Trabajo fin de grado
Escuela de Ingeniería Informática

Detección, seguimiento y análisis de personas en streams de vídeo

Autor: ***Rubén Bolaños Naranjo***

Tutores: ***José Javier Lorenzo Navarro***
Profesor titular de la Universidad

Modesto F. Castrillón Santana
Profesor Titular de la Universidad

Las Palmas de Gran Canaria, Diciembre de 2013

Agradecimientos

Deseo dedicárselo a mi familia, en especial a mi madre y mi abuela por su apoyo incondicional durante todos estos años.

Me gustaría agradecer a Silvia Cristina Gil Marrero el estar a mi lado y ayudarme en lo que podía cuando la necesitaba

Por último, agradezco a mis tutores, en especial a Don José Javier Lorenzo Navarro por su ofrecimiento como tal, su atención y la gran ayuda prestada.

INDICE

1. Estado Actual y objetivos.

1.1 Introducción

En la actualidad el procesamiento de vídeo ha alcanzado mayores cotas debido a las mayores prestaciones de los ordenadores, a la constante aparición de nuevos métodos y técnicas y a la disminución de los precios de las propias videocámaras (al igual que da la tecnología en general), lo que se traduce en que el uso de estas sea mucho más habitual en cualquier local, hasta en los propios ordenadores personales. Un problema que ha recibido bastante atención en los últimos años ha sido la detección de personas y por tanto han surgido multitud de técnicas para su resolución. Es en este campo donde se enmarca este proyecto, es decir, en el procesamiento de vídeo obtenido a partir de cámaras para la detección y seguimiento de personas. El proyecto lógicamente no investigará nuevas técnicas ni métodos sino que hará uso de librerías existentes y las integrará en una aplicación que permita su configuración y evaluación de una forma sencilla.

1.2 Marco social y económico

El proyecto está relacionado con la detección de personas mediante imágenes provenientes de una cámara. Esto se engloba dentro de la tecnología de la visión por computador, la cual es un área de conocimiento dentro de la inteligencia artificial y un amplio campo de futuro para la robótica de uso doméstico e industrial. La visión por computador y especialmente el reconocimiento de personas y caras también se usa para aparatos de grabación como cámaras de fotos y videocámaras.

1.3 Objetivos Académicos

Desde el punto de vista académico, este trabajo fin de grado (TFG) se ha orientado al aprovechamiento de los conocimientos adquiridos durante la titulación, de entre los que se destacan las materias

- Programación
- Estructura de Datos
- Base de Datos
- Ampliación de Análisis Numérico

La obtención del título que acredite los estudios de Grado en Ingeniería Informática perteneciente a la Universidad de Las Palmas de Gran Canaria

1.4 Objetivos Generales del trabajo

Un sistema de detección personal para personas de cuerpo entero con interfaz gráfica, para mejorar la eficiencia de la ya existente librería OpenCV, ayudándonos de Encara V2.2 aportando una aplicación basada en el uso conjunto.

La aplicación desarrollada deberá permitir almacenar información relativa a las personas detectada la misma. Para una mayor facilidad a la hora de manejar estos datos, esa información se almacenará en una base de datos de forma que otras aplicaciones puedan consultarla.

2. Competencias

2.1 CII01

“Capacidad para diseñar, desarrollar, seleccionar y evaluar aplicaciones y sistemas informáticos, asegurando su fiabilidad, seguridad y calidad, conforme a principios éticos y a la legislación y normativa vigente.”

Esta competencia queda cubierta en los capítulos de Análisis, Diseño y Resultados. Se encuentra en estos capítulos los motivos de elaboración que explican exhaustivamente las decisiones tomadas en cada caso.

2.2 CII02

“Capacidad para planificar, concebir, desplegar y dirigir proyectos, servicios y sistemas informáticos en todos los ámbitos, liderando su puesta en marcha y su mejora continua y valorando su impacto económico-social.”

De todos los elementos que intervienen en la gestión de un proyecto, la planificación es posiblemente la más importante, puesto que una mala o inexistente planificación conducirá a una mala realización en el proyecto, lo cual repercutirá enormemente tanto en la calidad como en el plazo de terminación del proyecto, ocasionando grandes perjuicios. Esta es la razón por la que esta competencia queda cubierta, puesto que la buena planificación de este trabajo de fin de grado ha dado lugar al propósito fundamental del mismo.

Esta competencia queda explicada en el capítulo de aportaciones, donde se explica el impacto del proyecto en el ámbito de la aplicación al que va destinado

2.3 CII04

Capacidad para elaborar el pliego de condiciones técnicas de una instalación informática que cumpla los estándares y normativas vigentes.

Pliego de especificaciones técnicas que han de regir en la contratación de una aplicación informática destinada a la realización del presente TFG.

2.3.1 Introducción

La aplicación desarrollada en este TFG para la detección de personas en streams de vídeo incorpora diferentes técnicas que se utilizan en la actualidad en la investigación en Visión por Computador.

La aplicación está diseñada para la detección y seguimiento de personas en un stream de Vídeo, pero no se concluye que sea una versión final para el uso la realizada en este TFG, sino una base, a partir de la cual podrían seguirse desarrollando futuras investigaciones.

La aplicación realizada se puede considerar en parte un prototipo ya que se pretende comprobar la validez de la combinación de las técnicas que se implementaron. De esta forma se puede evaluar sus puntos fuertes y débiles y tomarlos en consideración de cara a otra aplicación.

2.3.2 Objeto del contrato y condiciones

Condiciones generales

El presente pliego de prescripciones técnicas tiene como objeto definir las condiciones para la realización y una hipotética implantación del presente proyecto. En líneas generales el sistema dispone de las siguientes características:

- Capacidad para la visión online y en tiempo real, de las imágenes transmitidas por una cámara y su correspondiente señalización (también en tiempo real) de las personas detectadas.
- Almacenamiento en vídeo de las imágenes suministradas por las cámaras, de forma que se posibilite el acceso offline a las secuencias grabadas.

Cuando el uso del material desarrollado conlleve el tratamiento de datos personales se ha de aplicar la legislación vigente en España. En la actualidad la ley que regula este tema es la Ley Orgánica de Protección de Datos de Carácter Personal (LOPD). Además, en el ámbito de la vigilancia existe la Instrucción 1/2006, de 8 de noviembre que trata de adecuar la LOPD al tratamiento de imágenes que contengan personas físicas identificadas o identificables.

Ámbito

El ámbito en que se prestará el servicio será en cualquier entorno abierto o cerrado con tránsito personal donde sean visibles personas de cuerpo entero, las cuales se intentarán detectar y realizar seguimiento, mediante una cámara fija.

Propiedad del resultado de los trabajos

Todos los documentos y resultados de los trabajos realizados serán propiedad del autor y los tutores del proyecto que podrán ejercer el derecho de explotación en las condiciones estipuladas por la Universidad.

La empresa adjudicataria podrá hacer uso de los mismos, ya sea como referencia o base para trabajos futuros, siempre que cuente con la autorización expresa del contratante.

Organización del trabajo

Se llevarán a cabo diferentes reuniones a lo largo del proyecto con el tutor hasta dejar claras las especificaciones y diferentes formas de seguir avanzando en la realización del mismo.

Requisitos de fiabilidad

El desarrollo de la aplicación debe acometerse de tal forma que su resultado garantice:

• Comportamiento estable

• Conseguir una evaluación de la viabilidad del mismo, ya que se considera mayormente un demostrador donde comprobar las capacidades de las técnicas utilizadas.

• Seguridad de los datos sensibles

Transferencia Tecnológica

Durante la ejecución de los trabajos objeto del contrato, el adjudicatario se compromete a facilitar en todo momento a las personas designadas por el contratante a tales efectos la información que ésta solicite para disponer de un pleno conocimiento de las circunstancias en que se desarrollan los trabajos, así como de los eventuales problemas que pueden plantearse y de las tecnologías, métodos y herramientas utilizados para resolverlos.

Propiedad del resultado de los trabajos: la propiedad de los resultados del presente TFG vendrá determinado por la normativa de la ULPGC en este aspecto.

Entrega de la solución

Junto con el software se suministrará al contratante documentación, manuales de instalación y configuración, y manual de usuario, desarrollo de pruebas, plan de marcha atrás.

El adjudicatario deberá generar la documentación necesaria para asegurar el despliegue exitoso del producto y servicio de acuerdo con los procedimientos que siga su departamento de Informática.

Entorno Pre-explotación

De ser necesarias tareas correctivas una vez implementada la solución objeto de este contrato, con el fin de facilitar los trabajos, y realización de pruebas sin afectar al sistema en producción, será preciso que cualquier actividad a realizar sobre la plataforma se realice previamente en el entorno de pre-explotación similar al de producción. Será preciso verificar y testear el correcto funcionamiento en este entorno, como paso previo al despliegue en producción

2.4. CII18

Conocimiento de la normativa y la regulación de la informática en los ámbitos nacional, europeo e internacional.

El cumplimiento de esta competencia queda descrito en el capítulo “Normativa y Legislación”.

3. Aportaciones

En los últimos años, la robótica y la visión por computador están experimentando un fuerte auge, dentro de este ámbito se encuentra nuestro proyecto, más concretamente en la detección de personas.

La visión artificial, también conocida como visión por computador (del inglés *computer vision*) o visión técnica, es un subcampo de la Inteligencia Artificial. El propósito de la visión artificial es programar un computador para que "entienda" una escena o las características de una imagen.

Los objetivos típicos de la visión artificial incluyen:

- La detección, segmentación, localización y reconocimiento de ciertos objetos en imágenes (por ejemplo, caras humanas).
- La evaluación de los resultados (por ejemplo, segmentación, registro).
- Registro de diferentes imágenes de una misma escena u objeto, es decir, hacer concordar un mismo objeto en diversas imágenes.
- Seguimiento de un objeto en una secuencia de imágenes.
- Mapeo de una escena para generar un modelo tridimensional de la escena; este modelo podría ser usado por un robot para navegar por la escena.
- Estimación de las posturas tridimensionales de humanos.
- Búsqueda de imágenes digitales por su contenido.

Este TFG no es más que una aplicación de prueba la cual en un futuro podría tener múltiples aplicaciones, como por ejemplo: videovigilancia, visión robótica…

En cuanto a los sistemas de videovigilancia se puede añadir que en los últimos años han experimentado un enorme incremento ya que permiten monitorizar el comportamiento o actividades, generalmente de personas, en un espacio físico limitado.

Especialmente videovigilancia IP que es una tecnología de vigilancia visual que combina los beneficios analógicos de los tradicionales CCTV (Circuito Cerrado de Televisión) con las ventajas digitales de las redes de comunicación IP (Internet Protocol), permitiendo la supervisión local y/o remota de imágenes y audio así como el tratamiento digital de las imágenes, para aplicaciones como el reconocimiento de matrículas o reconocimiento personal y/o facial. La utilización del protocolo IP para la transferencia de las imágenes permite utilizar las infraestructuras de comunicación

existentes sin tener que lanzar cableado específico para la adquisición de las imágenes. La principal finalidad de los sistemas de videovigilancia es la seguridad pues resultan especialmente útiles en la prevención e investigación de posibles delitos.

La principal aportación por tanto de este proyecto es la mejora del detector de personas de cuerpo entero de la librería OpenCV, apoyándonos en el detector facial de la librería Encara v2.2.

4.Normativa y Legislación

A continuación se incluye la legislación vigente que afecta al presente Trabajo Fin de Grado

4.1. Normativa

4.1.1. Ley orgánica de protección de datos

El primer artículo de la Ley Orgánica de Protección de Datos de Carácter Personal (LOPD) dispone: "La presente Ley Orgánica tiene por objeto garantizar y proteger, en lo que concierne al tratamiento de los datos personales, las libertades públicas y los derechos fundamentales de las personas físicas, y especialmente de su honor e intimidad personal y familiar".

El su artículo 2.1 se especifica que la presente Ley Orgánica será de aplicación a los datos de carácter personal registrados en soporte físico, que los haga susceptibles de tratamiento, y a toda modalidad de uso posterior de estos datos por los sectores público y privado.

En el artículo 3 define en su letra a) los datos de carácter personal como "cualquier información concerniente a personas físicas identificadas o identificables", como puede ser en caso del presente TFG las imágenes o vídeos de cámaras IP. Nuevamente en el artículo 3 define en su letra c) el tratamiento de datos como aquellas "operaciones y procedimientos técnicos de carácter automatizado o no, que permitan la recogida, grabación, conservación, elaboración, modificación, bloqueo y cancelación, así como las cesiones de datos que resulten de comunicaciones, consultas, interconexiones y transferencias".

En definitiva, el principal objetivo de la LOPD es regular el tratamiento de los datos y ficheros, de carácter personal, independientemente del soporte en el cual sean tratados, los derechos de los ciudadanos sobre ellos y las obligaciones de aquellos que los crean o tratan. Las cámaras IP, y la videovigilancia en general, permite la captación, y en su caso la grabación, de información personal en forma de imágenes. Cuando su uso afecta a personas identificadas o identificables esta información constituye un dato de carácter personal a efectos de la presente Ley Orgánica 15/1999 (LOPD).

4.1.2. Instrucción 1/2006, de 8 de noviembre

Con el fin de adecuar los tratamientos de imágenes con fines de vigilancia a los principios de la Ley Orgánica de Protección de Datos y garantizar los derechos de las

personas cuyas imágenes son tratadas por medio de tales procedimientos se dictó la Instrucción 1/2006, de 8 de noviembre, de la Agencia Española de Protección de Datos.

El primer artículo define el ´ámbito de aplicación: "la presente Instrucción se aplica al tratamiento de datos personales de imágenes de personas físicas identificadas o identificables, con fines de vigilancia a través de sistemas de cámaras y videocámaras".

Además, el primer artículo en su sección 3 también especifica que "no se considera objeto de regulación de esta Instrucción el tratamiento de imágenes en el ´ámbito personal y doméstico, entendiéndose por tal el realizado por una persona física en el marco de una actividad exclusivamente privada o familiar." Diferentes artículos de esta instrucción regulan la videovigilancia, entre sus obligaciones están:

El artículo 3 indica que los responsables que cuenten con sistemas de videovigilancia deberán cumplir con el deber de información previsto en el artículo 5 de la Ley Orgánica 15/1999, de 13 de diciembre 6.1.1. A tal fin deberán colocar, en las zonas vídeovigiladas, al menos un distintivo informativo tanto en espacios abiertos como cerrados. Además, se deberá tener a disposición de los interesados la información relativa al propietario de los ficheros y el lugar donde el ciudadano puede ejercitar sus derechos.

El artículo 4.3 advierte que las cámaras instaladas en espacios privados no podrán obtener imágenes de espacios públicos salvo que resulte imprescindible para la finalidad de vigilancia pretendida, o resulte imposible evitarlo por su ubicación.

El artículo 6 determina que las imágenes no deberán almacenarse por un plazo superior a un mes (30 días). El artículo 7.1 especifica que la creación de un fichero de imágenes deberá ser notificado previamente a la Agencia Española de Protección de Datos, para su inscripción en el Registro General. El artículo 9 determina que el responsable deberá adoptar las medidas técnicas y organizativas necesarias que garanticen la seguridad de los datos y eviten su alteración, perdida, tratamiento o acceso no autorizado. Por ello las cámaras deberán estar protegidas mediante usuario y contraseña.

En definitiva, esta instrucción comprende la grabación, captación, transmisión, conservación, y almacenamiento de imágenes, incluida su reproducción o emisión, así como el tratamiento que resulte de los datos personales relacionados con ellas. No obstante, se excluyen de ella los datos personales grabados para uso doméstico y el tratamiento de imágenes por parte de las fuerzas y cuerpos de seguridad.

4.1.3. Directiva 95/46/CE

A escala europea la directiva 95/46/CE constituye el texto de referencia en materia de protección de datos personales. Su aplicación concierne a los datos tratados por

medios automatizados, así como a los contenidos en ficheros no automatizados. Sin embargo, no será aplicable al tratamiento de datos efectuado por una persona en el ejercicio de sus actividades particulares. La presente Directiva tiene como objetivo proteger los derechos y las libertades de las personas en lo que respecta al tratamiento de datos personales, estableciendo principios para determinar la licitud de dicho tratamiento. Dichos principios son los siguientes:

Los datos personales serán tratados de manera lícita y recogidos con fines determinados, explícitos y legítimos. El tratamiento de datos personales solo podría efectuarse con el consentimiento del interesado, salvo excepciones como: el cumplimiento de una obligación jurídica, protección del interés vital del interesado, cumplimiento de misión de interés público, etc. Deberá prohibirse el tratamiento de datos personales que revelen el origen racial o étnico, opiniones políticas, convicciones religiosas y la pertenencia a sindicatos, así como el tratamiento de los datos relativos a la salud o a la sexualidad. El responsable deberá facilitar datos del tratamiento (identidad, finalidad del tratamiento, destinatarios de los datos, etc) a la persona de quien se recaben los datos. Se podrá limitar el alcance de los principios anteriores con objetivo de salvaguardar, por ejemplo, la seguridad del Estado o la seguridad pública. El interesado tendrá derecho a oponerse a que los datos que le conciernen sean objeto de tratamiento. El responsable del tratamiento deberá aplicar las medidas adecuadas para proteger los datos personales contra la destrucción, accidental o ilícita, la perdida accidental, la alteración, la difusión o accesos no autorizados. El responsable del tratamiento notificara a la autoridad de control nacional con anterioridad a la realización del tratamiento. Al igual que la Ley Orgánica de Protección de Datos (LOPD), esta Directiva de referencia europea afecta a las cámaras IP y a la videovigilancia en general cuando las imágenes captadas o almacenadas disponen de información personal.

4.1.4 Real Decreto Legislativo 1/1996, de 12 de abril

Por el que se aprueba el texto refundido de la Ley de Propiedad Intelectual 23 (LPI), regularizando, aclarando y armonizando las disposiciones vigentes en la materia. Esta normativa de Propiedad Intelectual describe y regula el conjunto de derechos que pertenecen a los autores y otros titulares respecto de las obras. Ésta ley protege todo tipo de creaciones artísticas, literarias o científicas expresadas en cualquier tipo de soporte. La propiedad intelectual se define como el conjunto de derechos que se reconocen al autor sobre la obra o producto de su inteligencia y creatividad. Esta ley regula al titular del programa de ordenador, de la forma siguiente: según el artículo 97 "será considerado autor de un programa de ordenador la persona o grupo de personas naturales que lo hayan creado, o la persona jurídica que sea contemplada como titular de los derechos de autor en los casos expresamente previstos en la ley". Respecto a la

piratería, esta se define como la copia ilegal de la obra literaria, artística, musical y audiovisual. Recientemente, con el impacto de las nuevas tecnologías, podemos hablar de Piratería del software. En este contexto cabe destacar la Comisión Antipiratería cuyo objeto es la coordinación de las Administraciones Públicas entre sí y de estas con las organizaciones que se encargan de representar los derechos de propiedad intelectual. Finalmente, la Ley de Propiedad intelectual establece una serie de mecanismos de protección que pueden darse en caso de usurpación de derechos de propiedad intelectual: - Registro de la Propiedad Intelectual

- Depósito notarial - Comisión mediadora o arbitral de los derechos de propiedad intelectual- Licencia copyright

- Marcas de agua

4.2. Licencias

4.2.1 Licencias BSD

OpenCV fue liberado bajo la licencia BSD. Las licencias BSD son una familia de licencias de software libre permisivas. La licencia original se utilizó para la Berkeley Software Distribution (BSD), un sistema operativo tipo Unix, de donde adoptó su nombre.

Los dueños originales de BSD fue la Universidad de California, porque BSD se escribió por primera vez en dicha Universidad, en Berkeley. La primera versión de la licencia ha sido revisada, y las licencias resultantes son más adecuadamente denominadas *modified BSD licenses*.

Dos variantes de la licencia, la New BSD License/Modified BSD License y la Simplified BSD License/FreeBSD License, han sido verificadas como compatibles con las licencias de software libre GPL por la Free Software Foundation, y han sido seleccionadas como las licencias de código abierto por la Open Source Initiative, mientras que la original, licencia de 4 cláusulas, no ha sido aceptada como una licencia de código abierto y, aunque la original es considerada una licencia de software libre por la FSF, la FSF no considera que sea compatible con la GPL debido a la cláusula de publicidad.

