusiones:** Los resultados de estas pruebas se pueden observar tanto en la Tabla 7.4 como en la Figura 7.5. Para una mayor claridad a la hora de comparar los resultados y apreciar las mejoras en la Tabla 7.5 se muestran los resultados de ambos detectores.

Tabla 7.4 Resultados detector 532 *features* sobre las bases de datos de Jensen y Viola.

	Base de datos Jensen		Base de datos de Viola	
CASCADA	true positive rate	87,42%	true positive rate	89,79%
	false positive rate	1,23%	false positive rate	0,170%
	accuracy	94,99%	tasa de aciertos	95,97%
NO CASCADA	true positive rate	98,52%	true positive rate	97,27%
	false positive rate	2,24%	false positive rate	0,53%
	accuracy	98,01%	tasa de aciertos	98,62%

Es remarcable la mejora obtenida en todos los aspectos. En referencia al desarrollo implementado en cascada, la mejora de la tasa de detección llega al 30% colocándose en resultados competitivos frente a los detectores que se pueden encontrar implementados. Tasas en torno al 90% de detección nos permitieron deducir que la implementación del sistema se había hecho de manera correcta y que además el sistema tendría un gran potencial a la hora de realizarse un entrenamiento con un mayor número de características que pudiese llevar a un sistema aún más competitivo. Esto también nos hizo seguir hacia el siguiente paso de la implementación, el escalado del detector, que ya se explicó en el apartado anterior.

Tabla 7.5 Comparación entre los detectores de 100 y 532 *features*.

		Detector 100 *features*		Detector 532 *features*	
		Jensen	Viola	Jensen	Viola
CASCADA	true positive rate	58,76%	47,57%	87,42%	89,79%
	false positive rate	0,02%	0%	1,23%	0,170%
	accuracy	86,24%	79,84%	94,99%	95,97%
NO CASCADA	true positive rate	97%	95,52%	98,52%	97,27%
	false positive rate	2,62%	2,15%	2,24%	0,53%
	accuracy	97,25%	96,96%	98,01%	98,62%

Por otro lado, los resultados mostrados sobre este segundo detector implementado parecen tener una mayor lógica, desde el punto de vista de que el sistema ofrece un mejor rendimiento al aplicarse sobre el conjunto de entrenamiento que sobre el de test de Jensen.

Si nos fijamos en la tasa de falsos positivos observamos claramente que la mejora del sistema en cuanto a detección no implica un empeoramiento significativo de los falsos positivos. Con esto mantenemos nuestro compromiso de intentar obtener una alta detección sin descuidar los molestos falsos positivos.

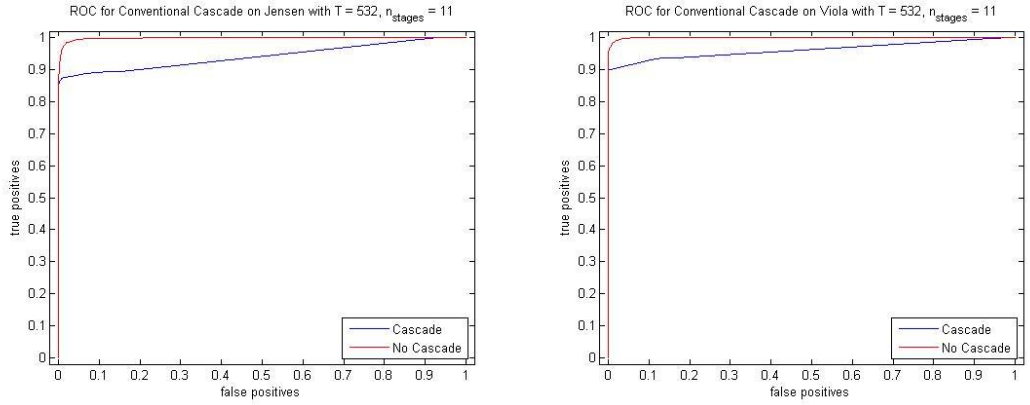


Figura 7.5 Gráfica ROC del detector de 532 *features* evaluado sobre el conjunto de imágenes de Jensen (izq.) y Viola (dcha.). En ambos casos se diferencia entre el detector en cascada y el formado por un único clasificador.

Por último cabe destacar que a pesar de que el sistema desarrollado como un único clasificador fuerte ofrece mejores resultados, el sistema desarrollado en cascada acerca bastante su rendimiento. Parece lógico deducir que un sistema con un número de características más competitivo ofrecerá mejores resultados.

7.3.3 Evaluación del detector de caras sobre las imágenes CMU

- **Descripción:** Como ya se ha descrito en apartados anteriores, la base de datos de imágenes de CMU está dividida en 4 sub-categorías: *newtest, test, testlow* y *rotated*, formadas por imágenes en blanco y negro.

- **Metodología:** Las imágenes de esta base de datos contienen una o varias imágenes faciales. Cada una de las sub-categorias que forman la base de datos se evaluó de manera separada.

- **Resultados y conclusiones:** Esta fue la primera prueba realizada sobre un conjunto de imágenes reales, de diferentes tipos, en las que se pueden encontrar diferentes números de caras por imagen y a diferente escala. Por tanto los resultados obtenidos fueron claves a la hora de evaluar el rendimiento del detector así como para ajustar sus parámetros.

Por cada imagen se realizó una evaluación como la mostrada en la Tabla 7.6.

Tabla 7.6 Método de clasificación del funcionamiento de los detectores sobre CMU

Imagen	Nº caras presentes	Aciertos	No detectadas	Falsos positivos	Porcentaje aciertos	Porcentaje fallos
addams-family.jpg	7	7	0	4	100	0
aeon1a.jpg	1	0	1	4	0	100
audrey1.jpg	1	1	0	3	100	0
...	...	...	...	...	...	...

Los resultados obtenidos se describen para cada una de las sub-categorías en la Tabla 7.7. En la primera fila se muestra el rendimiento del sistema a través del porcentaje de detecciones, teniendo en cuenta el número de caras que aparecen por cada imagen y el número de detecciones de áreas faciales. La segunda fila corresponde a la relación entre los falsos positivos obtenidos y el número de caras presentes. Con esto se busca mostrar una idea acerca lo molestos que serían los falsos positivos en sistemas comerciales como por ejemplo en una cámara de fotos, donde el usuario no debe observar simplemente las detecciones correctas sino que no desea ver la pantalla de su aparato lleno de identificaciones de falsos positivos. A partir de los resultados mostrados en la tabla podemos deducir que sobre la base de datos CMU se obtiene una relación máxima de alrededor de 1 falso positivo por cada cara presente.

Tabla 7.7 Resultados detector de 532 *features* sobre la base CMU.

Categoría	NEWTEST	TEST	TESTLOW	ROTATED
Detection rate	85,6%	80,4%	91,6%	…
False positives/ number of faces	1,26	0,67	0,61	…

La última columna de la Tabla 7.7, hace referencia a los resultados obtenidos sobre la categoría de caras rotadas. En este caso las detecciones son prácticamente inexistentes, razón por la cual los resultados no han sido introducidos. Si nos paramos a evaluar este resultado veremos que tiene lógica, ya que el sistema fue entrenado con caras frontales rectas y no rotadas. Para detectar las caras rotadas, simplemente habría que entrenar el detector con este otro tipo de imágenes y además con características inclinadas. Parece lógico inferir que el resultado sería satisfactorio viendo el funcionamiento general del detector.

A partir de la Figura 7.6 podemos observar las distintas etapas del detector y su rendimiento final en una imagen de ejemplo típica extraída de la base CMU. Cabe mencionar que las dos fotografías que muestran las detecciones han sido además post-procesadas mediante *Photoshop* para obtener una mayor claridad de colores a la hora de imprimir la memoria. Sin embargo, la imagen original no ha sido modificada para reflejar exactamente el tipo de imágenes con el que se trabajó.

Figura 7.6 Descripción del funcionamiento del detector. Imagen original (arriba izq.), imagen con todas las detecciones (arriba a la drcha.) e imagen final post-procesada (abajo).

Gracias a la Figura 7.6 observamos la necesidad de realizar el post-procesamiento de la imagen después de que se realizasen las detecciones. Una imagen, como la mostrada con todas las detecciones marcadas en color rojo, nos ofrece un resultado casi tan inútil como una imagen sin ninguna detección y es más que desaconsejable para multitud de aplicaciones. Sin embargo, la imagen post-procesada, a pesar de que pueda tener algún fallo, muestra una detección más que satisfactoria.

Por último, se debe remarcar que aquellas imágenes en las que hay un fondo homogéneo ofrecen una tasa de falsos positivos menor que aquellas imágenes en las que el fondo pueda ofrecer más variaciones. Parece lógico que este sea el comportamiento que se observa, y por tanto, en una aplicación en la que el fondo se pueda establecer por parte del usuario del sistema se recomienda utilizar un entorno controlado, con un fondo homogéneo que permita el mejor funcionamiento del detector. Véase por ejemplo, los controles de acceso a edificios o aeropuertos en los que se registre la entrada de personas o incluso se permita esta entrada una vez comprobada su identidad de manera automática mediante un sistema de reconocimiento facial.

7.3.4 Evaluación OpenCV detectando caras sobre las imágenes CMU

➢ **Descripción:** OpenCV es el sistema de referencia utilizado por la mayoría de los desarrolladores. Para esta primera prueba comparativa se utilizó la cascada de clasificadores *frontalface_alt2.xml* que ya fue explicada en el apartado 7.2.

