Universidad
Carlos III de Madrid

Trabajo de Fin de Grado

Detección de incidentes basada en flujos de movimiento en cámaras de la DGT

Jose Antonio Peláez Escobar
Grado en Ingeniería de Sistemas Audiovisuales

Tutor:
Iván González Díaz

Índice

Abstract

Every year, thousands of accidents occur in the roads. Thanks to the effort and the investment of different institutions, this number has been reduced year after year, but it is not enough yet. To reduce the number of mortal victims, it is crucial to react as quick as possible, so that wounded can receive the needed cares in the briefest period of time.

In this project, it is presented an automatic incident detection system that, installed on DGT (*Dirección General de Tráfico*), would allow to decrease the response time in case of accident. Moreover, this system informs about other anomalous situations like traffic jams or vehicles moving in forbidden directions that must also be taken into account in road surveillance. To achieve this functioning, the usual vehicle flow in the road is statistically modelled making use of video motion estimation. Then, if a group of motion vectors presents abnormal values and satisfies some spatial and temporal conditions, it can be considered that an anomaly is taking place on the scene.

The evaluation is performed over a database of videos provided by the DGT, containing three accidents and some slow traffic situations. The system is able to properly detect all of the accidents, but the performance decreases slightly when detecting traffic jams. Nevertheless, obtained results are more robust than the ones from the reference system based on vehicle tracking.

1. Introduction

1.1. Socio-economic context and project motivation

One of the main goals of the State, by means of its DGT (*Dirección General de Tráfico*) department, and some other private organizations like Mapfre or Grupo Atresmedia, is to reduce year by year the number of mortal victims on traffic accidents. According to [1], and as it can be observed on Figure 1.1., this objective has been accomplished since 2003. At that time, the number of deceased people on the road was about 4000 people. In 2015, the number was 1126. In this progress, video surveillance in road environments has played a very important role, allowing faster responses to traffic incidents. Nevertheless, this is a high load task for humans, so the response is not as fast as it could be.

As time goes on and technology is improved, finding powerful computers able to deal with tedious tasks traditionally performed by humans is more and more habitual. This progress allows that people can focus on high-level tasks, letting machines to carry out the least desirable ones. Moreover, ICTs environment is evolving towards a period in which a big amount of data (Big Data) is daily processed, so that performing this task without computers' help would become unfeasible at all.

It is also worth considering the current global economic situation as well as the Spanish

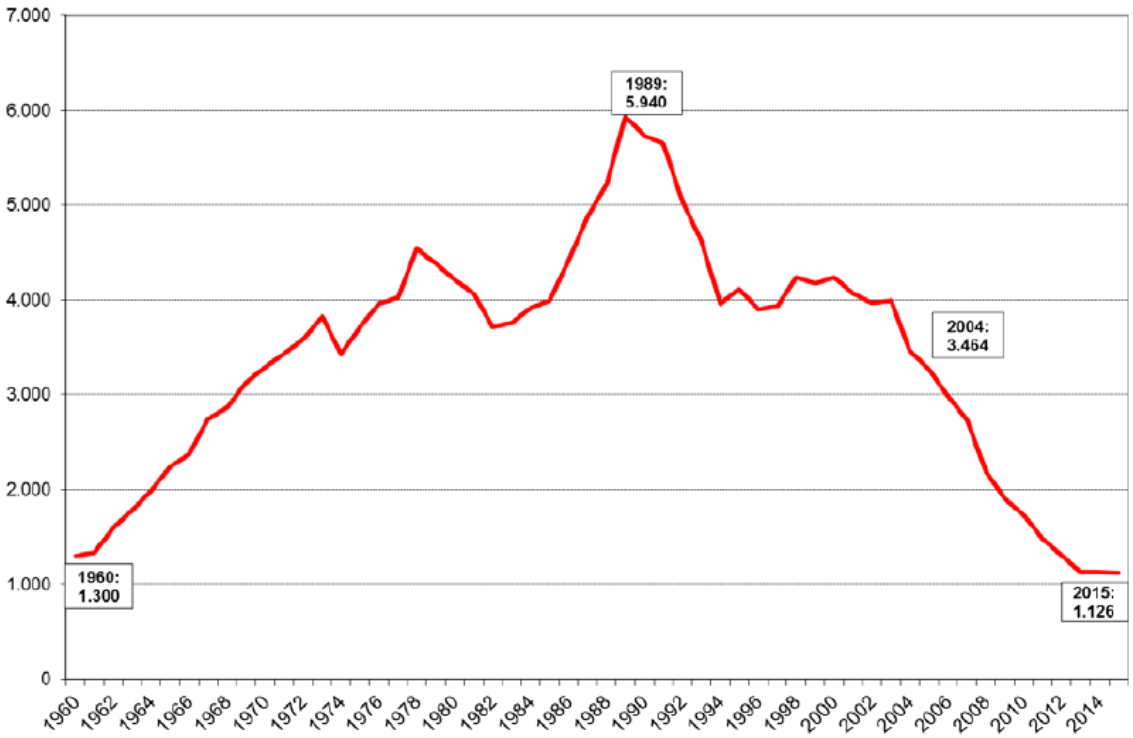

Fig. 1.1: evolution of deceased people on roads from 1960 to 2015. Extracted from [1].

particular context. It seems that the financial crisis is finally coming to its end, so this should allow a bigger investment on all these technologies. These investments will hopefully bring important savings for big companies in the near future, since, as it is said before, people in charge of most tedious tasks could be reallocated in more useful positions.

In this context, the Multimedia Processing Group (known by its Spanish acronym, GPM) of Carlos III University developed the project *Automatic traffic incidents detection on DGT cameras*. The main goal of the project was to design a complete system that, working on current DGT traffic cameras, is capable of automatically detecting traffic incidents in real-time. As an additional but key requirement, the cameras can be moved at any time by a human camera operator, so the system was also required to update its state and get adapted to these changing conditions.

This previous version of the system based the detection of traffic incidents on the analysis of vehicle trajectories. The system first makes use of particle filters to track vehicles, then learns the usual trajectories, and finally detects anomalies by identifying those trajectories that differ from the learnt ones. Two are the main limitations of the current system: first, this system presents a reduced performance on high traffic cases, in which trackers either follow groups of vehicles (as in traffic jams), or "jump" from one vehicle to another (as shown in Figure 1.2). Second, the computational cost due to the concurrent use of many particle filters (one for each vehicle in the scene) is highly demanding, which prevents its implementation in a real environment. The latter issue, although might be mitigated by the use of parallel architectures (e.g. GPUs), represents the main computational bottleneck of the system. In this project, a new alternative solution for the detection of traffic incidents in road environments is proposed. Instead of tracking individual vehicles, this work is oriented towards a statistical modelling of vehicle flows based on video motion estimation. Our goal is to obtain similar detection performances than in the previous version of the system, but solving both problems of low precision in high traffic situations and high computational cost (as motion estimation is much lighter in processing terms than the particle filter).

1.2. Regulatory framework

Regarding legal aspects that may affect the scope of this project, two different issues should be taken into account: on one side, the fact that the national government, by means of the DGT, is permanently recording everything that happens on the road; on the other side, the processing that can be performed over these data.

Fig. 1.2: high traffic situation in which the tracker “jumps” from a motorbike to a car.

The Organic Law 17574 deals with all these issues. It states that, in order to prevent criminal acts and achieve people protection in public places with a higher efficiency, it is legitimate employing image and sound recording devices and performing some processing over the obtained material. Nevertheless, this material must be destroyed one month after its recording, except if the video footage is related to serious criminal or administrative offences in matters of public safety, to an ongoing police investigation or to an initiated judicial procedure. Moreover, people must be informed of the existence of these fixed video cameras and have the right to access and cancel the images where they appear.

In addition, for the particular purpose of this project, and the one previously developed by the Multimedia Processing Group of Carlos III University, a particular Non-Disclosure Agreement (NDA) was signed. On it, the DGT cedes a database of videos to Carlos III University for the development of an automatic incident detection system under certain conditions of confidentiality.

1.3. Project objectives

The aim of the project is to develop a system for automatic incident detection which, using traffic surveillance video cameras, is based on the modeling of the motion flow. For that end, we will build up our proposal over the previously developed system by the GPM, and incorporate the new algorithms concerning the modelling and analysis of the motion flow. Our goal is to achieve similar performances than in the previous version of the system (the one based on motion trajectories), but notably reducing the computational cost, thus allowing its real-time implementation in real environments.

Performing a deeper analysis of the project, the following particular objectives can be also considered:

- To develop a motion estimation system appropriated for vehicle surveillance and incident detection.
- To identify the areas of interest in the road, candidate for later analysis.
- To develop a statistical model that allows a proper representation of motion flows in roads.
- To establish a criterion to select the most appropriate statistical model for each situation.
- To develop online learning mechanisms which adapt to camera movements performed by human operators.
- To develop a detection system capable of identifying anomalous trajectories and slow or stopped vehicles.
- To integrate the new developed modules with the previous version of the automatic anomaly detection based on trajectories system.
- To evaluate this new system by means of the videos provided by the DGT in terms of both detection performance and computational complexity (for real-time implementations).

To accomplish this last objective, a database with videos provided by the DGT has been employed in our experiments. In these videos, the events labeled as incidents can be categorized in one of the next three types:

- Slow traffic or traffic jams, and, in some cases, fully stopped vehicles.
- Vehicles collision, both with several vehicles moving and with a vehicle crashing with a stopped one.
- Animals crossing the road, causing vehicles stopping and crashes.

The detection of these incidents requires from three different phases:

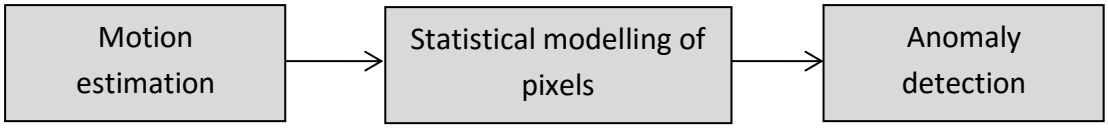

Fig. 1.3: designed phases of the project to perform anomaly detection

- Motion estimation: motion vectors of the image are obtained, but only in those parts where vehicles are detected.
- Statistical modelling of pixels: a two-dimensional Gaussian model is obtained for every pixel from estimated motion vectors.
- Anomaly detection: for new estimated motion vectors, *a posteriori* probabilities are evaluated in every pixel and it is decided whether it is an anomaly or not.

Taking all this into account, the final purpose is that the system should be capable of triggering an alarm in an autonomous way in case anomalies are detected on the video.

1.4. Methodology

In this project, a classic methodology has been followed. In Figure 1.4, it can be found a general approach to this methodology in a schematic form.

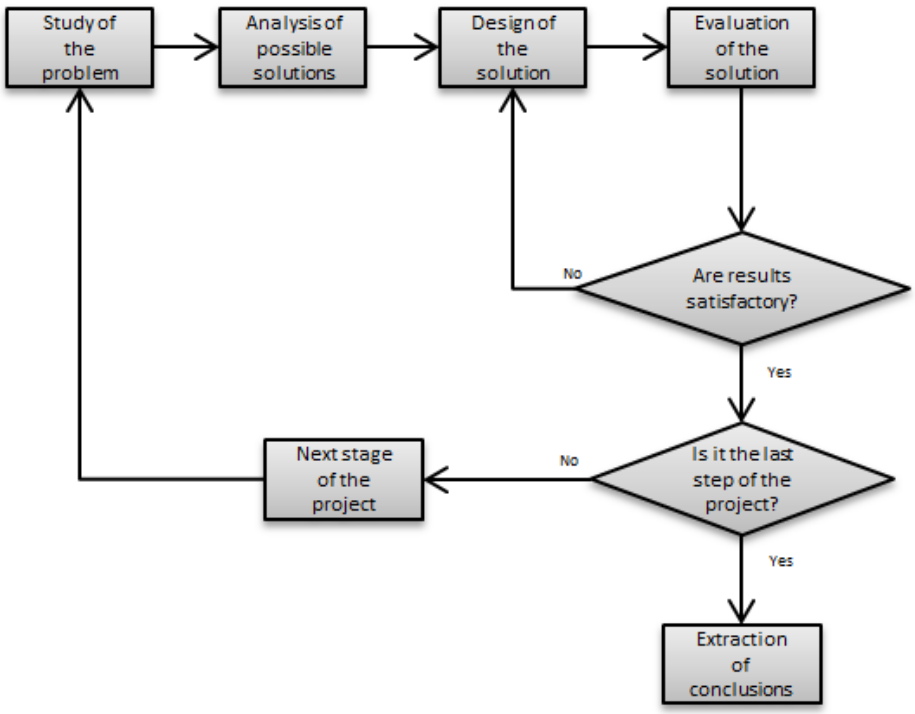

Fig. 1.4: general methodology applied during the development of the project.

In this approach, it is first performed a study and analysis of the problem that is being faced. Then, after analyzing theoretically pros and cons of different solutions, a first version of the selected option is designed and evaluated. In case the results are not as expected, the solution is re-designed and re-evaluated. This is repeated as long as obtained results are not optimal. After obtaining the desired results, the project moves to the next stage or module to be designed until the last stage is reached. Finally, conclusions derived from the work are extracted.

It is important to consider that the phases of design and evaluation only occur when dealing with original solutions of this project. In case some existing algorithms are employed, these phases can be omitted, since its proper functioning can be assumed.

1.5. Main contributions of the work

In the development of this project, part of the solutions have been taken from existing algorithms and part have been entirely designed. More precisely, the following implementations can be considered own contributions:

- A module for efficient storage of estimated motion vectors.
- An automatic decider for the number of components of a Gaussian Mixture Model, with a maximum of two components.
- A module for detection of anomalies based on estimated motion vectors and learnt Gaussian Mixture Models on every road pixel.

These sections have supposed the hardest stages of the project and have required the most part of the coding and computing time.

1.6. Report structure

This report is structured in six different chapters plus three annexes:

- Chapter 1: current socio-economic context is presented, as well as a brief project motivation and the regulatory framework. Objectives to be reached and memory structure are also included here.
- Chapter 2: it is performed an overview of literature related with anomaly detection systems up to now, making more emphasis on those papers that have influenced this project the most.

- Chapter 3: proposed solution is exposed in this chapter, detailing each of the modules. Mathematica of algorithms and base models used in the project is also detailed.
- Chapter 4: experiments performed are exposed here, explaining and reasoning the choice of the parameters. Limitations or restrictions of the system are also described.
- Chapter 5: conclusions derived from this work are drawn and possible future lines of investigation are proposed.
- Chapter 6: budget associated to the project is presented.
- Annex I: summary (in English) of the project report.
- Annex II: it is briefly explained an efficient module to store motion vectors that has been used during this project.
- Annex III: a more detailed explanation of EM algorithm is developed, making more emphasis on the detailed equations.

2. Estado del arte y Sistema Previo

2.1. Estado del arte y alternativas de diseño en la detección automática de incidentes de tráfico

Tomando como referencia el trabajo de Piciarelli [2], el análisis de eventos en vídeos puede abordarse desde dos perspectivas: el reconocimiento explícito de eventos y la detección de anomalías. El **reconocimiento explícito de eventos** implica que el sistema es capaz de aprender los diferentes tipos de eventos a los que va a ser expuesto. Una vez que el evento es detectado, se podrá etiquetar de manera adecuada entre alguna de las categorías consideradas. Para ello será necesario que el sistema disponga de una base de datos *a priori* que contenga toda la información sobre los eventos que es capaz de reconocer, de tal forma que el sistema pueda aprender de los eventos de entrenamiento disponibles y, durante su ejecución en test, clasifique un vídeo de entrada con el evento más adecuado (ver Figura 2.1).

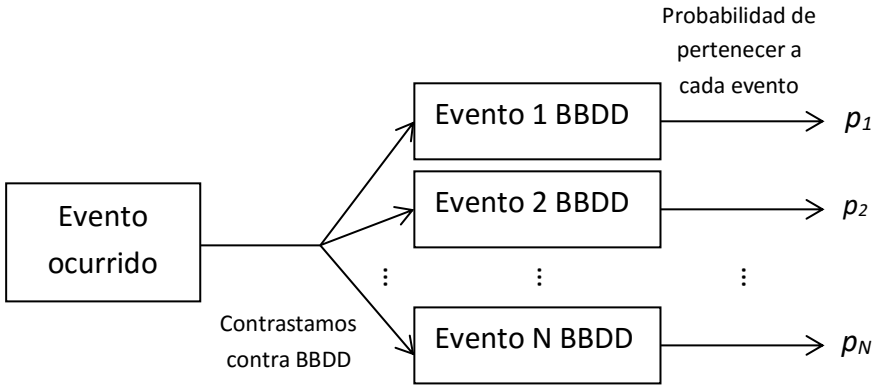

Fig. 2.1: representación simple del funcionamiento del reconocimiento explícito de eventos.

La gran ventaja de un sistema de este tipo es su amplia expresividad, puesto que permite modelar una serie de eventos heterogéneos etiquetándolos con descripciones semánticas de alto nivel, otorgando una interpretación más realista de la escena analizada. Sin embargo, presenta un gran inconveniente, y es que precisa un modelado previo para cada tipo de evento de interés. En el mundo real, en el que nos encontramos con un número infinito de posibles incidentes, el número de modelos crecería de forma exponencial a medida que un escenario se vuelve más complejo. Es por ello que algunos trabajos hacen uso de un subconjunto de eventos sujetos a una serie de condiciones lógicas y temporales, una base de datos modelada a partir de técnicas de aprendizaje máquina en lugar de una gramática definida manualmente, o la integración de conocimiento previo a bajo nivel de datos adquiridos a partir de procesado de vídeo, como puede ser la detección de cambios de escena o el seguimiento de objetos.

La otra posibilidad para el análisis de eventos es la **detección de anomalías**. Para este caso no se requiere un aprendizaje *a priori* de los eventos a reconocer, sino que el sistema construye modelos probabilísticos de los eventos mediante aprendizaje no supervisado a medida que recibe los datos. Este tipo de modelos permite detectar los patrones de actividad más frecuentes que ocurren en la escena, pero sin efectuar una distinción entre dichos patrones ni proporcionar una descripción detallada del evento en cuestión. Simplemente, mide la verosimilitud del evento ocurrido, siendo anomalías los casos más excepcionales o de menor verosimilitud.

Una de las aproximaciones más habituales para la detección de anomalías es el *clustering* (agrupación de vectores de datos según un criterio establecido) de trayectorias de los objetos en movimiento. Estos *clusters* se utilizan posteriormente como modelos de normalidad para detección de anomalías.

En el ámbito particular de la detección de incidentes de tráfico, algunos de los primeros trabajos de investigación, realizados por Johnson y Hogg, hacían uso de la cuantificación vectorial para la representación de las trayectorias y de redes neuronales multicapa para la identificación de patrones comunes. Posteriormente publicaron otro artículo en el que la identificación de patrones pasaba a modelarse a través de las f.d.p. (funciones de densidad de probabilidad) de una Mezcla de Gaussianas multivariante, actualizadas con el algoritmo EM [3]. Otras investigaciones llevadas a cabo en este ámbito hacen uso de matrices de co-ocurrencia de los vectores prototipo [4], de técnicas *clustering* dinámico para el análisis de tráfico [5] o de SVMs [2]. Todos estos trabajos centran sus esfuerzos en localizar vehículos concretos y analizar sus trayectorias, que, aún proporcionando unos resultados satisfactorios, supone una alta carga computacional que dificulta su implementación en tiempo real.

Buscando precisamente una reducción de este elevado tiempo de cómputo, cabe destacar el artículo de Robertson y Reid [6]. Este trabajo, aunque versa sobre la detección de actividad humana en secuencias de vídeo y no sobre las trayectorias de vehículos, sí que modela dicha actividad a partir del flujo óptico con el algoritmo de Lucas y Kanade [7] (además de con otros descriptores), al igual que en este proyecto. Este sistema tiene la ventaja de que la estimación de movimiento es un proceso menos costoso computacionalmente que el *tracking* de vehículos individuales (o, en este caso, personas), lo que facilita su implementación en tiempo real.

Asimismo, el trabajo de Morris y Trivedi [8][9] también aporta desde un punto de vista más global el modelado de escenas como autopistas a partir de flujos de movimiento,

entre otras soluciones, si bien dichos flujos se obtienen a partir de trayectorias de vehículos concretos, no de la escena en conjunto.

En [10] se presenta un trabajo similar al propuesto aquí, basado igualmente en flujos de movimiento estimados con Lucas y Kanade y haciendo uso de GMM en cámaras de baja calidad, pero se centra más en entornos urbanos y la Mezcla de Gaussianas la emplea únicamente para detectar las zonas estáticas, haciendo uso de modelos ocultos de Markov para la posterior detección de anomalías.

Sin embargo, es la investigación llevada a cabo en [11] la que más aporta al trabajo desarrollado a continuación. En dicho trabajo se propone una solución para la detección de vehículos en dirección contraria basada en flujos de movimiento. Para ello, se realiza en primera instancia una estimación de movimiento sobre las imágenes de las cámaras. Dicha estimación se realiza por bloques de 8 x 8 píxeles y, para cada bloque, se elige como vector representativo la mediana de todos ellos, reduciendo así la cantidad de datos a tratar. Para el modelado de los datos se emplea un modelo de Mezcla de Gaussianas de tres componentes con actualización dinámica. La única información necesaria para detectar vehículos en dirección contraria es la dirección θ de los vectores, por lo que el modelo tan solo necesita este parámetro para su construcción.

