Empleo de sistemas biométricos faciales aplicados al reconocimiento de personas en aeropuertos

Autor: David Vázquez Bermúdez

22 de junio de 2006

Agradecimientos

Quiero expresar mi amor y mi gratitud a mi madre y a mi padre, sin los cuales este proyecto no habría llegado a existir, a mi abuela y a la vida.

Así mismo, me gustaría agradecer a todos aquellos que, por su interés y participación en mi trabajo, me han ayudado a adquirir la experiencia que reflejan estas páginas: los componentes del grupo de Reconocimiento Facial y Visión Artificial, especialmente Cristina Conde, Ángel Serrano, Licesio J. Rodríguez e Isaac así como mi tutor Enrique Cabello, por su ayuda y complicidad en todo el periodo que ha supuesto el desarrollo del proyecto.

Finalmente quiero agradecer en nombre de FRAV la concesión del proyecto realizada por el IUISI, las facilidades ofrecidas por AENA y a los voluntarios que accedieron a transitar por Barajas para ser estudiados. También queremos agradecer muy sinceramente, la inestimable y valiosa ayuda del personal de la Guardia Civil destinado en Barajas.

> "Mientras los filósofos discuten si es posible o no la inteligencia electrónica, los investigadores la construyen" C. Frabetti

Índice general

1. Introducción **1**

2. Objetivos **7**

2.1. Descripción del problema 7

2.2. Principales objetivos 8

2.3. Estudio de alternativas 9

2.3.1. Detección de la cara en la escena. 11

2.3.2. Representación y clasificación de la cara. 12

2.4. Metodología empleada 13

3. Descripción informática **15**

3.1. Introducción 15

3.2. Especificación 16

3.2.1. Requisitos 16

3.2.2. Especificación de casos de uso 19

3.3. Diseño 21

3.3.1. Diagramas de estados 22

3.3.2. Diagrama de casos de uso 22

3.4. Implementación 30

3.4.1. Detección de imágenes en la escena 30
3.4.2. Preprocesado de imágenes 30
3.4.3. Métodos de reducción de la dimensionalidad 31
3.4.4. Clasificador SVM 32
3.4.5. Post-procesado: Votación 33

4. Pruebas y resultados **34**
4.1. Pasos previos 34
4.1.1. Elección de las cámaras 34
4.1.2. Elaboración de las bases de datos 36
4.2. Pruebas de detección 38
4.3. Pruebas de identificación 43
4.3.1. Introducción: Curvas ROC 43
4.3.2. Experimentos realizados 45

5. Conclusiones **49**
5.1. Evaluación global del sistema 49
5.2. Conclusiones 50
5.3. Posibles trabajos futuros 51

A. Requisitos del sistema **54**
A.1. Requisitos Hardware 54
A.2. Requisitos software 54

B. Guía del usuario **56**
B.1. Instalación 56
B.2. Configuración de la base de datos 56
B.3. Entrenamiento del sistema 58
B.4. Análisis de un vídeo 63

C. Definición de Funciones **68**

C.1. Función principal del entrenamiento 68

C.2. Dependencias de primer orden: Dependencias de la función entrenar. 69

C.3. Función analizar 85

Índice de figuras

3.1. Diagrama de estados: Entrenamiento 23
3.2. Diagrama de estados: Análisis 24
3.3. Caso de Uso: Iniciar módulo de entrenamiento 25
3.4. Caso de uso: Iniciar entrenamiento 26
3.5. Caso de uso: Definir parámetros de la gráfica 27
3.6. Caso de uso: Iniciar módulo de análisis 28
3.7. Caso de uso: Iniciar el análisis de un vídeo 29
3.8. Estructura de una red neuronal 33

4.1. Imágenes de un sujeto en la BD Arcos 37
4.2. Imágenes del mismo sujeto en la BD Cinta 1 37
4.3. Imágenes del mismo sujeto en la BD Cinta 2 37
4.4. Imágenes del mismo sujeto en la BD FRAV2D 38
4.5. BD Arcos. Situaciones de localización normal 39
4.6. BD Cinta 1. Situaciones de localización normal 39
4.7. BD Cinta 2. Situaciones de localización normal 40
4.8. BD Cinta1. Situaciones de localización adversas 40
4.9. BD Cinta2 Ejemplos de fallos razonables 41
4.10. Ejemplos de fallos exagerados 42

4.11. Resultado de entrenar con FRAV2D y probar con Arcos . . . 46
4.12. Resultado de entrenar con FRAV2D y probar con Cinta 1 . . . 46
4.13. Resultado de entrenar con FRAV2D y probar con Cinta 2 . . . 47
4.14. Resultado de entrenar con Arcos y probar con Cinta 1 47
4.15. Resultado de entrenar con Cinta 1 y probar con Cinta 2 48

B.1. Esquema de una BD 57
B.2. Pantalla de configuración del entrenamiento 59
B.3. Pantalla de configuración del análisis de un vídeo 63
B.4. Pantalla de análisis de un video 64
B.5. Pantalla de individuo identificado 65

Índice de cuadros

2.1. Diseño en espiral 13

3.1. Inicio del entrenamiento 20
3.2. Cargar una configuración 20
3.3. Modificar los parámetros 20
3.4. Guardar una configuración 20
3.5. Iniciar el entrenamiento 20
3.6. Definir los parámetros de la gráfica 21
3.7. Inicio del análisis de un vídeo 21
3.8. Analizar un vídeo 22

4.1. Características de las Bases de Datos Obtenidas. 36
4.2. Resultados de localización. 42
4.3. BD Arcos. Número de imágenes obtenidas de cada individuo (Analizando los frames pares únicamente) 43
4.4. BD Cinta2. Número de imágenes obtenidas de cada individuo (Analizando los frames pares únicamente) 43
4.5. Resultados obtenidos con los diferentes métodos 45

5.1. EER con FRAV2D 50

Resumen

El presente proyecto se desarrolló a lo largo del año 2005 y 2006, probando un prototipo de un sistema de verificación facial con imágenes extraídas de las cámaras de video-vigilancia del aeropuerto de Barajas. Se diseñaron varios experimentos, agrupados en dos clases. En el primer tipo, el sistema es entrenado con imágenes obtenidas en condiciones de laboratorio y luego probado con imágenes extraídas de las cámaras de video-vigilancia del aeropuerto de Barajas. En el segundo caso, tanto las imágenes de entrenamiento como las de prueba corresponden a imágenes extraídas de Barajas.

Se ha desarrollado un sistema completo, que incluye adquisición y digitalización de las imágenes, localización y recorte de las caras en escena, verificación de sujetos y obtención de resultados. Los resultados muestran que, en general, un sistema de verificación facial basado en imágenes puede ser una valiosa ayuda a un operario que deba estar vigilando amplias zonas.

Capítulo 1

Introducción

Los sistemas del reconocimiento facial son sistemas de seguridad basados en computación que son capaces de detectar e identificar rostros humanos automáticamente. Estos sistemas dependen de un algoritmo de reconocimiento, por ejemplo, "Eigenface" o el "Modelo Oculto de Markov". El primer paso de un sistema de reconocimiento facial es detectar el rostro humano y extraerlo del resto de la escena. Después, el sistema mide puntos de interés del rostro, por ejemplo, la distancia entre los ojos, la forma de las mejillas y otras características distintivas. Finalmente, estos puntos se comparan con los puntos de interés de las fotos guardadas en la base de datos para encontrar una semejanza entre ellos. Obviamente, el sistema está limitado por el ángulo con el que rostro capturado y la luminosidad presente en la imagen. Actualmente, se están desarrollando nuevas tecnologías para crear modelos tridimensionales del rostro humano basados en una foto digital para obtener más puntos de interés que comparar.

Por causa de los ataques terroristas del 11 de septiembre 2001 a los EE.UU., defensores de los derechos de privacidad, grupos de ciudadanos, líderes políticos y los fabricantes de esta misma tecnología, están debatiendo si estas tecnologías deben ser utilizadas aún más extensamente; y si éste es el caso, ¿cómo serán regulados para proteger la privacidad del público? Algunos aeropuertos están considerando la instalación de sistemas de reconocimiento facial como una medida de seguridad. Sin embargo, el aeropuerto interna-

cional de T.F. Green en la ciudad de Providence, Rhode Island, uno de los primeros aeropuertos en considerarlo, decidió en enero 2002 que debido a la posibilidad de un reconocimiento erróneo y otros defectos tecnológicos del sistema de reconocimiento facial, no iban a utilizarlo.

El gobierno de los EE.UU. está invirtiendo mucho dinero en la tecnología y programas de vigilancia a la espera de otros proyectos. Los departamentos de Estado, Energía, Justicia y Defensa han gastado al menos *47 000 000 $* en estos sistemas. En el año 2006, el gobierno planea añadir el reconocimiento facial a las aplicaciones con Visa que también incluiría huellas dactilares. Ésto, a pesar de que el gobierno estima que la incorporación de sistemas biométricos en las tarjetas Visa costará entre 1,3 y 2,9 billones de $ al inicio, y entre 700 millones y 1500 billones al año. Los fondos federales han sido usados por ciudades y estados para comprar dispositivos de reconocimiento facial para vehículos del departamento de policía. Los nuevo pasaportes de los EE.UU y los DNI creados bajo REALID 2005 incluirán fotografías digitales con una posible conexión con los sistemas de reconocimiento facial. De cualquier modo, varias pruebas, incluyendo las conducidas por el Instituto Nacional de Normas y Tecnología y el Departamento de Defensa, muestran que estos sistemas de reconocimiento facial pueden ser fácilmente burlados por gente que no colabore y por cambios en el ambiente como la posición o las luces. Estos sistemas corren riesgos importantes de privacidad y actualmente no hay leyes que puedan regular estos sistemas para prevenir abusos.

Simplificando, los sistemas de reconocimiento facial comparan un patrón digital con la cara de una persona que camina por la calle o entre la multitud con imágenes de bases de datos. Cambios en el entorno como posición, luz y sombras pueden afectar a la recolección de datos biométricos. Cambios en la expresión facial, ángulo de la cara hacia la cámara, peinados y barbas son particularmente efectivos para confundir a los sistemas de reconocimiento facial. En una prueba la oficina de contabilidad del gobierno encontró que una persona moviendo su cara de izquierda a derecha fue capaz de burlar el sistema. La tecnología de reconocimiento del departamento de defensa ha descubierto que la tecnología interpreta de forma inefectiva una diferencia

de posición de 45 grados entre la imagen obtenida y la de la base de datos, y una diferencia de 15 grados es lo único que se necesita para afectar desfavorablemente la tecnología de reconocimiento facial.

La efectividad de un sistema de identificación biométrica está determinada por cómo el sistema es actualizado, protegido y mantenido. Hay varias maneras de comprometer la efectividad de un sistema biométrico: por falsa identificación en el desarrollo o hakeando para falsificar los datos. Los errores de los sistemas de reconocimiento facial pueden involucrar a gente inocente y/o permitir a los sospechosos pasar desapercibidos.

En el 2002, NIST llevó a cabo un estudio de los sistemas de reconocimiento facial. NIST descubrió que la media de reconocimiento para caras capturadas en entornos no controlados como exteriores podían ser menos del 50 %. NIST también determinó que debido a las altas tasas de fallos cuando era aplicado a grupos grandes de personas no hacía al reconocimiento facial una tecnología viable para la identificación a gran escala. El tiempo también afecta a los sistemas de reconocimiento facial, cuanto más tiempo pase entre la fotografía original de la base de datos y de la imagen capturada, menos correcto será el resultado del sistema de reconocimiento facial.

Tampa es una de las ciudades de los EEUU que ha usado tecnología de reconocimiento facial con sistemas de vídeo-vigilancia para escanear al público a escondidas. Tampa utilizó cámaras para escanear las caras de la gente en la SuperBowl del 2001 y posteriormente el departamento de policía utilizó un sistema parecido para vigilar la zona de vida nocturna del districto de Albor. Un informe de la Unión Americana para las Libertades Civiles encontró que:

> El sistema no encontró ninguna cara en su base de datos de sospechosos por lo que no les sirvió en sus arrestos. Además, su base de datos fotográfica contenía, a parte de una selección de sospechos buscados por la policía, gente que pudiese tener información valiosa.

En agosto de 2003, Tampa dejó de usar este software por culpa de sus fallos

y lo sustituo por 'Identix'. El portavóz del departamento de policía aseguró que el sistema no tiene nungún tipo de utilidad para ellos.

El caso de Tampa es un ejemplo de los riesgos para la privacidad que crea algo como un sistema de reoconocimiento facial. Lo que empezó como un sistema para encontrar criminales se ha convertido en un sistema para encontrar gente que podría tener información que la policía está buscando. Ésto es una *pobre razón* para invadir la privacidad del público en general. Una persona puede ser escaneada sin su conocimiento o su consentimiento. Así mismo, se considera actividad sospechosa merodear una zona de vida nocturna.

El aeropuerto de Logan en Boston puso en marcha dos pruebas distintas de sistemas de reconocimiento facial -sustituidos por la tecnología Visage e Identix- en 2002. Ambos sistemas tenían unas altas tasas de error y el aeropuerto prescindió de esta tecnología. Los sistemas de reconocimiento facial también fallaban en los aeropuertos de Dallas/Fort Worth, Fresno, California y Palm Beach County... Un ejemplo de l*os* fallos de los sistemas de reconocimiento facial sucedió cuando dos personas intercambiaron sus pasaportes como una broma. Su engaño no fue detectado por los sistemas de reconocimiento facial por lo que las autoridades Australianas aun están probando esta tecnología, lo que podría abarcar 3 años más.

A pesar de ello, los sistemas de reconocimiento facial todavía se utilizan. El departamento de Defensa de EEUU gastó 1.6 millones de dólares en pagar a Identix para investigar sobre estos sistemas. El registro de vehículos de Massachusets gastó 1,5 millones de dólares de los fondos federales en el sistemas de reconocimiento facial suministrado por Digimarc. La playa de Virginia empleó 150 000 dólares de los fondos federales más 50 000 dólares de sus propios fondos en activar en un sistema de cámaras de reconocimiento facial suministrado por Identix. El departamento de Salud Pública de Texas contrató recientemente a Identix por 1.8 millones de dólares para incluir capacidad de reconocimiento facial en la actualización de sus sistemas. Una vez más, una persona puede estar tranquilamente bajando una calle y tener una terminal de reconocimiento facial fijándose en él.

Las dos mayore compañías suministradoras de tecnologías de reconocimien-

to facial son Massachusetts (basado en Viisage) Technology and Minnesota (basado en Identix).

Como hemos visto anteriormente, en los EEUU se han incrementado los sistemas de cámaras de video-vigilancia. Ciudades como Chicago y Baltimore invierten dinero en garantizar la seguridad creando redes de cámaras para vigilar a la gente por las calles, aeropuertos, centros comerciales... De cualquier forma, los estudios han llegado a la conclusión de que estos sistemas tienen pocos efectos en la reducción de los crímenes y es más efectivo poner más guardias en las calles y más luz en las zonas de alta criminalidad.

El software de reconocimiento facial también ha aumentado su uso a nivel internacional. La Organización Internacional de Aviación Civil ha sugerido incorporar sistemas de huellas dactilares y reconocimiento facial en los pasaportes. El Reino Unido que tiene una amplia red de cámaras de video-vigilancia, ha incluido sistemas de reconocimiento facial a ésta. Londres que posee más de 200 000 cámaras y el resto del país más de 4 millones de cámaras. Se estima que hay una cámara por cada 14 personas y la media en Briton es de que cada persona sea vista por 300 cámaras cada día.

Los departamentos de estado han hecho una nueva entrega de pasaportes para EEUU en los que pedían fotografías digitales. Podrían incluir estas fotografías a sus actuales grandes bases de datos de imágenes digitales, recolectadas durante los pasados 10 años desde que se pusieron en funcionamiento las aplicaciones para pasaportes de 70 millones de americanos. El REALID solicitaba fotografías digitales en sus DNI. Pronto, cada persona que continue con su pasaporte de los EEUU o su tarjeta REALID tendrá su cara en una base de datos. Actualmente hay 1.2 billones de fotos digitales en bases de datos al rededor del mundo según el presidente de Identix.

Los sistemas de reconocimiento facial todavía se usan para escanear a multitudes en busca de criminales o terroristas sospechosos pero también en busca de gente que podría tener información valiosa para la policía, sea lo que sea esta información valiosa. No es tan difícil para el gobierno usar las redes de reconocimiento facial habilitando cámaras de vigilancia para buscar entre las multitudes a alguien que está en una base de datos que contiene millones de

personas aunque no sean sospechosos y todavía no es ilegal monitorizarlos sin previo aviso.

Hay quien recomienda que se suspendan estos sistemas hasta que se complete una evaluación de su utilidad y efectividad. Algunos todavía van más allá recomendando que la evaluación se reconduzca antes de desplegar ningún sistema de reconocimiento facial y que se establezcan las directrices legales para el uso de los sistemas de reconocimiento facial.

Capítulo 2

Objetivos

2.1. Descripción del problema

El presente proyecto,"Empleo de sistemas biométricos faciales aplicados al reconocimiento de personas en aeropuertos", surge como respuesta a la oferta de la Convocatoria de Ayudas de fecha 10 de septiembre de 2004 realizada por el Instituto Universitario de Investigación sobre Seguridad Interior.