➢ **Metodología:** Para obtener estos resultados se evaluó el clasificador indicado sobre la base de CMU de manera separada sobre cada uno de los subconjuntos que la forman.

➢ **Resultados y conclusiones:** Los resultados del rendimiento del sistema anteriormente mencionado se pueden observar en la Tabla 7.8.

Tabla 7.8 Resultados de OpenCV sobre la base CMU utilizando la cascada *frontalface_alt2*.

Subset	Detection performed
newtest	89.07%
test	86.98%
rotated	19.28%
test-low	83.56%

Según podemos observar el sistema comercial funciona ligeramente mejor que el desarrollado para este proyecto. La diferencia es de aprox. un 5% de media en los subconjuntos *Newtest* y *Test*. Además, sobre el subconjunto *Test-low,* el sistema implementado funciona mejor que OpenCV.

A partir de estos resultados vemos que el rendimiento sobre el subconjunto *Rotated* tampoco es eficiente en OpenCV. Estos resultados son los esperados, ya que el sistema de Viola y Jones no es capaz de aceptar más que ligeras variaciones en rotación con respecto a los ejemplos positivos de entrenamiento utilizados [36]. La Figura 7.7 muestra los resultados de una misma imagen sobre ambos clasificadores, el implementado en el proyecto y el OpenCV.

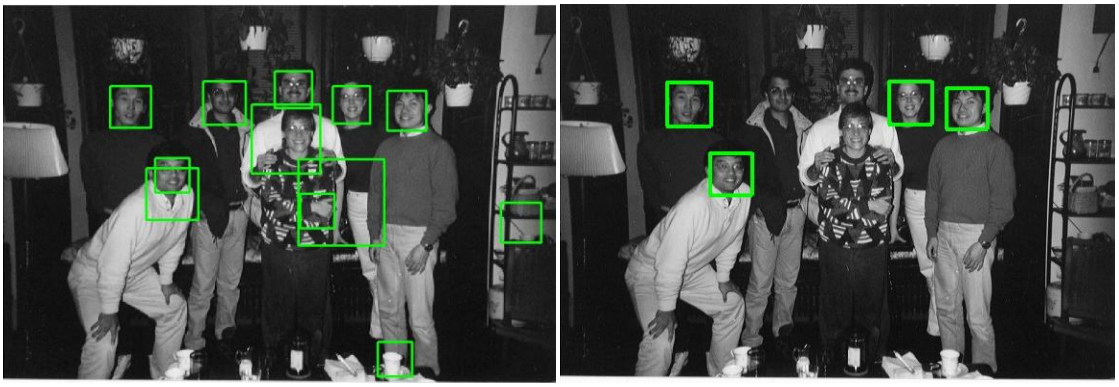


Figura 7.7 Detecciones utilizando el sistema desarrollado (izq.) frente a OpenCV (dcha.)

7.3.5 Evaluación del detector sobre las imágenes NIST-MBGC

➢ **Descripción:** La base de datos de NIST-MBGC, está formada por 3484 imágenes a color de una sola cara, de diferentes tamaños, en distintos lugares y a diferentes escalas. Las imágenes fueron evaluadas una a una identificando tanto las detecciones como los falsos positivos.

➢ **Metodología:** Las imágenes fueron transformadas a escala de grises para poder ser evaluadas de manera adecuada por parte del detector. Los ejemplos expuestos en el siguiente apartado se muestran sobre imágenes en escala de grises. La evaluación del sistema se realizó siguiendo la misma metodología empleada para la base de datos CMU, ya explicada y mostrada en la Tabla 7.6.

➢ **Resultados y conclusiones:** Los resultados obtenidos en este caso se muestran en la Tabla 7.9. Como se puede observar no son tan buenos como los desarrollados sobre la base de datos CMU, sin embargo la complicación que presentan dichas imágenes es también considerablemente mayor.

El rendimiento obtenido refleja una tasa de detección del 33,21% y una relación entre el número de falsos positivos y el de imágenes de 1,7. Estos resultados muestran una mala tasa de detección, sin embargo los falsos positivos no se han visto disparados excesivamente. Estos resultados han de encuadrarse en la situación de un detector con un menor número de características y entrenado con menos tipos de *features* que un detector competitivo del mercado. Por otro lado, el tamaño de las imágenes puede ser también un problema a la hora de escalar el detector y mantener las proporciones.

Tabla 7.9 Evaluación del sistema desarrollado sobre las imágenes de NIST-MBGC.

Número de imágenes evaluadas	**3482**	**500**
Detection rate	33,21%	34,2%
False positives/ number of faces	1,7	1,51

Debido al costoso trabajo de evaluación de un número tan grande de imágenes el resto de los detectores fueron evaluados sobre las 500 primeras imágenes del conjunto NIST. Es por ello que para poder comparar los detectores también se han extraído los datos del rendimiento del sistema actual analizando esas 500 imágenes. Como es lógico, se puede ver que los resultados son similares en términos porcentuales.

Figura 7.8 Imagen original del conjunto NIST (izq.) y la detección realizada por el sistema (dcha .)

A partir de la observación de los resultados se pudieron extraer algunas cosas sobre el funcionamiento del detector. En la imagen superior vemos que la detección facial llevada a cabo tiene forma cuadrada. Esto se debe a que las imágenes utilizadas para el entrenamiento eran cuadradas de 24x24 píxeles y las características así como el escalado se realizan sobre una base cuadrada, con lo cual las sub-ventanas que se calculan y pueden ser identificadas como caras también tienen esa forma. Por tanto las imágenes de caras muy grandes, que ocupan la mayor parte de la imagen no han sido detectadas ya que tienen una forma rectangular.

Se ha observado que las imágenes en las que la cara se encuentra en un primer plano propician una mejor detección que aquellas imágenes en las que la imagen se encuentra más alejada (ver Figura 7.9). En la imagen también podemos observar el efecto del fondo de la imagen sobre los falsos positivos. También tienen una influencia destacable las camisas y la ropa en general y la iluminación. A pesar de normalizar respecto a la desviación estándar, la iluminación sigue siendo un problema.

Figura 7.9 Diferencia entre las detecciones a un individuo en segundo plano (izq.) o en primer plano (dcha.)

7.3.6 Evaluación de OpenCV sobre NIST-MBGC

- **Descripción:** Se han evaluado las 500 primeras imágenes de la base de datos NIST-MBGC. Con esto se pretendió utilizar las imágenes de varios usuarios en todas las localizaciones a las que se sometió a cada uno de ellos y poder así tener una visión general del detector, al menos en lo que a diferentes localizaciones se refiere.

- **Metodología:** La metodología utilizada es similar a la anterior, con el objetivo de mantener la coherencia. Se han evaluado dos cascadas: *frontalface_default* y *frontalface_alt2*.

Se eligieron estas cascadas por similitud con la desarrollada en este proyecto, como es el caso de la primera que utiliza imágenes del mismo tamaño y un algoritmo de entrenamiento idéntico; y por continuidad con el trabajo de evaluación anterior sobre CMU, como es el caso de la segunda cascada que además es la presentada como más eficiente y de mejor rendimiento.

➢ **Resultados y conclusiones:** El resultado obtenido se muestra en la Tabla 7.10, donde se adjuntan también los resultados desarrollados obtenidos en el punto anterior sobre esas mismas 500 imágenes.

Tabla 7.10 Resultados del sistema implementado y de OpenCV sobre NIST-MBGC.

Detector	Sistema desarrollado	frontalface_default	frontalface_alt2
Detection rate	34,2%	95%	91,2%
False positives/ number of faces	1,51	3,1	3,53

A partir de los datos mostrados se puede observar que los resultados de OpenCV ofrecen una mayor tasa de detección. Esto es lógico y comprensible ya que este sistema fue entrenado con un número mucho mayor de características a lo largo de más de etapas. Sin embargo, a pesar de esta buena tasa de detección la tasa de falsos positivos es significativamente mayor. El sistema muestra una gran cantidad de detecciones pequeñas identificadas como caras en una misma imagen.

Figura 7.10 Ejemplo de detección con múltiples falsos positivos usando *frontalface_default.*

En la Figura 7.10 se puede observar como al evaluar una imagen sobre un fondo homogéneo y con ropa con un color aparentemente homogéneo también, se producen 9 falsos positivos. A pesar de que el sistema haya detectado la cara correctamente, en algunas aplicaciones, la aparición de tantos falsos positivos implicaría problemas significativos.

Resultados muy similares se obtuvieron con ambas cascadas, con lo cual no vamos a comentarlas por separado más allá de lo ya hecho.

Sin embargo cabe destacar, que según las pruebas aplicadas, la cascada *frontaface_default.xml*, que se suponía más débil, ha dado mejores resultados de detección.

7.3.7 Evaluación del detector de ojos sobre CMU

➢ **Descripción:** El detector de ojos fue entrenado con imágenes de 24x24 píxeles a partir de la base de datos BioSecurID como ya se indicó. Esta base de datos mantuvo los mismos ajustes que el sistema anterior de detección de caras. El objetivo de esta continuidad es evaluar también el rendimiento obtenido al aplicar estos ajustes.

➢ **Metodología:** El sistema de evaluación ha sido idéntico al seguido para la estimación del rendimiento del detector de caras sobre esta base de datos.