Al ser licencias de software libre permisivas, estas ponen requisitos mínimos acerca de como se puede redistribuir el software. Esto está en contraste con las *copyleft licenses*, que tienen reciprocidad/requisitos de compartir por igual.

El autor, bajo esta licencia, mantiene la protección de los derechos de autor únicamente para la renuncia de garantía y para requerir la adecuada atribución de la autoría en trabajos derivados, pero permite la libre redistribución y modificación,

permitiéndose el uso en software comercial. Por este motivo, se ha utilizado en software propietario, como es el caso de Mac OS X.

Puede argumentarse que esta licencia asegura “verdadero” software libre, en el sentido de que el usuario tiene libertad ilimitada con respecto al software, y que puede decidir incluso redistribuirlo como no libre. Otras opiniones están orientadas a destacar que este tipo de licencia no contribuye al desarrollo de más software libre

Las licencias derivadas de la BSD se conocen comúnmente como “licencia BSD”. Hoy en día, la licencia BSD típica es la versión de 3 cláusulas.

A continuación se muestra una serie de tablas de cada licencia BSD

BSD License	
Autor	Universidad de California
Versión	N/a
Edición	Dominio público
Fecha de publicación	1989
Compatible con DFSG	Si
Software libre	Si
Aprobada por OSI	No
Compatible GPL	No
Copyleft	No
Utilizable junto con otras licencias	Si

"Simplified" BSD License	
Autor	El proyecto FreeBSD
Versión	N/a
Edición	Dominio público

Fecha de publicación	?
Software libre	Si
Aprobada por OSI	Si
Compatible GPL	Si
Copyleft	No
Utilizable junto con otras licencias	Si

4.2.2 GNU General Public License

XAMPP es una compilación de software libre (comparable a una distribución Linux), es gratuito y libre para ser copiado conforme los términos de la licencia GNU General Public License. Sin embargo, sólo la compilación de XAMPP está publicada bajo la licencia GPL.

La *Licencia Pública General GNU* o más conocida por su nombre en inglés *GNU General Public License* o simplemente sus siglas del inglés *GNU GPL*, es una licencia creada por la *Free Software Foundation* en 1989 (la primera versión), y está orientada principalmente a proteger la libre distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios. Existen varias licencias "hermanas" de la *GPL*, como la *licencia de documentación libre de GNU* (*GFDL*), la *Open Audio License*, para trabajos musicales, etc., y otras menos restrictivas, como la *MGPL*, o la *LGPL* (*Lesser General Publical License*, antes *Library General Publical License*), que permiten el enlace dinámico de aplicaciones libres a aplicaciones no libres.

La licencia *GPL*, al ser un documento que cede ciertos derechos al usuario, asume la forma de un contrato, por lo que usualmente se la denomina *contrato de licencia* o *acuerdo de licencia*

5. Recursos Hardware y Software

5.1 Recursos Hardware:

Una versión práctica de este proyecto requeriría equipo hardware relativamente potente, capaz de trabajar con imágenes y procesarlas en tiempo real, debido al alto coste computacional del algoritmo de procesamiento llevado a cabo.

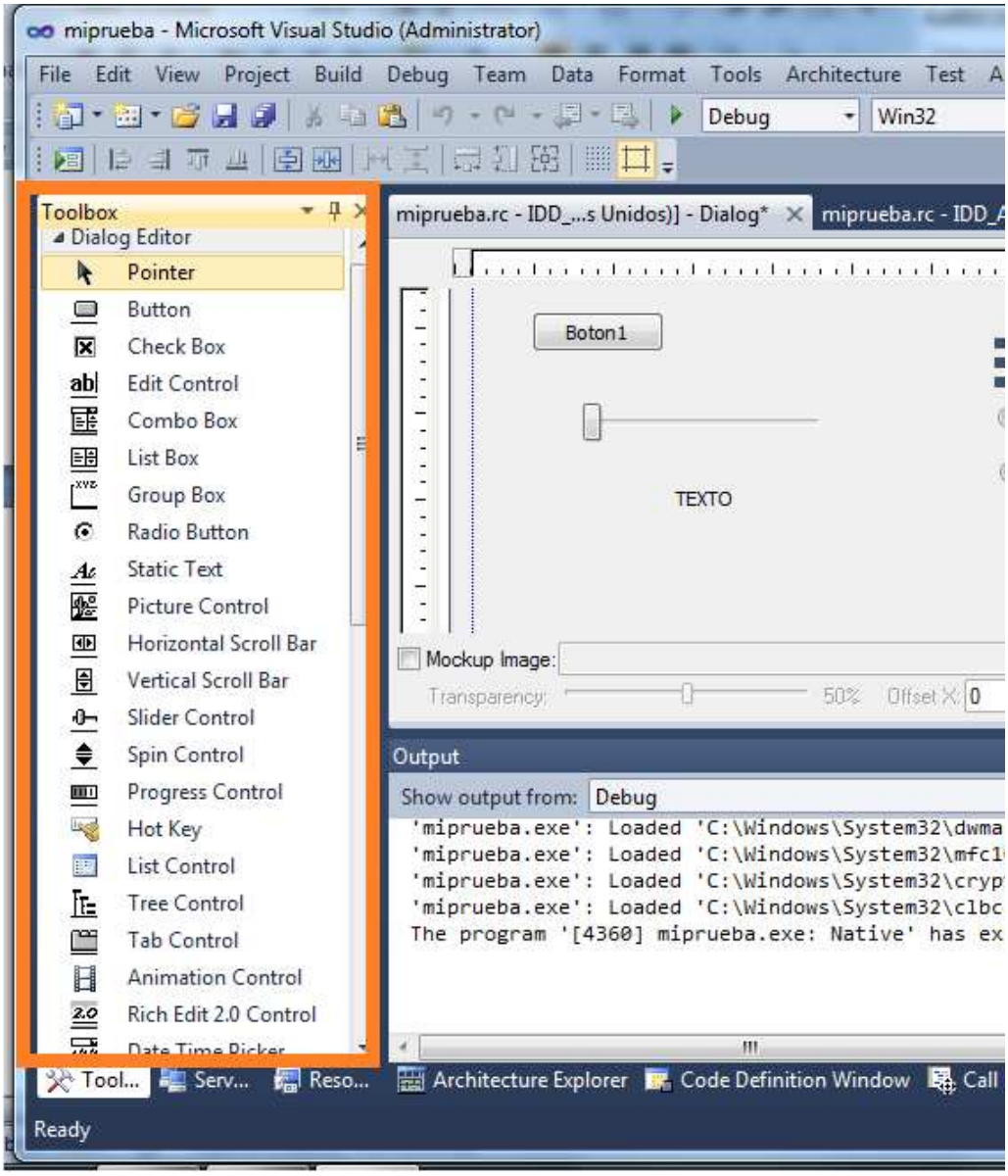
En cuanto al sistema de adquisición de imágenes, sería deseable contar con una cámara de alta calidad (para unos mejores resultados de detección al trabajar el programa con unas imágenes de más calidad), y un buen procesador sobre todo, para llevar a cabo dicho proceso.

Para nuestro caso que ha sido la prueba del mencionado algoritmo ha sido suficiente con un portátil hacer ASPIRE 5738, con procesador Intel Pentium Dual Core T4300 (2,10 Ghz, FSB 800 Mhz, 1 Mb. de cache); con este equipo no ha sido posible el procesamiento de imágenes en tiempo real con la fluidez deseada, que supondría el objetivo para que está pensado el proyecto. Pero si ha sido más que suficiente para las pruebas, realizadas algunas a partir de fragmentos de vídeos y otras a partir de la propia webcam.

5.2 Requisitos Software

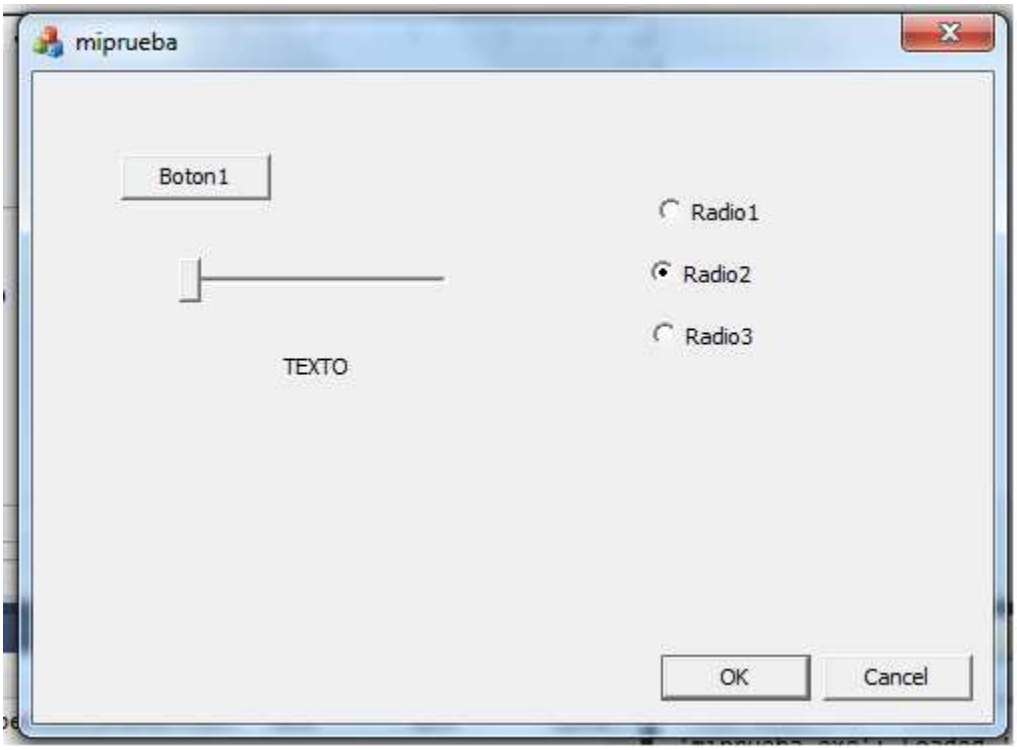
5.2.1 Visual Studio

Para la edición del programa y compilación se ha usado Microsoft Visual Studio 2010. Es un entorno de desarrollo integrado (IDE, por sus siglas en inglés) para sistemas operativos Windows, concretamente ha sido sobre windows 7 y en lenguaje C++. Visual Studio permite a los desarrolladores crear aplicaciones ejecutables gráficamente con pequeños objetos, sitios y aplicaciones web.



En el menú de selección de objetos se encuentran diferentes objetos como botones, cuadros de texto, recuadros de selección…..

Un ejemplo sencillo de ventana ejecutable sería la siguiente:



5.2.2 OpenCV

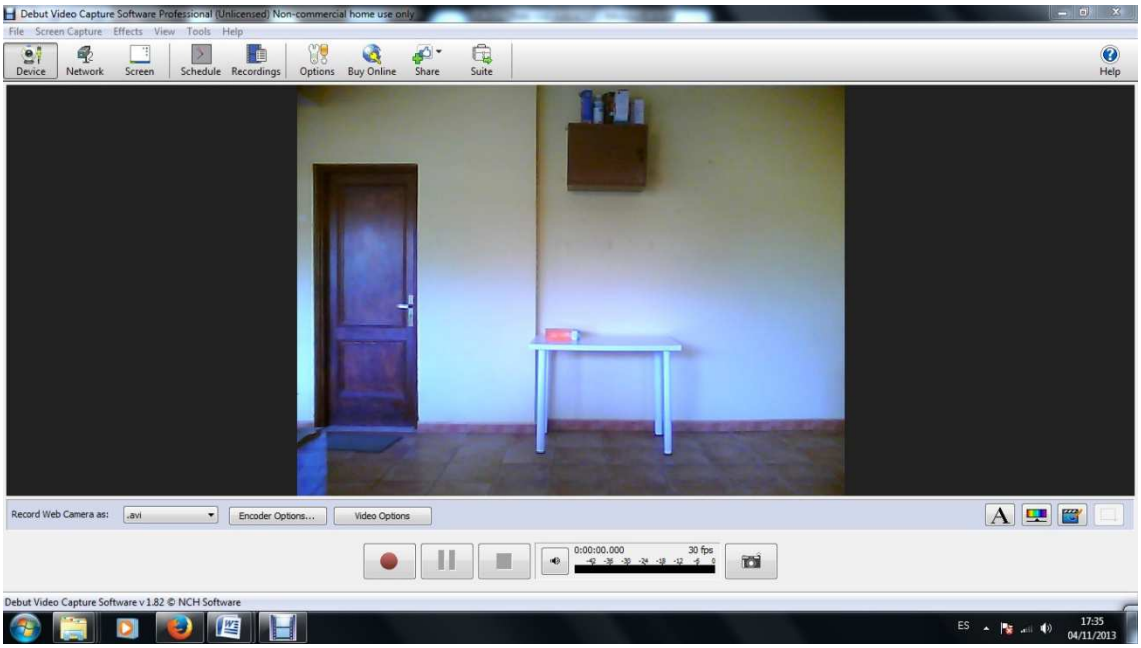
OpenCV es una librería orientada principalmente para el procesamiento de imágenes en tiempo real. Es una herramienta multiplataforma que proporciona diferentes APIs en múltiples lenguajes de programación. Usaremos la clase de histograma de gradiente orientado (HOG), en nuestro caso para la detección de personas (de cuerpo entero) en una imagen.

5.2.3 Librería Encara

En nuestro caso se utilizó la versión 2.2 de ésta librería. Esta librería permite la detección de rostros, ha sido diseñada y desarrollada en el Instituto Universitario SIANI y en el Departamento de Informática y Sistemas de la ULPGC.

5.2.4 Debut Vídeo Capture

Programa de grabación de Vídeo Debut Vídeo Capture: Programa utilizado para la grabación de vídeos para la posterior realización de las pruebas.

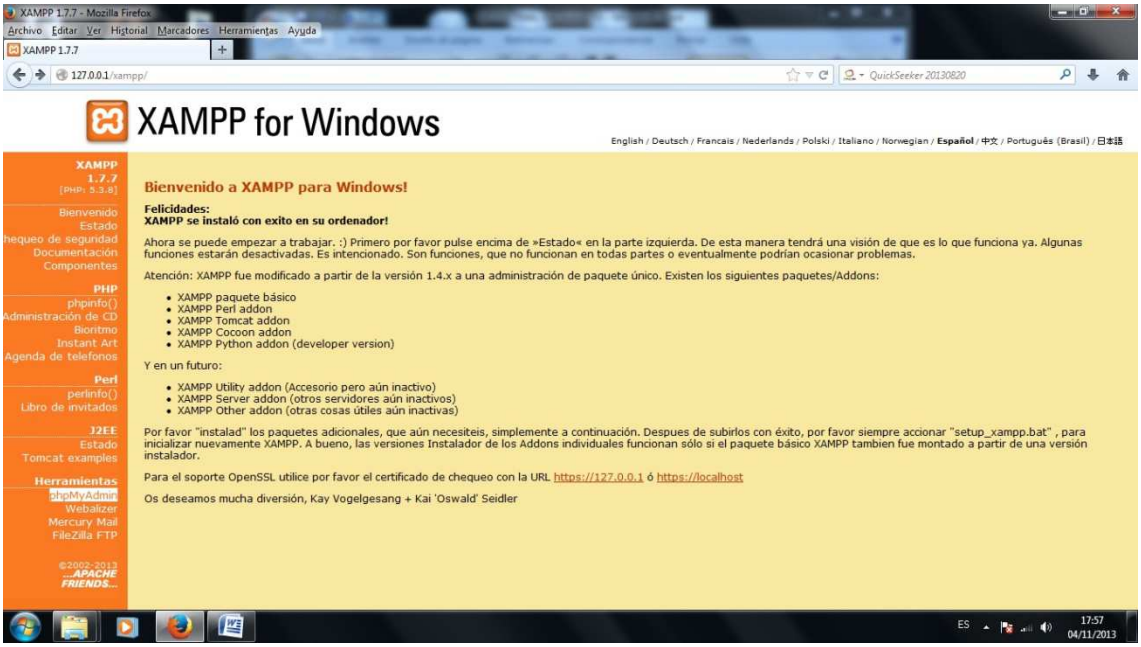


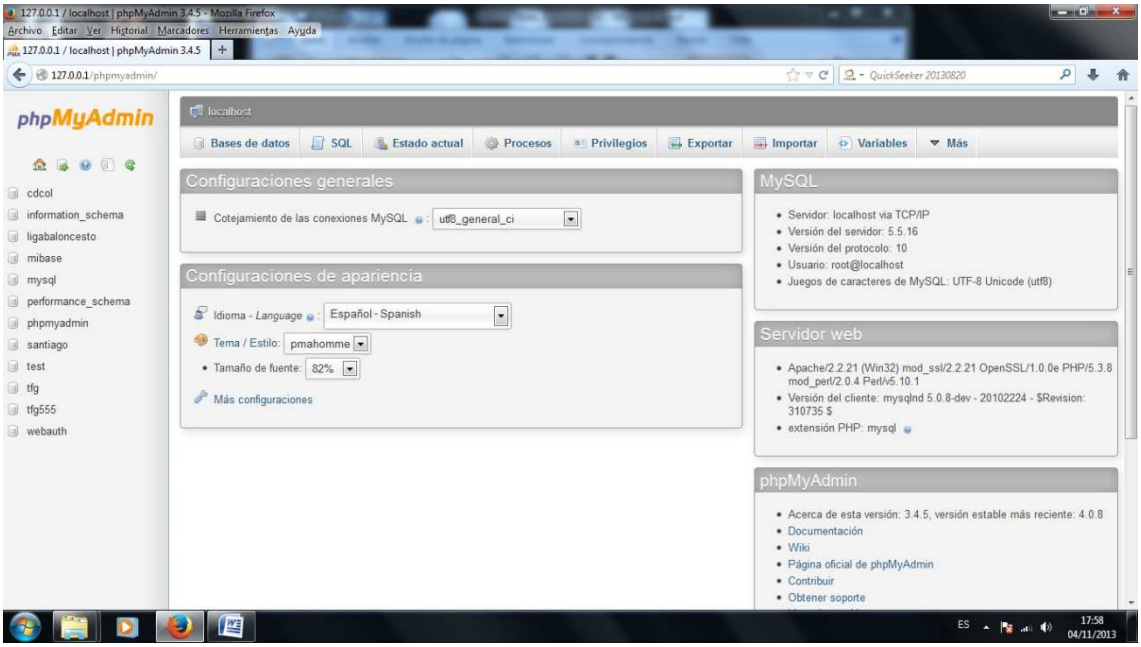
5.2.5 Xampp

XAMPP es un servidor independiente de plataforma, software libre, que consiste principalmente en la base de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script: PHP y Perl. El nombre proviene del acrónimo de X (para cualquiera de los diferentes sistemas operativos), Apache, MySQL, PHP, Perl.

Aunque en nuestro caso solo utilizaremos MySQL. Y el Apache simplemente para acceder a la base de datos mediante el explorer.

El programa está liberado bajo la licencia GNU y actúa como un servidor web libre, bastante fácil de usar y capaz de interpretar páginas dinámicas.