El sistema que se propone en este proyecto, que supone una continuación del proyecto desarrollado anteriormente [19] por el grupo GPM de la Universidad Carlos III de Madrid, trata de ampliar la funcionalidad de [11] para que, teniendo en cuenta los vectores de movimiento al completo y usando el algoritmo EM para la actualización del modelo GMM, pueda ser empleado para la detección general de anomalías en carreteras. Así pues, ya no solo será posible detectar vehículos en dirección contraria, sino que también podrán detectarse accidentes de tráfico, embotellamientos, invasiones de la calzada… Se reduce también la complejidad del modelado estadístico haciendo uso de un modelo GMM con únicamente una o dos componentes, siendo esta decisión tomada de manera automática por el sistema en base a una serie de restricciones.

2.2. Descripción del sistema previo

Antes de detallar los módulos implementados en el proyecto, se incluye a continuación una breve descripción de los módulos incluidos en el sistema previo [19], cuyo diagrama de bloques se muestra en la Figura 2.2:

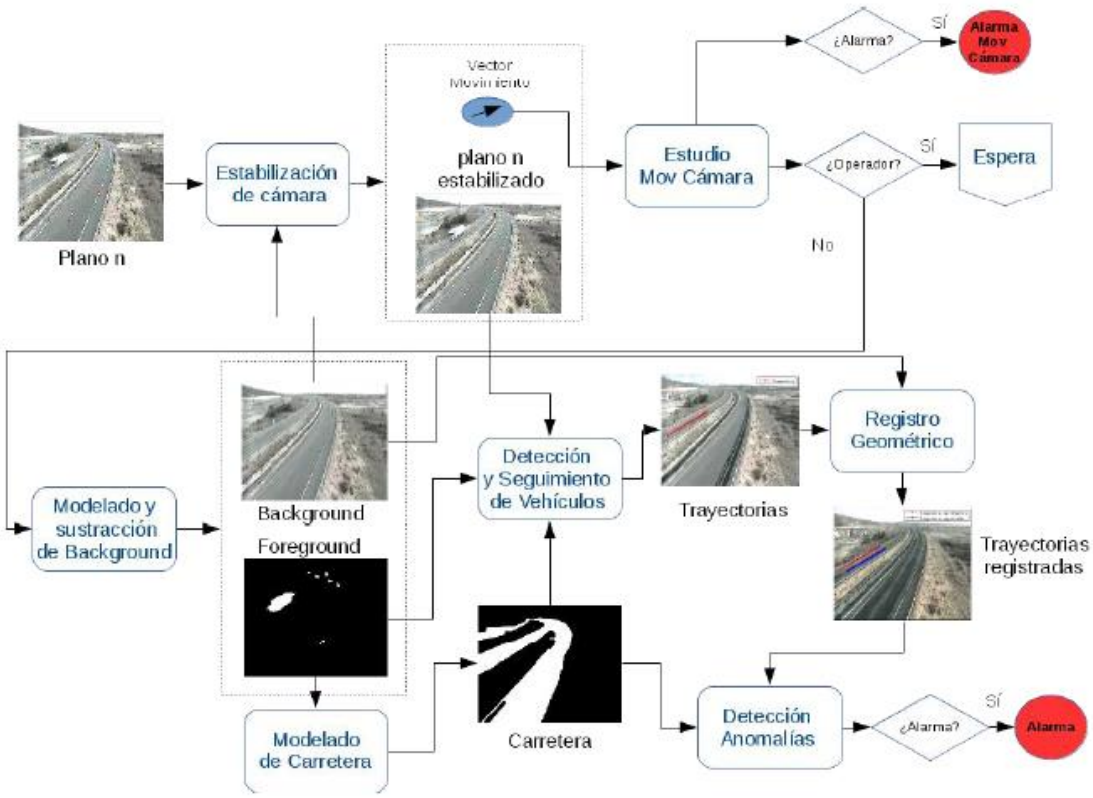

Fig. 2.2: diagrama de flujo del sistema desarrollado con anterioridad [19].

- Estabilización de cámara: en primer lugar, debido al movimiento de cámara ocasionado por fuertes vientos racheados, se realiza una estabilización de cámara para facilitar el procesado posterior. Esta estabilización se realiza mediante un alineamiento de Fourier con precisión subpíxel [12].
- Estudio de movimiento de cámara: este módulo se encarga de detectar los casos en los que el algoritmo mencionado en el punto anterior no puede funcionar con normalidad, bien por causas naturales, bien por movimiento de cámara efectuado por un operador, desactivando así el procesado posterior hasta alcanzar una nueva situación estable. Se emplea para ello un algoritmo de estimación de movimiento.
- Modelado y sustracción de background: el objetivo de este módulo es crear y actualizar en primera instancia el modelo de fondo (background) para después poder sustraerlo y detectar los objetos en movimiento. Tras efectuar diversas pruebas, se decidió emplear un algoritmo de mediana [13].
- Modelado de carretera: en este módulo se trata de diferenciar la zona de la carretera del resto de la imagen a partir de los objetos en movimiento extraídos en el módulo anterior. Para ello, se realiza un mapa energético en

base a la cantidad de vehículos que han pasado por cada zona, considerando carretera las zonas del mapa que estén por encima de un cierto umbral.

- Alineamiento de vistas: una vez se tiene la carretera totalmente caracterizada, se busca alinear la vista de cámara actual con una aprendida anteriormente para poder emplear los datos (vectores de movimiento, modelos estadísticos, trayectorias aprendidas…) ya obtenidos a nivel de píxel. Se realiza para ello una estimación robusta de la matriz de homografía, con el método RANSAC [14]. Esto permitirá, en caso de movimientos de cámara no demasiado grandes, relacionar vectores de movimiento y trayectorias con los de vistas anteriores, actualizar modelos de píxeles ya aprendidos, y aprender solo aquellos que sean realmente nuevos.
- Detección y seguimiento de vehículos: para el seguimiento de vehículos se ha optado por utilizar filtros de partículas [12][15] que suponen una variación del filtrado bayesiano. La idea consiste en que, a cada vehículo detectado en la imagen, se le asigna un *bounding box* (un rectángulo de unas ciertas dimensiones centrado en una cierta localización). El filtro de partículas permite asignar distintos pesos, distintas probabilidades, a los posibles píxeles hacia los que se podría mover el *bounding box* de un frame a otro, manteniendo así posibles trayectorias abiertas y no comprometiendo la elección del sistema desde un primer instante. La selección de esos píxeles se hace en función del histograma HSV y de la máscara de *foreground* entre otros parámetros.

  Para detectar vehículos en carretera y asignarles un filtro de partículas se buscan regiones de *foreground* suficientemente grandes como para poder ser un vehículo y, en caso de que el solape con un *bounding box* ya existente no supere un cierto umbral, se le asigna un nuevo *bounding box* a dicha región.

- Detección de anomalías: es la etapa final del sistema. El sistema trata de aprender cuáles son las trayectorias “normales” durante un cierto periodo de tiempo, asumiendo que durante ese tiempo no ocurren incidentes y, una vez superada la fase de aprendizaje, compara las trayectorias actuales con las aprendidas. Si las diferencias superan un cierto umbral, se dispara la alarma; en caso contrario, se actualizan los modelos con las nuevas trayectorias.

  Además de trayectorias anómalas, también se han diseñado submódulos para detectar anomalías en los alrededores de la carretera (vehículos que se salen de la calzada, incendios, ganado…) y detectar vehículos parados en la calzada.

3. Sistema propuesto

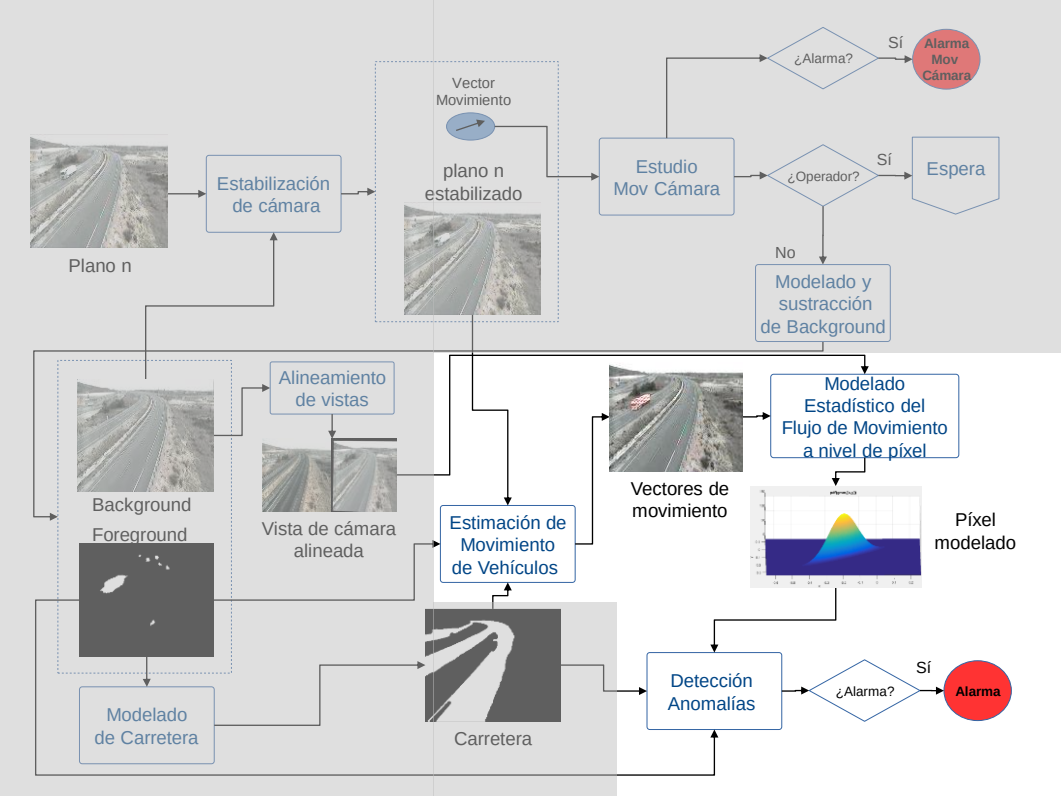

Fig. 3.1: pipeline del sistema completo, haciendo énfasis sobre las partes desarrolladas en este proyecto

El sistema que se propone en la Figura 3.1 trata de abordar la identificación de incidentes desde el punto de vista de la detección de anomalías [19]. Este sistema requiere un aprendizaje del flujo habitual de vehículos en una cierta carretera durante un determinado periodo de tiempo para después ser capaz de reportar incidentes, si apareciesen, ante un comportamiento anómalo de dicho flujo. La principal diferencia con el método propuesto en [19] es que se pretende modelar el flujo habitual de movimiento en cada punto de la carretera, sin necesidad de modelar individualmente las trayectorias de los vehículos que pasan por ella.

Es interesante destacar que, en este nuevo sistema, la detección se basa en el análisis del movimiento instantáneo, puesto que no se tiene el concepto de trayectoria. Es cierto que a este análisis instantáneo se le aplican posteriormente una serie de condiciones espacio-temporales para decidir si se ha de activar la alarma por incidente o no, pero aun así, ciertas situaciones teóricas no podrían ser detectadas precisamente por no tener conciencia de trayectoria.

Un ejemplo de esta situación podría ser un vehículo zigzagueando entre varios carriles. A nivel instantáneo, es muy probable que las variaciones de dirección sean entendidas por los modelos estadísticos de los píxeles como algo normal, puesto que un cambio de carril a nivel de píxel no supone una desviación muy elevada respecto a los valores habituales de los vectores de movimiento. Sin embargo, si lo que se aprenden son las trayectorias habituales, una trayectoria en zigzag representa claramente una anomalía.

Otro caso podría ser el de un vehículo que hace un cambio de dirección en mitad de una carretera de doble sentido. Con este sistema, el tiempo durante el que los vectores de movimiento son anómalos (el tiempo del giro) sería tan breve que probablemente no se llegaría a activar la alarma por incidente, puesto que una vez ha terminado el giro, los vectores de movimiento volverían a ser normales. En cambio, en un sistema como el de referencia [19], una trayectoria en forma de herradura en una carretera de doble sentido será siempre detectada como incidente.

A pesar de estos inconvenientes teóricos, el sistema gana una prestación muy importante en entornos de trabajo en tiempo real, y es su bajo coste computacional. Al estar basado en el análisis instantáneo de los vectores de movimiento, el sistema no tiene memoria, lo que otorga mayor rapidez en los cálculos y posibilita su implementación en cámaras de vigilancia para el tráfico, donde el tiempo de respuesta es crítico.

Para el funcionamiento de este detector de anomalías se ha hecho uso de parte del trabajo realizado en [19]. Tal y como se puede observar en la Figura 3.1, se ha partido de los *outputs* de los módulos de estabilización, estudio de movimiento de cámara, modelado y sustracción de background y modelado de carretera, desarrollados en [19], para la construcción del sistema implementado. Una vez obtenidos, se avanza en primer lugar a un módulo de estimación de movimiento de vehículos en el que, haciendo uso del algoritmo de Lucas & Kanade, se pretende obtener los vectores de movimiento asociados a los vehículos presentes en la imagen. A medida que llegan nuevos vectores de movimiento se procede a modelar estadísticamente los píxeles de la carretera mediante un modelo de Mezcla de Gaussianas (GMM) con actualización dinámica de los parámetros. Por último, una vez finalizado el entrenamiento, se procede a la detección de incidentes en sí.

3.1. Estimación de movimiento de vehículos

Partiendo de una situación en la que se tiene el plano de cámara estabilizado, la máscara de carretera y la máscara de *foreground*, se realiza una estimación de

movimiento sobre la imagen previo filtrado gaussiano. De esta forma se consigue, suavizar la imagen y obtener unos vectores de movimiento en los ejes x e y más uniformes. A continuación se detallan los dos métodos evaluados en este proyecto.

3.1.1. Método de estimación de movimiento basado en bloques

El método de estimación de movimiento basado en bloques tiene como objetivo hallar el desplazamiento $\mathbf{u} = (\mathrm{u}, \mathrm{v})$ asociado a cada píxel de la imagen $\mathbf{x} = (\mathrm{x}, \mathrm{y})$. Para ello, se define alrededor del píxel un bloque de referencia $\mathbf{I}_0(\mathbf{x})$ de lado 2*BSL+1 compuesto por un conjunto de píxeles $\mathbf{x}_\mathrm{i} = (\mathrm{x}_\mathrm{i}, \mathrm{y}_\mathrm{i})$, que se tratará de buscar en la otra imagen $\mathbf{I}_1(\mathbf{x})$ [16]. Para ello, se realizan dos suposiciones previas:

- Que todos los píxeles del bloque definido experimentan un movimiento constante de traslación.
- Y que el movimiento del bloque no puede ser superior a un cierto umbral en las componentes horizontal y vertical.

Finalmente, se plantea un problema de ajuste de plantillas, como se puede observar en la Figura 3.3., en el que es preciso determinar dos elementos:

- La métrica de error, que permitirá decidir cuál es la solución óptima en base a que los bloques sean lo más parecidos posible.
- Y el patrón de búsqueda, para decidir cómo alcanzar dicha solución en el menor número de pasos posible.

Se ha optado en este caso por una métrica mínimos cuadrados (SSD, Sum of Square Differences) y un patrón de búsqueda exhaustiva.

La métrica de error SSD calcula el ajuste entre dos bloques haciendo uso de la ecuación:

$$E_{SSD}(\mathbf{u}) = \sum_i [I_1(\mathbf{x}_i + \mathbf{u}) - I_0(\mathbf{x}_i)]^2 = \sum_i e_i^2 \qquad (1)$$

donde los e_i son el error residual.

Sin embargo, haciendo uso únicamente de la métrica de error, no se penaliza la distancia a la que se encuentra el bloque elegido como solución. *A priori*, entre un frame y el siguiente, el movimiento de un cierto bloque no debería ser muy grande, por lo que si el bloque elegido como óptimo en base a la métrica de error se encuentra

muy alejado del original, parece razonable pensar que no es realmente la mejor solución (ver Figura 3.2).

Para evitar que esto pueda llegar a ocurrir, se trabaja con funciones de coste regularizadas, ya que permiten ajustar la penalización que se le aplica a la distancia de la solución escogida por la métrica de error. La función de coste utilizada tiene la siguiente forma:

$$C(\mathbf{u}) = E_{SSD}(\mathbf{u}) + \lambda\|\mathbf{u}\|_2 \quad (2)$$

siendo $\|\mathbf{u}\|_2$ el módulo al cuadrado de $\mathbf{u}$ y λ el parámetro que permite ajustar la penalización por la distancia entre el bloque original y el solución. Cuanto mayor sea λ, más se penalizará la distancia entre los bloques.

El patrón de búsqueda exhaustiva funciona evaluando todas las posibles soluciones dentro de un área o Rango de Búsqueda (SR, Search Range). Para un SR cuadrado de lado 2SRL+1 píxeles se evalúan $(2SRL + 1)^2$ puntos, y la magnitud máxima de movimiento de un cierto bloque queda fijada en $\sqrt{2}SRL$.

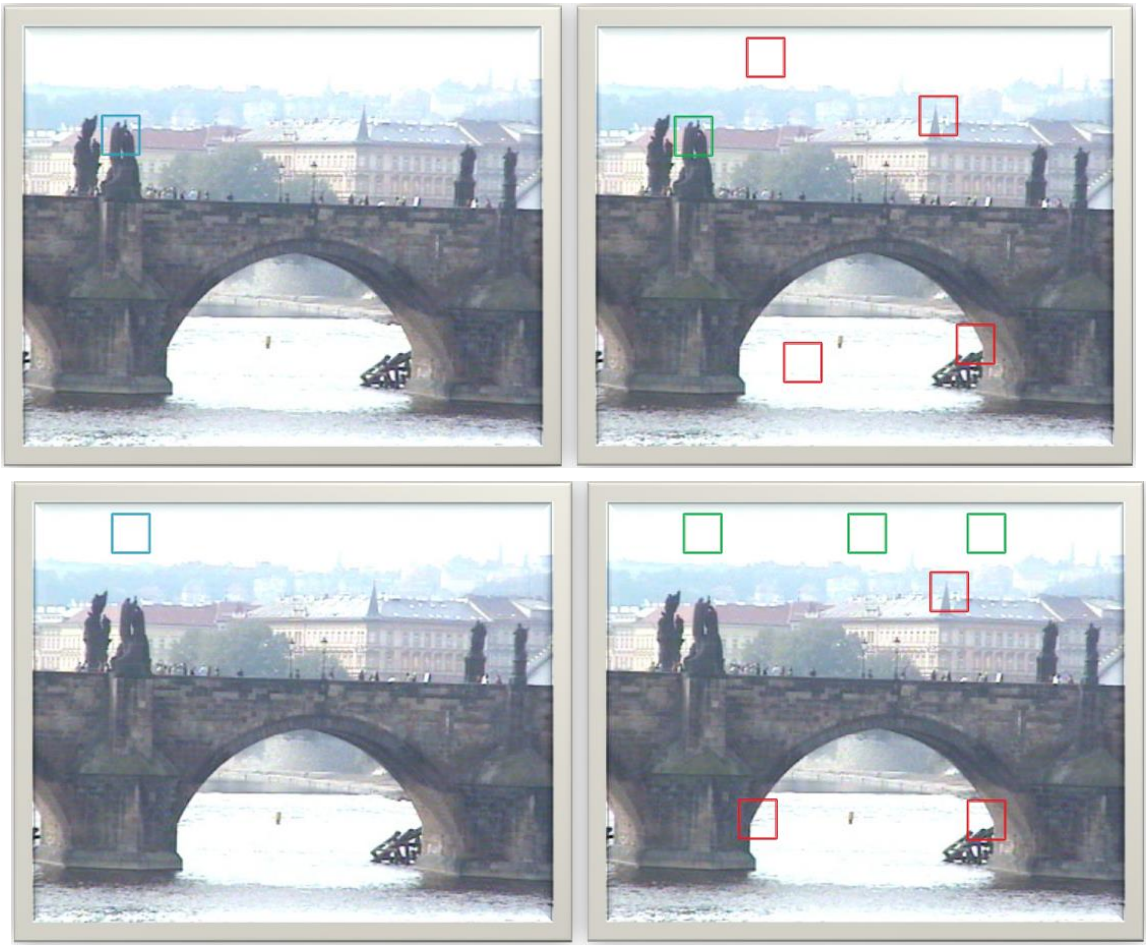

Fig. 3.2: posibles soluciones al matching de un bloque entre un frame y otro. Arriba, una única solución posible. Abajo, varias soluciones que encajan con el bloque original si no se tiene en cuenta la distancia. Obtenido de [16].

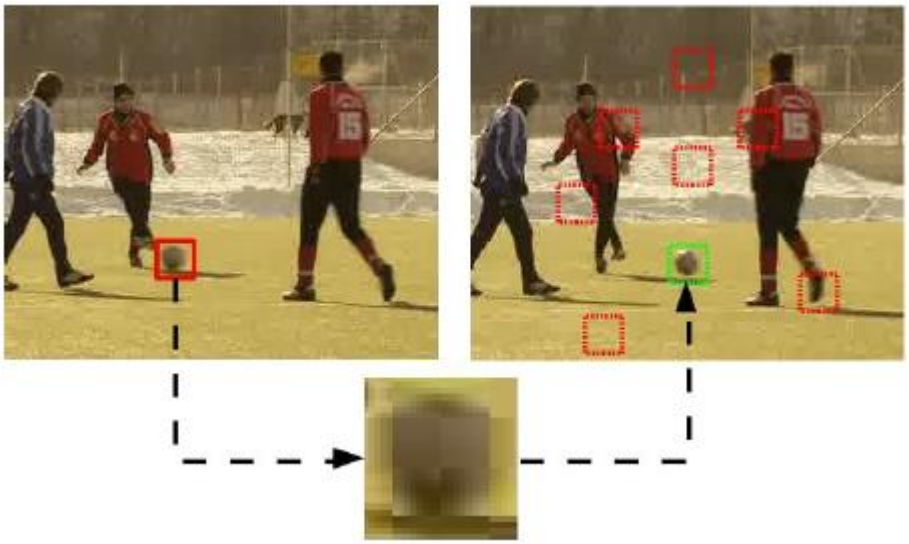

Fig. 3.3: posibles soluciones al matching de un bloque entre un frame y otro. En verde se muestra la solución óptima. Obtenido de [16].