➢ **Resultados y conclusiones:** Los resultados obtenidos se muestran en la Tabla 7.11 y mediante un ejemplo en la . Se puede observar que son claramente peores que aquellos obtenidos para caras sobre esta misma base de datos. Sin embargo no hay que olvidar que estos resultados han de contextualizarse dentro del sistema implementado con un número de características mejorable. Es posible además que el tipo de características no sea el adecuado para la detección de ojos. Las *features* utilizadas son adecuadas a las caras, pero no están orientadas en un principio a la detección de ojos. Quizás el uso de las características rotadas pudiese ser mejor para la adaptación a la forma de los ojos, o simplemente serían complementarias, mejorando significativamente la detección.

Tabla 7.11 Rendimiento del detector de ojos desarrollado sobre la base de datos de CMU

Categoría	NEWTEST	TEST	TESTLOW	ROTATED
Detection rate	16,4%	11%	8%	21,1%
False positives/ number of eyes	0,3	0,2	0,3	0,3

Por otro lado, al ser los ojos elementos de pequeño tamaño, se hace aún más problemático detectarlos en imágenes en las que los individuos se encuentran alejados respecto de la cámara. En imágenes donde los individuos se encuentran cerca del sensor la tasa de detección es mayor.

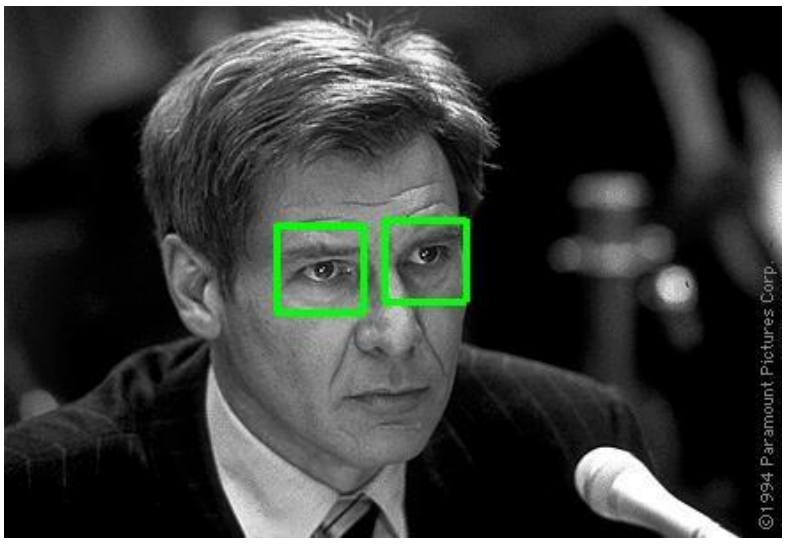


Figura 7.11 Ejemplo de detección ojos del sistema implementado sobre CMU.

Finalmente hay que remarcar que el detector fue entrenado con ojos de 24x24 píxeles aprovechando así la estructura que se poseía para el entrenamiento de los clasificadores de caras. Estos ojos fueron redimensionados a este tamaño y por tanto no mantienen la proporción con un ojo real, el cual no tiene forma cuadrada sino rectangular.

7.3.8 Evaluación de OpenCV para ojos sobre CMU

➢ **Descripción:** El detector de ojos fue entrenado con imágenes de 18x12 píxeles a partir de la base de datos BioSecurID [35] como ya se indicó. Se utilizó lo descrito por Castrillón-Santana [36] donde se indica que el clasificador entrenado con ojos derechos de ese tamaño (u ojos izquierdos) ofrece el mejor rendimiento.

➢ **Metodología:** Los datos de evaluación del sistema sobre la base de imágenes CMU han sido extraídos de la publicación del autor referenciando el rendimiento de su sistema [36].

➢ **Resultados y conclusiones:** Los datos del rendimiento del sistema se pueden observar en la Tabla 7.12.

Tabla 7.12 Detección del ojo izquierdo sobre cada subconjunto.

Subset	Detection performed
newtest	39,89%
test	29,69%
rotated	32,29%
test-low	10,96%

Observando los resultados vemos que no son mucho mejores que los obtenidos por nuestro detector, aunque si algo mejores. Esto puede ser achacado a un entrenamiento más exhaustivo por parte del autor de este detector. Según el mismo, las detecciones utilizando las imágenes de ojos derechos para entrenar el detector, proporcionarían resultados similares en cuanto al rendimiento.

7.3.9 Evaluación del detector de ojos desarrollado sobre NIST-MBGC

➢ **Descripción:** Se evalúa el sistema desarrollado entrenando con imágenes de ojos de 24x24 píxeles sobre la base de NIST-MBGC para completar el ciclo de pruebas.

➢ **Metodología:** El método seguido fue el mismo que en el caso de las evaluaciones anteriores, analizando cada una de las 500 imágenes de manera independiente y extrayendo el número de detecciones y de falsos positivos.

➢ **Resultados y conclusiones:** Los datos del rendimiento del sistema desarrollado son:

* Tasa de detección de 28,1%.
* Relación falsos positivos/número de ojos presentes de 1.97.

Estos resultados parecen lógicos y coherentes con aquellos obtenidos sobre la base de datos de CMU y algo menores que aquellos resultados calculados a partir de las detecciones del clasificador facial. En la Figura 7.12 se puede observar un ejemplo de imagen de NIST-MBGC sobre la que se llevó a cabo la detección.

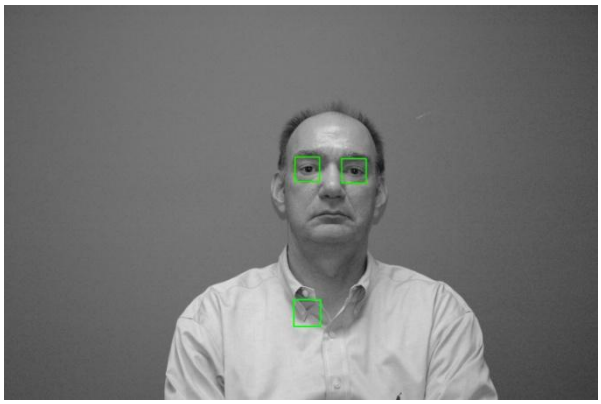


Figura 7.12 Resultado detector de ojos desarrollado sobre NIST.

A pesar de tener una tasa de falsos positivos mayor de lo que se obtuvo en la detección facial, seguimos manteniendo una relación razonable referente a este tema.

7.3.10 Evaluación detector de Castrillón-Santana sobre NIST-MBGC

- **Descripción:** Finalmente se evalúa el sistema de Castrillón-Santana ya comentado sobre CMU, pero en este caso calculando el rendimiento sobre NIST.

- **Metodología:** El método seguido fue el mismo que en el caso de las evaluaciones anteriores.

- **Resultados y conclusiones:** En la Tabla 7.13 se muestran los resultados tanto de este detector como del desarrollado sobre este conjunto de imágenes.

Tabla 7.13 Resultados del sistema implementado y del sistema de Castrillón-Santana sobre NIST.

Detector	Sistema desarrollado	Castrillon-Santana
Detection rate	28,1%	77,8%
False positives/ number of eyes	1,97	5,8

Comparativamente el detector de Castrillón-Santana tiene una mejor detección sobre NIST-MBGC, superando ampliamente el rendimiento ofrecido sobre CMU. Sin embargo la tasa de falsos positivos es elevadísima. Claramente se está ofreciendo una tasa extremadamente alta, sobre todo si tenemos en cuenta que el dato mostrado se estudió referenciando el número de falsos positivos sobre el número de ojos. Por tanto, en una imagen con un individuo, y por tanto 2 ojos, obtendríamos aprox. 11 falsos positivos ($5{,}8 \times 2 \approx 11$) además de las áreas correctamente detectadas (ver Figura 7.13).

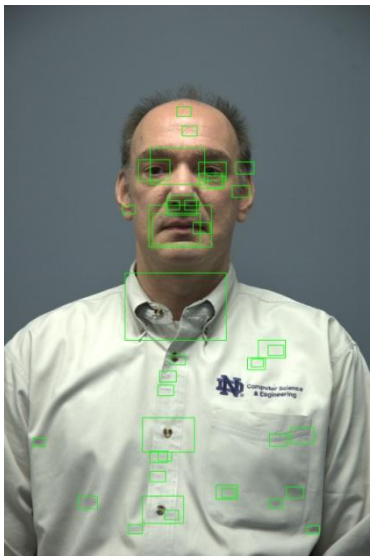


Figura 7.13 Ejemplo de alta tasa de falsos positivos obtenidos con el detector de Castrillón-Santana sobre NIST

8

CONCLUSIONES Y TRABAJO FUTURO

A lo largo del trabajo presentado se ha estudiado, desarrollado, implementado y documentado un sistema de detección facial y un sistema de detección de ojos siguiendo técnicas del estado del arte. También se ha recopilado información acerca de la implementación utilizando archivos MEX.

Como se comentó en el prólogo del proyecto, el objetivo de este proyecto es el estudio e implementación efectiva de un detector de caras y de un detector de ojos. A partir de esta implementación se pretendía obtener una evaluación sólida del rendimiento del sistema implementado.