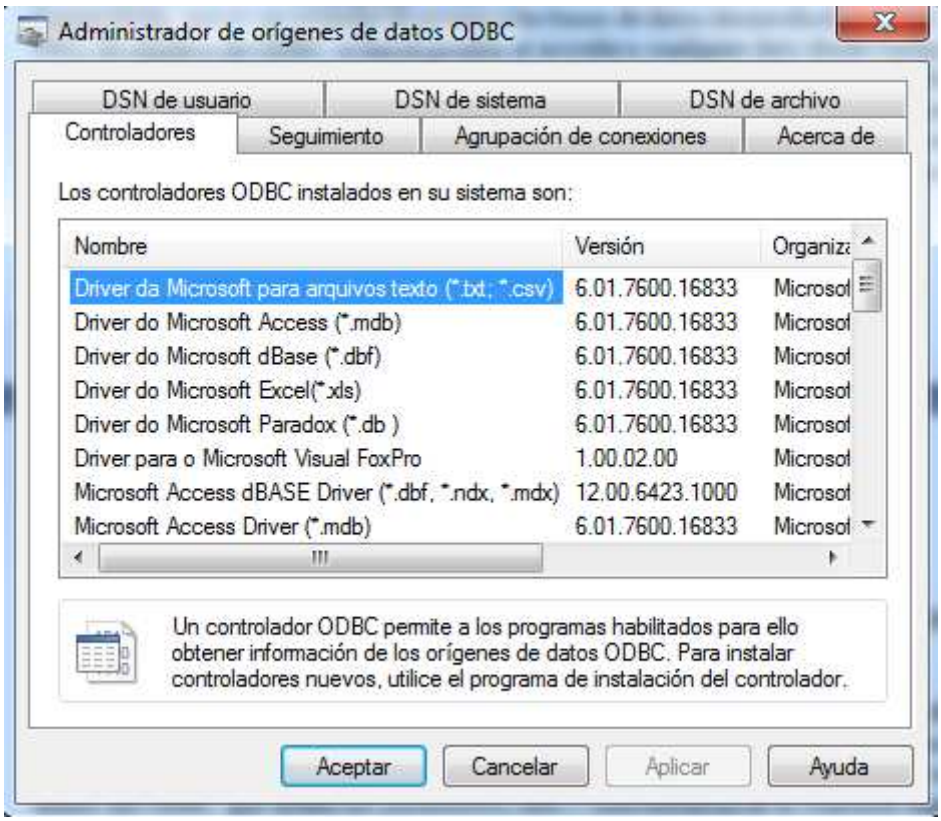


5.2.6 ODBC

La forma de acceder del programa principal a la Base de datos MySQL integrada en XAMPP se ha realizado mediante ODBC (Open DataBase Connectivity) que es un estándar de acceso a las bases de datos desarrollado por SQL Access Group en 1992. El objetivo de ODBC es hacer posible el acceder a cualquier dato desde cualquier aplicación, sin importar qué sistema de gestión de bases de datos (DBMS) almacene los datos. ODBC logra esto al insertar una capa intermedia (CLI) denominada nivel de Interfaz de Cliente SQL, entre la aplicación y el DBMS. El propósito de esta capa es traducir las consultas de datos de la aplicación en comandos que el DBMS entienda.

Para que esto funcione tanto la aplicación como el DBMS deben ser compatibles con ODBC, esto es que la aplicación debe ser capaz de producir comandos ODBC y el DBMS debe ser capaz de responder a ellos. Desde la versión 2.0 el estándar soporta SAG y SQL.

El software funciona de dos modos, con un software manejador en el cliente, o una filosofía cliente-servidor. En el primer modo, el driver interpreta las conexiones y llamadas SQL y las traduce desde el API ODBC hacia el DBMS. En el segundo modo para conectarse a la base de datos se crea una DSN dentro del ODBC que define los parámetros, ruta y características de la conexión según los datos que solicite el creador o fabricante.



6. Análisis

6.1 Estudio del problema

El problema a afrontar en este trabajo es la detección de personas a partir de un stream de vídeo. Luego pasado un stream de Vídeo, ya sea desde un archivo de vídeo o directamente desde una cámara, se deberá localizar a las personas presentes si las hubiera, especificar su ubicación, ya bien dibujando un recuadro que las señale y/o guardando coordenadas relativas de dichas personas en la propia imagen.

En la medida de lo posible se intentará hacer un seguimiento de las personas, contabilizando número de personas en cada frame, estudiando la continuidad de cada persona en frames consecutivos, añadiendo hora de aparición y desaparición de cada persona en el stream de vídeo. Una consideración a tener en cuenta es que no se hará ningún tipo identificación de una persona que estuviera, desapareciera y más tarde volviera a aparecer; al suceder esto esta persona se contabilizaría como dos personas distintas. Finalmente los datos relativos a cada persona, serán guardados en una base de datos.

Otro requisito a tener en cuenta es que las personas detectadas deberán estar "de cuerpo entero" ya que el detector de OpenCV empleado está pensado para ese fin, tal como se explicará posteriormente.

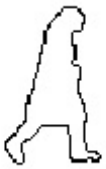
6.2 Metodos de Detección de Personas

Existen métodos de detección de personas basados en una imagen, sin tener en cuenta si esa imagen pertenece a un vídeo o no, (o sea sin tener en cuenta imágenes anteriores o posteriores) y métodos que si lo tienen en cuenta.

De los primeros métodos se diferencian dos grandes enfoques: basados en silueta o basados en apariencia.

El primer enfoque se basa en relacionar la región de interés con una silueta, donde la apariencia de las personas se modela mediante plantillas, de forma que asociar una imagen con una plantilla implica calcular la imagen de características y aplicar una transformación de distancia para obtener una imagen DT.

Imagen original

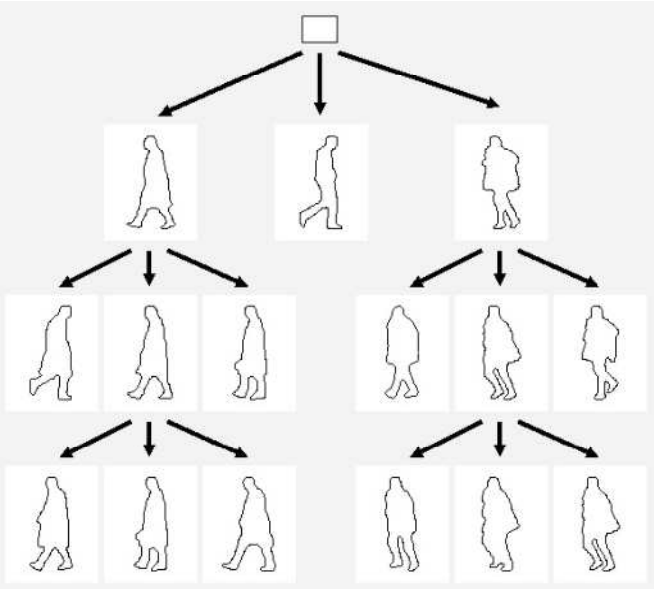


Plantilla

Imagen de Características

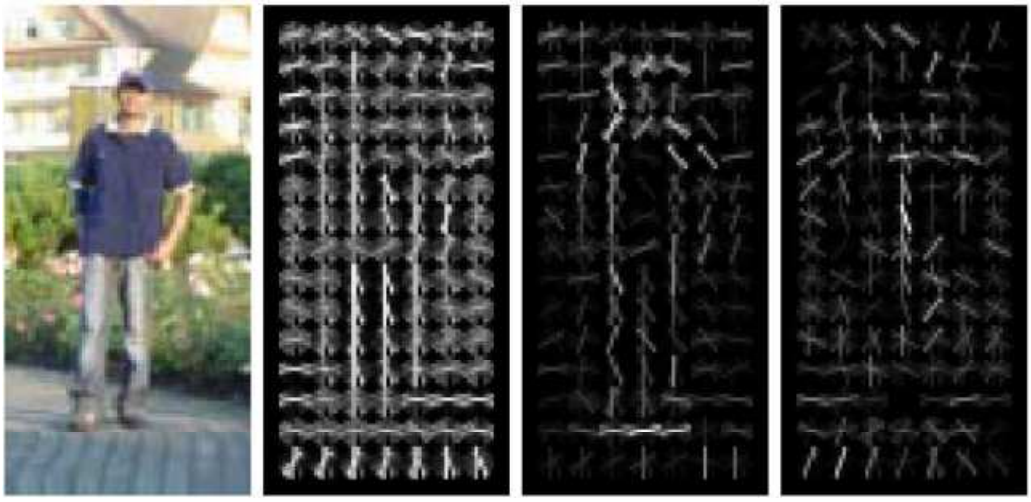
Imagen DT

Una transformación de distancia convierte una imagen binaria (consistente en píxeles, que pertenecen a las características y píxeles no pertenecientes) en una imagen donde el valor de cada píxel indica la distancia al píxel de la característica más cercana (Imagen DT). Un posible ejemplo de métrica es la distancia chamfer, que calcula una aproximación de la distancia euclídea usando cálculo integral. Para valorar la validez de la correspondencia se contabiliza el valor de los píxeles de la imagen DT que recaen sobre los píxeles de la plantilla. En la aplicación final se considera que una imagen corresponde con una plantilla si este valor es menor que un umbral determinado por el usuario. Según los autores de este tipo de trabajo, la principal contribución del método es el uso de una estructura jerárquica de plantillas de forma que se puede hacer una búsqueda eficiente entre todas las plantillas disponibles.



Ejemplo de jerarquía de plantillas de búsqueda

El segundo enfoque se basa en la apariencia de las personas: se define un conjunto de características de interés o descriptores y se entrena un clasificador con ellos. Dichos descriptores pueden hacer referencia al cuerpo completo (métodos de tipo holístico) o a partes individuales. Siguiendo el enfoque holístico se encuentran métodos como el propuesto por Dalal y Triggs, donde presentan un conjunto de características adecuado para la detección de personas llamado Histogram of Oriented Gradients (HOG) como el que se usa en el presente TFG. El método se basa en evaluar histogramas locales normalizados de una imagen de gradientes orientados. La imagen se divide en pequeñas celdas cada una de las cuales acumula direcciones del histograma de gradiente u orientaciones de los bordes de los píxeles de las celdas. Se recomienda para una mejor respuesta normalizar el contraste en unas zonas más grandes (denominadas bloques) y utilizar dicho resultado para normalizar las celdas del bloque. Estos bloques de descriptores normalizados son lo que los autores denominan descriptores HOG. Los descriptores HOG están inspirados en las características SIFT.



Por último, se utilizan los descriptores HOG de la ventana de detección como entrada a un clasificador SVMLight, que es una implementación de un SVM (Support vector machine) adecuada para trabajar con grandes conjuntos de datos. SVM es una técnica para entrenar clasificadores capaces de discernir entre patrones de la clase a detectar y cualquier otro patrón. El algoritmo SVM presenta el problema de entrenamiento como un problema de encontrar entre todas las superficies de separación posibles la que maximiza la distancia entre los elementos más próximos de dos clases. Esto lo consigue resolviendo un problema de programación cuadrática.

Los autores remarcan como principal ventaja de estos descriptores que captan estructuras características de la forma a detectar y que, al trabajar con celdas pequeñas, se mantiene invariante ante transformaciones exceptuando la orientación, por lo que es adecuado para detectar personas que se encuentren en posición vertical.

Otro método holístico es el presentado por Papageorgoiu y Poggio. En él se extraen características y se clasifican con un SVM. Las características son en este caso Haar Wavelets (HW). Los Haar Wavelets se pueden definir como un filtro que calcula la diferencia de nivel de gris entre dos zonas definidas.

El segundo bloque de métodos son los que se determina la presencia de personas mediante una secuencia de imágenes consecutivas, estos son métodos de extracción de background (fondo), que mediante esta extracción eliminan el fondo localizando un objeto en movimiento.

Una vez localizado el objeto en movimiento, cabe destacar que los algoritmos más importantes y utilizados para la detección de personas en la actualidad son el modelo de mezcla de Gaussianas (del inglés *Mixture of Gaussians: MOG*) y el modelo de Bayes (del inglés *Foreground Detection based on background modeling and Bayes classification:* FGD). El MOG es un modelo de caracterización de los píxeles del fondo basados en el método de mezcla de Gaussianas. Se caracteriza porque tiene en cuenta

a la hora de modelar el fondo los posibles cambios de iluminación en la imagen, secuencias multimodales, objetos moviéndose lentamente, y el ruido introducido por la cámara. En cuanto al FGD propone un marco bayesiano para incorporar características espectrales, espaciales y temporales en el modelado de fondo. Deriva una nueva fórmula de la regla de decisión de Bayes para la clasificación de fondo y objeto. El fondo es representado usando estadísticas de las principales características asociadas con objetos de fondo estacionarios y no estacionarios.

6.3 Detección Facial

La detección de caras puede ser considerada como un caso específico de la detección de objetos. Lo que se pretende con la detección de caras es localizar la cara o las caras, si existen, y devolver su ubicación a la imagen y el tamaño de estas. Una aproximación para la localización facial se realiza mediante la detección de características faciales, como ojos, boca, nariz, cejas… También existen métodos que consideran la cara como un todo y buscan en la imagen caras sin considerar los diferentes elementos que la componen.

Existen varios factores que dificultan la localización de la cara. Uno de estos factores es la posición de esta respecto a la de la cámara; otro factor seria la presencia o ausencia de componentes estructurales como la barba, el bigote o unas gafas, de estos hay una gran cantidad con una gran variabilidad (color, tamaño, forma,..). La expresión facial, aunque afecta más al reconocimiento facial, también puede dificultar la localización facial. Otros factores como la oclusión parcial de la cara por objetos o por otras caras, o simplemente condiciones de iluminación y características de la cámara, como puede ser el sensor, afectan a la localización de un rostro.

Métodos de detección de caras

Aproximaciones a características no variables

Estos algoritmos tienen como objetivo encontrar características estructurales que permanecen constantes por variaciones de posición, de punto de vista o de condiciones lumínicas variables para poder localizar las caras. Estos métodos son designados normalmente para la localización de caras. Los sistemas más utilizados son:

- *Facial Features*
- *Texture*
- *Skin Color*
- *Multiple Features*

Métodos de búsqueda de modelos

Constan de diversos patrones estándares de una cara que son almacenados para describir las caras como una parte entera o como características faciales. Las correlaciones entre una imagen de entrada y los patrones almacenados son utilizadas para la detección. Los sistemas más utilizados son:

- Predefined
- Face Templates
- Deformable Templates

Métodos basados en el aspecto

En contraste con los otros métodos, estos son generados desde una colección de imágenes de entrenamiento de las cuales han de capturar las variaciones representativas del aspecto final. Los sistemas más utilizados son:

- Eigenface
- Distribution-based
- Neural Network
- Support Vector Machine
- Naive Bayes Classifier
- Hidden Markov Model

La detección de caras en la actualidad

De los modelos anteriores, el que más se utiliza en la actualidad son los métodos basados en el aspecto ya que son los que dan mejores resultados. Esto se debe a que en función de la variabilidad de la colección de imágenes o muestras con las que se realiza en entrenamiento se obtendrían detectores con tasas altas de detección y bajas tasas de falsos positivos. Además de una gran robustez, presentan una eficiencia en el sistema de detección y reducción de coste computacional.

Encara V2.2

Encara V2.2 es la librería empleada en este proyecto para la detección de rostros, esta librería detecta rostros múltiples en tiempo real en sistemas de secuencias de vídeo. Ha sido diseñada y desarrollado en el Instituto Universitario SIANI y en el Departamento de Informática y Sistemas de la ULPGC.

El sistema consigue buenos resultados proporcionados por un planteamiento de la ventana deslizante, la combinación de diferentes señales disponibles en secuencias de vídeo debido a la coherencia temporal. Los resultados alcanzado por esta solución combinada consiguen una tasa de éxito para la detección de rostros del 98% para alrededor de 27.000 imágenes, proporcionando, además detecciones de ojos y una relación entre las detecciones sucesivas en el tiempo por medio de hilos de detección.

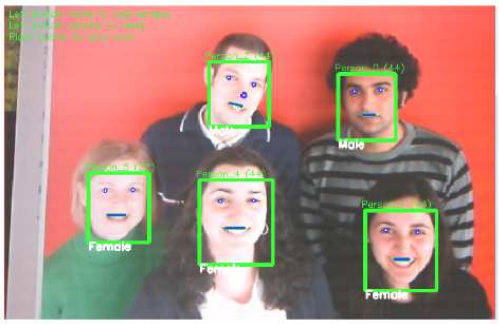


Imagen tras detector facial de Encara

Encara es un sistema de visión en tiempo real que va más allá de los detectores de cara imágenes fijas tradicionales. El sistema resultante es un enfoque robusto en tiempo real de detección de rostros múltiples multirresolución que combina diferentes señales sobre la base de una clara relación que existe entre los frames.

El resultado logra mejores tasas de detección para el procesamiento de flujo de vídeo y el costo de procesamiento más eficiente que en cualquier sistema libre de detección de rostros disponibles.

Encara se basa en dos enfoques para la detección de rostros:

- Uno basado en el diseño ó implícito: que es el que se consigue mediante una búsqueda exhaustiva de un patrón previamente aprendido en todas las posiciones y diferentes escalas de la imagen de entrada. Esta información previa se obtiene en gran parte de la librería Abra Computer Vision Library (OpenCV). A partir de aquí se consigue un clasificador en cascada que amplía la capacidad de búsqueda y ofrece diferentes variables para impulsar el aprendizaje. Ejecutándose cada vez clasificadores más complejo, permitiendo descartar regiones fondo de la imagen más rápidamente, mientras que al avanzar en los frames se consigue rapidez, basándose en resultados de frames anteriores.
- Otro basado en el conocimiento ó explícito: estos métodos aumentan la velocidad de búsqueda basándose en datos como el color, geometría y apariencia de la cara obtenidos a través de un previo estudio implícito.

7. Diseño

7.1 Algoritmo Empleado

En este punto se explicará un en líneas generales el algoritmo seguido para el proceso principal.

La interfaz gráfica contiene diversos botones, uno de ellos, el principal cuya etiqueta es "comenzar visualización", es el que pone en marcha el proceso principal al ser pulsado, una vez sucedido esto, se empezarán a obtener frames uno tras otro ya sea de un archivo de vídeo seleccionado o de la propia webcam por defecto. Diferenciaremos dos estados (según esté activa o no la variable trabajar): uno en el que únicamente se visualiza del vídeo normalmente sin aplicarle ningún tipo de proceso de detección, y otro aplicándole el propio proceso de detección. Durante el propio proceso se podrá cambiar de un estado a otro picando en el botón pertinente.

Inicialmente al abrir el vídeo y obtener el frame correspondiente, se reduce a un tamaño estándar, en caso de que dicho frame sea demasiado grande.

En caso de que nos encontremos en el estado de procesamiento de vídeo se pasa el detector de personas de OpenCV a dicha imagen, de tal modo que obtendremos un vector de rectángulos con la información del rectángulo donde se ha encontrado cada supuesta persona. Se pasará el detector de personas con un coeficiente bastante permisivo predefinido en una constante, para intentar detectar a todas las personas presentes, posteriormente se descartará los falsos positivos de personas con el detector facial.

Se creará una imagen con el recorte de cada persona detectada y se cambiará a un tamaño estándar predefinido por una constante. Por si el rectángulo de la persona detectada fuera muy pequeño que se amplíe para facilitar la detección de rostros del paso posterior y además para que el detector de caras trabaje con un único tamaño de imagen.

Para el estudio realizado en este proyecto y futuros estudios se tiene una función que guarda los datos referidos a cada persona en un archivo, para posteriormente ser visualizados en tablas de un archivo de Excel, datos como las coordenadas de cada persona encontrada. En el funcionamiento habitual del programa no se produce la creación del archivo y guardado de los mencionados datos, tendrá que estar definido la macro #define MODO_PRUEBAS para que esto se produzca. Una vez obtenida la imagen de la persona recortada y ampliada, se la pasamos al detector de caras.

Este devuelve un vector con la información referente a cada cara detectada en cada persona (podrían haber muchas hipotéticas caras dentro del rectángulo de la imagen de la supuesta persona).

Igual que antes, solo si estuviera definido #define MODO_PRUEBAS la información de cada supuesta cara será guardada en el archivo, junto con la información referente a cada persona detectada.

Más tarde habrá que comprobar si alguna de las caras detectadas está correctamente situada dentro de la persona, para ello trabajaremos con dos coeficientes preestablecidos:

- uno el del margen inferior, la cara deberá estar a una altura concreta dentro del cuerpo.
- otro el del los márgenes laterales, la cara deberá estar más o menos centrada dentro del cuerpo, con lo cual habrá unos márgenes laterales a ambos lados.

Si alguna de las hipotéticas caras detectadas está correctamente ubicada, determinaremos que esa detección es correctamente una persona de cuerpo entero. Luego habrá que dibujar su rectángulo sobre el frame inicial obtenido del vídeo, pero primero debemos de calcular si es una persona recién aparecida o estuvo presente en imágenes anteriores.

Por lo que luego compararemos el rectángulo que encierra a la persona detectada con los de las detecciones previas presentes en la lista de personas detectadas anteriormente en el vídeo que podrían estar presentes en la imagen (mediante sus coordenadas en la imagen original). Si su área coincide en un porcentaje preestablecido, con alguna de dichas personas presentes determinaremos que es la misma y actualizaremos la información contenida en ésta. Este algoritmo de seguimiento es válido porque se espera que las personas no se muevan excesivamente rápido, para que entre el área del rectángulo que encierra a la misma persona, de dos imágenes consecutivas, exista cierta intersección.

