UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

EVALUACIÓN COMPARATIVA DE ALGORITMOS DE DETECCIÓN DE PERSONAS EN SECUENCIAS DE VÍDEO

PROYECTO FIN DE CARRERA

INGENIERÍA DE TELECOMUNICACIÓN

Borja Alcedo Moreno

JULIO 2014

EVALUACIÓN COMPARATIVA DE ALGORITMOS DE DETECCIÓN DE PERSONAS EN SECUENCIAS DE VÍDEO

AUTOR: Borja Alcedo Moreno

TUTOR: Álvaro García Martín

PONENTE: José M. Martínez Sánchez

Video Processing and Understanding Lab
Dpto. de Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Julio de 2014

Trabajo parcialmente financiado por el gobierno español bajo el proyecto TEC2011-25995 (EventVideo) (2012-2014)

Resumen

El incremento de datos que se recogen a diario no cesa de multiplicarse y se calcula que, para el año 2020, la cantidad anual de datos recogida será de unas 40 veces la actual. Además, el 80% del total de los nuevos datos será información no estructurada como datos multimedia (que representan el 70% de esta información)[1] lo que permite comprender la importancia de su análisis. Este hecho hace que cada vez sea más importante disponer de herramientas que procesen esos datos para ofrecer los resultados deseados sin tener que analizar personalmente los mismos.

Por ello, son cada vez más los investigadores que desarrollan técnicas y algoritmos para realizar el tratamiento de datos multimedia y específicamente de vídeo. Una de las tareas más complejas y a la vez más útiles es la detección de personas por su aplicabilidad en el día a día. Su utilización en el ámbito de la seguridad, de la asistencia a la conducción, para la realización de resúmenes e indexación de vídeos entre otras, la han convertido en una de las principales áreas de investigación en el análisis de vídeo. Debido a ello, en la actualidad, existen gran cantidad de algoritmos de detección de personas evaluados con sus propios *dataset* y diferentes métricas para su evaluación.

Este Proyecto Fin de Carrera tiene como objetivo principal realizar una evaluación comparativa de diferentes algoritmos de detección de personas del estado del arte ante un *dataset* común. De esta forma evaluaremos uniformemente los algoritmos más representativos y sobre un entorno no específico y muy variado que pueda dar una evaluación fiable de los mismos así como optimizar las configuraciones de cada uno de los algoritmos modificando sus parámetros.

Por ello, en primer lugar se hará un estudio del estado del arte en cuanto a la detección de personas, se analizarán las fases críticas y se escogerán los algoritmos más representativos de las diferentes opciones de las fases críticas. Se estudiarán individualmente, a continuación se ejecutarán con sus parámetros iniciales fijados por el autor. Tras esto se estudiarán los parámetros de cada algoritmo y se buscarán aquellos que puedan tener una influencia en su rendimiento ante un *dataset* "neutral". Una vez seleccionados se compararán los resultados para cada algoritmo y a continuación entre ellos con sus configuraciones óptimas y de mejor media para el *dataset* en su conjunto. Para esto, se realiza una evaluación de cada algoritmo sobre el *dataset* propuesto con diferentes configuraciones. En primer lugar con la configuración por defecto del autor para comparar los resultados originales propuestos, tras esto ejecutaremos cada algoritmo con todas las configuraciones de sus parámetros escogidas, en tercer lugar la configuración de parámetros óptima de entre los elegido por cada video y en cuarto lugar la que ofrezca la mejor media para el conjunto del *dataset* en general para cada algoritmo.

Palabras clave

Detección de personas, vídeo vigilancia, HOG, ISM, DTDP, Fusion, Edge, ACF, Caltech, Inria, histograma, gradiente, segmentación, modelo de persona, regiones de interés, rendimiento, changedetection, dataset.

[1] Fuente: Global Technology Outlook 2013, IBM Research.

Abstract

The volume of daily collected data continues to increase and it is estimated that by 2020, the annual amount of data collected will grow to 40 times the existing amount. 80% of all new data will be unstructured and multimedia data, which makes up 70% of this information[1], allows us to understand the importance of information analysis more easily.

Therefore, this makes it increasingly important to have the tools that process the data more efficiently to provide the desired results, without having to personally analyse them.

There are more and more researchers developing techniques and algorithms for processing multimedia data and specifically more video based multimedia. One of the most useful and complex tasks is detecting people's applicability from day to day in a simultaneous manner. One of the main areas of research in video analysis (in the area of safety driving assistance) is abstracting and indexing video. As a result, there are many people detection algorithms being evaluated on their dataset and their different metrics.

This PFC's main objective is to target different algorithms for state of the art People Detection technology to a common dataset. As a result, accurately evaluating the most reliable algorithms in a specific environment and varying them so that they can perform a reliable assessment. In optimising the settings of each of the parameters and therefore changing each algorithms will give additional results.

Furthermore, a study will be made of the current state of the art technology regarding the detection of people and critical phases will be analyzed. The most representative algorithms of different options will be chosen in critical phase and studied individually, and run with initial parameters, set by the author. After that, the parameters of each algorithm will be examined and those that may have an influence on their performance to a "neutral" dataset, will be searched.

A selected time is given for each individual algorithm and then together with their best average result for the whole dataset and after that, the configurations are compared. For this, an assessment of each of the proposed algorithm with different configurations dataset is performed. First with the default settings of the author (to compare the original results) and after that we run each algorithm with all configurations of parameters chosen. Secondly a setting of optimal parameters between each video elected is performed. Finally, the result which gives the best average for the entire dataset in general for each algorithm.

Keywords

People detection, video surveillance, HOG, ISM, DTDP, Fusion, Edge, ACF, Caltech, Inria, histogram, gradient, segmentation, people model, region of interest, performance, changedetection, dataset.

[1] Source: Global Technology Outlook 2013, IBM Research.

Agradecimientos

En primer lugar quiero acordarme de todos los profesores que, a lo largo de este camino en la EPS, han ido depositando su semilla en cada uno de nosotros, no solo con conocimientos técnicos sino además con una forma de enfrentar el día a día. Gracias a sus aportaciones, hoy estoy a un paso de lograr esa meta que me propuse hace años, ser ingeniero de telecomunicación. De entre estos profesores, muchos me demostraron lo que es la vocación del servicio público, la pasión por la enseñanza y el amor a su trabajo, pero sin duda me llevo un recuerdo muy especial de Susana Holgado; sus ganas y alegría por vivir nos sirven de inspiración a muchos.

No puedo olvidarme de todos los compañeros y amigos que he hecho en la universidad, con más o menos trato pero siempre recogiendo y aprendiendo de cada uno de ellos, por eso quiero darles las gracias. También quiero agradecer a los compañeros del VPULab todos los momentos que hemos pasado dentro y fuera del laboratorio.

Además, quiero tener un agradecimiento especial a Jesús Bescós y a José María Martínez por la oportunidad que en su día me brindaron para poder desarrollar este Proyecto Fin de Carrera en su grupo de investigación. Una oportunidad que además apoyaron en cada tutoría y reunión con gran entrega y profesionalidad, gracias a ellos se despertó en mí un profundo interés por este campo.

Desde estas líneas, quiero rendir un merecido y sincero homenaje a la persona que ha hecho posible que este PFC sea una realidad y que además ha tenido una inmensa paciencia conmigo, mi tutor, Álvaro García Martín, sin él no estaría escribiendo estas palabras. Respondiendo a dudas en eternas madrugadas y siempre con palabras de ánimo, ha hecho que esta travesía que está llegando a su fin haya sido más llevadera.

No puedo olvidarme de aquellas personas que han estado a mi lado todo este tiempo. Empezando por mis jefes en los diferentes trabajos que he tenido y que confiaron a ciegas en mí, ofreciéndome grandes responsabilidades, pero dejándome claro también que la primera responsabilidad que tenía era conmigo mismo y con el deber de terminar mi carrera, pasando por cada uno de los compañeros que me animaban en todo momento.

Pero si hay personas de las que me acuerdo especialmente mientras escribo estas líneas son de mi familia y amigos que nunca dejaron de creer en mí, a pesar de picarme siempre con la broma del eterno proyecto. Mis padres que en su día me brindaron la oportunidad de elegir sobre mi futuro, dándome la opción de trabajar o de estudiar la carrera que libremente decidiera y apoyando la misma. Tras mi decisión, su frase más repetida hacía mí ha sido "¿cuándo presentas?". Mis hermanos que siempre se ofrecieron a echarme una mano a pesar de no entender de qué iba el proyecto, pero que gracias a sus ánimos me salvaron en muchas ocasiones de volverme loco. Y no puedo olvidarme de mi novia, Marga, que ha sufrido, especialmente en la última etapa, mis ataques de histeria por la demora en la ejecución de algoritmos, los re-cálculos de última hora o la desesperación por no lograr la mejor manera de expresar por escrito las ideas que se apelotonaban en mi cabeza.

De todos vosotros es este trabajo y mi más sincero agradecimiento.

"El mundo hay que fabricárselo uno mismo, hay que crear peldaños que te suban, que te saquen del pozo. Hay que inventar la vida porque acaba siendo verdad." - Ana María Matute (1925-2014)

INDICE DE CONTENIDOS

INDICE DE FIGURAS

INDICE DE TABLAS

INDICE DE ECUACIONES

1 Introducción

1.1 Motivación

Los sistemas de visión artificial han adquirido una gran relevancia en nuestra sociedad. La popularización del uso de cámaras y su abaratamiento han llevado consigo un auge del procesado digital de imágenes y vídeo que tendrán un gran impacto en la calidad de vida.

Esta gran implantación conlleva la necesidad de automatizar el procesado de imágenes y lograr un correcto control y gestión de las grabaciones realizadas. Es por ello que, el tratamiento automático de vídeo se ha convertido en una disciplina fundamental en nuestros días, principalmente en el entorno de la seguridad, vehículos inteligentes o robótica avanzada.

Esto ha impulsado a la detección de personas a ser una fuente continua de investigación que abarca numerosas técnicas y métodos para lograr su objetivo. Siendo una etapa previa y fundamental para posteriores partes del análisis de sistemas de visión artificial como el seguimiento o reconocimiento de acciones.

Debido a su gran importancia y repercusión en este ámbito, los investigadores han "profesionalizado" sus algoritmos y han conseguido grandes resultados sobre concretos *datasets* seleccionados por los propios autores que luego amoldaban sus algoritmos a esta base de datos de vídeo/imágenes.

Debido a que, en su mayoría por no decir totalidad, estos sistemas son *ad hoc* para cada tipo de hardware (cámara), de objetivo (detección de personas, de objetos, de acciones, etc.), de escenario,… consideramos necesario realizar un estudio, sobre una base común de pruebas [*dataset*], que nos permita una evaluación uniforme sobre los algoritmos.

De esta forma, trataremos de "poner nota" a cada uno de los algoritmos en una misma base de pruebas con un *dataset* que recoja diferentes escenarios, así como tratar de elegir la configuración óptima de cada uno de los algoritmos ajustando sus parámetros de entrada para la misma base de datos de imágenes. Al concluir el estudio trataremos de identificar el algoritmo óptimo para unas características genéricas y que no dependan de parámetros como el hardware utilizado para captar las imágenes, características de iluminación o el escenario en el que nos encontramos.

1.2 Objetivos

El objetivo de este Proyecto Fin de Carrera (PFC) es hacer un estudio del estado del arte y una evaluación exhaustiva de algoritmos de detección de personas en secuencias de vídeo con independencia de las características de la base de datos de vídeo utilizada.

En una primera etapa analizaremos el espectro de algoritmos de detección de personas, después se realizará una clasificación de los mismos y se seleccionarán aquellos que resulten más interesantes para este estudio. Tras ello los someteremos a un banco de pruebas (*dataset*) común para poder evaluarlos y compararlos objetivamente.

Existen múltiples métricas de evaluación para este tipo de análisis [1], [2] y nosotros definiremos una para poder realizar la comparación entre los resultados de los algoritmos. Se implementará esta solución de medida para obtener los resultados de la comparación.

Se ha "desestimado" medir el tiempo de procesado y recursos consumidos por los algoritmos puesto que la finalidad no es un estudio de algoritmos enfocados a tiempo real sino algoritmos más veraces en la detección general para *datasets* genéricos y no enfocados a uno concreto.

A modo de resumen se puede fijar el alcance y objetivos de este Proyecto Fin de Carrera como sigue:

i. Estudio del Estado del Arte.
ii. Clasificación de los algoritmos de detección de personas.
iii. Selección e implementación de los algoritmos más relevantes de detección de personas.
iv. Selección de una base de datos de vídeo (*dataset*) adecuado para la evaluación de los algoritmos.
v. Etiquetado manual de los vídeos para generar el *Ground Truth (GT)*.
vi. Procesado de los algoritmos seleccionados con los diferentes parámetros escogidos sobre el *dataset*.
vii. Desarrollo e implementación del sistema de evaluación elegido.
viii. Análisis de los resultados y conclusiones.
ix. Elaboración de la memoria del PFC.

1.3 Medios

Los medios necesarios para la realización de este Proyecto Fin de Carrera han sido facilitados por el grupo de investigación Video Processing and Understading Lab (VPULab) del Departamento de Tecnología Electrónica y de las Comunicaciones de la Escuela Politécnica Superior de la Universidad Autónoma de Madrid. Los principales elementos utilizados para la realización de este Proyecto han sido:

i. Parque de PC's (Windows/Linux) interconectados a través de la red de área local y con acceso a Internet y a los servidores del VPULab.
ii. Herramientas para el desarrollo de proyectos software (Matlab, C, etc.).
iii. Base de datos de vídeo.
iv. Biblioteca de libros y revistas de Tratamiento Digital de Señal e Imagen.

1.4 Organización de la memoria

La memoria de este Proyecto Fin de Carrera está organizada de la siguiente manera:

- **Capítulo I: Introducción.**
  - Motivación, objetivos, medios y organización de la memoria.
- **Capítulo II: Estado del arte.**
  - Introducción, estado del arte, arquitecturas de los sistemas de detección de personas, clasificación de los algoritmos de detección de personas y conclusiones.
- **Capítulo III: Desarrollo.**
  - Introducción, algoritmos utilizados (HOG, ISM, DTDP, Fusion, Edge, ACF) y conclusiones.
- **Capítulo IV: Experimentación y conclusiones.**

- Dataset, etiquetado de vídeos y generación del Ground Truth, concepto de persona, métrica de evaluación y resultados.

- **Capítulo V: Conclusiones y trabajo futuro.**
- **Capítulo VI: Anexos.**
  - A: Configuración y ejecución de los algoritmos.
  - B: Ejemplos de fotogramas del *dataset*.
- **Capítulo VII: Bibliografía y glosario.**

2 Estado del arte

2.1 Introducción

La detección automática de personas en secuencias de vídeo [3], [4], [5], [6] es una de las tareas más complicadas en el campo del tratamiento de imagen. Cuando intentamos detectar personas nos encontramos con infinidad de diferencias de lo que podemos considerar personas. La apariencia física, las partes articuladas y cambiantes del cuerpo, posturas, movimientos, puntos de vista, resolución de las imágenes, tamaño de las personas, grupos de personas, interacción entre personas y con objetos, etc., hacen que tengamos un diccionario casi infinito de lo que se puede "considerar persona". Si a esta complejidad le sumamos los escenarios reales con cambios de fondo que las rodea, nos encontramos ante un reto, si cabe, aún más complicado.

En este capítulo, basándonos en los documentos anteriormente citados, haremos una revisión general del estado del arte de detección automática de personas en secuencias de vídeo. Comenzaremos con un estudio del estado del arte previo a este proyecto (sección 2.2), a continuación, describiremos la arquitectura básica de todo detector de personas (sección 2.3). La sección 2.4 describirá la clasificación propuesta para los algoritmos de detección automática de personas. Para concluir el capítulo 2, presentaremos unas conclusiones globales del estado del arte (sección 2.5).

2.2 Estado del arte

Actualmente existen muchos estudios sobre detección automática de personas, algunos de ellos cubren solo parcialmente el estado del arte y otros están completamente centrados en aplicaciones específicas de vigilancia. El autor [3] presenta un estudio de detección de personas y la integración de los detectores en sistemas completos, la base de datos de vídeo utilizada ha sido capturada con una cámara situada en un vehículo en movimiento en área urbana. En su estudio nos presenta la tarea de detección de personas descompuesta en tres procesos: selección de las regiones de interés (ROI, Region of Interest), clasificación y seguimiento. Por su parte [4], también presenta un estudio sobre detección de personas. En este caso, el estudio está centrado en los sistemas de protección de peatones para la conducción de automóviles. Definen un proceso lineal con las siguientes fases: pre-proceso, segmentación de fondo, clasificación de objetos, verificación, seguimiento y aplicación. [5] presenta una visión de los algoritmos de detección centrado en ventanas deslizantes, el *dataset* utilizado en este caso también ha sido capturado desde la cámara de un vehículo en movimiento en un área urbana. En el caso de [6], nos presenta un estudio amplio y general del espectro de la detección de personas en el año 2013, sin centrarse en aplicación y haciendo una clasificación de las técnicas. Primero descompone la detección de personas en subtareas, identifica las labores críticas y clasifican el estado del arte de acuerdo con ellas.

Nuestro objetivo es realizar un estudio sobre las diferentes técnicas independientemente de la aplicación para la que hayan sido diseñadas. Tampoco tendremos en cuenta el coste computacional de los algoritmos por no ser la finalidad de este estudio. Propondremos una clasificación de las principales técnicas de detección automática de personas, seleccionaremos los algoritmos más representativos y los someteremos a una evaluación exhaustiva modificando sus parámetros para que nos permita señalar fortalezas y debilidades de las diferentes configuraciones de los algoritmos.

2.3 Arquitecturas de los sistemas de detección de personas

Definimos la tarea de detección de personas en una arquitectura de pasos en cascada. En una primera etapa capturaremos el vídeo, en nuestro caso utilizamos un *dataset* libre con diferentes medios de captura. Descomponemos los vídeos en sus *frames*, nuestro análisis desecha la información de movimiento y no realizamos tracking tratando cada imagen como independiente del resto. El siguiente paso es detectar las regiones de interés (ROI) dentro de la imagen. A continuación verificamos que la ROI coincide con las características que podemos considerar como personas para en un paso final decidir, en base a una puntuación umbral, si finalmente calificamos ese *blob* como persona. Para concluir tendremos el análisis de los resultados y la verificación de lo fidedigno que es con "*nuestra realidad*". La Figura 2.1 muestra la arquitectura básica del sistema de detección de personas.

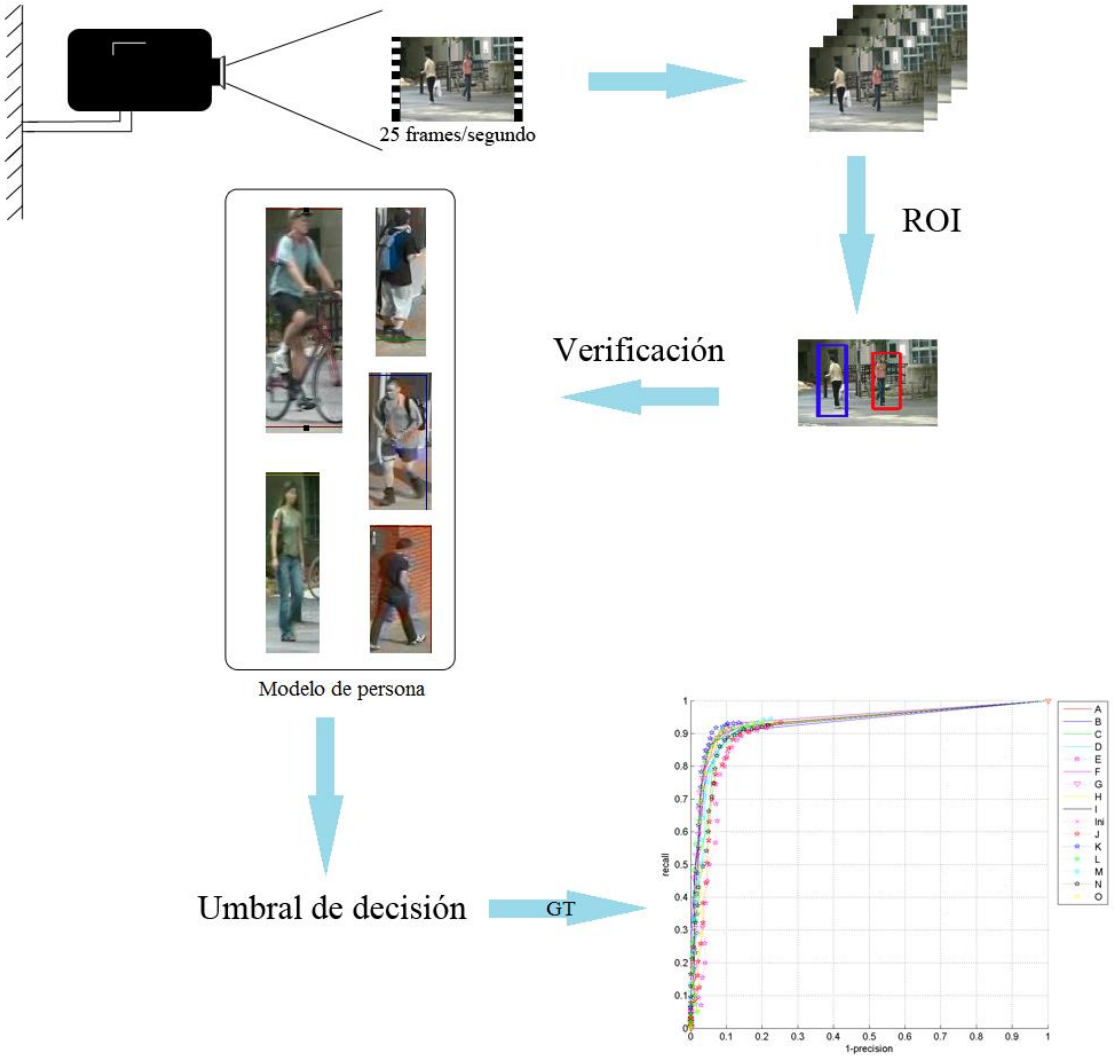


Figura 2.1: Arquitectura de un sistema de detección de personas.

Entrada

En los algoritmos de detección de personas se pueden utilizar múltiples formatos de entrada: secuencias de vídeo, imágenes en color, imágenes en escala de grises, 2D, 3D, imágenes infrarrojas, listas de imágenes. En este proyecto la entrada inicial son secuencias de vídeo a color con diferentes tamaños y resoluciones que se descomponen en sus *frames* (25 imágenes/segundo) antes de pasar a ser tratados por cada uno de los algoritmos que tienen como secuencia de entrada un fichero con las ubicaciones de cada uno de los *frames* del vídeo.

Detección de las Regiones de Interés (ROI)

Este paso consiste en extraer de la imagen las regiones candidatas a ser persona. Ésta es la tarea más crítica del proceso de detección de personas. Técnicas como la extracción de fondo, la ventana deslizante, la detección de bordes, etc. serán definitivas para el funcionamiento del algoritmo en cuanto a su fiabilidad ante cambios de fondo o en cuanto a los recursos requeridos por el algoritmo para poder procesar el vídeo.

Modelo de persona y clasificación

Este paso, también crítico, es el segundo que utilizaremos para hacer la clasificación de los algoritmos de detección de personas que consideramos en este Proyecto Fin de Carrera. En este paso verificamos que las regiones detectadas como posibles personas cumplen con los modelos definidos y por lo tanto son "aspirantes reales". Una vez que la ROI es considerada como candidata a ser persona y con la puntuación que el algoritmo le concede decidiremos si esta región es persona definitivamente si su confianza supera el umbral establecido.

2.4 Clasificación de los algoritmos de detección de personas

La clasificación propuesta para los algoritmos de detección de personas no tendrá en cuenta el movimiento, el seguimiento (*tracking*), la resolución o características de la imagen, ni los recursos consumidos o el tiempo de proceso de los algoritmos. Como ya se ha indicado, las etapas críticas para esta clasificación son las fases de detección de regiones de interés y el modelo de persona basado únicamente en información de apariencia. La Figura 2.2 muestra los criterios utilizados para la clasificación propuesta.

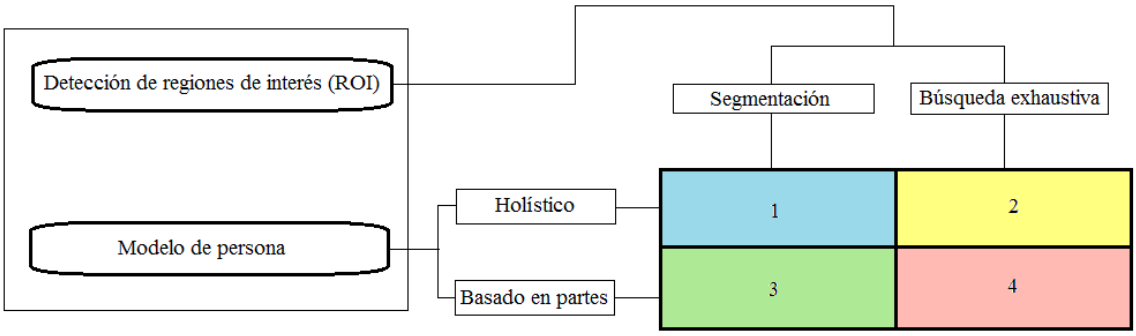


Figura 2.2: Clasificación de algoritmos de detección de personas por las fases críticas de detección de región de interés y modelo de persona.

2.4.1 Detección de regiones de interés

Actualmente existen multitud de aproximaciones a la detección o extracción de regiones de interés en las imágenes, entre ellas podemos identificar dos ramas principales, la primera rama es la segmentación (frente/fondo) y la segunda es la búsqueda exhaustiva.

El objetivo de esta primera etapa es fijar las zonas/regiones de interés (ROI) dentro de la imagen y obtener así su posición y medidas. En la Tabla 2.1 se recogen algunos enfoques del estado del arte clasificados por la técnica de detección de región de interés que utilizan.

Segmentación [7, 8, 9, 10, 11, 12, 27]

La segmentación se basa en analizar las características de cada pixel, y asignarle una etiqueta dependiendo de sus características como el color, intensidad, textura, etc. De esta manera, los pixeles que comparten características similares tendrán la misma etiqueta y aquellos que sean diferentes en su entorno serán diferenciados con otra etiqueta diferente.

Con esta información podremos distinguir las diferentes regiones de nuestra imagen y localizar y discriminar las regiones de interés.

Existen diversas técnicas de segmentación utilizadas como un primer paso para la detección de personas. La aproximación más típica en escenarios de video seguridad es el uso de la información de movimiento para identificar las regiones de interés; la técnica *background subtraction* se utiliza mucho en aplicaciones de vigilancia y se basa en "restar" la imagen actual con una imagen de referencia (modelo de fondo) y así obtener los objetos en movimiento de la imagen que podemos considerar "frente". Otra técnica utilizada es la segmentación del color que se basa en que el color de la piel facilita la segmentación y el proceso de detección. También se encuentran algoritmos que utilizan algún tipo de información 3D para realizar la segmentación bien con *stereo-vision* o directamente con la utilización de cámaras 3D.

Búsqueda exhaustiva [13, 14, 15, 16, 17, 18, 19]

La búsqueda exhaustiva consiste en recorrer la imagen con una ventana de varios tamaños y analizar la similitud de esta porción de imagen con el modelo de persona escogido. Como resultado obtenemos un gran número de regiones candidatas (diferentes escalas, rotaciones y localizaciones). Esta metodología requiere, como norma general, un gran coste computacional por lo que no son útiles/utilizados para algoritmos de funcionamiento en tiempo real.

A pesar de que, como norma general esta técnica consume más recursos, es la más extendida en los algoritmos de detección de personas en el estado del arte y podemos discernir dos métodos. En primer lugar, hay algoritmos que utilizan detectores basados en ventanas deslizantes, que consisten en muestrear, en una rejilla discreta 3D (ubicación y escala), y evaluar diferentes ventanas de detección con un clasificador. Por otro lado, tenemos los detectores basados en características, cuyo enfoque de abajo hacia arriba se basa en votos probabilísticos obtenidos a partir del emparejamiento local por características.

		Técnica	Enfoque
Detección de región de interés	Segmentación	Extracción de fondo	[7] [8] [9] [27]
		Información de color	[10]
		Información 3D	[11] [12]
	Búsqueda exhaustiva	Ventana deslizante	[13] [14] [15]
		Ventana deslizante o basado en características	[16]
		Basado en características	[17] [18] [19]

Tabla 2.1: Estado del arte para detección de personas, fase de detección de regiones de interés.

2.4.2 Modelo de persona

En esta segunda fase crítica para la detección de personas, nos encontramos con el problema de, una vez que hemos "separado" las regiones de interés que son candidatas a personas, discernir si coinciden con los modelos de personas previamente definidas o entrenadas y poder así decidir si nos encontramos ante una persona o cualquier otro objeto. En nuestro caso, como hemos decidido no considerar movimiento o *tracking*, el modelo de persona se basará en únicamente en apariencia. En la Tabla 2.2 se recogen algunos enfoques del estado del arte clasificados por la técnica de modelo de persona que utilizan.

Basado en apariencia [7, 9, 13, 14, 15, 16, 17, 19, 20, 21, 22, 23, 24, 25, 27]

El modelado de personas tiene dos vertientes, aquellas que definen a la persona como un todo (como una única región, holísticas) y las que la definen como partes que forman ese todo (modelo basado en partes). El primer caso, generalmente, es menos complejo pero tiene el problema de no tener en la imagen a la persona "completa" puesto que es sensible a oclusiones y variación de posturas. El segundo caso, es más complejo pero permite que el algoritmo funcione correctamente con la existencia de oclusiones o variaciones de postura.

En ambos casos nos encontramos con diferentes técnicas, una de ellas consiste en estudiar la silueta de la persona y compararla con ciertos estándares o modelos entrenados para decidir si nos encontramos con un "objeto válido", otros utilizan la información de distribución del color del objeto (basado en el color de la piel), pero el más común es el que define la persona de acuerdo a su información de borde característica utilizando algún tipo de descriptor local como las características *Haar-like*, HOG (Histogram of Oriented Gradients) o ISM (Implicit Shape Model).

		Técnica	Enfoque
Modelo de persona	Holísticos	Silueta	[7] [9] [27]
		Características Haar-like	[20] [21] [22]
		HOG	[13] [23] [15]
		ISM	[17] [24]
	Basados en partes	HOG	[14] [16]
		ISM	[25] [19]

Tabla 2.2: Estado del arte para detección de personas, fase de modelo de persona.

En la actualidad existen infinidad de algoritmos que podemos situar en las diferentes regiones (Véase Figura 2.2 con las regiones del 1 al 4) en que hemos dividido el estado del arte dependiendo de sus dos fases críticas de detección de región de interés modelo de persona. De todos ellos, en este Proyecto Fin de Carrera hemos seleccionado 7 como representación de las diferentes regiones. Esta selección se ha basado en diferentes criterios como disponibilidad del código, actualidad o haber sido desarrollados por el VPULab. La Tabla 2.3 muestra los algoritmos elegidos así como la región correspondiente a cómo afronta las fases críticas.

		Modelo de persona	
		Holístico	Basado en partes
Detección de región de interés	Segmentación	Fusion[1] [7]	Edge[1] [27]
	Búsqueda exhaustiva	HOG [13] ACF [28] ISM [26]	TUD[2] [25] DTDP [14]

Tabla 2.3: Algoritmos analizados en este PFC.

[1] Algoritmo desarrollado por el VPULab.

[2] El TUD fue descartado debido a su lentitud, malos resultados y los problemas para ejecutar el código por incompatibilidades con el sistema.

2.5 Conclusiones

Actualmente, el espectro de algoritmos de detección de personas, es muy amplio y se encuentra en continua expansión. Como se ha observado a lo largo de este capítulo, se identifican claramente dos fases críticas que nos ofrecen una clara clasificación como la que se puede ver en la Figura 2.2 para cualquier algoritmo de detección de personas. De esta clasificación saldrán las principales características de los algoritmos en lo que se refiere a fortaleza, flexibilidad de detección con respecto a las variaciones en el tipo de vídeo o consumo de recursos.

La primera etapa es la de detección de regiones de interés en la imagen y la segunda es la del modelado de persona.

Así, en la fase de detección de ROI, los algoritmos que utilizan la búsqueda exhaustiva tienen mayor coste computacional pero también son menos sensibles a las varianzas de fondo, por su lado los que se basan en la segmentación son excesivamente dependientes de este proceso y toda su validez depende directamente de cuán bueno sea esta fase por lo que depende también directamente de la complejidad y variabilidad del fondo.

Por su parte, para el modelo de persona observamos como aquellos basados en partes son mejores a la hora de enfrentarse a variaciones en las personas, son menos sensibles a cambios de ropa, de postura, uso de mochilas o maletines por ejemplo. En cambio, cuando nos centramos en un modelo holístico de persona, la variación de esas características afectan en mucho a la clasificación. En este punto, los holísticos requieren entrenamientos más sencillos y son más estructurados y los basados en partes tienen un entrenamiento más complejo que les ofrece una mayor flexibilidad en la labor de detectar la persona.

Estas ventajas e inconvenientes que observamos en los diferentes modelos son consideradas por los autores para decidir qué opción escoger en función de la finalidad del algoritmo que estén desarrollando. Si queremos por ejemplo algoritmos en tiempo real, la utilización de la búsqueda exhaustiva junto con un modelo basado en partes, como puede ser el caso de DTDP [14], nos obligará a tener una infraestructura con una arquitectura *hardware* con gran poder de procesado y esto no nos asegura unos resultados mejores.