Se pretende evaluar un sistema biométrico basado en rasgos faciales en un entorno real, como puede ser el aeropuerto de Barajas[1] que permita el procesamiento de imágenes sin que los sujetos estudiados se percaten de ello[2]. Además, los sistemas basados en caras son muy útiles a los operadores humanos, ya que utilizan imágenes que luego pueden ser mostradas para su cotejo por parte de dicho operador.

Así mismo, se necesita conseguir un estudio de viabilidad del sistema de cámaras de Barajas para conocer si el uso de un sistema biométrico podría suponer el tener que realizar la instalación de ordenadores y de cámaras de videovigilancia de alta calidad.

[1] La mayoría de los sistemas biométricos utilizados en zonas de seguridad suelen estar basados en huellas dactilares, pero este es un método muy intrusivo, es decir, obliga a que el usuario sea muy colaborativo.

[2] No es posible tomar la huella de una persona sin que se percate de ello y por lo tanto, dé su consentimiento.

Además, se desea probar un sistema biométrico en un entorno en el cual la iluminación no esté controlada, pero requiera, al menos potencialmente, unas condiciones de seguridad bastante altas. Estas circunstancias se dan en el aeropuerto de Barajas, sumadas además al hecho de que mantiene un flujo de sujetos muy alto.

Finalmente, se pretende que el sistema de verificación facial basado en imágenes pudiera ser una ayuda a un operario que deba estar vigilando amplias zonas. Para ello, se debía desarrollar un sistema completo, que incluyera adquisición y digitalización de las imágenes, localización y recorte de las caras en escena, verificación de sujetos y obtención de resultados.

2.2. Principales objetivos

Los distintos objetivos que se pretendían alcanzar se detallan a continuación:

- Evaluar un sistema biométrico basado en rasgos faciales en un entorno real. Para ello se plantean nuevos objetivos:

  1. Elección de cámaras
  2. Creación de bases de datos de los individuos a identificar
  3. Diseño de los experimentos
  4. Probar los métodos de reconocimiento facial del programa NRECO del FRAV
  5. Obtención de los resultados de la identificación

- Implementación de un sistema de verificación facial basado en imágenes pudiera ser una ayuda a un operario que deba estar vigilando amplias zonas. Para ello se plantean nuevos objetivos:

  - Diseño de una interfaz gráfica para el NRECO
  - Modificación y ampliación de las prestaciones del NRECO

- Implementación de un módulo de análisis de vídeos
  - Obtención de frames de los vídeos
  - Detección de las caras en los frames
  - Agrupación de las caras de un mismo sujeto
  - Obtención de las proyecciones de las caras
  - Identificación del sujeto
  - Obtención de los resultados
- Diseño de los experimentos
- Realización de las pruebas
- Obtención de los resultados de identificación

2.3. Estudio de alternativas

En la Bibliografía se mencionan diferentes planteamientos para el reconocimiento facial, aunque sólo se aborda un problema, con poca o nula incidencia de los otros[3]. Los distintos trabajos difieren tanto en la forma de representar las caras como en el esquema de correspondencia aplicado.

Todos estos métodos toman como hipótesis de partida el hecho de que para un rostro, los valores de las características que lo definen no varían mucho en diferentes imágenes[4]. Es más, si un conjunto de características es muy diferente en dos imágenes los rostros correspondientes serán también diferentes.

La mayoría de los métodos de reconocimiento facial utilizan una serie de supuestos, más o menos explícitos, como son:

- La imagen suele ser frontal o perfil, por lo tanto va a ser posible encontrar todos los rasgos en la imagen.

[3]Dado que se busca identificar, se supone que la localización ha sido ya hecha y las características ya extraídas

[4]Hipótesis no del todo cierta, puesto que en las imágenes faciales influyen mucho aspectos como la posición, la rotación, los gestos o la iluminación de la cara.

- La cara esta derecha, con poca o ninguna inclinación o giro.
- No hay ocultamiento de trozos de imagen.
- El número de casos de prueba es relativamente pequeño.
- No aparece vello facial, gafas, sombreros, etc.
- La mayor parte de los casos son hombres de raza blanca.

Estos métodos tratan de simular las técnicas empleadas por los seres humanos en la identificación de personas. Estas son:

- La importancia de los rasgos faciales decrece de arriba hacia abajo (pelo >ojo >nariz >boca >barbilla). Pero caras con rasgos no habituales son identificadas rápidamente.
- Caras familiares son identificadas más rápidamente que otras menos familiares.
- Habitualmente, una persona reconoce a unos 700 casos, pero puede llegarse a algunos miles.
- Es difícil describir un rostro humano, ya que está lleno de numerosos estímulos visuales.
- El reconocimiento de una cara es el resultado de un análisis de rasgos globales junto con otros locales.

En general, todos los sistemas utilizan la misma secuencia de etapas para la identificación o verificación:

- Determinar un conjunto de características independientes para representar una cara.
- Representar las caras de entrenamiento en función de los valores que toman en ellas el conjunto de características seleccionado.

- Determinar los valores de una cara nueva (desconocida).
- Utilizar un criterio y un esquema de correspondencia para encontrar el mejor emparejamiento con las caras conocidas.

Podemos agrupar los cuatro puntos anteriores en tres problemas que deben ser resueltos y que son detallados a continuación.

1. Detección de la cara en escena.
2. Representación de la cara.
3. Clasificación de la cara, indicando el nombre del sujeto al que pertenece[5].

2.3.1. Detección de la cara en la escena.

En algunos casos las condiciones bajo las que se obtiene la imagen son controladas, por ejemplo las fotografías obtenidas por la policía o las obtenidas en el laboratorio. Por lo tanto la localización de la cara en la escena puede ser fácilmente determinada. En otros casos, la localización de la cara en la imagen no es conocida a priori. El primer paso, por lo tanto, es determinar si en la escena hay caras y si una cara está presente, localizarla en la imagen.

Varios factores tornan este problema complejo. Uno de ellos es el problema del vello facial (bigote, barba, etc.), maquillaje, etc. que enmascaran las características faciales. Otro es la variación en la escala y orientación de la cara en la imagen. Por último, existen otros dos factores que van a dificultar la detección: la iluminación que tenga la escena y la calidad de las imágenes.

La mayoría de los métodos de detección de caras se basan en plantillas flexibles, autocaras, redes neuronales o en el color de la cara.

---

[5]

a) Si el sistema debe averiguar el nombre del sujeto partiendo únicamente de una imagen de la cara, se dice que el sistema identifica; si se le proporciona una imagen de la cara junto con un nombre y el sistema determina si el nombre corresponde a esa cara, se dice que es un sistema verificador.

2.3.2. Representación y clasificación de la cara.

El problema de la representación y el de la clasificación están muy unidos. El primer paso es la selección de las características que definen una cara. Éstas deben de reunir una serie de condiciones:

- Fáciles de estimar.
- Independientes de la iluminación.
- Independientes de cambios en la expresión facial.
- Altamente discriminantes.
- Habitualmente es necesario incluir una etapa de normalización porque las medidas han de ser independientes de la posición, escala y orientación de la cara en escena.

Clasificación de los métodos de reconocimiento facial:

- Basados en características globales[6].
- Basados en características locales distintivas[7].
- Híbridos (combinan ambos tipos de características)[8].

Según las características que se quieran extraer o identificar, así será la representación. Las representaciones más utilizadas son:

- Imágenes como matrices bidimensionales de niveles de gris[9].
- Vectores de características[10].

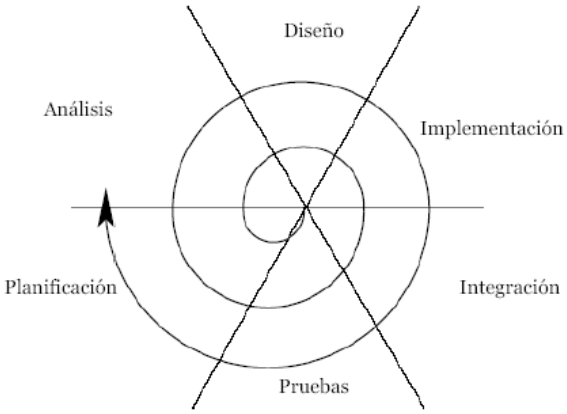
[6]Ejemplos de esta representación son los métodos basados en Análisis de Componentes Principales (PCA) Autocaras, Análisis Lineal Discriminante (LDA) Fisherfaces, redes neuronales, modelos de cara en 3D.

[7]Ejemplos de esta representación son los métodos basados en características geométricas, EBGM. Transformada wavelet (onditas).

[8]Ejemplos de esta representación son los métodos basados en correspondencia de plantillas.

[9]Ejemplos de esta representación son los métodos basados en compactación usando PCA, Fisherfaces y redes neuronales.

[10]Se engloban en esta categoría los trabajos centrados en emparejamiento de plantillas flexibles, jets de Gabor, y demás técnicas.



Cuadro 2.1: Diseño en espiral

2.4. Metodología empleada

El desarrollo del sistema se ha realizado mediante un proceso iterativo en forma de espiral (Ver figura: 2.1), de esta forma en cada iteración hemos ido añadiendo o quitando funcionalidades.

Cada iteración consta de las siguientes partes:

1. *Planificación de la funcionalidad del sistema:* El proceso de planificación ha sido complejo, se puede observar en el gráfico que en esta fase se han ido planteando nuevas funcionalidades para añadir al sistema y también se han ido quitando otras cuando en las otras fases del proyecto se han visto inviables.

2. *Análisis de las funcionalidades:* Se realiza un análisis pormenorizado de cada una de las funcionalidades, la viabilidad, la complejidad, tiempo de realización. Como se puede observar en el gráfico se han ido replanteando los análisis a medida que se han ido planificando nuevas funcionalidades, estas hay que encajarlas dentro de lo que ya está realizado.

3. *Diseño del sistema:* A partir del análisis se realiza el proceso de cómo se han de implementar las funcionalidades de las que consta el sistema,

división en clases, subprogramas, parámetros de subprogramas, relacionar los diversos componentes y darle sentido global a todo esto. Igual que en fases anteriores, al introducir nuevas funcionalidades se ha de integrar lo antiguo con lo nuevo, rediseñando nuevamente el proyecto.

4. *Implementación:* Se ha utilizando el lenguaje de programación Matlab y C++. Se utiliza este lenguaje por la versatilidad que ofrece, la variedad de paquetes y clases predefinidas de las que cuenta y la posibilidad de implementar sin dificultad el diseño realizado en la fase anterior.

5. *Integración del nuevo código al sistema*: Esta fase es muy útil cuando se introducen o quitan funcionalidades en el sistema, ya que en estos casos se hace muy importante integrar el resto del sistema para no alterar su correcto funcionamiento.

6. *Realización de pruebas del sistema:* Se trata de realizar pruebas de todas las casuísticas que puedan aparecer y ver qué tipo de repuestas devuelve el sistema. A partir de estos resultados se realiza un estudio para ver si se han de realizar modificaciones en el proyecto, en caso afirmativo se ha de realizar la planificación de dichas modificaciones y volver a realizar, por consiguiente, todas y cada una de las fases.

Capítulo 3

Descripción informática

3.1. Introducción

A continuación se van a introducir a grandes rasgos los elementos que componen los algoritmos, las librerías utilizadas y desarrolladas, su ordenación y las funciones que las componen.

Para el desarrollo del sistema de reconocimiento facial se ha tomado como base el sistema NRECO. Se trata de un sistema de reconocimiento facial basado en características globales y en métodos reducción de la dimensión[1] que realiza la clasificación mediante el SVM. El sistema permite entrenar un conjunto de modelos correspondientes a cada persona, con las características globales extraídas mediante los métodos de reducción de la dimensión y comprobar la efectividad del reconocimiento de los modelos.

En los siguientes puntos se hará una revisión más en detalle de lo implementado en el proyecto incluyendo las funciones del NRECO empleadas[2]. Se realizará la especificación, diseño e implementación de las funciones más importantes, así como la enumeración del resto. También se intenta hacer

[1] Un método basado en técnicas estadísticas tradicionales (PCA) y tres nuevos métodos adaptados a la estructura bidimensional de las imágenes (2DPCA, 2DLDA y CSA).

[2] Las funciones extraídas del NRECO y que no han sufrido modificaciones importantes en el código se marcarán como [Copia de NRECO] y las que hayan sido modificadas [Basada en NRECO].

un enfoque del proceso de programación, el porqué de las decisiones más importantes que se han tomado y por último mostrar el pseudocódigo de la implementación de las funciones más importantes del código.

3.2. Especificación

El análisis y diseño del sistemas ha sido realizado mediante UML[3]. La primera etapa de la creación del proyecto es la extracción de los requisitos del sistema, tanto los funcionales como los no funcionales. Estos requisitos se dividen según si es el usuario el que interactúa con el sistema o si es el propio sistema el que realiza la función. Veamos los requisitos extraídos en la etapa de análisis del sistema.

3.2.1. Requisitos

Requisitos del Usuario

Fase de entrenamiento

- *Creación de la base de datos:* El usuario debe realizar una base de datos con las imágenes de los individuos que desea que el sistema sea capaz de reconocer.
- *Iniciar el entrenamiento del sistema:* El usuario debe iniciar el módulo de entrenamiento del sistema antes de utilizar la aplicación en modo de análisis de vídeos.
- *Definir los parámetros del entrenamiento:* El sistema dejará al usuario elegir los valores de los parámetros.
- *Definir los parámetros de la gráfica:* El usuario indicará al sistema qué resultados quiere representar y de qué forma.

[3]Lenguaje Unificado de Modelado

- *Analizar los resultados del entrenamiento:* El usuario debe analizar los resultados ofrecidos por el sistema tras el entrenamiento para modificar los parámetros del entrenamiento si fuese necesario.

Fase de análisis

- *Iniciar el análisis del vídeo:* El usuario debe inicializar el módulo de análisis de vídeos para realizar la identificación de los individuos.
- *Definir los parámetros del análisis del vídeo:* El sistema dejará al usuario elegir los valores de los parámetros.
- *Analizar los resultados del análisis del vídeo:* El usuario debe analizar los resultados ofrecidos por el sistema para determinar finalmente que no se trate de un falso positivo o un falso negativo[4].

Requisitos funcionales del Sistema

Fase de Entrenamiento

- *Leer la base de datos:* El sistema debe obtener las imágenes de la Base de Datos proporcionada por el usuario.
- *Pedir los parámetros del entrenamiento:* El sistema debe pedir al usuario todos los parámetros necesarios para realizar el entrenamiento.
- *Seleccionar los individuos y crear los conjuntos:* El sistema debe seleccionar los individuos de la BD con los que trabajar y definir qué imágenes formarán cada conjunto[5].
- *Preprocesado:* El sistema debe realizar un preprocesado de las imágenes: Recorte, ecualización...

[4] Véase la sección 4.3.1
[5] Conjunto de rasgos, entrenamiento y test.

- Extracción de los rasgos: A partir del conjunto de imágenes de rasgos, el sistema debe de obtener los rasgos de una cara: Cara media, Auto valores y Auto vectores, desviación típica...
- *Proyección de las caras:* El sistema debe realizar la proyección de las caras con las que, posteriormente, realizará el entrenamiento.
- *Entrenamiento y Test:* El sistema debe entrenarse a sí mismo para reconocer las caras de la BD y realizar posteriormente un test que indique el error cometido.
- *Representación del error:* El sistema debe representar la gráfica del error en función de los parámetros del usuario.

Fase de Análisis

- *Pedir los parámetros del análisis:* El sistema debe pedir al usuario todos los parámetros necesarios para realizar el análisis.
- *Leer el vídeo:* El sistema debe localizar el vídeo a analizar y obtener las imágenes a procesar.
- *Detectar las caras:* El sistema debe detectar las caras presentes en la escena y recortarlas.
- *Proyectar y Clasificar las caras:* El sistema debe proyectar las imágenes de las caras y clasificarlas según los modelos almacenados tras el entrenamiento.
- *Realizar la votación:* El sistema debe determinar en función de la salida de la clasificación de todas las caras obtenidas de cada individuo si conoce alguno con suficiente certeza. En caso afirmativo comunicárselo al usuario.

Requisitos no funcionales del sistema

Los requisitos no funcionales son aquellas propiedades del sistema como las restricciones del entorno o de implementación, rendimiento, dependencias de la plataforma, mantenibilidad o fiabilidad. Estos requisitos especifican restricciones físicas sobre los requisitos funcionales.

- *Usabilidad:* Deberá de mostrar una interfaz gráfica con el usuario sencilla, de manera que el usuario no experto en reconocimiento facial pueda, intuitivamente, hacer un uso correcto.
- *Eficiencia:* Dado que los algoritmos de reconocimiento facial requieren mucho tiempo de cómputo, se debe, en la medida de lo posible, optimizar el código de las funciones.
- *Fiabilidad:* Deberá ser una aplicación sin errores. Además, la fiabilidad del sistema en cuanto a "Falsos Positivos" y "Falsos Negativos" debe ser ajustable.
- *Portabilidad:* Debido al lenguaje utilizado, Matlab y C++, el sistema podría funcionar con modificaciones mínimas en Windows, Unix y OS2.

3.2.2. Especificación de casos de uso

En este apartado se enumeran los distintos casos de uso que generan los requisitos funcionales, estudiando los papeles que juegan el usuario y el sistema.

Tras el estudio de los requisitos del sistema, aparecen dos fases claramente diferenciadas: La fase de Entrenamiento y la de Análisis de Vídeos.