En primer lugar, se implementó el sistema de entrenamiento facial exclusivamente sobre Matlab. En segundo lugar, para aumentar la velocidad del sistema se estudió el método de implementación utilizando archivos MEX y se procedió a realizar este desarrollo. El sistema se entrenó con las mismas caras que utilizaron Viola y Jones con el objetivo de mantener una similitud con el estado del arte. Sin embargo, el entrenamiento sigue siendo remarcablemente lento, habiendo entrenado una cascada de 532 características y 11 etapas en alrededor de una semana. Debido a este espacio temporal no se consideró adecuado para los objetivos de este proyecto entrenar un detector más fuerte sino evaluar el rendimiento del que se ha implementado y analizar de este modo la posibilidad de entrenar un detector más fuerte infiriendo los resultados que se obtendrían. Acto seguido el algoritmo se utilizó para entrenar un detector de ojos tal y como se ha explicado.

Una vez entrenado el sistema se procedió a evaluarlo sobre los distintos conjuntos de imágenes disponibles: conjunto de Viola y Jones, conjunto de Jensen, conjunto de CMU y conjunto NIST-MBGC. Los resultados obtenidos fueron explicados en el capítulo 7.

Analizando los resultados obtenidos podemos extraer que el sistema implementado funciona mejor a la hora de evitar la producción de falsos positivos frente

a los sistemas sólidos analizados a modo de comparación. Esto nos lleva a abrir dos frentes de pensamiento:

I. Es normal que el sistema desarrollado no tenga un rendimiento tan alto como aquellos sistemas con los que se ha comparado. Esto se debe a que el sistema no se ha entrenado tanto como los otros. Obtener un rendimiento más competitivo sería simplemente una cuestión de tiempo, mediante un entrenamiento más prolongado del algoritmo.

II. El hecho de que el sistema ofrezca una menor tasa de detección y a la vez una menor tasa de falsos positivos, podría dar a entender que relajando las restricciones a la hora de identificar los falsos positivos, obtendríamos una mayor tasa de detección con tasas de falsos positivos similares a los sistemas implementados.

Por otro lado, obtener unos resultados competitivos sobre la base de datos CMU, similares al OpenCV, y resultados aceptables en comparación al sistema de Castrillón-Santana detectando ojos, nos indican que el desarrollo realizado ha sido un éxito y se abren por tanto nuevos frentes de actuación futura.

El primer paso consistiría en entrenar el detector con alrededor de 4000 características, siguiendo por ejemplo el modelo ya indicado de Ming-Hsuan Yang [34]. El objetivo de esto sería evaluar el algoritmo implementado después de un entrenamiento competitivo. Calculo que el tiempo necesario para ello sería de alrededor de 2 meses. Este entrenamiento puede hacerse de manera más rápida mediante una programación orientada a aprovechar de manera eficiente los procesadores *multi-core*.

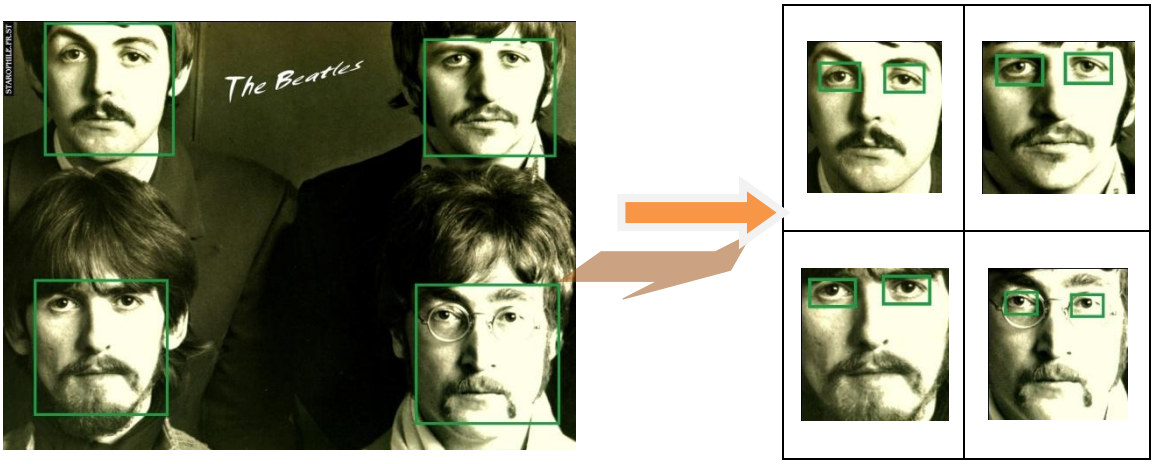


Figura 8.1 Funcionamiento del sistema futuro propuesto para detectar caras y ojos

Una vez obtenido un detector competitivo de caras, otra línea de trabajo sería aplicar el mismo algoritmo para entrenar un detector de ojos. Podrían utilizarse imágenes cuadradas o imágenes rectangulares como las utilizadas por Castrillón-Santana. Con el fin de mejorar la detección se propone utilizar ambos detectores de manera consecutiva tal y como se muestra en la Figura 8.1.

A partir de la identificación de un área como cara se procedería a evaluar mediante el detector de ojos dicha área. Con esto, si el sistema identifica alguna zona como ojo, el área seleccionada sería definitivamente catalogada como cara. En caso contrario sería desechada. El objetivo de este sistema sería disminuir el número de falsos positivos tanto del detector de caras como del detector de ojos. Por un lado, para catalogar un área como cara haría falta utilizar dos detectores, y por otro lado, una el detector de ojos utilizaría imágenes reducidas, ya que en vez de analizar la totalidad de la imagen analizaría zonas menores. Con esto, queda considerablemente reducido el tiempo de trabajo del detector de ojos, por lo que su introducción como parte de un sistema completo no penalizaría de manera significativa el rendimiento en términos temporales.

Una vez evaluado el rendimiento de la técnica descrita anteriormente, la culminación del sistema consistiría en introducir un sistema de reconocimiento facial al final de los bloques anteriores. Con esto se cerraría el ciclo del sistema sin realizar un trabajo excesivamente costoso ya que los detectores competitivos se conseguirían simplemente empleando más tiempo de entrenamiento, pero sin necesidad de ninguna implementación alternativa. En todo caso podrían utilizarse imágenes de otro tamaño para el entrenamiento del detector, pero la creación de esas bases de datos no conllevaría excesivo trabajo y siempre pueden utilizarse las que se entregan. Es por ello que las líneas de investigación abiertas por este proyecto son múltiples y muy interesantes, sin poner un punto final en el trabajo sino un punto y seguido lleno de oportunidades.

9

BIBLIOGRAFÍA

1. P. Viola, M. J. Jones, "Robust Real-Time Face Detection", *International Journal of Computer Vision*, Vol. 57, pp. 137-154, 2004.

2. A. K. Jain, A. Ross and S. Pankanti, "Biometrics: A Tool for Information Security", *IEEE Transactions on information forensics and security*, Vol. 1, pp. 125-143, June 2006.

3. A. K. Jain, A. Ross, S. Prabhakar, "An Introduction to Biometric Recognition" *IEEE Transactions on Circuits and Systems for video Technology*, Vol. 14, pp. 4-20, January 2004

4. A. Ross, K. Nandakumar, Anil K. Jain. *Handbook of Multibiometrics.* ed.: Springer, 2006. ISBN 0387222960.

5. D. D. Zhang, *Automated biometrics: technologies and systems* . ed.: Kluwer Academic Publichers, 2000. ISBN 0792378563.

6. Y. Zhu, T. Tan, Y. Wang, "Biometric Personal Identification Based on Handwriting", *International Conference on Pattern Recognition*, Vol. 2, 2000.

7. D. Maio, D. Maltoni, R. Cappelli, J. L. Wayman, A. K. Jain. "FVC2002: Fingerprint verification competition", *Proc. Int. Conf. Pattern Recognition (ICPR)*, pp. 744-747, Agosto 2002.

8. J. Galbally, J. Fierrez and a. J. Ortega-Garcia, "Vulnerabilities in biometric systems: attacks and recent advances in liveness detection", in *Proc. Spanish Workshop on Biometrics, SWB*, Girona, Spain, June 2007.

9. N. Ratha, J. Connell, R. Bolle, "An analysis of minutiae matching strength", *Proc. AVBPA, International Conference on Audio- and Video-Based Biometric Person Authentication III*, pp. 223-228, 2001.

10. A. Ross, A. K. Jain, "Information fusion in biometrics", *Pattern Recognition Letters*, Vol. 24, pp. 2115—2125, 2003

11. U. M. Bubeck, "Multibiometric Authentication - An Overview of Recent Developments" *Term Project CS574, San Diego State University*, Spring 2003.

12. L.-F. Chen, H.-Y.M. Liao, J.-C. Lin, C.-C. Han, "Why Recognition in a Statistics-based Face Recognition System Should be based on the Pure Face Portion: a Probabilistic Decision-based Proof" *Pattern Recognition*, Vol.34, No.5, pp. 1393-1403, 2001.

13. G. Yang, T.S. Huang, "Human Face Detection in a Complex Background" *Pattern Recognition*, pp. 53-63, Vol. 27, 1994.

14. J. Ohya, R. Nakatsu, J. Tang, S. Kawato, "Locating human faces in a complex background including non-face skin colors", *International Conference on Imaging Science, Systems, and Technology*, Vol. 12, 2000.

15. P. Peer, F. Solina, "An Automatic Human Face Detection Method", *Proceedings of Computer Vision Winter Workshop, Ed. N. Brändle*, pp. 122-130, 1999.