En la Figura 3.4 se puede observar un ejemplo con los distintos parámetros del método.

3.1.2. Método de estimación de movimiento de Lucas y Kanade

El método de Lucas y Kanade es un método diferencial de estimación densa de movimiento en el que, para un vecindario circular Ω de un cierto píxel, se asume un movimiento constante [16]. La función que se ha de minimizar es:

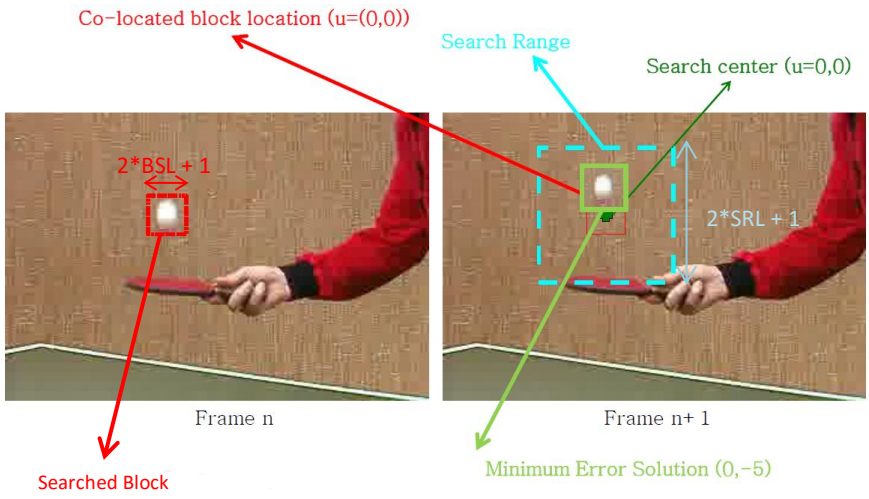

Fig. 3.4: parámetros de la estimación de movimiento por bloques con patrón de búsqueda exhaustiva. Obtenido de [16].

$$\arg\min_{\mathbf{u}} \sum_{\mathbf{x}_i \in \Omega} [\nabla I(\mathbf{x}_i, t)\mathbf{u} + I_t(\mathbf{x}_i, t))]^2 \quad (3)$$

siendo ∇ el gradiente de *I*; *I* la imagen en la que se desea estimar el movimiento; $\mathbf{x_i}$ **son** los píxeles que pertenecen al vecindario Ω; t el instante temporal; y **u** el vector desplazamiento. Esta misma ecuación también se puede expresar en forma matricial:

$$\mathbf{Au} = \mathbf{b} \quad (4)$$

siendo:

$$\boldsymbol{A} = \begin{pmatrix} I_x(\boldsymbol{x}_1) & I_y(\boldsymbol{x}_1) \\ I_x(\boldsymbol{x}_2) & I_y(\boldsymbol{x}_2) \\ \dots & \dots \\ I_x(\boldsymbol{x}_n) & I_y(\boldsymbol{x}_n) \end{pmatrix} \qquad \boldsymbol{b} = \begin{pmatrix} -I_t(\boldsymbol{x}_1) \\ -I_t(\boldsymbol{x}_2) \\ \dots \\ -I_t(\boldsymbol{x}_n) \end{pmatrix}$$

Se presenta ahora un sistema sobredeterminado con solución de mínimos cuadrados:

$$\mathbf{u} = (\mathbf{A^T A})^{-1} \mathbf{A^T b} \quad (5)$$

del que se obtendrán distintas soluciones en función del radio r que se le asigne a Ω, siendo más robusto y menos preciso cuanto mayor sea r (ver Figura 3.5.).

Es también posible establecer un umbral de estimación de movimiento satisfactoria, thrLK, por debajo del cual toda estimación se considera nula. Se suele escoger un valor muy pequeño para eliminar estimaciones tan pequeñas que no pueden ser consideradas como válidas.

Para la elección del algoritmo se ha hecho uso de las conclusiones alcanzadas en [20], en el que los experimentos realizados otorgan unos mejores resultados al algoritmo de Lucas y Kanade. Asimismo, y como se muestra en el apartado 4.2., los experimentos

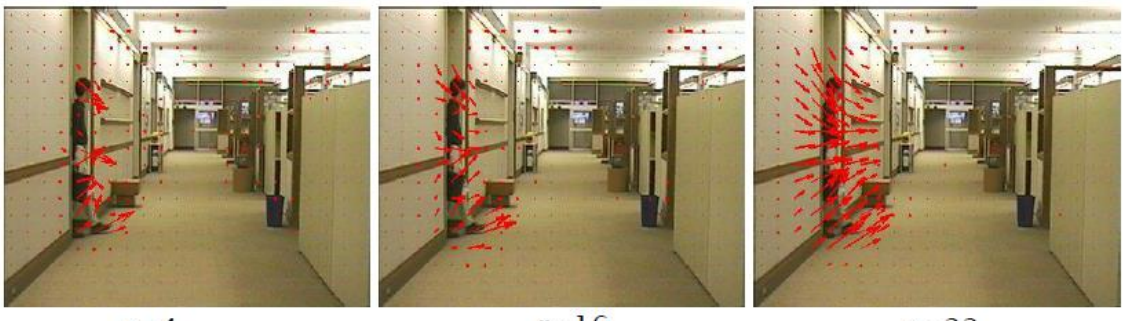

Fig. 3.5: vectores de movimiento resultantes de aplicar el método de Lucas&Kanade con diferentes valores de r. Obtenido de [16].

propios realizados entre la estimación por bloque y Lucas y Kanade otorgan una cierta ventaja a este último, por lo que se ha decidido realizar la estimación de movimiento con dicho algoritmo.

Se ha optado por un valor de 1 para la desviación estándar σ del prefiltrado gaussiano de las imágenes, un umbral de estimación de movimiento satisfactoria thrLK de $2x10^{-5}$, y una r de 15. La elección de este valor tan elevado, que otorga mayor robustez aunque menor precisión, se debe a que el problema de la precisión puede ser fácilmente compensado multiplicando la matriz con los vectores de movimiento por la máscara de *foreground* obtenida del módulo de modelado y sustracción de background. Así, los vectores tienen mayor robustez sin que la precisión suponga un inconveniente, como se puede observar en la Figura 3.6.

En el anexo II se incluye la descripción de un módulo de almacenamiento eficiente de vectores de movimiento estimados, empleado en este proyecto para reducir los tiempos de cómputo durante las simulaciones.

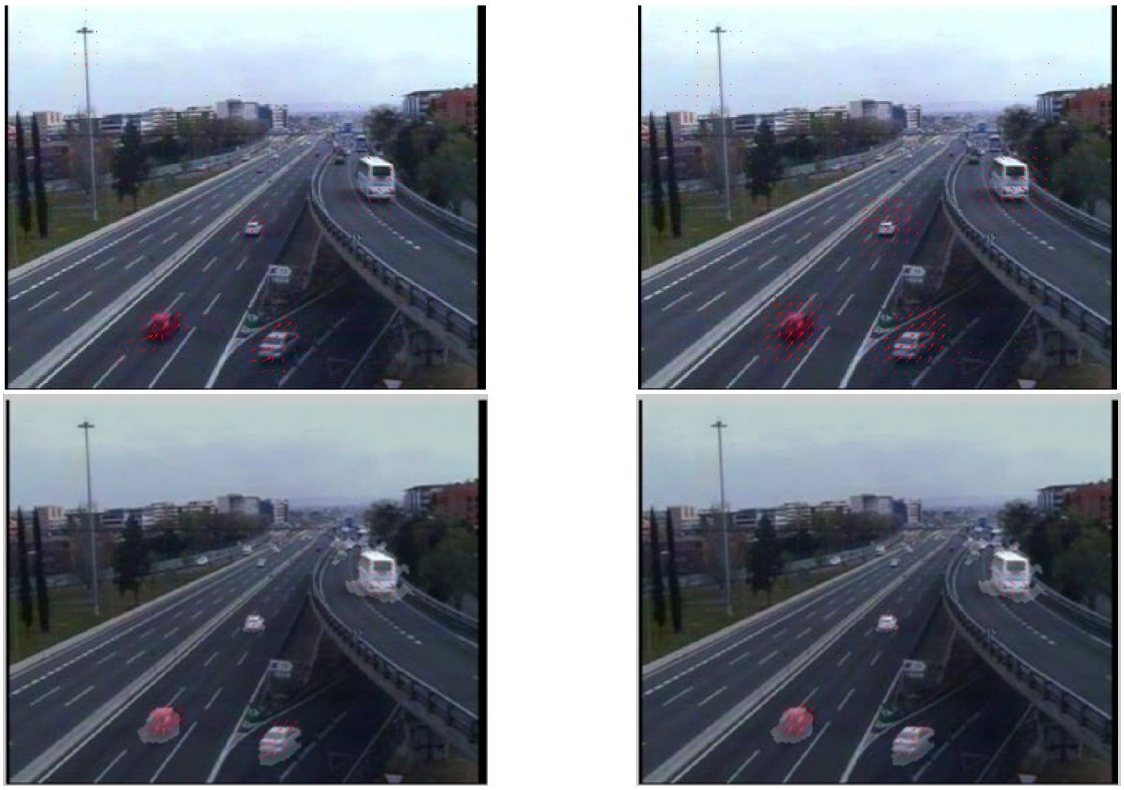

Fig. 3.6: vectores resultantes de la estimación de movimiento con el algoritmo L&K. Izquierda, r=5. Derecha, r=15. Arriba, sin aplicar máscara de foreground. Abajo, aplicando máscara de foreground.

3.2. Modelado estadístico del flujo de movimiento a nivel de píxel

Una vez obtenidos los vectores de movimiento de todos los vehículos de la escena, se procede a modelar estadísticamente dicho movimiento en los píxeles de la imagen pertenecientes a la carretera. Para reducir el coste computacional, a partir de este módulo se ha optado por trabajar con las imágenes escaladas por un factor de ¼ en cada dimensión, puesto que los resultados son muy parecidos y el tiempo de ejecución se reduce drásticamente. Habrá, por tanto, que dividir por 4 los vectores de cada eje obtenidos del módulo anterior, dando lugar a un tamaño final de 1/16 de la imagen original.

El modelado estadístico de cada píxel se realiza mediante un modelo de Mezcla de Gaussianas multivariante (2-D, eje x y eje y). El número de componentes en cada punto de la carretera, como se describirá más adelante, se decide tras finalizar el periodo de entrenamiento inicial.

3.2.1. Modelo de cámara extendido y normalización de nuevas localizaciones similares

En este punto, conviene recordar la necesidad de diseñar un sistema que se pueda adaptar a los movimientos de cámara por parte de los operadores. En este escenario, un operador dejará la cámara en una posición que, si bien será similar a otra vista con anterioridad (si no lo es, nuestro sistema tendrá que comenzar su periodo de aprendizaje), no tendrá exactamente el mismo campo de visión. Esto causará que haya puntos en la nueva vista que quizás no se hayan estudiado con anterioridad y puntos de la anterior que ahora no sean visibles.

Para solventar esta problemática, cada vez que el sistema detecta una vista de cámara nueva, codificará dicha vista mediante una imagen extendida de tamaño 3Hx3W centrada en el origen, de modo que se incorpore robustez frente a movimientos en la cámara y variaciones en el campo de visión (si una cámara se mueve más allá de esta extensión, el solape entre vistas será tan bajo que requerirá modelar la nueva vista como una independiente).

En la práctica, los parámetros de las Gaussianas de cada píxel en una vista de cámara nueva se almacenan en una estructura con dimensiones 3H’ x 3W’, siendo H’ y W’ la cuarta parte del alto y el ancho originales de la imagen. Además, en esta estructura todos los datos están referenciados a la vista inicial, de modo que una nueva posición

de cámara asociada a la misma vista será previamente transformada mediante una homografía.

En la Figura 3.7 se puede encontrar una descripción esquemática del funcionamiento de este sistema.

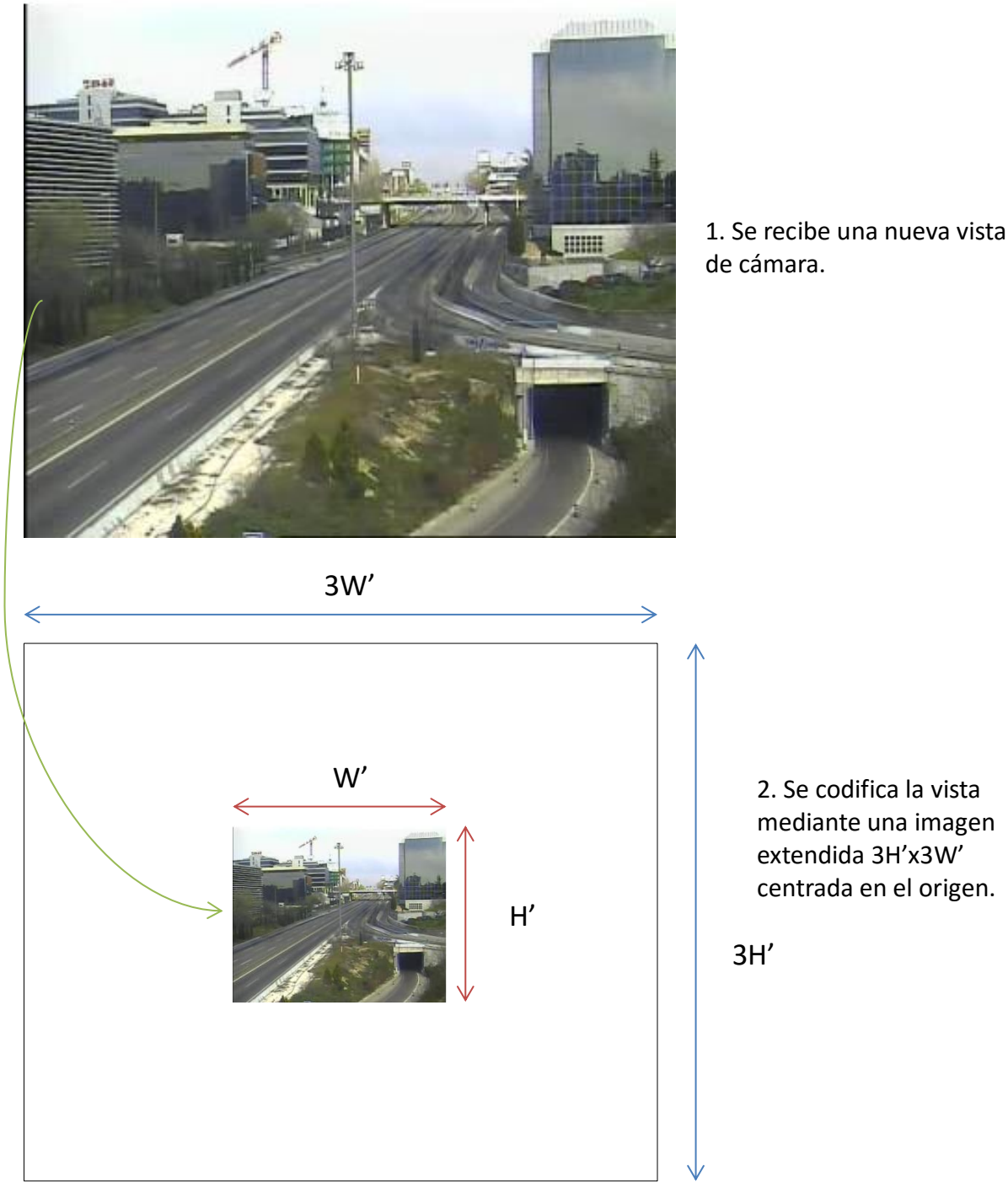

Fig. 3.7: descripción esquemática de la codificación de vistas similares.

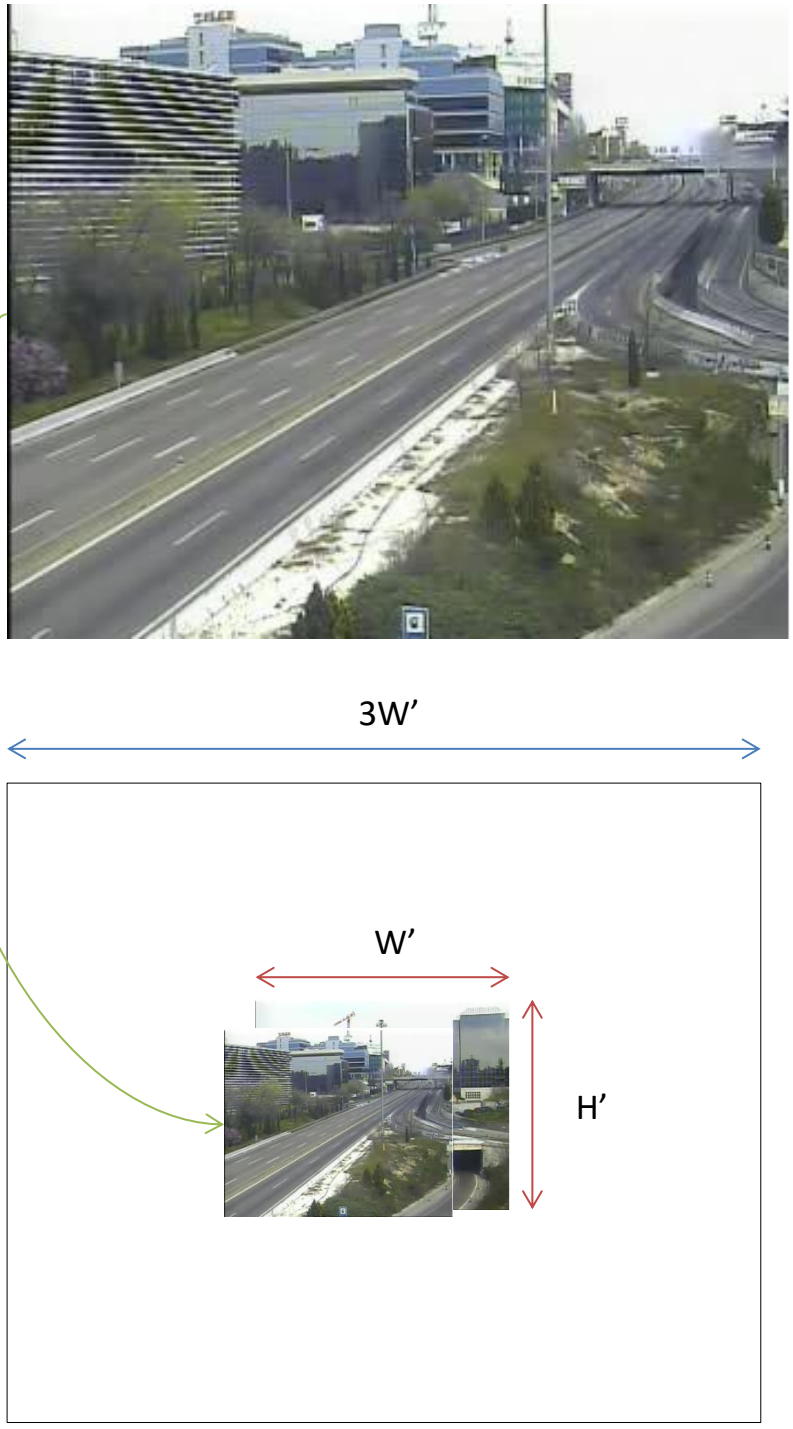

Fig. 3.7: descripción esquemática de la codificación de vistas similares.

3.2.2. Modelo de Mezcla de Gaussianas para el modelado estadístico del movimiento

Para explicar el Modelo de Mezcla de Gaussianas (*Gaussian Mixture Model* o GMM), así como el algoritmo EM utilizado para aprender los parámetros del mismo, se hará

uso de la terminología empleada por Bishop en [17]. En la Figura 3.8. se pueden observar dos ejemplos del modelado de datos con una Mezcla de Gaussianas de dos componentes.

Sea **u** el vector de movimiento estimado para un cierto píxel de la imagen, una Mezcla de Gaussianas puede expresarse como la combinación lineal de componentes Gaussianas, con el fin de proporcionar una representación más compleja y completa de los datos. Su distribución se puede expresar como:

$$p(\boldsymbol{u}) = \sum_{k=1}^{K} \pi_k N(\boldsymbol{u} \,|\boldsymbol{\mu}_k, \sum_k) \tag{6}$$

donde $N(\boldsymbol{u}|\boldsymbol{\mu}_k, \sum_k)$ hace referencia a cada una de las *K* componentes Gaussianas de media $\boldsymbol{\mu}_k$ y matriz de covarianza $\sum_k$, y π_k son los coeficientes de mezcla.

Para poder llegar a esta expresión, se introduce una variable aleatoria (v.a.) indexadora *K*-dimensional **z** (vector de longitud K), en la que un elemento particular z_k tiene valor 1 y el resto de elementos, valor 0. Hay, por lo tanto, *K* posibles estados para el vector **z** dependiendo de cuál sea el elemento de valor 1.