Fase de Entrenamiento

- *Caso de uso:* Iniciar el entrenamiento del sistema. Ver tabla 3.1 en la página siguiente
  - Caminos alternativos:

Usuario	Sistema
Ejecuta el módulo de entrenamiento	
	Muestra la interfaz de parámetros
	Espera un evento

Cuadro 3.1: Inicio del entrenamiento

Usuario	Sistema
	Muestra la interfaz de parámetros
Carga una configuración	
	Muestra los parámetros cargados

Cuadro 3.2: Cargar una configuración

Usuario	Sistema
	Muestra la interfaz de parámetros
Modifica los parámetros	

Cuadro 3.3: Modificar los parámetros

Usuario	Sistema
	Muestra la interfaz de parámetros
Guardar una configuración	
	Almacena la configuración

Cuadro 3.4: Guardar una configuración

Usuario	Sistema
Inicia el entrenamiento	
	Lee configuración del Ínventario de la BD
	Lee la configuración de los Met. de Red.
	Lee la configuración de los Conjuntos
	Obtiene los Nº de las imágenes de cada Conj.
	Preprocesa las imágenes
	Proyecta las imágenes
	Reconstruye las imágenes
	Entrena el sistema
	Realiza el test del sistema
	Pide los parámetros de las gráficas

Cuadro 3.5: Iniciar el entrenamiento

Usuario	Sistema
	Muestra la interfaz de parámetros
Modifica los parámetros	
	Representa la gráfica

Cuadro 3.6: Definir los parámetros de la gráfica

Usuario	Sistema
Ejecuta el módulo de análisis	
	Muestra la interfaz de parámetros
Modifica los parámetros	
Inicia el análisis	

Cuadro 3.7: Inicio del análisis de un vídeo

  - Evento 1: Cargar una configuración. Ver tabla 3.2 en la página anterior
  - Evento 2: Modificar los parámetros. Ver tabla 3.3 en la página anterior
  - Evento 3: Guardar la configuración. Ver tabla 3.4 en la página anterior

- *Caso de uso:* Iniciar el entrenamiento. Ver tabla 3.5 en la página anterior
- *Caso de uso:* Definir los parámetros de la gráfica. Ver tabla 3.6

Fase de Análisis de Vídeos

- *Caso de uso:* Iniciar el análisis de un vídeo. Ver tabla 3.7
- *Caso de uso:* Analizar un vídeo. Ver tabla 3.8 en la página siguiente

3.3. Diseño

En el diseño damos forma a la aplicación para que soporte los requisitos, incluyendo los no funcionales que hasta ahora no se habían tenido en cuenta.

Usuario	Sistema
Inicia el análisis	
	Muestra la interfaz de análisis
	Lee el vídeo
	Detecta las caras
	Proyecta y clasifica las caras
	Realiza la votación
	Muestra los resultados

Cuadro 3.8: Analizar un vídeo

En esta fase se adquiere una comprensión de los aspectos relacionados con los requisitos y restricciones que tienen que ver con el lenguaje de programación, componentes reutilizables, sistemas operativos... Se creará un punto de partida para las actividades de implementación capturando requisitos individuales, interfaces y clases.

3.3.1. Diagramas de estados

Los diagramas de estado nos muestran el conjunto de estados por los que pasa la aplicación durante su ejecución, junto con los cambios que permiten pasar de un estado a otro. En él se indican qué funciones hacen que se desencadene uno u otro estado y cuales son las acciones y respuestas que lo generan. Se pueden ver en las figuras 3.1 en la página siguiente y 3.2 en la página 24.

3.3.2. Diagrama de casos de uso

Un caso de uso es una descripción de la secuencia de iteraciones que se producen entre un actor y el sistema cuando el actor usa al sistema para llevar a cabo una tarea específica. Expresa una unidad coherente de funcionalidad y se representa en el Diagrama de Casos de Uso mediante una elipse con su nombre en el interior.

Debido a la sencillez que se ha perseguido que tuviera el sistema desarrollado, el usuario solo puede realizar cinco casos de uso. Iniciar la fase de