3 Desarrollo

3.1 Introducción

Como ya indicamos anteriormente, en la actualidad existen infinidad de algoritmos de detección de personas. Como norma general, cuando se fijan los objetivos de un algoritmo de detección de personas, se diseña el sistema para unas condiciones prefijadas. Condiciones predefinidas durante la captura de imágenes (resolución, foto/vídeo, color/escala de grises, fijo/movimiento, interior/exterior, fondo fijo/dinámico, iluminación, etc.) o el objetivo de la detección (toda la imagen, áreas de interés dentro de la imagen, agrupaciones de personas/personas individuales, personas con ciertas características, etc.). Al optimizar los algoritmos para estas condiciones u objetivos, los ajustes se producen para lograr el máximo rendimiento con las especificaciones planteadas en la fase de inicio, desechando cualquier otro escenario posible. Es por esto que, esta solución, se convierte en una desventaja a la hora de evaluar los algoritmos en condiciones diferentes a las entrenadas. Esto hace que los resultados obtenidos no sean generalizables, cuando salimos de esas características los resultados varían.

En objetivo de este PFC consiste en evaluar estos algoritmos eliminando ese enfoque *ad hoc* para lograr una evaluación fidedigna para un *dataset* que recoge múltiples escenarios, situaciones, condiciones de iluminación, resolución, etc.

Para esta evaluación hemos escogido, de la gran diversidad existente, siete algoritmos de detección de personas [7, 13, 14, 26, 27, 28], que cubren todas las opciones de clasificación descritas en el apartado 2 de la presente memoria.

En las siguientes secciones se introducirán los diferentes algoritmos objeto de estudio de este PFC basado en la descripción de los autores.

Los criterios utilizados para la elección de los algoritmos han sido los siguientes en orden de prioridad:

- Evaluar al menos un algoritmo característico de cada una de las zonas de la clasificación del estado del arte, para tratar de representar todo el espectro reflejado en la Figura 2.2.
- Código disponible para su utilización de forma libre.
- Que fueran desarrollados por el VPULab.
- Fecha de publicación lo más actual posible.
- Velocidad de ejecución.

En el anexo A pueden verse los pasos dados para descargar los diferentes algoritmos, configurarlos y ejecutarlos así como una muestra de los ficheros de salida.

3.2 Histogram of Oriented Gradients (HOG)

3.2.1 Introducción

En esta primera parte introduciremos las líneas generales en las que se basa el primer algoritmo, HOG [13]. Dentro de la clasificación propuesta en este trabajo, nos encontramos ante un algoritmo que realiza la labor de detección de regiones de interés de forma exhaustiva, a partir de histogramas de gradientes orientados calculados en bloques dentro de la imagen, y utiliza un modelo de persona holístico.

El gradiente en una imagen es un vector que mide la magnitud y dirección de la variación (intensidad o color) de un pixel con los píxeles de su entorno. El módulo de este vector será su magnitud y la dirección señalará la máxima variación del mismo con

respecto a los de su entorno, es por ello que este cálculo ser realiza en una "malla" de dimensión impar.

El autor parte de la premisa de que la apariencia y la forma del objeto se caracterizan por la distribución de los gradientes como por las direcciones de los mismos en los bordes de dicho objeto. A partir de aquí, a través de un modelo entrenado de persona con una SVM (Support Vector Machine), el autor clasifica si la ROI se considera persona o no atendiendo a un todo y no a partes del mismo. Para su implementación, se divide cada ventana en pequeñas regiones, denominadas celdas, en cada una de ellas se acumula un histograma de las direcciones del gradiente o de las orientaciones de borde sobre los píxeles que se encuentran en cada celda.

En [13], describe una serie de ventajas que el uso de HOG tiene para la tarea de detección de personas. Calcular los valores del gradiente, especialmente en los bordes de los "objetos", es una propiedad muy característica de las ROI. Además, esos valores, no se ven alterados fácilmente ante transformaciones geométricas o fotométricas locales en la imagen. Translaciones o rotaciones afectan menos al resultado siempre que sean mucho más pequeñas que el tamaño local de espacio u orientación de los bloques en los que dividimos la imagen. La mejor estrategia para optimizar la detección de personas es utilizar un muestreo de orientación fina y una fuerte normalización de la intensidad, esto es debido a que permite que las extremidades y las partes del cuerpo varíen la apariencia de la persona siempre que mantengan aproximadamente una orientación vertical. A continuación, en la Figura 3.1, puede verse un ejemplo de funcionamiento de los HOG.

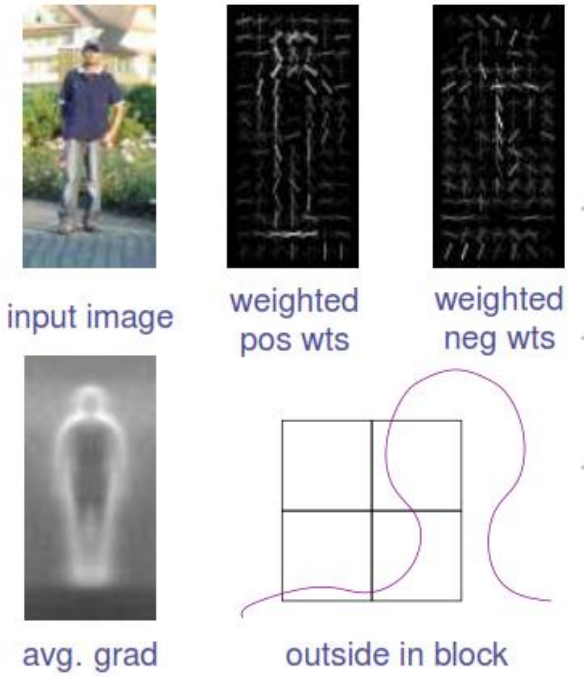


Figura 3.1: Funcionamiento del cálculo del gradiente en una imagen, figura extraída de [13].

3.2.2 Funcionamiento

Se divide la imagen en cierto número de sub-imágenes del mismo tamaño que denominaremos celdas. Estas celdas se agrupan en bloques con un mismo número de celdas de ancho y alto. Además, como puede verse en la Figura 3.2 que recoge el funcionamiento del algoritmo, entre bloques adyacentes existen celdas en común que servirán para un mejor funcionamiento del algoritmo por ser menos sensible a los efectos de borde entre bloques.

A partir de aquí el algoritmo calcula el HOG de cada celda y a continuación agrupa los HOG's de cada bloque.

Una vez que hemos aplicado la técnica de HOG para la detección de las regiones de interés utilizaremos una SVM que, a partir de patrones (positivos y negativos) de modelos de persona aprendidos, decidirá si el objeto detectado es persona o no. El modelo entrenado parte de una selección de 1.239 patrones positivos y 1.218 imágenes sin personas para el caso de los patrones negativos.

Para la propuesta de [13] el autor divide el proceso en 6 etapas: la primera es la normalización Gamma/Color, en una segunda etapa se realizará el cálculo de los gradientes, las distintas formas de calcular los gradientes afectan directamente al rendimiento de este detector, el autor propone el uso de suavizado. A continuación, se calculará el voto ponderado de las contribuciones espaciales y de orientación de las celdas. El cuarto paso es la normalización de los diferentes gradientes en las zonas solapadas de los bloques. Para ello, se acumula la energía resultante de los histogramas locales sobre cualquier región de bloques y se usan los resultados para normalizar todas las celdas del bloque. Tras esto, recogeremos los valores de los HOG's que se encuentren dentro de la ventana de detección (64 x 128 píxeles). El paso final consiste en evaluar con una SVM lineal estas regiones candidatas a personas para decidir si la detección es positiva o negativa.

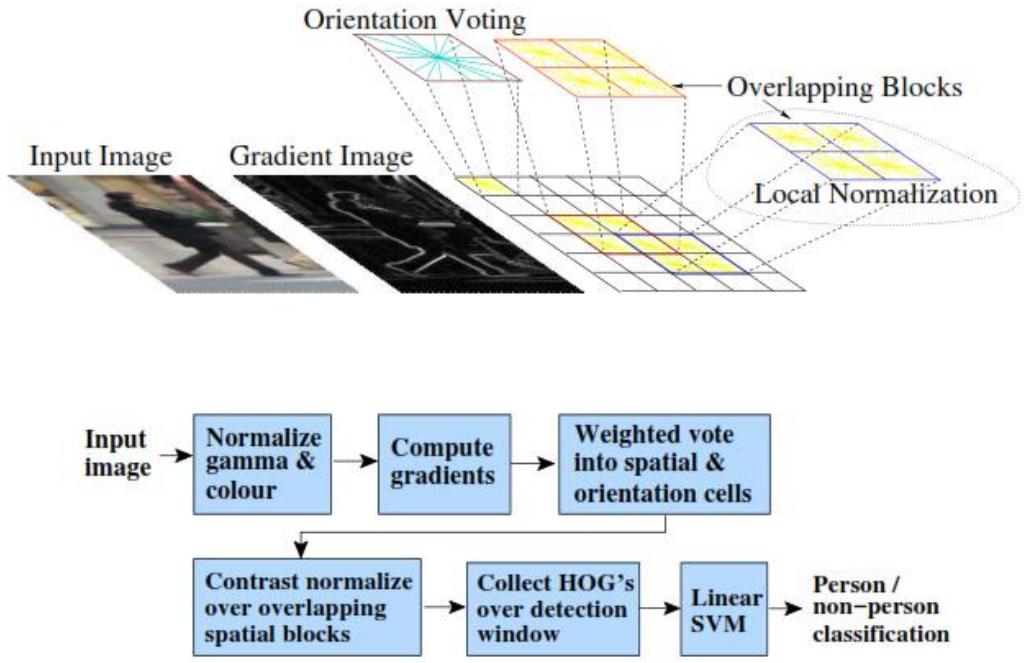


Figura 3.2: Funcionamiento del algoritmo HOG, figura extraída de [13].

3.2.3 Parámetros configurables y valores

Desde la página web del autor (http://pascal.inrialpes.fr/soft/olt/), descargamos el código fuente para sistema operativo Linux descargando el archivo *OLTbinaries.zip*. Una vez descomprimido, dentro de la carpeta *OLTbinaries*, encontramos un archivo *readme* con las instrucciones para ejecutar el programa. Entre los archivos existentes en esta carpeta, encontramos el archivo *runonimage.sh* donde se encuentran los parámetros de funcionamiento para este algoritmo. En la Figura 3.3 se muestra el contenido de este fichero. Inicialmente, de entre los parámetros existentes, se pensó en modificar los valores de ***scaleratio*** (1,05) y de tamaño base de la persona (64 x 128). Tras una serie de pruebas

modificando los valores observamos que el tamaño de la persona, entre otros (ver sección 3.2.4), no son parámetros modificables debido a que se corresponden con los modelos de persona entrenados por el autor y por lo tanto no los modificamos. Por ello, de los parámetros opcionales, escogimos variar ***scaleratio*** que es el coeficiente de incremento del tamaño de la persona con respecto al tamaño de la imagen y que nos permite observar qué ocurre al modificar cuantas veces se escala la ventana en el barrido. Además, una vez realizada la configuración por defecto, fijamos el umbral de detección al mínimo para poder tener la información de todas las detecciones desde ese valor mínimo hasta el máximo y así poder tener las curvas ***1-Precisión/Recall*** para realizar la comparación de todos los algoritmos y sus diferentes configuraciones. El resto de configuraciones con diferentes valores de ***scaleratio*** utilizan el umbral mínimo (valor 0) y pueden verse en la Tabla 3.1.

Para decidir los valores que daríamos a la variable ***scaleratio*** utilizamos la razón de 2/4; 3/4; 4/4; 5/4 sobre el valor original. El tamaño base de persona fijado por el autor en este algoritmo es de 64x128. En cada una de sus iteraciones este valor permanece constante, lo que varía es el tamaño de la imagen que se transforma con un factor 1/*scaleratio*. De esta forma el resultado que obtenemos en la detección es partir de una persona de tamaño 64X128 en la imagen inicial a que esta relación de aspecto admita personas más grandes con cada iteración, no variamos el tamaño en píxeles de persona pero al variar el tamaño de la imagen conseguimos una relación de persona más grande con respecto a la imagen. Este proceso se repite mientras el tamaño de la imagen que se procesa sea mayor que el tamaño base de persona. Así la relación de aspecto entre persona e imagen en una primera iteración pasará de 64X128 a 68X131,75 (con respecto a la imagen de tamaño original).

HOG					
Parámetro	Inicial	A (2/4)	B (3/4)	C (4/4)	D (5/4)
Umbral	0,1	0	0	0	0
Scaleratio	1,05	1,025	1,0375	1,05	1,0625

Tabla 3.1: Configuración de parámetros del algoritmo HOG.

```
#!/bin/sh
if [ $# -lt 3 ]
then
    echo "<usage>: runonimage.sh <image name/image directory/list file> "
    echo "          <out text file> <out image file/out image dir>"
else
    InFile=$1
    OutFile=$2
    ImageFile=$3

WIDTH=64; export WIDTH
HEIGHT=128; export HEIGHT

BinDIR=./bin
Classifier=$BinDIR/classify_rhog

Option=" -W $WIDTH,$HEIGHT \
    -C 8,8 -N 2,2 -B 9 -G 8,8 -S 0 --wtscale 2 --maxvalue 0.2 --epsilon 1 \
    --fullcirc 0 -v 3 --proc rgb_sqrt --norm l2hys "

    ########
Margin=4
ExtraOption2=" -t 0.1 -m 0 "
ExtraOption1=" -p 1,0 --no_nonmax 0 -z 8,16,1.3 \
    --cachesize 128 --scaleratio 1.05 --winstride 8 \
    --margin $Margin,$Margin --avsize 0,96 "
ExtraOption=$ExtraOption1$ExtraOption2

echo Running hog on image list
ResultDir=$OutFile
CMD="$Classifier $Option  $ExtraOption \
    $InFile $OutFile HOG/model_4BiSVMLight.alt -i $ImageFile "
$CMD
fi
```

Figura 3.3: Código del ejecutable para el algoritmo HOG con los parámetros configurables.

3.2.4 Configuración inicial del autor

Las características por defecto para este algoritmo fijadas por el autor se describen a continuación:

- Espacio de color RGB sin corrección gamma.
- [-1, 0, 1] filtro de gradiente sin suavizado.
- Votación lineal del gradiente en 9 bins de votación entre 0º-180º.
- Bloques de 16 x 16 píxeles de cuatro celdas de 8 x 8 píxeles.
- Ventana espacial Gaussiana con $\sigma = 8$ píxeles.
- Normalización de bloque L2-Hys (norma de recorte estilo Lowe L2).
- Tamaño de bloque de 8 píxeles (por lo tanto la extensión es de 4 veces por cada celda).
- Ventana de detección de 64 x 128.
- Clasificador lineal SVM.

3.3 Implicit Shape Model (ISM)

3.3.1 Introducción

El siguiente algoritmo estudiado está enfocado a la detección de personas en escenarios muy concurridos, reflejando lo que ocurre en multitud de ocasiones en la vida real. Para llevar a cabo su tarea, [26], nos presenta un algoritmo que integra evidencias en múltiples iteraciones y de diferentes fuentes. Como en el caso anterior, nos encontramos un algoritmo que se basa en una detección de las regiones de interés mediante una búsqueda exhaustiva basada en características y una fase de decisión que utiliza el modelo holístico de persona. Para lograrlo, deben unir los puntos fuertes de varios enfoques que utilizan tanto la apariencia como la forma, e integrar esta información de forma local y global.

A través del muestreo de características de forma local y su combinación, se genera la hipótesis de la localización de objetos en la imagen. Para cada uno de estos "supuestos objetos" se realiza una segmentación probabilística que puede ser utilizada para resolver ambigüedades en regiones con superposiciones entre posibles personas.

3.3.2 Funcionamiento

El autor [26] divide en 2 fases principales el funcionamiento del algoritmo. La primera es la aproximación de reconocimiento inicial. Esta fase se basa en una modificación invariante a escala del modelo de forma implícita (ISM - *Implicit Shape Model*).

En primer lugar tenemos la fase de entrenamiento en la que aprendemos un diccionario a partir de modelos de persona. Esta tarea se realiza aplicando un extractor de puntos de interés invariante a escala, el autor, por ejemplo, utiliza los puntos SIFT (Scale-Invariant Feature Transform) [37]. Tras esto, todas estas zonas extraídas alrededor de los puntos son escaladas a un tamaño fijo (25 x 25 píxeles por defecto) y agrupadas utilizando un esquema de agrupación de aglomeraciones (*agglomerative clustering scheme*).

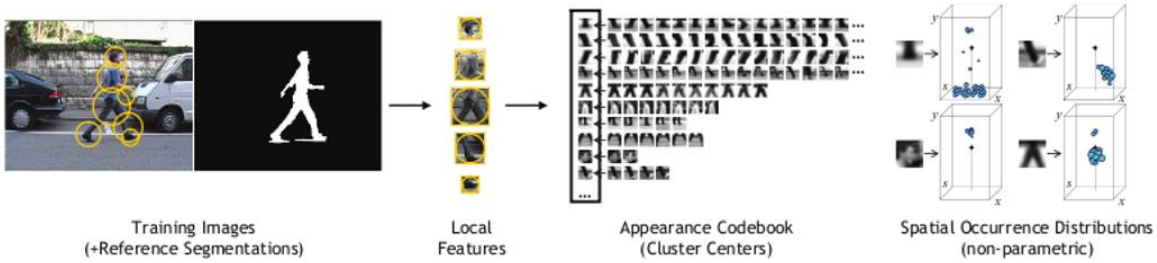


Figura 3.4: Fase de entrenamiento del algoritmo ISM, figura extraída de [6].

Esta agrupación forma una representación compacta de la estructura local y aquellas agrupaciones promedio son guardadas como palabras del diccionario. A continuación, la información local de las zonas muestreadas es recogida en un procedimiento de voto probabilístico de Hough (*probabilistic Hough voting procedure*). Cada zona es emparejada con las palabras del diccionario y votada conforme a la distribución de probabilidad espacial aprendida (ver Figura 3.4).

La fase de test o ejecución consistirá en extraer los puntos de interés, a continuación emparejamos esos puntos de interés con nuestro diccionario y generamos la votación espacial con la información de características de nuestro diccionario. Tras esto, realizamos la votación espacial en 3 dimensiones para calcular los centros de la persona dentro de la imagen. Esta cálculo se hace utilizando una Transformada de Distancia (*Distance Transform*), que calcula para cada pixel la distancia más corta a un pixel característico, y tras esto se calcula la media de todas las distancias de los píxeles que componen el objeto

candidato. Una vez que tenemos estos valores máximos que nos ubican el centro de la persona y con la información de los modelos entrenados generamos nuestra hipótesis de persona dentro de la imagen inicial, como fase final, se procede a la segmentación de nuestra detección. En la Figura 3.5 puede verse el funcionamiento del algoritmo.

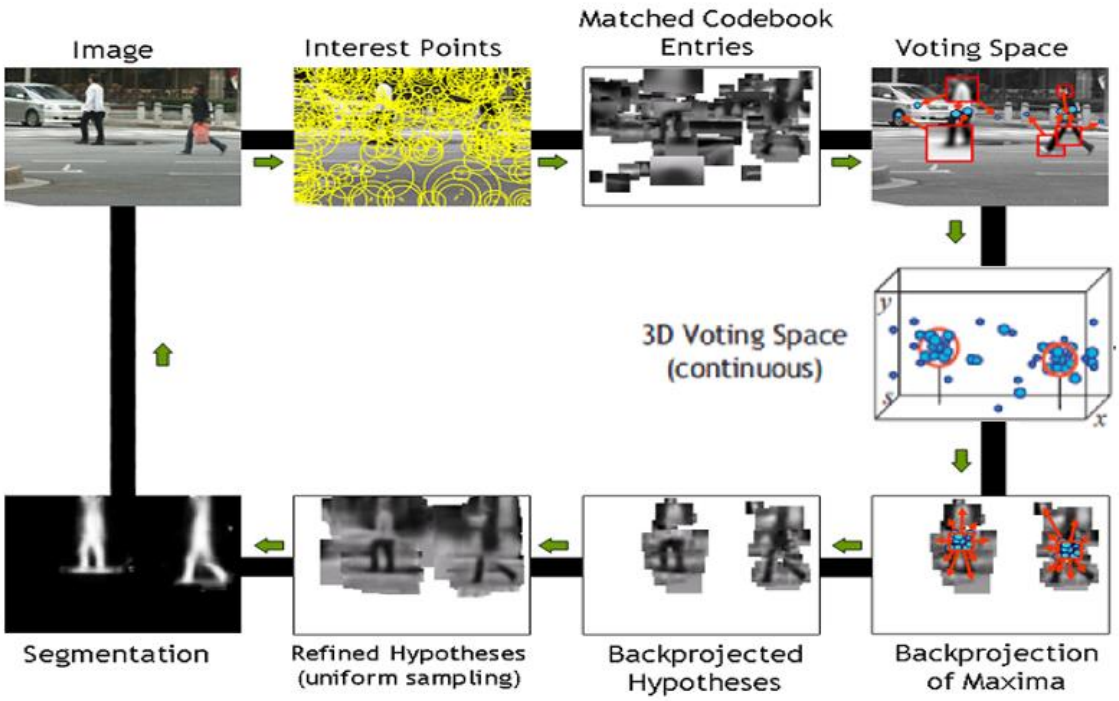


Figura 3.5: Funcionamiento del ISM, figura extraída de [26].

3.3.3 Parámetros configurables y valores

Desde la página web del autor (http://www.vision.rwth-aachen.de/software/ism/ism-detector-code) podemos acceder a descargar tanto la aplicación como las instrucciones de instalación. En esta ocasión también trabajaremos en entorno Linux. Una vez instalado, arrancaremos el programa, entorno gráfico, mediante la sentencia ./start.sh. En la Figura 3.6 se muestra el entorno de ejecución.

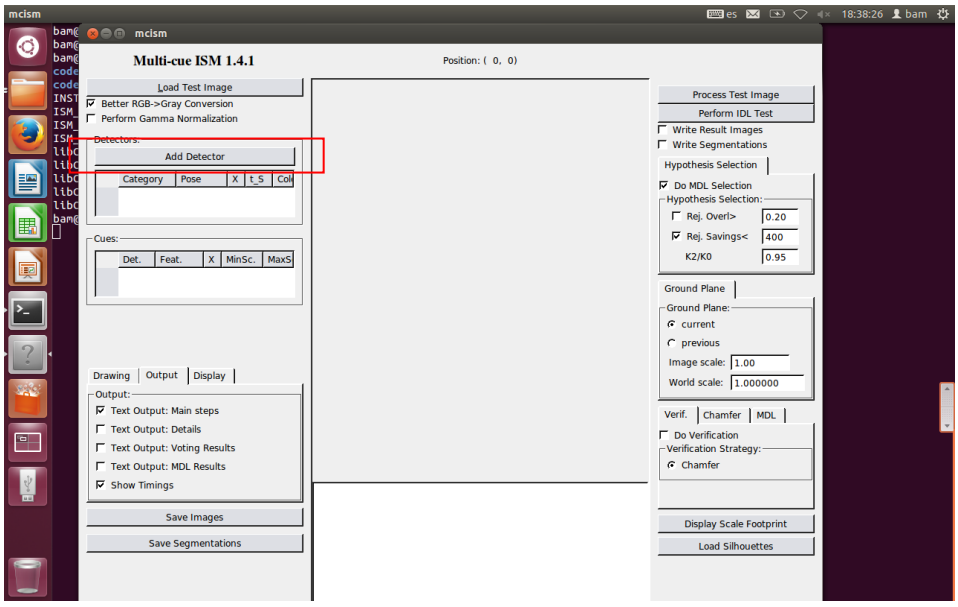


Figura 3.6: Entorno gráfico del algoritmo ISM.

Una vez aquí, deberemos seleccionar el detector a utilizar, el autor ofrece tres detectores por defecto con sus parámetros asociados, los ficheros de entrada y salida (ambos serán .idl) y aquello parámetros que queramos configurar de entre los posibles. La Figura 3.6 muestra la forma de cargar los detectores, así como la denominación de los mismos.

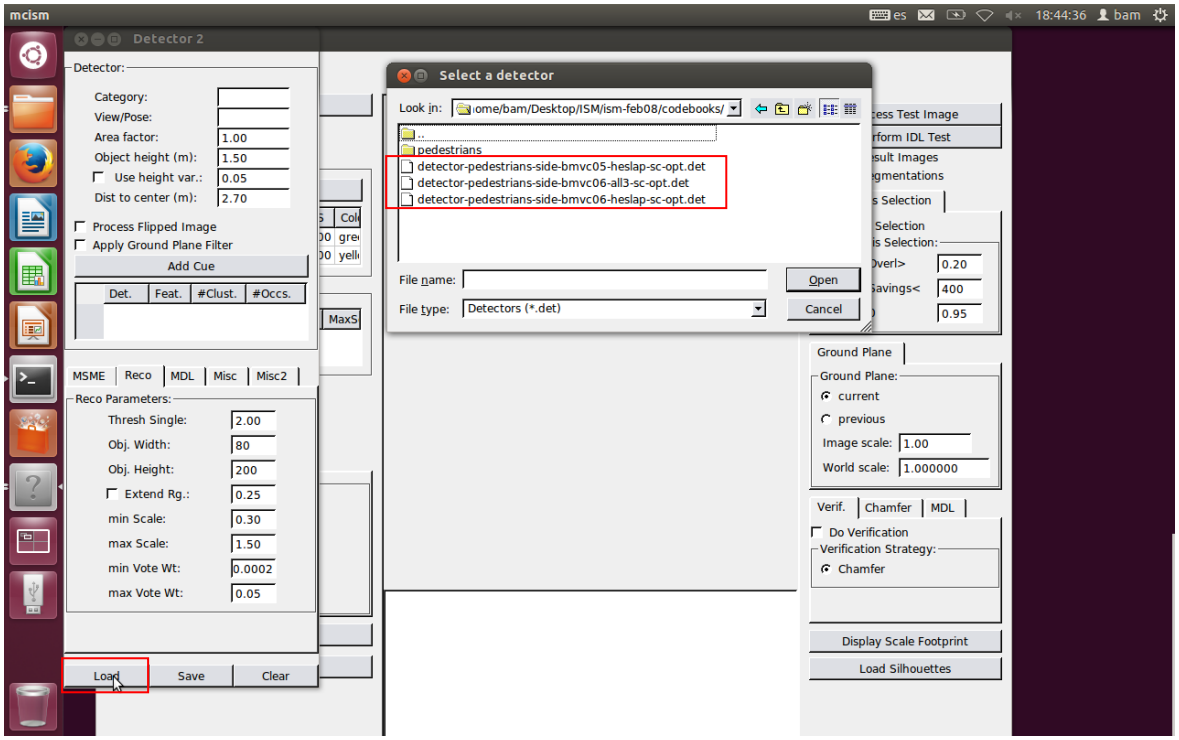


Figura 3.7: Detectores por defecto del algoritmo ISM.

Para la prueba con nuestro *dataset* hemos escogido el tercer detector, detector-pedestrians-side-bmvc06-heslap-sc-opt.det, que en las pruebas es el que mejores resultados obtenía en las labores de detección. De los múltiples parámetros configurables en este algoritmo hemos decidido modificar solamente dos, además del umbral. Parámetros como Reco→Thresh Single, Area Factor o Reco→Min Scale son específicos del modelo de persona y por eso se ha decidido no modificarlos. Por una parte el umbral se ha probado con el "por defecto", 600, y se ha decidido bajar a 300 para poder incluir todo tipo de detecciones, en la pantalla de configuración este parámetro se denomina MDL y en la memoria de configuración del autor *final hypotesis verification threshold*. Así mismo hemos modificado el valor de *Max Scale* para lograr detectar figuras "grandes", como nuestro *dataset* consta de muy diversas capturas con distintos enfoques debemos asegurar que el algoritmo sea capaz de detectar "personas pequeñas" (lejanas), pero también debemos asegurar detectar a "personas grandes" (cercanas/primer plano). En el manual de configuración esta característica se encuentra en *feature extractor scale range* y dentro del mismo es el parámetro *Max Scale*. De esta forma, el tamaño de persona realizará un barrido con escalas desde 0.3 hasta el valor máximo que hemos fijado (inicial 1.5; final 2.5), este escalado se aplica al valor por defecto de tamaño de persona de 80x200. De esta manera los tamaños de persona mínimos que se considerarán serán de (80X200)*0,3 = (24X60) y en el caso del máximo, valor que fijamos con *Max Scale*, tendremos (80X200)*1,5 = (120X300), llegando, con el valor *Max Scale* de 2,5, a un tamaño de persona de (200X500). La Tabla 3.2 recoge las diferentes configuraciones probadas para el

algoritmo ISM. En este caso tenemos tres configuraciones iniciales del autor, una por cada uno de los detectores, sus etiquetas son det1, det2, det3.

ISM									
Parámetro	det 1	det 2	det 3	det 1	det 2	det 3	A	B	C
Umbral	600	600	600	300	300	300	300	300	300
Max Scale	1,5	1,5	1,5	1,5	1,5	1,5	1,5	2	2,5

Tabla 3.2: Configuración de parámetros del algoritmo ISM.

3.3.4 Configuración inicial del autor

Como la Figura 3.7 muestra, el autor ofrece 3 configuraciones base en función de sus tres detectores. Denominados detector-pedestrians-side-bmvc05-heslap-sc-opt.det (det 1), detector-pedestrians-side-bmvc06-heslap-sc-opt.det (det 2) y detector-pedestrians-side-bmvc06-all3-sc-opt.det (det 3), el autor ha entrenado sus detectores con diferentes grupos de imágenes de entrenamiento y diferentes parámetros de configuración. La Figura 3.8 muestra las diferentes configuraciones para cada uno de los tres detectores.

Figura 3.8: Configuraciones iniciales del autor para sus tres detectores.

3.4 Discriminatively Trained Deformable Part-based model (DTDP)

3.4.1 Introducción

El siguiente algoritmo estudiado [14], es un modelo que utiliza un filtro raíz similar al HOG [13] añadiendo un conjunto de filtros por partes y modelos de deformación asociados a ellos. A diferencia del primer algoritmo que hemos estudiado en este PFC nos encontramos con un modelo de persona no holístico, sino basado en partes, la detección de las regiones de interés se sigue basando en búsqueda exhaustiva pero en esta ocasión utilizando ventanas deslizantes. Para la caracterización de las persona reformulan una SVM en cuanto a las variables latentes (variables que no se observan directamente) y lo denominan Latent SVM (LSVM), esto es lo que principalmente diferencia esta técnica de la vista en el primer algoritmo que utilizaba una SVM lineal. La utilización de esta variante de la SVM es utiliza para poder aprender el modelo de persona y sus partes sin tener un etiquetado fiable de las partes.

3.4.2 Funcionamiento

En este caso, [14], obtiene las puntuaciones del filtro de raíz y de los filtros por partes utilizando la técnica de HOG sobre cada parte del cuerpo de forma independiente. Los filtros por partes capturan las características al doble de resolución espacial que las que se capturan con el filtro raíz y permiten "deformaciones", i.e., modificaciones de sus posiciones respecto del filtro raíz mediante modelos de deformación aprendidos durante el entrenamiento. Tras esto, y como ya hemos indicado anteriormente, utilizan un clasificador LSVM para lograr la calificación final de las personas. Por lo tanto, logran un mejor rendimiento al utilizar modelos por partes más complejos y que se comportan mejor frente a posturas o "deformaciones" sobre los modelos aprendidos. La Figura 3.9 muestra el funcionamiento del algoritmo. En la parte superior derecha de la figura se puede observar la flexibilidad en la detección a través de las dilataciones, cuanto más nos alejamos del centro más diferencia del modelo aprendido se acepta. Por ejemplo, observamos que se permite un mayor grado de diferencia en la zona de los brazos, permitiendo movimiento lateral.

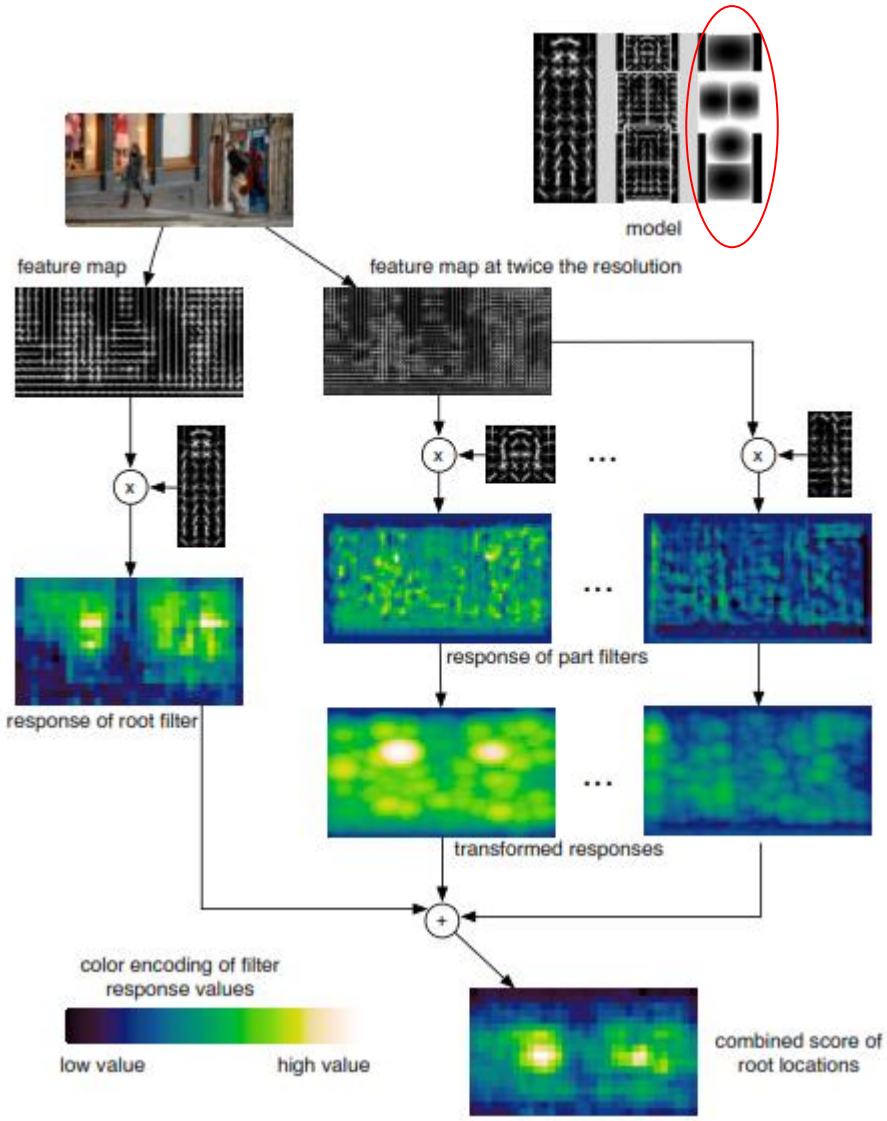


Figura 3.9: Funcionamiento del algoritmo, imagen extraída de [14].