La distribución conjunta de **u** y de **z**, $p(\mathbf{u}, \mathbf{z})$, se puede obtener, aplicando el Teorema de Bayes, como el producto de la distribución marginal de **z**, $p(\mathbf{z})$, y la distribución condicionada, $p(\mathbf{u}|\mathbf{z})$. $p(\mathbf{z})$ se puede expresar en función de los coeficientes de mezcla, π_k, siendo:

$$p(z_k = 1) = \pi_k \tag{7}$$

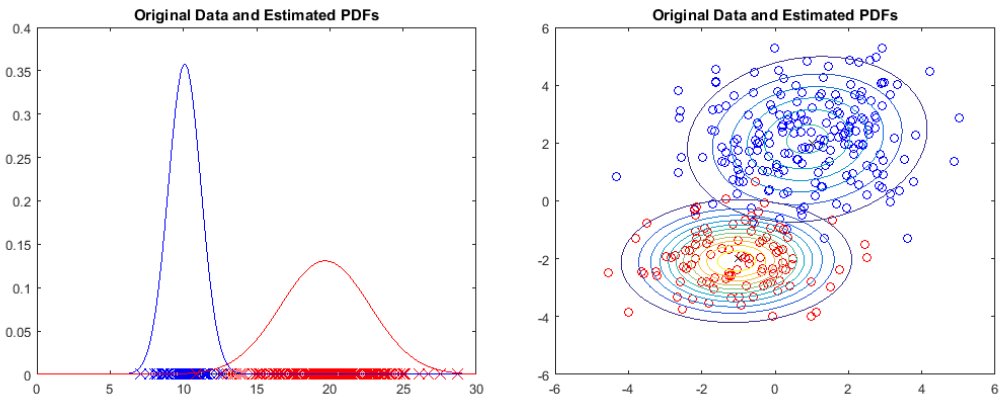

Fig. 3.8: ejemplos de modelos de Mezcla de Gaussianas en 1-D y 2-D generados a partir del código de McCormick, C. en su artículo web *Gaussian Mixture Models Tutorial and MATLAB Code*.

donde los π_k han de estar comprendidos entre 0 y 1 y la suma de ellos ha de ser igual a 1. La distribución de **z**, al tratarse de un vector de dimensión (1, K), será, por tanto:

$$p(\boldsymbol{z}) = \prod_{k=1}^{K} \pi_k^{z_k} \tag{8}$$

Siguiendo un razonamiento similar para la distribución condicionada $p(\mathbf{u}|\mathbf{z})$, para un valor determinado de **z**, la distribución tiene la forma de una Gaussiana:

$$p(\boldsymbol{u}|z_k = 1) = N(\boldsymbol{u}|\boldsymbol{\mu}_k, \sum_k) \tag{9}$$

Para el vector **z** será, por tanto y de forma análoga a $p(\mathbf{z})$:

$$p(\boldsymbol{u}|\boldsymbol{z}) = \prod_{k=1}^{K} N(\boldsymbol{u}|\boldsymbol{\mu}_k, \sum_k)^{z_k} \tag{10}$$

Finalmente, se obtiene $p(\mathbf{u})$ como la suma en **z** de la distribución conjunta $p(\mathbf{u}, \mathbf{z})$, siendo ésta el producto de $p(\mathbf{z})$ y $p(\mathbf{u}|\mathbf{z})$ (de nuevo por el Teorema de Bayes):

$$p(\boldsymbol{u}) = \sum_{\boldsymbol{z}} p(\mathbf{u}, \mathbf{z}) = \sum_{\boldsymbol{z}} p(\mathbf{z}) p(\mathbf{u}|\mathbf{z}) = \sum_{k=1}^{K} \pi_k N(\mathbf{u}\,|\boldsymbol{\mu}_k, \sum_k) \tag{11}$$

Si se tienen varias observaciones de **u** ($\mathbf{u}_1$, …, $\mathbf{u}_N$), puesto que $p(\mathbf{u})$ se ha representado como $\sum_z p(\mathbf{u},\mathbf{z})$, se puede concluir que, para cada observación $\mathbf{u}_n$, ha de existir una variable latente $\mathbf{z}_n$. Así pues, de acuerdo con [18], se podrá expresar:

$$p(\boldsymbol{u}, \boldsymbol{z}) = \prod_{n=1}^{N} p(\boldsymbol{z}_n) p(\boldsymbol{u}_n|\boldsymbol{z}_n) = \prod_{n=1}^{N} \prod_{k=1}^{K} \pi_k^{z_{nk}} N(\boldsymbol{u}_n|\boldsymbol{\mu}_k, \sum_k)^{z_{nk}} \tag{12}$$

que será muy útil para explicar más adelante el método EM.

También será necesario definir $\gamma(z_k) = p(z_k = 1|\mathbf{x})$, cuyo valor es posible obtener aplicando una vez más el Teorema de Bayes:

$$\gamma(z_k) = p(z_k = 1|\boldsymbol{u}) = \frac{p(z_k=1,\boldsymbol{u})}{p(\boldsymbol{u})} = \frac{p(z_k=1)p(\boldsymbol{u}|z_k=1)}{\sum_{j=1}^{K} p(z_j=1)p(\boldsymbol{u}|z_j=1)} = \frac{\pi_k N(\boldsymbol{u}|\boldsymbol{\mu}_k,\sum_k)}{\sum_{j=1}^{K} \pi_j N(\boldsymbol{u}|\boldsymbol{\mu}_j,\sum_j)} \tag{13}$$

Por tanto, π_k se puede interpretar como la probabilidad *a priori* de que $z_k = 1$, y $\gamma(z_k)$, como la probabilidad *a posteriori* correspondiente.

3.2.3. Algoritmo Esperanza-Maximización (EM)

El algoritmo EM es una de las formas más potentes y habituales de encontrar la máxima verosimilitud en modelos con variables latentes, como es el caso de la Mezcla de Gaussianas. Esto se debe a que la presencia de un sumatorio en *k* dentro de un logaritmo, de tal forma que éste no actúa directamente sobre la Gaussiana, complica los cálculos de manera considerable. Es por ello que se ha de emplear otra técnica para encontrar la máxima verosimilitud. Se muestra, a continuación, el funcionamiento del algoritmo, cuyo desarrollo detallado se puede encontrar en el Anexo III.

1. En primer lugar, es preciso inicializar los valores de las medias $\boldsymbol{\mu}_k$, de las covarianzas $\sum_k$ y de los coeficientes de mezcla $\boldsymbol{\pi}_k$. Esta inicialización puede hacerse con valores aleatorios, si bien es recomendable aplicar el algoritmo *k-means* para inicializar los parámetros $\boldsymbol{\mu}_k$, $\sum_k$ y $\boldsymbol{\pi}_k$ a unos valores óptimos y reducir así el número de iteraciones.

2. Una vez se tienen los valores de $\boldsymbol{\mu}_k$, $\sum_k$ y $\boldsymbol{\pi}_k$, se efectúa el **paso E,** que consiste en evaluar las probabilidades *a posteriori* de z_{nk}=1 dado u. Es decir:

$$\gamma(z_{nk}) = p(z_{nk} = 1|\mathrm{u}) = \frac{\pi_k N(\mathrm{u}_n|\mu_k,\sum_k)}{\sum_{j=1}^{K}\pi_j N(\mathrm{u}_n|\mu_j,\sum_j)} \tag{14}$$

3. A continuación se realiza el **paso M**, reestimando los parámetros $\boldsymbol{\mu}_k$, $\sum_k$ y $\boldsymbol{\pi}_k$ a partir de las nuevas probabilidades *a posteriori*:

$$\boldsymbol{\mu}_k^{new} = \frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_{nk})\boldsymbol{x}_n \tag{15}$$

$$\sum_k^{new} = \frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_{nk})(\boldsymbol{u}_n - \boldsymbol{\mu}_k^{new})(\boldsymbol{u}_n - \boldsymbol{\mu}_k^{new})^T \tag{16}$$

$$\pi_k = \frac{N_k}{N} \tag{17}$$

siendo

$$N_k = \sum_{n=1}^{N}\gamma(z_{nk}) \tag{18}$$

4. Por último, se evalúa la log-verosimilitud:

$$\ln p(\mathrm{U}|\mu,\sum,\pi) = \sum_{n=1}^{N} ln\{\sum_{k=1}^{K}\pi_k N(\mathbf{u}_n|\boldsymbol{\mu}_k,\sum_k)\} \tag{19}$$

y se comprueba la convergencia o bien de los parámetros o bien de la log-verosimilitud. Si no se cumple el criterio de convergencia, se vuelve al paso 2.

En la Figura 3.9 se puede observar la evolución del algoritmo EM conforme se van sucediendo las distintas iteraciones.

3.2.4. Aprendizaje online de Mezclas de Gaussianas

Uno de los objetivos del proyecto es lograr un sistema que se adapte a nuevas localizaciones debido al movimiento de cámara por parte de los operadores. Debido esta restricción, la aplicación de un modelo de Mezcla de Gaussianas con aprendizaje batch no parece realista, ya que es poco probable disponer de todos los datos de entrenamiento. Asimismo, las condiciones de tráfico en las carreteras varían con el tiempo (el tráfico puede desviarse de una carretera por obras, dando lugar a una mayor densidad en las vías alternativas, en verano aumenta el número de desplazamiento en unas vías, disminuyendo en otras, etc.).

Cabe destacar que, dado que se trata de un sistema adaptativo, el aprendizaje de los modelos no finaliza en ningún momento, está en constante actualización. Es por ello que, en este proyecto, se ha optado por un modelo de Mezcla de Gaussianas con un procedimiento de entrenamiento incremental basado en el método de actualización directa [21][22]. Este método supone una variación del algoritmo EM tradicional, permitiendo la actualización del modelo en base a los nuevos datos que estén disponibles según avancen los frames recibidos. Para ello, se separan los datos en dos: los datos que se han utilizado ya para entrenar el modelo en cuestión, y los nuevos datos disponibles, con la asunción de que el conjunto de probabilidades *a posteriori*

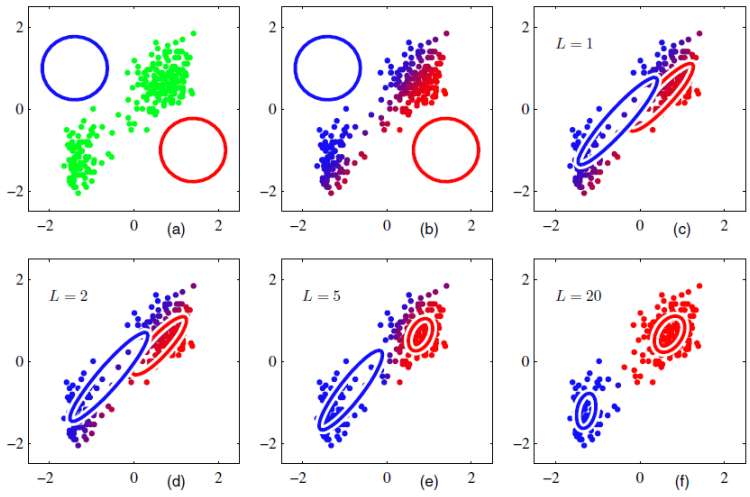

Fig. 3.9: inicialización y evolución del algoritmo EM tras 1, 2, 5 y 20 pasos. Extraído de [17].

$\{p(\mu_k, \sum_k|\mathbf{u})\}_{j=1}^{N}$ se mantiene cuando los nuevos datos $\{\overline{\mathbf{u}}\}_{j=1}^{\overline{N}}$ se emplean para actualizar el modelo. Esto solo es cierto si los datos son próximos al modelo ya existente. Así, las ecuaciones del algoritmo EM se pueden reescribir como:

$$\bar{\gamma}(z_k) = \frac{\overline{\pi}_k N(\overline{\boldsymbol{u}}|\overline{\boldsymbol{\mu}}_k, \overline{\sum}_k)}{\sum_{j=1}^{K} \overline{\pi}_j N(\overline{\boldsymbol{u}}|\overline{\boldsymbol{\mu}}_j, \overline{\sum}_j)} \tag{20}$$

$$\overline{N}_k = \sum_{n=1}^{\overline{N}} \bar{\gamma}(z_{nk}) \tag{21}$$

$$\overline{\pi}_k = \frac{\overline{N}_k^{init} + \overline{N}_k}{N + \overline{N}} \tag{22}$$

$$\overline{\boldsymbol{\mu}}_k = \frac{1}{\overline{N}_k^{init} + \overline{N}_k} \left(\overline{N}_k^{init} \overline{\boldsymbol{\mu}}_k^{init} + \sum_{n=1}^{\overline{N}} \bar{\gamma}(z_{nk}) \overline{\boldsymbol{u}}_n \right) \tag{23}$$

$$\overline{\sum}_k = \frac{1}{\overline{N}_k^{init} + \overline{N}_k} \Big(\overline{N}_k^{init} (\overline{\sum}_k^{init} + (\overline{\boldsymbol{\mu}}_k^{init} - \overline{\boldsymbol{\mu}}_k)(\overline{\boldsymbol{\mu}}_k^{init} - \overline{\boldsymbol{\mu}}_k)^T) + \sum_{n=1}^{\overline{N}} \bar{\gamma}(z_{nk}) (\overline{\boldsymbol{u}}_n - \overline{\boldsymbol{\mu}}_k)(\overline{\boldsymbol{u}}_n - \overline{\boldsymbol{\mu}}_k)^T \Big) \tag{24}$$

donde el superíndice *init* hace referencia a los parámetros existentes en el momento de la actualización(1).

Para la inicialización de los parámetros se hace uso del algoritmo *k-means* una vez recibidos 200 datos (200 vectores de movimiento no nulos, en los que la máscara de *foreground* y de carretera son '1') para un mismo píxel (un mismo modelo). Obtenidos ya los parámetros iniciales, cada vez que se reciban 200 nuevos datos se aplica el método de actualización directa descrito arriba. En la Figura 3.10 se puede ver la evolución de un modelo de Gaussianas con 3 componentes a medida que se reciben nuevos datos.

(1) En este proyecto se ha utilizado el código de Calinon, S. para la implementación del algoritmo de actualización online de Mezcla de Gaussianas. El código se puede encontrar en http://lasa.epfl.ch/sourcecode/#8

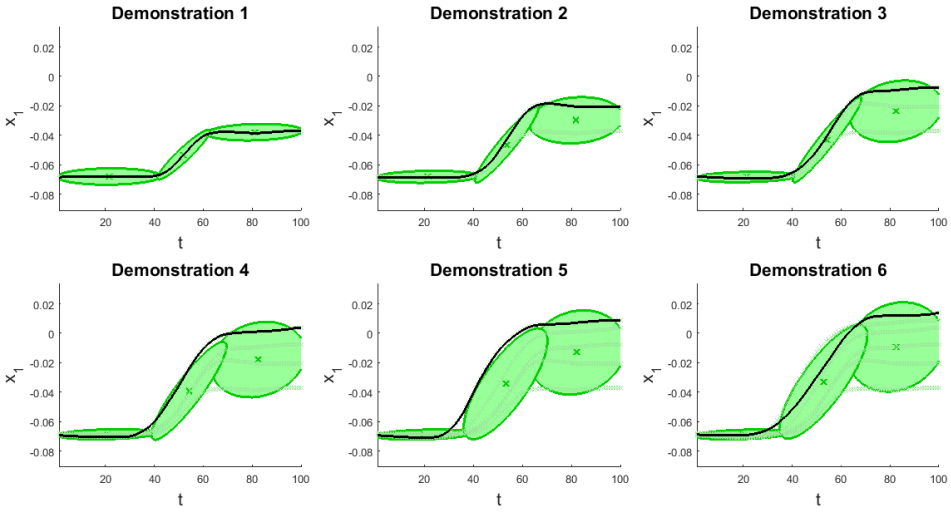

Fig. 3.10: ejemplo de actualización de Gaussianas para nuevos datos con el método de actualización directa a partir del código de Calinon, S. En él se observa un modelo GMM en dos dimensiones (tiempo en el eje x, posición en el eje y) con tres componentes que evoluciona a medida que se reciben nuevas trayectorias (línea negra de cada gráfica).

3.2.5. Eligiendo el número óptimo de componentes en la mezcla

Para cada píxel de la carretera se realiza un aprendizaje en paralelo de dos modelos de Mezcla de Gaussianas: uno con una sola componente y otro con dos componentes. En principio, para la mayoría de los píxeles debería ser suficiente con una única Gaussiana, dado que todos los vehículos se mueven aproximadamente en una misma dirección y una velocidad variable, siendo las variaciones capturadas por la varianza de la Gaussiana. Sin embargo, en algunos píxeles concretos (como incorporaciones, cruces…) podría ocurrir que una sola Gaussiana no fuese capaz de modelar la varianza de los datos de forma eficiente, por lo que sería preferible hacer uso de una segunda componente. Un ejemplo lo podemos encontrar en la Figura 3.11. En ella se presenta el caso de un píxel cercano a una incorporación, en el que parte de los vectores de movimiento estimados corresponden a vehículos que circulan por el carril derecho y parte a vehículos que se incorporan a la carretera. Así, el modelado estadístico de los datos se puede separar en dos Gaussianas, una para cada caso, tal como se observa en la Figura.

En caso de que fuese necesario supervisar intersecciones (ver Figura 3.12), como puede ocurrir en entornos urbanos, podría llegar a ser necesario un sistema con tres Gaussianas, puesto que un vehículo que llegase a la intersección podría optar por girar a la derecha, a la izquierda, o continuar de frente. Sin embargo, dado que no se cuenta

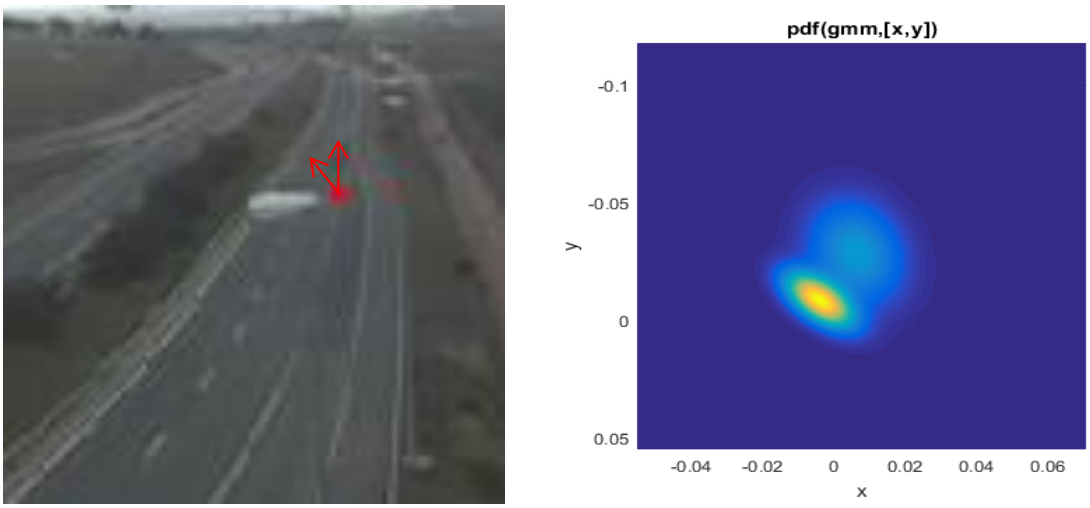

Fig. 3.11: píxel cercano a una incorporación que cumple las condiciones para ser modelado con dos componentes.

con estos casos en la base de datos y el coste computacional se incrementaría considerablemente, se ha optado por el modelo de dos Gaussianas.

El primer criterio para decidir si son necesarias dos Gaussianas es la relación entre el eje mayor y el eje menor de la elipse resultante de la proyección de la Gaussiana sobre el plano. Para ello, se obtienen los autovalores de la matriz de covarianza de la Gaussiana, $\sum_K$, ya que, como se ve en la Figura 3.13., el mayor de los autovalores se corresponde con el semieje mayor, y el menor de ellos, con el semieje menor. Si la relación entre ellos es mayor de 0.3 (es decir, si el semieje menor es mayor que 0.3 veces el semieje mayor), se considera que el píxel en cuestión es susceptible de ser modelado con una mezcla de dos Gaussianas. La elección de esta restricción, calculada empíricamente, se debe a que, en general, en un píxel la Gaussiana debería ser muy estrecha, puesto que la mayoría de los coches pasarán por un mismo punto con direcciones y velocidades similares. Solo en caso de que realmente haya dos posibles direcciones, la Gaussiana resultante será más ancha y podría ser, por tanto, caracterizado por una segunda Gaussiana.

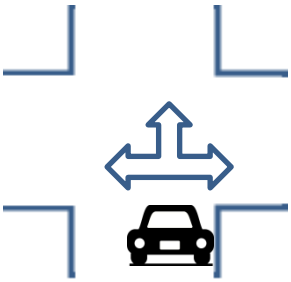

Fig. 3.12: posibles direcciones que puede tomar un vehículo en una intersección, susceptibles de ser asignadas a 3 Gaussianas distintas.