16. R. Lienhart, J. Maydt, "An Extended Set of Haar-Like Features for Rapid Object Detection", *IEEE ICIP*, pp. 900-903, 2002.

17. E. Saber, A. Murat Tekalp, "Frontal-View Face Detection and Facial Feature Extraction using Color, Shape and Symmetry Based Cost Functions". 1998.

18. A. L. Yuille, P. W. Hallinan, D. S. Cohen, "Feature extraction from faces using deformable templates", *International Journal of Computer Vision*, Vol. 8, pp. 99-111, 1992.

19. P. Sinha, "Object Recognition via Image Invariants: A case Study", *Investigative Ophthalmology and Visual Science*, Vol. 35, pp. 1735-1740. 1994.

20. T. V. Pham, M. Worring, A. W. M. Smeulders, "Face detection by aggregated Bayesian network classifiers", *Pattern Recognition Letters*, Vol. 23, págs. 451-461, 2001.

21. H. A. Rowley, S. Baluja, T. Kanade, "Neural Network-Based Face Detection" *IEEE Transactions On Pattern Analysis and Machine Intelligence*, Vol. 20, pp. 23-38, 1996.

22. D. Roth, M.-H. Yang, N. Ahuja. "A SNoW-Based Face Detector", *MIT Press*, pp. 855-861, 2000.

23. H. Schneiderman, T. Kanade, "A Statistical Model for 3D Object Detection Applied to Faces and Cars", *IEEE Conference on Computer Vision and Pattern Recognition*, June 2000.

24. M.-H. Yang, N. Ahuja, "Detecting Faces in Images: A Survey", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, 2002.

25. Y. Freund, R. Schapire, "Experiments with a new boosting algorithm" *International Conference on Machine Learning*, pp. 148-156, 1996.

26. C. Garcia, G. Tziritas, "Face Detection Using Quantized Skin Color Regions Merging and Wavelet Packet Analysis", *IEEE Transactions on Multimedia*, Vol. 1, pp. 264-277, 1999.

27. B. Menser, F. Muller, "Face detection in color images using principal components analysis", *Seventh Int'l Conf. on Image Processing and Its Applications*, pp. 620-624, 1999.

28. P. Viola, M. Jones, "Rapid Object Detection Using a Boosted Cascade of Simple Features", *IEEE Proc. Computer Vision and Pattern Recognition*, pp. 511-518, 2001.

29. C.P. Papageorgiou, M. Oren, T. Poggio, "A general framework for object detection", *International Conference on Computer Vision*, 1998.

30. Y. Freund, R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to Boosting" *Computational Learning Theory: Eurocolt 95*, ed. Springer- Verlag, pp. 23-37, 1995.

31. F. Rosenblatt, "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain" *Psychological Review*, Vol. 65, pp. 386-408, 1958.

32. R. Rajalingham, "Object Detection Using Feature Selection and a Classifier Cascade" *The Delta-Epsilon, McGill Undergraduate Mathematics Magazine.*

33. G. Bradski, A. Kaehler, *"Learning OpenCV: Computer Vision with the OpenCV Library"*, Cambridge : O'REILLY, 2008, ISBN 978-0-596-51613-0.

34. M.-H. Yang, "Recent Advances in Face Detection". *Cambridge : IEEE ICPR 2004 Tutorial*, 2004.

35. J. Fierrez, J. Galbally, J. Ortega-Garcia, M. R. Freire, F. Alonso-Fernandez, D. Ramos, D. T. Toledano, J. Gonzalez-Rodriguez, J. A. Siguenza, J. Garrido-Salas, E. Anguiano, G. Gonzalez-de-Rivera, R. Ribalda, M. Faundez-Zanuy, J. A. Ortega, V. Cardeñoso-Payo, A. Viloria, C. E. Vivaracho, Q. I. Moro, J. J. Igarza, J. Sanchez, I. Hernaez, C. Orrite-Uruñuela, F. Martinez-Contreras and J. J. Gracia-Roche,

"BiosecurID: A Multimodal Biometric Database", *Pattern Analysis and Applications*, Vol. 13, n. 2, pp. 235-246, May 2010.

36. M. Castrillón-Santana, O. Déniz-Suárez, L. Antón-Canalís and J. Lorenzo-Navarro, *"*Face and facial feature detection evaluation".

37. O. H. Jensen, **"**Implementing the Viola-Jones Face Detection Algorithm", *IMM-M.Sc.-2008-93*, Kongens Lyngby 2008.

38. H. Schneiderman, T. Kanade, Robotics Institute: Object Recognition Using Statistical Modeling. [Online]

http://www.ri.cmu.edu/research_project_detail.html?project_id=320&menu_id=261.

39. MBGC, "Multiple biometric grand challenge," NIST - National Institute of Standard and Technology, http://face.nist.gov/mbgc/.

40. SourceForge, "*OpenCV Wiki"*. [Online] http://opencv.willowgarage.com.

41. J. Friedman, T. Hastie, R. Tibshirani, "Additive logistic regression: a statistical view of boosting", *Annals of Statistics*, Vol. 28, 2000.

42. Mathworks, "*The Mathworks - Product Support."* Online http://www.mathworks.com/support/tech-notes/1600/1605.html#intro.

43. H.-S. Kim, W.-S. Kang, J.-I. Shin, S.-H. Park, "Face Detection Using Template Matching and Ellipse Fitting", *IEICE TRANSACTIONS on Information and Systems*. Vols. E83-D, pp. 2008-2011.

44. F. Rosenblatt, "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain" *Psychological Review*, Vol. 65, pp. 386-408, 1958.

45. M., Dr. Mario I. Chacon. *"State of the Art in Face Recognition"* In-Teh, 2009 ISBN 978-3-902613-42-4.

46. T. Kanade, "Picture Processing System by Computer Complex and Recognition of Human Faces", *PhD thesis, doctoral dissertation*, Kyoto University 1973.

47. S. Z. Li, A. K. Jain, "*Handbook of Face Recognition"* Springer. ISBN# 0-387-40595-x..

48. W. Zhao et al. "Face Recognition: A Literature Survey" *ACM Computing Surveys (CSUR)*, Vol. 35, pp. 399 – 458, ISSN:0360-03002003.

ANEXO I
PRESUPUESTO

1) Ejecución Material

- Compra de ordenador personal (Software incluido)..... 2.000 €
- Alquiler de impresora láser durante 6 meses ..... 260 €
- Material de oficina ..... 150 €
- Total de ejecución material..... 2.410 €

2) Gastos generales

- 16 % sobre Ejecución Material ..... 385,6 €

3) Beneficio Industrial

- 6 % sobre Ejecución Material ..... 144,6 €

4) Honorarios Proyecto

- 640 horas a 15 € / hora ..... 9600 €

5) Material fungible

- Gastos de impresión ..... 60 €
- Encuadernación..... 200 €

6) Subtotal del presupuesto

- Subtotal Presupuesto ..... 12800,2 €

7) I.V.A. aplicable

- 16% Subtotal Presupuesto ..... 2048,03 €

8) Total presupuesto

- Total Presupuesto ..... 14848,23 €

Madrid, Junio de 2010
El Ingeniero Jefe de Proyecto

Fdo.: Andrei Bogdan Dragolici
Ingeniero Superior de Telecomunicación

ANEXO II
PLIEGO DE CONDICIONES

Este documento contiene las condiciones legales que guiarán la realización, en este proyecto, de un sistema de detección facial y un sistema de detección de ojos. En lo que sigue, se supondrá que el proyecto ha sido encargado por una empresa cliente a una empresa consultora con la finalidad de realizar dicho sistema. Dicha empresa ha debido desarrollar una línea de investigación con objeto de elaborar el proyecto. Esta línea de investigación, junto con el posterior desarrollo de los programas está amparada por las condiciones particulares del siguiente pliego.

Supuesto que la utilización industrial de los métodos recogidos en el presente proyecto ha sido decidida por parte de la empresa cliente o de otras, la obra a realizar se regulará por las siguientes:

Condiciones generales

I. La modalidad de contratación será el concurso. La adjudicación se hará, por tanto, a la proposición más favorable sin atender exclusivamente al valor económico, dependiendo de las mayores garantías ofrecidas. La empresa que somete el proyecto a concurso se reserva el derecho a declararlo desierto.

II. El montaje y mecanización completa de los equipos que intervengan será realizado totalmente por la empresa licitadora.

III. En la oferta, se hará constar el precio total por el que se compromete a realizar la obra y el tanto por ciento de baja que supone este precio en relación con un importe límite si este se hubiera fijado.

IV. La obra se realizará bajo la dirección técnica de un Ingeniero Superior de Telecomunicación, auxiliado por el número de Ingenieros Técnicos y Programadores que se estime preciso para el desarrollo de la misma.

V. Aparte del Ingeniero Director, el contratista tendrá derecho a contratar al resto del personal, pudiendo ceder esta prerrogativa a favor del Ingeniero Director, quien no estará obligado a aceptarla.

VI. El contratista tiene derecho a sacar copias a su costa de los planos, pliego de condiciones y presupuestos. El Ingeniero autor del proyecto autorizará con su firma las copias solicitadas por el contratista después de confrontarlas.