3.4.3 Parámetros configurables y valores

Desde la página web del autor (http://www.cs.berkeley.edu/~rbg/latent/) podemos acceder al código y descargarlo. En esta ocasión el código está escrito en MATLAB y su ejecución la realizamos en un entorno Windows. El autor fija un umbral mínimo de -0.3 para las detecciones, como en los anteriores casos, se ha decidido utilizar un umbral aún menor para poder trazar la curva de detección completa y lo hemos fijado en el valor -1.5.

A diferencia de los anteriores algoritmos, el autor no facilita un entorno de ejecución de vídeos o de secuencias de imágenes, por ello hemos implementado dos ficheros de Matlab, el primero test_DTDP_detector.m, encargado de pasar parámetros como el vídeo a ser analizado o los que queremos modificar. El segundo DTDP_Detector.m será el encargado de recibir los parámetros y realizar la llamada a la función de detección del autor imgdetect.m que se encuentra dentro de la carpeta voc-release4.01, que descargamos desde la web del autor y debemos descomprimir en una carpeta con ese mismo nombre (fichero descargado voc-release4.01.tgz). En este segundo fichero es donde nosotros fijaremos los valores de los parámetros que hemos decidido configurar.

Los parámetros a modificar los pasaremos como argumentos de funciones y pueden verse dentro de la estructura de la Figura 3.10.

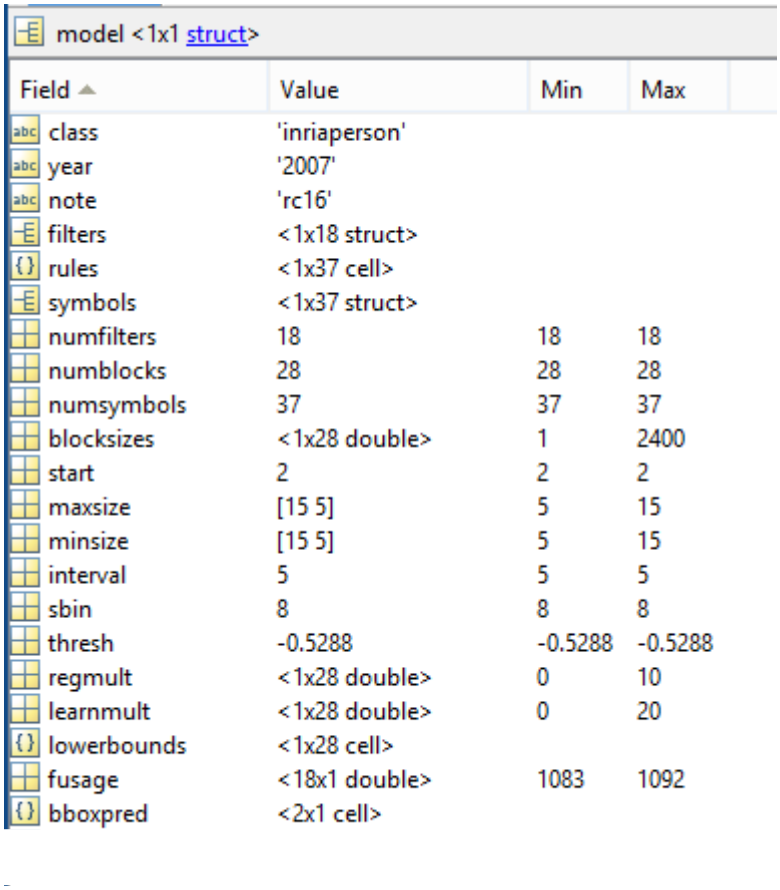


Figura 3.10: Estructura model con los parámetros del algoritmo DTDP.

De los diferentes parámetros que se pueden ver en la figura anterior, la mayoría pertenecen al modelo de persona entrenado por el autor. Por ejemplo *filters* corresponde con los 18 modelos de partes del cuerpo propios del modelo de entrenamiento, *rules* se corresponde con la forma de combinar esas partes. El tamaño mínimo de persona se basará en los parámetros *maxsize* y *minsize* y será proporcional al valor de *sbin*. Para el cálculo del tamaño de persona partimos del tamaño base fijado por el autor (15X5) y le aplicaremos un factor de escala basado en el valor de *sbin*. Así para un valor *sbin* de 8 tendremos que el tamaño base de persona será de (15X5)**sbin*/2 = (60X20). El otro parámetro que hemos modificado, *interval*, nos fijará el número de inter-escalas. Para modificar los parámetros que no dependen del modelo aprendido utilizaremos los argumentos de las funciones como puede verse a continuación en las capturas de pantalla de ejecución, Figura 3.11, Figura 3.12:

```
DTDP_detector(images_names,threshold,video_names{i},video_dir,4,5);
```

Figura 3.11: Llamada a la función DTDP_detector.m con el paso de los parámetros sbin e interval del modelo.

```
function DTDP_detector(images_names,threshold,video,video_dir,sbin,int)

addpath('./voc-release4.01');
load('./voc-release4.01/INRIA/inriaperson_final.mat');
model.sbin = sbin;
model.interval = int;
for i=1:size(images_names,2)
    cadena=sprintf('%s/%s',video_dir,images_names{i});
    test(cadena, model,images_names{i},threshold,video,video_dir,i);
end
clear model
```

Figura 3.12: Asignación de los nuevos valores a los parámetros elegidos.

DTDP										
Parámetro	Ini	A	B	C	D	E	F	G	H	I
Umbral	-0,3	-1,5	-1,5	-1,5	-1,5	-1,5	-1,5	-1,5	-1,5	-1,5
sbin	8	4	4	4	8	8	8	16	16	16
interval	10	5	10	15	5	10	15	5	10	15

Tabla 3.3: Configuración de parámetros del algoritmo DTDP.

3.4.4 Configuración inicial del autor

Los valores por defecto considerados por el autor para el DTDP pueden verse en la Figura 3.10. En la misma observamos como el resto de valores pertenecen al modelo de persona entrenado por el autor por los que no se modificarán esos parámetros. En el caso de DTDP hemos modificado el valor de *interval* que nos dirá cuántas inter-escalas existirán por escala grande, a mayor valor te *interval*, mejor puedo detectar pequeñas variaciones de tamaño de persona dentro de un rango específico pero también corro el riesgo de obtener más falsos positivos. En el caso de *sbin* define el tamaño base de persona y por lo tanto si podemos detectar personas grandes, pequeñas o ambas. Las configuraciones elegidas para la parametrización del algoritmo pueden verse en la Tabla 3.3.

3.5 Fusion

3.5.1 Introducción

El siguiente algoritmo [7] estudiado ha sido desarrollado por el VPULab y se basa en una primera fase de extracción de fondo, basado en la propuesta de [29], después se realiza la extracción del objeto mediante operadores morfológicos con el fin de reducir el ruido de la imagen extraída, tras esto, se fusionan las evidencias de tres detectores que trabajan de forma independiente. Los tres detectores son basados en silueta. En este caso el algoritmo utiliza la segmentación de fondo y el modelo holístico de persona. Tanto *Fusion* como *Edge* han sido introducidos en la plataforma DiVa (Distributed Video Analysis Framework) por [38].

3.5.2 Funcionamiento

Una vez realizada la extracción de fondo y de objeto, se recogen una serie de características relacionadas con el modelo de persona para cada blob. A continuación se

fusionan en una evidencia combinada que será umbralizada para decidir si ese blob es persona o no.

El primer detector se basa en la relación de aspecto, definida como el cociente entre la altura y la anchura. El segundo detector se basa en el cálculo, de forma iterativa hasta un máximo fijado, de la mayor elipse contenida dentro de la región extraída como objeto para ese blob. De esta medida sacamos la relación entre número de iteraciones para alcanzarlo y el máximo prefijado; porcentaje de puntos de la elipse que se quedan fuera del blob; y la relación de aspecto de la elipse. El resultado final de esta evidencia es la media de estos tres valores. El último detector se basa en el algoritmo *Ghost* [31], que aproxima el contorno del blob al de un polígono cerrado. De este polígono sacamos el número de puntos del polígono; relación entre cantidad de vértices convexos y no convexos que normalmente es balanceada para una persona; y el inverso del número de vértices que pertenecen a la parte superior del polígono. La evidencia final de este tercer detector es la media de las tres evidencias siempre que las dos últimas superen el valor umbral de 0,6.

Para calcular el valor de la evidencia final de cada blob, y por lo tanto si es aspirante real a persona o no, se obtiene promediando las evidencias proporcionadas por los tres detectores, teniendo en cuenta que las del primer y segundo detector solo serán consideradas si superan un valor umbral prefijado. Para la decisión final este valor de evidencia debe superar el umbral de decisión.

3.5.3 Parámetros configurables y valores

Tanto *Fusion* como *Edge* son algoritmos del VPULab por lo que accedemos al código mediante el repositorio en red del laboratorio. En este caso el algoritmo está programado en C++ y utilizaremos el Visual Studio para compilarlo y poder generar los archivos para su ejecución. En ambos casos tanto los parámetros como la forma de convocarlos a la hora de su utilización se realizarán mediante un script y serán los mismos en los dos algoritmos. A la hora de configurar los parámetros los únicos que tiene sentido modificar son los que se refieren a la segmentación de frente/fondo, *varnoise* (sensibilidad al ruido) y *window*_Q (ventana de segmentación).

Cuando mayor sea el valor de *varnoise* mayor será la robustez de la detección frente al ruido. El balanceo adecuado de este valor es fundamental puesto que un valor muy elevado puede hacer que desechemos mucha información y un valor muy pequeño puede hacer que la segmentación no sea efectiva.

En cuanto al otro parámetro, nos indica el tamaño de la ventana de segmentación, esto quiere decir que a mayor tamaño mayor número de píxeles vecinos son utilizados para la ventana de segmentación, resultando en una segmentación más gruesa o más fina.

Las diferentes configuraciones elegidas para estos parámetros pueden verse en la Tabla 3.4.

Fusion (umbral = 0)																				
Par	Ini	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
var	13	5	5	5	5	5	10	10	10	10	10	17	17	17	17	17	29	29	29	29	29
winQ	3	1	3	5	7	9	1	3	5	7	9	1	3	5	7	9	1	3	5	7	9

Tabla 3.4: Configuración de parámetros del algoritmo *Fusion*.

3.5.4 Configuración inicial del autor

El autor asigna unos valores por defecto de 13 para *varnoise* y de 3 para la ventana de detección. Para el valor del umbral hemos fijado el valor 0 puesto que el autor no determinaba un valor inicial para este parámetro.

3.6 Edge

3.6.1 Introducción

Como el algoritmo anterior, *Fusion* [7], éste [27] también ha sido desarrollado por el VPULab, comparte con Fusion la primera etapa de segmentación para extraer las ROI y a continuación, a diferencia de *Fusion*, utilizaremos un modelo basado en partes para la persona. Basado en [32], el autor propone un método de detección de personas para escenas concurridas, trabajando con imágenes estáticas y sin tracking, a diferencia del caso anterior que utilizaba esta información para la detección de personas. Una vez que tenemos las ROI, para el modelo de persona, utilizaremos cuatro detectores de borde independientes entrenados previamente.

3.6.2 Funcionamiento

El algoritmo se basa en identificar las características de borde de las partes del cuerpo (cuerpo entero, cabeza, torso y piernas) y caracterizarlos mediante *Edgelets*, definidos como un conjunto de desplazamientos y orientaciones que describen el borde. En la Figura 3.13 se pueden observar los diferentes detectores que se utilizan en este algoritmo.

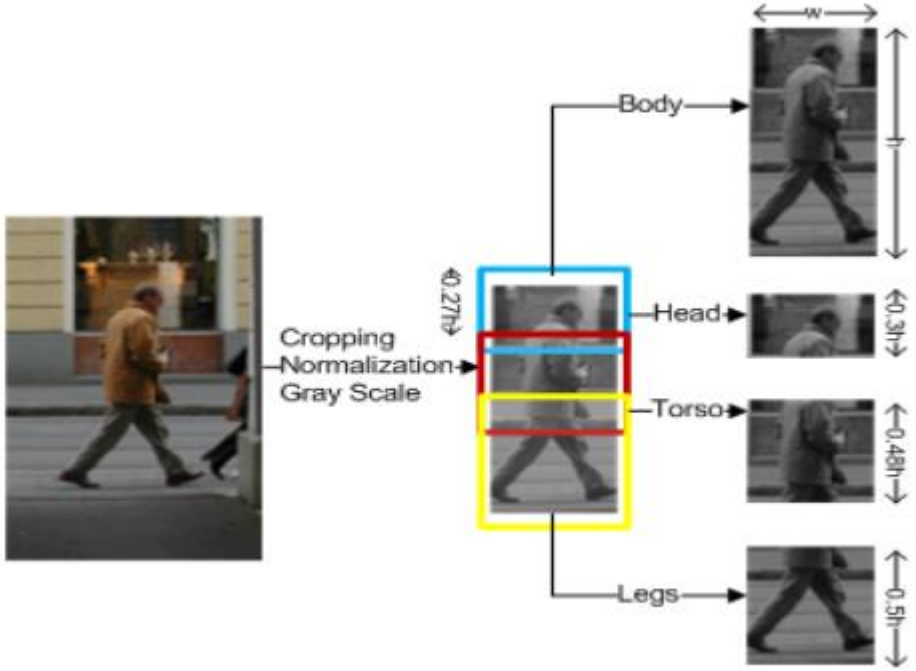


Figura 3.13: Detectores de borde para cada parte del cuerpo, imagen extraída de [27].

Para los detectores eliminamos la información de color de las imágenes y trabajamos con las imágenes normalizadas en escala de grises. Para entrenar los modelos el autor utiliza el algoritmo *Real Adaboost* [33] y una estructura en cascada anidada [34]. Se combinarán las respuestas de cada detector para obtener un modelo de probabilidad conjunta. Se construye un clasificador débil para cada borde característico y a continuación se utiliza *AdaBoost* [33] para convertirlos en clasificadores fuertes. Dentro de los bordes entrenados obtendremos una mejora en el tiempo de computación y la fortaleza del detector utilizando el algoritmo *AdaBoost* [33] para, de entre todos los modelos entrenados, obtener los 100 más característicos. Una vez que tenemos las respuestas de cada uno de los detectores las combinamos utilizando un modelo de probabilidad conjunta.

3.6.3 Parámetros configurables y valores

Como hemos indicado en *Fusion* tanto los parámetros como su configuración y valores iniciales coinciden con los de *Edge*, en la Tabla 3.5 pueden verse los valores de los parámetros para este algoritmo.

Edge (umbral = 0)																				
Par	Ini	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
var	13	5	5	5	5	5	10	10	10	10	10	17	17	17	17	17	29	29	29	29	29
winQ	3	1	3	5	7	9	1	3	5	7	9	1	3	5	7	9	1	3	5	7	9

Tabla 3.5: Configuración de parámetros del algoritmo Edge.

3.6.4 Configuración inicial del autor

Coinciden con los de *Fusion*, ver sección 3.5.4.

3.7 Aggregate Channel Features (ACF)

3.7.1 Introducción

El último algoritmo analizado, [28], realiza búsqueda exhaustiva y modelo de persona holístico.

El autor busca lograr un algoritmo más rápido para la detección de personas. Propone el uso de algoritmos de ventanas deslizantes y de detección con clasificadores en cascada tipo Adaboost, sin embargo el coste computacional de este sistema lo hace inviable para muchas aplicaciones. Para superar esto, realiza la reflexión de que en la actualidad el procesado de las ventanas adyacentes se hace de forma independiente sin considerar que existen correlaciones de escala y localización de las zonas anexas que pueden ser utilizadas para optimizar el proceso. Por ello propone la utilización de lo que denomina *Crosstalk Cascades*. El autor propone una implementación optimizada de las características multi escala de [36] e introduce un nuevo marco que acopla las evaluaciones en cascada para las zonas adyacentes, lo que reduce el coste computacional. La Figura 3.14 muestra los resultados de los histogramas de gradientes para la imagen original (centro), la magnitud del gradiente (derecha) y la orientación del mismo (debajo) para la imagen original (borde azul claro), la imagen sobre muestreada –x2- (borde azul oscuro) y la imagen sub muestreada –x2- (borde amarillo). Como podemos observar este histograma de gradientes muestra que existe correlación cuando se calcula para las ventanas adyacentes de escala.

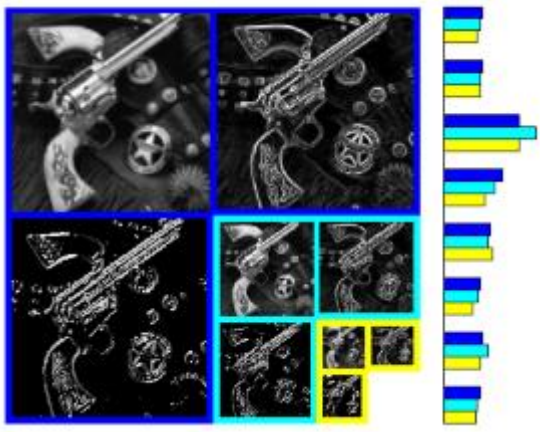


Figura 3.14: Histograma de gradientes para ventanas adyacentes en escala tanto en magnitud como en orientación. Imagen extraída de [36].

3.7.2 Funcionamiento

El autor se basa en la utilización del cómputo de características (gradiente, color, …) con la misma dimensión. Las sumas de cada canal sobre regiones rectangulares sirven como características y se pueden calcular de manera eficiente utilizando la imagen integral. En la detección multi-escala, las características se calculan normalmente sobre una pirámide densa de imágenes. En este caso, el autor observó que las estadísticas demuestran que las imágenes siguen una ley de potencias en las zonas cercanas sobre una pirámide ligera de imágenes. Por ello, el autor propone una re implementación optimizada de las características de los canales. Para la detección de las ROI se utilizan los canales de magnitud de gradiente (1 canal), HOG (6 canales) y canales de color LUV (3 canales), un ejemplo de la descomposición de una imagen en estos 10 canales puede verse en la zona superior de la Figura 3.15, en la parte inferior de la misma imagen observamos las aportaciones individuales y la combinación final de todas. Para cada escala estos 10 canales son sub muestreados por un factor de 2 para mejorar la velocidad. Para el detector base el autor utiliza la misma configuración que [35]. Aplicando AdaBoost para entrenar y combinar con 4096 árboles *depth-two* utilizando una fuente de 30000 características candidatas aleatorias calculadas sobre los 10 canales descritos anteriormente. Como ya comentamos en la introducción sobre este algoritmo, existe una correlación de las respuestas del detector para las zonas y escalas adyacentes. Cada detector tiene una región de apoyo (ROS), que es el centro de la persona entre los diferentes puntos cercanos que son candidatos. Un detector de ROS se define por las características, capacidad discriminadora del clasificador y por el alineamiento con los datos entrenados.

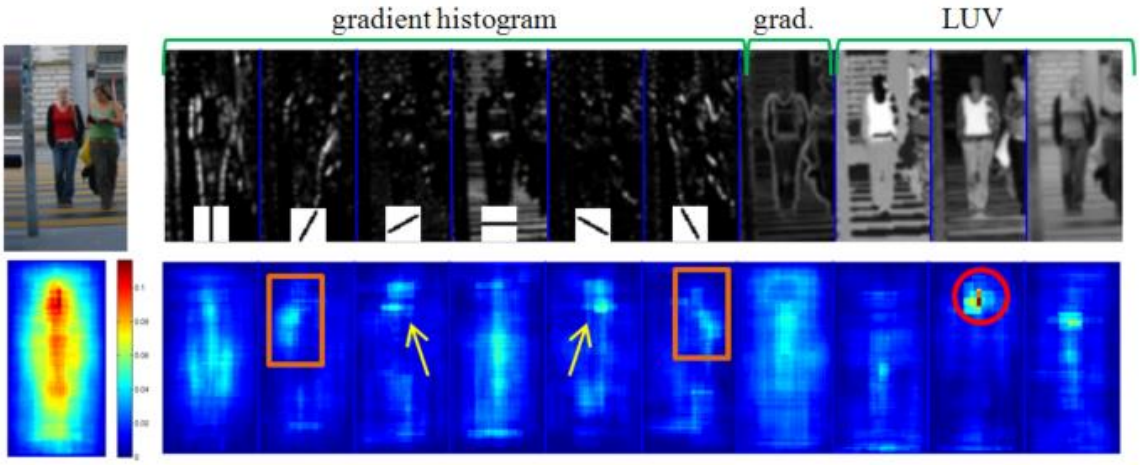


Figura 3.15: Análisis de la imagen mediante su descomposición en los 10 canales considerados por el autor, extraída de [35].

La detección final por cada ROS se realiza haciendo el promedio de las respuestas del detector a través de múltiples ventanas cuyo centro contiene un máximo de respuesta.

Para terminar, el autor presenta 3 posibles modelos de cascada que, combinadas darán lugar a *Crosstalk Cascades*. Estas son *soft cascades*, *excitatory cascades* e *inhibitory cascades*. En cuanto al modelo de persona, el entrenamiento utiliza dos bases de datos diferentes para aprender el modelo. Por un lado tendremos los que, para su modelo de persona, utilizan la base de datos Inria [13] y por otro las que utilizan la de Caltech [5].

3.7.3 Parámetros configurables y valores

Desde la página web del autor (http://vision.ucsd.edu/~pdollar/toolbox/doc/index.html) podemos acceder al código y descargarlo. En esta ocasión el código está escrito en MATLAB y su ejecución la realizamos en un entorno Windows. El autor fija un umbral mínimo de -1 para las detecciones, como en los anteriores casos, se ha decidido utilizar un umbral aún menor para poder trazar la curva de detección completa y lo hemos fijado en el valor -10.

Se han implementado dos ficheros, el primero test_ACF_detector.m, encargado de seleccionar el modelo (Inria o Caltech) así como de pasar parámetros como el vídeo a ser analizado. El segundo ACF_Detector.m será el encargado de recibir los parámetros y realizar la llamada a la función de detección del autor acfDetect.m que se encuentra dentro de la carpeta toolbox/detectors, que se encuentra en el fichero descargado desde la web del autor tras descomprimirlo (fichero descargado piotr_toolbox.zip). En este segundo fichero es donde nosotros introduciremos los parámetros que hemos decidido configurar.

En primer lugar debemos seleccionar el modelo que vamos a utilizar, la Figura 3.16 muestra, dentro del fichero test_ACF_detector.m, cómo variar la utilización del modelo de Inria o de Caltech.

```
model_filename='./toolbox/detector/models/AcfCaltechDetector.mat';
model_id='Caltech';

% model_filename='./piotr_toolbox/toolbox/detector/models/AcfInriaDetector.mat';
% model_id='Inria';
```

Figura 3.16: Selección de modelo para el algoritmo ACF.

Una vez seleccionado el modelo, los diferentes parámetros a modificar los fijaremos en el fichero ACF_Detector.m mediante las líneas de código que se observan en la Figura 3.17 con los parámetros que le hemos pasado anteriormente al llamar a la función. La Figura 3.20 muestra los diferentes parámetros configurables para este algoritmo. Los parámetros que han decidido configurarse, *nPerOct* y *nOctUp*, son los que no pertenecen al modelo de persona con los que el autor ha entrenado su modelo de detector. En el caso de *nPerOct* modificamos el valor del número de inter escalas habrá por octava, a mayor valor mejor puedo detectar pequeñas variaciones de personas dentro de un rango específico, sería el equivalente a *interval* del DTDP. *nOctUp* define el número de escalas grandes, es decir si podemos detectar personas grandes, pequeñas o ambas, sería el equivalente a *sbin* en el DTDP y sus valores nos dan el tamaño base de persona, con el valor 0 tendremos el tamaño por defecto, con un valor -1 dividiremos el tamaño entre 2 y si vale 1 multiplicaremos el tamaño base por 2. En este caso, como podemos ver en la parametrización por defecto del autor en la Figura 3.19 el tamaño base de persona con *padding* será de (32X64), si a esto le aplicamos un valor de nOctUp de -1 tendremos un tamaño base de (32X64)*2 = (64X128), mientras que si aplicamos un nOctUp de 1, tendremos (32X64)*1/2 = (16X32) como tamaño base de persona que nos permitirá detectar personas más pequeñas pero también puede presentar más detecciones de falsos positivos.

```
ACF_detector.m × test_ACF_detector.m ×
 7 -   end
 8 -   clear model
 9
10
11
12
13
14
15    function test(name, model_filename,model_id,image_name,video,video_dir,num_frame,PerOct,OctUp)
16
17 -   load(model_filename);
18 -   detector.opts.pPyramid.nPerOct=PerOct;
19 -   detector.opts.pPyramid.nOctUp=OctUp;
20 -   detector.opts.cascThr=-10;
21 -   I = imread(name);
22
23    % detect objects
24 -   bbs = acfDetect( I, detector);
25
26
27 -   out_filename=sprintf('SALIDA/ACF DEFINITIVOS/%s/%s_acf_%s_nPerOct%d_nOctUp%.1f.idl',video,video,model_id,PerOct,OctUp);
28 -   if(num_frame==1)
29 -       fid=fopen(out_filename,'w+');
30 -       fclose(fid);
31 -   end
32
33 -   save_blobs(bbs,name,out_filename);
34 -   clear top clear bbox
35 -   clear final_blobs
```

Figura 3.17: Fijación de parámetros ACF.

ACF																
Par	Ini	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Umbral	-1	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10
nPerOct	8	4	4	4	4	4	8	8	8	8	8	12	12	12	12	12
nOctUp	0	-1	-0,5	0	0,5	1	-1	-0,5	0	0,5	1	-1	-0,5	0	0,5	1

Tabla 3.6: Configuración de parámetros del algoritmo ACF.

3.7.4 Configuración inicial del autor

La estructura del detector donde se encuentran sus parámetros se puede ver en la Figura 3.18. La configuración inicial de los parámetros del autor puede observarse a continuación en Figura 3.19 y Figura 3.20.

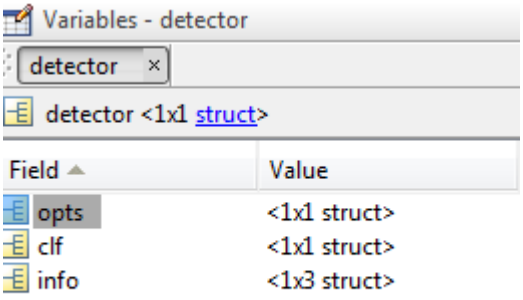


Figura 3.18: Estructura del detector ACF.

Field	Value	Min	Max
pPyramid	<1x1 struct>		
modelDs	[50 20.5000]	20.5000	50
modelDsPad	[64 32]	32	64
pNms	<1x1 struct>		
stride	4	4	4
cascThr	-1	-1	-1
cascCal	0.0050	0.0050	0.0050
nWeak	[32 128 512 2048]	32	2048
pBoost	<1x1 struct>		
seed	0	0	0
name	'models/AcfCaltech'		
posGtDir	'D:\code\research\de...		
posImgDir	'D:\code\research\de...		
negImgDir	''		
posWinDir	''		
negWinDir	''		
imreadf	@imread		
imreadp	<0x0 cell>		
pLoad	<1x1 struct>		
nPos	Inf	Inf	Inf
nNeg	5000	5000	5000
nPerNeg	25	25	25
nAccNeg	10000	10000	10000
pJitter	<1x1 struct>		
winsSave	0	0	0

Figura 3.19: Opciones del detector para ACF, *detector.opts*.

detector.opts.pPyramid <1x1 struct>

Field	Value	Min	Max
pChns	<1x1 struct>		
nPerOct	8	8	8
nOctUp	0	0	0
nApprox	7	7	7
lambdas	[0 0.1223 0.1223]	0	0.1223
pad	[8 8]	8	8
minDs	[50 20.5000]	20.5000	50
smooth	0.5000	0.5000	0.5000
concat	1	1	1
complete	1	1	1

Figura 3.20: Parámetros de *detector.opts.pPramid* del modelo ACF.

3.8 Conclusiones

En este capítulo hemos descrito los diferentes algoritmos que se han utilizado en este PFC. Para cada uno de ellos hemos explicado su funcionamiento, hemos hablado de sus parámetros de configuración y hemos escogido aquellos que no dependían del modelo de persona con que el autor ha decidido entrenarlos para su modificación y poder evaluar la influencia de esta variación sobre el funcionamiento del algoritmo.

4 Experimentación y resultados

4.1 Dataset

Durante todo este PFC hemos hablado de diferentes algoritmos que han sido probados en *datasets* fijados por sus autores obteniendo, en la mayoría de los casos, grandes resultados. Pero nuestro objetivo era saber cómo de válidos son esos resultados si los comparamos sobre una base común que incluya todo tipo de situaciones, escenarios, tipologías, etc. Uno de los grandes problemas a los que se enfrenta la investigación de detección de personas es encontrar la base de datos de prueba adecuada que de la suficiente fiabilidad a su algoritmo sin encasillarlo demasiado por las condiciones de vídeo, si a esto se le suma que para poder lograr un marco adecuado de comparación esta base de datos de imágenes debe ser abierta, o al menos estar disponible en general, y a ser posible de utilización gratuita, nos encontramos con una dificultad extra, no solo debemos desarrollar un gran algoritmo que resulte eficiente sino que necesitamos esa base de datos de prueba.

A la hora de elegir el mejor *dataset* para proceder a la evaluación de los diferentes algoritmos se tuvieron en consideración varios de ellos. En el *dataset* de Caltech [5] nos encontramos con un vídeo de muchas horas tomado desde un vehículo en movimiento y dirigido a la asistencia/ayuda en la conducción, esto hace que no recogiera el amplio espectro de situaciones que queríamos evaluar para lograr una mejor evaluación de los algoritmos. Por su parte el *dataset* de Inria [13] es un *dataset* que únicamente contiene imágenes y ha sido muy utilizado para aprender modelos de personas pero no tanto para evaluar. Otro de los *dataset* que consideramos para la evaluación es el PDds [39] propio del VPULab que no fue seleccionado debido a que mi trabajo debía basarse en realizar el estudio de los algoritmos sobre un *dataset* que añadiera escenarios a los ya existentes en el PDds puesto que ya habían sido evaluados sobre ese *dataset*.

En nuestro caso hemos escogido el *dataset* [30] http://www.changedetection.net/ que nos ofrece las características que buscamos en cuanto a diversidad, es abierto y además gratuito. Además este *dataset* fue diseñado para la labor de evaluación de la segmentación de fondo, por ello está diseñado para contemplar diferentes dificultades en la extracción de fondo que, como ya se ha comentado, es una de las fases críticas en la detección de personas.

Este *dataset* ha sido diseñado para recoger todo tipo de fondos e incidir en la primera etapa crítica de cualquier sistema de detección de personas, extraer las ROI's. El *dataset* completo comprende 31 vídeos con aproximadamente 70.000 *frames* recogidos en 6 categorías, dependientes del tipo de vídeo, diseñadas para la tarea de segmentación o extracción de fondo. A continuación se muestran las categorías y una breve descripción de las mismas.

Baseline

Esta primera categoría contiene cuatro vídeos con buena nitidez y fondos estáticos, en tres de ellos aparecen personas y han sido utilizados para la evaluación, en el otro, vídeo de una autovía, no se ha utilizado por no incluir personas en sus *frames*. Las personas que aparecen en estos vídeos contienen un tamaño mediano y la posición de la cámara es horizontal en dos de los tres utilizados. En la Figura 4.1 se puede observar la "portada" de cada uno de los vídeos.

Figura 4.1: Vídeos *Baseline.*

Dynamic Background

En esta categoría encontramos seis vídeos con un fondo dinámico, bien por el movimiento del agua, el movimiento de las ramas y hojas de los árboles o ambas. Se han utilizado dos de los seis vídeos. La disparidad en estos dos vídeos es grande, por una parte tenemos personas muy pequeñas y de fondo en el vídeo *fall*, mientras que en *overpass* la persona es un primer plano con un gran tamaño, en ambos casos la cámara captura la imagen horizontalmente. En la Figura 4.2 se puede observar la "portada" de cada uno de los vídeos.

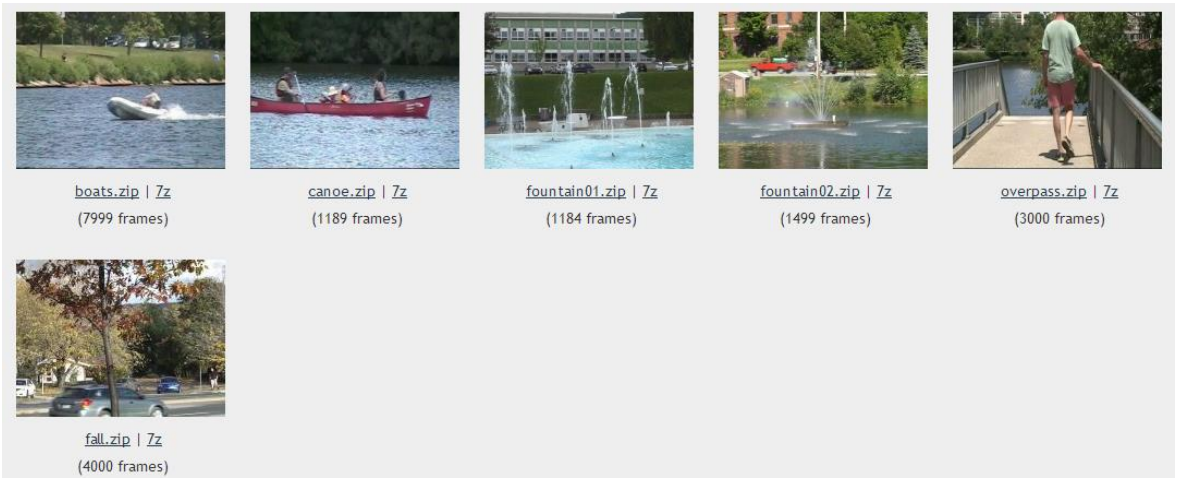


Figura 4.2: Vídeos *Dynamic Background*.