Al reconocerse la persona dibujaremos un rectángulo sobre ella en la imagen original, dicho rectángulo será del mismo color que el dibujado sobre la misma persona en la imagen anterior. En caso de ser una persona nueva, añadiremos información referente a dicha persona en la ya nombrada lista (hora aparición, coordenadas en la imagen…..) y le otorgaremos además un color para dibujar el rectángulo, color generado aleatoriamente. Cada vez que encontremos a la misma persona en frames posteriores será encuadrada con el mismo color.

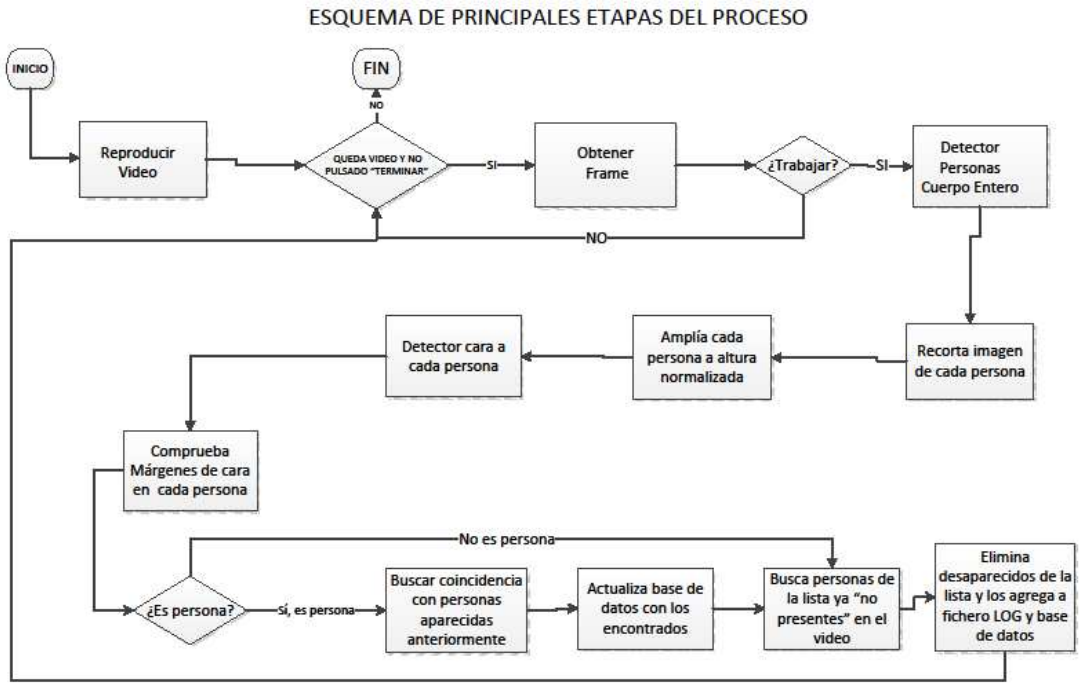
Una vez buscadas todas las personas presentes en el frame, hay que actualizar la lista, para aumentar el número de frames sin aparecer de las personas que no aparecieron

en el actual frame, y eliminar de la misma a las personas que superaron el número máximo de frames sin aparecer, ya que en la lista solo se guardan las personas que actualmente pueden estar presentes, se permiten unos pocos frames consecutivos sin aparecer para evitar que un fallo de detección suponga la consideración de una nueva persona.

Cuando se va a eliminar una persona de la lista, se deben incluir sus datos tanto en la base de datos, como en el fichero plano (fichero de Log). De tal modo que estos dos sistemas contendrán siempre la misma información. Estos datos contendrán el ID de la persona, el frame de entrada, el frame de salida, hora de entrada y hora de salida.

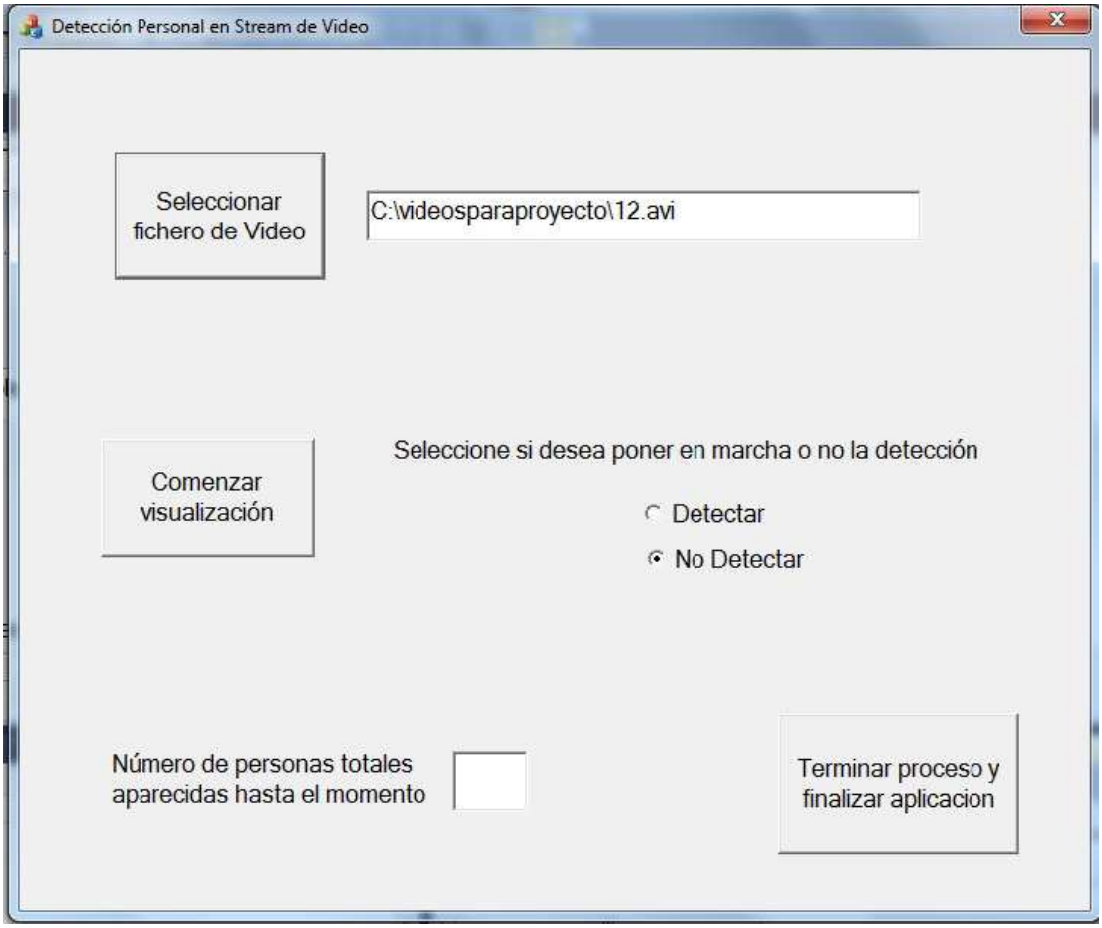
Por último se muestra por pantalla en una ventana aparte la imagen con los recuadros sobre cada una de todas las personas detectadas, dando apariencia de vídeo de detecciones. Y si está definido MODO_PRUEBAS esta misma imagen se guarda en la carpeta pertinente.

7.2 Diagrama de etapas del proceso



7.3 Interfaz Gráfica

Se ha diseñado una interfaz gráfica bastante directa e intuitiva a la vez que sencilla, es la que se muestra a continuación:



La interfaz contiene en la parte superior el botón de "seleccionar fichero de vídeo", (en caso de no seleccionar ninguno será obtenido la imagen desde la webcam por defecto). A la derecha de este botón se encuentra un recuadro que mostrará cual es el fichero seleccionado para visualizar.

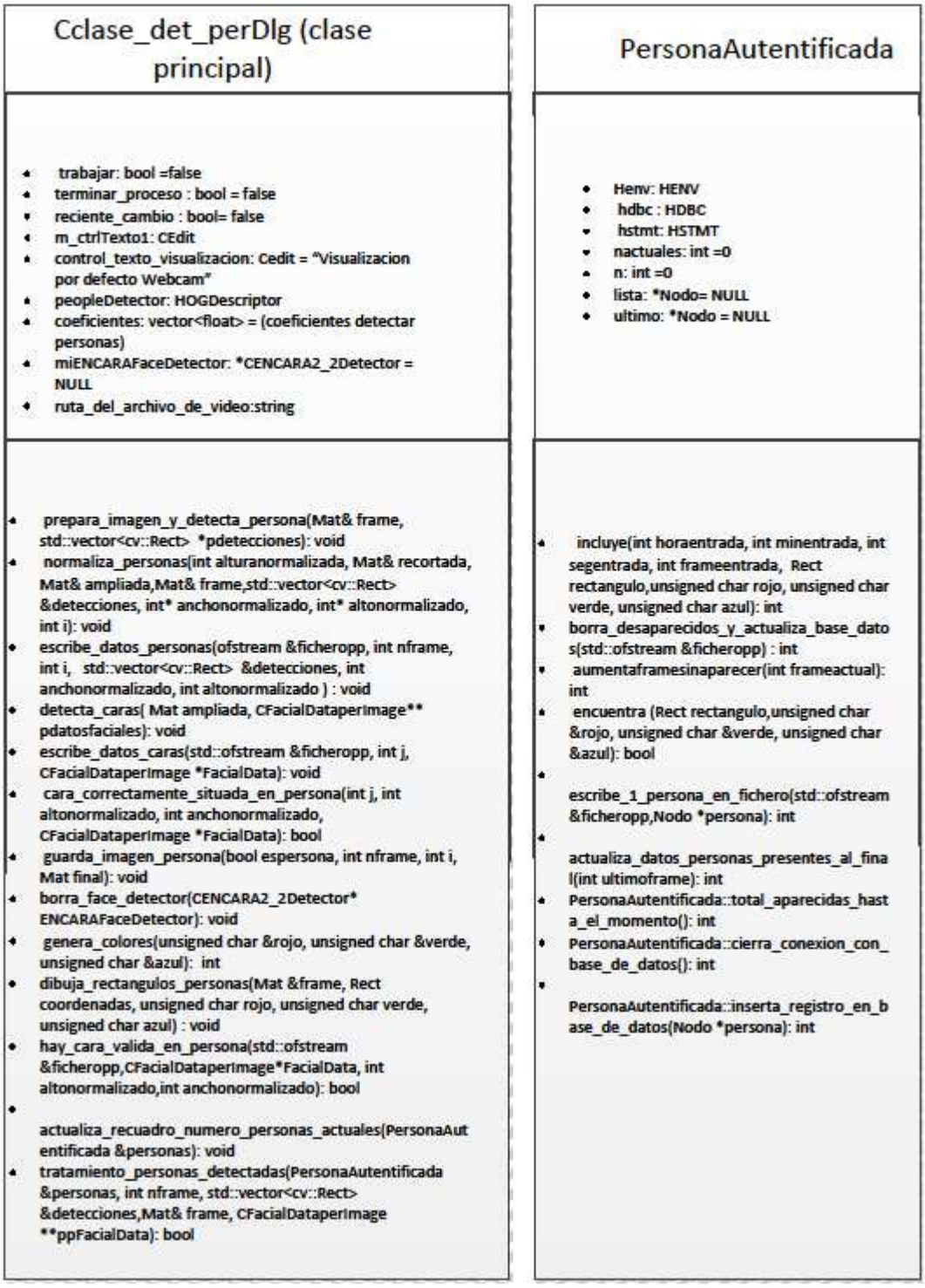
El botón de "Comenzar la visualización", que es el que activa todo el proceso, al pulsarlo, en un primer momento se abrirá la ventana y en caso de estar seleccionada la opción "No Detectar" solo se visualizará el vídeo tal cual, hasta que se pulse el radio-botón de "Detectar", y comience la detección, en ese momento se seguiría visualizando el vídeo, pero aplicando la detección personal aplicada en este TFG y por tanto dibujando los recuadros sobre las personas encontradas.

Cuando se pulsa el radio-botón de "No Detectar" se detendría el proceso de detección, volviendo al estado de solo visualización.

En la parte inferior de la interfaz se encuentra un pequeño recuadro que va mostrando el número de personas totales detectadas hasta el momento.

Por último el botón de “Terminar el proceso y finalizar aplicación”, detiene toda visualización y detección, finalizando el proceso y cerrando la ventana de la aplicación.

7.4 Diagrama UML Clases Proyecto



7.5 Descripción de variables de la clase principal

Las variables generales del trabajo, definidas en el archivo TFG_det_petDLG.h son parte de la clase Cclase_det_perDlg (clase principal). Estas son:

```
bool trabajar;//variable que mientras esta a “true”, se estará realizando la
detección de personas en el Vídeo, al estar a false, simplemente se estará
visualizando el vídeo
```

Esta variable es controlada por el botón “Detectar” y el “No Detectar”

```
bool reciente_cambio; //variable que se pone a true, tras un cambio de “detectar”
a “No Detectar”, para que se elimine la lista
```

La razón de esta variable es para que cuando se ponga a true, se vacíe la lista de las personas presentes en ella (ya que se va a parar la detección aunque se sigue visualizando el vídeo) y se guarde los datos de todas estas personas presentes en la lista en el respectivo fichero y en la base de datos.

```
bool terminar_proceso; //variable que se pone a true al pulsar el botón de
finalizar proceso y termina el mismo.
```

Esta variable se inicializa a “false” al ser pulsado el botón de “Comenzar visualización”; se cambia su valor a “true” al ser pulsado el botón de “Terminar proceso”

```
CEdit m_ctrlTexto1; //esta es la variable de control del recuadro de la interfaz
que muestra el número de personas aparecidas hasta el momento

CEdit control_texto_visualizacion; //variable de control del recuadro que muestra
el nombre del fichero a visualizar


HOGDescriptor peopleDetector; //objeto de la clase HOGDescriptor para crear un
detector de personas mediante el histograma de gradientes orientados de OpenCV

vector<float> coeficientes;// en este vector se guardan los coeficientes
predeterminados para que lo use el HOGDescriptor de OpenCV, que en
//este caso serán coeficientes acordes para encontrar personas de cuerpo entero.


CENCARA2_2Detector *miENCARAFaceDetector; //objeto de la clase CENCARA2_2Detector
usado por en la libería ENCARA para la detección facial


string ruta_del_archivo_de_vídeo; //variable string que guarda la dirección del
archivo de vídeo a abrir
```

7.6 Clase persona_autentificada: variables y funciones miembro

Para el almacenamiento de los datos sobre las personas encontradas, se creó una clase, que interiormente contiene una lista, donde cada nodo contiene la información de cada una de las personas encontradas en el vídeo.

Cada nodo de la lista contendrá por tanto variables referentes a cada persona, como son:

- Identificador de persona: número que identifica a la detección de manera unívoca
- Frame de entrada y de salida: frames de aparición y desaparición de la persona
- Horas de aparición y desaparición correspondientes los frames de entrada y salida (guardando cada componente de la hora como un número entero)
- tres parámetros unsigned char para cada una de las tres componentes que forman el color aleatorio generado para dibujar el rectángulo de dicha persona.
- Variable del tipo rectángulo con la información del rectángulo que contiene a la persona (al igual que otras variables, ésta irá cambiando conforme pasan los frames)
- Variable booleana que indica si la persona estuvo presente ó no en el último frame
- Variable que indica el número de frames sin aparecer de dicha persona (recordamos que la persona se considera como no presente en el vídeo cuando alcanza un número de frames consecutivos preestablecidos sin aparecer). Cada vez que reaparece la persona, se resetea este contador.
- Puntero al siguiente nodo de la siguiente persona (para conformar la lista).

La clase contiene funciones para el tratamiento de la misma.

- Función **`incluye`**: función que incluye un nodo de nueva persona en la lista y inicializa sus datos
- Función **`aumentaframesinaparecer`**: función que aumenta el contador del número de frames sin aparecer, para las personas que estuvieran en la lista y no se encontraran en el frame actual. Además rellena los datos de hora de desaparición de las personas que recién acaban de desaparecer (por si acaso no volvieran a aparecer).
- Función **`Encuentra`**: función que recorre la lista para encontrar una persona con un porcentaje de coincidencia de área con la actualmente buscada, según

el establecido por la constante; en caso de encontrar tal persona devuelve verdadero, y devuelve las componentes rojo, verde y azul con las que hay de dibujar el rectángulo en el nuevo frame.

- Función **actualiza_datos_personas_presentes_al_final**: Función ejecutada solo en determinadas ocasiones: tras la finalización del vídeo ó tras el paso de "detección" a "no detección" ó tras la terminación del proceso al pulsar el botón. Tras suceder alguno de estos tres casos hay que determinar como hora de desaparición y frame de salida, la del actual frame (para seguidamente ser eliminados de la lista y añadidos a la base de datos)
- Función **escribe_1_persona_en_fichero**: función que guarda en el fichero de log los datos de una persona que recientemente se ha eliminado de la lista.
- Función **inserta_registro_en_base_de_datos**: función que guarda los datos de una persona recientemente eliminada de la lista en la base de datos. O sea ejecuta la misma acción que la función anterior, pero en lugar de en el fichero de log, en la base de datos.
- Función **cierra_conexion_con_base_de_datos**: función que como su propio nombre indica, cierra conexión con la base de datos.
- Función **borra_desaparecidos_y_actualiza_base_datos**: función que elimina de la lista todos los registros de personas que han superado el máximo de frames permitidos sin aparecer en el vídeo y seguidamente llama a las respectivas funciones para insertar su información tanto en el fichero de log como en la base de datos.
- Función **total_aparecidas_hasta_el_momento:** devuelve el número "n" (total de personas aparecidas contabilizadas hasta el momento)

7.7 Constantes preestablecidas

En el código fuente se han establecido una serie de constantes con las que trabaja el programa. Estas constantes se encuentran definidas en el fichero TFG_det_perDlg.cpp

```
const string  ARCHIVO_DATOSPP = "c:/ archivos_ejecucion/datos_pruebas.xls";
//fichero datos obtenidos tanto de detecciones de caras como de personas. Está
pensado para ser abierto con Excel (esta constante solo se usa si está definido
el modo #define MODO_PRUEBA)

const string  FICHERO_APARICIONES_PERSONAS = "c:/archivos_ejecucion/log.xls";
//fichero de apariciones de personas "confirmadas"(fichero de log), alberga frame
de entrada y salida y hora de entrada y salida de cada persona
```

```
const int ALTURA_NORMALIZADA_PERSONA= 600;
//altura a la cual se amplía ó reduce cualquier detección de supuesta persona de
cuerpo entero en la imagen, para luego pasársela al detector facial

const string DIRECTORIO_PERSONAPP= "c:/archivos_ejecucion/imagenes/capturas/";
//directorio donde se guardarán las imágenes con las detecciones
//(esta constante solo se usa si está definido el modo #define MODO_PRUEBA)

const string DIRECTORIO_NO_PERSONAPP=
"c:/archivos_ejecucion/imagenes/sin_caras/";
//directorio donde se guardarán las imágenes de las supuestas personas detectadas
//por el detector de personas de cuerpo entero de OpenCV, pero no aceptadas como
tal porque no se detectó en ellas cara ó bien ésta no estaba correctamente
situada (esta constante solo se usa si está definido el modo #define MODO_PRUEBA)

const float TANTO_POR_UNO_COINCIDENCIA_AREA_PERSONA=0.4;
//tanto por uno de área de coincidencia de 2 personas detectadas en frames
consecutivos para que se considere la misma persona (0.4 = 40%)

const int FRAMES_PERMITIDOS_SIN_APARECER=6;
//número de frames consecutivos que han de transucrrir sin que se localice una
persona para que se considere que ya no está presente en la secuencia de vídeo

const double COEF_RIGUROSIDAD_PERSONA_CUERPO_ENTERO= -0.5;
// Variable para el detector de personas. A valor más pequeño más supuestas
personas detecta como personas de cuerpo entero.
//el valor "medio" es el 0. El rango habitual está entre el -1 y +1,
//usamos un valor bastante permisivo (-0.5) para intentar que localice a todas
las personas, aunque haya muchos falsos positivos,estos falsos positivos
intentaremos descartarlos posteriormente con el detector facial

const float MARGEN_LATERAL_CARA_EN_PERSONA= 0.25; ;//margen lateral que ha de
tener una cara dentro de la imagen de la persona de cuerpo entero, para que se
considere cara bien situada (significaría que ha de dejare un cuarto de imagen de
margen a cada lado)

const float MARGEN_INFERIOR_CARA_EN_PERSONA= 2.0/3; ;//margen inferior deberá ser
2/3 de la imagen total

const int ANCHO_MAX_IMAGEN= 640;//ancho estándar al que se pasará la imagen
original para luego trabajar con ella
```

7.8 OpenCV

7.8.1 Definición

OpenCV es una librería dirigida principalmente para el procesamiento de imágenes en tiempo real. OpenCV es una biblioteca libre de visión artificial originalmente desarrollada por Intel. Desde que apareció su primera versión alfa en el mes de enero de 1999, se ha utilizado en infinidad de aplicaciones. Desde sistemas de seguridad con detección de movimiento, hasta aplicativos de control de procesos donde se requiere reconocimiento de objetos. Esto se debe a que su publicación se da bajo licencia BSD, que permite que sea usada libremente para propósitos comerciales y de investigación con las condiciones en ella expresadas.