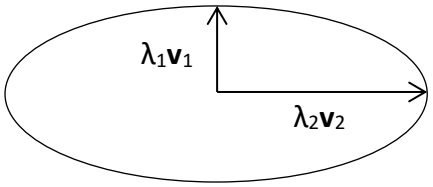

Fig. 3.13: autovalores y autovectores de una elipse

Para dicho píxel se comprueba a continuación si el modelo de dos Gaussianas aprendido realmente caracteriza mejor al píxel que el modelo de una Gaussiana. Para ello se tienen en cuenta dos aspectos: si su distancia de Bhattacharyya [23] es superior a 0.6 y si las probabilidades *a priori* π_1 y π_2 de las Gaussianas son ambas superiores a 0.005, definiéndose la distancia de Bhattacharyya como:

$$D_B = \frac{1}{8}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T \sum^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) \; + \frac{1}{2} ln \left(\frac{det\left(\frac{\sum_1 + \sum_2}{2}\right)}{\sqrt{det(\sum_1) det(\sum_2)}} \right) \qquad (25)$$

La elección de estos valores se debe, para el caso de la distancia de Bhattacharyya, a las observaciones realizadas sobre varios píxeles de varios modelos de carretera. Se calcularon sus distancias de Bhattacharyya llegando a la conclusión de que un valor superior a 0,6 supone un solape de las Gaussianas suficientemente grande como para no ser necesaria una segunda Gaussiana. Para el caso de las probabilidades *a priori*, se ha considerado que, si en un cierto punto ocurre un evento asociado a esa Gaussiana menos de una vez de cada 200 eventos, los datos que se utilizaron para modelar esa Gaussiana son outliers y no pueden considerarse representativos del comportamiento habitual.

Si se verifican ambas condiciones, se asume que la región de solapamiento entre las dos Gaussianas es suficientemente pequeña como para aceptar el modelo de dos Gaussianas y que ambas tienen una probabilidad suficientemente grande como para no ser consideradas despreciables (ver Figura 3.11). En caso contrario, se mantiene el modelo de una única Gaussiana, obteniendo finalmente un modelo estadístico para cada píxel de la imagen a tamaño reducido (uno para cada 16 píxeles originales). Para Gaussianas multivariantes, la distancia de Bhattacharyya se puede expresar como:

3.3. Detección de anomalías

La detección de anomalías constituye el último módulo para determinar si se ha de activar la alarma por anomalía o no. Es la parte más compleja del sistema y la que

mayor procesado requiere. En la siguiente Figura se ilustran sus etapas de funcionamiento:

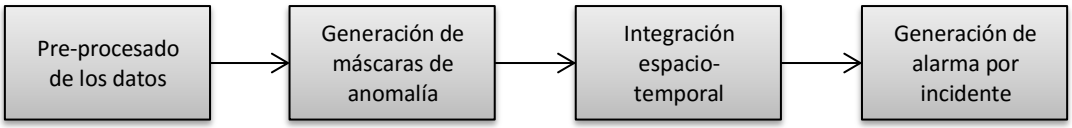

Fig. 3.14: ejemplo de frame con máscara de *foreground* quemada por cambio de iluminación en la escena

En primera instancia, se realiza un pre-procesado de los datos para eliminar zonas de la carretera en los que la estimación de movimiento es defectuosa y descartar frames cuya máscara de *foreground* asociada se encuentre "quemada". A continuación, se generan por umbralización dos máscaras binarias de anomalía por separado: una de píxeles anómalos y otra de vehículos lentos. Finalmente, se buscan regiones que solapen al menos un cierto porcentaje entre las máscaras de un frame y las del siguiente. Si esto ocurre durante un cierto número de frames consecutivos, se genera una alarma por anomalía.

3.3.1. Pre-procesado de los datos

Una vez entrenado el sistema y habiendo modelado estadísticamente el movimiento en cada píxel, se procede a la detección de anomalías. En primer lugar, debido a que los vectores de movimiento estimados en los bordes de la imagen no son demasiado precisos, se eliminan para la detección 5 píxeles de cada lateral y de la zona inferior de la imagen. Asimismo, en las zonas más alejadas de la imagen los vectores también son ruidosos y se complica la detección, generando falsas alarmas, por lo que se ha optado por obviar el 15% superior de la máscara de carretera.

También se comprueba antes de empezar con la detección si el frame de vídeo recibido tiene una máscara de *foreground* correcta o si se encuentra "quemada" por un cambio de iluminación (ver Figura 3.15). En caso de que la imagen esté quemada, ese frame se descarta, ya que se generarían falsas alarmas por supuestos vehículos lentos en las zonas quemadas (zonas en las que teóricamente hay un vehículo pero la estimación de movimiento es cero). Para ello, cuando el sistema detecta que al menos un 5% de píxeles en los que la máscara de carretera vale '0' están a '1' en la máscara de *foreground*, la ejecución del módulo de detección de anomalías se pausa hasta que el porcentaje caiga por debajo del 5%. De esta forma, se asegura que tan pronto como la máscara de *foreground* comience a quemarse, el sistema entre en suspensión, evitando las falsas alarmas.

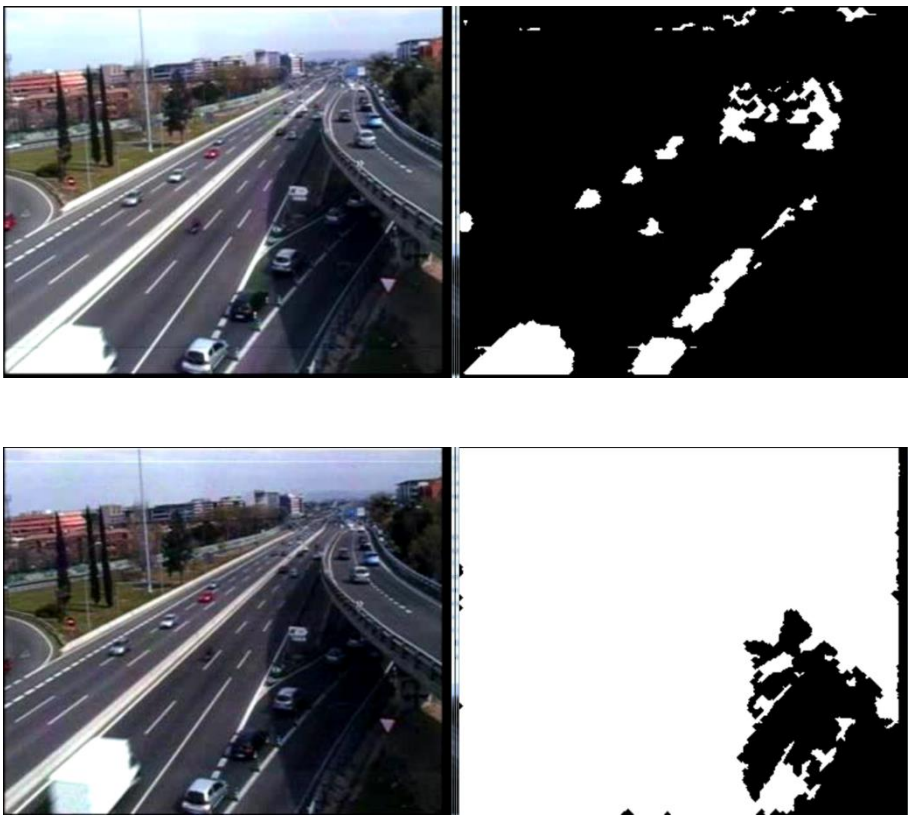

Fig. 3.15: ejemplo de frames consecutivos en los que la máscara de *foreground* resulta quemada por un cambio de iluminación en la escena, que puede apreciarse mejor en la zona superior de la imagen.

A continuación, se realiza la estimación de los vectores de movimiento sobre la imagen que se va a evaluar, se reducen a la cuarta parte y se referencian a la vista del modelo de cámara aprendido.

3.3.2. Generación de máscaras de anomalía

Para evaluar si aparecen anomalías en la imagen, se hace uso de dos submódulos:

- Detección de píxeles anómalos: en este submódulo se evalúan todos los vectores de movimiento estimados en las zonas de *foreground* sobre los modelos del píxel correspondiente. La verosimilitud de un modelo de Mezcla de Gaussianas multivariante como el que se emplea en este proyecto se define mediante la siguiente ecuación:

$$p_{\mathbf{u}}(\mathbf{u}|\boldsymbol{\mu}_i, \sum_i) = \sum_{i=1}^{M} \pi_i \frac{1}{2\pi|\sum_i|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{u} - \boldsymbol{\mu}_i)^T \sum_i^{-1} (\mathbf{u} - \boldsymbol{\mu}_i)\right) \qquad (26)$$

siendo **u** el vector de movimiento de componentes u y v, M el número de componentes del modelo de Mezcla de Gaussianas (en este caso, M=1 o M=2, dependiendo del píxel), $\boldsymbol{\mu}_i$ el vector de medias de la componente *i,* $\sum$ la matriz de covarianza de la componente *i*, y π_i el coeficiente de mezcla de la componente *i*.

Tras el análisis de las distribuciones, se concluyó que eran más abruptas de lo deseado para el problema de la detección. Para hacer más "suaves" estas distribuciones y, además, acotarlas entre 0 y 1, siendo 1 anomalía y 0 evento habitual, se les aplica la siguiente función:

$$p_{anom} = e^{-\gamma \cdot p_{\mathbf{u}}(\mathbf{u})} \tag{27}$$

siendo γ igual a 0.1 en nuestro caso particular.

En la Figura 3.16 se muestran los pasos para obtener la máscara binaria de píxeles anómalos. Una vez calculados los niveles de anomalía en cada píxel, se obtiene un mapa de anomalías de la imagen, que se corresponde con la imagen central de la Figura. A partir de él, y umbralizando para valores superiores a 0.9, se extrae la máscara binaria de píxeles considerados como anómalos. Sobre esta máscara es preciso aplicar postprocesado para eliminar pequeños píxeles anómalos independientes, ya que los vehículos ocupan un cierto número de

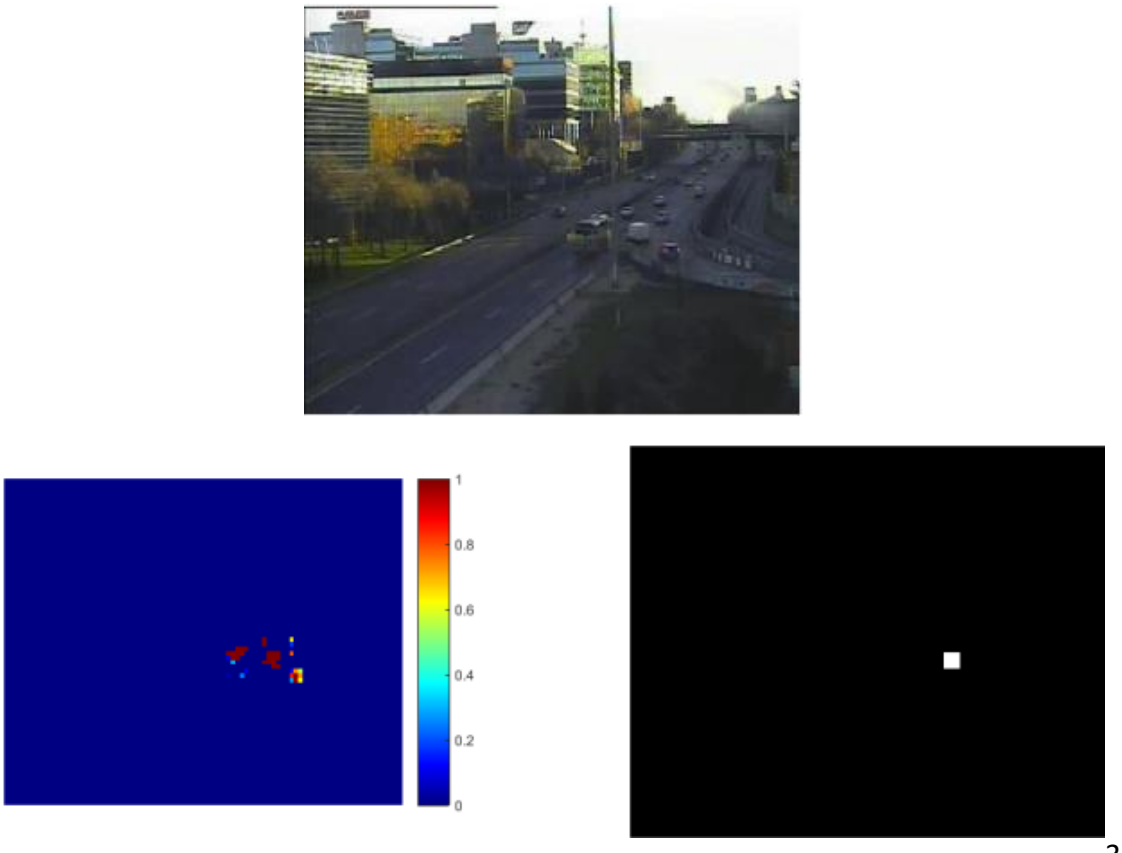

Fig. 3.16: vehículo con trayectoria anómala: mapa de anomalías y máscara binaria.

píxeles y, si realmente hay alguna anomalía, deberá aparecer un conjunto de píxeles cercanos anómalos, no píxeles aleatorios en la imagen. Se aplican por tanto operaciones de cierre y apertura consecutivas con un elemento estructurante de tipo cuadrado y lado igual a 3 píxeles para eliminarlos, resultando así la imagen de la derecha de la Figura.

- Detección de vehículos lentos o parados: la implementación de este submódulo responde a que la detección de vehículos con velocidad inusualmente lenta es también de gran relevancia, puesto que pone de manifiesto situaciones anómalas como tráfico elevado o vehículos detenidos por accidente. Es por ello que se optó por diseñar una detección de este tipo en paralelo.

  El funcionamiento es conceptualmente sencillo: hay que detectar aquellas zonas de la máscara de *foreground* y la máscara de carretera que se hayan mantenido durante un cierto tiempo y que, además, presenten unos vectores de movimiento inferiores a un cierto valor. En concreto, se buscan zonas de la máscara de *foreground* que solapen con la máscara de un frame anterior, con la máscara de 25 frames anteriores (un segundo de vídeo) y con la máscara de carretera. Esas zonas han de cumplir además la siguiente condición, cuyos parámetros se han obtenido de forma empírica:

$$||\mathbf{u}|| < \max\left(\min\left(\left||\boldsymbol{\mu}|\right| - 30||\boldsymbol{\sigma}||, 0.015\right), 0\right) \tag{28}$$

  Es decir, las zonas cuya media sea muy pequeña ya de por sí, no se tendrán en cuenta para la detección de vehículos parados, puesto que son zonas lejanas en las que los vectores son ruidosos y conducen a falsas alarmas. Aquellas zonas cuyas medias tengan valores intermedios, se considerará que hay vehículos parados si el módulo del vector de movimiento es inferior al módulo de la media menos 30 veces el módulo de la desviación típica. Para zonas con vectores más grandes (las más cercanas a la cámara), los vectores han de ser inferiores a 0.015. En píxeles modelados con una mezcla de dos Gaussianas, la media y la desviación típica elegidas son las de la Gaussiana a la que los datos pertenecen con mayor probabilidad *a posteriori*.

  Además, las zonas más lejanas son las que con mayor probabilidad tendrán un umbral igual a 0, lo cual complementa a la condición de suprimir el 15% superior de la máscara de carretera para evitar un gran número de falsas alarmas por vehículos lentos.

Se obtiene finalmente una máscara binaria de vehículos parados (ver Figura 3.17) a la que se le aplica el mismo postprocesado que a la máscara de píxeles anómalos y por el mismo motivo, solo que con un elemento estructurante de lado 2.

3.3.3. Integración espacio-temporal y generación de alarmas por anomalías de tráfico

Para activar la alarma de anomalía, se construye un contador que incrementa su valor en 1 en caso de que haya alguna anomalía en el frame y lo disminuye en 1 en caso de que el frame esté libre de anomalías, pudiendo tener valores entre 0 y 100. La forma de decidir si en un frame hay anomalía o no consta de varios pasos (ver Figura 3.18):

- En primer lugar se comprueba si hay píxeles a '1' en la máscara de píxeles anómalos. En caso afirmativo, se buscan aquellos píxeles anómalos que se han mantenido entre un frame y el anterior. Si en el frame anterior no hubo anomalías, esta máscara será la que se considere para el siguiente frame.

  Si el porcentaje de píxeles anómalos que se mantienen de un frame al siguiente es superior al 10%, se realiza en primer lugar el mismo postprocesado a la máscara de *foreground* que el que se hizo sobre la máscara de píxeles

Fig. 3.17: vehículo parado por accidente, máscara de *foreground* y máscara binaria asociada

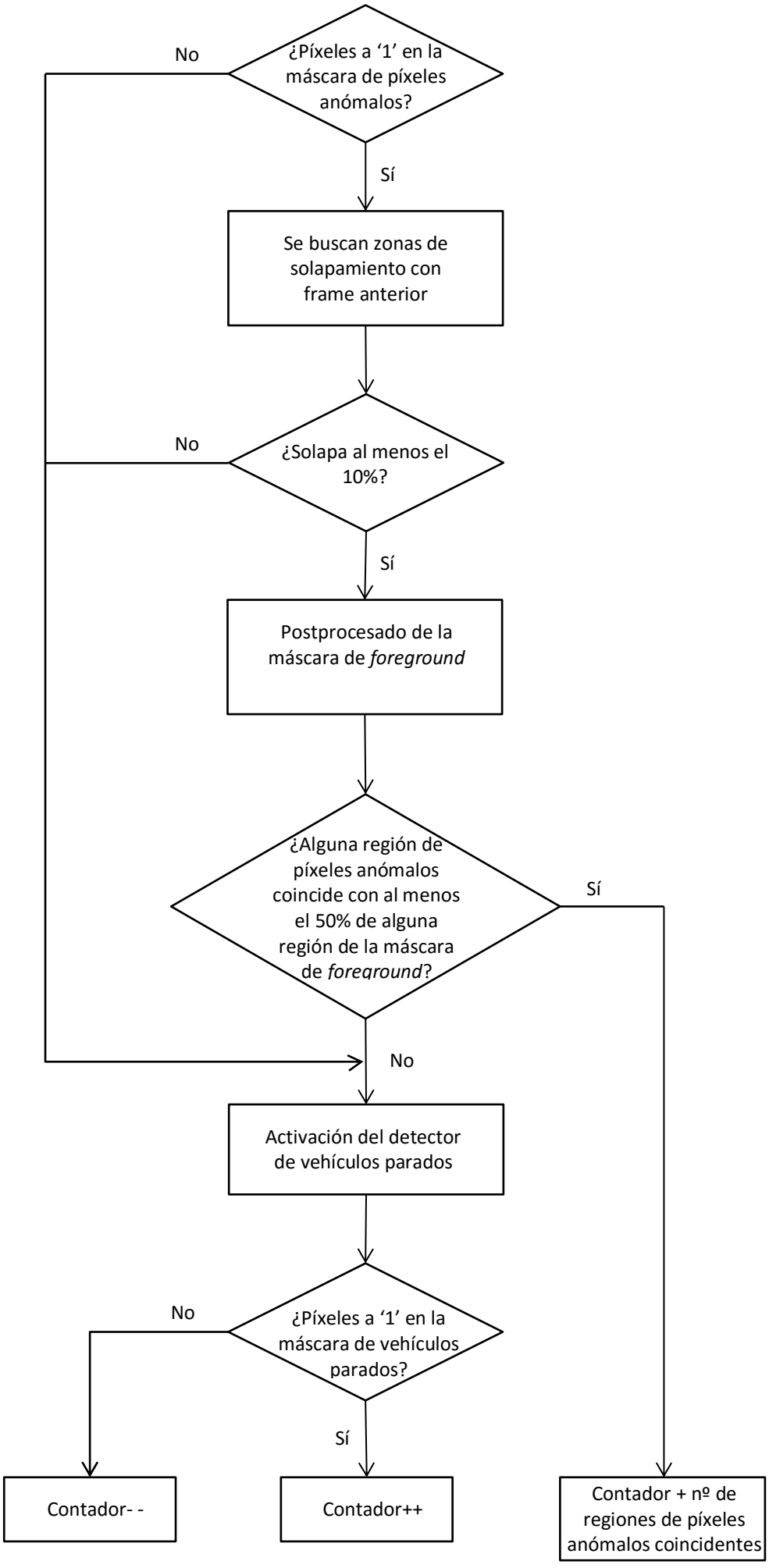

Fig. 3.18: Diagrama de flujo del funcionamiento del decisor de anomalía.

anómalos para eliminar píxeles independientes y tener la región de cada coche por separado. Se ha optado por un porcentaje pequeño para los píxeles anómalos para evitar que casos como la oclusión parcial de un vehículo, en los que la anomalía dejaría de aparecer durante unos pocos frames en pantalla, ocasionen un "parpadeo" en la máscara de anomalía que implique un decremento precipitado del contador.

A continuación se buscan regiones conectadas de la máscara de píxeles anómalos que supongan al menos el 50% de alguna de las regiones conectadas de la máscara de *foreground*, ya que si realmente hay algún vehículo con un movimiento anómalo, la mayoría de sus píxeles deberán ser anómalos, no solo una parte minoritaria de ellos. Esto permite solventar los casos en los que la parte superior de un vehículo "ocupe" un carril en dirección contraria (ver Figura 3.19) y el de los vectores ruidosos en la parte trasera de los vehículos (ver Figura 3.20), que podrían ocasionar alarmas por trayectorias anómalas.

- Si se detecta algún vehículo anómalo, el contador se incrementa en tantas unidades como vehículos anómalos se han detectado y se avanza al siguiente frame. En caso contrario, se activa el detector de vehículos parados.

- Si la máscara de vehículos parados contiene algún píxel a '1', el contador se incrementa en 1. En caso contrario, el contador se reduce en 1.

Se ha optado por incrementar el contador de anomalía por vehículo anómalo y no por píxel anómalo porque esto podría ocasionar que vehículos cercanos a la cámara (en primer plano), que ocupan mayor cantidad de píxeles, generasen más cantidad de anomalía que vehículos lejanos, cuando lo ideal es que ambas sean tratadas por igual.