Camera Jitter

En esta categoría encontramos cuatro vídeos cuya característica común es una vibración en los vídeos, posiblemente producida por una defectuosa fijación de la cámara. Se han utilizado tres de los cuatro vídeos. En los dos vídeos utilizados encontramos, por un lado personas de un tamaño mediano para *badminton* con movimientos bastante rápidos, y por otro, personas de tamaño pequeño en el vídeo *sidewalk*. En la Figura 4.3 se puede observar la "portada" de cada uno de los vídeos.

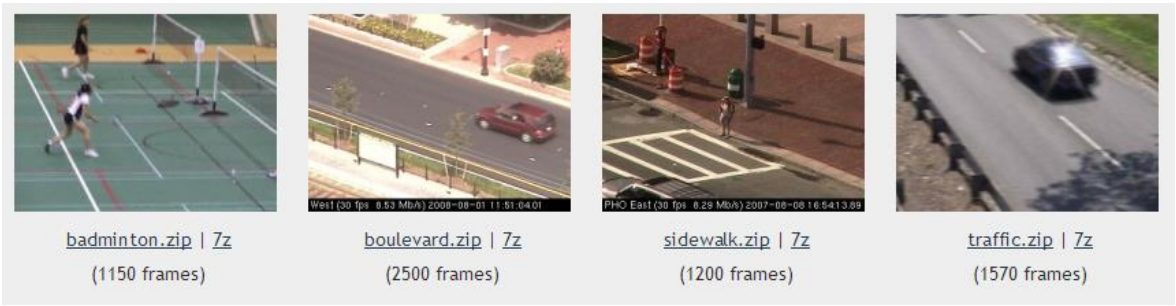


Figura 4.3: Vídeos *Camera Jitter*.

Intermittent Object Motion

Esta categoría contiene seis vídeos en los que aparecen y desaparecen, de forma continua, personas entrando y saliendo, han sido utilizados cuatro de los seis vídeos. De estos cuatro encontramos dos con personas de tamaño mediano (*sofa* y *winterdriveway*) y otros dos con tamaño pequeño (*abandonedBox* y *tramstop*). Los puntos de vista de la cámara también varían, así tanto *abandonedBox* como *winterdriveway* tienen una posición más cenital y los otros dos una perspectiva más horizontal. En la Figura 4.4 se puede observar la "portada" de cada uno de los vídeos.

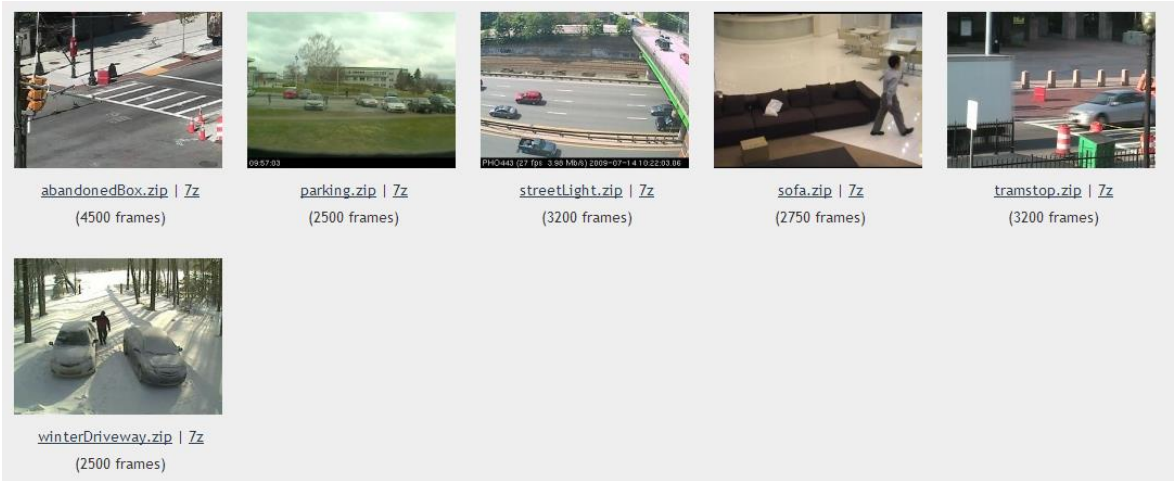


Figura 4.4: Vídeos *Intermittent Object Motion*.

Shadows

En esta categoría encontramos seis vídeos en el que aparecen sombras y reflejos de las personas que aparecen en las imágenes. En esta ocasión hemos utilizado cinco de los seis vídeos. En ellos encontramos personas de tamaño mediano, en la mayoría con un punto de vista horizontal. En la Figura 4.5 se puede observar la "portada" de cada uno de los vídeos.

Figura 4.5: Vídeos *Shadows*.

Thermal

En esta categoría encontramos cuatro vídeos tomados por cámaras térmicas, en este caso no han sido analizados por encontrarse fuera del alcance de este PFC. En la Figura 4.6 se puede observar la "portada" de cada uno de los vídeos.

Figura 4.6: Vídeos *Thermal*.

A partir de aquí se procedió a seleccionar todos aquellos vídeos que contuvieran personas con el suficiente tamaño como para poder ser analizados, puede verse una muestra de *frame* de cada uno de los vídeos seleccionados en el anexo B. Se desechó así mismo los de visión térmica por escapar al objetivo de este estudio. De esta manera, los vídeos que han sido utilizados para este PFC pueden verse a continuación en la Tabla 4.1.

Vídeo	**Categoría**	**Número de *Frames***
abandonedBox	*Intermittent Object Motion*	4500
backdoor	*Shadow*	2000
badminton	*Camera Jitter*	1150
busStation	*Shadow*	1250
copyMachine	*Shadow*	3400
cubicle	*Shadow*	7400
fall	*Dynamic Background*	4000
office	*Baseline*	2050
overpass	*Dynamic Background*	3000
pedestrians	*Baseline*	1099
peopleInShade	*Shadow*	1199
PETS2006	*Baseline*	1200
sidewalk	*Camera Jitter*	1200
sofa	*Intermittent Object Motion*	2750
tramstop	*Intermittent Object Motion*	3200
winterdriveway	*Intermittent Object Motion*	2500
Total		41898

Tabla 4.1: Vídeos utilizados y número de *frames*.

Según la descripción y la clasificación propuesta en el estado del arte (ver sección 2.3), en la Tabla 4.1 se puede identificar la problemática de la primera fase crítica en la detección de personas, la extracción de las regiones de interés, identificada como categoría del vídeo. Cada una de las diferentes categorías de los vídeos identifica una problemática diferente a la hora de realizar esta tarea, de hecho este *dataset* ha sido elaborado centrándose en esta tarea y por eso ofrece esa gran diversidad de escenarios.

Esta tarea es especialmente crítica en los algoritmos que utilizan la segmentación en esta primera etapa por contener objetos con movimientos intermitentes, "movimiento" de fondo (posiblemente debido a una mala fijación de la cámara), aparición de sombras, etc. En los algoritmos basados en búsqueda exhaustiva esta dificultad no afecta tan directamente como a los otros, si bien es cierto que a mayor complejidad del fondo (texturado, variabilidad, etc.) peores son los resultados de los algoritmos basados en búsqueda exhaustiva también.

En la siguiente etapa crítica, modelo de persona, lo que afectará de forma más directa será tanto el tamaño de la persona como el punto de vista donde se encuentra situada la cámara.

4.2 Etiquetado de vídeos y generación del Ground Truth

Una vez que tenemos nuestro *dataset* debemos generar nuestro *Ground Truth* para poder fijar la base de medición para todos los algoritmos, ya que sin ella no tendríamos con qué comparar. Para esta labor hemos realizado un etiquetado manual *frame* a *frame* con el programa *Video Image Annotation Tool* (via – http://sourceforge.net/projects/via-tool/) versión 1.0 en entorno Windows. En la Figura 4.7 se observa la interface de usuario del programa.

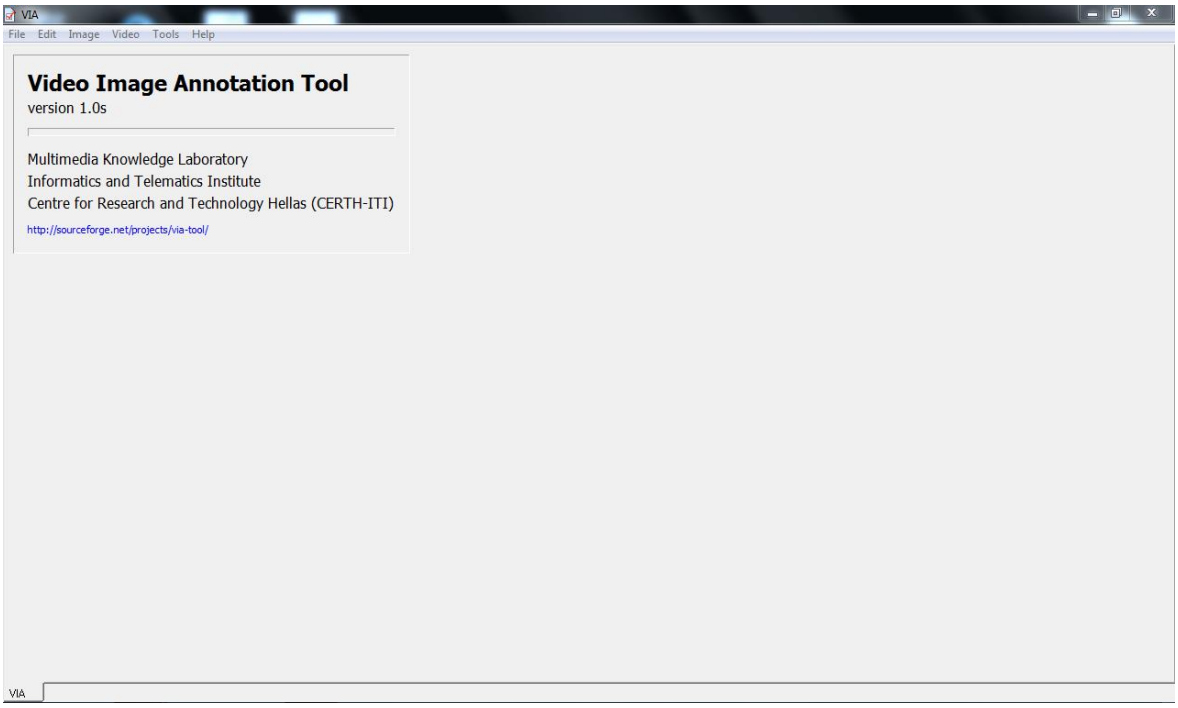


Figura 4.7: Interface del programa utilizado para el etiquetado manual de las personas en los vídeos que componen el *dataset*.

Para realizar la labor de etiquetado se deben seguir los siguientes pasos:

1. Iniciar el programa Video Image Annotation Tool.
2. File → New Video Project.
3. *Clickar* sobre *Record Mode*.
4. Comenzar a etiquetar los vídeos pulsando sobre el icono de + para añadir un nuevo marcador (Figura 4.8) y dimensionando el mismo para englobar a la persona. Repetir esta acción *frame* a *frame* (*Frame FWD*) y sobre todos los "blobs" existentes.
5. Una vez terminada la anotación salvaremos el proyecto, en nuestro caso el fichero de salida que utilizaremos como entrada para la evaluación de los algoritmos será con formato .txt, en la Figura 4.9 se muestra un ejemplo de salida. Para cada persona etiquetada nos dará sus coordenadas en la imagen, su anchura y altura y el *frame* donde ha sido etiquetado, otra información que nos ofrece pero que en este PFC no utilizamos, al no realizar *tracking* de los blobs, son el *frame* de comienzo y el *frame* de final de la detección.

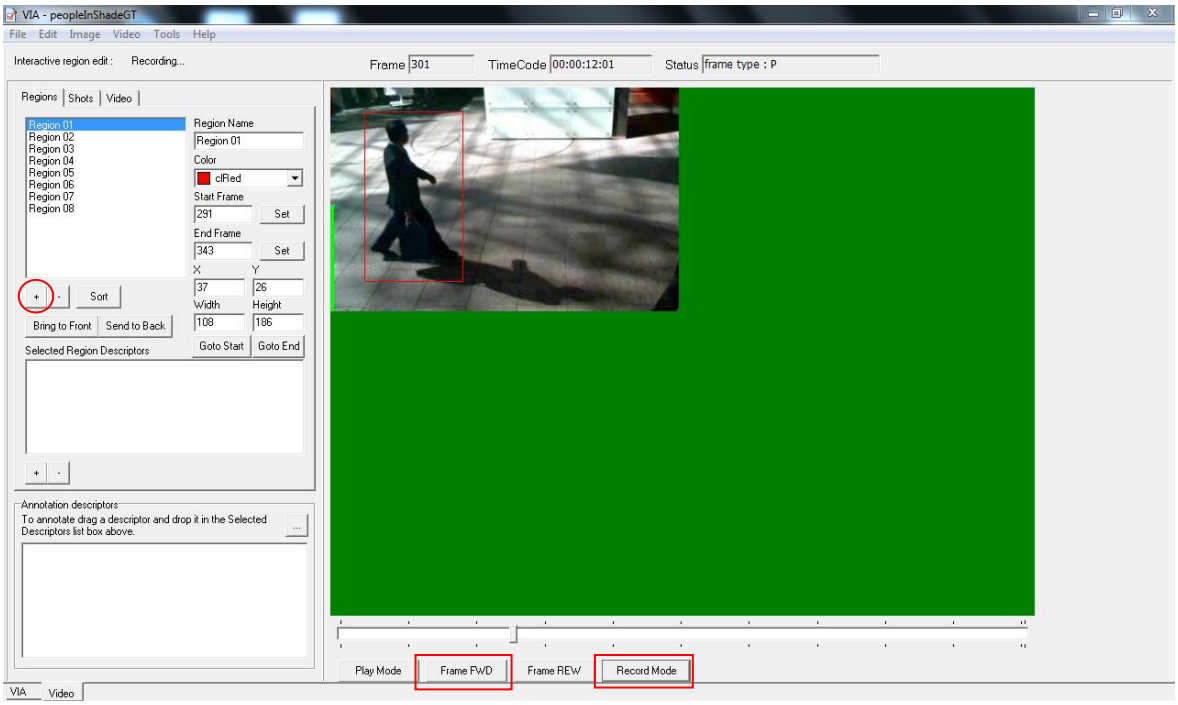


Figura 4.8: Programa *Video Image Anottation Tool* en funcionamiento.

```
[Project]
Name=officeGT
Type=0
[Video]
Title=
Date=06/05/2014
Description=
Annotator=
VideoFilename=C:\Users\AdminNuevo\Desktop\Evaluar Algoritmos
\Videos\office.avi.MPG
VideoDescFilename=
MaxFrame=2047
VideoDescriptorsNumber=0
[Regions]
Count=1
[Region#0]
Description=Region 01
StartFrame=597
EndFrame=2026
Color=clYellow
RegionDescriptorsNumber=0
RegionFrameDataNumber=93
[Region#0.1]
FrameDataFrameStart=597
FrameDataFrameEnd=605
FrameDataFrameX=72
FrameDataFrameY=50
FrameDataFrameW=66
FrameDataFrameH=137
[Region#0.2]
FrameDataFrameStart=606
```

Figura 4.9: Ejemplo de salida del programa de etiquetado.

4.3 Concepto de persona

Para poder evaluar correctamente los algoritmos debemos tener un baremo o medida que mida los resultados de las detecciones. Para ello, hemos creado nuestra "verdad" etiquetando a mano el *dataset*, creando así nuestro *Ground-Truth*. Al comparar los resultados de las detecciones de los algoritmos con nuestro etiquetado manual evaluaremos los resultados de los algoritmos. Junto con la elección de un *dataset* apropiado y diverso, describir qué consideramos persona es el punto de inflexión para lograr un estándar de puntuación de la eficacia de los algoritmos.

Esto nos plantea el problema de qué consideramos persona a la hora de etiquetar. Los distintos vídeos del *dataset* recogen diversos entornos, con escenarios interiores y exteriores, con cámaras cercanas y lejanas y con personas que cambian de tamaño, entran y salen de escena o sufren oclusiones. Por ello, hemos fijado el criterio de persona, además de cuando aparezca completa, cuando en la imagen se vea la cabeza y parte del cuerpo (más de la mitad del mismo) o todo el cuerpo sin la cabeza. No consideraremos persona cuando aparezca solamente la cabeza o cuando aparezca solamente la mitad del cuerpo. Como ejemplos se puede observar la siguiente Figura 4.10 en la que se muestra este criterio en la práctica. Sin señalar los ejemplos de personas (parte superior), tachado con una línea roja las consideradas como no personas (parte inferior).

Figura 4.10: Consideraciones de persona / no persona en el etiquetado de vídeos.

4.4 Métrica de evaluación

Para medir los resultados y poder compararlos hemos decidido utilizar una métrica de evaluación basada en los resultados globales por vídeo y no estudiar cada *frame* por separado. Para ello representaremos en gráficas 1-*Precision/Recall*. En una primera etapa se normalizan los valores obtenidos de la ejecución de los algoritmos para tener magnitudes comparables. La manera de realizar los cálculos de ambas puede verse a continuación en la Ecuación 1.

$$\text{Precision} = \frac{\#\ \text{TruePositivePeopleDetections}}{\#\ \text{TruePositivePeopleDetections} + \#\ \text{FalsePositivePeopleDetections}}$$

$$\text{Recall} = \frac{\#\ \text{TruePositivePeopleDetections}}{\#\ \text{TruePositivePeopleDetections} + \#\ \text{FalseNegativePeopleDetections}}$$

Ecuación 1: Ecuaciones para el cálculo de Precision y Recall a la hora de evaluar los algoritmos.

Esta forma de evaluación es más fidedigna que considerar solamente las detecciones positivas (TruePositiveDetections), puesto que tiene en cuenta los falsos positivos y negativos para evaluar la corrección del algoritmo. Para lograr unos resultados fiables en el cálculo de la curva 1-P/R hemos decidido que cada evaluación tenga 100 puntos.

Siguiendo los criterios de evaluación establecidos en [6] tenemos en cuenta no sólo la clasificación de persona o no, sino también la localización de la detección, el solape y la distancia relativa entre detecciones.

Finalmente calcularemos el área encerrada bajo la gráfica 1-Precision/Recall, para poder tener una medida de cuán efectivos son cada una de las configuraciones de los algoritmos y con esta medida podremos sacar las conclusiones.

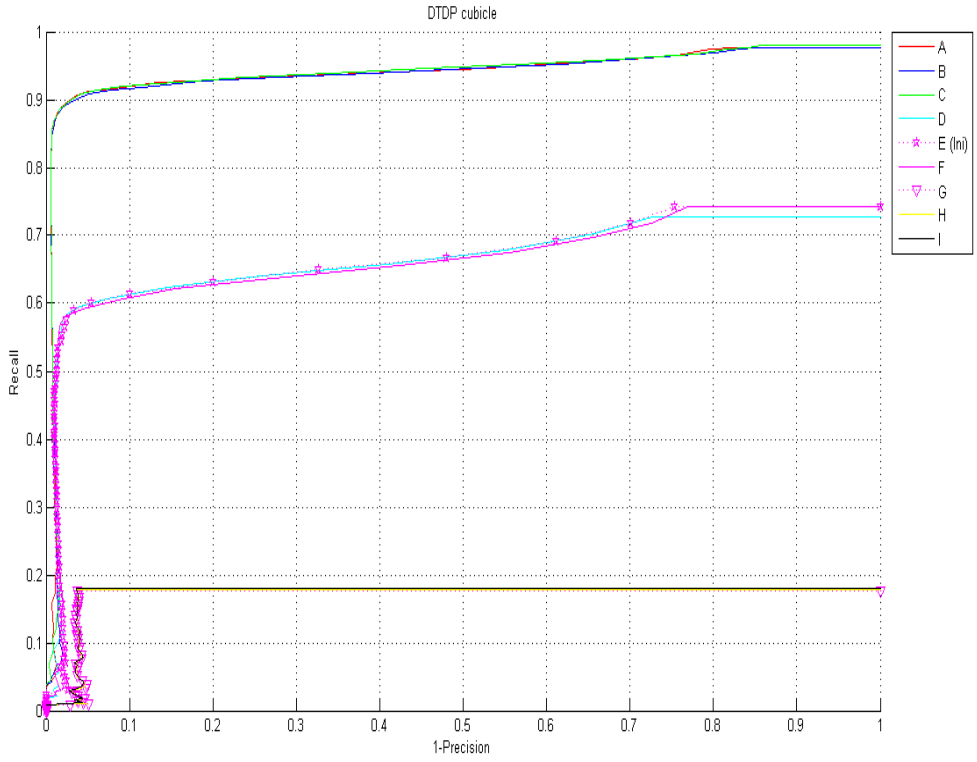


Figura 4.11: Gráfica 1-Precision/Recall que muestra el funcionamiento del algoritmo DTDP para todas sus posibles combinaciones.

4.5 Resultados

4.5.1 Introducción

Para evaluar los resultados hemos decidido dividirlo en cuatro escenarios:

1. El primero de ellos son los algoritmos por defecto para poder evaluar los algoritmos con sus configuraciones iniciales y compararlos entre ellos, de este resultado podemos observar cuál logra mejores resultados para cada vídeo y realizando una media de cuál sería más efectivo, por defecto, para la generalidad de los vídeos del *dataset* y, extrapolando este resultado, para su uso a nivel general. Para realizar una medición correcta aplicaremos el umbral escogido para el experimento en lugar del fijado por el autor.
2. A continuación mostraremos los resultados para todas las diferentes configuraciones elegidas de cada algoritmo y así ver cómo, en función de estos parámetros los resultados de los algoritmos varían y poder decidir qué configuración obtiene mejores resultados para cada vídeo y qué configuración ofrece los mejores resultados en media para el *dataset*.
3. En el siguiente paso comparamos la configuración paramétrica óptima para cada vídeo y cada algoritmo y evaluamos los resultados entre los diferentes algoritmos. En este caso lo que hacemos es fijarnos en los parámetros para cada vídeo de forma individual y no del *dataset* en su conjunto.
4. El último escenario planteado es escoger la configuración que mejores resultados obtiene de media en todo el *dataset* para cada uno de los algoritmos. Dicha media sería la mejor configuración de cada algoritmo para su uso general en cualquier escenario ofreciendo un mayor rendimiento en general.

Para poder realizar un análisis aclaratorio de los resultados en lo concerniente al tipo de vídeo existe un *frame* de muestra de cada uno de los vídeos analizados en el anexo B.

4.5.2 Algoritmos “por defecto”

En esta sección mostramos los resultados correspondientes al primer escenario de evaluación propuesto, los algoritmos con los parámetros por defecto del autor. La Tabla 4.2 muestra el rendimiento de los diferentes algoritmos con las configuraciones propuestas por el autor para cada uno de los vídeos del *dataset*. En el margen derecho de la tabla observamos la media del rendimiento de los algoritmos para cada uno de los vídeos y en el margen inferior se recoge la media de comportamiento de cada algoritmo para el *dataset* en general.

Observando los resultados podemos comprobar como algunos de los vídeos obtienen un rendimiento muy bajo independientemente del algoritmo que los evalúe. Por el contrario, en otros se observa el comportamiento contrario, un buen rendimiento general. Atendiendo a las fases críticas de todo algoritmo de detección de personas descritas en el estado del arte (ver sección 2.3) y como se ha descrito en la sección dedicada al *dataset* (ver sección 4.1), la primera fase crítica, detección de regiones de interés depende directamente de la categoría de vídeo, tanto si el algoritmo se basa en segmentación (Fusion y Edge), que se ven mucho más afectados por esta fase, como si se basa en búsqueda exhaustiva (el resto de los algoritmos aquí estudiados). De esto podemos observar que aquellos videos con una extracción de ROI’s más complicada, como los que tienen “movimiento” en la captura, obtienen de media notas más bajas, por ejemplo *fall*.

Además, la segunda fase crítica, basada en el modelo de persona utilizado, nos da la otra clave a la hora de evaluar los resultados. En esta ocasión, en lugar de fijarnos en la “calidad” de la imagen como elemento determinante, nos influirán el tamaño de la persona y el enfoque de la cámara. De esta forma, aquellos algoritmos que utilicen tamaños base de

persona más pequeños funcionarán mejor en aquellos vídeos que contengan personas más pequeñas, hecho que perjudicará a los que tengan un tamaño base de persona demasiado grande y no permita estas detecciones. El otro factor determinante en el modelo de persona es el enfoque de la cámara, si la cámara tiene un ángulo horizontal, los modelos de personas holísticos obtendrán buenos resultados, incluso mejores que aquellos basados en partes. Sin embargo, cuando se eleva el ángulo de la cámara estas detecciones se ven perjudicadas y se observa una clara mejoría en los algoritmos que utilizan los modelos por partes, por ejemplo *copyMachine*.

VÍDEO	HOG	ISM 1	ISM 2	ISM 3	DTDP	FUSION	EDGE	ACF CALTECH	ACF INRIA	MEDIA
abandonedBox	0,0001	0,0593	0,0484	0,0043	0,0003	0,1845	0,2109	**0,3238**	0,0470	0,0976
backdoor	0,8371	0,5505	0,6309	0,4562	0,9152	0,6091	0,7028	0,9128	**0,9212**	**0,7262**
badminton	0,4712	0,0712	0,2723	0,0772	**0,7211**	0,0804	0,3076	0,5234	0,6537	0,3531
busStation	0,6955	0,2191	0,3106	0,0759	0,7897	0,1426	0,5809	0,7993	**0,8555**	0,4966
copyMachine	0,1208	0,0023	0,0153	0,0118	0,4171	0,1274	0,4487	**0,4554**	0,1956	0,1994
cubicle	0,4663	0,1052	0,0698	0,0568	0,6687	0,1699	0,4132	**0,8516**	0,6893	0,3879
fall	0,0996	0,0077	0,0072	0,0014	0,1342	0,0050	0,0545	**0,5947**	0,3425	0,1385
office	0,8927	0,1294	0,4296	0,0140	0,9668	0,3767	0,8454	0,8635	**0,9930**	0,6123
overpass	0,4464	0,2430	0,3702	0,2616	**0,6880**	0,0365	0,0613	0,2488	0,4580	0,3126
pedestrians	0,6319	0,6032	0,5874	0,3075	0,6645	**0,9271**	0,8966	0,9248	0,7732	0,7018
peopleInShade	0,2120	0,2082	0,2910	0,2106	0,8445	0,3988	0,6171	0,5715	**0,9102**	0,4738
PETS2006	0,3848	0,3284	0,4688	0,4009	0,5537	**0,5346**	0,5265	0,3588	0,5256	0,4536
sidewalk	0,0002	0,0116	0,0042	0,0000	0,1105	0,0770	0,0715	**0,8836**	0,0272	0,1318
sofa	0,4748	0,1473	0,1823	0,0807	0,9032	0,4671	0,7438	**0,7486**	0,7215	0,4966
tramstop	0,0723	0,0094	0,0190	0,0003	0,0065	0,2127	0,1430	**0,5863**	0,0874	0,1263
winterdriveway	0,1062	0,0710	0,0568	0,0000	0,1156	0,0222	0,3369	**0,3433**	0,0859	0,1264
MEDIA	0,3695	0,1729	0,2352	0,1224	0,5312	0,2732	0,4350	**0,6244**	0,5179	

Tabla 4.2: Resultado de los algoritmos configurados por defecto.

Cada uno de los algoritmos ha sido parametrizado por su autor con arreglo a un conjunto de vídeos seleccionados por el propio autor. Sin embargo, cuando enfrentamos estos algoritmos con sus configuraciones originales a un entorno generalista se obtienen resultados diferentes a los ofrecidos por el autor, que además puede haber utilizado una métrica diferente a la propuesta en este PFC. Si nos fijamos ahora en la última columna de la derecha podemos observar la sensibilidad que todos los algoritmos tienen al tipo de vídeo. La calidad y el ruido de la imagen o el punto de vista o la distancia de enfoque de la misma son determinantes para los resultados de la detección de todos los algoritmos.

Se observa que, si el vídeo tiene personas muy pequeñas (por ejemplo los vídeos *abandonedBox* o *fall*), es uno de los parámetros que hemos variado en las distintas configuraciones, o el ángulo de visión es muy cenital (por ejemplo video *copyMachine* o *sidewalk*) la detección empeora considerablemente.

Todo esto reafirma el objetivo y alcance de este PFC tratando de ofrecer un entorno de pruebas “neutral” que nos ayude a evaluar los diferentes algoritmos para su uso directo en la vida real. En la Figura 4.12 se puede observar el comportamiento de los algoritmos por defecto para cada uno de los vídeos.

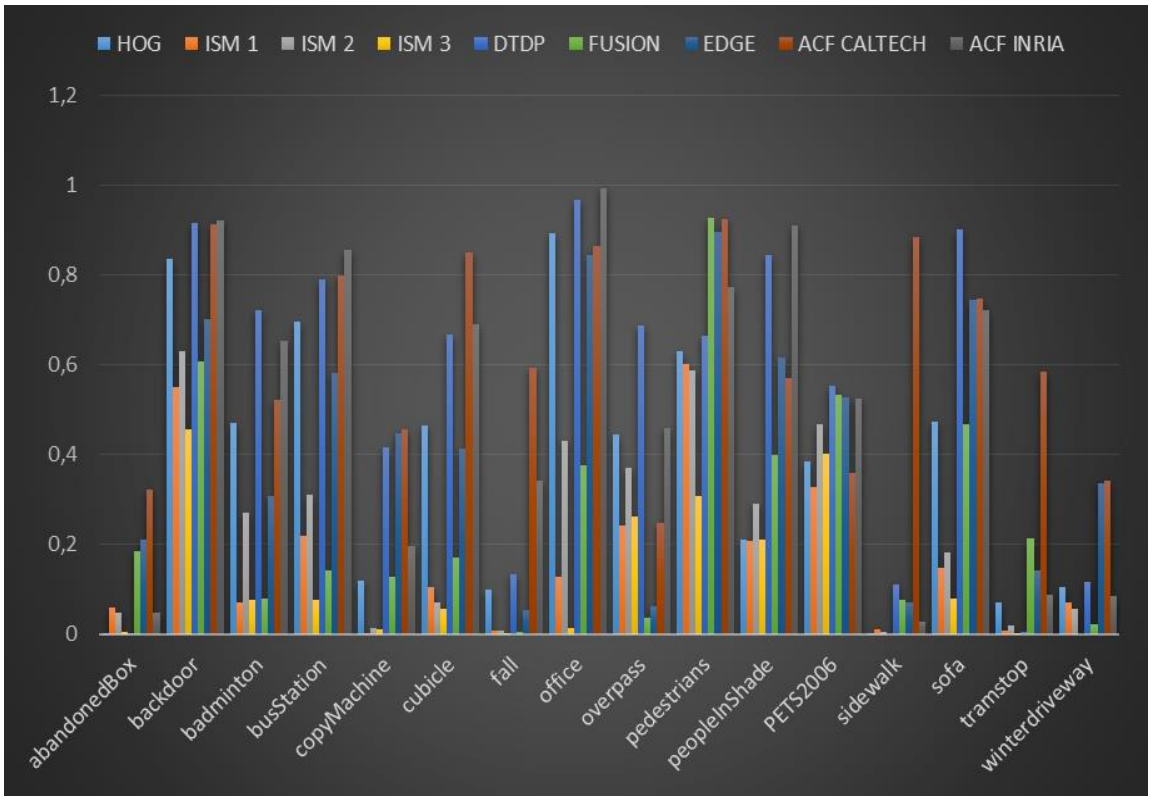


Figura 4.12: Gráfico que muestra el rendimiento de los algoritmos por defecto.

Como ya hemos comentado anteriormente y puede observarse en la Figura 4.12 observamos un comportamiento bastante común en cuanto a los bajos resultados obtenidos por todos los algoritmos en los vídeos *abandonedBox* y *winterdriveway*. Como ya comentamos anteriormente el tamaño de la persona o la calidad de la imagen así como la posición de la cámara nos dan pistas directas sobre el resultado que en condiciones iniciales obtendrán los algoritmos. Además, en el caso del vídeo *winterdriveway* nos encontramos con que el vídeo tiene una sola persona por lo que los resultados no son demasiado extrapolables a la hora del análisis.

En el extremo opuesto encontraríamos los vídeos *backdoor*, *office* y *pedestrians* que como características comunes son vídeos más nítidos y con un tamaño de persona bastante grande y con un enfoque horizontal.

A la vista de estos resultados observamos que, por defecto, los algoritmos se comportan mejor ante vídeos que cuenten con una buena resolución que facilite la detección de los ROI con la técnica utilizada por el algoritmo y tengan un enfoquen en horizontal (ayuda al modelo de persona, especialmente en los modelos holísticos).

De media, el mejor algoritmo en su configuración por defecto, es el ACF con el modelo Caltech.

4.5.3 Parametrización de algoritmos

Comenzamos ahora a exponer los resultados para las diferentes configuraciones que se han elegido para cada uno de los algoritmos. En esta sección analizaremos el rendimiento de cada algoritmo con su parametrización tanto a nivel general del *dataset* como para cada vídeo en particular. De aquí saldrán los parámetros que ofrecen un mejor rendimiento para el algoritmo y que nos brindará la configuración óptima por vídeo como de mejor media para el *dataset* en general.

HOG

HOG				
Parámetro	A (2/4)	B (3/4)	C (Ini)	D (5/4)
Umbral	0	0	0	0
Scaleratio	1,025	1,0375	1,05	1,0625

Tabla 4.3: Configuraciones HOG propuestas.