VII. Se abonará al contratista la obra que realmente ejecute con sujeción al proyecto que sirvió de base para la contratación, a las modificaciones autorizadas por la superioridad o a las órdenes que con arreglo a sus facultades le hayan comunicado por escrito al Ingeniero Director de obras siempre que dicha obra se haya ajustado a los preceptos de los pliegos de condiciones, con arreglo a los cuales, se harán las modificaciones y la valoración de las diversas unidades sin que el importe total pueda exceder de los presupuestos aprobados. Por consiguiente, el número de unidades que se consignan en el proyecto o en el presupuesto, no podrá servirle de fundamento para entablar reclamaciones de ninguna clase, salvo en los casos de rescisión.

VIII. Tanto en las certificaciones de obras como en la liquidación final, se abonarán los trabajos realizados por el contratista a los precios de ejecución material que figuran en el presupuesto para cada unidad de la obra.

IX. Si excepcionalmente se hubiera ejecutado algún trabajo que no se ajustase a las condiciones de la contrata pero que sin embargo es admisible a juicio del Ingeniero Director de obras, se dará conocimiento a la Dirección, proponiendo a la vez la rebaja de precios que el Ingeniero estime justa y si la Dirección resolviera aceptar la obra, quedará el contratista obligado a conformarse con la rebaja acordada.

X. Cuando se juzgue necesario emplear materiales o ejecutar obras que no figuren en el presupuesto de la contrata, se evaluará su importe a los precios asignados a otras obras o materiales análogos si los hubiere y cuando no, se discutirán entre el Ingeniero Director y el contratista, sometiéndolos a la aprobación de la Dirección. Los nuevos precios convenidos por uno u otro procedimiento, se sujetarán siempre al establecido en el punto anterior.

XI. Cuando el contratista, con autorización del Ingeniero Director de obras, emplee materiales de calidad más elevada o de mayores dimensiones de lo estipulado en el proyecto, o sustituya una clase de fabricación por otra que tenga asignado mayor precio o ejecute con mayores dimensiones cualquier otra parte de las obras, o en general, introduzca en ellas cualquier modificación que sea beneficiosa a juicio del Ingeniero Director de obras, no tendrá derecho sin embargo, sino a lo que le correspondería si hubiera realizado la obra con estricta sujeción a lo proyectado y contratado.

XII. Las cantidades calculadas para obras accesorias, aunque figuren por partida alzada en el presupuesto final (general), no serán abonadas sino a los precios

de la contrata, según las condiciones de la misma y los proyectos particulares que para ellas se formen, o en su defecto, por lo que resulte de su medición final.

XIII. El contratista queda obligado a abonar al Ingeniero autor del proyecto y director de obras así como a los Ingenieros Técnicos, el importe de sus respectivos honorarios facultativos por formación del proyecto, dirección técnica y administración en su caso, con arreglo a las tarifas y honorarios vigentes.

XIV. Concluida la ejecución de la obra, será reconocida por el Ingeniero Director que a tal efecto designe la empresa.

XV. La garantía definitiva será del 4% del presupuesto y la provisional del 2%.

XVI. La forma de pago será por certificaciones mensuales de la obra ejecutada, de acuerdo con los precios del presupuesto, deducida la baja si la hubiera.

XVII. La fecha de comienzo de las obras será a partir de los 15 días naturales del replanteo oficial de las mismas y la definitiva, al año de haber ejecutado la provisional, procediéndose si no existe reclamación alguna, a la reclamación de la fianza.

XVIII. Si el contratista al efectuar el replanteo, observase algún error en el proyecto, deberá comunicarlo en el plazo de quince días al Ingeniero Director de obras, pues transcurrido ese plazo será responsable de la exactitud del proyecto.

XIX. El contratista está obligado a designar una persona responsable que se entenderá con el Ingeniero Director de obras, o con el delegado que éste designe, para todo relacionado con ella. Al ser el Ingeniero Director de obras el que interpreta el proyecto, el contratista deberá consultarle cualquier duda que surja en su realización.

XX. Durante la realización de la obra, se girarán visitas de inspección por personal facultativo de la empresa cliente, para hacer las comprobaciones que se crean oportunas. Es obligación del contratista, la conservación de la obra ya ejecutada hasta la recepción de la misma, por lo que el deterioro parcial o total de ella, aunque sea por agentes atmosféricos u otras causas, deberá ser reparado o reconstruido por su cuenta.

XXI. El contratista, deberá realizar la obra en el plazo mencionado a partir de la fecha del contrato, incurriendo en multa, por retraso de la ejecución siempre que éste no sea debido a causas de fuerza mayor. A la terminación de la obra, se hará una recepción provisional previo reconocimiento y examen por la

dirección técnica, el depositario de efectos, el interventor y el jefe de servicio o un representante, estampando su conformidad el contratista.

XXII. Hecha la recepción provisional, se certificará al contratista el resto de la obra, reservándose la administración el importe de los gastos de conservación de la misma hasta su recepción definitiva y la fianza durante el tiempo señalado como plazo de garantía. La recepción definitiva se hará en las mismas condiciones que la provisional, extendiéndose el acta correspondiente. El Director Técnico propondrá a la Junta Económica la devolución de la fianza al contratista de acuerdo con las condiciones económicas legales establecidas.

XXIII. Las tarifas para la determinación de honorarios, reguladas por orden de la Presidencia del Gobierno el 19 de Octubre de 1961, se aplicarán sobre el denominado en la actualidad "Presupuesto de Ejecución de Contrata" y anteriormente llamado "Presupuesto de Ejecución Material" que hoy designa otro concepto.

Condiciones particulares

La empresa consultora, que ha desarrollado el presente proyecto, lo entregará a la empresa cliente bajo las condiciones generales ya formuladas, debiendo añadirse las siguientes condiciones particulares:

I. La propiedad intelectual de los procesos descritos y analizados en el presente trabajo, pertenece por entero a la empresa consultora representada por el Ingeniero Director del Proyecto.

II. La empresa consultora se reserva el derecho a la utilización total o parcial de los resultados de la investigación realizada para desarrollar el siguiente proyecto, bien para su publicación o bien para su uso en trabajos o proyectos posteriores, para la misma empresa cliente o para otra.

III. Cualquier tipo de reproducción aparte de las reseñadas en las condiciones generales, bien sea para uso particular de la empresa cliente, o para cualquier otra aplicación, contará con autorización expresa y por escrito del Ingeniero Director del Proyecto, que actuará en representación de la empresa consultora.

IV. En la autorización se ha de hacer constar la aplicación a que se destinan sus reproducciones así como su cantidad.

V. En todas las reproducciones se indicará su procedencia, explicitando el nombre del proyecto, nombre del Ingeniero Director y de la empresa consultora.

VI. Si el proyecto pasa la etapa de desarrollo, cualquier modificación que se realice sobre él, deberá ser notificada al Ingeniero Director del Proyecto y a criterio de éste, la empresa consultora decidirá aceptar o no la modificación propuesta.

VII. Si la modificación se acepta, la empresa consultora se hará responsable al mismo nivel que el proyecto inicial del que resulta el añadirla.

VIII. Si la modificación no es aceptada, por el contrario, la empresa consultora declinará toda responsabilidad que se derive de la aplicación o influencia de la misma.

IX. Si la empresa cliente decide desarrollar industrialmente uno o varios productos en los que resulte parcial o totalmente aplicable el estudio de este proyecto, deberá comunicarlo a la empresa consultora.

X. La empresa consultora no se responsabiliza de los efectos laterales que se puedan producir en el momento en que se utilice la herramienta objeto del presente proyecto para la realización de otras aplicaciones.

XI. La empresa consultora tendrá prioridad respecto a otras en la elaboración de los proyectos auxiliares que fuese necesario desarrollar para dicha aplicación industrial, siempre que no haga explícita renuncia a este hecho. En este caso, deberá autorizar expresamente los proyectos presentados por otros.

XII. El Ingeniero Director del presente proyecto, será el responsable de la dirección de la aplicación industrial siempre que la empresa consultora lo estime oportuno. En caso contrario, la persona designada deberá contar con la autorización del mismo, quien delegará en él las responsabilidades que ostente.

ANEXO III
MANUAL DEL PROGRAMADOR

Anexo III - Programación C/C++ con Matlab

En este capítulo se muestra una breve descripción de la programación en Matlab combinando archivos C/C++. Con esto se mantienen las funcionalidades y beneficios de la programación en Matlab aumentando la velocidad de trabajo de los programas gracias a los archivos C/C++. Con esto pretendemos subsanar los fallos producidos en la etapa anterior. La información necesaria para realizar esta funcionalidad se ha extraído de Internet basándonos principalmente en la página de Mathworks [42]. El objetivo de este trabajo no es explicar en detalle el funcionamiento de este método de programación si no simplemente mostrar unas referencias generales al mismo.

MEX Files

MEX Files se refiere a Matlab Executable Files (archivos ejecutables Matlab). Estos archivos son subrutinas enlazadas de manera dinámica desde código fuente C, C++ o Fortran, el cual una vez compilado puede lanzarse desde Matlab del mismo modo que los archivos .m. Gracias a las funciones de interfaces externas obtenemos la funcionalidad para transferir datos entre los archivos MEX y Matlab, y la habilidad para llamar a las funciones Matlab desde C/C++.