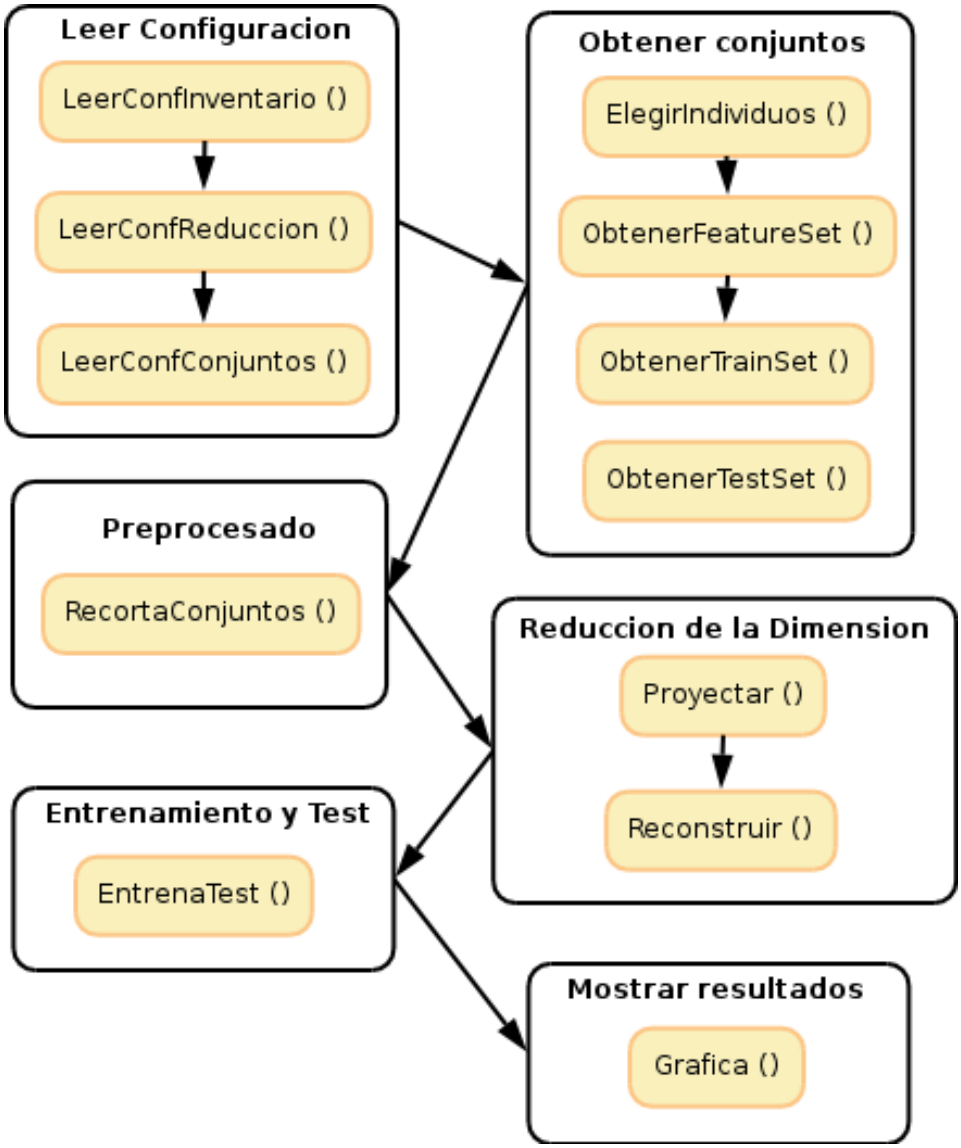


Figura 3.1: Diagrama de estados: Entrenamiento

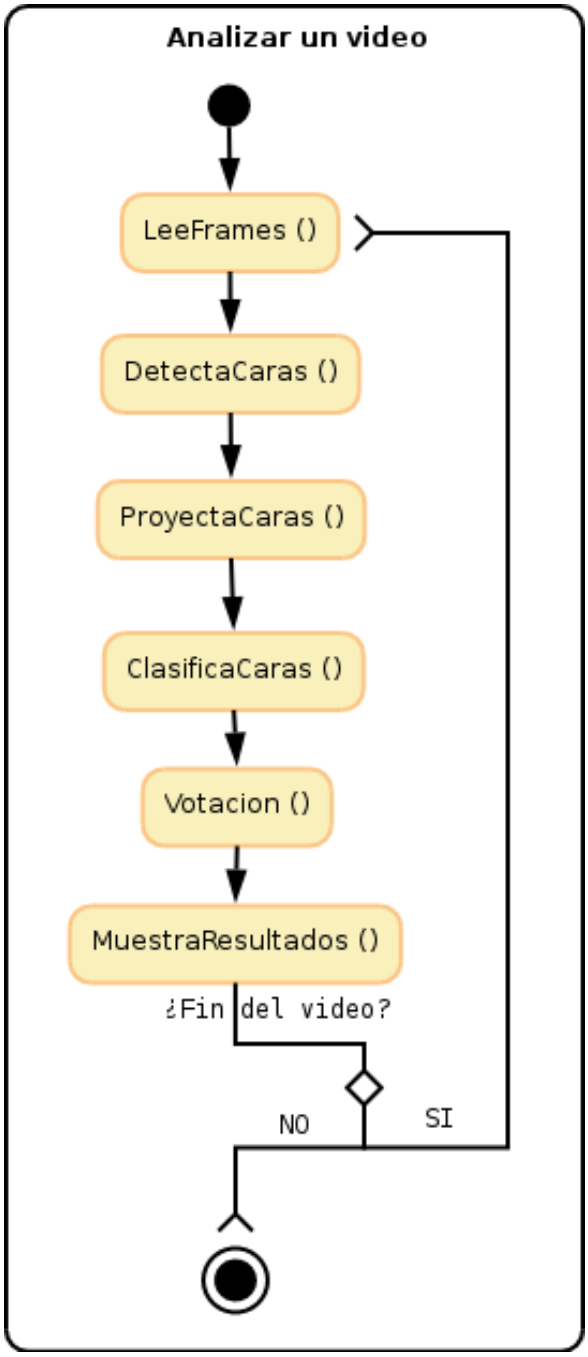


Figura 3.2: Diagrama de estados: Análisis

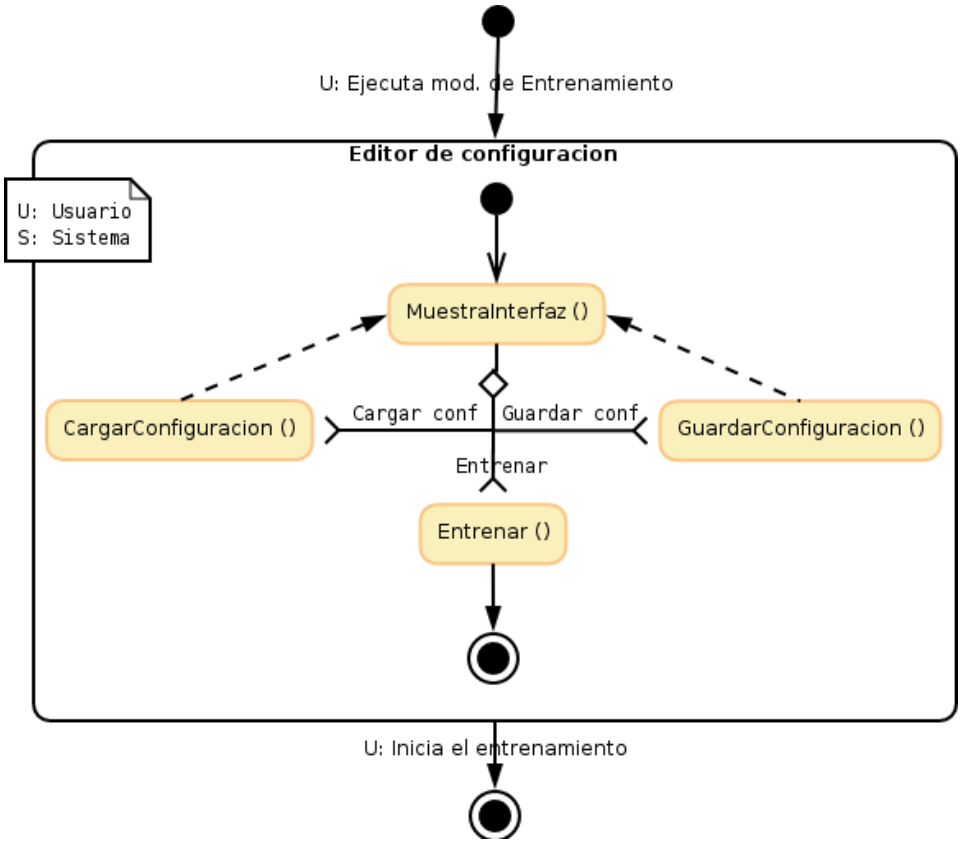


Figura 3.3: Caso de Uso: Iniciar módulo de entrenamiento

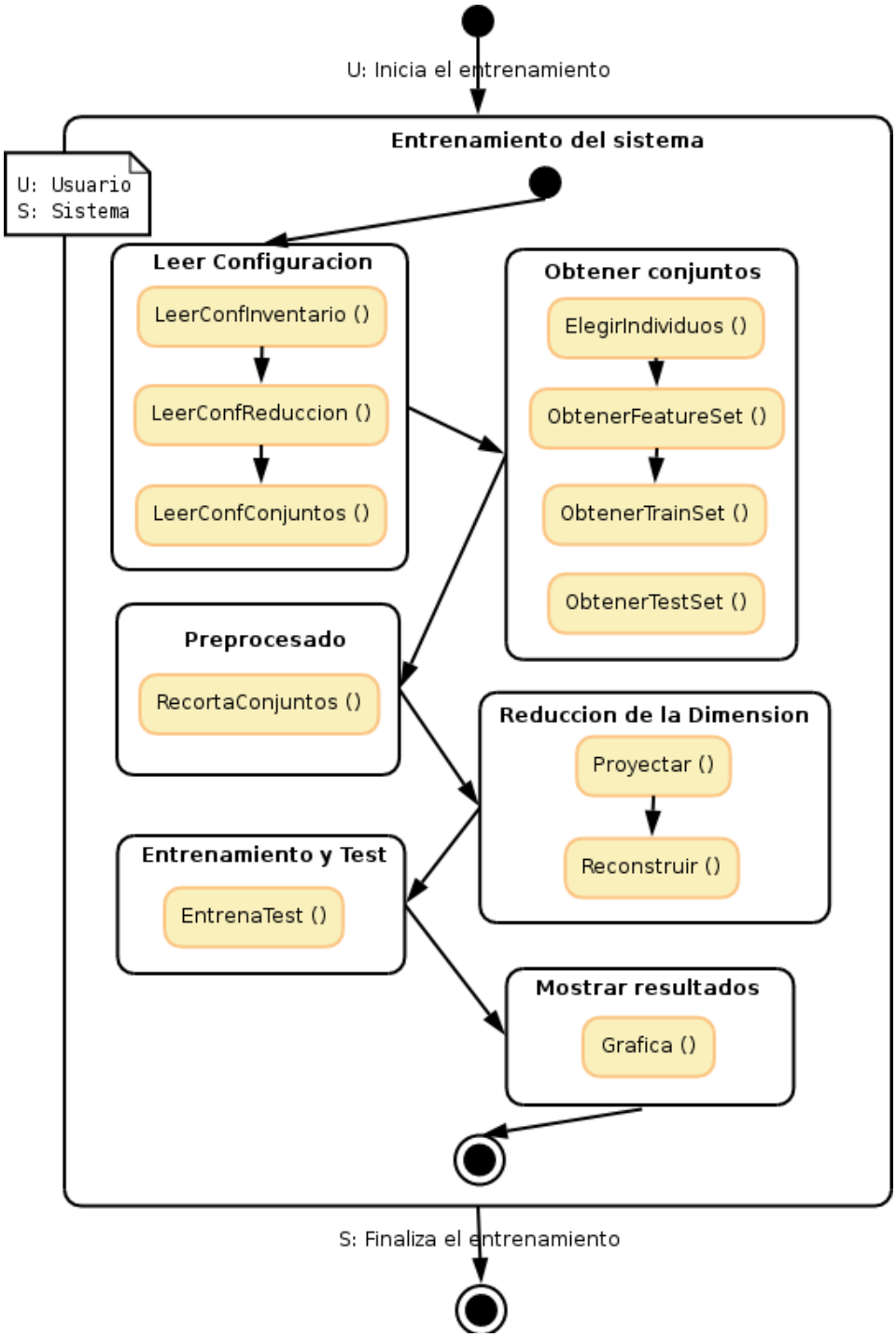


Figura 3.4: Caso de uso: Iniciar entrenamiento

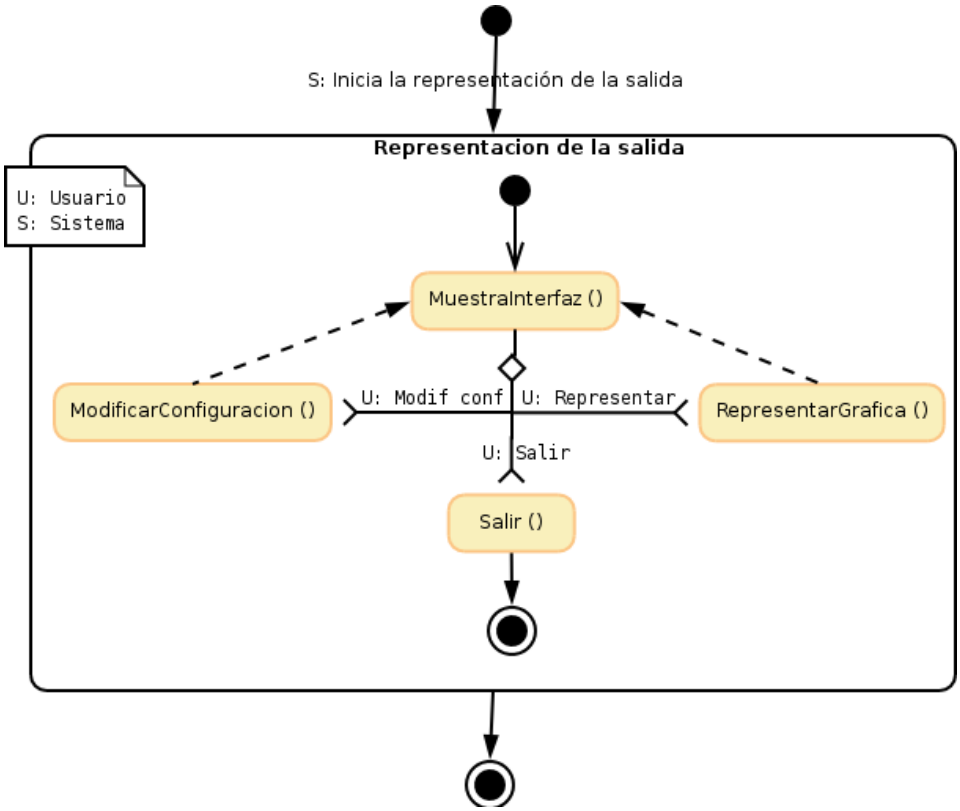


Figura 3.5: Caso de uso: Definir parámetros de la gráfica

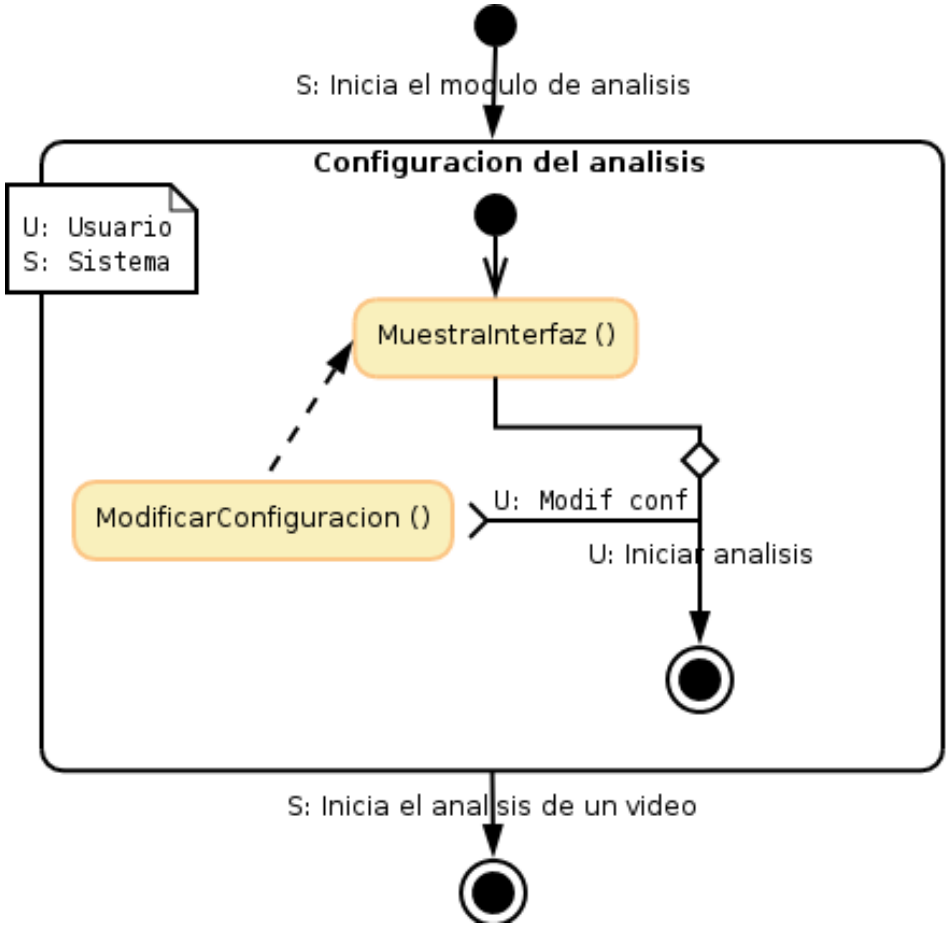


Figura 3.6: Caso de uso: Iniciar módulo de análisis

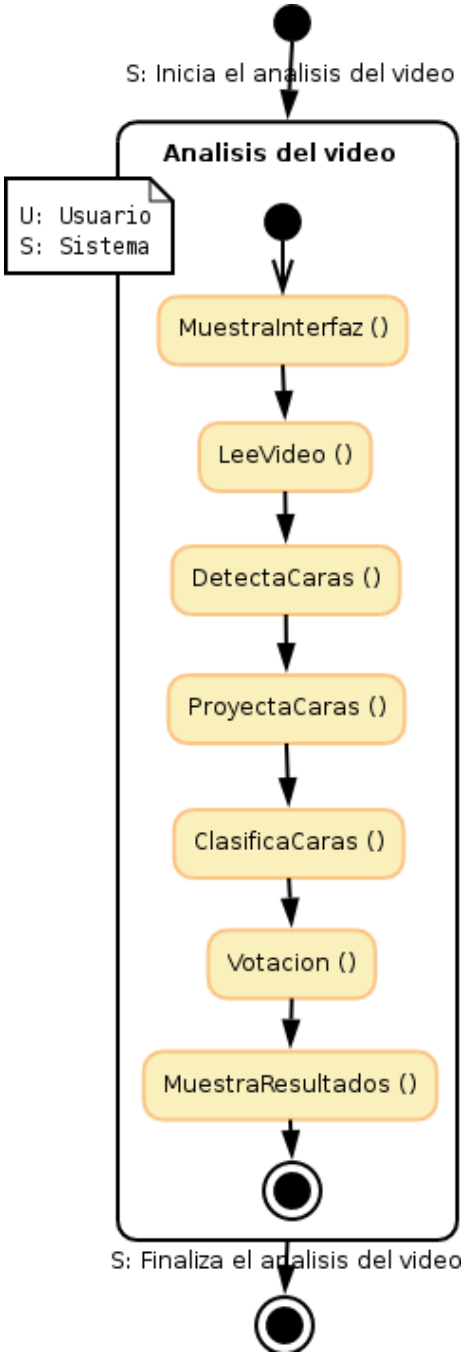


Figura 3.7: Caso de uso: Iniciar el análisis de un vídeo

entrenamiento (Figura: 3.3 en la página 25), entrenar el sistema (Figura: 3.4 en la página 26), definir los parámetros de representación (Figura: 3.5 en la página 27), iniciar la fase de análisis (Figura: 3.6 en la página 28) y analizar un vídeo (Figura: 3.7 en la página anterior).

3.4. Implementación

La fase de implementación es donde el sistema se ha desarrollado como ficheros en código fuente, scripts, ficheros ejecutables, etc... Esta parte se encarga de escribir la aplicación en una sucesión de pasos pequeños y manejables. Implementar las funciones y subfunciones encontradas en la fase de diseño. En el Anexo C en la página 68 se puede encontrar una descripción de las funciones implementadas.

A continuación se indica qué algoritmos se han empleado para resolver los principales problemas del sistema.

3.4.1. Detección de imágenes en la escena

El algoritmo utilizado se basa en el método de Viola-Jones de deteccion automática de cara, en nuestro caso hemos utilizado la implementación Torch. Se trata de un algoritmo robusto que proporciona muy buenos resultados incluso con imágenes de baja calidad.

3.4.2. Preprocesado de imágenes

Interpolación de las imágenes

Mediante interpolación se adecúan las imágenes de las caras al tamaño de las fotos de las de la base de datos. El algoritmo empleado es la Interpolación bicúbica puesto que es el método de interpolación considerado estándar

(promedia 16 píxeles adyacentes). Es importante tener en cuenta que la interpolación nunca conseguiría la misma calidad que una fotografía realizada al mismo tamaño, ya que la información añadida es "inventada".

Escala de grises

Puesto que los algoritmos de reconocimiento facial trabajan con imágenes en blanco y negro y las imágenes que queremos emplear son en RGB, hemos empleado un algoritmo de conversión de imágenes de RGB a Escala de grises.

Ecualización del histograma

La ecualización del histograma permite uniformizar los niveles de gris de la imagen, es decir, que en imágenes que están muy concentradas en determinados niveles de gris aumente el contraste. La ecualización maximiza la Entropía (información) de la imagen.

3.4.3. Métodos de reducción de la dimensionalidad

Una de las características del tratamiento de imagen es la gran información de la que se dispone. Para manipular y procesar toda esta información, es necesario utilizar alguna técnica estadística de síntesis de la información, o reducción de la dimensión (número de variables). Es decir, ante un banco de datos con muchas variables, el objetivo será reducirlas a un menor número perdiendo la menor cantidad de información posible. En el transcurso de este proyecto se han utilizado diferentes técnicas de reducción de la dimensión que se describen someramente a continuación:

- PCA (Análisis de Componentes Principales): trata de hallar componentes (factores) que sucesivamente expliquen la mayor parte de la varianza total. Para ello se considera una imagen de MxN píxeles, como un vector de MxN componentes. Cada componente viene dado por la intensidad de cada uno de los píxeles.

- 2DPCA (Análisis de Componentes Principales Bidimensional): se trata de una técnica semejante a PCA, pero donde se tiene en cuenta la vecindad de cada píxel dentro de la imagen, tratando una imagen como una matriz, y realizando el análisis en dos dimensiones.

- 2DLDA (Análisis Lineal Discriminante Bidimensional): técnica estadística en la que se pretende encontrar las variables que mejor discriminen o distingan a elementos de clases diferentes. Para ello se tiene en cuenta también la información de vecindad de los píxeles, con una imagen como matriz, realizando un tratamiento en dos dimensiones.

- CSA (Coupled Subspace Analysis): generalización de las técnicas anteriores. Mediante este método se infieren dos subespacios acoplados de baja dimensión que reconstruyen de manera óptima las matrices imagen en las direcciones fila y columna.

3.4.4. Clasificador SVM

Las redes de neuronas artificiales[6] son un paradigma de aprendizaje y procesamiento automático inspirado en la forma en que funciona el sistema nervioso de los animales. Consiste en simular las propiedades observadas en los sistemas neuronales biológicos a través de modelos matemáticos recreados mediante mecanismos artificiales[7]. El objetivo es conseguir que las máquinas den respuestas similares a las que es capaz el cerebro que se caracterizan por su generalización y su robustez.

La teoría del aprendizaje estadístico, ha impulsado el desarrollo de técnicas de aprendizaje que maximizan las dos principales capacidades en cualquier método de aprendizaje: aprendizaje y generalización. El aprendizaje en las redes neuronales está basado, en general, en la minimización del error, lo que no asegura, por sí sólo, la maximización de sus capacidades de generalización, aunque existen diferentes técnicas que abordan este problema de diferentes

[6] RNA

[7] como un circuito integrado, un ordenador ver figura 3.8 en la página siguiente

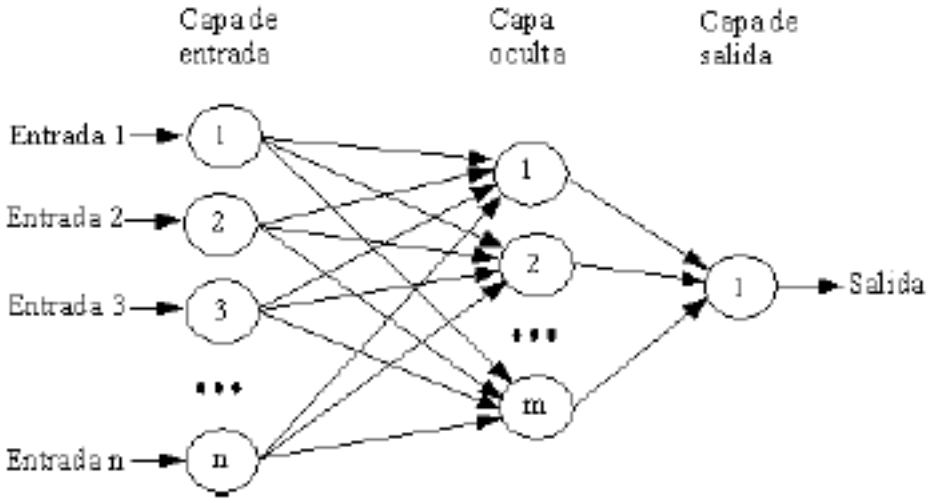


Figura 3.8: Estructura de una red neuronal

maneras[8], pero sin una demostración matemática formal. Las máquinas de vector soporte[9] son el principal método de aprendizaje basado en la teoría del aprendizaje estadístico, bajo una arquitectura de red neuronal. Este método aporta mejoras a los métodos clásicos de aprendizaje: la talla de la red no se fija desde un principio y se garantiza, matemáticamente, el máximo nivel de generalización.

3.4.5. Post-procesado: Votación

En una secuencia de vídeo se pueden obtener varias imágenes de un mismo individuo. Sabiendo de antemano qué imágenes pertenecen al mismo individuo, se pueden presentar todas juntas al sistema y a partir de las salidas individuales obtenidas el sistema puede ofrecer un respuesta más fiable que con una única imagen.

[8] heurísticas
[9] support vector machines, SVM

Capítulo 4

Pruebas y resultados

4.1. Pasos previos

4.1.1. Elección de las cámaras

En una serie de visitas iniciales al aeropuerto de Barajas (6, 13 y 20 abril, 13 y 26 mayo y 1 y 2 de junio de 2005) se observó el entorno en el cual se iba a desarrollar el proyecto, a la vez que se adquirían las imágenes de prueba de los diferentes algoritmos .

En estas visitas iniciales se comprueba que las cámaras del aeropuerto graban en formato analógico (VHS) desde el centro de cámaras en las instalaciones de AENA. Se consideró que esta es la mejor forma de adquirir las imágenes ya que se utiliza el sistema sin interferir o añadir ningún elemento. Además se observó que hay cámaras con una calidad de imagen muy buena, con un potente zoom. Hay otras cámaras, más antiguas, en las que la calidad es bastante peor. Debido a la naturaleza del aeropuerto, la iluminación no está controlada, apareciendo iluminación natural, reflejos y brillos en la imagen. En los arcos detectores de metales, la iluminación es más uniforme, lo cual a priori nos permite suponer que los resultados de verificación facial en estas condiciones serán mejores.

En el presente proyecto se han utilizado las cámaras ya instaladas, por lo

tanto las imágenes obtenidas son en el rango del espectro visible. Se decidió que la cámara se mantuviera estática[1] pero aumentando el zoom para que la cara ocupara la mayor parte de la imagen. Las cámaras no enfocan a los sujetos de frente y la posición de las mismas no es uniforme, se encuentran situadas a alturas muy variables. La implantación de un sistema de verificación o reconocimiento facial por una parte supondría la colocación de cámaras dedicadas a la verificación[2] y también un coste computacional[3]. Las cámaras tienen ganancia automática.

- Experimentos propuestos y cámaras seleccionadas
- Cinta 1 y Cinta 2:
- Cinta

Experimento 1

En este experimento se pretende aplicar un sistema de verificación en un sujeto cuando es observado secuencialmente por dos cámaras distintas. Para ello se pretende que los sujetos deban pasar primero por una cámara y luego por la otra. Las cámaras que se consideraron son las que se encuentra en zonas como puede ser un pasillo.

Debido a las condiciones de iluminación, posición y calidad de las imágenes, la selección final es la secuencia con las cámaras 2127 - 79. De estas dos cámaras la 79 es de bastante peor calidad que la 2127. Estos vídeos son los que se designan como Cinta 1 (cámara 79) y Cinta 2 (cámara 2127).

Experimento 2

En este experimento se pretende verificar la identidad de un sujeto. Se decidió seleccionar una cámara en un arco detector de metales. Se comparan las imágenes de laboratorio[4] con las imágenes aquí obtenidas para el mismo sujeto.

[1] Sin seguir a ningún sujeto
[2] Por ejemplo en los controles de los arcos
[3] En equipos informáticos dedicados exclusivamente a este proceso
[4]

Nombre de la base de datos	Nº individuos	Nº de fotos / individuo	Ejemplos
Arcos	46	10	Figura4.1
Cinta 1	94	10	Figura4.2
Cinta 2	183	10	Figura4.3
Frav2d	109	32	Figura4.4

Cuadro 4.1: Características de las Bases de Datos Obtenidas.

Debido a la calidad de la imagen y a otras consideraciones, las cámaras seleccionadas para este experimento son las 131 y 2154[5]. Este vídeo es el que se designa como Arcos.

Experimento 3

Este experimento se pensó como complementario al 1. Se buscaba realizar el experimento 1 pero con dos cámaras enfocando al mismo sujeto simultáneamente (en el experimento 1 el sujeto aparece secuencialmente primero en una cámara y luego en la siguiente).Este experimento fue descartado ya que no es un proceso que se realice por parte del servicio de vigilancia y además la disposición de las cámaras no esta diseñada para que sea utilizada de esta forma. Las cámaras probadas en este experimento fueron la 60 y la 2022.