OpenCV es multiplataforma, existiendo versiones para GNU/Linux, Mac OS X y Windows y para diversos lenguajes de programación como C++, Python y Java. Contiene más de 500 funciones que abarcan una gran gama de áreas en el proceso de visión, como reconocimiento de objetos (reconocimiento facial), calibración de cámaras, visión estéreo y visión robótica.

El proyecto pretende proporcionar un entorno de desarrollo fácil de utilizar y altamente eficiente. Esto se ha logrado, realizando su programación en código C y C++ optimizados, aprovechando además las capacidades que proveen los procesadores multinúcleo. OpenCV puede además utilizar el sistema de primitivas de rendimiento integradas de Intel, un conjunto de rutinas de bajo nivel específicas para procesadores Intel (IPP).

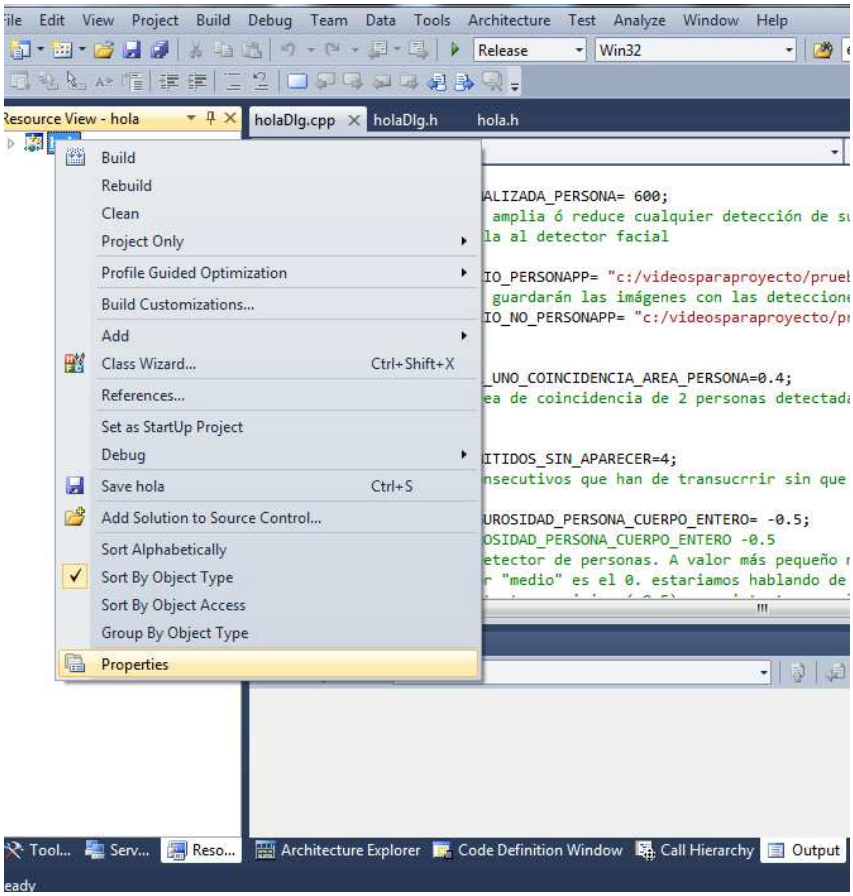
7.8.2 OpenCV: Histograma de gradientes orientados

El algoritmo Histograma de Gradientes Orientados (HOG) se usa en ámbitos como la visión por computador y el procesamiento de imágenes con el propósito de detectar objetos en una imagen. La esencia de dicho algoritmo es que la forma de un objeto en una imagen puede ser descrito por medio de la distribución de los gradientes El objetivo principal de esta técnica es la extracción de características de una imagen. Las características son extraídas teniendo en cuenta los bordes. Los contornos de la imagen se obtienen a partir del cómputo del gradiente en la imagen ya que los bordes se corresponden con los píxeles con valor de gradiente más alto.

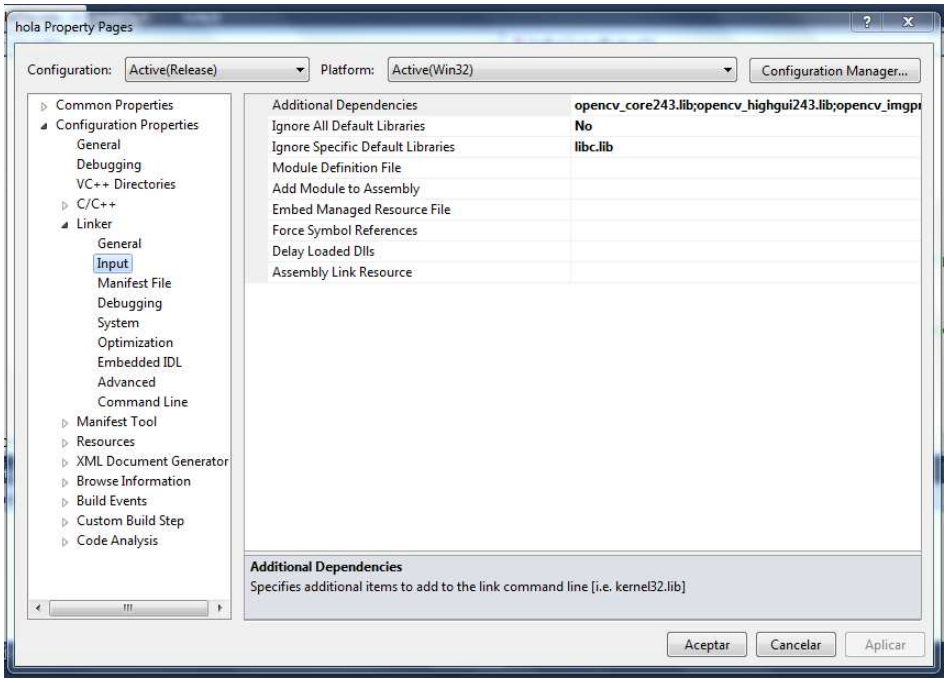
El proceso de extracción de características es el proceso más importante del proyecto, de él depende en gran medida el cumplimiento de los objetivos marcados. Esto se debe a que estamos extrayendo características de una imagen para que la representen. Pero esas características tienen que ser las que se precisen para el propósito de este trabajo. No sirve, por ejemplo, que las características extraídas representen el color de piel de una persona. Lo que se exige es que las características representen la silueta de una persona, para poder discriminar de forma precisa que se está presente ó alguna de sus características.

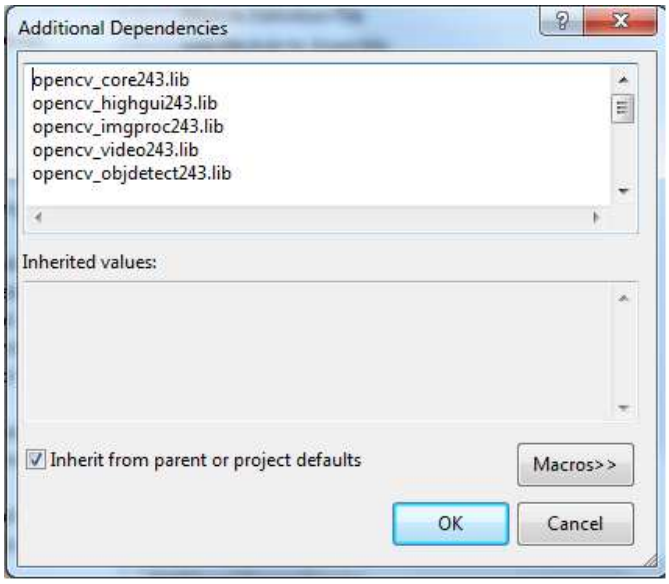
7.8.3 Agregar OpenCV al proyecto

La forma de agregar openCV en Visual Studio es agregando los directorios donde se encuentra las librerías en **`propiedades-> Additional Include Directories.`**



Y agregando las propias librerías para en **`linker->Input`**





7.8.4 Uso en el proyecto

Una vez tenemos las librerías agregadas, ya podemos usar las respectivas clases que nos facilita OpenCV, que para nuestro caso concreto será la clase "HOGDescriptor" (Histograma de Gradientes Orientados). Cada objeto de esta clase contiene parámetros ajustables como el tamaño del recuadro a buscar, tamaño de la ventana…. y diversas funciones para detectar. En nuestro caso el objetivo será usarlo para la detección de personas "de cuerpo entero".

Ejemplo persona detectada por OpenCV

Una vez declaremos el objeto de la clase HOGDescriptor, debemos obtener el vector de coeficientes con el que trabajará la clase, para que sea una persona de cuerpo entero, y no otro objeto de búsqueda en la imagen. Esto lo haremos con la función, `getDefaultPeopleDetector();` perteneciente también a dicha clase.

```
vector<float> coeficientes;
coeficientes = peopleDetector.getDefaultPeopleDetector();
//en este vector se guardan los coeficientes predetermina

peopleDetector.setSVMDetector(coeficientes);
//le pasamos los coeficientes al HOG
```

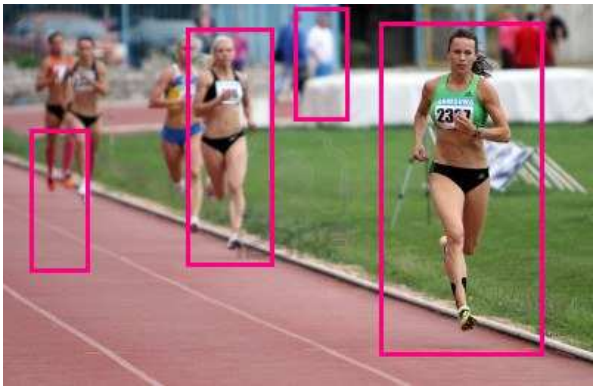
El detector puede trabajar con imágenes en color y en blanco y negro pero se ha optado en este TFG en no utilizar las de color porque no se considera que aporte información adicional, por lo que el primer paso será pasar la imagen obtenida a blanco y negro.

```
cvtColor(frame, imagen, COLOR_BGR2GRAY);
//frame es la original
//imagen es en blanco y negro
```

Por último se realiza el proceso de detección mediante la función: **detectMultiScale**

```
peopleDetector.detectMultiScale(imagen,*pdetecciones,
COEF_RIGUROSIDAD_PERSONA_CUERPO_ENTERO,     cv::Size(8,8),   cv::Size(0,0),   1.05,
2.0);

//pdetecciones es un puntero a un vector de rectángulo que contendrá al final la
información del rectángulo de cada una de las personas detectadas.
// COEF_RIGUROSIDAD_PERSONA_CUERPO_ENTERO es una constante preestablecida sobre
el umbral de detección de personas de cuerpo entero.
// 1.05 en cuanto va aumentado la ventana poco a poco la ventana en la que
búsqueda, cada vez aumenta un 5% el tamaño de la ventana en que busca una persona
// el 2 significa que hasta el doble de tamaño llegaría empezando por 64*128
```

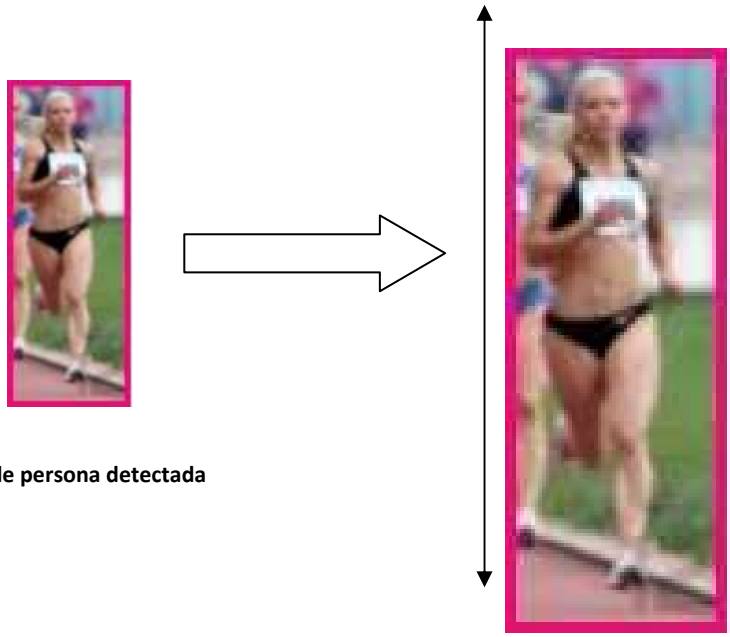


Ejemplo personas detectadas por OpenCV

7.9 Encara: Aplicación del detector facial

Una vez obtenemos las múltiples detecciones de personas obtenidas por el detector de personas de OpenCV se procede a usar el detector de caras.

Para cada persona obtenida, se crea una imagen cuyo contenido es un recorte de dicha persona, pero ampliada a una altura estándar predefinida en la constante **ALTURA_NORMALIZADA_PERSONA**

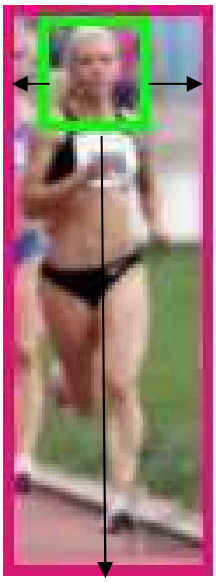


Recorte de persona detectada

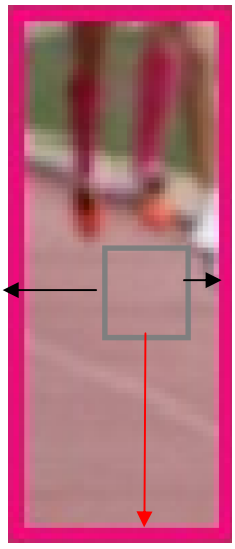
Imagen con altura normalizada .Esta imagen es pasada al detector facial

Imagen tras aplicarle el detector facial

Al encontrar una supuesta cara, se comprueba que cumple unos márgenes establecidos por las constantes en el código, tanto el margen inferior como los márgenes laterales



Si estos tres márgenes se respetan, se concluye que es realmente una persona lo que se ha detectado; en caso de que no haya ninguna cara presente o que no cumpla los márgenes se concluye que no es persona.

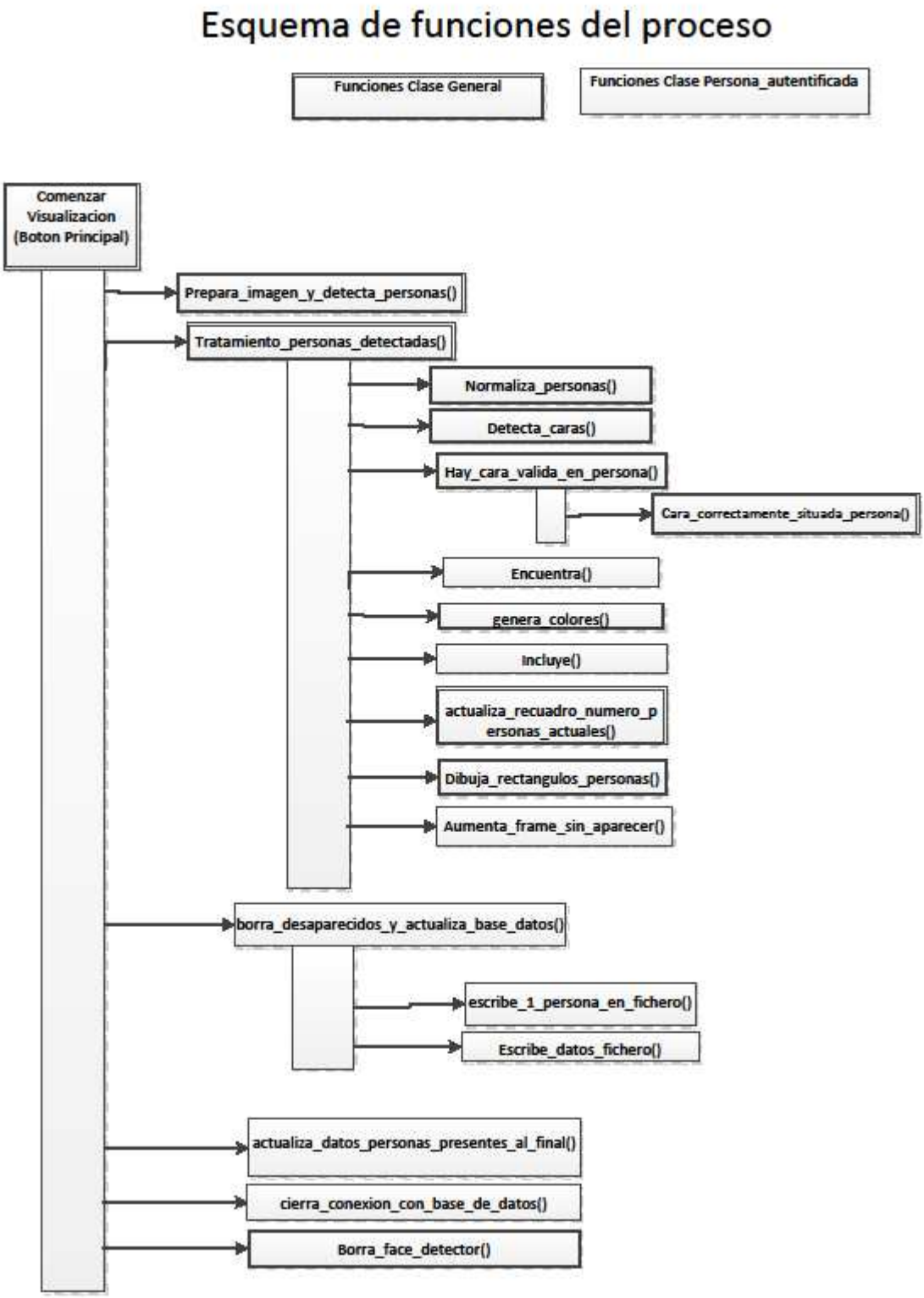


Descartada por insuficiente margen inferior de la supuesta cara

Descartada por ausencia de cara

Estas dos últimas detecciones de hipotéticas personas, serían descartadas.

7.10 Diagrama UML secuencial de la ejecución de funciones



7.11 Descripción de funciones asociadas a la clase principal del proyecto

En esta sección se explicarán cada una de las funciones añadidas a la clase principal del proyecto, clase que viene creada por defecto y cuyo nombre es **Cclase_det_perApp.**

Función **prepara_imagen_y_detecta_persona**: función que recibe la imagen, la pasa a escala de grises

```
cvtColor(frame, imagen, COLOR_BGR2GRAY);
```

y luego la pasa al detector de personas que devuelve en un vector de rectángulos las coordenadas de cada persona encontrada.

```
peopleDetector.detectMultiScale(imagen,*pdetecciones,
COEF_RIGUROSIDAD_PERSONA_CUERPO_ENTERO,     cv::Size(8,8),   cv::Size(0,0),   1.05,
2.0);
```

Función **normaliza_personas**: función que recibe la imagen original y el vector con los datos referentes a las detecciones de personas, y crea una imagen recortando a la persona número "i" luego amplia esa imagen hasta la altura normalizada, calculando el coeficiente de ampliación para aplicarlo también al ancho de la imagen. Devolviendo la imagen recortada y la ampliada que es la que usaremos posteriormente.

Función **escribe_datos_personas**: función que guarda en un fichero los datos relativos a la detección de cada persona para posteriormente ser visualizados en un fichero Excel. Esta función solo se ejecuta si esta activo el MODO_PRUEBA.