Los componentes de un archivo MEX de C/C++ son los siguientes:

- Una rutina *gateway* que sirve de interfaz entre C/C++ y Matlab. Es el punto de entrada a la librería compartida del archivo MEX. A través de esta rutina Matlab accede al resto de rutinas de los archivos MEX. El nombre de la rutina ha de ser: *mexFunction*. Un ejemplo sería [42]:

```
void mexFunction(
    int nlhs, mxArray *plhs[],
    int nrhs, const mxArray *prhs[])
{
   /* more C/C++ code ... */
}
```

Este tipo de rutinas debe contener los siguientes parámetros:

1. prhs: Array de argumentos de entrada.
2. plhs: Array de argumentos de salida.
3. nrhs: El número de argumentos de entrada, es decir, el tamaño del array prhs.
4. nlhs: El número de argumentos de salida, es decir, el tamaño del array plhs.

- Una rutina computacional, que es la encargada de realizar las operaciones para las que se construye el archivo.

Ejemplo de flujo de datos en archivos MEX

A continuación se muestra un ejemplo del flujo de datos en los archivos MEX, reflejado en la Figura A.1 Ejemplo de flujo de datos en un archivo MEXFigura A.1:

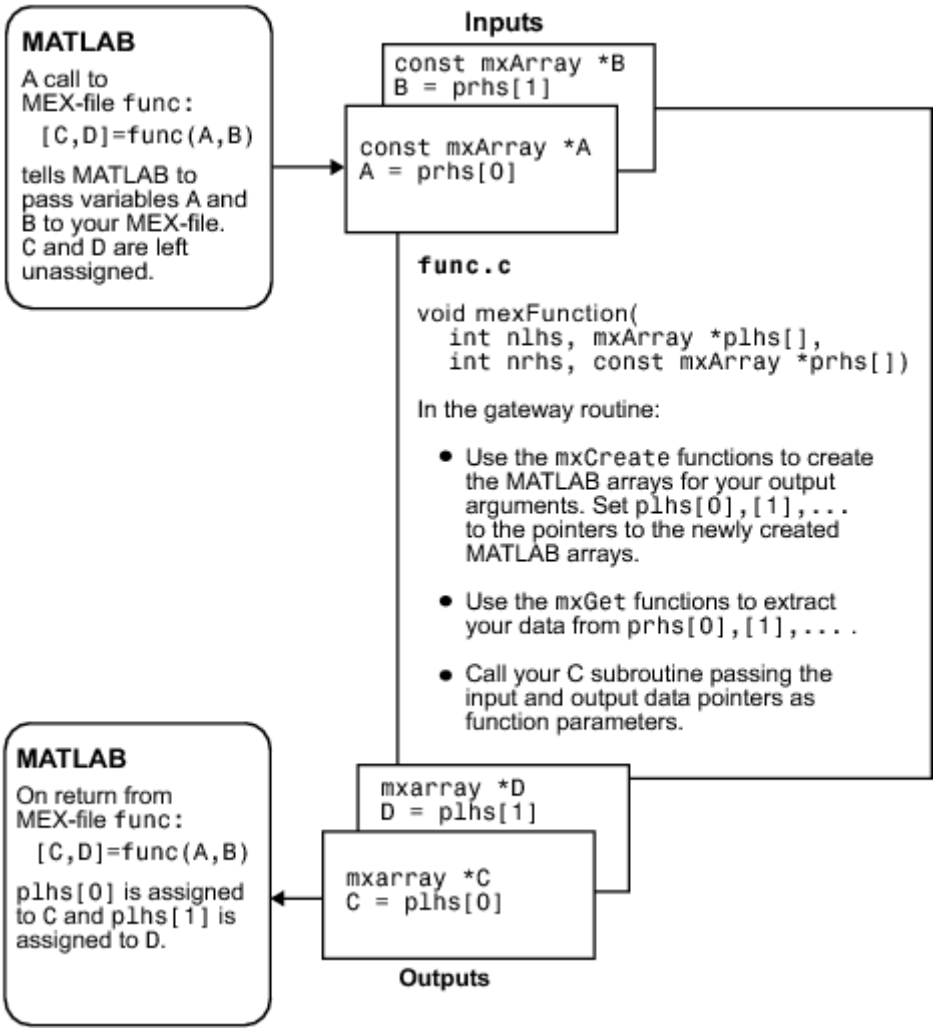


Figura A.1 Ejemplo de flujo de datos en un archivo MEX [42].

En este ejemplo la sintáxis de la función representada es `[C, D] = func(A,B)`. La rutina *gateway* en `func.c` utiliza la función `mxCreate*` para crear los *arrays* de Matlab para las variables de salida y relaciona los punteros `plhs[0]` y `plhs[1]` con estas variables. Utiliza las funciones `mxGet*` para extraer los datos de entrada que se le envían a las funciones desde Matlab. Por último se llama a las rutinas computacionales para realizar las operaciones necesarias.

Creación paso a paso de un archivo MEX.

Se adjunta la creación de un archivo MEX "Hola mundo" de manera sencilla. Con esto se pretende dejar claros cuales son los pasos básicos para la creación de estos archivos de manera esquemática.

4. Todo archivo MEX, incluye la librería `mex.h`:

```
#include "mex.h"
```

5. Cada archivo tiene la función `mexFunction` como entrada y una o varias funciones dentro que hagan el trabajo requerido:

```
#include "mex.h"
void mexFunction(int nlhs, mxArray *plhs[],
    int nrhs, const mxArray *prhs[]) {
  mexPrintf("Hello, world!\n");
}
```

6. Debemos guardar el archivo como *nombre_archivo.c*, en este caso *hello.c*

7. El siguiente paso es decirle a Matlab que compilador utilizar para construir el archivo MEX. Utilizando la instrucción: mex –setup. Con esta instrucción se puede elegir el compilador que se quiera utilizar, de entre los que hay instalados en el sistema. En caso de no tener ninguno, Matlab posee un compilador, LCC.

8. Ahora se puede compilar el archivo mediante la instrucción:

```
mex hello.c
```

9. Con esto ya tenemos preparado el archivo. Simplemente con una llamada desde la línea de comandos, o un archivo .m, se ejecutaría:

```
hello
```

SALIDA:	Hello, world!

```
function [D] = haar_cascade_detector (I , modelo , escalado ,
cascada);

Entrada:
---------
    I             = Imagen de entrada (Ny x Nx) con format UINT8
    modelo        = Estructura del modelo entrenado
      Param             = Matriz con las características entrenadas (3
                        x T). Cada fila corresponde a:
           featureIdx   = Índices de las mejores características(1 x T)
           th           = Umbrales óptimos para cada caract. (1 x T)
           a            = Pesos de las características (1 x T)
      dimsItraining     = Tamaño de las imágenes de entrenamiento.
      rect_param        = Estructura de las características utilizadas.
      F                 = Valores de las características
      Postprocessing    = 0: Mostrar todas las detecciones; 1 procesar
                        las que se solapan en más de un 25%.
    escalado      = [scale_ini , scale_inc , step_ini]
      scale_ini         = Escalado inicial sobre el tamaño de las img
                        de entrenamiento.
      scale_inc         = Factor de increment.
      step_ini          = Factor de desplazamiento.
    cascade       = Cascada (2 x T): fila1= nº de features por etapa;
                    fila2= umbral para cada clasificador fuerte.
Salida:
--------
    D             = Resultado de las detecciones (4 x numDetecciones)
```

```
function [model]  = haar_adaboost_model_cascade(II , y , [rect_param] ,
                    [F] , [T] , [options] , [premodel]);
Entrada:
---------
    II            = Imagenes integrales (Ny x Nx x N). Formato double.
    y             = Etiquetas cara/no_cara {1 , -1}
    rect_param    = Parámetros de las características (10 x nR):
      ip =              = [1,...,nP], donde nP es el número total de
                        patrones.
      wp                = ancho del patrón
      hp                = alto del patrón
      nrip              = numero de rectángulos del patrón
      nr                = índice del rectángulo en la característica
      xr,yr             = coordenadas (superior izq.) del rectángulo
      wr,hr             = ancho y alto del rectángulo actual.
      sr                = peso del rectángulo {-1; +1}
    F             = Lista de las características:
      if                = indice de la característica actual.
      xf,yf             = coordenadas (superior izq.) de la caract.
      wf,hf             = ancho y alto de la caract.
      ir                = indice del rectángulo en la caract. actual
    T             = Número de weak learners
    premodel      = Modelos de las etapas anteriores
```

```
Salida:
--------
    Modelo        = Modelo del clasificador de la etapa tratada
           featureIdx   = Índices de las mejores características(1 x T)
           th           = Umbrales óptimos para cada caract. (1 x T)
           a            = Pesos de las características (1 x T)
```

```
function [II] = image_integral_standard(X)

Normaliza y calcula la imagen integral de las imágenes de entrada

Entrada:
---------
    X             = Imagenes (Ny x Nx x N) doublé.
Salida:
--------
    II            = Imagenes integrales (Ny x Nx x N). Formato doublé.
```

```
function [] = lanzador_train_haar()

Lanza el entrenamiento de la cascada llamando a las funciones
adecuadas.
```

```
function [F] = haar_featlist(ny , nx , rect_param);

Calcula una lista de las características presentes en las imágenes de
entrenamiento.

Entrada:
---------
    ny            = Número de filas de las imágenes de entrenamiento.
    nx            = Número de columnas de las imágenes de
                    entrenamiento.
    rect_param    = Parámetros de las características (10 x nR):
      ip =              = [1,...,nP], donde nP es el número total de
                        patrones.
      wp                = ancho del patrón
      hp                = alto del patrón
      nrip              = numero de rectángulos del patrón
      nr                = índice del rectángulo en la característica
      xr,yr             = coordenadas (superior izq.) del rectángulo
      wr,hr             = ancho y alto del rectángulo actual.
      sr          = peso del rectángulo {-1; +1}
Salida:
--------
    F             = Lista de las características:
      if                = indice de la característica actual.
      xf,yf             = coordenadas (superior izq.) de la caract.
      wf,hf             = ancho y alto de la caract.
      ir                = indice del rectángulo en la caract. actual
```

```
function [] = detecta_sobre_BD_jpg();

Lanza el detector sobre una base de datos de imágenes en formato .jpg.
```

```
function [] = detecta_sobre_BD_gif();

Lanza el detector sobre una base de datos de imágenes en formato .gif.
```

```
function [h] = plot_rectangle(D , color);

Dibuja un rectángulo alrededor de las detecciones calculadas en la
imagen mostrada

Entrada:
---------
    D             = Resultado de las detecciones (4 x numDetecciones)
Salida:
--------
    II            = Imagenes integrales (Ny x Nx x N). Formato doublé.
```

ANEXO IV
IMÁGENES DE EJEMPLO DE LAS BASES DE DATOS

Base de datos de entrenamiento de Viola y Jones

Figura 0.1 Ejemplo de imágenes frontales utilizadas por Viola y Jones [1].