4.1.2. Elaboración de las bases de datos

La Tabla 4.1 resume las bases de datos que se han obtenido. La primera corresponde a la grabación en los arcos de seguridad, la segunda y la tercera a la cinta 1 y a la 2, la cuarta es una base de datos de imágenes de laboratorio obtenida por el equipo de investigación[6]. Puede observarse a continuación la secuencia de imágenes para un sujeto, observándose la distinta calidad de las imágenes.

Puede observarse en las Figuras 4.1,4.2,4.3 y 4.4 que la calidad de las imáge-

En este caso, de la Base de datos FRAV2D

[5]El arco en el cual la imagen es más frontal
[6]http://frav.escet.urjc.es

Figura 4.1: Imágenes de un sujeto en la BD Arcos

Figura 4.2: Imágenes del mismo sujeto en la BD Cinta 1

Figura 4.3: Imágenes del mismo sujeto en la BD Cinta 2

Figura 4.4: Imágenes del mismo sujeto en la BD FRAV2D

nes es muy variada, desde una alta calidad obtenida en el laboratorio a una calidad bastante baja en los vídeos de Barajas. Otros factores que se observan es la variabilidad en la iluminación y en la posición de la cara. En el laboratorio la cara es casi frontal y en las imágenes obtenidas en el aeropuerto el sujeto puede girar libremente su cara.

4.2. Pruebas de detección

A continuación, en las Figuras 4.5, 4.6 y 4.7 se muestran ejemplos de recorte válido.

Como resultado de aplicar el método de localización, de la cámara enfocada en la cinta 1 se obtienen imágenes de peor calidad que las de la cinta 2 y los arcos. Además, la iluminación no es adecuada para un proceso de verificación o reconocimiento. Sin embargo, el algoritmo probado es capaz de localizar correctamente la cara en un porcentaje grande de casos, incluso en algunas situaciones tan adversas cómo las mostradas en la Figura 4.8.

Este método de recorte ha sido seleccionado para localizar y recortar las caras en los diferentes vídeos. Sin embargo, en algunos casos se han observado detecciones incorrectas, como puede verse en la Figura 4.9. Hay un grupo de ellas que corresponden a aquellas zonas de una imagen que se asemejan a una

Figura 4.5: BD Arcos. Situaciones de localización normal

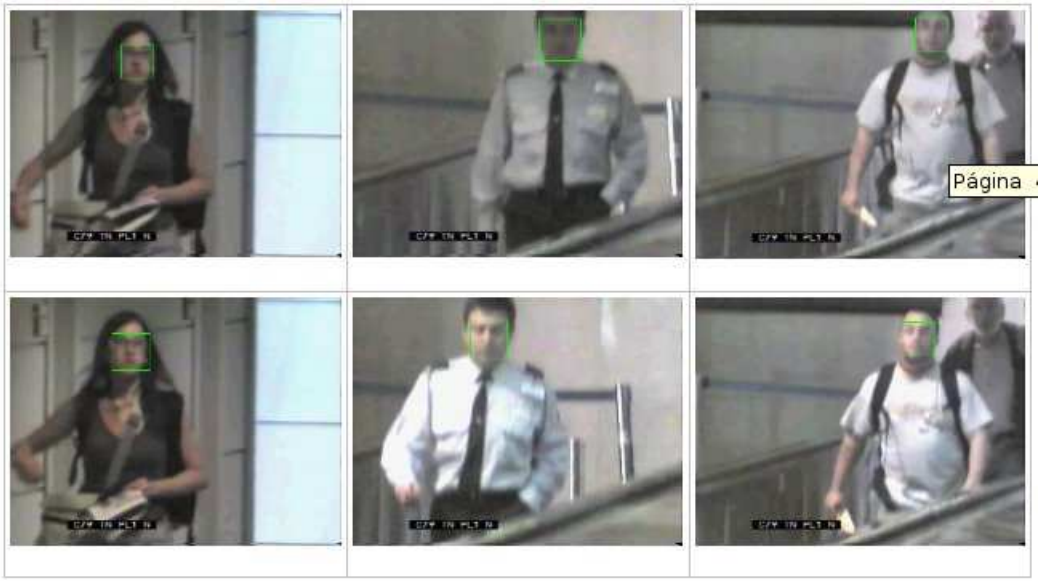


Figura 4.6: BD Cinta 1. Situaciones de localización normal

Figura 4.7: BD Cinta 2. Situaciones de localización normal

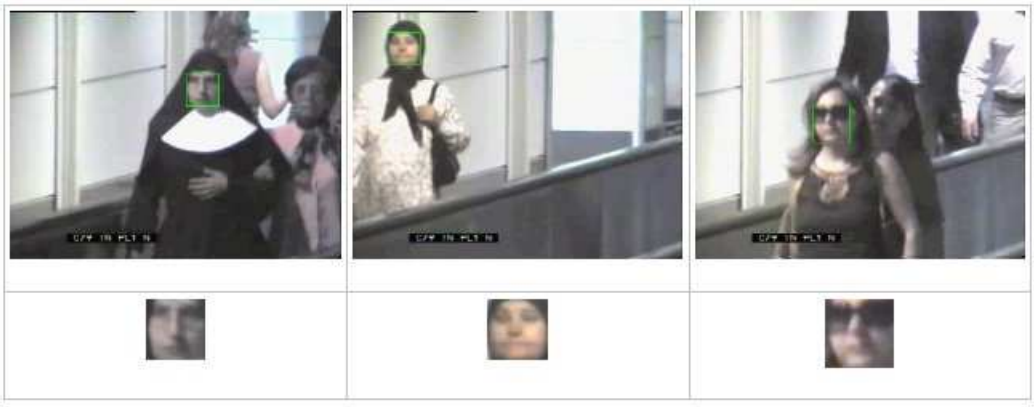


Figura 4.8: BD Cinta1. Situaciones de localización adversas

Figura 4.9: BD Cinta2 Ejemplos de fallos razonables

cara pero que en realidad son sombras, arrugas de una tela o a que debido a la perspectiva de la cámara se funden varias caras (normalmente dos) en una sola.

Otras veces el algoritmo reconoce como caras zonas que no tienen ninguna relación con el rostro humano, se muestran algunos ejemplos en la Figura 4.10. Esto se produce casi de forma anecdótica no llegando al 1 % de los casos. Hemos de tener presente que, al estar trabajando con secuencias de vídeo, los fallos de localización en una imagen pueden recuperarse con localizaciones correctas en las imágenes siguientes.

Empíricamente se ha observado que el algoritmo suele fallar en las siguientes situaciones:

- El sujeto se encuentra de espaldas o de lado
- Parte de la cara está oculta: Lleva gafas de sol o la cabeza cubierta, la cara aparece tapada por la mano, va hablando por el móvil, o bien sencillamente aparece detrás de otra persona
- Mantiene un giro muy grande de la cabeza[7]
- El sujeto se encuentra lejos de la cámara

[7] mirando al suelo o hacia los lados

Figura 4.10: Ejemplos de fallos exagerados

Porcentaje de aciertos

Para conocer cuál es el porcentaje de caras detectadas por el algoritmo, se contaron el número de personas a las que se les ve la cara con un tamaño mínimo para poder ser verificadas. Después se realizó una clasificación manual de los rostros detectados.

BD	Duración	Nº Ind	Nº Ind. Detect	% Loc. correcta
Arcos	20 min	34	34	100 %
Cinta1	26 min	250	137	54 %
Cinta2	20 min	238	158	66 %
Frav2D				100 %

Cuadro 4.2: Resultados de localización.

Como podemos observar en la Tabla 4.2, el porcentaje de caras detectadas depende directamente de la calidad de las imágenes. La cara debe tener un tamaño mínimo de 38x38. Las caras con un tamaño menor no se han tenido en cuenta en el proceso de localización. En las cintas 1 y 2 se ve a la gente desde lejos, pero solo cuando la persona se encuentra muy próxima a la cámara se tiene en cuenta. Esto provoca que la probabilidad de detección sea menor.

En cambio en la cinta de Arcos, se localiza el 100 % de las caras, pues éstas se encuentran en primer plano.

Otro parámetro de la localización de la cara que nos interesaba medir era el número de imágenes que se puede obtener de cada sujeto. En las Tablas 4.3 y 4.4 se muestra el número de individuos a los que se les localizaba un número de imágenes dado.

Nº de Img. Detect.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	+15
Nº de Ind.	2	2	1	1	0	2	0	1	2	2	0	0	2	4	5	9

Cuadro 4.3: BD Arcos. Número de imágenes obtenidas de cada individuo (Analizando los frames pares únicamente)

Nº de Img. Detect.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	+15
Nº de Ind.	33	18	12	5	5	9	3	4	6	5	3	4	4	6	8	33

Cuadro 4.4: BD Cinta2. Número de imágenes obtenidas de cada individuo (Analizando los frames pares únicamente)

En las tablas puede observarse que se dispone de un número suficiente de sujetos con varias imágenes, lo cual permite verificar varias veces a un mismo sujeto. En promedio se analizaron 20 minutos de vídeo.

4.3. Pruebas de identificación

4.3.1. Introducción: Curvas ROC

La evaluación de la calidad de un sistema de verificación requiere de un detallado análisis de los posibles aciertos y fallos del sistema. Un sistema de verificación tiene como salida dos valores: aceptación o rechazo. Una salida con valor “acepto” indicará que efectivamente la imagen introducida y la persona que se reclama se corresponden. En caso contrario indicaría que se trata de un impostor, y la salida del sistema sería “rechazo”. Si se repitiera

esta prueba para un conjunto de individuos, se obtendrían unas distribuciones de impostores y de auténticos.

Existen por lo tanto cuatro posibilidades:

- Una persona se identifica correctamente ante el sistema y éste le acepta: verdadero positivo (TP, de "True Positive").
- Una persona se identifica correctamente ante el sistema, pero este le rechaza: falso positivo (FN, de "False Negative").
- Un impostor se identifica con la identidad de otro y el sistema acepta: Falso Positivo (FP, de "False Positive").
- Un impostor se identifica con la identidad de otro y el sistema le rechaza: Falso Negativo (TN, de "True Negative").

Es decir, las salidas del sistema pueden ser aciertos (a los que llamaremos como Verdaderos positivos o negativos) o pueden ser fallos (serán los Falsos positivos y negativos). En todo sistema de verificación obviamente se pretenden minimizar los Falsos Positivos y los Falsos Negativos.

Para determinar las prestaciones del sistema biométrico, nos remitiremos al análisis y valoración de los siguientes parámetros estándares:

- FAR (False Acceptance Rate): Porcentaje de personas no autorizadas aceptadas por el sistema.
- FRR (False Reject Rate): Porcentaje de personas autorizadas no aceptadas por el sistema.
- SR (Success Rate): Responde a una combinación de los dos factores anteriores que se utiliza como indicador de la resolución total del sistema. $SR = 1 - (FAR + FRR)$
- ERR (Equal Error Rate): El FAR y el FRR responden a parámetros inversamente proporcionales, por tanto, variarán en función de

	Entrenamiento		Prueba			EER			
Figura	BD	Nº Img	BD	Nº Img	Nº Ind	1dpca	2dpca	2dlda	csa
4.11	Frav2D	5	Arcos	5	6	39.44	39.44	40	42.22
4.12	Frav2D	5	Cinta 1	5	4	45.45	26.70	45.37	25
4.13	Frav2D	5	Cinta 2	5	5	27.00	30.64	47.50	33.33
4.14	Arcos	5	Cinta 1	5	3	32.50	33.75	33.33	34.00
4.15	Cinta 1	5	Cinta 2	5	4	28	28.66	36	29.33

Cuadro 4.5: Resultados obtenidos con los diferentes métodos

las condiciones prefijadas por el programa de identificación biométrica. Así, si por ejemplo debemos utilizar el programa en un entorno de máxima seguridad, intentaremos que el FAR sea el más pequeño posible, aunque esta acción signifique de forma implícita, el incremento drástico del factor FRR. Deberemos fijar un parámetro o umbral que nos permita igualar los dos factores, asegurando de esta manera el óptimo funcionamiento del sistema. Este umbral se denomina Equal Error Rate (ERR), y es el que determinará, finalmente, el poder de identificación del sistema. El la figura 1, mostramos gráficamente la relación descrita.

4.3.2. Experimentos realizados

Se realizaron cinco experimentos. En los tres primeros se entreno el sistema con imágenes de laboratorio y se probó con imágenes de Barajas. En los dos siguientes se entrenó y se probó con imágenes obtenidas en Barajas. El tamaño de las imágenes es de 50x50 píxeles[8]. Las pruebas realizadas se resumen en la tabla 4.5, junto con el EER (Equal Error Rate). Los experimentos realizados han sido los siguientes:

[8]Las imágenes de laboratorio tienen un tamaño de unos 300x300 píxeles

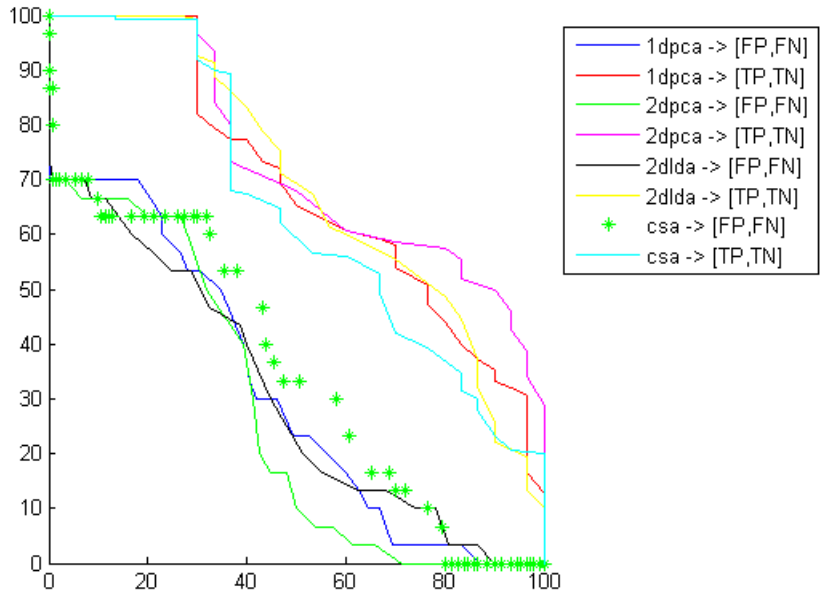


Figura 4.11: Resultado de entrenar con FRAV2D y probar con Arcos

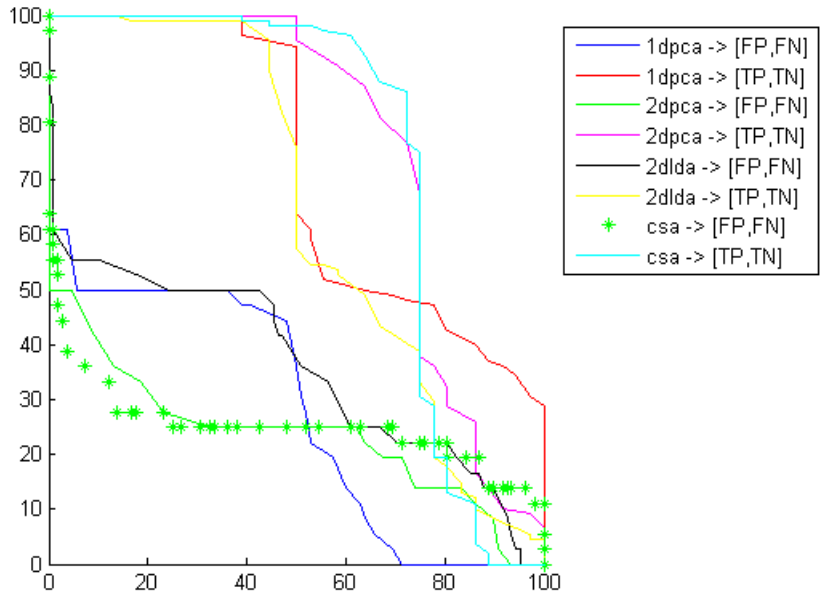


Figura 4.12: Resultado de entrenar con FRAV2D y probar con Cinta 1

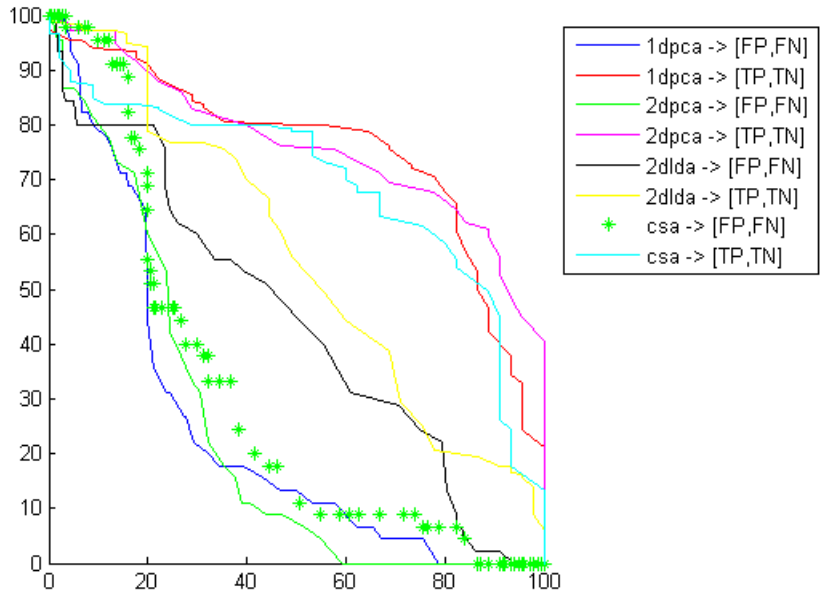


Figura 4.13: Resultado de entrenar con FRAV2D y probar con Cinta 2

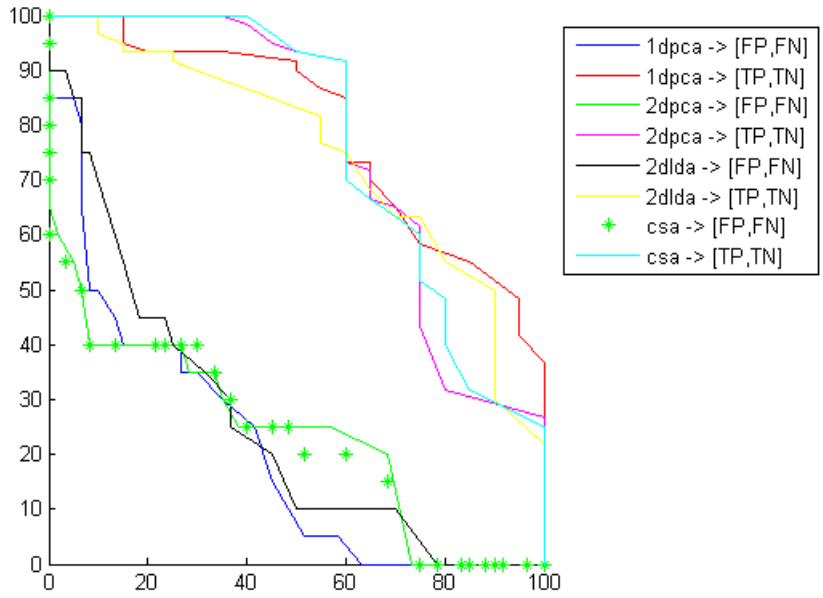


Figura 4.14: Resultado de entrenar con Arcos y probar con Cinta 1

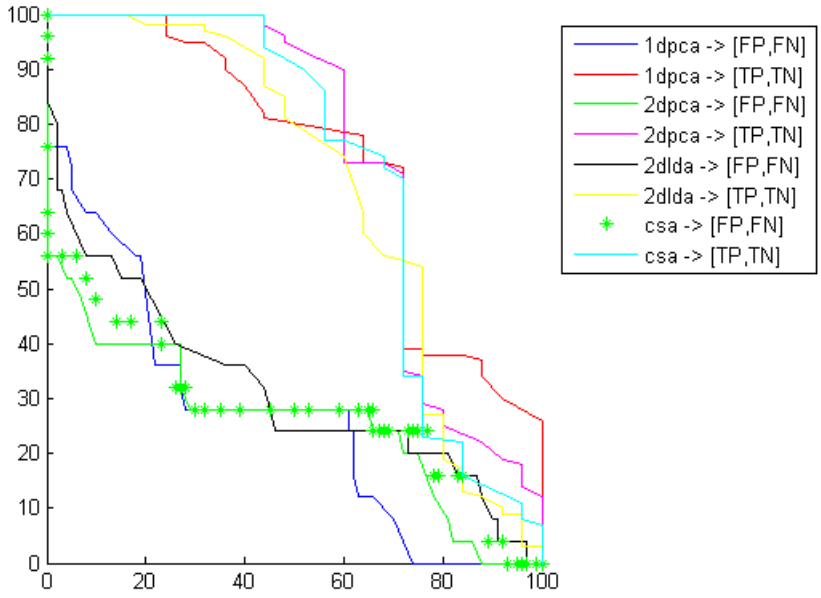


Figura 4.15: Resultado de entrenar con Cinta 1 y probar con Cinta 2

Capítulo 5

Conclusiones

5.1. Evaluación global del sistema

Puede observarse que en la localización de la cara en un entorno real, el método de Viola-Jones proporciona unos resultados muy buenos incluso con imágenes de baja resolución como son los vídeos de Barajas.

En cuanto a los experimentos de verificación, los resultados se han realizado con una base de datos bastante pequeña, debido sobre todo a que el emparejamiento en el entrenamiento entre las imágenes del mismo sujeto en las distintas cintas ha de ser hecho manualmente. Una vez recortadas las imágenes y reducida su dimensión, este emparejamiento es muy complejo.

Antes de proceder al análisis de los resultados de verificación, conviene indicar que el grupo de investigación, utilizando las imágenes FRAV2D[1], obtiene unos porcentajes de EER como los mostrados en la Tabla 5.1. Para un sistema comercial, como es Face-intellect[2], la empresa afirma que obtiene un 22,6 % de identificaciones falsas.

La tasa de error obtenida en Barajas es bastante alta frente a los experimentos de laboratorio, pero hemos de tener en cuenta que las condiciones de trabajo

[1] Obtenidas en condiciones de laboratorio
[2] Disponible en http://www.itvsec.com

EER		
Conjunto de prueba	PCA	2DPCA
Imágenes frontales	1.8	1.0
Iluminación cenital	6.6	2.1
Rotación 15 grados	27.3	18.3
Imágenes con gesto	15.8	12.9
Oclusión en la cara	34.7	29.5

Cuadro 5.1: EER con FRAV2D

son muy diferentes. En los experimentos realizados para el presente proyecto se han utilizado dos cámaras que ofrecen una calidad de imagen aceptable y una que ofrece una calidad muy pobre. Los resultados muestran que, en general, cuando se incluyen las imágenes de baja calidad provenientes de cinta 1 los resultados empeoran.

Los métodos considerados de verificación facial, desarrollados por el equipo de investigación ofrecen resultados muy esperanzadores al compararlos con los del sistema comercial Face-intellect. El sistema Face-intellect sólo funciona en interiores, con cámaras dedicadas y de alta calidad, así mismo se insiste en que el sistema Face intellect sólo funciona en condiciones de iluminación adecuadas. En nuestros experimentos nada de esto ha sido posible, ya que las cámaras no son de alta calidad y las condiciones de iluminación son las habituales de un aeropuerto, muy diferentes a las ideales necesarias incluso para los sistemas comerciales.

5.2. Conclusiones

El presente estudio ha permitido probar un sistema de verificación facial en el aeropuerto de Barajas. Se han abordado todas las fases del mismo: selección de las cámaras, adquisición de vídeos, digitalización de los mismos, localización y recorte de las caras, creación de las bases de datos y pruebas de verificación.

Los resultados obtenidos son prometedores, comparables con los obtenidos por sistemas comerciales, teniendo en cuenta que las condiciones de adquisi-

ción no son las más favorables. El sistema utilizado ha buscado utilizar las cámaras instaladas en Barajas, sin añadir ningún tipo de elemento adicional o interferir con el funcionamiento habitual. Esto ha limitado tanto la posición y orientación de las cámaras como la calidad de la imagen.

Aunque las bases de datos utilizadas son de un número de sujetos reducido, permiten mostrar la validez de la aproximación considerada. Mostrando que un sistema de verificación facial puede ser una ayuda a la decisión. Las pruebas realizadas indican que es posible el procesamiento de imágenes en tiempo real, a pesar de no ser uno de los objetivos del proyecto.

Uno de los objetivos alcanzados ha sido un estudio de las diferentes cámaras disponibles en Barajas y su posible utilización para un sistema de biometría facial. Las cámaras más modernas ofrecen una calidad aceptable, las más antiguas tienen una calidad muy pobre para su uso por un sistema de verificación facial. Además, una de las ventajas de un sistema basado en cámaras de vídeo-vigilancia es que en ningún momento los sujetos se han sentido estudiados.

La apertura de la Terminal T4 de Barajas, donde los sistemas de visión son ya digitales permitirá mejorar las condiciones de adquisición de imágenes y obtener mejores resultados.

5.3. Posibles trabajos futuros

Probar el sistema con vídeos digitales de buena calidad.

Puesto que se ha llegado a la conclusión que el sistemas de cámaras de Barajas no tiene suficiente calidad para realizar una identificación eficiente de los individuos, se sugiere probar el sistema con vídeos digitales de mayor calidad.

Probar el sistema implementado para reconocer otro tipo de objetos

Puesto que el sistema de detección de caras (Viola-Jones) y el de reconocimiento de caras (NRECO) se pueden entrenar para detectar y reconocer cualquier tipo de objetos, se podría emplear para cualquier otro uso. Son ejemplos de éstos:

- Control de calidad
- Ordenación por calidades (grading)
- Detección de desperfectos en superficies metálicas
- Aplicaciones bio-médicas
- Agroalimentarias
- Regulaciones semafóricas...

Mejoras del sistema

- Empleo de nuevos algoritmos de reconocimiento facial basados en características locales.
- Implementar módulo de test mediante votación.
- Implementar módulo de creación automática de bases de datos.
- Implementar la posibilidad de añadir una nueva cara tras el entrenamiento.
- Permitir la posibilidad de la creación de varios modelos diferentes para un mismo sujeto. Por ejemplo un modelo con imágenes frontales y otro con imágenes giradas.

Bibliografía

[1] Tapiador, M. Y Siguenza, J.A. Tecnologías biométricas aplicadas a la seguridad. Editorial Ra-Ma. Marzo 2005

[2] http://www.tecnociencia.es/monograficos/biometria/biometria1.html

[3] Morales, Cesar. Métodos bidimensionales aplicados al reconocimiento facial. Proyecto de fin de carrera de la URJC, 2005.

[4] E. Cabello y otros. Empleo de sistemas biométricos para el reconocimiento de personas en aeropuertos. Instituto Universitario de Investigación sobre Seguridad Interior, Mayo 2006.

[5] Redes neuronales y sistemas borrosos. Ra-ma, 2001.

[6] P. Viola and M. J. Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. In Computer Vision and Pattern Recognition, 2001a.

[7] R. Collobert, S. Bengio, and J. Mariéthoz. Torch: a modular machine learning software library. Technical Report IDIAP-RR 02-46, IDIAP, 2002.

[8] http://www.epic.org/privacy/facerecognition/

Apéndice A

Requisitos del sistema

A.1. Requisitos Hardware

Concretamente el sistema hardware empleado es:

- Pentium 4 centrino 2.8 GHz

Debido al gran cómputo realizado por el sistema, se recomienda el uso del equipo disponible más potente puesto que a mayor potencia de cómputo, mayor número de frames por segundo podrán ser analizados. Así mismo, se sugiere que el equipo esté únicamente dedicado al reconocimiento facial.

El sistema permite adaptar parámetros como el número de frames por segundo a analizar. Este parámetro permite adaptar el sistema a la potencia hardware de la máquina.

A.2. Requisitos software

Concretamente el sistema software empleado es:

- Linux Ubuntu V 5.10

- Matlab 7.0
- gcc V 3.3[1]
- Torch Vision 3.0

[1] Con versiones superiores la librería Torch 3.0 no se puede compilar.

Apéndice B

Guía del usuario

B.1. Instalación

- Instalar Linux y Matlab 7.0 for Linux
- Instalar la librería Torch Vision 3.0
- Compilar los ejecutables de C++ facedetect y avi2ppm
- Añadir los ejecutables haarscan y avi2ppm al path para que puedan ser llamados desde cualquier carpeta
- Añadir el path de la carpeta con las funciones de matlab y todas las subcarpetas al path de Matlab

B.2. Configuración de la base de datos

- Definir los tipos de imágenes y el orden (El mismo para todos los individuos)
- Todas las imágenes deben tener el mismo formato de imagen (.jpg, .bmp...)

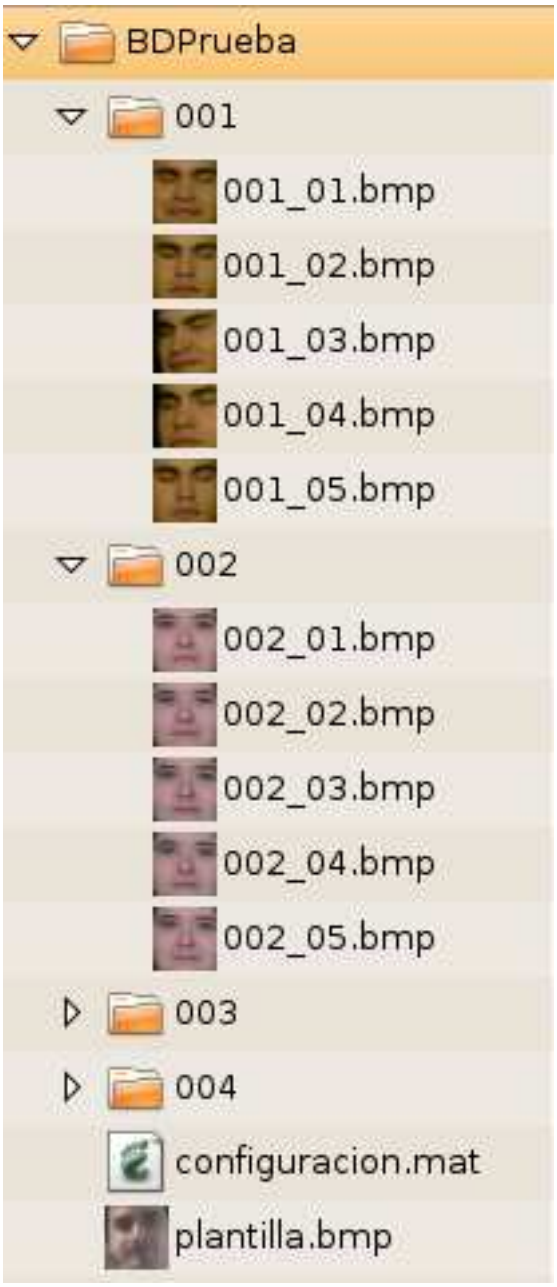


Figura B.1: Esquema de una BD

- Decidir si recortar las caras previamente o mediante el algoritmo de convolución

En la figura B.1 se puede ver el esquema de una BD.

B.3. Entrenamiento del sistema

Tras haber diseñado la base de datos, el paso siguiente es el entrenamiento del sistema. Con el entrenamiento, el sistema aprende a reconocer a los individuos de la base de datos. Para ello se deben definir los siguientes parámetros (Ver figura B.2):

RUTAS

- *Ruta a la BD:* Es la ruta a la carpeta raíz en la que se encuentra la BD alojada. (Debe ser una ruta válida y la BD estar bien construida)
- *Ruta destino:* Es la ruta a la carpeta raíz en la que se almacenarán los modelos generados tras el entrenamiento necesarios para la fase de análisis de vídeos. (Debe ser una ruta válida y existir)
- *Ruta a la BD:* Es la ruta a la imagen de la plantilla. (Debe ser una ruta válida). Ésta se empleará para:
  - Conocer cuál es el formato de imagen empleado en la BD.
  - Localizar las caras en las imágenes de laboratorio.
  - Definir el tamaño al que se interpolarán las imágenes de las caras.

ÍNDICE DE LA BASE DE DATOS

Especifica la estructura y el contenido de la base de datos de las imágenes. Se debe especificar:

- *Número de Individuos:* Es el número de personas[1] que contiene la base de datos.

[1] O lo que es lo mismo, de carpetas

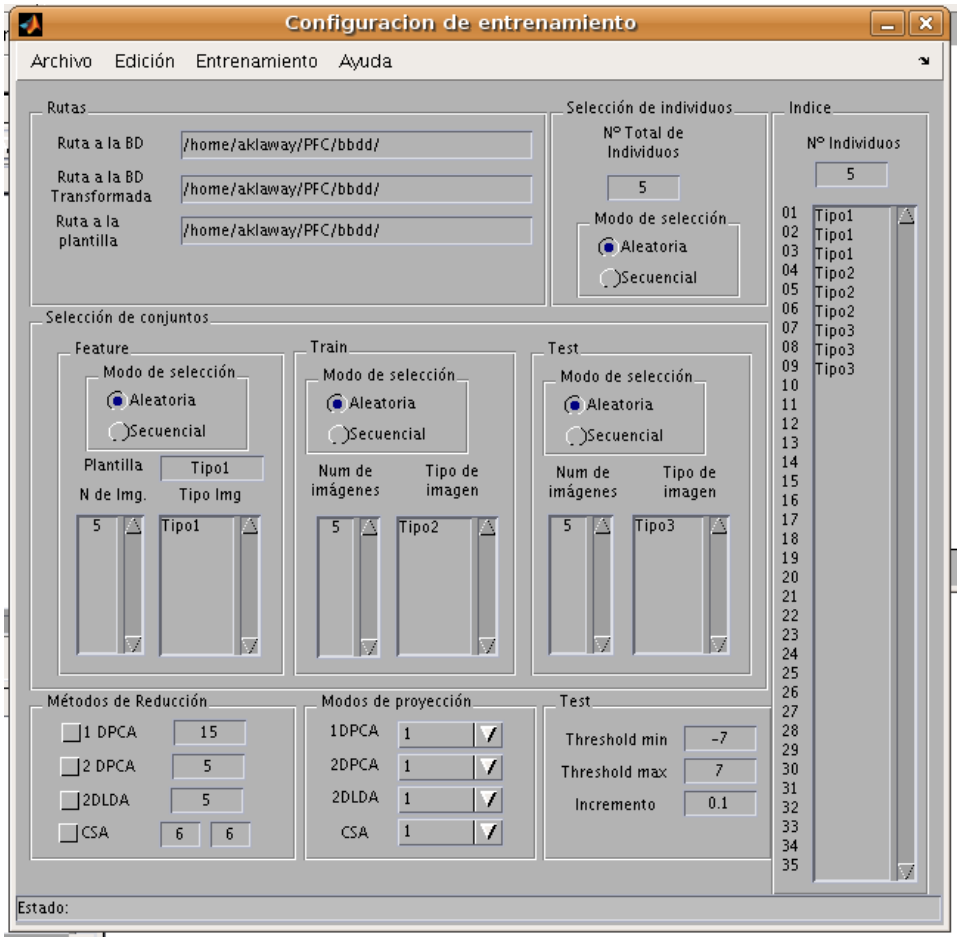


Figura B.2: Pantalla de configuración del entrenamiento

- *Tipo de imágenes:* Conocer el tipo de cada imagen de la base de datos es útil para poder personalizar el entrenamiento[2]. Indica implícitamente cuántas imágenes hay de cada persona y explícitamente de qué tipo es cada imagen (Frontal, Rotado 15º, Oclusión...). Los tipos son definidos por el usuario. Cada línea se corresponde con una imagen de la Base de Datos por orden.