Finalmente, se decide que hay anomalía cuando el contador se mantiene por encima de 70 durante al menos dos segundos.

Fig. 3.19: vehículo cuya parte superior obstruye el carril contrario

Fig. 3.20: vectores de movimiento ruidosos en la parte posterior del vehículo

4. Experimentos y resultados

4.1. Base de datos y protocolo de experimentación

Todos los experimentos realizados en este proyecto se han llevado a cabo sobre la base de datos proporcionada por la DGT. Esta base de datos consta de vídeos de 7 puntos distintos de carreteras españolas, obtenidas con cámaras colocadas sobre mástiles que permiten cambiar la inclinación y la dirección de apuntamiento según la voluntad del operario de cámara, así como variar el zoom de la imagen. Para cada cámara se cuenta con 11 franjas horarias, que van desde las 7 de la mañana hasta las 6 de la tarde, si bien precisamente las franjas límite (de 7 a 8, de 5 a 6) no se han utilizado debido a que el sistema original sobre el que se ha desarrollado este proyecto es impreciso en condiciones de baja luminosidad. Únicamente en una de las cámaras, las franjas horarias disponibles van desde las 7 de la mañana hasta las 2 de la tarde. Además, se dispone de 3 vídeos adicionales de 1 minuto cada uno en los que aparece un accidente de tráfico. La resolución de los vídeos proporcionados es de 352x288, que es una resolución pobre para realizar actividades de procesado de vídeo (ver Figura 4.1).

En cuanto al protocolo de experimentación, para evaluar la *estimación de movimiento* se ha realizado un visionado de diversas secuencias correspondientes a las diferentes vistas de cámara en los puntos kilométricos disponibles, y se ha llevado a cabo una comparación subjetiva de la calidad de la estimación. Además, se ha evaluado el tiempo de procesado de cada uno de los métodos evaluados.

Para el *modelado estadístico del flujo de movimiento* a nivel de píxel, la elección de los parámetros se ha realizado de forma empírica, analizando los resultados obtenidos y eligiendo el valor más apropiado en base a observaciones subjetivas.

Para la *detección de anomalías*, debido a la escasez de datos para el entrenamiento y

Fig. 4.1: ejemplos de cámaras pertenecientes a la base de datos del proyecto

prueba del sistema, se ha optado por emplear el método de la validación cruzada, de modo que cada fragmento de vídeo se evalúa en test considerando como datos de entrenamiento el resto de los fragmentos de vídeo que pertenezcan a la misma vista de cámara. La evaluación de los resultados se ha realizado comparándolos con aquellos obtenidos por el sistema previo desarrollado por el Grupo de Procesado Multimedia de la UC3M para la DGT, el cual analizaba las trayectorias de los vehículos capturadas usando algoritmos de tracking visual.

4.2. Experimentos sobre el módulo de estimación de movimiento

Para la estimación de movimiento se antojaban dos potenciales soluciones: estimación por bloques (sección 3.1.1) o Lucas y Kanade (sección 3.1.2). Además, esta estimación de movimiento debía limitarse únicamente a las zonas de la máscara de *foreground*. Si bien para la estimación por bloques fue sencillo limitar los cálculos a la zona de *foreground*, para Lucas y Kanade, que hace uso del gradiente, no lo fue tanto, llegando a obtener un funcionamiento más lento e impreciso que aplicándolo sobre toda la imagen. Se compararon, por tanto, los resultados de calcular la estimación por bloques directamente sobre las zonas de *foreground* y los resultados de aplicar Lucas y Kanade a la imagen completa y multiplicarlo a continuación por la máscara de *foreground*.

Los parámetros utilizados para la estimación por bloques, cuya definición se puede encontrar en el capítulo 3.1.1., fueron:

- BSL = 3
- SRL = 5
- $\sigma = 1$
- $\lambda = 10$

siendo 2*BSL+1 la longitud del bloque de referencia $\mathbf{I}_0(\mathbf{x})$ centrado en un cierto píxel $\mathbf{x}$; 2*SRL+1 el lado del rango de búsqueda SR; σ la desviación estándar del prefiltrado gaussiano de las imágenes; y λ el parámetro que penaliza las soluciones de módulo grande. La elección de los parámetros se debe a que se consideran valores suficientemente grandes de acuerdo a los movimientos que se esperan en las imágenes a procesar, sin comprometer el tiempo de cómputo.

Los parámetros utilizados para Lucas y Kanade, definidos en 3.1.2., fueron:

- r = 15
- $\sigma = 1$

- thrLK = $2x10^{-5}$

siendo r el radio del círculo que define el vecindario Ω alrededor de un píxel; σ la desviación estándar del prefiltrado gaussiano de las imágenes; y thrLK el umbral para considerar que la estimación de movimiento ha sido satisfactoria. La elección de los parámetros se debe a que así se consigue aumentar en gran medida la robustez de la estimación a cambio de perder precisión, pero ésta puede ser compensada multiplicando los resultados por la máscara de *foreground*.

Los resultados obtenidos en los experimentos se presentan a continuación, con los tiempos de cómputo promedio en milisegundos para cada cámara:

Carretera	Tiempo BB (ms)	Tiempo LK (ms)
A-2 PK 4.5	52.14	32.69
A-2 PK 10.7	74.37	36.78
A-2 PK 21.1	120.74	36.68
A-2 PK 135	62.87	32.29
A-2 PK 136.2	36.53	37.06
A-4 PK 62	41.30	27.41
M-607 PK 30.8	23.18	27.12
Promedio	58.73	32.86

Tabla 4.1: tiempos de cómputo promedio en milisegundos por frame para realizar la estimación de movimiento por bloques y por Lucas y Kanade para cada uno de los vídeos de la base de datos.

En esta tabla se puede apreciar claramente que la estimación de movimiento mediante el algoritmo de Lucas y Kanade presenta unos tiempos de cómputo muy inferiores a los de la estimación por bloques en la mayoría de los casos, además de ser más constantes. Esto se debe a que, en términos generales, el algoritmo de estimación por bloques es más lento, pero además, para esta aplicación en particular, se han realizado ciertas modificaciones sobre el algoritmo que explican la variabilidad de los tiempos de dicha estimación.

Puesto que se tiene conocimiento *a priori* de las zonas en las que ha habido movimiento gracias a la máscara de *foreground*, únicamente se realiza la estimación de movimiento sobre dichos píxeles, evitando así tener que calcularla en zonas innecesarias y dando lugar a una fuerte variación de tiempos en función de la máscara de *foreground* de cada instante.

En Lucas y Kanade, por contra, se realiza la estimación de movimiento sobre toda la imagen y a continuación se multiplica la matriz con los vectores estimados por la

máscara de *foreground* correspondiente, puesto que, en este caso, limitar *a priori* la operación del gradiente es más complejo. Así, los tiempos son más constantes.

Esto explica también que en carreteras con poco tráfico, como es el caso de la A-2 PK 136.2 o la M-607 PK 30.8, los tiempos de ambos algoritmos sean similares e incluso inferiores para la estimación por bloques, ya que realmente hay muy pocos píxeles por frame en los que realizar dicha estimación por bloques mientras que en Lucas y Kanade la estimación se sigue realizando para toda la imagen.

Con respecto a la calidad del seguimiento, Lucas y Kanade también presenta ciertas ventajas que se pueden apreciar de forma subjetiva en la Figura 4.2 y que se exponen a continuación:

- Mayor precisión en situaciones de oclusión parcial, como ocurre en (a) tras el árbol que aparece en primer plano.
- Más robustez en la estimación en los bordes de la imagen, como se aprecia en (b) en el coche de la derecha.
- Mejor estimación en la parte trasera de los vehículos y en zonas homogéneas, como se ve en (c) en las partes posterior y superior del autobús.
- Mejor funcionamiento en situaciones de tráfico denso, como en (d), obteniendo como resultado vectores más robustos.
- Reducción de vectores ruidosos ocasionados por el propio ruido de la imagen, como ocurre en (e) en la parte inferior derecha de la imagen.

Por todo lo anterior, se ha optado por emplear el algoritmo de Lucas y Kanade para la estimación de movimiento de este proyecto.

4.2.1. Limitaciones del sistema de Estimación de Movimiento

Como ya se ha mencionado en varias ocasiones en puntos anteriores, una de las principales limitaciones del sistema se encuentra en las zonas más alejadas. Estas zonas no son tenidas en cuenta por el sistema debido a la escasa precisión de las máscaras y la baja resolución de las cámaras, que no permiten realizar unas estimaciones de movimiento adecuadas.

Dada la falta de resultados de módulos previos, tampoco se puede garantizar su correcto funcionamiento en periodos nocturnos. Asimismo, en entornos urbanos no ha sido probado y es probable que el sistema no se adapte correctamente a ese tipo de situaciones (cruces a modelar con 3 Gaussianas, paradas en semáforos o stops que no

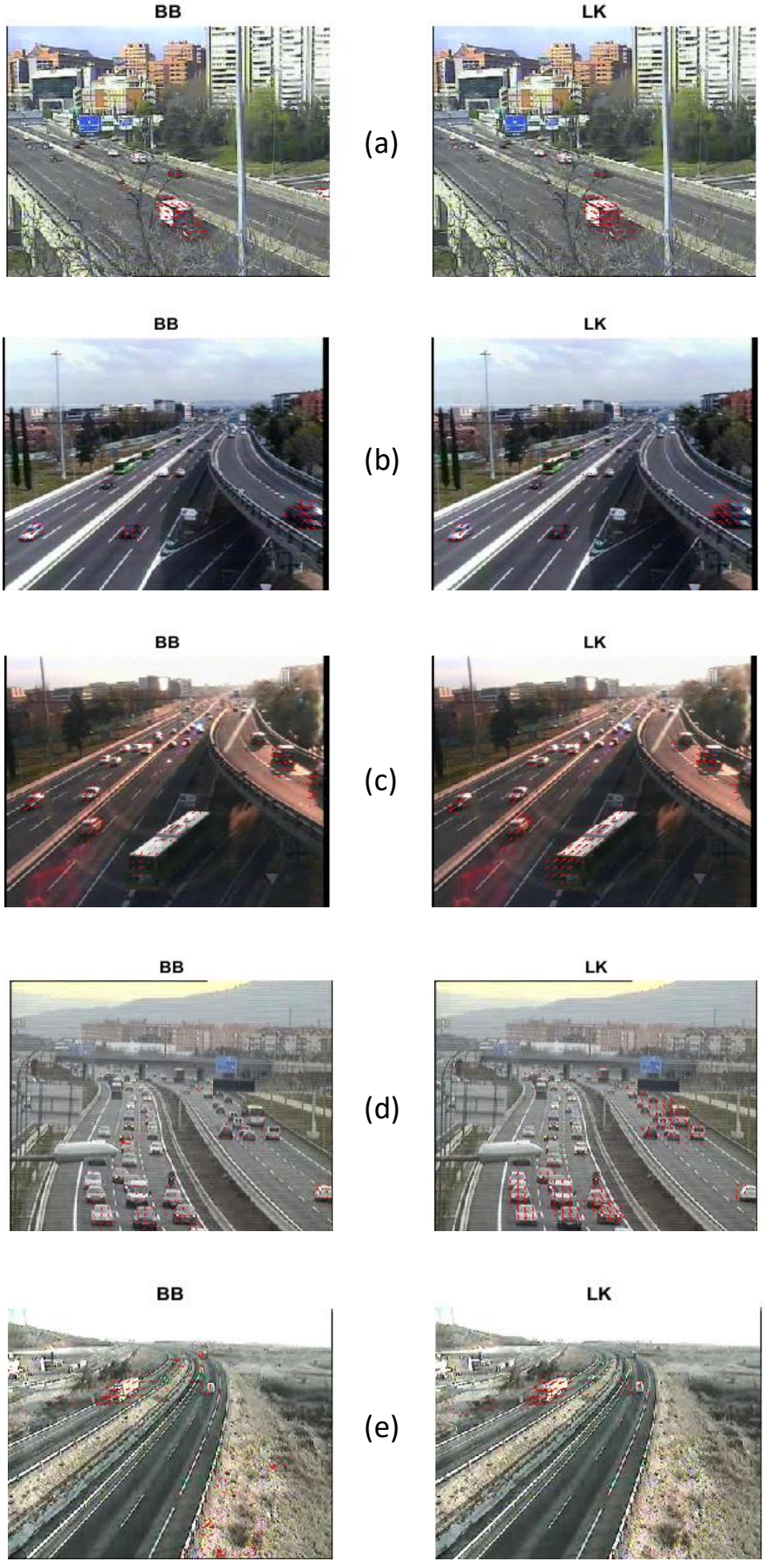

Fig. 4.2: comparación subjetiva de estimación por bloques y Lucas y Kanade sobre diversos vídeos

deberían ser detectadas como atascos…). Tampoco se han realizado pruebas en condiciones meteorológicas desfavorables. Si bien no debería suponer un problema, convendría comprobar el comportamiento de los vectores de movimiento ante secuencias de esas características.

Por último, cabe mencionar que el tiempo de aprendizaje de los modelos es variable en función de la carretera. En carreteras con tráfico escaso, el sistema podría requerir varios días para aprender correctamente el modelo. Sin embargo, en carreteras más transitadas, podría ser suficiente con unas horas. También existe el riesgo de que el sistema aprenda de una situación anómala. Por ejemplo, si un día el tráfico es lento debido a un percance en la carretera, eso será lo que el sistema considerará como normal, y cuando el tráfico se normalice, podría ocurrir que se detectase como anomalía. Es necesario, por tanto, una mínima supervisión para evitar esta situación.

4.3. Experimentos sobre el sistema global: detección de anomalías

En este apartado se estudiará el comportamiento del sistema completo de detección de anomalías. Para ello, se analizará cuantitativa y cualitativamente los resultados obtenidos sobre la base de datos disponible, comparándolos también con los del sistema previo, tratando de identificar las fortalezas y debilidades del nuevo sistema.

En las Figuras 4.3 y 4.4 se presentan los resultados obtenidos con el sistema previo y con el actual, respectivamente. Cada gráfica corresponde a una cámara distinta (un punto kilométrico distinto) en el periodo utilizado para la experimentación. En aquellas cámaras en las que se disponía de un vídeo de un minuto con un accidente de tráfico, los resultados se han concatenado al final de la gráfica etiquetados como “test”. En verde se indica el nivel de anomalía instantáneo en un determinado frame; en azul, el umbral por encima del cual, si se mantiene el nivel de anomalía instantánea durante un cierto tiempo, se activa la alarma de anomalía; y en rojo, los instantes considerados como anomalía. Estas anomalías deberán aparecer en las franjas de “test”, por los accidentes de tráfico, y también en otras franjas horarias en las que se produzcan retenciones.

Como se puede observar en la Figura 4.4, los tres segmentos de test en los que se producen accidentes son detectados con éxito por este nuevo sistema. Asimismo, en la A-2 PK 4.5 se han detectado incidentes por tráfico lento que han llegado a ocasionar retenciones importantes. En las Figuras 4.5, 4.6, 4.7 y 4.8 se pueden encontrar los mapas de anomalía de estos incidentes, así como la máscara binaria resultante.

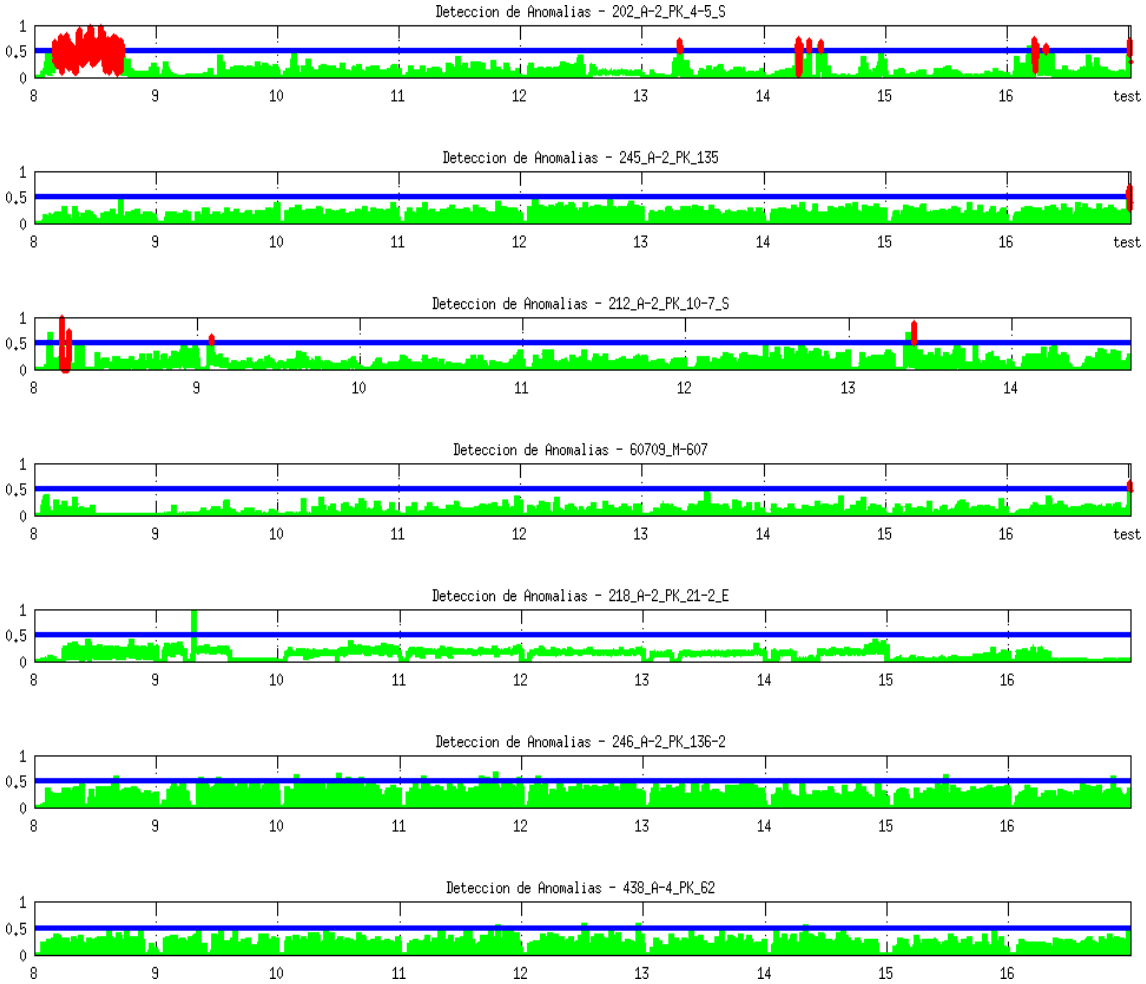

Fig. 4.3: resultados finales del sistema anterior

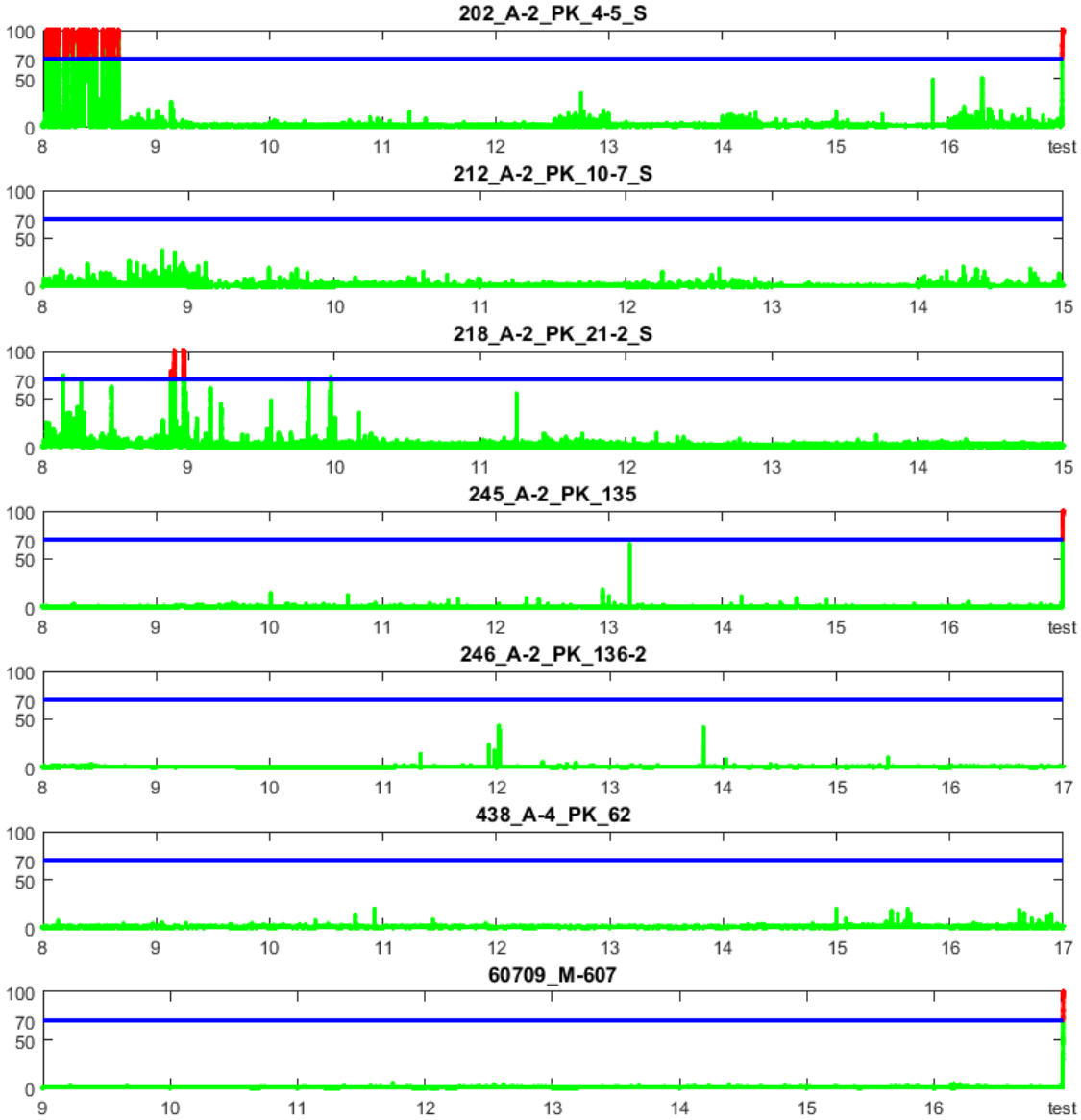

Fig. 4.4: resultados finales obtenidos en todos los segmentos de vídeo disponibles. Se ha considerado como umbral de decisión TH=70 y se muestran en rojo los incidentes detectados como anomalía.