HOG					
VÍDEO	A	B	C (Ini)	D	MEDIA
abandonedBox	0,0001	0,0002	0,0001	0,0001	0,0002
backdoor	**0,8472**	0,8355	0,8371	0,8348	0,8386
badminton	**0,4843**	0,4785	0,4712	0,4659	0,4750
busStation	0,7052	**0,7063**	0,6955	0,6942	0,7003
copyMachine	0,1203	0,0909	**0,1208**	0,1058	0,1094
cubicle	**0,4908**	0,4788	0,4663	0,4786	0,4786
fall	**0,1115**	0,1004	0,0996	0,0958	0,1018
office	**0,9334**	0,9005	0,8927	0,7415	0,8670
overpass	0,4209	0,4333	**0,4464**	0,3639	0,4161
pedestrians	**0,6598**	0,6549	0,6319	0,6094	0,6390
peopleInShade	**0,2449**	0,2430	0,2120	0,2071	0,2267
PETS2006	**0,4151**	0,4089	0,3848	0,3938	0,4006
sidewalk	**0,0002**	**0,0002**	**0,0002**	**0,0002**	0,0002
sofa	0,5281	**0,5433**	0,4748	0,4361	0,4956
tramstop	**0,0852**	0,0684	0,0723	0,0723	0,0746
winterdriveway	**0,0980**	0,0550	0,1062	0,0925	0,0879
MEDIA	**0,3840**	0,3749	0,3695	0,3495	

Tabla 4.4: Resultados configuraciones HOG.

En la Tabla 4.3 se pueden ver los valores de los parámetros elegidos para el algoritmo HOG, por su parte la Tabla 4.4 muestra el rendimiento del algoritmo para las diferentes configuraciones. Como en el caso de los algoritmos por defecto se pueden analizar estos resultados separándolo en las dos fases críticas de todo detector de personas.

Para comenzar se observa que aquellos vídeos que tienen una primera etapa de detección de ROI's más compleja por sus características (ver sección 4.1) vídeos como *sidewalk* o como *abandonedBox* que tienen una calidad de imagen muy mala obtienen los peores resultados.

Atendiendo al modelo de persona utilizado se observa que este algoritmo obtiene muy malos resultados cuando la cámara no se coloca en horizontal sino con un ángulo elevado, el claro ejemplo de esto es el vídeo *copyMachine*.

Además, si nos fijamos en sus parámetros se puede observar que el tamaño base de persona en este algoritmo es un factor muy discriminatorio puesto que no detectará ninguna persona por debajo de un tamaño de 64X128, esto explica los malos resultados obtenidos en aquellos vídeos con tamaño de persona pequeño como es el caso de *fall*.

Sin considerar los vídeos con personas "pequeñas" y con el conocimiento de las limitaciones por sus dos fases críticas, a la luz de los datos se puede observar que a menor

valor de *scaleratio*, mayor número de escalas de detección, logramos una detección más fina que ofrece un mayor rendimiento. De ello observamos que, para la mayoría de los vídeos la mejor configuración en media para el *dataset* es la configuración A. En la Figura 4.13 puede observarse la comparación de todas las configuraciones del algoritmo HOG para todos los vídeos del dataset.

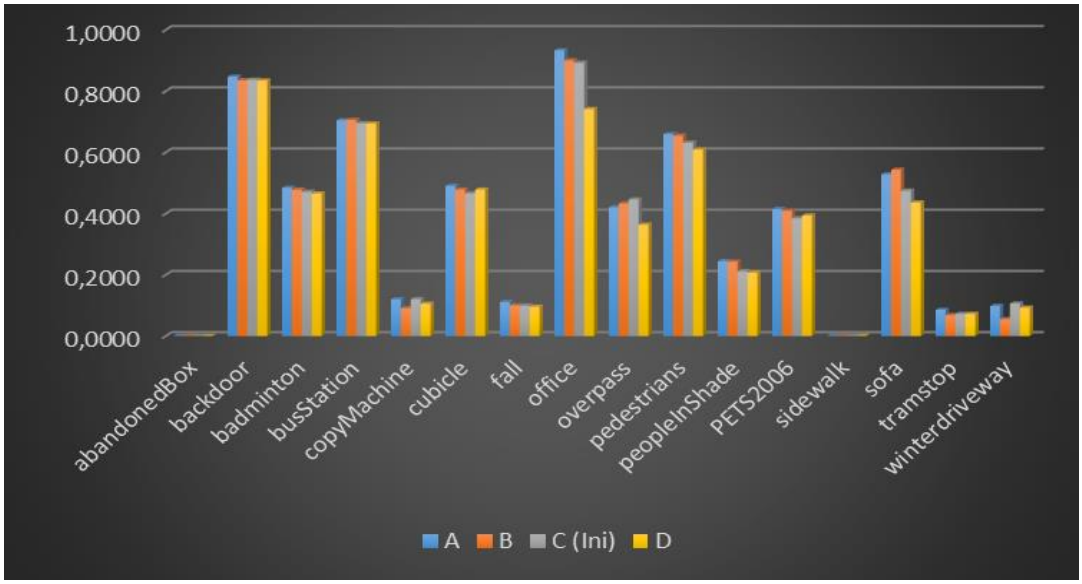


Figura 4.13: Gráfica de los resultados de las configuraciones del algoritmo HOG por cada vídeo.

ISM

En la Tabla 4.5 pueden verse los valores de los parámetros escogidos para el algoritmo ISM. Como ya hemos indicado anteriormente, tanto para los parámetros iniciales como para el algoritmo HOG, el rendimiento del algoritmo que se refleja en la Tabla 4.6, depende en primer lugar de la fase de extracción de ROI's y a continuación del modelo de persona. Como ya se vio en el algoritmo HOG, aquellos vídeos que tienen mayores dificultades en esta primera etapa, como puede ser el caso de *abandonedBox* o *sidewalk*, obtienen muy malos resultados. Por el contrario, vídeos con buena definición y mayor sencillez de extracción de ROI's como es el caso de *backdoor* obtienen unos resultados mucho mejores.

En la fase de modelo de persona, al encontrarnos con un modelo holístico, seguimos encontrando los mismos problemas que con el HOG. El cambio del punto de vista de la cámara ocasiona un mal rendimiento como es el caso de *winterdriveway*.

El parámetro que hemos modificado en este algoritmo *Max Scale*, nos daría mejoras de rendimiento en el caso de que contáramos con vídeos con tamaños de persona grande, debido a que este parámetro afecta directamente a estos valores, pero para personas pequeñas el autor fija el factor de escala inicial con el valor 0,3. De esta forma se observa que la mejor configuración de este algoritmo se obtiene cuanto mayor tamaño de persona permitimos puesto que no dejaremos sin detectar candidatos a persona discriminando por grandes tamaños y el tamaño base no se decrementa. En la Figura 4.14 puede observarse el rendimiento de este algoritmo para los diferentes vídeos y configuraciones.

ISM				
Parámetro	det 3	A (Ini)	B	C
Umbral	300	300	300	300
Max Scale	1,5	1,5	2	2,5

Tabla 4.5: Configuraciones ISM propuestas.

ISM					
VÍDEO	det 3	A	B	C	MEDIA
abandonedBox	0,0103	**0,0103**	0,0100	0,0093	0,0099
backdoor	0,6463	0,6463	0,6578	**0,6618**	0,6531
badminton	0,2192	**0,2192**	0,2106	0,1961	0,2112
busStation	0,4447	0,4447	0,4498	**0,4505**	0,4474
copyMachine	0,2331	0,2331	0,2757	**0,2903**	0,2580
cubicle	0,1267	0,1267	**0,1342**	0,1337	0,1303
fall	0,0033	**0,0033**	0,0020	0,0020	0,0026
office	0,2807	0,2807	0,2895	**0,3035**	0,2886
overpass	0,4528	0,4528	0,4470	**0,4538**	0,4516
pedestrians	0,4252	0,4252	**0,4339**	0,4281	0,4281
peopleInShade	0,3146	0,3146	0,3325	**0,3354**	0,3243
PETS2006	0,5903	0,5903	0,5918	**0,5931**	0,5914
sidewalk	0,0000	0,0000	0,0000	0,0000	0,0000
sofa	0,1085	0,1085	0,1114	**0,1129**	0,1103
tramstop	0,0055	**0,0055**	0,0053	0,0051	0,0054
winterdriveway	0,0003	0,0003	0,0007	**0,0009**	0,0005
MEDIA	0,2413	0,2413	0,2470	**0,2485**	

Tabla 4.6: Resultados configuraciones ISM.

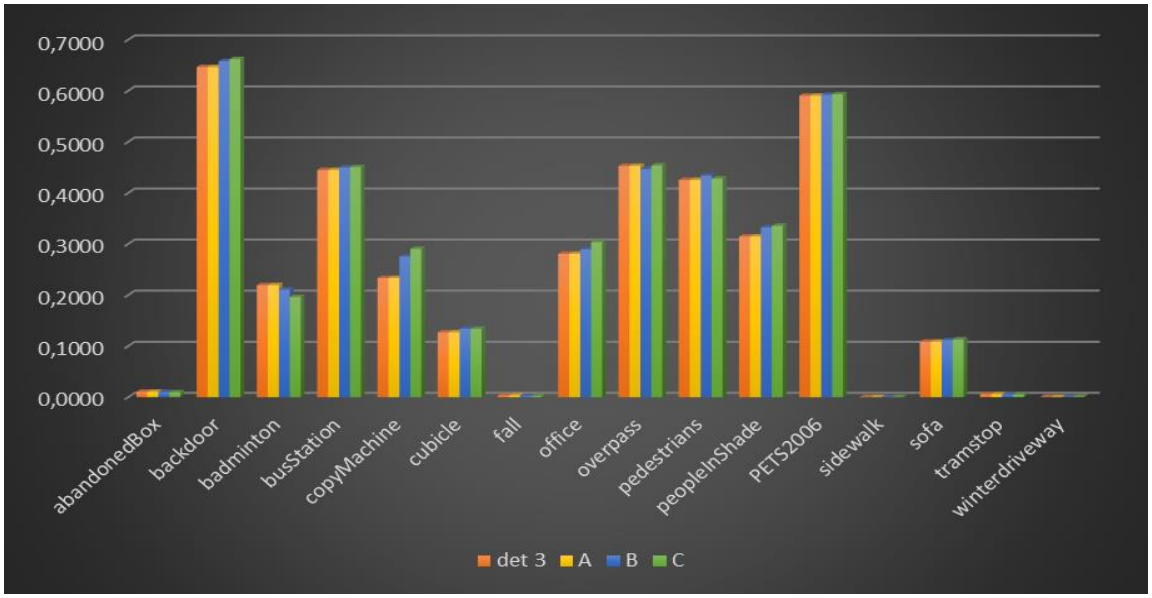


Figura 4.14: Gráfica de los resultados de las configuraciones del algoritmo ISM por cada vídeo.

DTDP

En la Tabla 4.7 se pueden observar los parámetros y valores escogidos para el algoritmo DTDP.

DTDP									
Parámetro	A	B	C	D	E (Ini)	F	G	H	I
Umbral	-1,5	-1,5	-1,5	-1,5	-1,5	-1,5	-1,5	-1,5	-1,5
sbin	4	4	4	8	8	8	16	16	16
interval	5	10	15	5	10	15	5	10	15

Tabla 4.7: Configuraciones DTDP propuestas.

En el caso del algoritmo DTDP observamos que los resultados mejoran considerablemente con respecto a sus predecesores. Manteniendo la dificultad ya comentada ante la dificultad de extracción de la ROI's, basándose también en búsqueda exhaustiva, este algoritmo obtiene mejores resultados que sus predecesores. Esta mejoría, por consiguiente, es achacada a la segunda fase crítica, el modelo de persona. En este sentido, al tratarse de un algoritmo que utiliza un modelo basado en partes se observa una menor dependencia, aunque continúe existiendo, en cuanto al punto de vista de la posición de la cámara. Seguimos con un mal rendimiento en vídeos como *abandonedBox* o *fall*, obteniendo mejores resultados en *office* o *peopleInShade*.

Además, a través de la parametrización logramos modificar el tamaño mínimo de persona que será capaz de detectar, a través del valor, así como el número de inter-escalas con el valor *interval*, que nos dará la nitidez entre tamaños de persona.

Si nos fijamos en la Tabla 4.8 observamos, en media, escalas de mejoría para un mismo valor de *sbin* y un incremento del valor de *interval*. Esto quiere decir, para un valor de *sbin* cuanto mayor sea el valor de *interval* mejor rendimiento tendrá el algoritmo.

Analizando los diferentes valores de *sbin*, como cabía esperar, a menor tamaño de *sbin*, menor tamaño base de persona empezamos a admitir por lo que detectaremos personas más pequeñas y por consiguiente se mejorará el rendimiento.

Fijándonos en conjunto para ambos valores la configuración que desprende mejor rendimiento es la equivalente a C, un valor *sbin* = 4 con *interval* = 15.

En la Figura 4.15 se puede observar el funcionamiento del algoritmo para las diferentes configuraciones y vídeos del *dataset*.

DTDP

VÍDEO	A	B	C	D	E (Ini)	F	G	H	I	MEDIA
abandonedBox	0,3576	**0,3715**	0,3610	0,0159	0,0003	0,0274	0,0000	0,0000	0,0000	0,1134
backdoor	0,8858	0,8846	0,8920	0,9149	**0,9152**	**0,9152**	0,0326	0,0326	0,0326	0,5505
badminton	0,6397	0,6430	0,6527	0,7141	0,7211	**0,7296**	0,4482	0,4643	0,4748	0,5488
busStation	0,8408	**0,8440**	0,8418	0,7870	0,7897	0,7939	0,0007	0,0007	0,0007	0,4899
copyMachine	0,1549	0,1739	0,1736	0,4006	0,4171	0,4299	0,6235	0,6219	**0,6325**	0,3628
cubicle	0,9393	0,9372	**0,9404**	0,6660	0,6687	0,6654	0,1706	0,1709	0,1732	0,5332
fall	0,4202	0,4256	**0,4610**	0,1475	0,1342	0,1585	0,0001	0,0001	0,0000	0,1747
office	0,9655	0,9654	0,9611	**0,9707**	0,9668	0,9671	0,9505	0,9505	0,9520	**0,8650**
overpass	**0,7644**	0,7213	0,7237	0,7277	0,6880	0,6850	0,2907	0,2910	0,2909	0,5183
pedestrians	0,8903	0,8989	**0,9028**	0,6554	0,6645	0,6646	0,0000	0,0000	0,0000	0,4676
peopleInShade	0,7138	0,7226	0,7233	0,8361	0,8445	**0,8532**	0,4660	0,4726	0,4776	0,6110
PETS2006	0,6038	0,6147	**0,6255**	0,5461	0,5537	0,5590	0,2058	0,2114	0,2170	0,4137
sidewalk	0,7968	0,7998	**0,8090**	0,0758	0,1105	0,1029	0,0000	0,0000	0,0000	0,2695
sofa	0,8611	0,8752	0,8908	0,8873	0,9032	**0,9138**	0,0124	0,0129	0,0163	0,5373
tramstop	0,6587	0,6776	**0,6832**	0,0050	0,0065	0,0079	0,0000	0,0000	0,0000	0,2039
winterdriveway	0,7447	0,7516	**0,7616**	0,0931	0,1156	0,1230	0,0000	0,0000	0,0000	0,2590
MEDIA	0,7023	0,7067	**0,7127**	0,5277	0,5312	0,5373	0,2001	0,2018	0,2042	

Tabla 4.8: Resultados configuraciones DTDP.

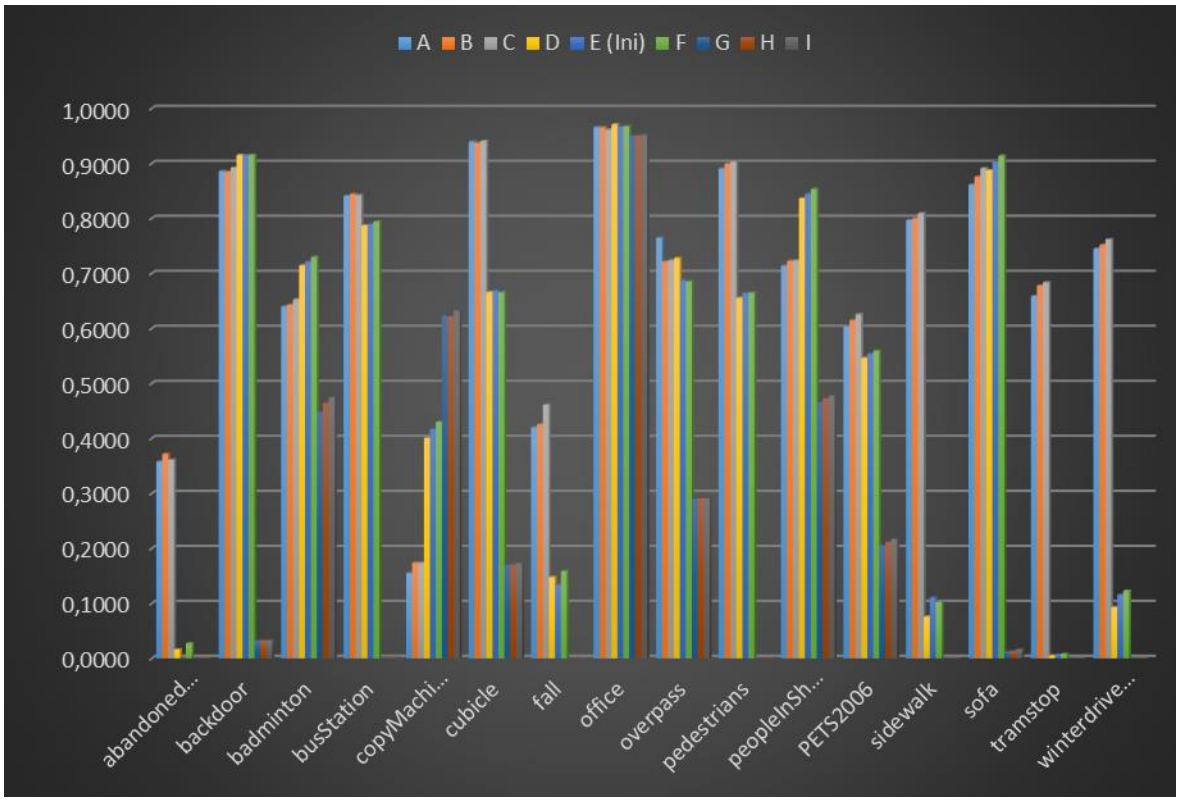


Figura 4.15: Gráfica de los resultados de las configuraciones del algoritmo DTDP por cada vídeo.

Fusion y Edge

En la Tabla 4.9 pueden observarse las diferentes configuraciones que compartirán tanto *Fusion* como *Edge*. En este caso se ha considerado el umbral directamente con valor cero puesto que el autor no seleccionaba ninguno por defecto y la configuración inicial no corresponde con ninguna de las parametrizadas por lo que se evalúa como una configuración más.

Fusion/Edge (umbral = 0)																				
Par	Ini	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
var	13	5	5	5	5	5	10	10	10	10	10	17	17	17	17	17	29	29	29	29	29
winQ	3	1	3	5	7	9	1	3	5	7	9	1	3	5	7	9	1	3	5	7	9

Tabla 4.9: Configuraciones Fusion y Edge propuestas.

En este caso nos encontramos con dos algoritmos desarrollados por el VPULab y cuya primera etapa crítica, en lugar de basarse en búsqueda exhaustiva, utiliza segmentación de fondo. Esta técnica hará que sea mucho más dependiente del tipo de vídeo y se vea afectado en mayor medida por la clasificación que se hizo al definir el *dataset*. De esta manera y como se observa en la Tabla 4.10 y en la Tabla 4.11 aquellos vídeos que presentan mayor dificultad en la segmentación (los que tienen movimiento de cámara como por ejemplo *badminton* o un fondo dinámico como *fall*) ofrecerán peores resultados frente a los que permiten una mejor etapa de segmentación.

En cuanto a la segunda fase crítica, modelo de persona, *Fusion* utiliza un modelo holístico frente al basado en partes del *Edge.* Esto aportará una flexibilidad en la tarea de detección al segundo sobre el primero logrando que tenga un mejor rendimiento, por ejemplo, cuando a cámara no se encuentra situada en la horizontal, como puede observarse claramente al consultar la Tabla 4.10 y la Tabla 4.11 para el vídeo *copyMachine*.

Sobre la parametrización se observa que en ambos casos a mayores tamaños de *varnoise* y *Win_Q* se mejoran los resultados. Esto es debido a que el sistema será más robusto frente al ruido y para la segmentación se utilizarán más píxeles vecinos. El ajuste del parámetro *varnoise* tiene que guardar un equilibrio lógico puesto que valores muy pequeños resultarían en gran cantidad de falsos positivos y valores muy grandes implicarían posible pérdida de información. En cuanto a la variable *Win_Q* nos servirá para realizar una segmentación más gruesa o fina en función del parámetro escogido.

En la Figura 4.16 y la Figura 4.17 pueden observarse los comportamiento para las diferentes configuraciones y para cada uno de los vídeos de ambos algoritmos.

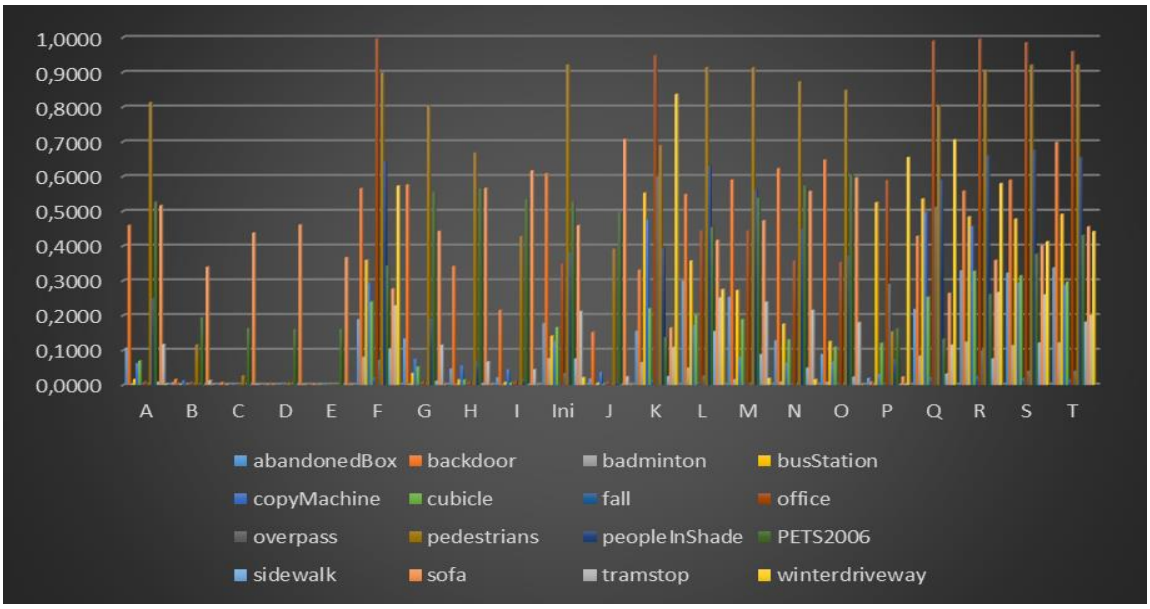


Figura 4.16: Gráfica de los resultados de las configuraciones del algoritmo Fusion por cada vídeo.

Como indicamos anteriormente los parámetros del Edge configurados son los mismos que los de Fusion. Ambos algoritmos comparten la primera parte de su proceso con la segmentación. Su diferencia se basa en el modelo de persona que utilizan para la detección, en uno es holístico (Fusion) y en el otro por partes.

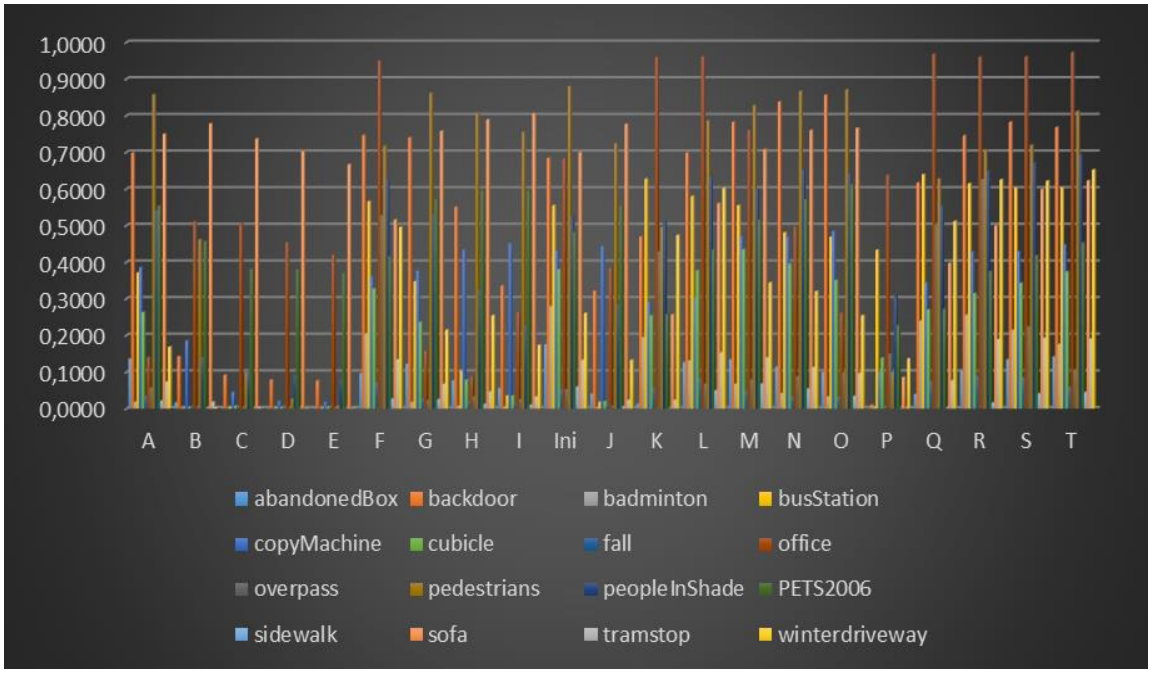


Figura 4.17: Gráfica de los resultados de las configuraciones del algoritmo Edge por cada vídeo.

Fusion

VÍDEO	A	B	C	D	E	F	G	H	I	Ini	J	K	L	M	N	O	P	Q	R	S	T	MEDIA
abandonedBox	0,1071	0,0076	0,0009	0,0006	0,0004	0,1891	0,1336	0,0468	0,0215	0,1777	0,0180	0,1556	0,3007	0,2530	0,1281	0,0885	0,0194	0,2184	0,3292	0,3225	**0,3380**	0,1360
backdoor	0,4597	0,0166	0,0075	0,0036	0,0041	0,5656	0,5763	0,3419	0,2156	0,6090	0,1524	0,3311	0,5488	0,5905	0,6233	0,6480	0,0086	0,4286	0,5583	0,5898	**0,6985**	0,3799
badminton	0,0030	0,0000	0,0000	0,0000	0,0000	0,0806	0,0034	0,0011	0,0007	0,0768	0,0004	0,0651	0,0495	0,0162	0,0090	0,0084	0,0005	0,0843	**0,1242**	0,1141	0,1217	0,0361
busStation	0,0167	0,0017	0,0008	0,0003	0,0001	0,3593	0,0339	0,0157	0,0082	0,1414	0,0062	**0,5528**	0,3573	0,2725	0,1757	0,1260	0,5253	0,5359	0,4843	0,4779	0,4918	0,2183
copyMachine	0,0623	0,0120	0,0005	0,0001	0,0001	0,2924	0,0754	0,0556	0,0454	0,1261	0,0366	0,4751	0,1724	0,0803	0,0641	0,0667	0,0311	**0,4969**	0,4566	0,2945	0,2876	0,1491
cubicle	0,0695	0,0000	0,0000	0,0000	0,0000	0,2397	0,0527	0,0149	0,0065	0,1661	0,0022	0,2204	0,2012	0,1893	0,1305	0,1094	0,1209	0,2532	**0,3275**	0,3147	0,2967	0,1293
fall	0,0036	0,0000	0,0000	0,0000	0,0000	0,0217	0,0013	0,0005	0,0005	0,0049	0,0005	0,0158	0,0099	0,0052	0,0033	0,0030	0,0011	0,0224	**0,0251**	0,0201	0,0151	0,0073
office	0,0097	0,0082	0,0015	0,0002	0,0000	**0,9968**	0,0097	0,0097	0,0112	0,3490	0,0112	0,9486	0,4448	0,4448	0,3581	0,3536	0,5891	0,9902	0,9955	0,9853	0,9598	0,4037
overpass	0,0003	0,0016	0,0069	0,0039	0,0015	0,0722	0,0000	0,0000	0,0000	0,0342	0,0001	**0,5996**	0,0270	0,0067	0,0004	0,0002	0,2906	0,5122	0,0996	0,0403	0,0399	0,0827
pedestrians	0,8133	0,1156	0,0265	0,0058	0,0012	0,8986	0,8040	0,6681	0,4286	**0,9217**	0,3916	0,6898	0,9139	0,9134	0,8730	0,8487	0,1539	0,8047	0,9064	0,9210	0,9212	0,6201
peopleInShade	0,2498	0,0000	0,0000	0,0000	0,0000	0,6443	0,1909	0,0676	0,0155	0,3827	0,0086	0,3957	0,6316	0,5609	0,4499	0,3723	0,0755	0,5895	0,6608	**0,6756**	0,6545	0,3155
PETS2006	0,5272	0,1943	0,1637	0,1606	0,1606	0,3436	0,5547	0,5659	0,5347	0,5277	0,4984	0,1372	0,4543	0,5391	0,5742	**0,6051**	0,1635	0,1326	0,2612	0,3772	0,4314	0,3765
sidewalk	0,0087	0,0006	0,0003	0,0002	0,0001	0,1052	0,0117	0,0049	0,0020	0,0761	0,0008	0,0254	0,1549	0,0885	0,0494	0,0234	0,0018	0,0322	0,0762	0,1216	**0,1815**	0,0460
sofa	0,5168	0,3396	0,4378	0,4611	0,3672	0,2768	0,4424	0,5661	0,6169	0,4592	**0,7071**	0,1646	0,4166	0,4732	0,5576	0,5965	0,0233	0,2639	0,3589	0,4011	0,4557	**0,4239**
tramstop	0,1169	0,0133	0,0025	0,0013	0,0010	0,2288	0,1151	0,0670	0,0449	0,2122	0,0239	0,1087	0,2515	0,2390	0,2153	0,1799	0,0026	0,1154	**0,2670**	0,2603	0,2016	0,1271
winterdriveway	0,0024	0,0000	0,0000	0,0000	0,0000	0,5732	0,0002	0,0000	0,0001	0,0216	0,0005	**0,8366**	0,2753	0,0193	0,0160	0,0005	0,6547	0,7058	0,5799	0,4127	0,4417	0,2162
MEDIA	0,1854	0,0445	0,0406	0,0399	0,0335	0,3680	0,1878	0,1516	0,1220	0,2679	0,1162	0,3576	0,3256	0,2932	0,2643	0,2519	0,1664	0,3866	0,4069	0,3955	**0,4085**	

Tabla 4.10: Resultados configuraciones *Fusion*.

Edge																						
VÍDEO	A	B	C	D	E	F	G	H	I	Ini	J	K	L	M	N	O	P	Q	R	S	T	MEDIA
abandonedBox	0,1363	0,0164	0,0020	0,0027	0,0023	0,0965	0,1222	0,0766	0,0550	**0,1752**	0,0406	0,0129	0,1262	0,1346	0,1146	0,0997	0,0000	0,0389	0,1041	0,1345	0,1417	0,0778
backdoor	0,6977	0,1421	0,0918	0,0786	0,0764	0,7463	0,7403	0,5504	0,3356	0,6837	0,3210	0,4692	0,6979	0,7821	0,8381	**0,8558**	0,0099	0,6163	0,7453	0,7824	0,7680	0,5252
badminton	0,0177	0,0007	0,0000	0,0000	0,0000	0,2051	0,0175	0,0069	0,0048	**0,2798**	0,0034	0,1949	0,1311	0,0677	0,0420	0,0329	0,0056	0,2398	0,2561	0,2146	0,1762	0,0903
busStation	0,3709	0,0011	0,0040	0,0019	0,0011	0,5657	0,3475	0,1033	0,0355	0,5547	0,0185	0,6274	0,5803	0,5547	0,4806	0,4686	0,4330	**0,6392**	0,6146	0,6035	0,6040	0,3624
copyMachine	0,3856	0,1857	0,0447	0,0202	0,0175	0,3601	0,3763	0,4342	0,4517	0,4330	0,4434	0,2918	0,3015	0,4704	0,4685	**0,4848**	0,1004	0,3434	0,4280	0,4289	0,4477	0,3294
cubicle	0,2632	0,0043	0,0077	0,0045	0,0056	0,3276	0,2365	0,0789	0,0352	0,3800	0,0201	0,2544	0,3772	**0,4348**	0,3958	0,3508	0,1384	0,2704	0,3143	0,3427	0,3740	0,2198
fall	0,0340	0,0074	0,0003	0,0000	0,0000	0,0719	0,0282	0,0201	0,0167	0,0540	0,0124	0,0557	0,0832	0,0488	0,0343	0,0315	0,0001	0,0752	**0,0889**	0,0849	0,0594	0,0384
office	0,1401	0,5109	0,5074	0,4533	0,4203	0,9494	0,1580	0,0867	0,2647	0,6819	0,3843	0,9593	0,9611	0,7601	0,4963	0,2614	0,6384	0,9683	0,9600	0,9608	**0,9727**	0,5950
overpass	0,0568	0,0011	0,0026	0,0030	0,0042	0,5280	0,0229	0,0334	0,0255	0,0534	0,0081	0,4289	0,0685	0,0795	0,0881	0,0988	0,1494	0,5037	**0,6254**	0,2243	0,1070	0,1482
pedestrians	0,8575	0,4618	0,0998	0,0264	0,0080	0,7173	0,8615	0,8081	0,7549	**0,8797**	0,7242	0,4952	0,7861	0,8282	0,8669	0,8711	0,0998	0,6274	0,7066	0,7198	0,8121	0,6196
peopleInShade	0,5410	0,1407	0,1022	0,0924	0,0805	0,6314	0,5291	0,3261	0,2281	0,5252	0,2864	0,5097	0,6344	0,5985	0,6518	0,6434	0,3069	0,5543	0,6486	0,6715	**0,6936**	0,4474
PETS2006	0,5534	0,4571	0,3819	0,3802	0,3701	0,4151	0,5724	0,5945	0,5947	0,4827	0,5554	0,2583	0,4348	0,5156	0,5713	**0,6114**	0,2273	0,2722	0,3744	0,4183	0,4527	0,4521
sidewalk	0,0215	0,0026	0,0001	0,0000	0,0000	0,0269	0,0257	0,0124	0,0102	0,0597	0,0062	0,0004	0,0499	**0,0684**	0,0546	0,0345	0,0000	0,0015	0,0165	0,0415	0,0447	0,0227
sofa	0,7494	0,7777	0,7367	0,7024	0,6665	0,5159	0,7578	0,7889	**0,8060**	0,6998	0,7767	0,2571	0,5604	0,7080	0,7602	0,7656	0,0856	0,3971	0,4992	0,5985	0,6222	**0,6301**
tramstop	0,0735	0,0182	0,0046	0,0016	0,0011	0,1342	0,0672	0,0463	0,0329	0,1329	0,0239	0,0234	0,1531	0,1396	0,1130	0,0968	0,0006	0,0769	0,1894	**0,1928**	0,1905	0,0816
winterdriveway	0,1687	0,0035	0,0000	0,0000	0,0000	0,4954	0,2148	0,2544	0,1735	0,2607	0,1326	0,4733	0,6023	0,3441	0,3203	0,2543	0,1364	0,5118	0,6250	0,6214	**0,6521**	0,2974
MEDIA	0,3167	0,1707	0,1241	0,1105	0,1034	0,4242	0,3174	0,2638	0,2391	0,3960	0,2348	0,3320	0,4092	0,4084	0,3935	0,3726	0,1457	0,3835	**0,4498**	0,4400	0,4449	

Tabla 4.11: Resultados configuraciones Edge.