Función **detecta_caras**: función que carga los datos de la imagen en la que se va a buscar un rostro, en el objeto de la clase CENCARA2_2Detector, en el caso de que estuviéramos en la primera imagen a buscar rostro. Luego se procede a la detección facial en la imagen que se le ha pasado, que en nuestro caso debería de ser la imagen de una hipotética persona detectada por el detector de personas de cuerpo entero

```
Process( ampliada, miENCARAFaceDetector );  //proceso de detección de rostros
```

Y por último carga en el mismo objeto los resultados de la detección facial en la imagen

Función **cara_correctamente_situada_en_persona**: función booleana que comprueba si la cara encontrada en la supuesta persona cumple con los márgenes

previamente establecidos de situación de la cara dentro del recuadro de la persona. Según los experimentos realizados se adoptaron valores de que la cara debe estar completamente situada en el tercio superior de la imagen y que debe haber ¼ de la imagen de margen tanto la derecha como a la izquierda).

Función **escribe_datos_caras**: función que escribe en el fichero los datos referidos a las supuestas caras halladas por el detector facial, este fichero será el mismo en donde se guardaron los datos referentes a las personas de cuerpo entero. Al igual que aquella función, esta solo se ejecuta si está definido el MODO_PRUEBA.

Función **hay_cara_valida_en_persona**: función que para cada una de las caras detectadas en la persona (si es que las hubiera) llama a la función que comprueba que estén correctamente situadas y a la que escribe los datos pertinentes. Devuelve verdadero en el caso de que encuentre alguna cara bien situada en la persona, devuelve falso en caso contrario.

Función **Guarda_imagen_persona**: función que guarda la imagen con los recuadros de las personas encontradas en el directorio pertinente. Esta solo se ejecuta si está definido el MODO_PRUEBA.

Función **borra_face_detector**: Función que libera la memoria ocupada por el objeto de la clase CENCARA2_2Detector (detector facial)

Función **Tratamiento_de_personas_detectadas**: función general que realiza un amplio número de tareas recurriendo a otras funciones. En resumen se puede decir que para cada una de las personas detectadas, crea una imagen de dicha persona normalizada, y busca en ella si hay una cara correctamente situada, en tal caso busca coincidencias anteriores de esa persona en la lista o añade información si fuera una nueva persona.

Función **Genera_colores**: devuelve un valor de rojo, verde y azul aleatorios de entre 0 y 255 para el color con el que se recuadrará a la supuesta persona

Función **`Dibuja_rectángulos_personas`**: dado un frame, las coordenadas de un rectángulo y las tres componentes del color, dibuja un rectángulo de dicho color en la imagen original.

Función **`actualiza_recuadro_numero_personas_actuales:`** escribe en el recuadro de la interfaz el número de personas totales aparecidas hasta el momento.

7.12 Base de datos

7.12.1 Servidor y Base de datos

La base de datos se ha implementado con el XAMPP, que como ya se explicó en apartados anteriores es un servidor independiente de plataforma que contiene MySQL, y demás servicios. Se decidió utilizar XAMPP ya que al contener más servicios proporcionaba más opciones de mejora en caso de una supuesta ampliación del proyecto, pero realmente hubiera valido el uso de simple servidor MySQL.

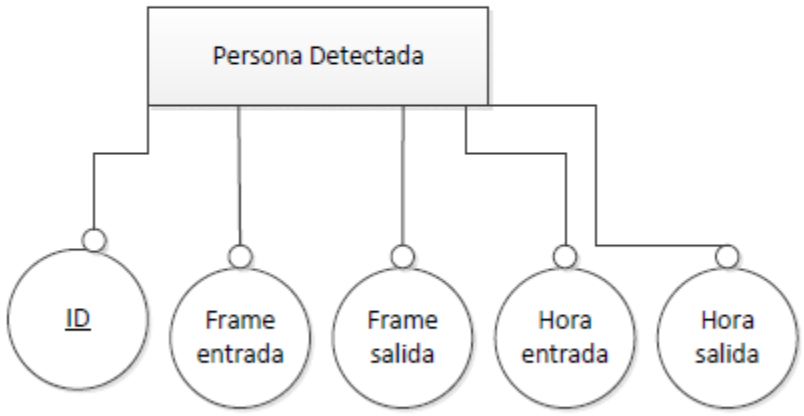
La base de datos contenida en el proyecto es sencilla, es monotabla, con 5 campos:

- ID (entero)
- Frame entrada (entero)
- Frame salida (entero)
- Hora entrada (TIME)
- Hora salida (TIME)

Donde la clave primaria de la tabla es el campo ID, que es el identificador de persona generado de forma consecutiva para cada una por el programa principal. Esta información es la misma que se encontrará contenida en el fichero de log.

7.12.2 Diagrama Base de Datos

Diagrama de tabla de la base de Datos



7.12.3 Interconexión Servidor-programa principal

La forma de acceder del programa principal a la base se ha realizado mediante ODBC (Open DataBase Connectivity). Por tanto una vez instalado el ODBC, hubo que crear un DSN (Data Source Name) para proporcionar referencia a la base de datos para trabajar por conexión ODBC. En los DSN se especifican los datos que necesita Windows para conectarse con una base de datos, como el nombre del servidor o el origen de datos, cadena de conexión, dirección IP, puerto, usuario y contraseña.

Una vez configurado el ODBC, hay que establecer la conexión mediante código en el programa principal. La forma empleada en el proyecto fue que en el propio constructor de la clase **`PersonaAutentificada`** (clase del objeto que contiene la lista de personas presentes en el vídeo) establece conexión con la base de datos

Primero se adquiere un identificador único de entorno o **henv**. Este identificador representa únicamente el uso del subproceso de ODBC

```
SQLAllocHandle(SQL_HANDLE_ENV, SQL_NULL_HANDLE, &henv);
```

Tras obtener un identificador de entorno, necesitamos establecer la versión de ODBC admitida por la aplicación. Esta tarea se realiza con la siguiente función:

En él se establece que la versión de ODBC admitida es la 3.0 (SQL_OV_ODBC3)

```
SQLSetEnvAttr(henv, SQL_ATTR_ODBC_VERSION, (void *) SQL_OV_ODBC3, 0);
```

Antes de que una aplicación pueda conectarse a un origen de datos, debe adquirir un identificador de conexión a una base de datos (hdbc) para ODBC. Esta tarea se realiza con la siguiente función; donde el primer argumento es la constante SQL_HANDLE_DBC, el segundo argumento debe ser un identificador henv válido, adquirido en el paso 1. El tercer y último argumento debe ser un puntero a una variable de tipo SQLHDBC.

```
SQLAllocHandle(SQL_HANDLE_DBC, henv, &hdbc);
```

Una de las formas más usadas para establecer la conexión es usar la función SQLConnect, que es una manera directa y sencilla ya que con esta función solo se pasa el DSN, el Id. De Usuario y la contraseña a ODBC

```
SQLConnect(hdbc, (SQLCHAR*)"dnsTFG", SQL_NTS, (SQLCHAR*)"usuario",
SQL_NTS,(SQLCHAR*) "contraseña", SQL_NTS);
SQLAllocHandle(SQL_HANDLE_STMT, hdbc, &hstmt);
```

Luego cada vez que sea necesario la función **inserta_registro_en_base_de_datos** realiza tal efecto, ejecutando internamente la siguiente función:

```
rc = SQLExecDirect(hstmt, sqlfinal, SQL_NTS);//ejecución de sentencia SQL
```

Finalmente terminada la aplicación la función **cierra_conexion_con_base_de_datos** finaliza dicha conexión con la base de datos, invocando internamente a la siguiente función:

```
SQLDisconnect(hdbc); //desconexión de la base de datos
```

7.13 Almacenamiento datos personales

7.13.1 Almacenamiento en la lista

En el caso de detectarse una persona confirmada se procede al almacenamiento en la lista correspondiente.

Primero hemos de buscar si la persona encontrada, se hallaba presente en el stream de vídeo, y por tanto en la lista, para ello empleamos la función **encuentra**. En caso de encontrarla, se actualizan los datos de la persona en la lista y se dibuja en el frame el recuadro de la persona encontrada. En el caso de no estar presente, habría que añadirla a la lista, insertando un nuevo nodo con la función **Incluye**. Generar un color

aleatorio con el que dibujar el rectángulo alrededor de la persona y guardar dicho color en la información de la lista.

Una vez se han comprobado todas las personas presentes en el frame actual, se procede a tratar las presentes en el vídeo que no se encontraran en el frame actual, aumentando su número de frames sin aparecer y eliminando a aquellas que superen el máximo de frames permitidos sin aparecer, para finalmente agregar sus datos al archivo de log y la base de datos.

7.13.2 Almacenamiento de datos de pruebas en fichero Excel

El almacenamiento de datos referentes a las detecciones de personas y de caras solo se efectúa si se encuentra definido el MODO_PRUEBAS. Este fichero se almacenaría en **"c:/archivos_ejecucion/datos_pruebas.xls"**; tal como se indica en la constante **ARCHIVO_DATOSPP**

Los datos referentes a las coordenadas de las supuestas personas encontradas por el detector de personas de OpenCV y los datos referentes a las supuestas caras encontradas en cada persona son guardados en un archivo de Excel para su posterior análisis.

Esta generación de datos ha sido especialmente útil en el las pruebas y elaboración de este TFG; pudiendo ser usada también para posteriores desarrollos e investigaciones.

Un ejemplo de archivo de datos generado sería el siguiente:

imagen_50000____persona0			
x1	y1	ancho normalizado	alto normalizado
162	323	270	539
datos caras-0			
x1	y1	ancho	alto
115	89	52	49
imagen_50000____persona1			
x1	y1	ancho normalizado	alto normalizado
82	162	273	539
imagen_50000____persona2			
x1	y1	ancho normalizado	alto normalizado
77	153	271	540

Se observa las coordenadas de cada persona encontrada en cada imagen, coordenadas x1 e y1 referente a la esquina superior izquierda de la supuesta persona encontrada. Y ancho y alto al que se normalizó la persona encontrada para luego aplicarle el detector

de caras; la altura normalizada ha de ser muy próxima sino igual a la altura normalizada especificada en las constantes.

Seguidamente a los datos de cada persona se encuentra una fila con los datos de cada cara de la persona (si es que se hubiera encontrado cara en la persona), estos datos incluyen coordenada x1 e y1 referentes a la esquina superior izquierda de la cara y ancho y alto de la propia cara. Cabe señalar que las coordenadas referentes a la esquina superior izquierda de la cara son coordenadas relativas a la imagen de la persona normalizada, no son relativas a todo el frame.

Se prefirió repetir en cada línea la información de cada columna para facilitar mejor la lectura al estar mezclados datos de caras y personas.

7.13.3 Almacenamiento de apariciones en fichero de log y base de datos

A diferencia del fichero de pruebas, este fichero si se crea y se almacenan los datos pertinentes, esté o no definido el MODO_PRUEBAS. Este fichero se almacena en **"c:/archivos_ejecucion/log.xls"** tal como se indica en la constante **FICHERO_APARICIONES_PERSONAS**

Aquí se observa un ejemplo de fichero de log:

ID de persona	frame de ent	frame salida	hora entrada	hora salida
0	15	18	0:44:50	0:44:53
1	27	27	0:45:11	0:45:12
2	31	31	0:45:19	0:45:19
3	46	46	0:45:45	0:45:46
4	50	50	0:45:51	0:45:52
5	60	69	0:46:07	0:46:21

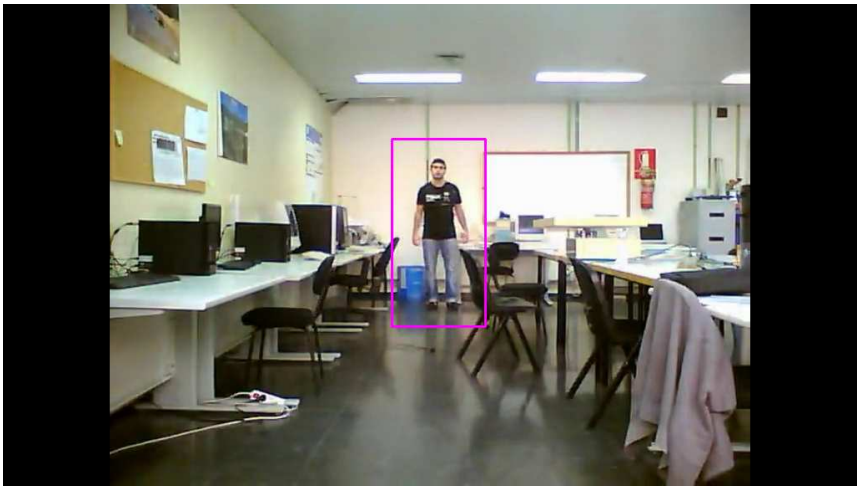
El cuadro contiene la información básica de cada persona extraída de la lista, como es el id asignado de forma automática y correlativa a cada persona, el frame entrada (primer frame donde apareció la persona), frame de salida (frame en el que se consideró que la persona dejó de estar presente), hora de entrada y hora de salida correspondientes a la persona y por tanto también al frame. Toda esta misma información es almacenada en la base de datos de la misma manera

8. Pruebas y Resultados

8.1 Prueba para determinar el umbral de detección de personas de cuerpo entero

Tras las pruebas para entender y comprobar el funcionamiento del detector de personas de OpenCV, primero realizado sobre una imagen y luego ya sobre un stream de Vídeo, se realizaron las pruebas para determinar el valor de la constante del umbral de detección de personas de cuerpo entero.

Primero se comprobó con un valor 0.0 que es el que está defecto. Se comprueba que con este coeficiente se detectan bien algunas personas, pero se dejan de detectar otras, (ver segunda imagen) siendo el número de falsos positivos reducido. Los falsos positivos en este caso son reconocimientos de personas por parte del programa, pero que nosotros sabemos que realmente no lo son, por la simple observación de la imagen, son por tanto fallos del programa.

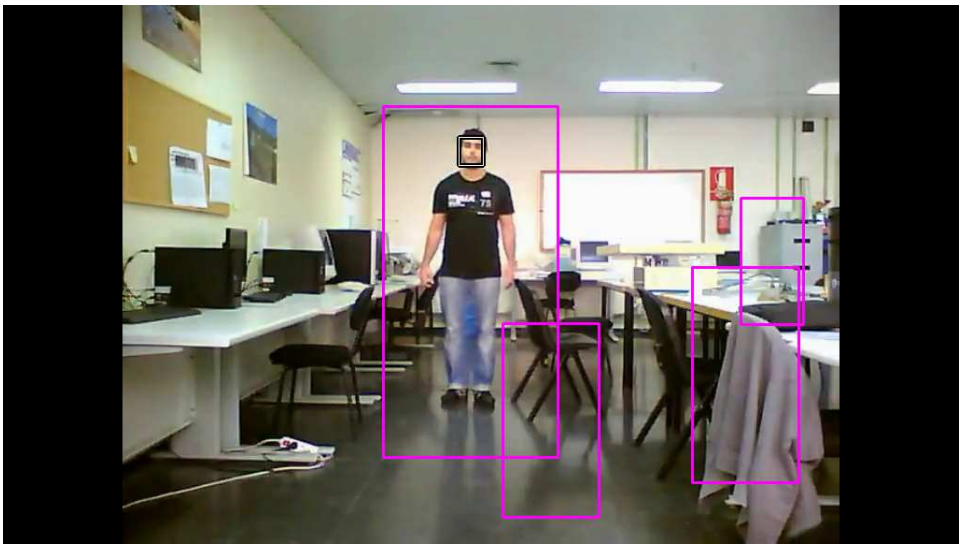


Detección de persona OpenCV con coeficiente 0.0

Detección de persona OpenCV con coeficiente 0.0

Dos imágenes obtenidas del mismo vídeo pasándole al detector de personas un coeficiente de 0.0, se observa como se obtiene en la segunda imagen un falso positivo, (el recuadro rosa que no es persona) y no se detecto como persona lo que debería; por lo que las siguientes pruebas consistieron en seguir disminuyendo el umbral de detección hasta llegar a un valor que se detecte a todas las personas de cuerpo entero de frente.

Tras numerosas pruebas se determinó como valor para el umbral de detección -0.5 que es un valor que aunque produce un elevado número de falsos positivos, detecta a todas o a casi todas las personas de cuerpo entero presentes



Detección de persona OpenCV con valor de umbral -0.5

Se observa como ahora sí que detecta a la persona donde antes no la había detectado, pero aumentando el número de falsos positivos. Estos se intentarán discriminar posteriormente en futuras pruebas con la inserción del detector de caras.

8.2 Pruebas para determinar el tamaño al que se ampliará cada detección de persona

La siguiente prueba que se realizó fue para detectar el tamaño al que se debe ampliar una detección de supuesta persona de cuerpo entero para facilitar el trabajo del detector facial.

Para ello se realizaron diversos vídeos de prueba, de personas acercándose a la cámara lentamente y analizar el tamaño de dichas personas de cuerpo entero en las que se detecta la cara perfectamente con el detector facial.

Se considera que la cara es perfectamente detectada con el detector facial, cuando éste la recuadra de color verde. Aunque posteriormente en el funcionamiento normal de la aplicación tomaremos como cara válida cualquier posible cara correctamente situada, aunque no sea enmarcada como una cara 100% segura por el detector facial. Este las recuadra de color gris, cuando la probabilidad de que sea un rostro no es muy elevada.

Siendo los siguientes los resultados obtenidos en diferentes vídeos.

Numero prueba	Número de frame	Tamaño Cara	Tamaño Persona
Prueba 1	450	35x40	251x502
Prueba 4	101	37x42	240x480
Prueba 5	332	44x50	231x461
Prueba 6	549	41x47	224x449

Por lo que se concluyó según la tabla se debía normalizar a una altura estándar de 480 pixeles cada detección de persona, quedando el ancho de aproximadamente 240, ya que quedará ampliado de manera proporcional.

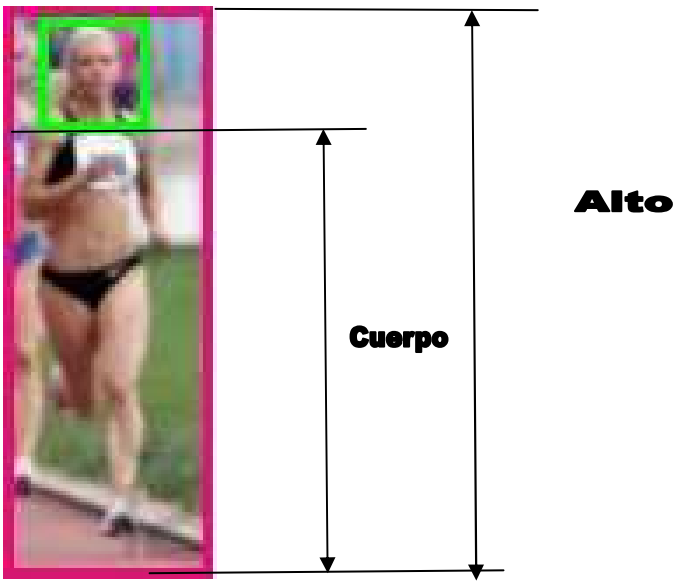
Posteriormente y debido a más resultados experimentales se concluyó mejor aumentar la altura normalizada a 540, debido a que quedaban algunas imágenes en las que no se detectaba la cara a una altura de 480 píxeles y si a 540.

Detección de persona en OpenCV, ampliada a altura normalizada de 480 pixeles: no se detecta cara

Detección de persona en OpenCV, ampliada a altura normalizada de 540 pixeles: si se detecta cara

8.3 Pruebas de determinación de los márgenes de la cara dentro de la persona.

Una vez establecido el valor del umbral de detección y el tamaño al que normalizar las detecciones de personas, la siguiente prueba consistiría en determinar una vez detectada una cara dentro de una persona, que márgenes laterales y que margen inferior ha de tener el área ocupada por la cara en la imágenes para considerarse bien situada.