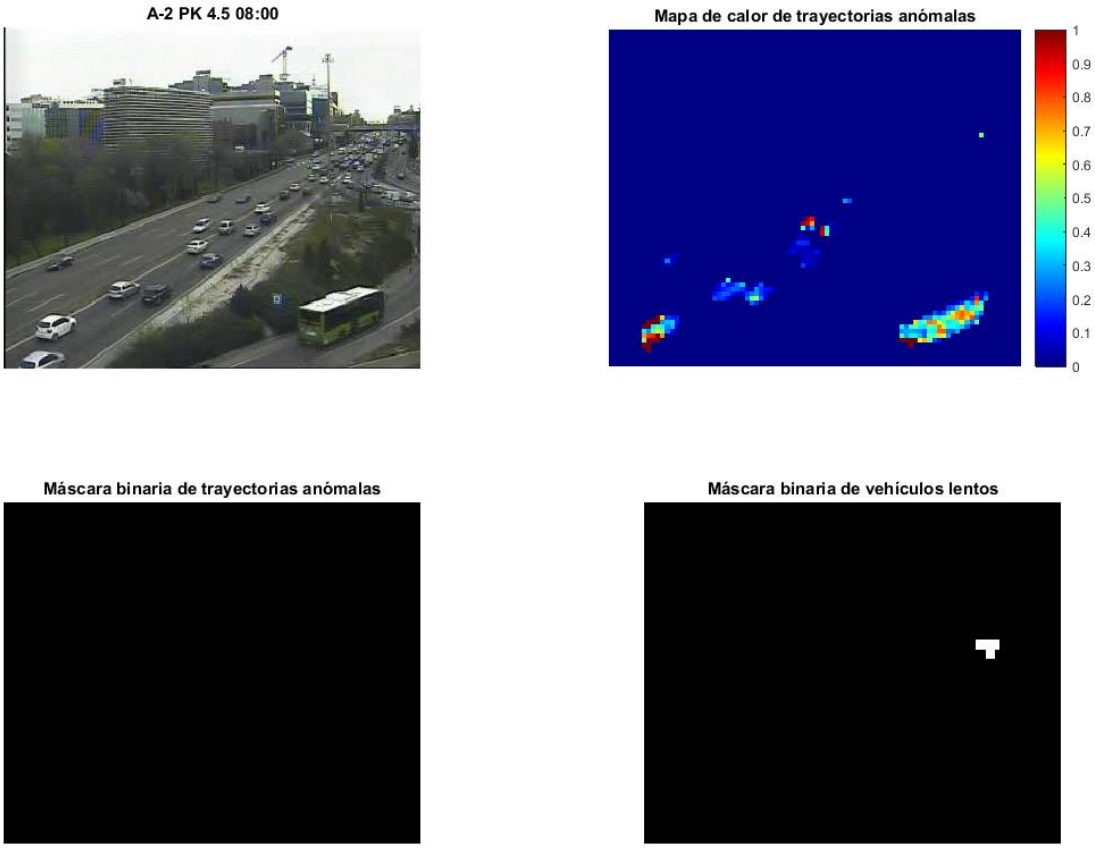

Fig. 4.5: retenciones en la A-2 PK 4.5 a las 08:00.

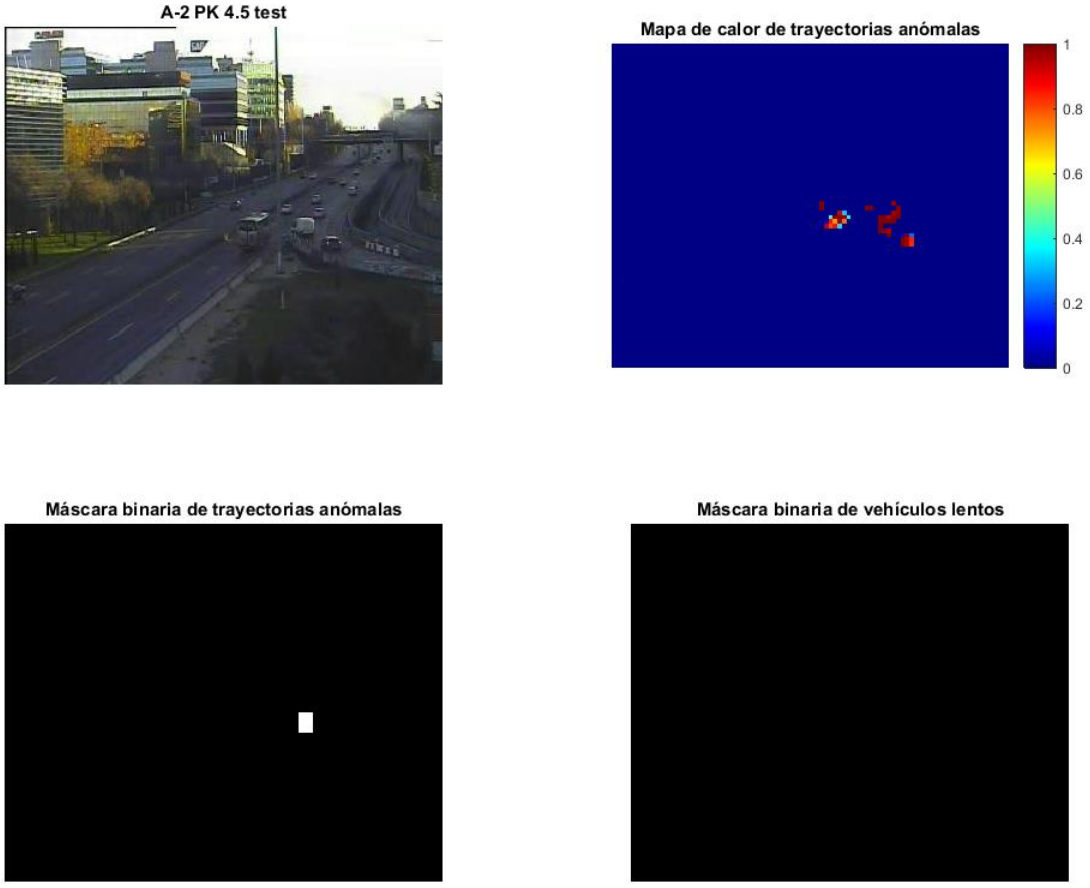

Fig. 4.6: accidente en la A-2 PK 4.5.

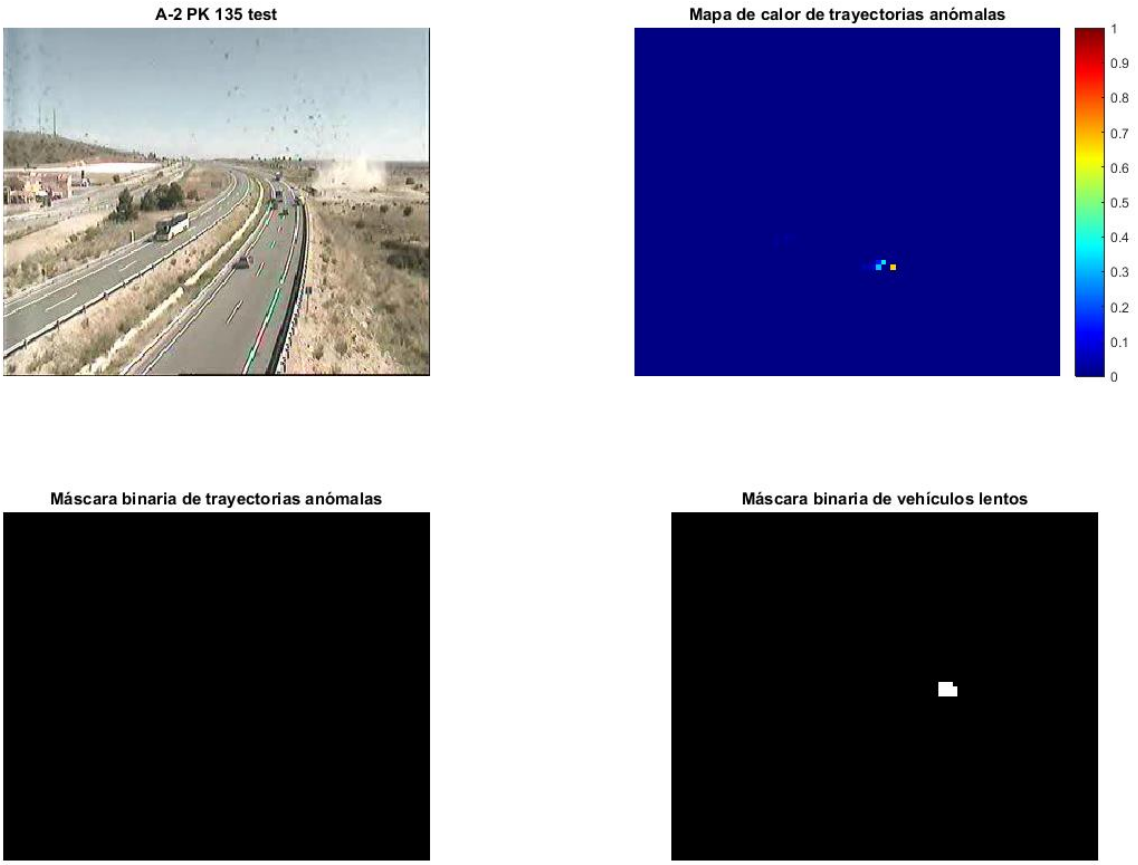

Fig. 4.7: accidente en la A-2 PK 135.

Fig. 4.8: accidente en la M-607.

Cabe destacar que, para el vídeo de test de la A-2 PK 135, en el que un camión arrolla a una furgoneta en la cuneta y éste se sale de la carretera, el sistema detecta el accidente a partir de la detención de los vehículos que vienen por detrás. Esto se debe a que el sistema desarrollado se centra únicamente en la detección de incidentes en zonas de carretera. No obstante, el sistema previamente desarrollado por el grupo incluye un módulo de detección de incidentes en los alrededores que no depende de las trayectorias de los vehículos y, por lo tanto, podría integrarse a nuestros algoritmos como un submódulo complementario, reduciendo el retardo en la detección de incidentes de este tipo.

Comparando los nuevos resultados con los del sistema de referencia, se puede observar que en ambos casos los vídeos de test (aquellos en los que ocurre algún accidente de tráfico) son detectados como anomalía, así como las retenciones de larga duración entre las 8 y las 9 de la mañana de la A-2 PK 4.5. Sin embargo, otra serie de retenciones más puntuales no son detectadas por nuestra propuesta debido a que la máscara de *foreground* desaparece cuando un vehículo permanece inmóvil durante un cierto tiempo, no pudiendo realizar la estimación de movimiento en esas regiones. Este error se analiza en el último punto del apartado 4.4.

En la siguiente tabla se muestra una comparación cuantitativa entre ambos sistemas. En ella, se expresan los resultados obtenidos en forma de porcentaje de incidentes detectados para cada uno de los tipos de incidentes disponibles en la base de datos. Cabe destacar que los datos para las retenciones son aproximados, puesto que, a diferencia de los accidentes (etiquetados como “test”), no se dispone de una base de datos etiquetada.

Tipo de incidente	Sistema de referencia	Sistema implementado
Accidente	100%	100%
Retenciones	90%	60%

Tabla 4.2: comparativa, en porcentaje, de incidentes detectados con éxito por ambos sistemas, desglosada por tipo de incidente.

Cabe también mencionar la mejora en cuanto a tiempo de cómputo del nuevo sistema frente al antiguo. Con este nuevo sistema, cada frame de vídeo se procesa en aproximadamente 0.25 segundos, con tráfico mixto y ejecutado sobre CPUs sin paralelizar ninguno de los procesos. Con el sistema anterior, cada frame de vídeo requería de aproximadamente 1.2 segundos de cómputo, si bien este tiempo sufría variaciones importantes en función del número de vehículos en escena. Esto supone una importante reducción del tiempo de cómputo además de unas oscilaciones de

tiempo mucho menores, lo que permitiría su implementación en tiempo real haciendo uso de GPUs y de paralelización.

Por último, destacar que este nuevo algoritmo de detección permite dotar de mayor robustez al sistema. Este hecho puede apreciarse en situaciones de normalidad, como ocurre en las carreteras A-2 PK 136.2 y A-4 PK 62, en las que apenas sucede nada durante las nueve horas de vídeo y, por tanto, el nivel de anomalía debería ser muy bajo casi permanentemente. Mientras que esto sí se ha conseguido con el nuevo sistema, no ocurre así con el sistema de referencia, en el que los niveles de anomalía se acercan de forma constante al umbral de detección.

En la tabla 4.3 se presenta una comparativa de la distancia normalizada entre el nivel de anomalía y el umbral de detección de ambos sistemas como medida de robustez. Cuanto más cercanos a '1' sean los valores, más robusto será el sistema, pues querrá decir que en situaciones de normalidad, el contador de anomalía permanece más cercano a '0'. Para ello, se han excluido los segmentos de "test", en los que se sabe que hay anomalía, y se asumen condiciones de normalidad durante el resto del vídeo, dado que no disponemos de una base de datos etiquetada (aunque en ciertos instantes el sistema ha demostrado que no es así, se puede asumir que esos instantes tienen una duración despreciable en comparación con la duración total del vídeo). En aquellos instantes donde el valor de anomalía supera al umbral, la distancia se considera cero. Finalmente, se normaliza la distancia por el valor del umbral.

Carretera	Distancia media al umbral normalizada en el sistema implementado	Distancia media al umbral normalizada en el sistema de referencia
A-2 PK 4.5	0.9608	0.8557
A-2 PK 10.7	0.9851	0.8719
A-2 PK 21.1	0.9750	0.9502
A-2 PK 135	0.9969	0.8956
A-2 PK 136.2	0.9888	0.8915
A-4 PK 62	0.9877	0.8971
M-607 PK 30.8	0.9868	0.9502
Promedio	0.9830	0.9017

Tabla 4.3: comparativa de robustez entre ambos sistemas a través de la distancia al umbral normalizada, desglosada por carretera y en nivel promedio.

Como se puede observar en las medidas, este nuevo sistema es capaz de mejorar la robustez del sistema de referencia en todas las carreteras de la base de datos, obteniendo unos valores muy cercanos a '1' en todos los casos.

4.4. Análisis de errores y mejoras del sistema

Tras evaluar las prestaciones del sistema, se han detectado una serie de errores puntuales causados por pequeñas incompatibilidades con los módulos previos pertenecientes al proyecto realizado con anterioridad por el departamento. Algunos de ellos se han podido solucionar *a posteriori* y otros no. Estos errores se detallan a continuación:

- Falsas alarmas en la A-2 PK 21.2 en la franja entre las 8 y las 9: estas alarmas se deben a que el módulo de estabilización de cámara no funcionó correctamente y el fuerte viento presente en esos instantes provocó una oscilación vertical de la cámara que dio lugar a una estimación errónea de los vectores de movimiento. Este error pudo ser solucionado restando a los vectores de movimiento de las zonas de *foreground* la mediana de los vectores de movimiento obtenidos en zonas de gradiente no nulo y no pertenecientes a la máscara de *foreground*. Se realizaron pruebas con otros estadísticos como la media o la moda, pero finalmente se optó por la mediana por ser la que mejores resultados arrojaba.

  Se efectuaron pruebas con esta nueva extensión, obteniendo resultados favorables en todos los casos. En la Figura 4.9 se adjunta el error de la A-2 PK 10.7 corregido.

- Picos de anomalía por supuestos vehículos lentos en la A-2 PK 21.2, 135 y 136.2: en ciertas ocasiones se pueden generar pequeñas falsas alarmas por vehículos lentos como consecuencia de una mala extracción de la máscara de *foreground*. Estas situaciones se producen cuando un vehículo con un color

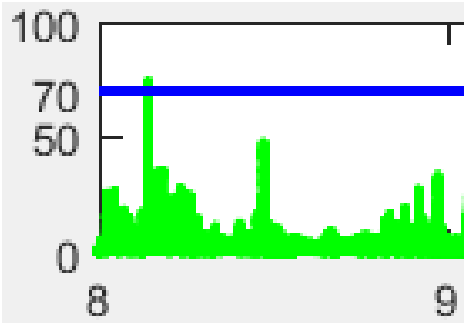

Fig. 4.9: gráfico resultante de aplicar la corrección de mediana a los vectores de movimiento en el segmento de 8 a 9 de la carretera A-2 PK 10,7

muy claro produce un alto contraste con el color de la carretera, activando el módulo de modelado y sustracción de background, y aprendiendo durante unos instantes un background erróneo, que da lugar consecuentemente a una máscara de *foreground* errónea. Es lo que ocurre en la Figura 4.10, en la que los vehículos blancos dejan a su paso un rastro de máscara de *foreground* que no se corresponde con ningún vehículo. Este hecho hace que en esa zona, en la que teóricamente debería haber un vehículo, se obtengan unos vectores de movimiento prácticamente nulos y se detecte que hay vehículos lentos cuando en realidad no es así. Este error no se ha logrado solventar, puesto que requeriría la modificación del funcionamiento de módulos previos ajenos al proyecto. En cualquier caso, no supone un problema grave, puesto que a los pocos segundos la máscara desaparece no llegando a generar alarma en la mayoría de los casos.

- No detección de vehículos parados: en algunos casos, como en la franja entre las 8 y las 9 de la A-2 PK 10.7, el sistema no es capaz de detectar vehículos totalmente parados. Esto se debe a que vehículos de la imagen permanecen inmóviles durante un cierto tiempo, la máscara de *foreground* comienza a desvanecerse, puesto que el módulo de modelado y sustracción de *foreground* las aprende como parte del background. Esto hace que al principio el contador se incremente pero que, en cuanto los vehículos permanecen estáticos durante un periodo largo de tiempo, la máscara desaparezca y el contador comience a disminuir. Este error tampoco se ha podido solventar, puesto que requeriría nuevamente cambiar el funcionamiento de módulos ajenos al proyecto.

Fig. 4.10: rastro de máscara de *foreground* ocasionada por el paso de vehículos

5. Conclusions and future lines

5.1. Conclusions

In this project, we have developed a traffic anomaly detection system in road environments. Differently from other previous systems, in this case the approach is made through motion flows modelling instead of individual vehicle tracking, which lightens the process and deals with some other inconveniences. It also improves other proposals based as well in motion flows modelling in the sense that it can deal with a much broader set of events. In addition, it is also capable of adapting its models to all possible geometric scenarios.

Most of objectives proposed at the beginning have been achieved, since a robust anomaly detection system that can react to events such as slow traffic or anomalous trajectories has been successfully developed. This situation, together with the fact that motion estimation is a low-cost computational process, allows its implementation in real time. However, even if the objective of identifying stopped vehicles has been in general achieved, there are still few problematic situations in which the *foreground* mask vanishes and vehicle detection cannot be performed. The solution would require a revision of previous modules, out of the scope of this project.

It is also important to remark that the main contributions stated in section 1.5. have presented successful results, since both the automatic decider for the number of components of a GMM and the detector of anomalies (and the module for efficient storage of estimated motion vectors described in Annex II) have worked in an appropriate manner. These are the modules that really suppose an innovative contribution in the detection field.

Moreover, the system has been trained over a real database, so it could be directly applied to real life given that it has been proved a proper functioning. Nevertheless, due to the small size of this database and, more concretely, to the short number of incidents that permit testing the algorithm, it is very possible that the system is overfitted. This implies that some situations can't have been tested, like the usefulness of modelling a certain pixel with a second Gaussian in scenarios like crossings or forks, where the second component could be actually advantageous, or the theoretical inconvenience of being a memoryless system, that, even if in general it has not supposed a big deal, evaluating situations like the ones exposed at the beginning of Section 3 would have been an interesting contribution.

All this must be considered in the extracted conclusions related to illumination, other type of incidents, other scenarios, etc., since the lack of these circumstances only allows a general prediction about them. To obtain more precise conclusions, further experiments over a bigger database should be performed in order to get a more realistic approach.

5.2. Further Work

Next are presented some possible lines of investigation that could be performed in the future making use of this system and its results:

- In order to improve training and testing of the system and re-adjust all needed parameters, a database enlargement could provide interesting results.
- Testing the system over adverse climatological conditions and night scenes would permit to confirm the initial suspicions and enlarge the application areas, even if for night scenes it would also be required to re-design previous modules that are used in this project.
- An implementation of the system in an urban environment, after re-adjusting some of the parameters, would be interesting too in case of proper functioning and would imply an important upgrade in the features of the system.
- The rollout of the proposed system in closed environments, such as race tracks, for accident detection would also provide a powerful tool for the improvement of medical services response time.
- Finally, global computing times could be reduced by implementing the system in a different programming language more efficient in terms of computational time, like C or C++. Moreover, the system would also improve its performance if certain processes were parallelized making use of GPUs, since both in statistical modelling module and in anomaly detection module, it is necessary to go over the whole image pixel by pixel. This process is quite inefficient if it is performed linearly, but it can be easily solved with parallelism and would allow its implementation in real time.