ACF

En el caso del algoritmo ACF tendremos también dos modelos, el entrenado con el modelo de persona de INRIA y el entrenado con el modelo de persona de Caltech. En ambos casos los parámetros escogidos son los mismos y pueden verse en la Tabla 4.12.

Como se ha observado anteriormente en el resto de algoritmos nos encontramos con sensibilidad respecto al tipo de vídeo en primer lugar por la fase de detección de ROI's. El vídeo abandonedBox sigue presentando esa dificultad a la hora de realizar la tarea de la extracción de regiones de interés.

En cuanto al modelo de persona nos encontramos con un algoritmo que utiliza el modelo holístico y por lo tanto presenta menor flexibilidad en cuanto al ángulo de la cámara, esto puede observarse en los vídeos *winterdriveway* o *copyMachine* para ambos modelos.

Merece la pena destacar la apreciable diferencia de rendimientos de ambos modelos debido a que solamente se diferencian en el modelo de persona con el que han sido entrenados. De ello podemos observar que el modelo entrenado con la base de datos de Caltech ofrece un rendimiento superior.

En cuanto a la parametrización se han logrado mejoras con respecto a la configuración inicial. En esta ocasión, y como ya hicimos con el algoritmo DTDP podemos conjuntar las mejoras primero fijando el valor de *nPerOct* y observando que para un mismo valor de este parámetro se obtienen mejores resultados a medida que aumenta el valor *nOctUp*, debido a que admitimos tamaño base de persona inferior. Si nos fijamos en el valor de *nPerOct* se observa una mejoría importante cuando saltamos del valor 4 al 8 y una no tan importante del 8 al 12 debido a que tanta nitidez entre escalas llegado un punto no es tan significante para el rendimiento.

Como puede verse en la Tabla 4.13 y en la Tabla 4.14 las mejores configuraciones de media para ACF Caltech y ACF Inria son respectivamente L y J.

La Figura 4.18 y la Figura 4.19 muestran los resultados de las diferentes configuraciones de ambos modelos para cada uno de los vídeos del *dataset*.

ACF															
Par	A	B	C	D	E	F	G	H (Ini)	I	J	K	L	M	N	O
Umbral	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10	-10
nPerOct	4	4	4	4	4	8	8	8	8	8	12	12	12	12	12
nOctUp	-1	-0,5	0	0,5	1	-1	-0,5	0	0,5	1	-1	-0,5	0	0,5	1

Tabla 4.12: Configuraciones ACF propuestas.

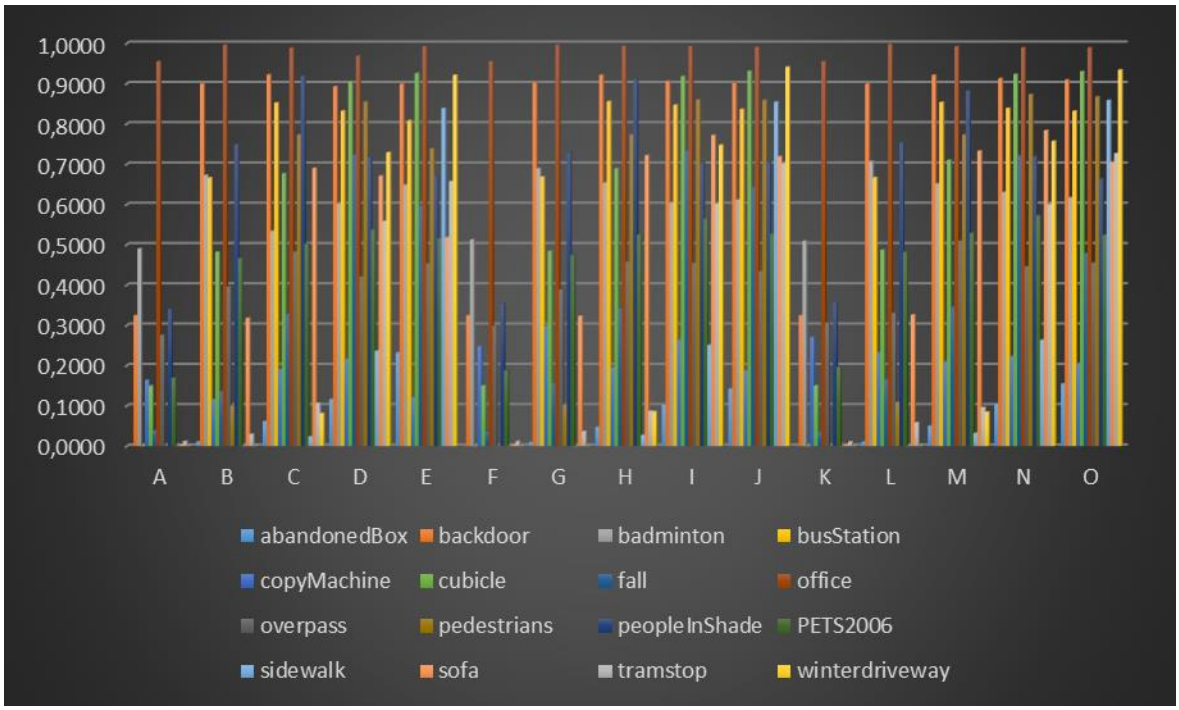


Figura 4.18: Gráfica de los resultados de las configuraciones del algoritmo ACF Inria por cada vídeo.

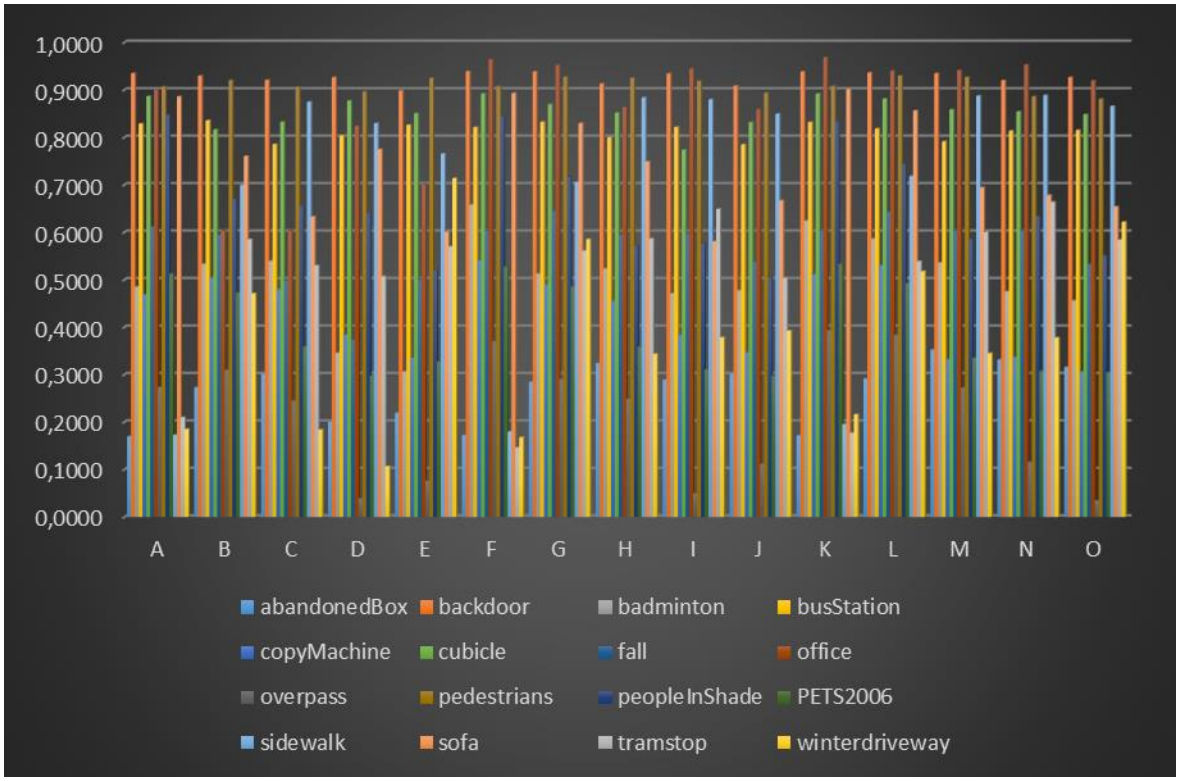


Figura 4.19: Gráfica de los resultados de las configuraciones del algoritmo ACF Caltech por cada vídeo.

ACF Inria

VÍDEO	A	B	C	D	E	F	G	H (Ini)	I	J	K	L	M	N	O	MEDIA
abandonedBox	0,0000	0,0111	0,0617	0,1167	**0,2325**	0,0000	0,0095	0,0470	0,1035	0,1427	0,0000	0,0105	0,0501	0,1053	0,1551	0,0653
backdoor	0,3244	0,8992	0,9219	0,8929	0,8989	0,3244	0,9023	**0,9212**	0,9060	0,9010	0,3244	0,8992	0,9210	0,9128	0,9092	0,7412
badminton	0,4901	0,6729	0,5338	0,6021	0,6487	0,5125	0,6893	0,6537	0,6048	0,6118	0,5090	**0,7073**	0,6515	0,6303	0,6171	0,5709
busStation	0,0007	0,6664	0,8523	0,8320	0,8083	0,0007	0,6684	**0,8555**	0,8469	0,8363	0,0007	0,6668	0,8537	0,8387	0,8319	0,5975
copyMachine	0,1637	0,1160	0,1910	0,2168	0,1210	0,2479	**0,2985**	0,1956	0,2642	0,1876	0,2708	0,2328	0,2102	0,2231	0,2057	0,1966
cubicle	0,1496	0,4819	0,6768	0,9055	0,9255	0,1502	0,4842	0,6893	0,9180	**0,9310**	0,1502	0,4872	0,7102	0,9232	0,9302	0,5946
fall	0,0388	0,1360	0,3283	0,7233	0,5964	0,0357	0,1563	0,3425	**0,7334**	0,6425	0,0351	0,1660	0,3458	0,7211	0,4783	0,3425
office	0,9552	0,9958	0,9885	0,9690	0,9922	0,9552	0,9960	0,9930	0,9926	0,9907	0,9552	**0,9977**	0,9920	0,9895	0,9893	**0,9220**
overpass	0,2759	0,3956	0,4826	0,4206	0,4534	0,2987	0,3892	0,4580	0,4549	0,4343	0,3065	0,3307	**0,5094**	0,4459	0,4550	0,3819
pedestrians	0,0000	0,1006	0,7732	0,8552	0,7388	0,0000	0,1034	0,7732	0,8599	0,8591	0,0000	0,1092	0,7729	**0,8731**	0,8675	0,4804
peopleInShade	0,3399	0,7489	**0,9192**	0,7170	0,6711	0,3553	0,7285	0,9102	0,7050	0,7064	0,3565	0,7533	0,8820	0,7196	0,6639	0,6360
PETS2006	0,1695	0,4660	0,5017	0,5371	0,5160	0,1877	0,4738	0,5256	0,5650	0,5270	0,1960	0,4816	0,5291	**0,5721**	0,5239	0,4233
sidewalk	0,0000	0,0007	0,0237	0,2363	0,8385	0,0000	0,0007	0,0272	0,2511	0,8543	0,0000	0,0010	0,0320	0,2635	**0,8585**	0,2117
sofa	0,0035	0,3175	0,6899	0,6709	0,5184	0,0035	0,3229	0,7215	0,7719	0,7190	0,0035	0,3258	0,7327	**0,7837**	0,7053	0,4556
tramstop	0,0118	0,0297	0,1068	0,5584	0,6573	0,0116	0,0366	0,0874	0,6020	0,7024	0,0109	0,0584	0,0962	0,6003	**0,7265**	0,2685
winterdriveway	0,0000	0,0000	0,0812	0,7289	0,9210	0,0000	0,0000	0,0859	0,7475	**0,9411**	0,0000	0,0000	0,0847	0,7569	0,9339	0,3301
MEDIA	0,1827	0,3774	0,5083	0,6239	0,6586	0,1927	0,3912	0,5179	0,6454	**0,6867**	0,1949	0,3892	0,5233	0,6474	0,6782	

Tabla 4.13: Resultados configuraciones ACF Inria.

ACF Caltech

VÍDEO	A	B	C	D	E	F	G	H (Ini)	I	J	K	L	M	N	O	MEDIA
abandonedBox	0,1698	0,2728	0,3017	0,2007	0,2192	0,1723	0,2846	0,3238	0,2892	0,3021	0,1718	0,2916	**0,3525**	0,3316	0,3158	0,2500
backdoor	0,9348	0,9291	0,9204	0,9260	0,8981	**0,9389**	0,9382	0,9128	0,9340	0,9083	0,9382	0,9360	0,9345	0,9195	0,9263	**0,8684**
badminton	0,4849	0,5330	0,5392	0,3455	0,3062	**0,6577**	0,5125	0,5234	0,4709	0,4773	0,6240	0,5862	0,5357	0,4752	0,4559	0,4705
busStation	0,8284	**0,8354**	0,7852	0,8036	0,8258	0,8210	0,8321	0,7993	0,8212	0,7845	0,8315	0,8181	0,7911	0,8133	0,8146	0,7628
copyMachine	0,4686	0,5037	0,4801	0,3841	0,3355	**0,5404**	0,4897	0,4554	0,3834	0,3471	0,5113	0,5302	0,3326	0,3369	0,3059	0,4003
cubicle	0,8864	0,8161	0,8321	0,8770	0,8503	**0,8918**	0,8695	0,8516	0,7736	0,8315	0,8916	0,8810	0,8584	0,8543	0,8482	0,8008
fall	0,6114	0,5947	0,4981	0,3733	0,5007	0,6023	**0,6449**	0,5947	0,5967	0,5376	0,6038	0,6423	0,6029	0,6027	0,5329	0,5337
office	0,8994	0,6036	0,6037	0,8237	0,7010	0,9641	0,9524	0,8635	0,9444	0,8591	**0,9681**	0,9399	0,9411	0,9529	0,9194	0,8085
overpass	0,2735	0,3095	0,2451	0,0393	0,0761	0,3706	0,2905	0,2488	0,0498	0,1121	**0,3930**	0,3836	0,2721	0,1159	0,0343	0,2009
pedestrians	0,9065	0,9200	0,9059	0,8960	0,9248	0,9072	0,9271	0,9248	0,9183	0,8934	0,9078	0,9293	0,9267	0,8859	0,8805	0,8534
peopleInShade	**0,8470**	0,6699	0,6554	0,6403	0,5204	0,8447	0,7164	0,5715	0,5763	0,5050	0,8332	0,7440	0,5853	0,6333	0,5506	0,6183
PETS2006	0,5126	0,4727	0,3594	0,2978	0,3276	0,5277	0,4851	0,3588	0,3109	0,2969	**0,5329**	0,4924	0,3353	0,3074	0,3046	0,3701
sidewalk	0,1728	0,6991	0,8744	0,8293	0,7654	0,1801	0,7056	0,8836	0,8796	0,8492	0,1955	0,7182	0,8873	0,8883	0,8655	0,6496
sofa	0,8856	0,7600	0,6328	0,7746	0,6005	0,8928	0,8295	0,7486	0,5805	0,6667	**0,9012**	0,8560	0,6942	0,6780	0,6536	0,6972
tramstop	0,2098	0,5847	0,5295	0,5071	0,5697	0,1463	0,5609	0,5863	0,6487	0,5038	0,1769	0,5386	0,6001	**0,6635**	0,5837	0,4631
winterdriveway	0,1850	0,4707	0,1837	0,1066	**0,7135**	0,1679	0,5851	0,3433	0,3782	0,3924	0,2159	0,5173	0,3448	0,3777	0,6214	0,3502
MEDIA	0,5798	0,6234	0,5842	0,5516	0,5709	0,6016	0,6640	0,6244	0,5972	0,5792	0,6061	**0,6753**	0,6247	0,6148	0,6008	

Tabla 4.14: Resultados configuraciones ACF Caltech.

Como observamos en las gráficas y en los resultados numéricos el algoritmo ACF, en sus dos variantes, ofrece buenos resultados. Pero, como ya hemos indicado, la disparidad del rendimiento entre ambos es debida al modelo de persona con el que cada uno ha sido entrenado. Esto nos lleva a concluir que el entrenamiento con el *dataset* de Caltech se ajustará más al modelo de persona que se considera "real" en este PFC y por lo tanto a las diferentes situaciones que podemos enfrentar en el día a día.

4.5.4 Configuraciones óptimas para cada vídeo

VÍDEO	HOG	ISM	DTDP	FUSION	EDGGE	CALTECH	INRIA	MEDIA
abandonedBox	0,0007	0,0593	**0,3715**	0,3380	0,1752	0,3525	0,2325	0,2185
backdoor	0,8472	0,6618	0,9152	0,6985	0,8558	**0,9389**	0,9212	0,8341
badminton	0,4843	0,2723	**0,7296**	0,1242	0,2798	0,6577	0,7073	0,4650
busStation	0,7063	0,4505	0,8440	0,5528	0,6392	0,8354	**0,8555**	0,6977
copyMachine	0,1208	0,2903	**0,6325**	0,4969	0,4848	0,5404	0,2985	0,4092
cubicle	0,4908	0,1342	**0,9404**	0,3275	0,4348	0,8918	0,9310	0,5929
fall	0,1115	0,0077	0,4610	0,0251	0,0889	0,6449	**0,7334**	0,2961
office	0,9334	0,4296	0,9707	0,9968	0,9727	0,9681	**0,9977**	0,8956
overpass	0,4464	0,4538	**0,7644**	0,5996	0,6254	0,3930	0,5094	0,5417
pedestrians	0,6598	0,6032	0,9028	0,9217	0,8797	**0,9308**	0,8731	0,8244
peopleInShade	0,2449	0,3354	0,8532	0,6756	0,6936	0,8470	**0,9192**	0,6527
PETS2006	0,4151	0,5931	**0,6255**	0,6051	0,6114	0,5329	0,5721	0,5650
sidewalk	0,0002	0,0116	0,8090	0,1815	0,0684	**0,8910**	0,8585	0,4029
sofa	0,5433	0,1823	**0,9138**	0,7071	0,8060	0,9012	0,7837	0,6911
tramstop	0,0852	0,0190	0,6832	0,2670	0,1928	0,6635	**0,7265**	0,3767
winterdriveway	0,0980	0,0710	0,7616	0,8366	0,6521	0,7135	**0,9411**	0,5820
MEDIA	0,3867	0,2859	**0,7611**	0,5221	0,5288	0,7314	0,7413	

Tabla 4.15: Configuraciones óptimas para cada vídeo.

A continuación se han escogido para cada vídeo y para cada algoritmo aquella configuración que ofrecía el mayor rendimiento y por lo tanto el óptimo por algoritmo y por vídeo. Esto sería útil para sistemas *ad-hoc* como son muchos de los que se diseñan porque están programados y parametrizados para unas condiciones pre fijadas y sin gran capacidad de adaptación. Esto nos llevará a una mejora considerable en el rendimiento local por vídeo y medias para todos los algoritmos y vídeos pero en la realidad no nos dará una pista sobre un sistema óptimo en cuanto a flexibilidad.

Como se ha observado en los resultados particulares de cada algoritmo, volvemos a encontrarnos (en la Tabla 4.15) con la sensibilidad de todos los algoritmos con respecto a la tarea de extracción de las ROI's. Así, *abandonedBox* o *fall*, a pesar de haber mejorado considerablemente sus valores medios con respecto a la configuración inicial siguen distando mucho de los valores de otros vídeos que presentan un fondo más sencillo y estático.

La sensibilidad al ángulo de la cámara así como al tamaño de persona no se aprecian tan considerablemente como en los anteriores análisis debido a que la parametrización de los diferentes algoritmos ha optimizado su rendimiento. Esto nos demuestra que mediante una correcta parametrización el rendimiento de todos los algoritmos es mejorable.

Se puede observar que, de media, el mejor algoritmo es el DTDP que principalmente supera a ACF en vídeos con tamaños de persona muy grandes o con el punto de captura en posición casi cenital, esto es debido a que el modelo de persona que adopta el algoritmo DTDP es por partes proporcionándole una mayor flexibilidad, mientras que el ACF utiliza un modelo holístico.

En la Figura 4.20 puede observarse la comparación del rendimiento de los diferentes algoritmos con sus configuraciones iniciales (arriba) y óptimas para cada uno de los vídeos (abajo). A pesar de que el rendimiento de cada algoritmo en este caso ha sido evaluado con su configuración óptima para cada uno de ellos se observa la gran diferencia entre ellos.

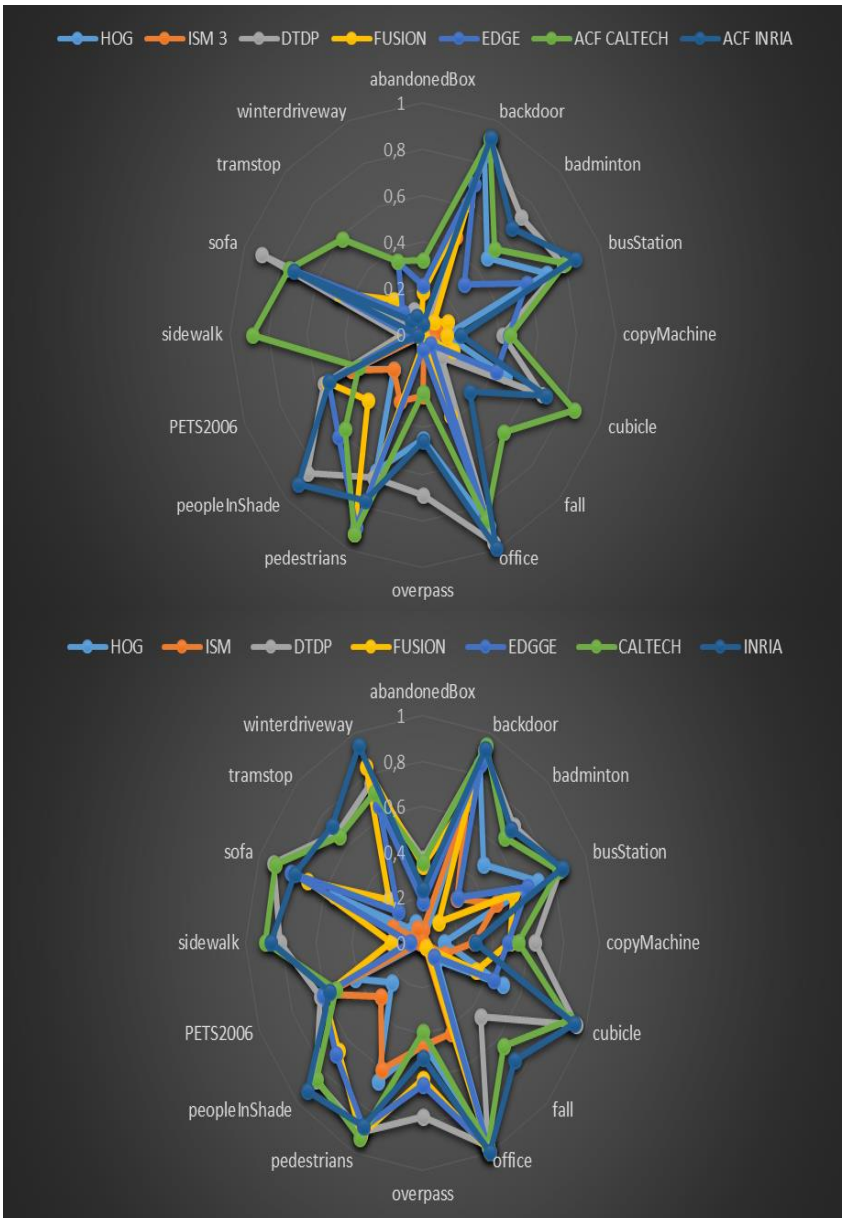


Figura 4.20: Gráfico que muestra los resultados de cada algoritmo para cada vídeo para la configuración inicial arriba y ad-hoc (óptima) abajo.

4.5.5 Configuraciones óptimas de media en el *dataset*

A continuación se mostrarán las configuraciones que ofrecen mejor media para el *dataset*. De esta forma podríamos configurar nuestro sistema de manera óptima para brindar la flexibilidad necesaria que nos exigiría casi cualquier escenario en la vida real.

VÍDEO	HOG	ISM	DTDP	FUSION	EDGGE	CALTECH	INRIA	MEDIA
abandonedBox	0,0001	0,0093	**0,3610**	0,3380	0,1041	0,2916	0,1427	0,1781
backdoor	0,8472	0,6618	0,8920	0,6985	0,7453	**0,9360**	0,9010	**0,8117**
badminton	0,4843	0,1961	**0,6527**	0,1217	0,2561	0,5862	0,6118	0,4155
busStation	0,7052	0,4505	**0,8418**	0,4918	0,6146	0,8181	0,8363	0,6797
copyMachine	0,1203	0,2903	0,1736	0,2876	0,4280	**0,5302**	0,1876	0,2882
cubicle	0,4908	0,1337	**0,9404**	0,2967	0,3143	0,8810	0,9310	0,5697
fall	0,1115	0,0020	0,4610	0,0151	0,0889	0,6423	**0,6425**	0,2805
office	0,9334	0,3035	0,9611	0,9598	0,9600	0,9399	**0,9907**	0,8641
overpass	0,4209	0,4538	**0,7237**	0,0399	0,6254	0,3836	0,4343	0,4402
pedestrians	0,6598	0,4281	0,9028	0,9212	0,7066	**0,9293**	0,8591	0,7724
peopleInShade	0,2449	0,3354	0,7233	0,6545	0,6486	**0,7440**	0,7064	0,5796
PETS2006	0,4151	0,5931	**0,6255**	0,4314	0,3744	0,4924	0,5270	0,4941
sidewalk	0,0002	0,0000	0,8090	0,1815	0,0165	0,7182	**0,8543**	0,3685
sofa	0,5281	0,1129	**0,8908**	0,4557	0,4992	0,8560	0,7190	0,5802
tramstop	0,0852	0,0051	0,6832	0,2016	0,1894	0,5386	**0,7024**	0,3436
winterdriveway	0,0980	0,0009	0,7616	0,4417	0,6250	0,5173	**0,9411**	0,4837
MEDIA	0,3840	0,2485	**0,7127**	0,4085	0,4498	0,6753	0,6867	

Tabla 4.16: Configuraciones óptimas de media para la totalidad del *dataset*.

Como ya se ha mencionado en este apartado de conclusiones y puede verse en la Tabla 4.16 seguimos sufriendo el efecto de la dificultad de la extracción de las ROI's/Segmentación y seguimos obteniendo rendimientos muy bajos con respecto al resto en vídeos como *abandonedBox*.

Como ya se ha comentado en el apartado anterior (ver sección 4.5.4) la parametrización nos ayuda a obtener mejores resultados frente a la variación del ángulo de cámara o de detectar personas más pequeñas (parámetro que se ha modificado en los algoritmos que lo permiten y que ofrece una mejora considerable de rendimiento).

El objetivo por el que nace este PFC es el estudio en un medio "neutral" de una representación de los algoritmos de detección de personas que conforman el estado del arte. En el apartado anterior proponíamos unas configuraciones óptimas por cada vídeo, lo que acarrea un modelo ad-hoc que fue precisamente la crítica que se exponía en la introducción de este trabajo. Por eso hemos buscado qué configuración ofrece unos mejores resultados de media por cada algoritmo para seleccionar los parámetros que ofrecen un mejor sistema de lo que sería un sistema real para implementar y que tuviera la menor sensibilidad posible al modelo de persona. De esta manera seremos capaces de ver qué algoritmo y configuración serán los candidatos óptimos para un sistema flexible de detección de personas válido para casi cualquier escenario de la vida real con buenos resultados.

El modelo que aquí presentamos de configuración nos ofrece el más adecuado para implementar, cada uno de los algoritmos, si queremos una utilización en cualquier entorno

y condiciones y sin conocerlas de forma predefinidas. Esto es debido a la flexibilidad que obtienen al haber variado sus diferentes parámetros. Podemos concluir por tanto que este modelo sería el que podríamos denominar el más realista.

Sobre el resto hay dos algoritmos que ofrecen un mayor rendimiento desde el comienzo, DTDP y ACF. Entre estos dos algoritmos se ha podido observar que el DTDP obtiene un mayor rendimiento y como ya se ha comentado esto es debido a que utiliza un modelo más flexible, basado en partes, para su modelo de persona mientras que ACF utiliza un modelo holístico. Si el escenario ofreciera una cámara en horizontal, que no demanda tanta flexibilidad como algunos de los escenarios aquí propuestos, posiblemente estos resultados se verían alterados y el algoritmo ACF superaría al DTDP.

Para concluir, en la Figura 4.21 y la Figura 4.22, mostramos el "antes" y el "después" de la parametrización, por un lado la inicial con nuestro umbral, y por otro la mejor media para el *dataset* en su conjunto. De esta manera tan gráfica se ve la expansión de rendimiento que se obtiene con una correcta parametrización de los algoritmos.

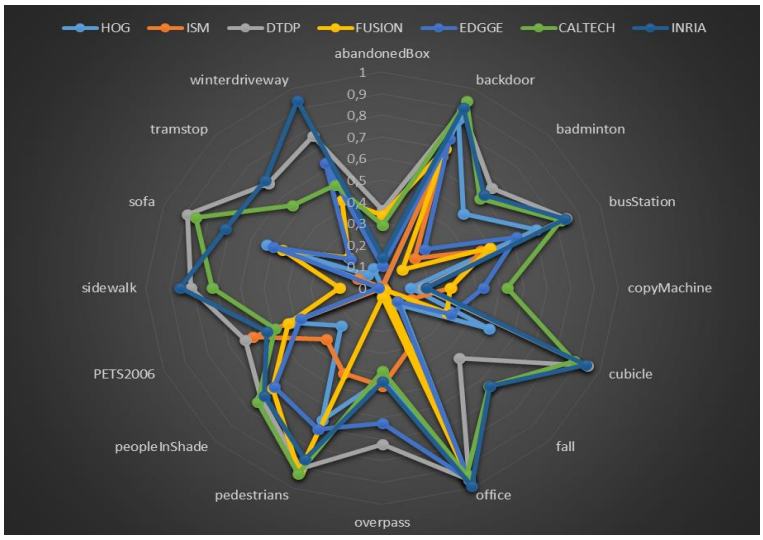


Figura 4.21: Gráfico que muestra el rendimiento de los algoritmos con la mejor configuración de media de todos los vídeos evaluado para todos ellos.

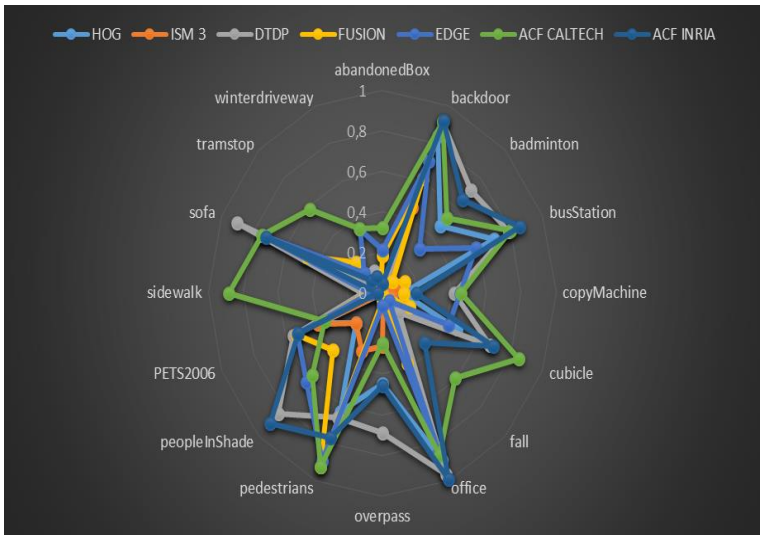


Figura 4.22: Gráfico que muestra el rendimiento de los algoritmos con la configuración inicial con umbral mínimo.