Base de datos de imágenes de Jensen

Figura 0.2 Imágenes de la Base de datos de Jensen

Base de datos BioSecurID

Figura 0.3 Ejemplo base de datos BioSecurID

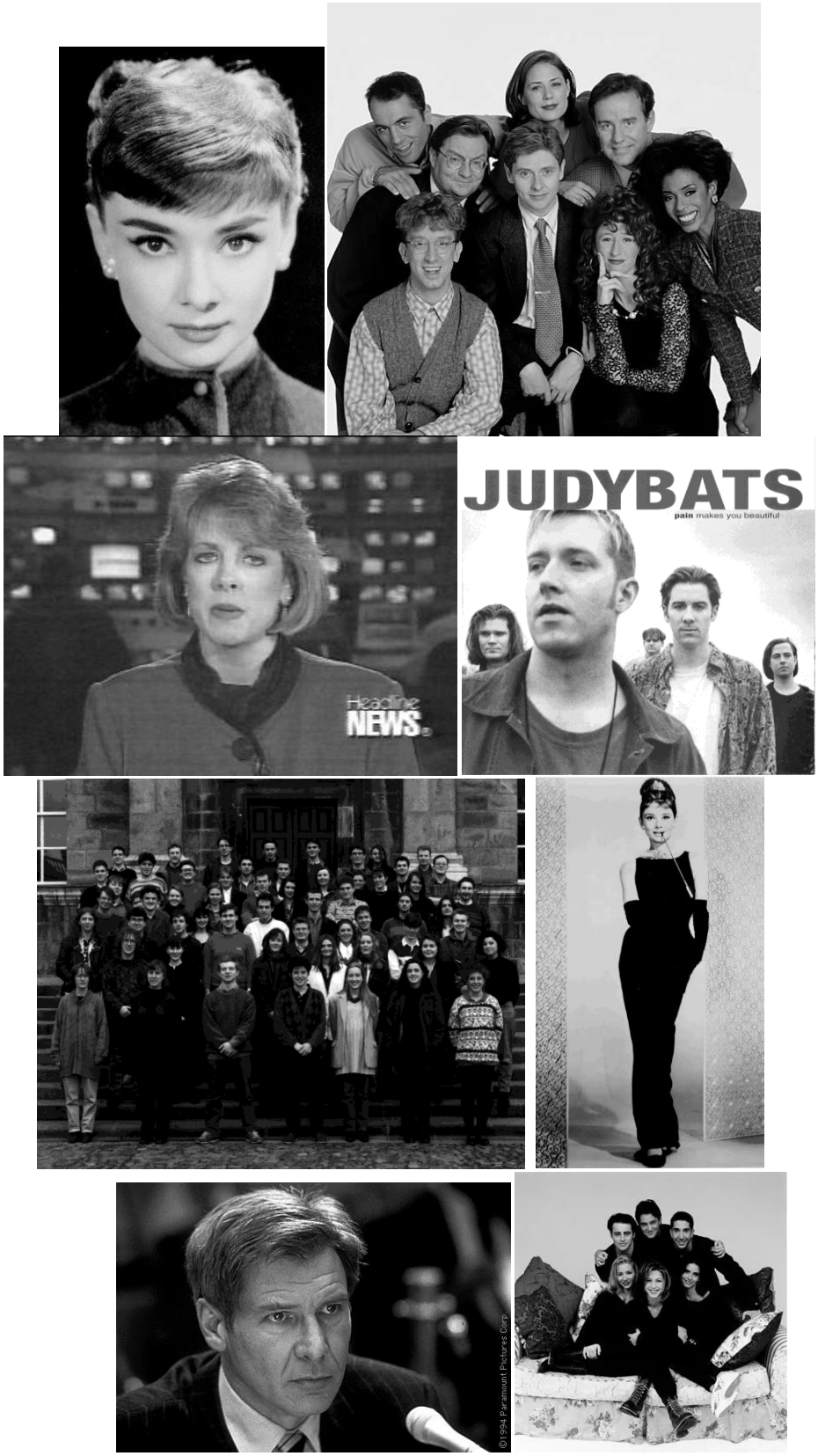


Figura 0.4 Ejemplo de imágenes de la base CMU.

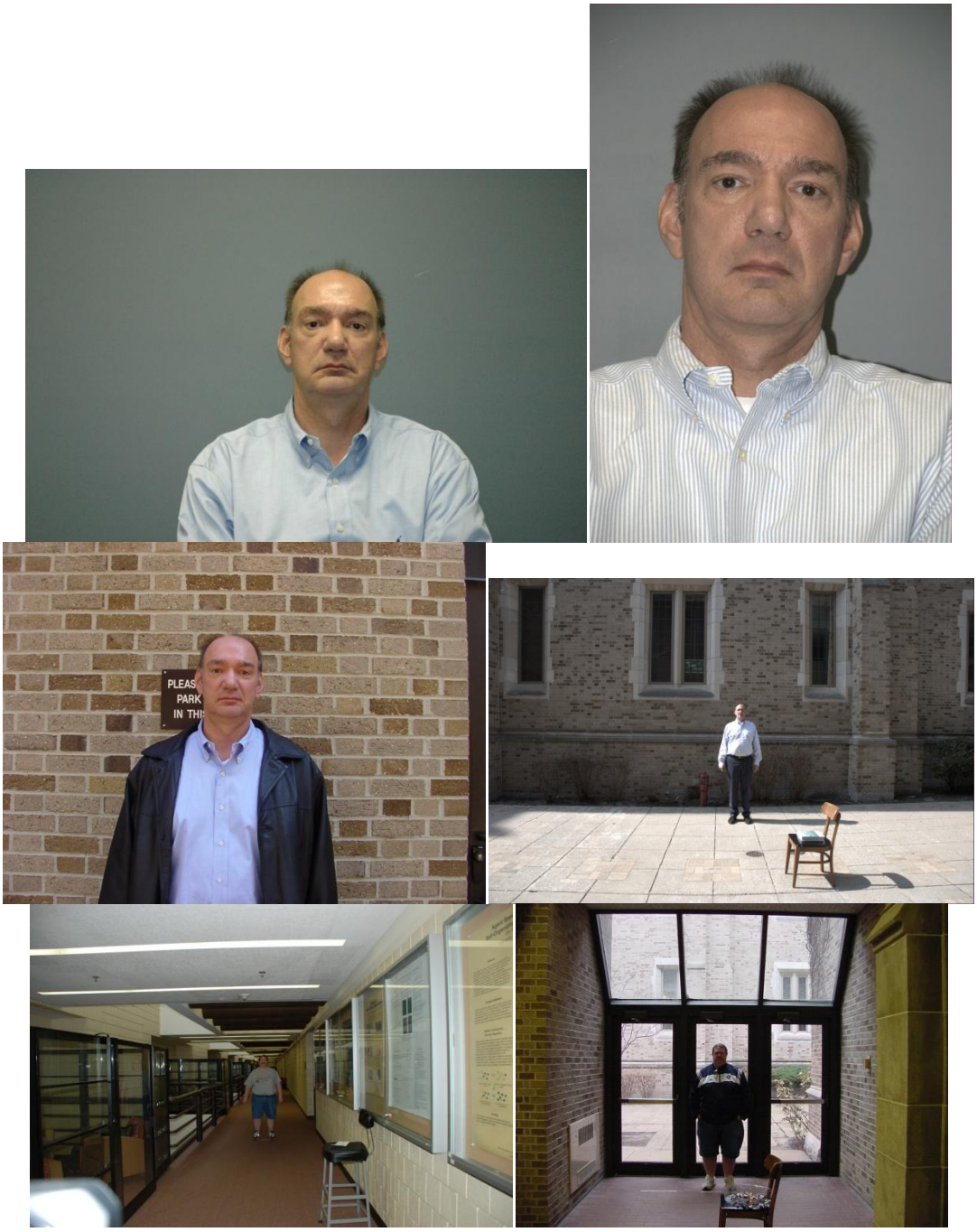


Figura 0.5 Ejemplo de imágenes de NIST.