ELECCIÓN DE LOS INDIVIDUOS

Indica el modo en el que se seleccionarán los individuos con los que entrenar el sistema en el caso de que no se entrene con todos.

- *Número de Individuos:* Es el número de personas con las que se entrenará el sistema. (Debe ser menor o igual al Nº de Ind. de la BD.)
- *Modo de selección:* Es la forma en la que se escogerán a los individuos. Puede ser aleatoria o secuencial.

CONJUNTOS

Para realizar el entrenamiento del sistema se dividen las imágenes disponibles de la base de datos en tres conjuntos:

1. *Feature (Rasgos):* A partir de las imágenes de este conjunto se extraen las características comunes a todas las caras de la base de datos. Estas características como la cara media, los autovalores y los autovectores son necesarias para la posterior proyección de las imágenes.
2. *Train (Entrenamiento):* Las imágenes de este conjunto son las que se emplearán para entrenar el sistema y obtener los modelos.
3. *Test (Prueba):* Las imágenes de este conjunto se emplearán para probar los modelos ya entrenados y generar las gráficas de error.

[2]Por ejemplo: Si en un a base de datos tenemos 4 imágenes frontales obtenidas en el laboratorio de cada persona y 5 imágenes extraídas de un vídeo. Indicaríamos que las imágenes 1, 2, 3 y 4 son de tipo 'FrontalLab' y las 5, 6, 7, 8 y 9 son 'VideoReal'. Esto nos permitirá definir entrenamientos personalizados.

Para cada conjunto hay que definir 3 parámetros y a mayores hay que definir uno más:

- *Modo de selección:* Es la forma en la que se escogerán a las imágenes. Puede ser aleatoria o secuencial
- *Tipo de imágen:* Indica que tipos de imágenes contendrá el conjunto (Frontales, 15º...)
- *Número de imágenes:* Indica la cantidad de imágenes de cada tipo que contendrá el conjunto.
- *Plantilla:* Indica el tipo de imagen que se empleará como plantilla.

Métodos de reducción

Debido a la gran cantidad de información redundante que contiene una imagen, se le aplica un método de reducción de dicha información con el objetivo de obtener una proyección de la cara con la mayor cantidad de información en un tamaño mucho menor. Actualmente, hay cuatro métodos de reducción implementados:

- PCA (Análisis de Componentes Principales): trata de hallar componentes (factores) que sucesivamente expliquen la mayor parte de la varianza total. Para ello se considera una imagen de MxN píxeles, como un vector de MxN componentes. Cada componente viene dado por la intensidad de cada uno de los píxeles.
- 2DPCA (Análisis de Componentes Principales Bidimensional): se trata de una técnica semejante a PCA, pero donde se tiene en cuenta la vecindad de cada píxel dentro de la imagen, tratando una imagen como una matriz, y realizando el análisis en dos dimensiones.
- 2DLDA (Análisis Lineal Discriminante Bidimensional): técnica estadística en la que se pretende encontrar las variables que mejor

discriminen o distingan a elementos de clases diferentes. Para ello se tiene en cuenta también la información de vecindad de los píxeles, con una imagen como matriz, realizando un tratamiento en dos dimensiones.

- CSA (Coupled Subspace Analysis): generalización de las técnicas anteriores. Mediante este método se infieren dos subespacios acoplados de baja dimensión que reconstruyen de manera óptima las matrices imagen en las direcciones fila y columna.

Debemos de indicar cuál es el tamaño del vector o matriz de la proyección para cada caso.

Modos de proyección

La proyección de las imágenes en los métodos 2DPCA, 2DLDA y CSA es en forma matricial pero se puede transformar a forma vectorial. La forma vectorial genera un único modelo para cada individuo. En cambio, la forma matricial genera para cada individuo tantos modelos como columnas tenga la matriz de proyección.

Por ello, se debe de indicar con un 1 que se desea la forma vectorial o con un 2 si se desea la forma matricial.

test

Tras el entrenamiento de los modelos el sistema realiza un test de éstos. El error del entrenamiento depende del parámetro de aceptación y para encontrar este parámetro óptimo el sistema necesita los siguientes parámetros:

- *Threshold mínimo:* Indica el parámetro de aceptación o threshold más bajo con el que se evaluará el funcionamiento del sistema.
- *Threshold máximo:* Indica el parámetro de aceptación o threshold más alto con el que se evaluará el funcionamiento del sistema.
- *Incremento:* Indica el incremento del parámetro de aceptación o threshold que se aplicará hasta llegar al threshold máximo.

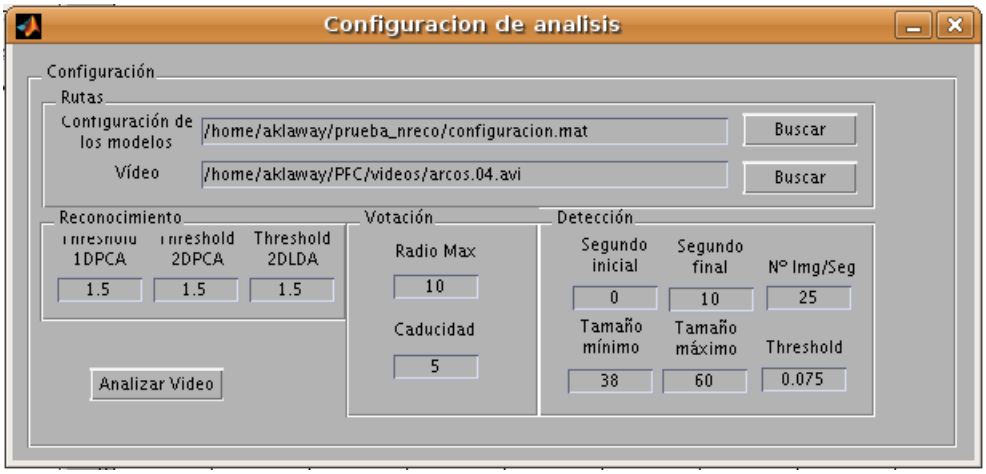


Figura B.3: Pantalla de configuración del análisis de un vídeo

B.4. Análisis de un vídeo

Tras el entrenamiento del sistema ya estamos preparados para analizar un vídeo (Ver figura B.4 y B.5) con el objetivo de identificar a las personas que aparecen por este. Para ello debemos de definir algunos parámetros (Ver figura B.3):

RUTAS

- *Configuración de los modelos:* Es la ruta al fichero de configuración de los modelos. (Debe ser una ruta válida y los modelos deben estar previamente entrenados)
- *Ruta dle vídeo:* Es la ruta al fichero del vídeo. (Debe ser una ruta válida y un fichero de vídeo válido)

DETECCIÓN

El primer paso al analizar una secuencia de vídeo es la detección de las caras en las escenas. Para ello hay que definir los siguientes parámetros:

- *Segundo inicial:* Es el segundo del vídeo en el que se comenzará el análisis.

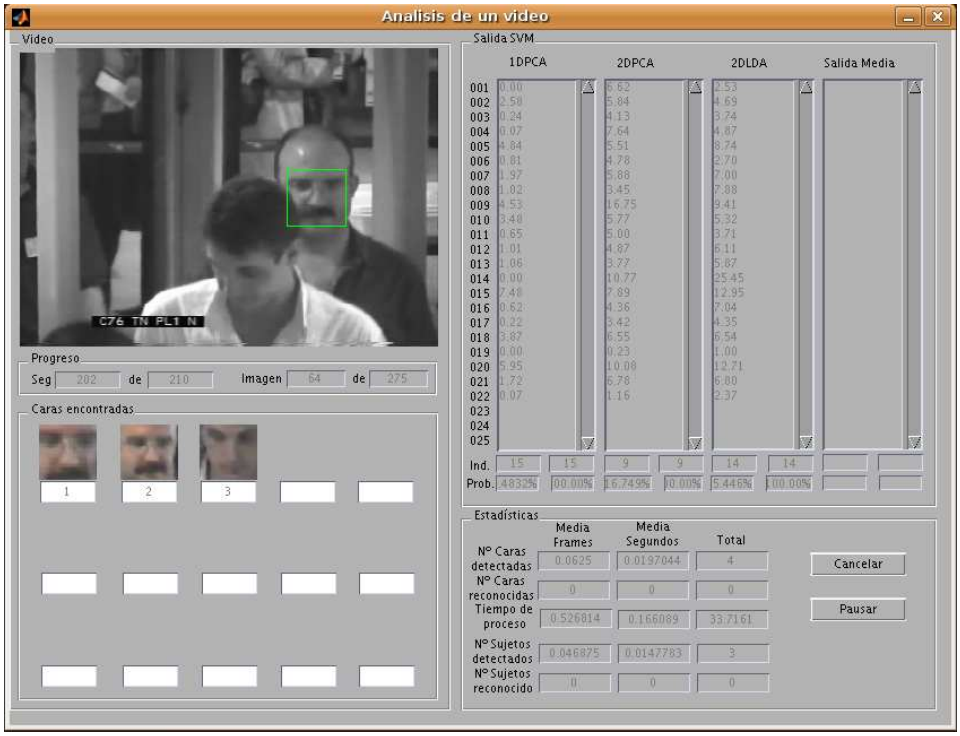


Figura B.4: Pantalla de análisis de un video

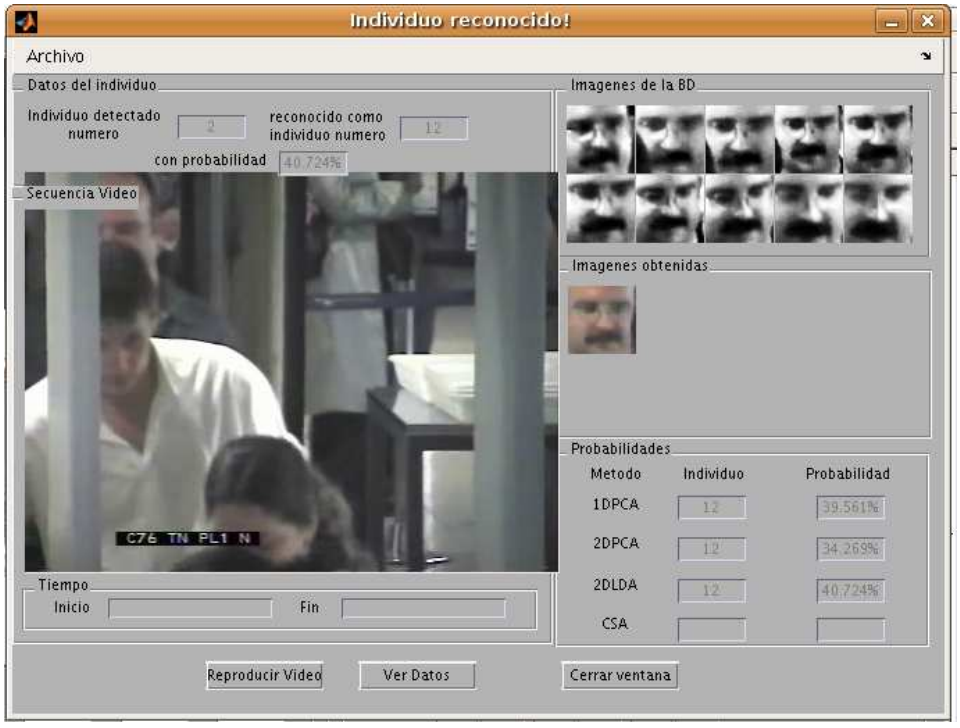


Figura B.5: Pantalla de individuo identificado

- *Segundo final:* Es el segundo del vídeo en el que se finalizará el análisis.
- *Número de imágenes por segundo:* Indica el número de frames que serán procesados de cada segundo del vídeo. Esto nos permite ajustar el sistema a la velocidad de la máquina con el objetivo de conseguir que el sistema funcione en tiempo real. A menor cantidad de frames/Seg mayor velocidad aunque también se reduce la probabilidad de éxito del sistema.
- *Tamaño mínimo:* Indica el tamaño mínimo de las caras que se buscarán en el vídeo.
- *Tamaño máximo:* Indica el tamaño máximo de las caras que se buscarán en el vídeo. Cuanto mayor sea el rango de tamaños de caras buscados más lento funcionará el sistema.
- Threshold: Es el parámetro de aceptación de una cara. Cuanto más bajo sea, mayor número de caras encontrará, pero además mayor número de falsas caras. Cuanto más alto sea, menor número de caras encontrará, pero también será menor el número de falsas caras. Por lo que se debe tener un valor intermedio.

VOTACIÓN

El sistema agrupa todas las caras de un mismo individuo y a partir de ellas decide quién es. Para poder agrupar las caras del mismo individuo se hace uso de la propiedad de localidad que nos indica que dos imágenes pertenecen al mismo individuo si se encuentran en posiciones espaciales y temporales cercanas. Para ello se definen los siguientes parámetros:

- *Radio máximo:* Indica la separación espacial máxima, en píxeles, entre dos caras para considerar que pertenecen al mismo individuo.
- *Caducidad:* Indica la separación temporal máxima, en número de frames, entre dos caras para considerar que pertenecen al mismo individuo.

Si los parámetros son altos, puede agrupar a varios individuos dentro de un mismo grupo. En cambio, si los parámetros son bajos, puede agrupar a un mismo individuo en varios grupos. Por lo que es importante llegar a un compromiso.

RECONOCIMIENTO

Para llegar a un compromiso entre los falsos positivos y falsos negativos del sistema se dispone del valor de threshold. Con valores bajos del parámetro el sistema presentará una alta tasa de falsos positivos y con un valor alto presentará una alta tasa de falsos negativos.

Apéndice C

Definición de Funciones

C.1. Función principal del entrenamiento

entrenar(): Función principal para entrenar los SVM.

Sintaxis: [salida] = entrenar(configuración)

Descripción: A partir de la configuración que se le pasa, realiza un preprocesado de las imágenes y finalmente entrena y realiza el test de los SVM.

Dependecias:

Algoritmo:

```
Definimos las variables
Leemos la configuración del inventario de la BD
[] = LeeConfInventario()
Leemos la configuración de los métodos de reducción
[] = LeeConfReduccion()
Leemos la configuración de los conjuntos Feature, Train y Test
[] = LeeConfConjuntos()
Creamos las carpetas.
Creamos la carpeta sets
```

```
Creamos la carpeta plantillas
Seleccionamos los individuos
[] = elegirindividuos()
Obtenemos los números de fotos correspondientes a los tres conjuntos
Obtenemos los números de foto de Feature_set
[] = obtener_Feature_set();
Obtenemos los números de foto de Train_set
[] = obtener_Train_set()
Obtenemos los números de foto de test_set
[] = obtener_test_set()
Recortamos las imágenes del conjunto
[] = recortaConjuntos()
Proyección de los conjuntos Feature_set Train_set y test_set
[] = proyectar()
Reconstrucción de los conjuntos Feature_set Train_set y test_set
[] = reconstruir()
Entrenamiento y test de los SVM
[] = EntrenaTest()
Mostramos la gráfica del error
Gráfica()
```

C.2. Dependencias de primer orden: Dependencias de la función entrenar.

La función *entrenar.m* se encarga de llamar a las siguientes funciones:

LeeConfInventario(): Lee la configuración del inventario de la BD

Sintaxis: [Inventario, NIndividuos, salida] = LeeConfInventario(configuración)

Descripción: Lee la información del indice de la BD almacenado en la configuración y devuelve esta información estructurada en las variable Inventario y NIndividuos.

Dependecias:

Entrada:

- configuración: (Estructura). Contiene la configuración

Salida:

- Inventario: (Vector de estructuras) Informacion de la estructura de la BD.
  - Inventario(x).tipo ->Tipo de foto
  - Inventario(x).numero ->Numero de fotos de ese tipo
- NIndividuos: Numero de individuos de la BD.
- salida: (entero 0 ó -1). Devuelve un 0 si la ejecucion fue correcta o -1 si se produjo algun error.