4.6 Conclusiones

Como se observa en los resultados obtenidos los algoritmos son muy sensibles a sus parámetros de configuración y al tipo de vídeo a analizar. Para cada tipo de vídeo a analizar, se pueden mejorar considerablemente los resultados con una configuración adecuada de sus parámetros, logrando resultados mejores a los que ofrece el algoritmo con la configuración inicial del autor.

Así mismo, cuando no nos centramos en el vídeo en sí y buscamos los parámetros óptimos para la generalidad de los vídeos la diferencia de rendimiento con el mejor de los casos también es muy grande.

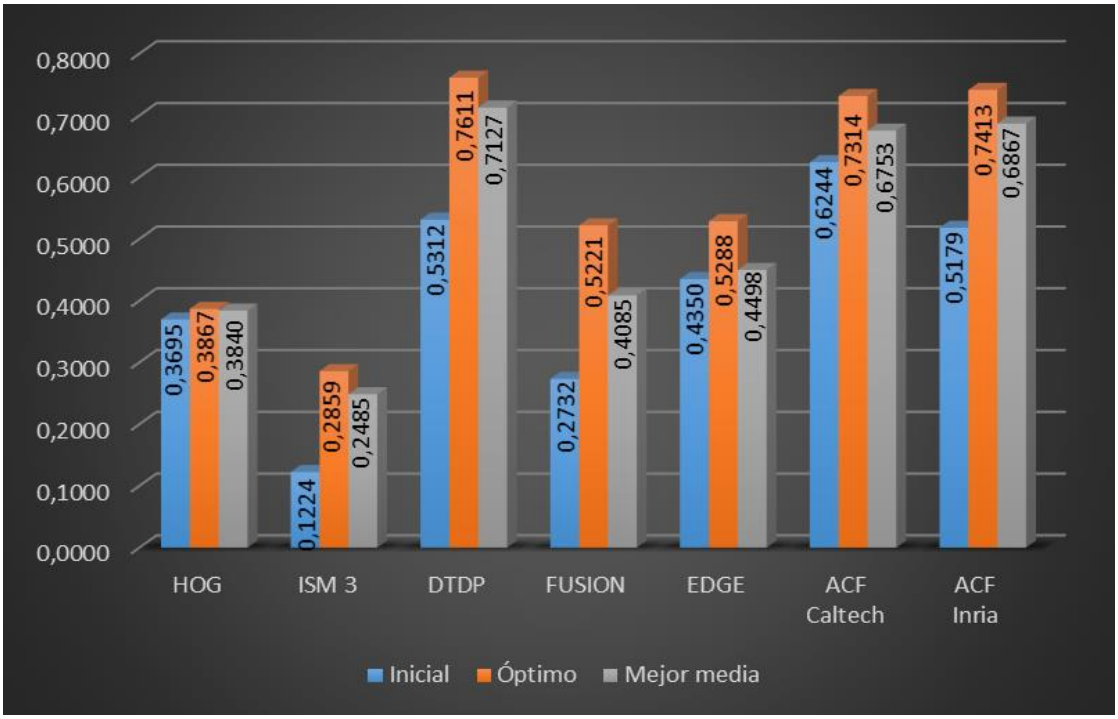


Figura 4.23: Gráfica que muestra la divergencia de resultados de los algoritmos según la configuración de parámetros.

Partiendo de la base de que se busca un algoritmo flexible y válido para cualquier escenario, no un modelo *ad-hoc*, lo que nos queda para mejorar los algoritmos es una parametrización óptima de media para todo el *dataset*.

De los datos anteriormente expuestos y como se muestra en la Figura 4.23 se observa que la parametrización de los algoritmos es un paso crítico para lograr un incremento en su rendimiento.

Si uno se fija en el algoritmo HOG se observa que la variación entre la configuración inicial, la óptima y la de mejor media a penas difiere. Esto es debido a que en los tres casos el rendimiento de este algoritmo, de media para todo el dataset, es muy bajo por lo que puede concluirse que este algoritmo es muy específico para el *dataset* con el que fue entrenado y por lo tanto ofrece muy poca sensibilidad en cuanto a modelo de vídeo a analizar.

En el resto de los casos se observa claramente una importante mejoría de rendimiento con la parametrización, tanto óptima como de media.

Como observamos la configuración óptima ofrece los mejores resultados, pero no es viable puesto que para cada vídeo, cada algoritmo debe tener una configuración diferente de sus parámetros y esto es incompatible con la flexibilidad que se busca.

En este caso la configuración que nos interesa es aquella que ofrece un mayor rendimiento para el escenario más amplio posible y este viene representado por la mejor media.

Como se ha podido observar a lo largo de este proyecto, la detección de personas es un campo en continua evolución. Los algoritmos presentes en el estado del arte pueden ofrecer, según sus autores, brillantes resultados en esta difícil tarea pero por desgracia esos resultados suelen estar evaluados para unas condiciones muy concretas de vídeo. En la práctica se observa que los algoritmos de detección de personas tienen una gran dependencia del vídeo a analizar (colores, escenarios, calidad, resolución, fondos, etc.) así como de sus parámetros de configuración. Esta dependencia puede suavizarse mediante una correcta parametrización, como se ha visto en este trabajo.

En los casos estudiados, esta configuración por defecto, mostraba peores resultados que otras configuraciones probadas, llegando, la inicial, a tener un rendimiento inferior al 50% que otra de las probadas. Esto refleja que cuando el autor escogió esta configuración lo hizo basándose en un *dataset* más homogéneo que el trabajado en este PFC, es por ello que estos sistemas *ad-hoc* ofrecen mejores resultados en las investigaciones de los autores y sobre un *dataset* muy concreto. Cuando salimos de esos entornos controlados y enfrentamos a los algoritmos a un *dataset* más generalista y probamos otras configuraciones se observa que pueden mejorarse considerablemente los rendimientos de los mismos.

Para concluir a continuación se detallan los parámetros que, para cada algoritmo, han ofrecido el mejor resultado de media.

Parámetro	**Algoritmo**						
	HOG	ISM	DTDP	FUSION	EDGE	ACF INRIA	ACF CALTECH
Scaleratio	1.025						
Max Scale		2.5					
Sbin			4				
Interval			15				
Var				29	29		
winQ				9	5		
nPerOct						8	12
nOctUp						1	-0.5

Tabla 4.17: La tabla muestra los valores de las mejores configuraciones de los parámetros que modificamos para la mejor media para cada algoritmo.

5 Conclusiones y trabajo futuro

En este proyecto fin de carrera se han desarrollado los siguientes puntos:

1. Estudio del estado del arte: a lo largo de este proyecto fin de carrera se ha realizado un estudio del estado del arte, se han detectado las características y técnicas comunes de los algoritmos de detección de personas.
2. Clasificación de los algoritmos de detección de personas: se ha propuesto una arquitectura básica para cualquier algoritmo de detección de personas y una clasificación que englobe todas las posibles técnicas tenidas en cuenta para este estudio.
3. Selección de los algoritmos más relevantes de detección de personas: a continuación se ha seleccionado una muestra representativa de los algoritmos y se ha realizado un estudio tanto del algoritmo como de sus parámetros a partir de la documentación del autor. Tras esto, se han elegidos los valores de los parámetros para realizar la evaluación comparativa de su rendimiento.
4. Selección de una base de datos de vídeo (*dataset*) adecuado para la evaluación de los algoritmos: una vez elegidos los algoritmos y sus parámetros se ha procedido a seleccionar un *dataset* generalista que contuviera situaciones y escenarios de todo tipo.
5. Etiquetado manual para generar el Ground Truth (GT): una vez seleccionado el dataset se ha fijado el criterio de persona y se ha etiquetado manualmente cada uno de los vídeos.
6. Procesado de los algoritmos seleccionados sobre el *dataset*: con todo esto se ha procedido a descargar y ejecutar el código de los algoritmos con los diferentes parámetros seleccionados y se han recogido los resultados arrojados por los mismos.
7. Desarrollo, implementación del sistema de evaluación elegido: para calcular el rendimiento de los algoritmos se ha desarrollado un sistema de evaluación basado en el cálculo del área bajo la curva generada de 1-Precission/Recall, esta métrica tiene en cuenta tanto la localización de personas en la imagen, su localización espacial y las falsas detecciones.
8. Análisis de los resultados y conclusiones: la fase final de este proyecto fin de carrera ha consistido en evaluar y comentar los resultados obtenidos. Se ha visto la dependencia de todos los algoritmos con el tipo de vídeo analizado (principalmente debido a la dificultad de la extracción de ROI's, el ángulo de captura de la imagen o el tamaño de persona) y de las configuraciones elegidas para cada algoritmo.

Como ya se ha visto a lo largo de este estudio la detección de personas es un área en continua investigación y evolución. Debido a ello el espectro de algoritmos se incrementa sin cesar. Además, el aumento de capacidad de cómputo y el abaratamiento de costes hacen que los investigadores puedan explorar nuevas soluciones para este ámbito. Por ello, este trabajo puede seguir evolucionando con los algoritmos e ir incrementando los mismos para las pruebas o incluso sustituir alguno usado por una evolución del mismo o por uno que, basado en lo mismo consiga mejores resultados.

Como ya se ha comentado en este PFC, hay otras características que no se han tenido en cuenta en este proyecto y que entrarían en la clasificación de los algoritmos para nuevas posibilidades de detección, por ejemplo la utilización de la información de tracking podría incluirse como un futuro desarrollo brindando la posibilidad de utilizar algoritmos que utilizan esta técnica para lograr resultados más fiables.

En la primera etapa crítica (ROI) existen algoritmos que utilizan información o técnicas que, debido a la naturaleza del *dataset* y al alcance de este PFC no se han tenido en cuenta. Por ello utilizar otro dataset con imágenes captadas con cámaras 3-D puede ser interesante porque los algoritmos que utilizan esta información se pueden ver muy beneficiados en la etapa de ROI al lograr mejores resultados por esta técnica de captación de imagen.

Otro punto de vista muy importante, especialmente en tareas como la asistencia a la conducción o a vídeo-vigilancia, podría ser introducir en los resultados un baremo de recursos consumidos y/o tiempo de proceso de cada imagen. Esto estaría enfocado a buscar los óptimos dentro de la detección de personas para sistemas en tiempo real. Una demanda que crece a diario y que podría dar otra métrica interesante a la hora de examinar los algoritmos obteniendo una medición combinada de efectividad/tiempo para cada uno de los algoritmos.

Uno de los pilares de la evaluación de algoritmos es la base de imágenes/vídeos que se utilizan para su prueba, en este sentido se podría buscar una base de vídeo más amplía para realizar esta evaluación. El gran problema de este punto, además de los recursos que algunos de los algoritmos consumen, es que esta base de datos debería ser abierta y estar disponible, cosa que, entre otros motivos por protección de datos, es complicado en la actualidad y un recurso muy solicitado. Además, la tarea de generar una base de datos que aúne todas las situaciones, fondos, condiciones de iluminación, etc, posibles es muy complicado porque nos veríamos ante algo que sería casi infinito.

Una posible futura contribución para este trabajo sería, a la luz de los resultados observados, implementar un sistema retroalimentado automático que, a partir de ciertos indicadores para el vídeo que se está analizando, ajuste los parámetros de los algoritmos conforme a lo visto en este PFC.

Para concluir, otra posible variante de este PFC o trabajo futuro, podría ser utilizar otras métricas de evaluación diferentes a *Precisión-Recall* o incluso variar el criterio de acierto en la detección.

6 Bibliografía

[1] A. T. Nghiem, F. Bremond, M. Thonnat, V. Valentin, Etiseo. (2007). Performance evaluation for video surveillance systems. *in: Proc. of AVSS*.

[2] R. Vezzani, R. Cucchiara. (2010). Video surveillance online repository (visor): an integrated framework. *Multimedia Tools and Applications, 50(2),* 359-380.

[3] M. Enzweiler and D. M. Gavrila. (2009). Monocular pedestrian detection: Survey and experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 31(12),* 2179-2195.

[4] D. Gerónimo, A. M. López, A. D. Sappa and T. Graff. (2010). Survey of pedestrian detection for advanced driver assistance systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 23(7),* 1239-1258.

[5] P. Dollár, C. Wojek, B. Schiele, and P. Perona. (2012). Pedestrian detection: An evaluation of the state of the art. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 34(4),* 743-761.

[6] García, A. (2013). Tesis Doctoral: Contributions to robust people detection in video-surveillance. Madrid: VPULab.

[7] V. Fernández Carbajales, M. A. García, J. M. Martínez. (2008). Robust people detection by fusion on evidence from multiple methods. *In Proc. of WIAMIS,* 55-58.

[8] M. Hussein, W. Abd-Almageed, Y. Ran, and L. Davis. (2006). Real-time human detection, tracking, and verification in uncontrolled camera motion environments. *In Proc. of ICVS,* 41-47.

[9] P. Kilambi, E. Ribnick, A. J. Joshi, O. Masoud, and N. Papanikolopoulos. (2008). Estimating pedestrian counts in groups. *Computer Vision and Image Understanding*. *110(1),* 43-59.

[10] S. Harasse, L. Bont, and M. Desvignes. (2006). Human model for people detection in dynamic scenes. *In Proc. of CVPR,* 335-354.

[11] D. M. Gavrila and S. Munder. (2007). Multi-cue pedestrian detection and tracking from a moving vehicle. *International Journal of Computer Vision, 73(1),* 41-59.

[12] Koening, N. (2007). Toward real-time human detection and tracking in diverse environments. *In Proc. of ICDL,* 94-98.

[13] N. Dalal and B. Triggs. (2005). Histogram of oriented gradients for human detection. *InProc. of CVPR,* 886-893.

[14] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. (2010). Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 32(9),* 1627-1645.

[15] J. Yu, D. Farin, and B. Schiele. (2011). Multi-target tracking in crowded scenes. *In Proc. of DAGM,* 406-415.

[16] A. Ess, B. Leibe, K. Schindler, and L. V. Gool. (2009). Robust multiperson tracking from a mobile platform. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 31(10),* 1831-1846.

[17] B. Leibe and B. Schiele. (2004). Scale invariant object categorization using a scale-adaptative mean-shift search. *In Proc. of DAGM,* 145-153.

[18] B. Leibe, A. Leonardis, and B. Schiele. (2008). Robust object detection with interleaved categorization and segmentation. *International Journal of Computer Vision, 77(1-3),* 259-289.

[19] M. Andriluka, S. Roth, and B. Schiele. (2008). People-tracking-by-detection and people-detection-by-tracking. *In Proc. of CVPR*.

[20] P. Viola, M. J. Jones, and D. Snow. (2003). Detecting pedestrians using patterns of motion and appearance. *In Proc. of ICCV*, 734-741.
[21] X. Cui, Y. Liu, S. Shan, X. Chen, and W. Gao. (2007). 3d haar-like features for pedestrian detection. *In Proc. of ICME*, 1263-1266.
[22] Ren, X. (2008). Finding people in archive films through tracking. *In Proc. of CVPR*.
[23] S. Stalder, H. Grabner, and L. V. Gool. (2010). Cascade confidence filtering for improved tracking-by-detection. *In Proc. of ECCV*, 369-382.
[24] B. Leibe, K. Schindler, and L. V. Gool. (2007). Coupled detection and trajectory estimation for multi-object tracking. *In Proc. of ICCV*, 1-8.
[25] M. Andriluka, S. Roth, and B. Schiele. (2009). Pictorial structures revisited: People detection and articulated pose estimation. *In Proc. of CVPR*, 1014-1021.
[26] B. Leibe, E. Seeman, and B. Schiele. (2005). Pedestrian detection in crowded scenes. *In Proc. of CVPR*, 878-885.
[27] Á. García-Martín and J. M. Martínez. (2010). Robust real time moving people detection in surveillance scenarios. *In Proc. of AVSS*, 241-247.
[28] P. Dollár, Ron Appel, and Wolf Kienzle. (2012). Crosstalk Cascades for Frame-Rate Pedestrian Detection. *In Proc. of ECCV*.
[29] A. Cavallaro and T. Ebrahimi. (2000). Video object extraction based on adaptative background and statistical change detection. *In Proc. of SPIE, 15(10)*, 465-475.
[30] N. Goyette, P. -M. Jodoin, F. Porikli, J. Konrad, amd P. Ishwar. (2012). changedetection.net: A new change detection benchmark dataset. *In Proc. of IEEE Workshop on Change Detection (CDW-2012) at CVPR-2012*.
[31] I. Haritaoglu, D. Harwood, and L. S. Davis. (1998). Ghost: A human body part labeling system using silhouettes. *In Proc. of ICPR, 1*, 77-82.
[32] B. Wu and R. Nevatia. (2005). Detection of multiple, partially occluded humans in a single image by bayesian comination of edgelet part detectors. *In Proc. of ICCV, 1*, 90-97.
[33] Y. Freund and R. E. Schapire. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences, 55(1)*, 119-139.
[34] C. Huang, H. Ai, B. Wu, and S. Lao. (2004). Boosting nested cascade detector for multi-view face detection. *In Proc. of ICPR*, 415-418.
[35] P. Dollár, Z. Tu, P. Perona, and S. Belongie. (2009). Integral Channel Features. *In Proc. of BMVC*.
[36] P. Dollár, S. Belongie, and P. Perona. (2010). The fastest pedestrian detector in the west. *In Proc. of BMVC*.
[37] D. Lowe. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision, 60(2), 91-110*.
[38] P. Marín Belinchón. (2014). Detección de personas en tiempo real. *Trabajo Fin de Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación, Escuela Politécnica Superior de la Universiada Autónoma de Madrid*.
[39] A. García-Martín, J.M. Martínez, and J. Bescós. (2012). A corpus for benchmarking of people detection algorithms, Pattern Recognition Letters, 33(2), 152-156.

Glosario

PFC	Proyecto Fin de Carrera
GT	Ground Truth
VPULab	Video Processing and Understanding Lab
PC	Personal Computer
ROI	Region Of Interest
BLOB	Binary Large Object
HOG	Histogram of Oriented Gradients
SIFT	Scale-Invariant Feature Transform
ISM	Implicit Shape Model
SVM	Support Vector Machine
RGB	Red Green Blue
DoG	Difference of Gaussians
MDL	Minimum Description Length
LSVM	Latent Support Vector Machine
ROS	Region Of Support
ACF	Aggregate Channel Features
DiVa	Distributed Video Analysis Framework

Anexos

A. Configuración y ejecución de los algoritmos

A continuación describiremos los pasos dados para poder ejecutar cada uno de los algoritmos así como modificar los parámetros seleccionados y mostraremos un ejemplo de los ficheros de salida.

HOG

Para poder ejecutar el algoritmo, en entorno Linux y sobre un sistema de 64 bits, hemos tenido que instalar una serie de librerías de compatibilidad, puesto que desde la web del autor (http://pascal.inrialpes.fr/soft/olt/) nos hemos descargado los archivos binarios para ejecutarlos directamente y se encontraban para sistemas de 32 bits.

Para empezar hemos instalado libimlib2:i386 que nos da la compatibilidad 32bits – 64bits necesaria para ejecutarlo en el PC del laboratorio, tras esto damos permisos de creación/lectura/escritura/ejecución a los diferentes archivos que son llamados desde el ejecutable, a continuación incluimos en el *path* la ubicación de las librerías utilizadas por el HOG, OLTbinaries/lib. Con estos pasos estamos preparados para ejecutar con la línea de comando que se puede ver en la Figura A.1.

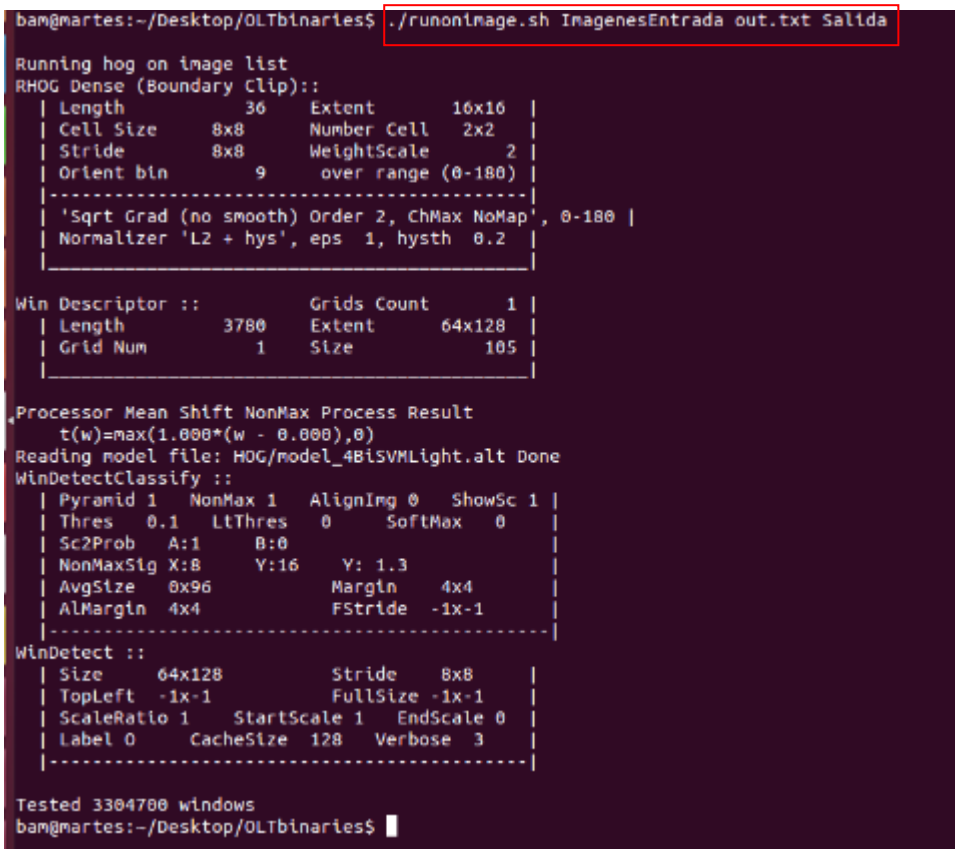
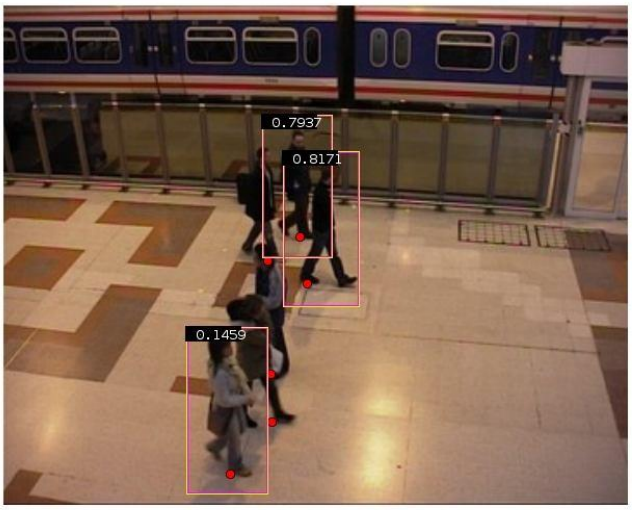


Figura A.1: Comando para ejecutar el algoritmo HOG.

```
./runonimage.sh ImagenesEntrada out.txt Salida
```

Donde **ImagenesEntrada** es el directorio de las imágenes con las que queremos trabajar, es importante que el nombre de las imágenes estén ordenadas con la secuencia de vídeo; **out.txt** es el archivo de texto donde tendremos la salida en formato texto y **Salida** es

el directorio donde se guardarán las salidas como imagen (en nuestro caso, y para evitar una excesiva ocupación de memoria, hemos decidido en lugar de un directorio poner directamente un archivo que sobrescribía con cada *frame*). A continuación, en la Figura A.2, se muestra un ejemplo de salida tanto en formato gráfico como en el contenido del fichero *txt* de salida.




```
606   264   126   252   1.96986 4.32705253e-01   2
  0     0     0     0         0       0       0
602   265   120   240   1.88147 4.94209886e-01   2
  0     0     0     0         0       0       0
595   264   122   244   1.90721 5.15482306e-01   2
  0     0     0     0         0       0       0
593   263   119   238   1.86676 8.44814956e-01   2
  0     0     0     0         0       0       0
590   264   121   242   1.89346 4.35211599e-01   2
  0     0     0     0         0       0       0
577   253   126   252   1.97596 3.27764899e-01   2
  0     0     0     0         0       0       0
574   250   127   254   1.98816 5.07696867e-01   2
  0     0     0     0         0       0       0
574   250   124   248   1.93962 5.60854554e-01   2
  0     0     0     0         0       0       0
572   255   113   226   1.77309 4.37336832e-01   2
  0     0     0     0         0       0       0
570   252   113   227   1.77442 1.87339246e-01   2
  0     0     0     0         0       0       0
566   238   117   235   1.84066 1.56586677e-01   2
  0     0     0     0         0       0       0
561   239   113   226   1.76612 4.19438004e-01   2
  0     0     0     0         0       0       0
560   240   109   218   1.70619 5.23162782e-01   2
  0     0     0     0         0       0       0
556   239   112   224   1.75717 4.24747765e-01   2
  0     0     0     0         0       0       0
551   238   113   226   1.76641 6.30882442e-01   2
  0     0     0     0         0       0       0
```

Figura A.2: Ejemplo de salida de HOG, a la izquierda imagen, a la derecha contenido de fichero de texto.

El fichero de salida, para cada línea, tiene los siguientes valores (si la línea son todo ceros no existe detección): coordenada x, coordenada y, ancho, alto, valor no utilizado, *score* del blob y número de *frame*. Por cada *frame* y de forma automática el algoritmo escribe una línea de ceros a continuación, por lo tanto, si encontramos varias filas de ceros seguidas significa que no hay detecciones en varios *frames*. En caso de que haya varias detecciones en el mismo *frame* irán seguidas sin líneas de cero entre medias.

ISM

La ejecución de este algoritmo la realizamos desde Linux y una vez que arranca nos manejaremos por su entorno gráfico. En primer lugar descargamos el código de la web del autor (http://www.vision.rwth-aachen.de/software/ism/ism-detector-code). A continuación y por temas de compatibilidad del sistema deberemos instalar librerías las siguientes librerías/funcionalidades en el sistema operativo: paquete *csh*, paquete *tcsh*, librería *libqt3-mt:i386*.

Una vez que hemos solucionado estos problemas de compatibilidad debemos seguir las instrucciones de instalación que encontraremos al descomprimir el archivo descargado desde la web del autor (ism-feb08.tgz). En el archivo INSTALL.txt tenemos las instrucciones para poder ejecutar el programa y comenzar a manejar su entorno gráfico. Y que se resumen en Figura A.3.

```
The provided archive contains all executables and libraries
needed to
run the ISM detector code. However, when first installing

1.) Create a symbolic link from "~/code" to the provided "code"
subdirectory. This is required, so that the detector can find the
feature extraction binaries.

ln -s $PWD/code ~/code


2.) Download and install the provided detector codebooks from our
web
page (http://www.vision.ee.ethz.ch/bleibe/code/). It is
recommended
that all detector codebooks are installed in the same directory
"~/codebooks" (or some other similarly accessible directory).
Since
the detector files still contain some references to absolute
paths, it
is necessary to run the provided script "prepare.sh" after
installing
a new batch of codebooks. This automatically adapts the stored
paths.

cd ~/codebooks
./prepare.sh


3.) Now, the detector binaries can be executed by running the
script
"./start.sh".
```

Figura A.3: Instrucciones de instalación ISM, extraído de INSTALL.txt.

En la siguiente imagen se puede ver el comando para, una vez instalado, arrancar la ejecución desde un terminal.

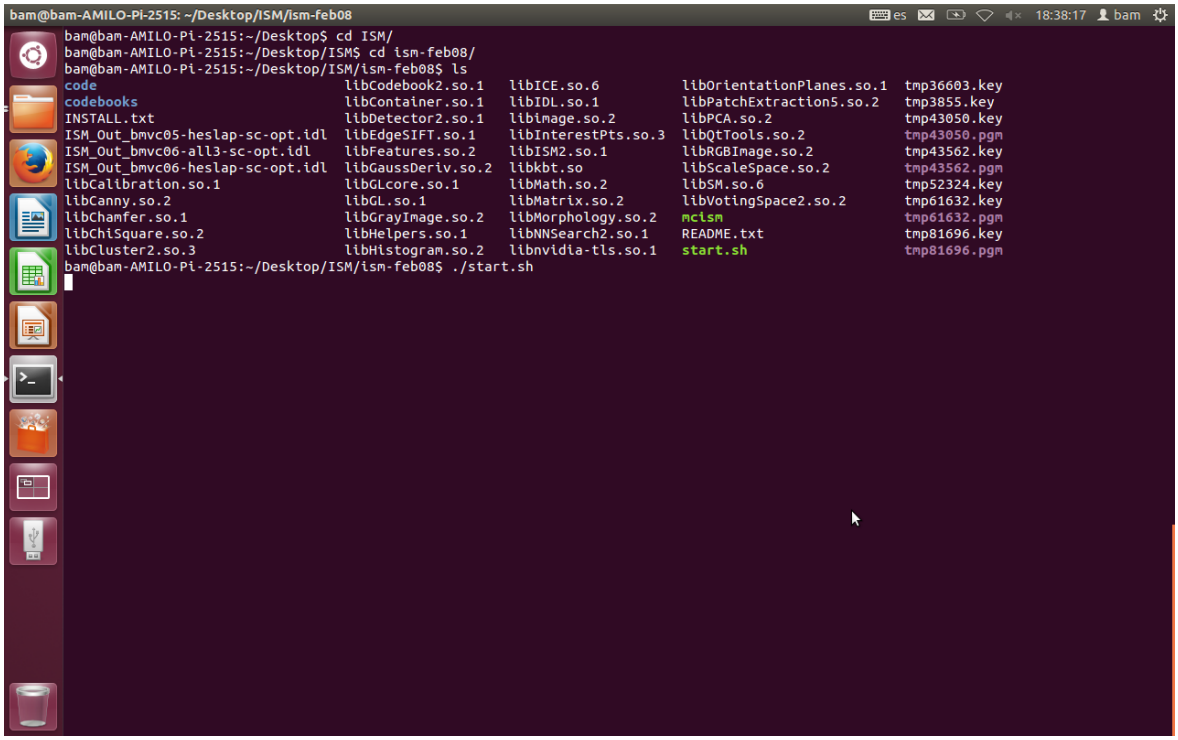


Figura A.4: Arranque de ISM desde consola.

Una vez arrancado nos encontramos con un entorno gráfico donde tendremos que seleccionar el detector que queremos utilizar, así como los parámetros del detector que queramos modificar para realizar el análisis, el archivo .idl que contiene las direcciones de cada uno de los *frames* del vídeo a analizar, el fichero de salida .idl donde guardar los resultados.

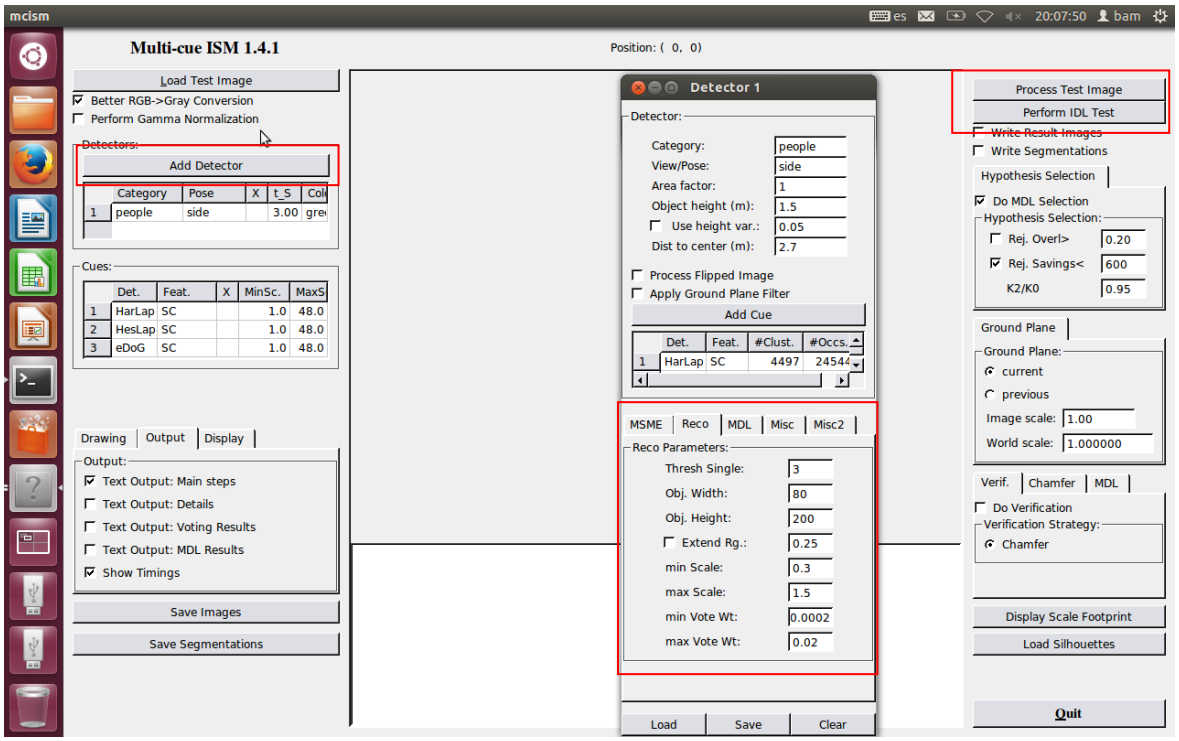


Figura A.5: Entorno gráfico de ISM donde realizar las acciones.