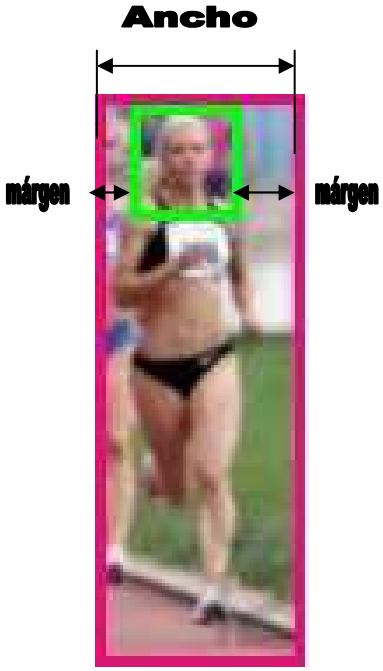
Luego para que la cara esté correctamente situada en la persona

$$\frac{Cuerpo}{Alto} > X$$

Siendo X esta cantidad a determinar;

Se realizaron pruebas con X=2/3, o lo que es lo mismo que la cara estuviera situada en el tercio superior de la cabeza y se comprobó que es el valor idóneo, según todas las pruebas realizadas.

Los siguientes márgenes a estudiar son los laterales (para que la cara quede más o menos centrada).



Ambos márgenes, (tanto el derecho como el izquierdo) han de ser al menos una parte del ancho total .

$$\frac{Margen}{Ancho} > Y$$

Siendo Y la cantidad a determinar.

En un primer momento se probó con Y= 1/3, o sea que la cara se encuentre situada en el tercio central, pero alguna imagen aislada de persona de perfil, se quedaba fuera la cara, por lo que definitivamente se redujo a 1/4.

Persona descartada como tal inicialmente por detectarse la cara (borde gris de mayor grosor) unos pocos pixeles fuera del margen hacia la derecha. Con el margen inicial a 1/3)

8.4 Problema del efecto memoria

Uno de los principales problema de la aplicación es el "efecto memoria" el cual proviene del detector de caras, ya que Encara está pensado para tratar frames consecutivos de una misma secuencia de vídeo, con lo cual, donde se encuentra una cara de forma segura en un recorte de persona (son las caras que el detector facial recuadra en verde), en el frame consecutivo se habrá de encontrar la misma cara en el mismo sitio o en una zona aproximada.

Pero el algoritmo de nuestro proceso lo que hace es enviarle al detector facial distintos recortes de supuestas personas encontradas en cada frame, con lo que no tendrían por qué ser la misma supuesta persona la que se pasa de forma consecutiva, propiciando el fenómeno ya comentado.

Ejemplo de efecto memoria: estas 2 imágenes consecutivamente pasadas al detector de Encara que fueron aceptadas como auténticas personas, en la segunda se detecto una cara con seguridad, cuando es evidente que no la hay, esto es propiciado porque en la primera imagen, pasada justo antes al detector facial, sí que había un rostro con seguridad.

8.5 Problema imprecisión en la detección personal por un mal encuadre de la persona por parte de el detector de OpenCV

Este problema consiste en obtención de falsos negativos, debido a que en algunos casos el detector de personas fue impreciso al recuadrar la persona, dejando la cabeza fuera, con lo cual, al no haber cabeza, no hay cara posible, evidentemente. Se intentó arreglar este problema disminuyendo el umbral de detección de personas de cuerpo entero, pero esto no produjo ninguna mejoría.

Ambas pruebas con distintas personas en distintos videos

Se observa como el detector de personas efectivamente detectó una persona donde la había, pero recuadrándola mal, con lo que el detector facial no pudo reconocer ninguna cara bien situada.

8.6 Pruebas para determinar número de frames permitidos sin aparecer y porcentaje de área de coincidencia

Para el tema del seguimiento de la persona hubo que establecer dos constantes:

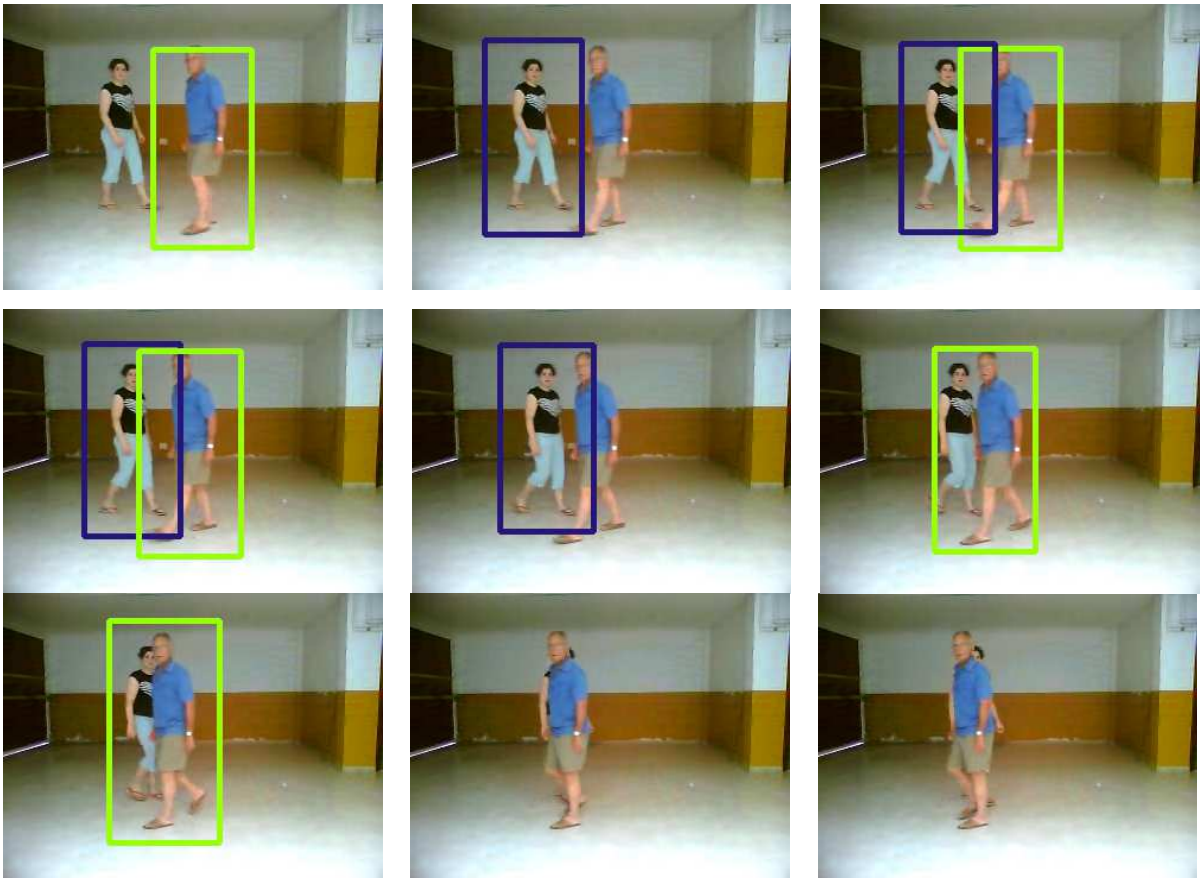
- el número de frames permitidos sin detectar una persona hasta que se considera que la persona ya no se encuentra definitivamente en el vídeo (**FRAMES_PERMITIDOS_SIN_APARECER).**
- el porcentaje de área de coincidencia entre la detección de dos personas de distintos frames para considerar que son la misma persona, **(TANTO_POR_UNO_COINCIDENCIA_AREA_PERSONA).**

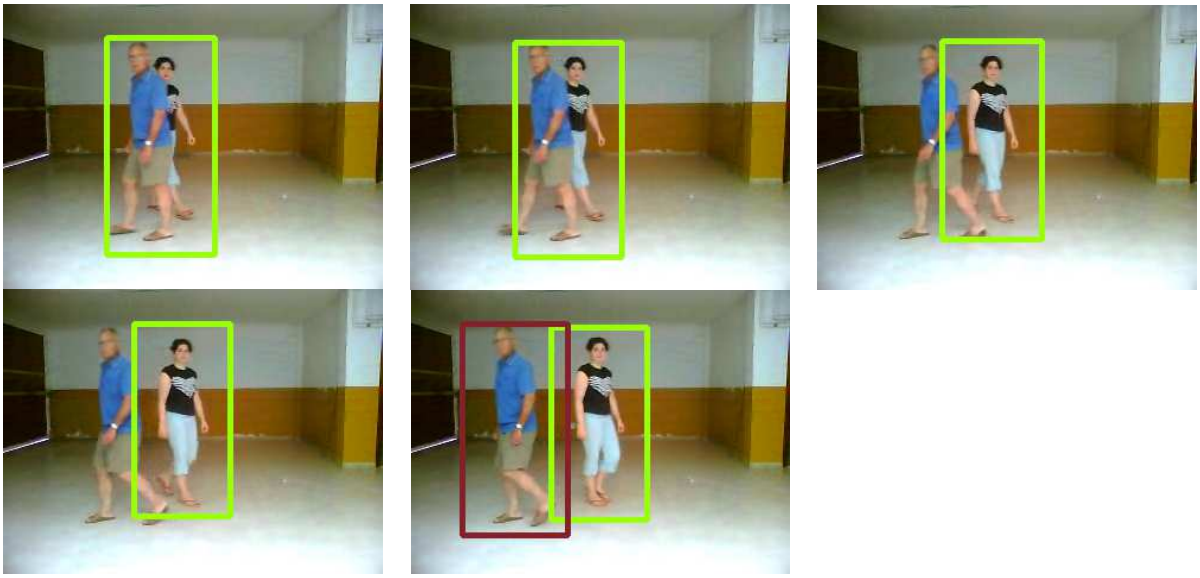
Se realizaron diversas pruebas y se concluyó establecer a un número de al menos seis frames como número de frames consecutivos permitidos para que una persona anteriormente detectada se considere que todavía se podría encontrar en la secuencia de vídeo y por tanto seguirla buscando, se contabilizaron vídeos en los que correctamente una persona volvía a ser detectada después de llevar 5 frames desaparecida.

Y el porcentaje de solapamiento entre dos detecciones de distintos frames se fijó en un 40% (0.4) para que se consideren la misma persona. Fue necesario reducir este porcentaje hasta esta cifra debido que cuando una persona se aleja, implica que el porcentaje de coincidencia sea menor, ya que ocupará un área menor al estarse alejando y por tanto disminuye notablemente el área coincidente con la imagen anterior.

8.7 Problema de cruce de personas

A diferencia de los otros problemas que son limitaciones presentes en el software de detección del que partíamos, este problema es propio de nuestra propia implementación, y ya contábamos con él. Es debido a que nuestra implementación no contempla la posibilidad del cruce de personas, ya que para comprobar que una persona es la misma persona que la del frame anterior se basa en el porcentaje del área de intersección de ambas, si hay personas que se cruzan, ya habría confusión en la propia ejecución; como en el siguiente ejemplo.





Como se puede observar en este cruce de 2 personas al principio diferenciadas, una con recuadro azul y otra con recuadro verde, se cruzan, primero se confunden con una única persona y por un momento ni si quiera hay reconocimiento personal, luego una persona sale con “la identidad” de la otra y la otra será reconocida como una “nueva persona” (recuadro marrón).

8.8 Ejemplos de secuencias finales.

Secuencia 1

En estos 9 frames consecutivos de una secuencia de vídeo se observa como se detecta y se realiza seguimiento de diferentes personas, siendo las mejor detectadas, el hombre recuadrado en verde y la mujer recuadrada en lila. También se detectó bien los primero frames la chica en color ocre.

Probablemente el recuadro de color más anaranjado del frame 2 sea un falso negativo, ya que es demasiado alto para contener una “cara real” correctamente ubicada.

Secuencia 2

En esta secuencia de frames se observa como se hace un buen seguimiento de las dos personas que caminan de frente. Fallando la detección en algunos frames, pero siendo recuperada en otros.

9. Manual de Usuario y Software

9.1 Instalación

9.1.1 Aspectos generales

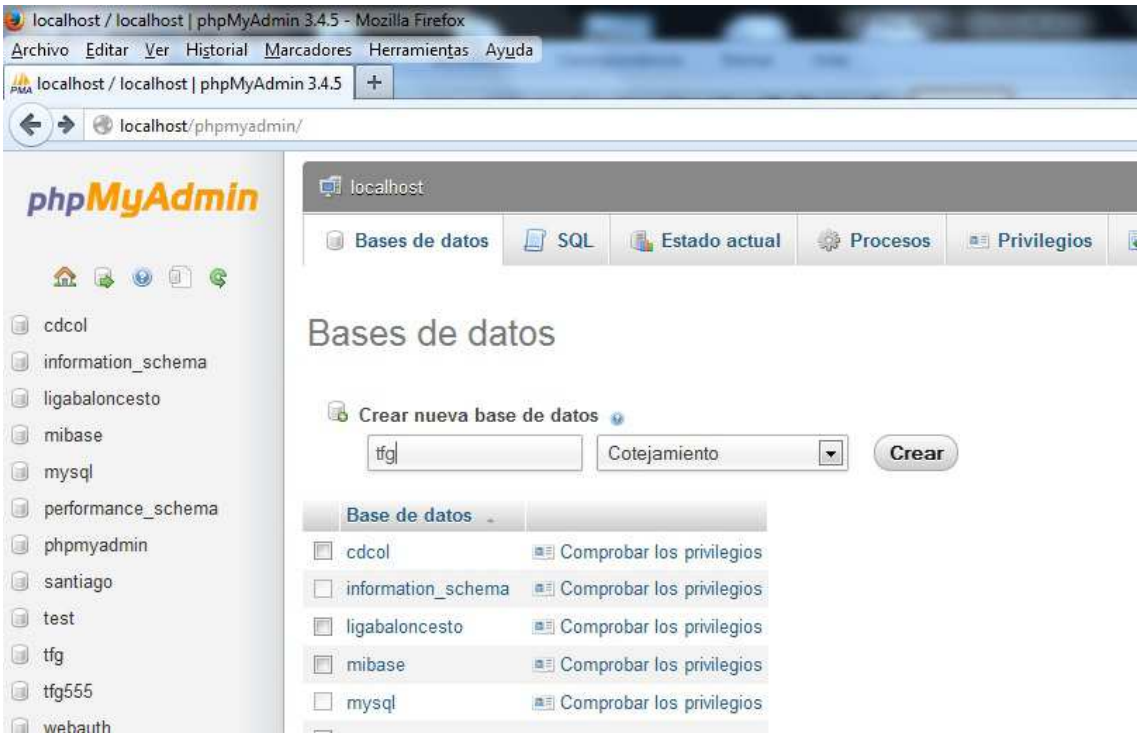
Para la ejecución del programa es necesario, el XAMPP (u otro tipo de software de base de datos que pueda ser utilizado mediante ODBC). También es necesario tener instaladas las liberías de OpenCV y Encara para la ejecución del programa.

Otro detalle a tener en cuenta es que el fichero de LOG se guardará en **`c:/archivos_ejecución`**, por lo tanto será necesario crear este directorio.

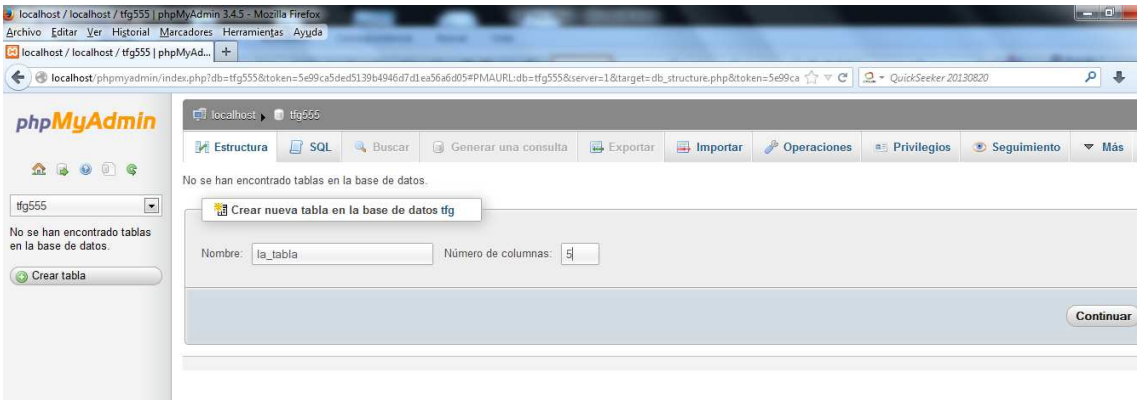
Ya que para el presente TFG se ha empleado XAMPP será este el software que se explicará la creación de la base de datos necesaria para la ejecución del programa.

9.1.2 XAMPP

Una vez instalado y puesta en funcionamiento el MySQL y el Apache (este último para el acceso mediante el navegador al localhost), Accedemos al menú de creación de bases de datos, y creamos una base de datos cuyo nombre sea "**`tfg`**"

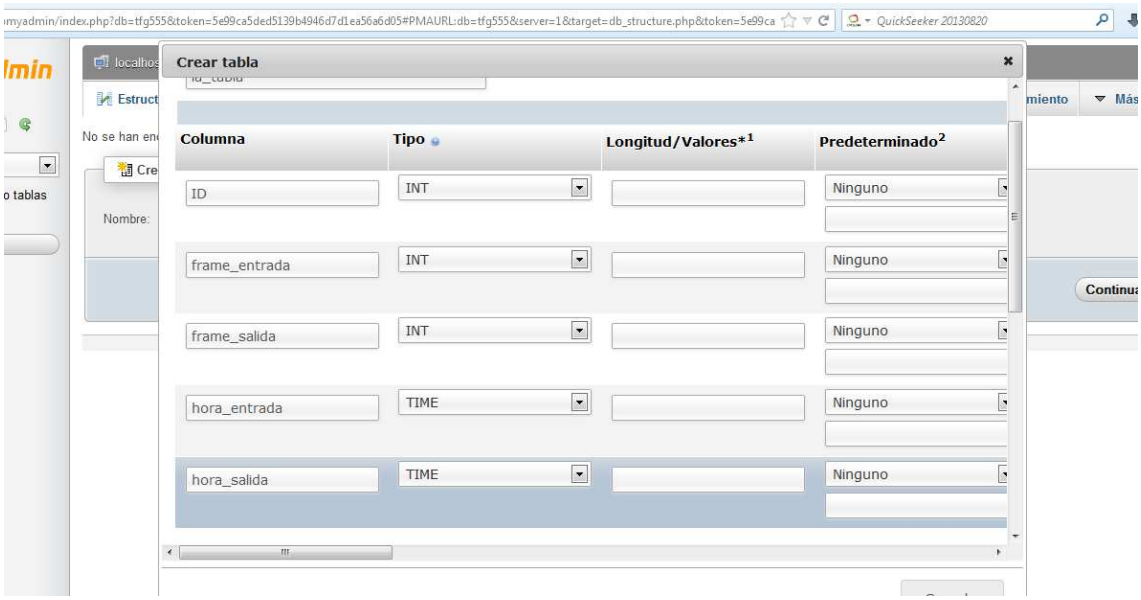


En el siguiente paso creamos una tabla para la base de datos, el nombre de la tabla ha de ser "**`la_tabla`**" y debe contener 5 campos

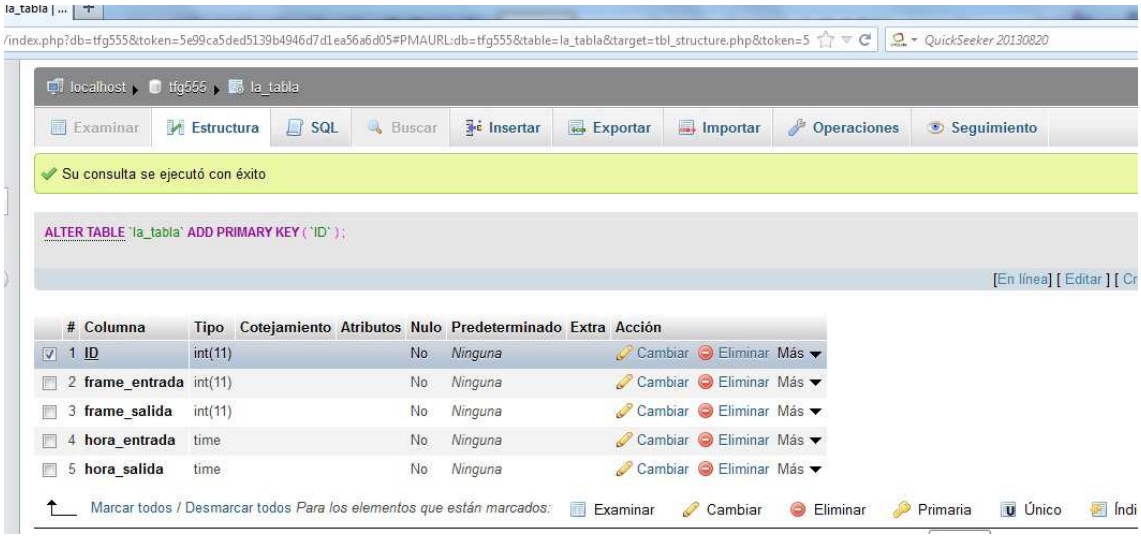


Posteriormente se ha de añadir los nombres de los campos y su tipo.
Los 5 campos son:

NOMBRE	TIPO
ID	INT
frame_entrada	INT
frame_salida	INT
hora_entrada	TIME
hora_salida	TIME



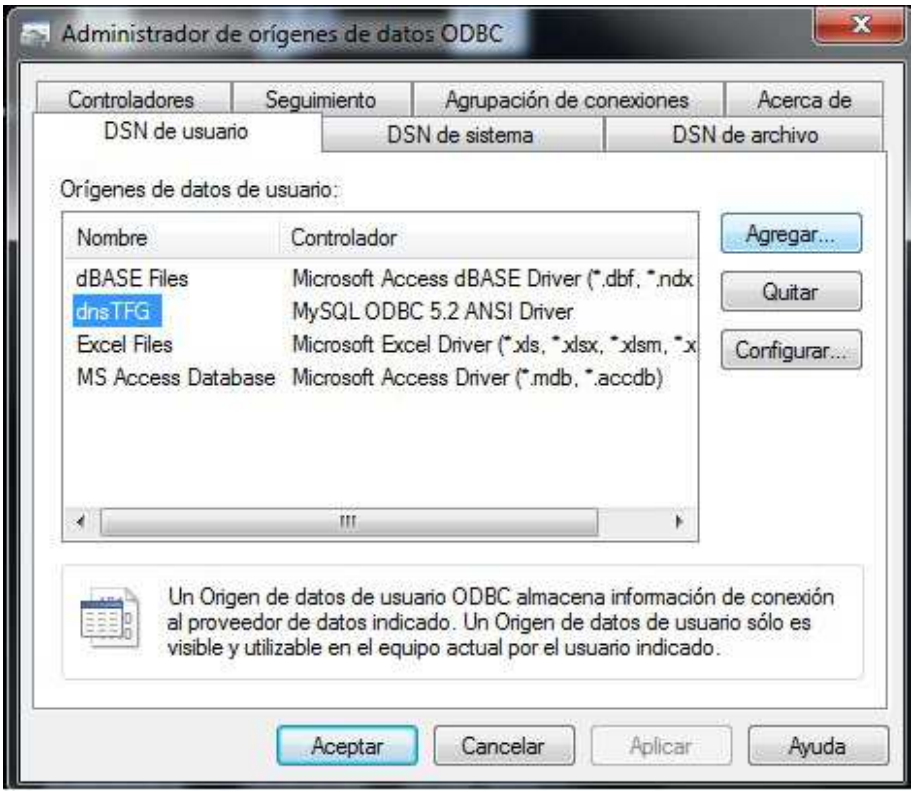
Una vez creados los cinco campos, se ha de seleccionar como primary key (clave primaria) el campo ID, si es que no se hizo anteriormente.Para ello debemos picar en el recuadro a la izquierda del campo y el botón “primaria”



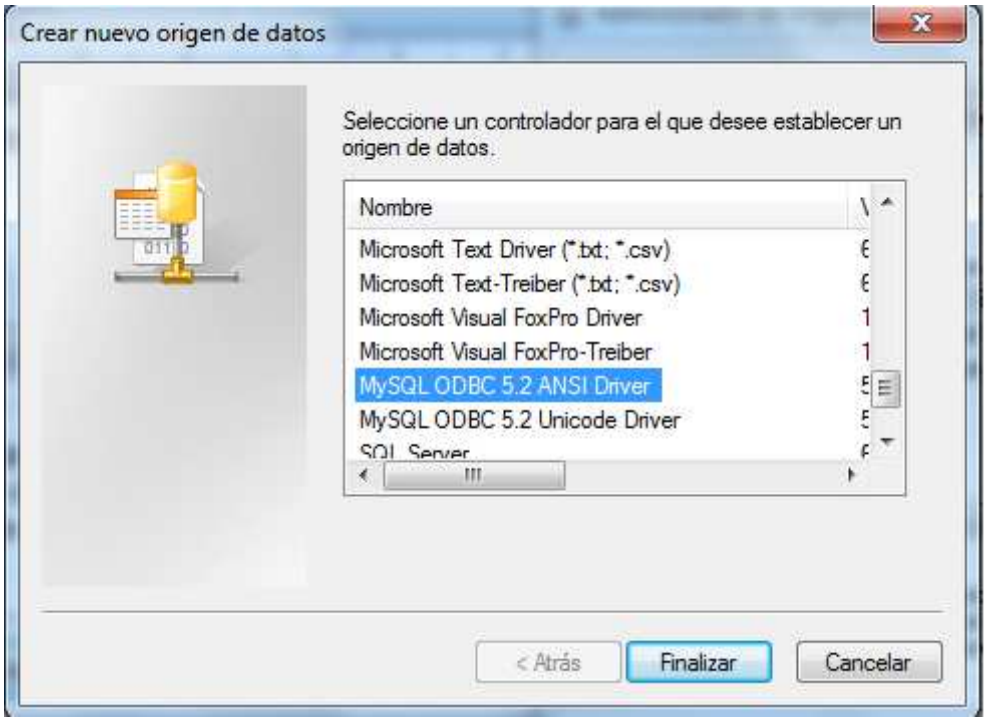
9.1.3 ODBC

Una vez hecho esto ya tenemos la base de datos, el siguiente paso sería instalar y configurar el ODBC, para que haga de intermediaria entre la aplicación final y la propia base de datos

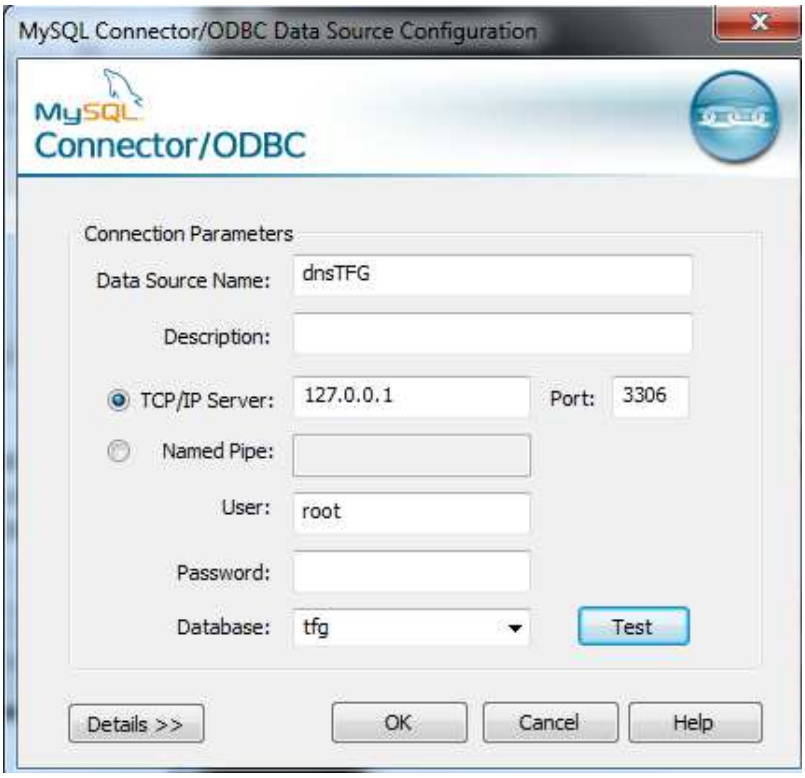
Primero hemos de instalar el ODBC, en nuestro caso se utilizó la versión 5.2.5. Luego crearemos un DSN determine una base de datos concreta. Será un DSN de usuario



Será del tipo MySQL ODBC Ansi Driver



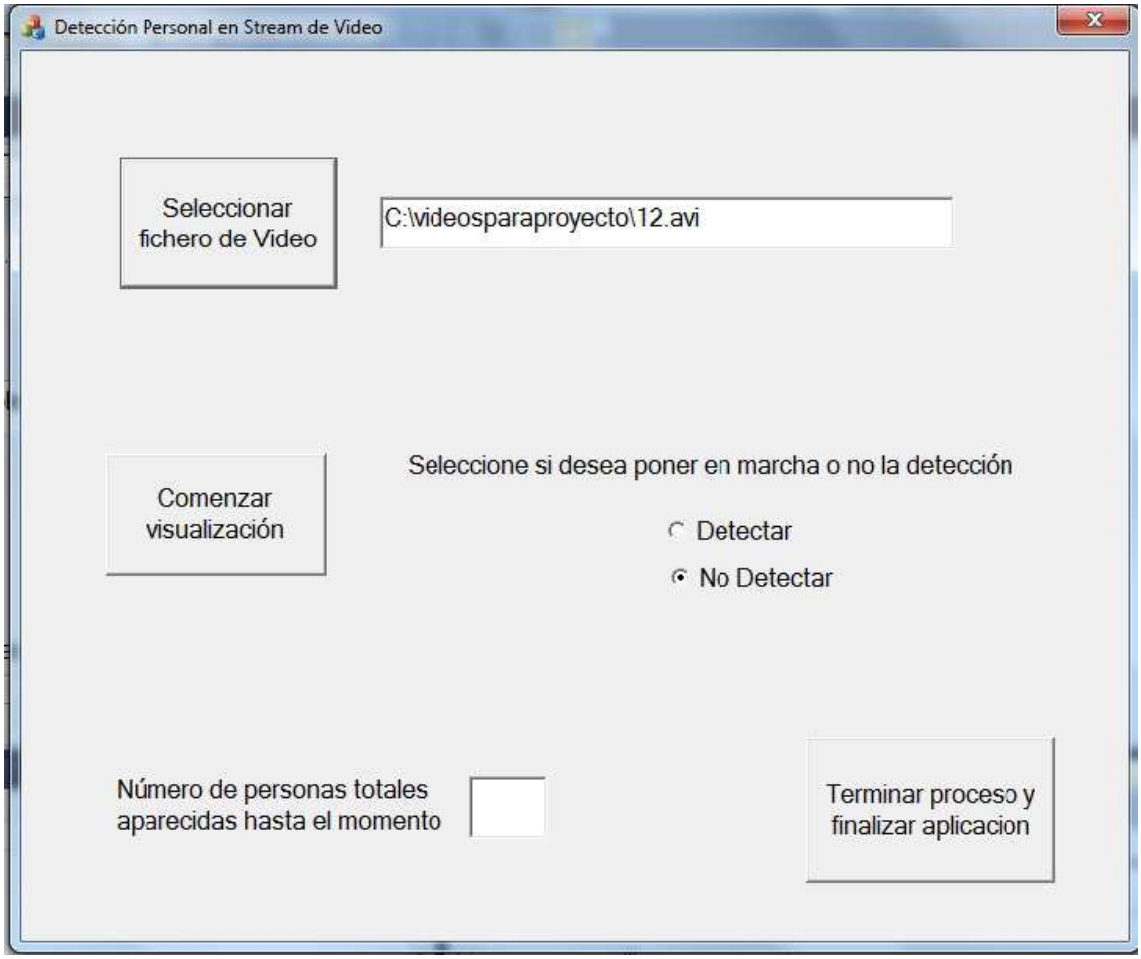
Y finalmente añadimos los parámetros, el nombre ha de ser **"dnsTFG",** dirección IP será 127.0.0.1 (dando por hecho que es en localhost donde se encuentra la base de datos), puerto 3306, usuario root (suponiendo que es éste el superusuario), contraseña si la hubiera y la base de datos **"tfg".**



Ya concluida la configuración aparecería el DSN creado en la pestaña DSN de usuario.

9.2 Uso de la aplicación final y consulta de resultados

La interfaz de la aplicación se muestra en la siguiente figura



La aplicación contiene dos radio-botones, el de "Detectar" y el de "No Detectar" que son para la activación ó no del proceso de detección.

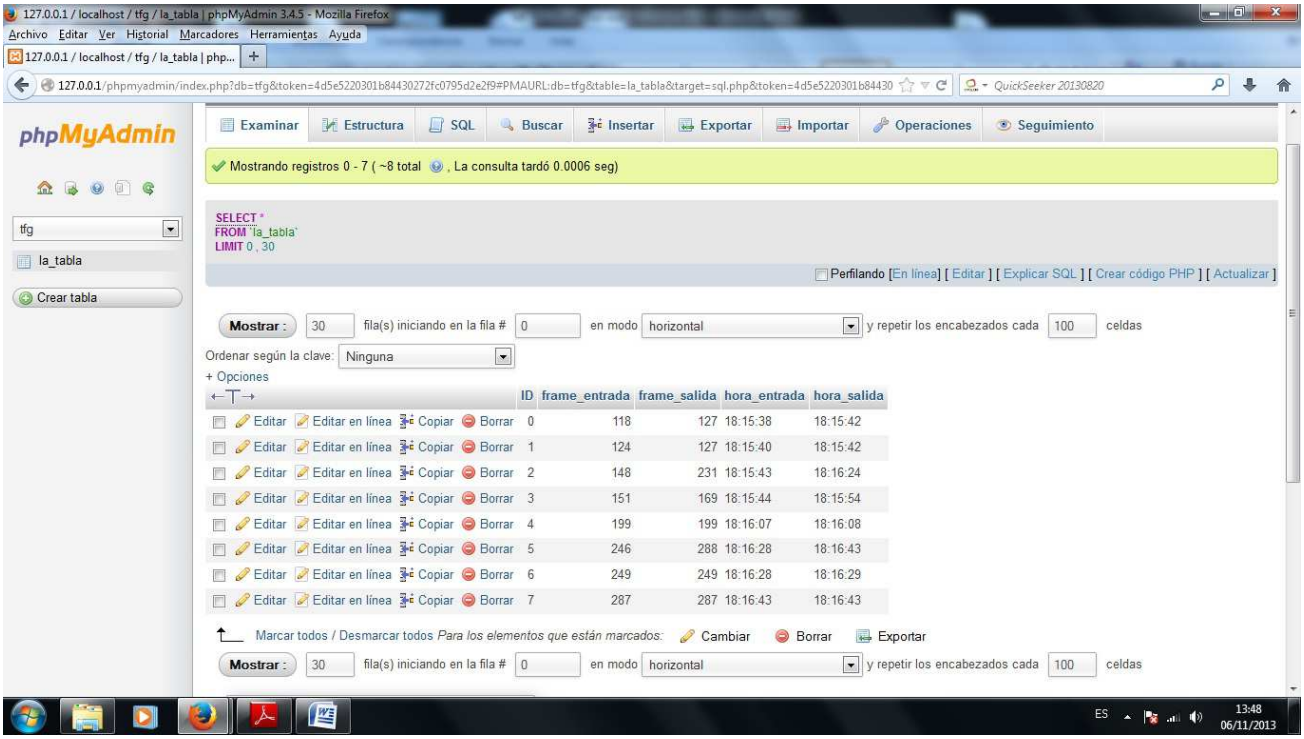
Por defecto, viene seleccionado el "No Detectar", de esta forma al apretar el botón de "comenzar visualización", únicamente se visualice el vídeo, sin aplicar la detección, hasta que se apreté el radio-botón de "detectar". Es posible picar el botón de "detectar" antes de apretar el botón de "comenzar visualización" para que el proceso de detección se esté aplicando desde el comienzo de la visualización.

Antes de apretar el botón de "comenzar visualización", se podría apretar el botón de "seleccionar vídeo" para que se abra un menú desplegable para seleccionar cualquier vídeo que esté disponible en cualquier ubicación. En caso de seleccionar un fichero no deseado, se puede volver a pulsar este botón para seleccionar el correcto. En caso de no seleccionar ningún fichero y apretar el botón de "comenzar visualización", será la webcam por defecto la que se visualice.

Durante la visualización del vídeo se puede alternar entre "No Detectar" y "Detectar", las veces que deseemos, teniendo en cuenta que cada vez que se pasa de "Detectar" a "No Detectar" se considera desparecidas, las personas actualmente presentes, pasando por tanto a ser insertadas en la base de datos y en el fichero de Log.

En la parte inferior de la ventana tenemos el recuadro que muestra siempre el número de personas aparecidas hasta el momento. Y al lado el botón de "terminar el proceso y finalizar aplicación", el cual termina el proceso, en cualquier punto en el que esté, de forma correcta, añadiendo los datos a la base de datos, cerrando la ventana y finalizando aplicación.

Para la consulta de los datos guardados accederemos a la base de datos, mediante el acceso al localhost, ahí seleccionaremos nuestra base de datos **"tfg"** y la tabla **"la_tabla"**.



La otra consulta de datos es en el archivo de LOG, donde debería encontrarse los mismos datos que en la base de datos. El archivo de LOG, se encuentra en **c:/archivos_ejecucion/log.xls.**

10. Conclusiones

La realización del TFG ha permitido tener una visión de como sería un verdadero proyecto en la vida real, y se empieza a ser consciente de las complicaciones que surgen al abordar un proyecto de una cierta envergadura. Se observa que para cada pequeña modificación añadida al proyecto se ha de comprobar el correcto funcionamiento de casi todo lo anteriormente presente. Se puede apreciar la importancia del trabajo ordenado, y de que cada cierto tiempo puede ser necesario volver a reestructurar todo lo hecho hasta el momento.

En cuanto a los resultados del trabajo se puede concluir que son más que satisfactorios, ya que se consigue un seguimiento de personas relativamente bueno. No llega a ser perfecto, por todo lo anteriormente mencionado, por lo que es un trabajo el cual se puede seguir ampliando y mejorando bastante en futuros proyectos.

Como punto negativo se destaca la necesidad de una gran capacidad de procesamiento para trabajar sobre streams de webcam en tiempo real. No siendo tan necesaria en el caso de procesar un archivo vídeo.

11. Posibles propuestas de ampliación

Una posible mejora es para confirmar que una persona es la misma, sería comparar el valor medio de los pixeles, ya que siendo la misma persona, el valor medio del pixelado debería ser aproximadamente el mismo.

Otra posible mejora sería que las constantes, no vinieran predefinidas en el código fuente, sino que hubiera cuadros de texto en la interfaz gráfica de donde se obtuvieran estos valores (márgenes laterales, umbral de detección de personas de cuerpo entero….) introducidos por el usuario. O bien disponer de un fichero de configuración donde se pudieran cambiar sin tener que introducirlos cada vez que se ejecuta la aplicación ni tener que recompilarla al estar incluidos en el código..

Una mejora a la hora de reducir falsos positivos podría ser la eliminación de personas que aparezcan durante un único frame, las cuales en la mayoría de los casos serían detecciones erróneas.

Se piensa también en la opción a calcular la efectividad de eliminar personas que solo aparecieran durante 1 frame, ya que en la mayoría de estos casos se trata de falsos positivos.

Otro aspecto a considerar es la mejora de la interfaz, en donde se podría profundizar hacia una interfaz más “visual” del presente TFG.

12. Bibliografía

- Microsoft Visual C++.net Editorial Anaya
- EncaraV2.2/doc
- http://www.buenastareas.com/ensayos/Histograma-Del-Gradiente-Orientado/543134.html
- http://opencv.willowgarage.com/documentation/
- http://opencv.willowgarage.com/documentation/cpp/
- http://opencv.jp/opencv-1.0.0_org/docs/ref/opencvref_highgui.htm
- http://berlioz.dis.ulpgc.es/roc-siani/publicaciones-principal/santana-icip-2003
- http://msdn.microsoft.com/es-es/library/k4cbh4dh%28v=vs.100%29.aspx
- http://www.boe.es/boe/dias/1999/12/14/pdfs/A43088-43099.pdf
- https://www.agpd.es/portalwebAGPD/canalresponsable/videovigilancia/common/Instruccion_1_2006_videovigilancia.pdf
- http://www.apachefriends.org/es/xampp.html.
- http://es.wikipedia.org/wiki/MySQL.
- http://stackoverflow.com/questions/4158244/opencv-draw-point-and-line-between-2-points
- http://docs.opencv.org/modules/core/doc/basic_structures.html#Rect_
- http://publib.boulder.ibm.com/infocenter/iseries/v5r4/index.jsp?topic=%2Frzaik%2Frzaikodbcapicplusplus.htm