Algoritmo:

1. `Definimos las variables`
2. `Obtenemos el numero de individuos de la BD`
3. `Contamos el numero de tipos distintos`

   `Buscamos si el tipo actual ya está en la base de datos`

   `Si ya esta, aumentamos su contador en uno`
   `Si aun no esta, lo anhadimos`
4. `Devolvemos un vector con los tipo`

LeeConfReduccion(): Lee la configuración de los metodos de reduccion

Sintaxis: [metodo, d, salida] = LeeConfReduccion(configuración)

Descripción: Lee la informacion sobre los metodos de reduccion almacenada en la configuración y devuelve esta informacion estructurada en las variable metodo y d.

Dependecias:

Entrada:

- configuración: (Estructura). Estructura con la configuración de la BD.

Salida:

- metodo: (Vector String). Vector con los nombres de los metodos que hay que utilizar.
- d: (Vector Integer). Parametro a utilizar en cada metodo.
- salida: (entero 0 ó -1). Devuelve un 0 si la ejecucion fue correcta o -1 si se produjo algun error.

Algoritmo:

```
1. Definimos las variables
2. Inicicalizamos el vector de metodos y el de dimensiones
3. Leemos la configuración del 1dpca
4. Leemos la configuración del 2dpca
5. Leemos la configuración del 2dlda
6. Leemos la configuración del csa
```

LeeConfConjuntos(): Lee la configuración de los conjuntos Feature, Train y Test

Sintaxis: [vc, modo, TipoPlantilla, salida] = LeeConfConjuntos(configuración)

Descripción: Lee la configuración de los conjuntos Feature, Train y Test.

Dependecias:

Entrada:

- configuración: (Estructura). configuración de la BD.
- InventarioBD: (Vector de estructuras) Informacion de la estructura de la BD.
  - InventarioBD(x).tipo ->Tipo de foto
  - InventarioBD(x).numero ->Numero de fotos de ese tipo

Salida:

- vc: (Estructura). Tipo de imagenes y que cantidad de ellas hay en cada conjunto.
  - vc{x}(i).numero
  - vc{x}(i).tipo
  - x pertenece a [1,2,3], siendo x el conjunto: 1 Feature, 2 Train y 3 Test
  - Como para cada conjunto se pueden emplear uno o mas tipos de imagenes, i indica cada uno de los tipos empleado en el del conjunto

modo: (Entero). Modo de seleccion de cada conjunto. 0 Aleatorio, 1 Secuencial.

TipoPlantilla: Tipo de imagen (De los del indice de la BD) de la plantilla a emplear.

salida: (Entero). 0 o -1:

Si se ejecuta correctamente devuelve un 0

Si se produce algun error devuelve un -1

mensaje: (String). Mensaje de error.

Algoritmo:

1. `Definimos las variables`
2. `Leemos la configuración del conjuto Feature`
3. `Leemos la configuración del conjuto Train`
4. `Leemos la configuración del conjuto Test`
5. `Validamos los conjuntos Feature, Train y Test`

elegirindividuos(): Selecciona los individuos

Sintaxis: [Individ, salida, mensaje] = elegirindividuos(num_indi_totales, num_indi_seleccion, modo, varargin)

Descripción: Selecciona segun el modo, "num_indi_seleccion" individuos de los 'num_indi_totales' individuos. Los numeros seleccionados se devuelven como Strings en un vector.

Dependecias:

n = randx2y()

n = randx2n()

Entrada:

- num_indi_totales: (Entero >= 0) N Max de individuos que se podra seleccionar.
- num_indi_selecion: (Entero >= 0) N de individuos que se quiere seleccionar.
- modo: (Entero 0 o 1) modo de seleccion de individuos. Valores posibles:
  - 0 ->seleccion aleatoria (Seleccion secuencial a partir de un indiviudo aleatorio)
  - 1 ->seleccion secuencial
- minimo: varargin{1}
- minimo: (Entero >= 1) opcional (si se introduce, los individuos se seleccionan entre el minimo y el num_indi_totales.

Salida:

- Idivid: (Matriz de caracteres). Cada fila corresponde con un numero entero positivo. Si se produce un error devuelve un vector vacio.
- salida: Tipo de error que se produce:
  - 0 ->Modo de selleccion no implementado.
  - -1 ->Numero de individuos totales menor que 1.
  - -2 ->Numero de individuos a sellecionar menor que 1.
  - -3 ->Numero de individuos totales menor que numero de individuos a seleccionar.
  - 1 ->no se ha producido ningun error.
- mensaje: (String). Mensaje de error.

Algoritmo:

```
0. Definimos las variables
1. Comprobamos que el numero de individuos sea seleccionado correcto.
2. Seleccionamos los individuos
2.1 Si el modo es 0: SELECCION ALEATORIA

    Obtenemos el numero de cifras del num_ind_totales
    Inicializamos a 0 el contador de individuos seleccionados.
    Calculamos un numero aleatorio entre minimo y el numero de individuc
    Anhadimos secuencialmente a la lista los individuos desde n hasta nu



        Anhadimos el individuo seleccionado a la lista de individuos
        Avanzamos al siguiente individuo, si llegamos al final, volvemos
2.2 Si el modo es 1: SELECCION SECUENCIAL

    Obtenemos el numero de cifras del num_ind_totales
    Inicializamos a 0 el contador de individuos seleccionados.
    Calculamos un numero aleatorio entre minimo y el numero de individuc
```

```
Anhadimos secuencialmente a la lista los individuos desde el min has


    Anhadimos el individuo seleccionado a la lista de individuos
    Avanzamos al siguiente individuo, si llegamos al final, volvemos
```

obtener_feature_set(): Obtiene los numeros de foto de feature_set

Sintaxis: [mat,err] = obtener_feature_set(n_BBDD, n_feature, modo, plantilla)

Descripción: Selecciona las fotos correspondientes al conjunto feature_set.

Dependecias:

elegirIndividuos()

Entrada:

- n_BBDD : vector de estructuras que almacena la composicion de la BD.
  - n_BBDD(x).tipo
  - n_BBDD(x).numero
  - siendo x de 1 al numero de tipos distintos de fotos en la base de datos.
- n_feature: vector de estructuras que almacena la composicion del conjunto feature_set.
  - n_feature(x).tipo
  - n_feature(x).numero
  - siendo x de 1 al numero de tipos distintos de fotos en la estructura.
- modo: (Entero >= 0) modo de seleccion.

- plantilla: (String). Tipo de foto que se utilizara para crear la plantilla.

Salida:

- mat: (Matriz de caracteres). Cada fila corresponde con un numero entero positivo.
- err: (Entero). Devuelve ERROR si se produce algun error y OK en el caso de que la funcion se ejecute correctamente.

Algoritmo: [Copia NRECO]

obtener_train_set(): Obtiene los numeros de foto de train_set

Sintaxis: [mat, err] = obtener_train_set(n_BBDD, train, n_train, modo)

Descripción: Selecciona las fotos correspondientes al conjunto train_set.

Dependecias:

elegirIndividuos()

Entrada:

- n_BBDD : vector de estructuras que almacena la composicion de la BD.
  - n_BBDD(x).tipo
  - n_BBDD(x).numero
  - siendo x de 1 al numero de tipos distintos de fotos en la base de datos.
- train: matriz de caracteres correspondiente a los numeros de fotos del conjunto train_set.
- n_train: vector de estructuras que almacena la composicion del conjunto train_set.

- n_train(x).tipo
- n_train(x).numero
- siendo x de 1 al numero de tipos distintos de fotos en la estructura.

▪ modo: (Entero >= 0) modo de seleccion.

Salida:

▪ mat: (Matriz de caracteres). Cada fila corresponde con un numero entero positivo.

▪ err: (Entero). Devuelve ERROR si se produce algun error y OK en el caso de que la funcion se ejecute correctamente.

Algoritmo: [Copia NRECO]

obtener_test_set(): Obtiene los numeros de foto de test_set

Sintaxis: [mat, err] = obtener_test_set(n_BBDD, test, n_test, modo)

Descripción: Selecciona las fotos correspondientes al conjunto test_set.

Dependecias:

elegirIndividuos()

Entrada:

▪ n_BBDD : vector de estructuras que almacena la composicion de la BD.

- n_BBDD(x).tipo
- n_BBDD(x).numero
- siendo x de 1 al numero de tipos distintos de fotos en la base de datos.

- test: matriz de caracteres correspondiente a los numeros de fotos del conjunto test_set.
- n_test: vector de estructuras que almacena la composicion del conjunto test_set.
  - n_test(x).tipo
  - n_test(x).numero
  - siendo x de 1 al numero de tipos distintos de fotos en la estructura.
- modo: (Entero >= 0) modo de seleccion.

Salida:

- mat: (Matriz de caracteres). Cada fila corresponde con un numero entero positivo.
- err: (Entero). Devuelve ERROR si se produce algun error y OK en el caso de que la funcion se ejecute correctamente.

Algoritmo: [Copia NRECO]

recortaConjuntos(): Recorta las imagenes de los conjuntos

Sintaxis: [salida]=recortaConjuntos(configuración, indi, feature, train, test)

Descripción: Recorta las imagenes de los conjuntos

Dependecias:

recortaGradd()

leerhastapunto()

juntar_cad2matcad()

juntar_matcad2cad

escribiratxt()

Entrada:

- configuración: (Estructura). configuración de la BD.
- indi: Individuos a recortar
- feature: Numeros de imagen del conjunto feature
- train: Numeros de imagen del conjunto train
- test: Numeros de imagen del conjunto test

Salida:

- salida : (Entero). 0 o -1:
  - Si se ejecuta correctamente devuelve un 0
  - Si se produce algun error devuelve un -1

Algoritmo:

```
Definimos las variables
0. Obtenemos el tipo de fichero de la plantilla (bmp, jpg, png...)
1. Definimos las matrices donde se almacenaran los conjuntos
2. Recortamos las imagenes de cada conjunto
Creamos la carpeta para el individuo i
2.1 Conjunto feature_set
Obtenemos los nombres de los ficheros de cada imagen
Guardamos los nombres en un ficero.txt
Creamos la plantilla del individuo si no existe
Recorte de imagenes
2.2 Conjunto train_set
Obtenemos los nombres de los ficheros de cada imagen
Guardamos los nombres en un ficero.txt
Creamos la plantilla del individuo si no existe
Recorte de imagenes
2.3 Conjunto test_set
Obtenemos los nombres de los ficheros de cada imagen
```

```
Guardamos los nombres en un ficero.txt
Creamos la plantilla del individuo si no existe
Recorte de imagenes
3. Guardamos los conjuntos en ficheros.mat
Guardamos el conjunto de features
Guardamos el conjunto de train
Guardamos el conjunto de test
```

proyectar(): Proyeccion de los conjuntos feature_set train_set y test_set

Sintaxis: salida = proyectar(path_trans, path_config)

Descripción: Obtencion de las proyecciones de los conjuntos feature_set, train_set y test_set. El tipo de proyeccion depende del fichero de configuración.

Dependecias:

metodo_1Dpca()
proyectar_1dpca()
metodo_2Dpca()
proyectar_2dpca()
metodo_2Dlda()
proyectar_2dlda()
metodo_Csa()
proyectar_csa()

Entrada:

- path_trans: (String). Path de la base de datos transformada.
- path_config: (String). Path del fichero de configuración.

Salida:

- salida: (Entero 0 ó 1)
  - 0 se produjo un error en la ejecucion.

- 1 la ejecucion se realizo correctamente.

Algoritmo:

```
1. Definimos las variables
2. Proyectamos 2dpca

    2.1 Recuperacion de la estructura global de datos
    2.2 Borrado y creacion del directorio de proyeccion
    2.3 Proyectamos el conjunto Feature
    2.4 Proyectamos el conjunto Train
    2.5 Proyectamos el conjunto Test
3. Proyectamos 1dpca

    3.1 Recuperacion de la estructura global de datos
    3.2 Borrado y creacion del directorio de proyeccion
    3.3 Proyectamos el conjunto Feature
    3.4 Proyectamos el conjunto Train
    3.5 Proyectamos el conjunto Test
4. Proyectamos 2dlda

    4.1 Recuperacion de la estructura global de datos
    4.2 Borrado y creacion del directorio de proyeccion
    4.3 Proyectamos el conjunto Feature
    4.4 Proyectamos el conjunto Train
    4.5 Proyectamos el conjunto Test
5. Proyectamos csa

    5.1 Recuperacion de la estructura global de datos
    5.2 Borrado y creacion del directorio de proyeccion
    5.3 Proyectamos el conjunto Feature
    5.4 Proyectamos el conjunto Train
    5.5 Proyectamos el conjunto Test
```

reconstruir(): Reconstruccion de los conjuntos feature_set train_set y test_set

Sintaxis: salida = reconstruir(path_trans, path_config)

Descripción: Reconstruccion de las proyecciones de los conjuntos feature_set, train_set y test_se. El tipo de proyeccion depende del fichero de configuración.

Dependecias:

existe_metodo()

reconstruir_1dpca()

reconstruir_2dpca()

reconstruir_2dlda()

reconstruir_csa()

Entrada:

- path_trans: (String). Path de la base de datos transformada.
- path_config: (String). Path del fichero de configuración.

Salida:

- salida: (Entero 0 ó 1)
  - 0 se produjo un error en la ejecucion.
  - 1 la ejecucion se realizo correctamente.

Algoritmo:

```
1. Definimos las variables
2. Reconstruimos 2dpca

    2.1 Recuperacion de la estructura global de datos
    2.2 Reconstruimos el conjunto Feature
    2.3 Reconstruimos el conjunto Train
    2.4 Reconstruimos el conjunto Test
```

```
3. Proyectamos 1dpca

    3.1 Recuperacion de la estructura global de datos
    3.2 Reconstruimos el conjunto Feature
    3.3 Reconstruimos el conjunto Train
    3.4 Reconstruimos el conjunto Test
4. Proyectamos 2dlda

    4.1 Recuperacion de la estructura global de datos
    4.2 Reconstruimos el conjunto Feature
    4.3 Reconstruimos el conjunto Train
    4.4 Reconstruimos el conjunto Test
5. Proyectamos csa

    5.1 Recuperacion de la estructura global de datos
    5.2 Reconstruimos el conjunto Feature
    5.3 Reconstruimos el conjunto Train
    5.4 Reconstruimos el conjunto Test
```

EntrenaTest(): Entrenamiento y test de los SVM

Sintaxis: [salida]=EntrenaTest(configuración)

Descripción: Realiza el entrenamiento y el test de los modelos SVM

Dependecias:

svm_model()
construccion_test_svm()
construir_tablas()
svm_modelcsa()
construccion_test_svmcsa()
construir_tablascsa()

Entrada:

- configuración: (Estructura). configuración de la BD.
- indi: Individuos a recortar

Salida:

- salida : (Entero). 0 o -1:
  - Si se ejecuta correctamente devuelve un 0
  - Si se produce algun error devuelve un -1

Algoritmo:

```
0. Definimos las variables
1. 1DPCA
    A) Entrenamiento
    B) Test
2. 2DPCA
    A) Entrenamiento
    B) Test
3. 2DLDA
    A) Entrenamiento
    B) Test
4. CSA
    A) Entrenamiento
    B) Test
```

Grafica(): Muestra la gráfica del error

Sintaxis: Grafica(configuración)

Descripción: Abre una interfaz gráfica en la que según los parámetros del usuario representa las curvas ROC y muestra el EER obtenido en el entrenamiento

Dependecias:

existe_metodo()

Entrada:

- configuración: (Estructura). configuración de la BD.

Salida:

Algoritmo:

1. Representa 1dpca
2. Representa 2dpca
3. Representa 2dlda
4. Representa csa

C.3. Función analizar

`analizar()`: Función principal para analizar un vídeo

Sintaxis: `[salida] = analizar(configuración)`

Descripción: A partir de la configuración que se le pasa, lee el vídeo, busca las caras en la escena, agrupa todas las caras de un individuo en una lista, proyecta las caras, las clasifica y finalmente identifica al sujeto ofreciendo una estimación de la probabilidad de acierto.

Dependecias:

Algoritmo:

```
Definimos las variables
Mostramos las marcas de tiempo del video
Inicializamos a cero los contadores
Comenzamos a extraer imagenes. Extraemos imagenes por cada segundo
for segundos=SegInicial:SegFinal
    Extraemos las imagenes del segundo actual
```

```
Analizamos las imagenes
for i=0:NImgPorSeg-1

    DETECCION: Recortamos las caras de la imagen
    recortaCara()
    DETECCION: Mostramos la imagen analizada con las caras encontradas
    ESTADISTICAS: Actualizamos los contadores y mostramos las estadisti
    Estadisticas de detecciones de caras
    Estadisticas de Reconocimiento de caras
    Estadisticas de Deteccion de sujetos

    ANALISIS DE LAS CARAS DETECTADAS
    Mostramos las caras encontradas y las reconocemos
    for j=1:NCaras

        PREPROCESADO: Redimensionado, conversion a B/N y ecualizado
        PROYECCION Y RECONOCIMIENTO: Proyectamos la imagen y se la pres
        ProyectaYSVM();

        VOTACION: Tenemos en cuenta las img ant. para la decision
        AnhadirCara();

        ESTADISTICAS: Mostramos la salida del SVM.
        MostrarSalidaSVM();

        Actualizamos el num de caras reconocidas.

        Mostramos las caras recortadas
        MostrarSujetos();

    end
    Forzamos a que redibuje la pantalla
```

```
        end
    end
```