En la Figura A.5 podemos ver el entorno gráfico, en la columna de la izquierda en rojo seleccionaremos *Add detector*. A continuación nos aparecerá una ventana en el centro de la pantalla donde podremos generar un detector, cargar uno predefinido, modificar uno que ya tengamos y grabarlo con los nuevos parámetros, en el centro de la imagen, señalado en rojo, observamos los parámetros que pueden ser modificados para adaptar los detectores a los parámetros escogidos. Para terminar, en la parte derecha observamos el espacio para la carga de una imagen individual o de un archivo que contenga todos los *frames* para realizar el análisis, al pulsar aquí seleccionaremos el origen de datos y el archivo donde guardar los mismos. Mientras el algoritmo se está ejecutando nos muestra en tiempo real las detecciones, mientras que en el terminal podemos los tiempos empleados por el algoritmo para cada una de sus fases. La Figura A.6 muestra la ejecución en un terminal.

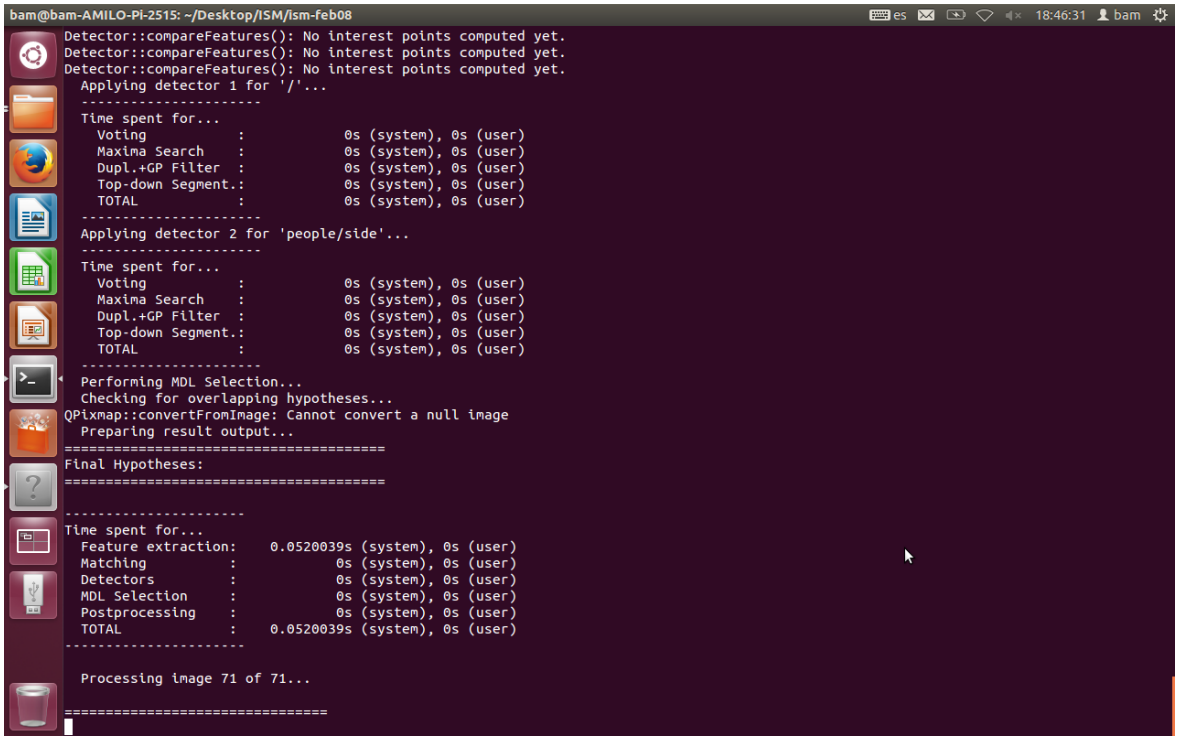


Figura A.6: Resultados de ejecución de ISM en consola.

A continuación, Figura A.7, se muestra una captura de ejemplo del fichero de salida .idl del detector donde se guardan las detecciones. En el mismo podemos observar que los valores que aparecen son:

- "Imagen/*frame* donde se ha realizado las detección/iones".
- : (Coordenada X, Coordenada Y, Ancho, Alto).
- : Puntuación obtenida.
- , siguiente detección.

```
"/home/gti/Desktop/dataset/abandonedBox/in000002.jpg": (19, 26,
62, 132):441.646, (247, -20, 324, 173):413.727, (102, -5, 161,
143):391.932, (162, 58, 196, 142):375.492, (320, 18, 384,
178):313.068, (63, 53, 121, 199):312.389;
"/home/gti/Desktop/dataset/abandonedBox/in000003.jpg": (247, -19,
323, 170):438.878, (14, 15, 64, 141):431.164, (101, -6, 161,
143):396.279, (159, 48, 200, 150):378.369, (319, 16, 384,
179):322.637;
"/home/gti/Desktop/dataset/abandonedBox/in000004.jpg": (288, -3,
364, 187):432.187, (19, 24, 62, 132):407.459, (102, -6, 161,
142):392.253, (193, 1, 259, 166):381.522;
"/home/gti/Desktop/dataset/abandonedBox/in000005.jpg": (26, 71,
64, 166):423.362, (101, -6, 161, 143):396.375, (224, -1, 289,
161):393.988, (160, 47, 201, 149):325.261;
"/home/gti/Desktop/dataset/abandonedBox/in000006.jpg": (18, 25,
61, 132):450.537, (234, -22, 313, 174):428.838, (102, -5, 161,
143):378.989, (156, 30, 206, 156):337.878;
"/home/gti/Desktop/dataset/abandonedBox/in000007.jpg": (13, 16,
63, 141):436.937, (224, -3, 289, 161):390.949, (102, -5, 162,
144):382.004, (161, 51, 200, 149):363.576, (321, 19, 385,
178):321.86;
"/home/gti/Desktop/dataset/abandonedBox/in000008.jpg": (246, -22,
323, 170):427.885, (19, 25, 62, 132):427.393, (101, -6, 161,
143):383.641, (158, 46, 200, 150):349.815;
"/home/gti/Desktop/dataset/abandonedBox/in000009.jpg": (19, 25,
62, 132):443.943, (248, -22, 327, 176):421.769, (102, -5, 162,
144):387.995, (160, 52, 199, 149):359.842,
```

Figura A.7: Fichero de salida .idl de ejecución de ISM.

DTDP

Para poder ejecutar este modelo descargaremos de la web del autor (http://www.cs.berkeley.edu/~rbg/latent/) el toolbox para Matlab y entorno Windows. Una vez descargado el toolbox hemos desarrollado un programa y una función en Matlab para que realicen la llamada a la función desarrollada por el autor y que se encarga de arrancar el detector.

Para poder ejecutar este algoritmo en Matlab, y debido a la utilización de funciones de C/C++ para lograr una mayor velocidad de procesado, debe instalarse el Visual Studio y ejecutar en Matlab la secuencia mex –setup para seleccionar compilador. Una vez que ya está vinculado el compilador, debemos ejecutar el fichero compile dentro de la carpeta voc-release4.01 del autor. Con esto ya podríamos ejecutar este algoritmo.

En primer lugar tenemos test_DTDP_detector.m, lo primero que debemos hacer es añadir el *path* donde tenemos guardado el *toolbox* del autor, tras esto, fijaremos tanto el vídeo a analizar como el modelo que queremos procesar. El último paso será llamar a la otra función desarrollada, DTDP_detector.m. El código puede verse en la Figura A.8. La Figura A.9 muestra cómo se modifica la llamada por defecto para agregar los parámetros que queremos modificar.

```
video_dir='./Videos';
threshold=-1.5;
video_names={'cubicle','fall','office','overpass','pedestrians','peopleInShade','PETS2006','sidewalk','sofa','tramstop','winterDriveway'};

for i=1:size(video_names,2)
video_names{i}
    %lista de imagenes
    video_list=sprintf('%s/%s/DataIn.txt',video_dir,video_names{i});
    fid=fopen(video_list,'r');
    num_images=0;
    cadena=fgets(fid);
    images_names={};
    while(size(cadena,2)>4)
        num_images=num_images+1;
        index=find((cadena==' ' | cadena==char(13))==1);
        if(~isempty(index))
            images_names{num_images}=cadena(1:index(1)-1);
        else
            images_names{num_images}=cadena;
        end
        cadena=fgets(fid);

    end
    fclose(fid);
    DTDP_detector(images_names,threshold,video_names{i},video_dir,4,5);
    DTDP_detector(images_names,threshold,video_names{i},video_dir,4,10);
```

Figura A.8: Llamada a la función con los parámetros que queremos modificar sobre el modelo inicial.

A continuación el código de la función donde se fijarán los valores de los parámetros a modificar y tras esto llama al detector del autor.

```
function DTDP_detector(images_names,threshold,video,video_dir,sbin,int)
addpath('./voc-release4.01');
load('./voc-release4.01/INRIA/inriaperson_final.mat');
model.sbin = sbin;
model.interval = int;
for i=1:size(images_names,2)
    cadena=sprintf('%s/%s',video_dir,images_names{i});
    test(cadena, model,images_names{i},threshold,video,video_dir,i,sbin,int);
end
clear model

function test(name, model,image_name,threshold,video,video_dir,num_frame,sbin,int)
im = imread(name);
% detect objects
[dets, boxes] = imgdetect(im, model, threshold);
top = nms(dets, 0.5);
% get bounding boxes
if(~isempty(top))
    bbox = bboxpred_get(model.bboxpred, dets, reduceboxes(model, boxes));
    bbox = clipboxes(im, bbox);
    top = nms(bbox, 0.5);
    final_blobs=bbox(top,:);
else
    final_blobs=[];
end
out_filename=sprintf('SALIDA/DTDP DEFINITIVOS/%s/%s_dtdp_sbin%d_int%d_%.2f.idl',video,video,sbin,int,threshold);
if(num_frame==1)
    fid=fopen(out_filename,'w+');
    fclose(fid);
end
```

Figura A.9: DTDP_detector.m con modificación de los parámetros elegidos.

A continuación, en la Figura A.10, se muestra una captura de ejemplo del fichero de salida .idl del detector donde se guardan las detecciones. En el mismo podemos observar que los valores que aparecen son:

- "Imagen/*frame* donde se ha realizado las detección/iones".
- ; (Coordenada X, Coordenada Y, Ancho, Alto).
- : Puntuación obtenida.
- ; siguiente detección.

```
"./Videos/abandonedBox/in000010.jpg";
"./Videos/abandonedBox/in000011.jpg";
"./Videos/abandonedBox/in000012.jpg";
"./Videos/abandonedBox/in000013.jpg"; (109, 9, 155, 146):-1.4707;
"./Videos/abandonedBox/in000014.jpg";
"./Videos/abandonedBox/in000015.jpg";
"./Videos/abandonedBox/in000016.jpg";
"./Videos/abandonedBox/in000017.jpg";
"./Videos/abandonedBox/in000018.jpg";
"./Videos/abandonedBox/in000019.jpg"; (30, 10, 71, 124):-1.4043,
(109, 10, 156, 143):-1.4521;
"./Videos/abandonedBox/in000020.jpg";
"./Videos/abandonedBox/in000021.jpg";
"./Videos/abandonedBox/in000022.jpg";
"./Videos/abandonedBox/in000023.jpg"; (30, 11, 72, 123):-1.4393;
"./Videos/abandonedBox/in000024.jpg"; (30, 11, 72, 123):-1.4615;
"./Videos/abandonedBox/in000025.jpg";
"./Videos/abandonedBox/in000026.jpg";
"./Videos/abandonedBox/in000027.jpg";
"./Videos/abandonedBox/in000028.jpg"; (30, 11, 72, 123):-1.3083;
"./Videos/abandonedBox/in000029.jpg";
"./Videos/abandonedBox/in000030.jpg";
"./Videos/abandonedBox/in000031.jpg"; (33, 12, 71, 122):-1.4497;
"./Videos/abandonedBox/in000032.jpg";
"./Videos/abandonedBox/in000033.jpg";
"./Videos/abandonedBox/in000034.jpg"; (42, 45, 110, 239):-1.4766;
"./Videos/abandonedBox/in000035.jpg"; (31, 10, 72, 122):-1.4967;
"./Videos/abandonedBox/in000036.jpg";
```

Figura A.10: Ejemplo de salida del algoritmo DTDP.

FUSION y EDGE

Para ejecutar los algoritmos Fusion y Edge, del VPULab, hemos utilizado la plataforma DiVa. El trabajo [38] integró ambos algoritmos en la plataforma permitiendo además su parametrización. La Figura A.11 muestra la plataforma en funcionamiento. En cuanto al código de ambos algoritmos difieren en el modelo de persona utilizado, en un caso holístico y en el otro por partes (cabeza, cuerpo, torso, piernas). Esto se traduce en las diferencias de código que pueden apreciarse en la Figura A.12 y Figura A.13.

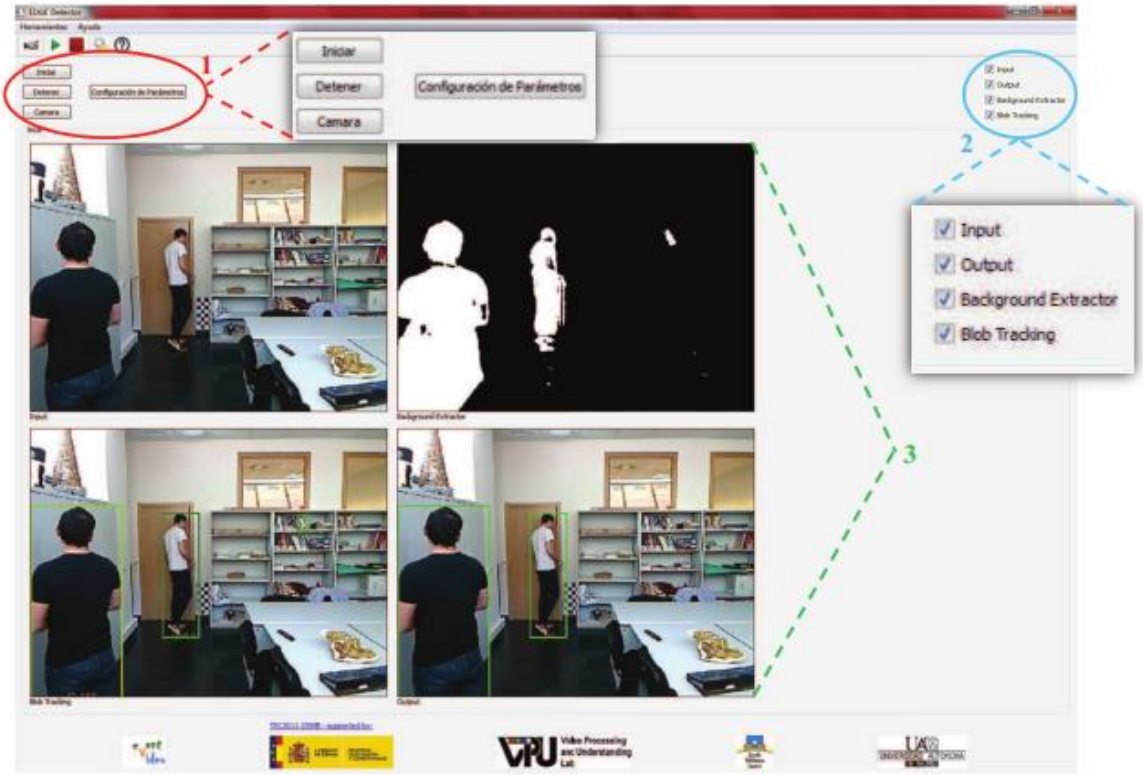


Figura A.11: Plataforma de ejecución DiVa, figura extraída de [38].

```
if (Fusion_Edge_alg == NULL) {
        Fusion_Edge_alg = new FusionEdgePeopleDetector(directorio_edge_detector,N);
        Fusion_Edge_alg->setHeadAlg(Edge_Head_alg);
        Fusion_Edge_alg->setBodyAlg(Edge_Body_alg);
        Fusion_Edge_alg->setTorsoAlg(Edge_Torso_alg);
        Fusion_Edge_alg->setLegsAlg(Edge_Legs_alg);
        Fusion_Edge_alg->setListBlob(new BlobList<PeopleBlob*>);
```

Figura A.12: Extracto de código de algoritmo *Fusion*, holístico.

```
if (Edge_Head_alg == NULL) {
        Edge_Head_alg = new EdgePeopleDetector(directorio_edge_detector,2);
        Edge_Head_alg->setListBlob(new BlobList<PeopleBlob*>);
}
if (Edge_Body_alg == NULL) {
        Edge_Body_alg = new EdgePeopleDetector(directorio_edge_detector,1);
        Edge_Body_alg->setListBlob(new BlobList<PeopleBlob*>);
}
if (Edge_Torso_alg == NULL) {
        Edge_Torso_alg = new EdgePeopleDetector(directorio_edge_detector,3);
        Edge_Torso_alg->setListBlob(new BlobList<PeopleBlob*>);
}
if (Edge_Legs_alg == NULL) {
        Edge_Legs_alg = new EdgePeopleDetector(directorio_edge_detector,4);
        Edge_Legs_alg->setListBlob(new BlobList<PeopleBlob*>);
}
```

Figura A.13: Extracto de código del algoritmo Edge, modelo basado en partes.

A continuación se puede observar los valores del fichero de salida y los separadores que utilizan:

- "Imagen/*frame* donde se ha realizado las detección/iones".
- : (Coordenada X, Coordenada Y, Ancho, Alto).
- : Puntuación obtenida.
- ; siguiente detección.

ACF

Para poder ejecutar este modelo descargaremos de la web del autor (http://vision.ucsd.edu/~pdollar/toolbox/doc/index.html) el toolbox para Matlab y entorno Windows. Una vez descargado el toolbox hemos desarrollado un programa y una función en Matlab para que realicen la llamada a la función desarrollada por el autor y que se encarga de arrancar el detector.

En primer lugar tenemos test_ACF_detector.m, lo primero que debemos hacer es añadir el *path* donde tenemos guardado el *toolbox* del autor, tras esto, fijaremos tanto el vídeo a analizar como el modelo que queremos procesar. El último paso será llamar a la otra función desarrollada, ACF_Detector.m. El código puede verse en Figura A.14. La Figura A.15 muestra cómo se modifica la llamada por defecto para agregar los parámetros que queremos modificar.

```
addpath(genpath('./toolbox')); savepath;
% toolboxCompile
video_dir='./Videos';
video_names={'train1'};

model_filename='./toolbox/detector/models/AcfCaltechDetector.mat';
model_id='Caltech';

for i=1:size(video_names,2)

video_names{i}
    video_list=sprintf('%s/%s/%slist.txt',video_dir,video_names{i},video_names{i});
    fid=fopen(video_list,'r');
    num_images=0;
    cadena=fgets(fid);
    images_names={};
    while(size(cadena,2)>4)
        num_images=num_images+1;
        index=find((cadena==' ' | cadena==char(13))==1);
        if(~isempty(index))
            images_names{num_images}=cadena(1:index(1)-1);
        else
            images_names{num_images}=cadena;
        end
        cadena=fgets(fid);
    end
    fclose(fid);
    ACF_detector(images_names,video_names{i},video_dir,model_filename,model_id);
end
```

Figura A.14: Código del programa de MATLAB test_ACF_detector.m.

```
ACF_detector(images_names,video_names{i},video_dir,model_filename,model_id,4,-1);
```

Figura A.15: Llamada a la función con modificación de parámetros de entrada.

El siguiente programa será el encargado de llamar al detector y es el que aprovecharemos para fijar las opciones que hemos elegido, asignando los valores que pasaremos como parámetros de entrada cuando realizamos la llamada a la función desde test_ACF_detector.m.

La Figura A.16 muestra tanto la función como la fijación de los valores de configuración para el detector.

```
function ACF_detector(images_names,video,video_dir,model_filename,model_id,PerOct,OctUp)

for i=1:size(images_names,2)
    cadena=sprintf('%s/%s/%s',video_dir,video,images_names{i});
    test(cadena, model_filename,model_id,images_names{i},video,video_dir,i,PerOct,OctUp);
end
clear model


function test(name, model_filename,model_id,image_name,video,video_dir,num_frame,PerOct,OctUp)

load(model_filename);
detector.opts.pPyramid.nPerOct=PerOct;
detector.opts.pPyramid.nOctUp=OctUp;
detector.opts.cascThr=-10;
I = imread(name);
% detect objects
bbs = acfDetect( I, detector);

out_filename=sprintf('SALIDA/ACF DEFINITIVOS/%s/%s_acf_%s_nPerOct%d_nOctUp%.1f.idl',video,video,model_id,PerOct,OctUp);
if(num_frame==1)
    fid=fopen(out_filename,'w+');
    fclose(fid);
end

save_blobs(bbs,name,out_filename);
clear top clear bbox
clear final_blobs
```

Figura A.16: Código de la función que llama al detector del autor y que fija los parámetros modificados, función ACF_detector.m.

A continuación, en la Figura A.17, se muestra una captura de ejemplo del fichero de salida .idl del detector donde se guardan las detecciones. En el mismo podemos observar que los valores que aparecen son:

- "Imagen/*frame* donde se ha realizado las detección/iones".
- ; (Coordenada X, Coordenada Y, Ancho, Alto).
- : Puntuación obtenida.
- , siguiente detección.

```
100):-8.2013, (124, 62, 41, 100):-8.3169, (356, 190, 41,
100):-8.4468, (108, 62, 41, 100):-8.9397, (68, -2, 41,
100):-9.7834;
"../dataset/abandonedBox/in000002.jpg"; (84, -2, 41, 100):8.8063,
(4, 134, 41, 100):8.3688, (-9, -4, 82, 200):6.6743, (348, -2, 41,
100):4.9709, (92, 38, 41, 100):0.8038, (316, 150, 41,
100):0.4690, (-5, 170, 49, 120):0.4209, (388, -2, 41,
100):-0.0799, (364, -2, 41, 100):-0.3099, (148, 30, 41,
100):-1.0897, (23, -4, 82, 200):-1.2446, (292, -2, 41,
100):-3.1643, (84, 62, 41, 100):-5.1802, (332, 142, 41,
100):-5.3867, (116, 38, 41, 100):-5.6160, (228, 22, 41,
100):-5.7118, (124, 62, 41, 100):-7.0476, (388, 158, 41,
100):-7.1910, (100, -2, 41, 100):-7.8374, (116, -2, 41,
100):-9.4356, (108, 62, 41, 100):-9.5058;
"../dataset/abandonedBox/in000003.jpg"; (84, -2, 41, 100):7.7620,
(-5, 113, 49, 120):7.3127, (356, -2, 41, 100):3.3438, (28, 22,
41, 100):2.7417, (156, 38, 41, 100):0.4832, (20, -2, 41,
100):-0.6038, (20, 70, 41, 100):-0.6429, (388, -2, 41,
100):-0.6439, (92, 38, 41, 100):-0.8290, (116, 38, 41,
100):-1.0585, (68, 6, 41, 100):-3.0114, (292, -2, 41,
100):-3.3867, (52, 94, 41, 100):-3.5926, (-5, 161, 49,
120):-3.6332, (84, 62, 41, 100):-4.1399, (316, 142, 41,
100):-4.6753, (388, 158, 41, 100):-4.8822, (332, 142, 41,
100):-5.1789, (20, 110, 41, 100):-6.3376, (-5, 6, 41,
100):-6.7863, (68, 62, 41, 100):-7.0269, (124, 62, 41,
100):-8.1709, (4, 70, 41, 100):-8.2799, (100, -2, 41,
100):-8.3547, (228, 22, 41, 100):-8.8510, (108, 62, 41,
100):-9.3544;
"../dataset/abandonedBox/in000004.jpg"; (84, -2, 41,
100):14.4157, (-5, 113, 49, 120):8.7541, (28, 22, 41,
100):5.3317, (156, 38, 41, 100):0.8278, (-5, 170, 49
```

Figura A.17: Ejemplo de salida del algoritmo ACF.

B. Ejemplos de fotogramas de los vídeos utilizados

En este anexo mostramos un ejemplo de *frame* de cada uno de los vídeos analizados y que contenga una persona para poder ayudar a su visualización, entender el entorno, el enfoque de la cámara la definición o el ruido de la imagen, etc., así como para poder entender la dificultad de las diferentes fases en la detección de personas.

Figura A.18: Frame del vídeo abandonedBox.avi.

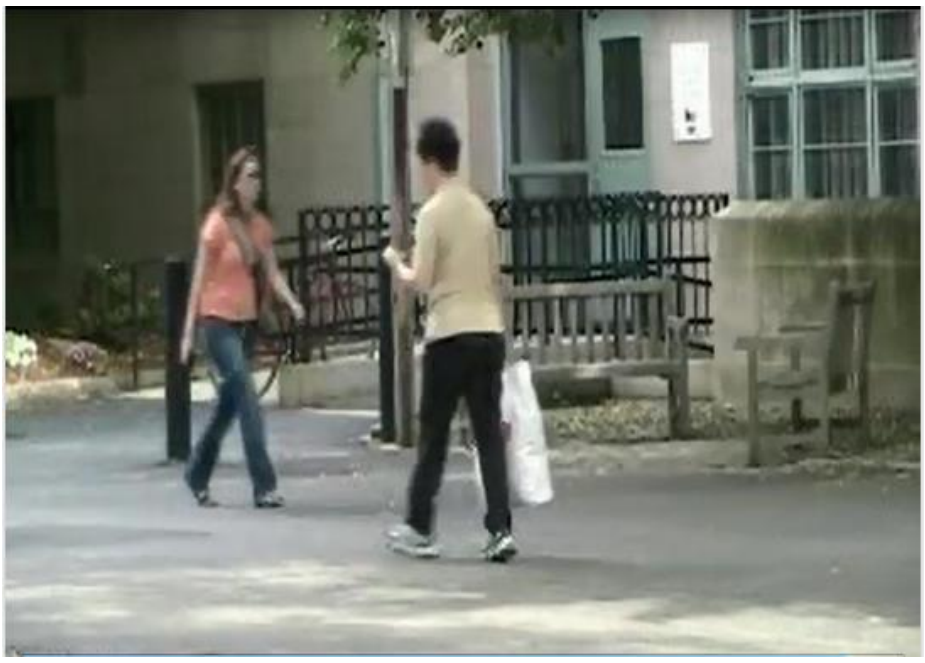


Figura A.19*: Frame* del vídeo backdoor.avi.

Figura A.20: *Frame* del vídeo badminton.avi.

Figura A.21: *Frame* del vídeo busStation.avi.

Figura A.22: *Frame* del vídeo copyMachine.avi.

Figura A.23: *Frame* del vídeo cubicle.avi.

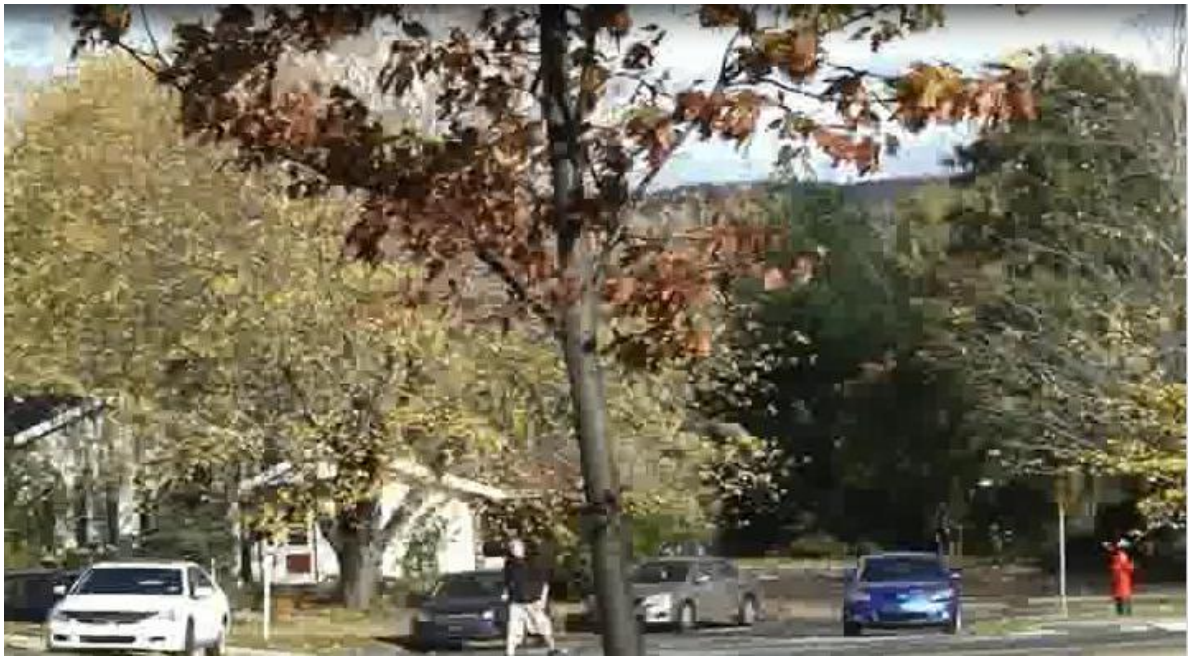


Figura A.24: *Frame* del vídeo fall.avi.

Figura A.25: *Frame* del vídeo office.avi.

Figura A.26: *Frame* del vídeo overpass.avi.

Figura A.27: *Frame* del vídeo pedestrians.avi.

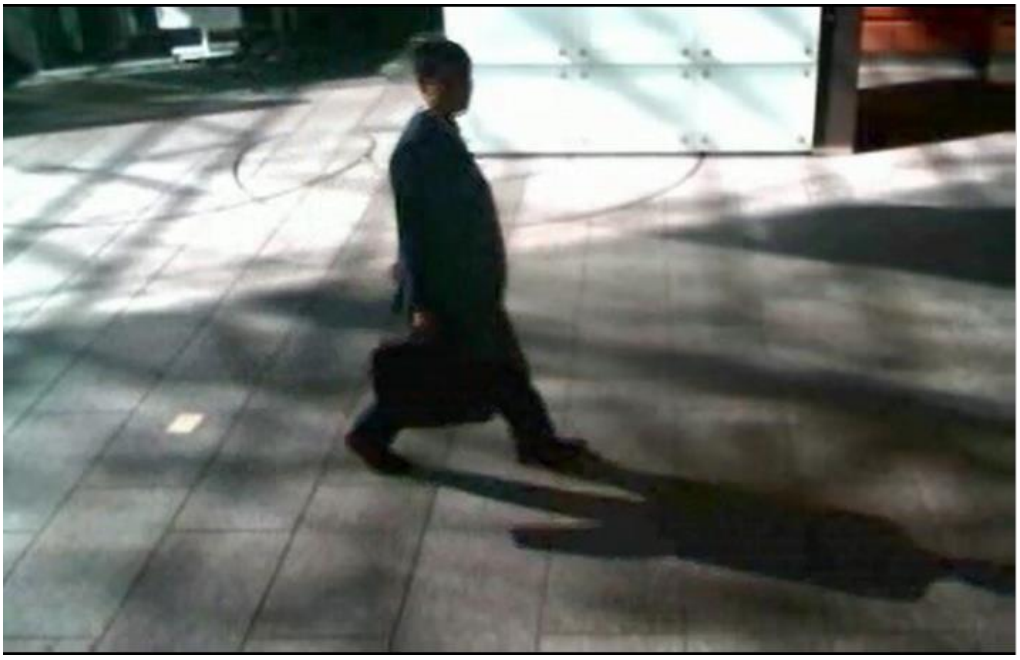


Figura A.28: *Frame* del vídeo peopleInShade.avi.

Figura A.29: *Frame* del vídeo PETS2006.avi.

Figura A.30: *Frame* del vídeo sidewalk.avi.

Figura A.31: *Frame* del vídeo sofa.avi.

Figura A.32: *Frame* del vídeo tramstop.avi.

Figura A.33: *Frame* del vídeo winterdriveway.avi.

PRESUPUESTO

1) Ejecución Material

- Compra de ordenador personal (Software incluido)..... ..... 685 €
- Alquiler de impresora láser durante 6 meses ..... 50 €
- Soportes de almacenamiento de datos y back-up..... 50 €
- Material de oficina ..... 100 €
- Total de ejecución material ..... 885 €

2) Gastos generales

- 21 % sobre Ejecución Material ..... 185.85 €

3) Beneficio Industrial

- 6 % sobre Ejecución Material ..... 53.1 €

4) Honorarios Proyecto

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

5) Material fungible

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

6) Subtotal del presupuesto

- Subtotal Presupuesto..... 10745 €

7) I.V.A. aplicable

- 21% Subtotal Presupuesto ..... 2256.45 €

8) Total presupuesto

- Total Presupuesto..... 13001,45 €

Madrid, Julio de 2014

El Ingeniero Jefe de Proyecto

Fdo.: Borja Alcedo Moreno
Ingeniero de Telecomunicación

PLIEGO DE CONDICIONES

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

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

Condiciones generales

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

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

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

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

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

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

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

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

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

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

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

12. Las cantidades calculadas para obras accesorias, aunque figuren por partida alzada en el presupuesto final (general), no serán abonadas sino a los precios de la contrata, según las condiciones de la misma y los proyectos particulares que para ellas se formen, o en su defecto, por lo que resulte de su medición final.

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

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

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

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

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

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

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

20. Durante la realización de la obra, se girarán visitas de inspección por personal facultativo de la empresa cliente, para hacer las comprobaciones que se crean oportunas. Es

obligación del contratista, la conservación de la obra ya ejecutada hasta la recepción de la misma, por lo que el deterioro parcial o total de ella, aunque sea por agentes atmosféricos u otras causas, deberá ser reparado o reconstruido por su cuenta.

21. El contratista, deberá realizar la obra en el plazo mencionado a partir de la fecha del contrato, incurriendo en multa, por retraso de la ejecución siempre que éste no sea debido a causas de fuerza mayor. A la terminación de la obra, se hará una recepción provisional previo reconocimiento y examen por la dirección técnica, el depositario de efectos, el interventor y el jefe de servicio o un representante, estampando su conformidad el contratista.

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

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

Condiciones particulares

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

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

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

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

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

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

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

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

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

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

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

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

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