6. Presupuesto y planificación

6.1. Costes materiales

- Lugar de trabajo: se incluyen todos los gastos del espacio de trabajo empleado durante el desarrollo del proyecto, como pueden ser la electricidad, la limpieza o el mantenimiento. El coste estimado asciende a 2000€ al mes, pero al estar compartido con otras 11 personas, queda reducido a 166,67€. La duración del proyecto ha sido de aproximadamente 6 meses, por lo que esta cantidad asciende a 1000€.
- Ordenador personal: se ha hecho uso de un ordenador personal valorado en 600€ para la programación del sistema, el estudio previo y la realización de la memoria. Su coste se puede amortizar en 4 años, por lo que el coste en 6 meses asciende a 75€.
- Ordenadores de procesamiento: para la ejecución de los algoritmos más complejos, se han empleado dos equipos de cómputo intensivo del grupo de investigación GPM. Puesto que son equipos de uso compartido por todo el grupo, el coste se estima en 200€.
- Software: la programación del sistema se ha realizado sobre Matlab R2013b de MathWorks, cuyo precio para facultades es de 500€. Puede ser amortizado en 4 años, con lo que su precio estimado es de 62,50€.

Descripción	Coste
Lugar de trabajo	1000€
Ordenador personal	75€
Ordenadores de procesamiento	200€
Software	62,50€
TOTAL	1337,50€

Tabla 6.1: coste material total del proyecto desgranado por elementos utilizados.

6.2. Costes de personal

- Jose Antonio Peláez Escobar: la dedicación al proyecto ha sido de 4 horas diarias, 5 días a la semana, durante 6 meses. Esto da lugar a un total de 480 horas, que, con unos honorarios de 8.75€/hora, hacen un total de 4200€.
- Iván González Díaz: encargado de la dirección del proyecto. El tiempo invertido en tutorías y correcciones asciende a 60 horas. Teniendo en cuenta unos honorarios de 20€/hora, el coste asciende a 1200€.

Persona	Honorarios	Horas	Total
Jose Antonio Peláez Escobar	8.75€/hora	480	4200€
Iván González Díaz	20€/hora	60	1200€
TOTAL			5400€

Tabla 6.2: coste personal total del proyecto desgranado por persona.

6.3. Presupuesto total

Descripción	Coste
Costes materiales	1337,50€
Costes personales	5400€
Subtotal	6737,50€
IVA (21%)	1414,875€
TOTAL	8152,38€

Tabla 6.3: coste total del proyecto desgranado por tipos de costes más IVA.

El presupuesto total asciende a OCHO MIL CIENTO CINCUENTA Y DOS EUROS Y TREINTA Y OCHO CÉNTIMOS.

6.4. Planificación temporal

En la tabla 6.4 se presentan de forma resumida las actividades desarrolladas a lo largo de los seis meses de la duración del proyecto. La distribución temporal de dichas actividades se puede encontrar en la Figura 6.1 mediante un diagrama de Gantt.

ID	Actividad	Duración (semanas)
A	Estudio de la propuesta y lectura de bibliografía relacionada	2
B	Primeras pruebas y valoraciones de la estimación de movimiento de vehículos sobre la base de datos	1
C	Diseño del sistema de almacenamiento eficiente de vectores de movimiento estimados	3
D	Lectura de bibliografía relacionada con los modelos GMM	1
E	Primeras pruebas de modelado estadístico de los píxeles con modelo GMM con aprendizaje *batch*	1
F	Lectura de bibliografía relacionada con el algoritmo EM para la actualización online de Gaussianas	1
G	Diseño de la solución para el modelado estadístico del flujo de movimiento a nivel de píxel con un modelo GMM con dos componentes y actualización online mediante el algoritmo EM	4
H	Diseño de una primera versión del módulo de detección de anomalías	2
I	Adaptación del sistema a los movimientos de cámara efectuados manualmente por un operador	2
J	Rediseño y resolución de errores en base a los resultados obtenidos en el módulo de detección de anomalías	7
K	Redacción y correcciones de la memoria	8

Tabla 6.4: actividades desarrolladas en el proyecto, identificador asociado y duración en semanas.

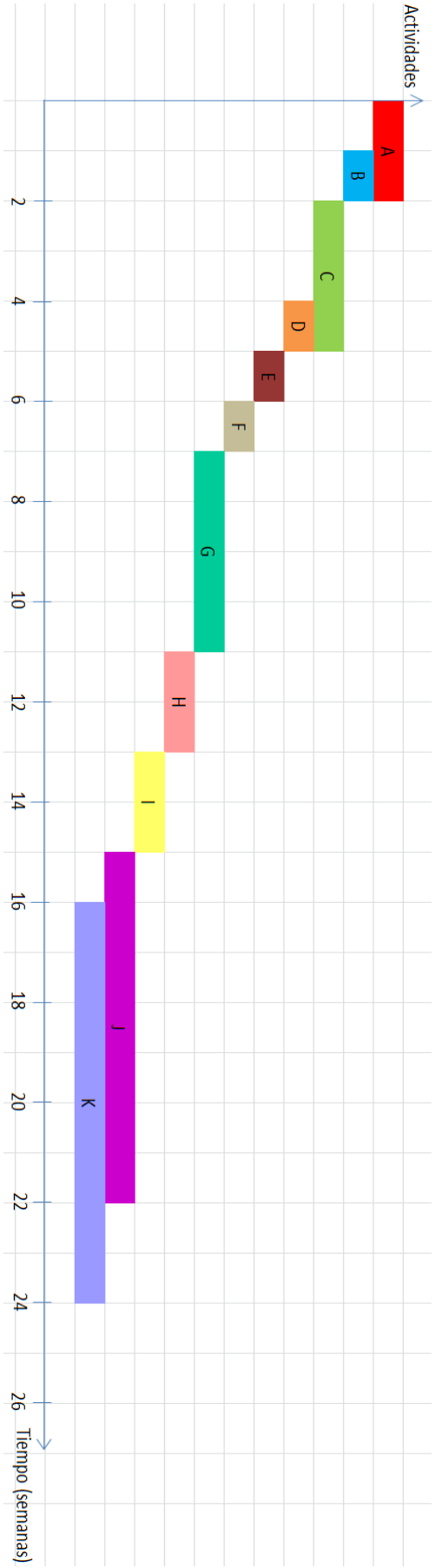

Figura 6.1: diagrama de Gantt asociado al proyecto.

Anexo I: Extended abstract

The aim of this project is to develop a system capable of automatically detecting incidents on DGT cameras based on motion flows. As opposed to most proposals up to this moment, the idea here is not to track independent vehicles to learn their common trajectories, but to learn the general traffic flow on each pixel of the road, without caring of each individual vehicle.

This project is embedded in a bigger one developed by Grupo de Procesado Multimedia of Carlos III University of Madrid. The pipeline previously proposed and the new one proposed here can be observed on Figure I.1. The first system was based on vehicle tracking making use of particle filter. This way, it can be learnt which are the normal trajectories and then, any trajectory that is not classified as normal would be considered as anomaly.

Nevertheless, even if this system has many advantages, it also has some disadvantages. In particular, it presents some troubles when there is a lot of traffic on the scene, and so there are many vehicles to be tracked at the same moment and all of them very close to each other, and the computational time the particle filter requires is quite high, what implies a difficult implementation of the system on real time.

This project proposes, then, a different anomalies detector based on motion estimation. The reason to face the problem from this point of view is that, for this particular purpose, there is no need to track every single vehicle, since it would be enough with learning the general traffic flow of the road. Moreover, computational cost of motion estimation is much lower than for particle filter, so real time implementation would be possible in this case.

I.1. Proposed system

The new system that is proposed here is composed of three modules, as it can be seen in Figure I.1. The experimental database is composed of videos of Spanish highways from 8:00 to 18:00. Each of the modules is explained in detail in the following sections.

I.1.1. Vehicle motion estimation

The objective of this module is to estimate motion vectors of vehicles to allow a posterior statistical modeling of the road. For this purpose, Lucas&Kanade algorithm

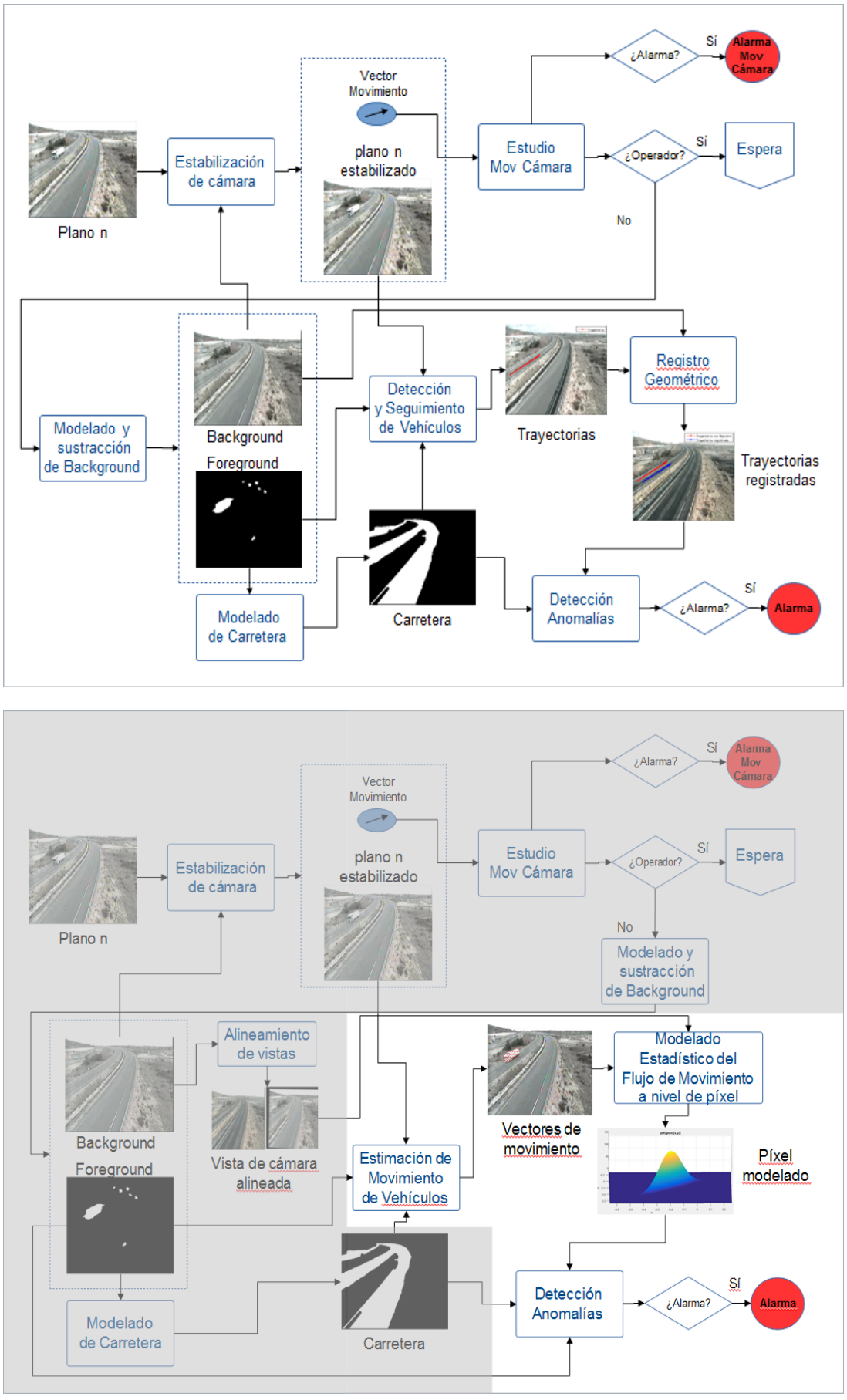

Fig. I.1: above, pipeline proposed on the previously developed project. Below, new pipeline proposed, emphasizing modules that are strictly developed on this project

has been employed, since it presents some advantages with respect to block-based algorithm. To take this decision, some experiments were performed and the results on [20] were taken into account.

In Lucas&Kanade, a tradeoff between robustness and precision must be done. The bigger the searching neighborhood, the more robust but the less precise the estimation is. However, in this module the outputs of the previous modules are already available. In particular, from Modeling and Background subtraction module, it is obtained a *foreground* mask, which permits to identify vehicles on the scene. Then, it is possible to select a bigger value for the neighborhood radius and lose some precision because this handicap can be balanced out by multiplying the map of estimated motion vectors by the *foreground* mask, which will limit the estimation strictly to the vehicles. This situation can be observed on Figure I.2.

Finally, vehicles' motion vectors are obtained with a neighbor radius equals to fifteen.

I.1.2. Statistical modeling of motion flow with pixel precision

In this section, the idea is to statistically model every pixel on the road mask of the image according to the estimated motion vectors. For this purpose, a multivariate Gaussian Mixture Model (GMM) has been employed. In most cases, a single Gaussian

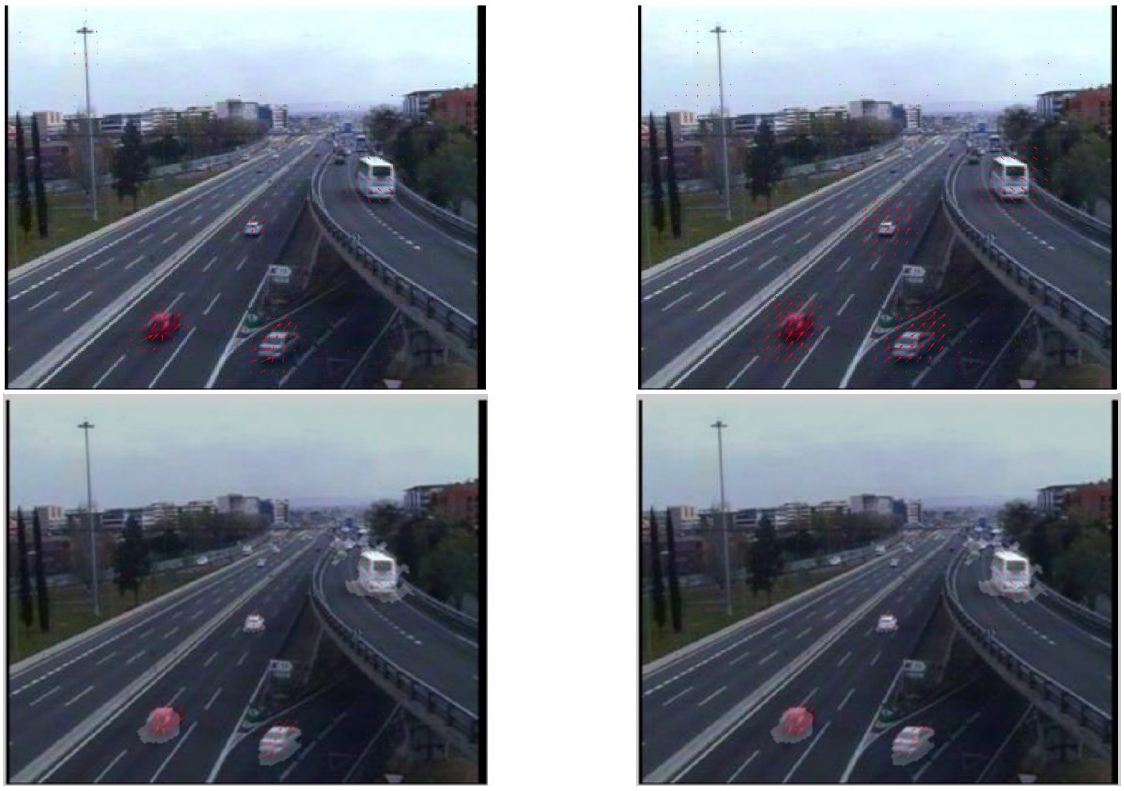

Fig. I.2: resultant vectors from motion estimation with L&K algorithm. Left, r=5, Right, r=15.Above, without applying *foreground* mask. Below, applying *foreground* mask.

would be enough to model a pixel, since, in general, in a point of the road cars pass with more or less the same direction and more or less the same speed, so the variance of the Gaussian should be able to absorb these small variations. Nonetheless, there might be some cases, like acceleration lanes or exits, where a second Gaussian would be needed.

In order to cover all possible cases, when the system is being trained, both models with one and two Gaussians are learnt simultaneously for each pixel. Then, the criterion to decide if a pixel is finally modeled with one or two Gaussians has several steps:

- First, it is assumed that all pixels can be modeled with a single Gaussian.
- Then, it is checked which Gaussians are wide enough to be candidates of being modeled with a second Gaussian. The decision is based on the ratio of the semi-axis of the elliptical projection of the Gaussian. If the semi-minor axis is larger than a third of the semi-major axis, then this Gaussian is considered to be wide.
- Next, for those wide Gaussians, it is verified if the model with two Gaussians is really better than the one single Gaussian model. The conditions that must be verified to keep the two Gaussians model are:
    - None of the priors of the Gaussians can be smaller than 0.005 (the event happens at least 1/200 times).
    - The Bhattacharyya distance between both Gaussians can't be smaller than 0.6. Bhattacharyya distance measures the overlap between Gaussians. If the distance is bigger than 0.6, it is assumed that Gaussians are far enough one from each other and then both of them provide useful information. In Figure I.3 appears a pixel that accomplishes all these criteria.
    - If one of these conditions is not satisfied, the pixel remains modeled with only one Gaussian.

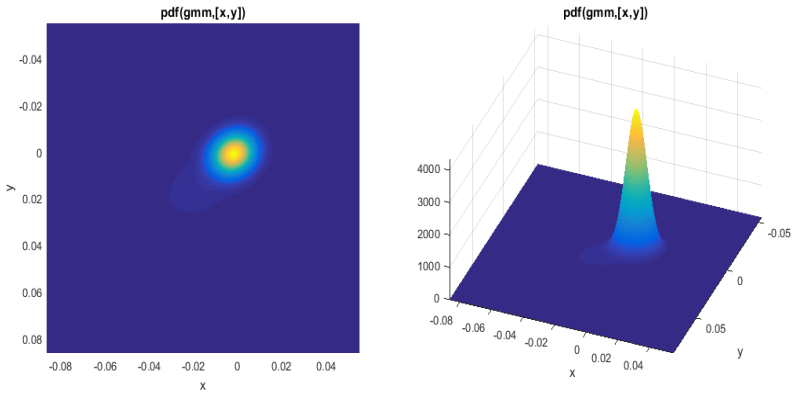

Fig. I.3: two component Gaussian mixture on a road pixel that accomplishes criteria defined in section I.1.2.

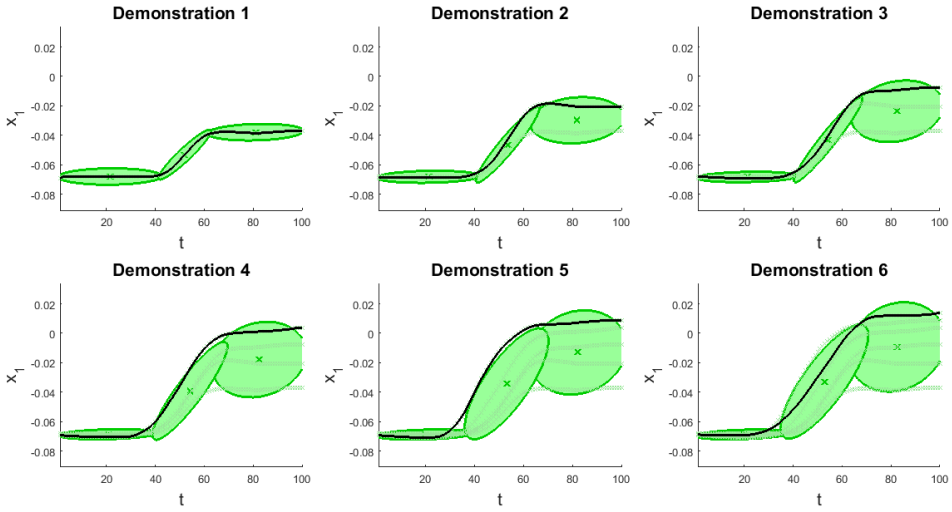

Fig. I.4: example of Gaussians update for new incoming data with direct update method from Calinon, S. code. In this case, trajectories are modeled with a mixture of three Gaussians and the system is updated with every new incoming trajectory.

In this scenario, where data arrives in real time for an infinite time, Gaussians cannot be learnt at once from the beginning. Then, there must be performed an online learning, that, in this case, is a modification of the *Expectation-Maximization* (EM) algorithm, that allows to re-compute mean, covariance and priors of the Gaussian as new data arrives. In Figure I.4 there is an example of trajectories being modeled with a GMM with three components. The black line represents a trajectory and each graph shows how the model is adapted to every new incoming trajectory.

Once every pixel on the road mask is modeled, detection can be performed.

I.1.3. Anomaly detection

This module is in charge of deciding whether an anomaly is detected or not. It is the most complex module and is the one that requires the most processing operations.

Once that the system has been trained and every pixel is properly modeled, every new frame that arrives can be tested to check if there is any anomaly. Before activating the anomaly detection module itself, several steps must be performed:

- First, since image borders produce noisy motion vectors and in the furthest zones, vectors are tiny and noisy, it is decided to suppress the 15% of the road mask from the upper part and 5 pixels of the image from left and right side and from the bottom.

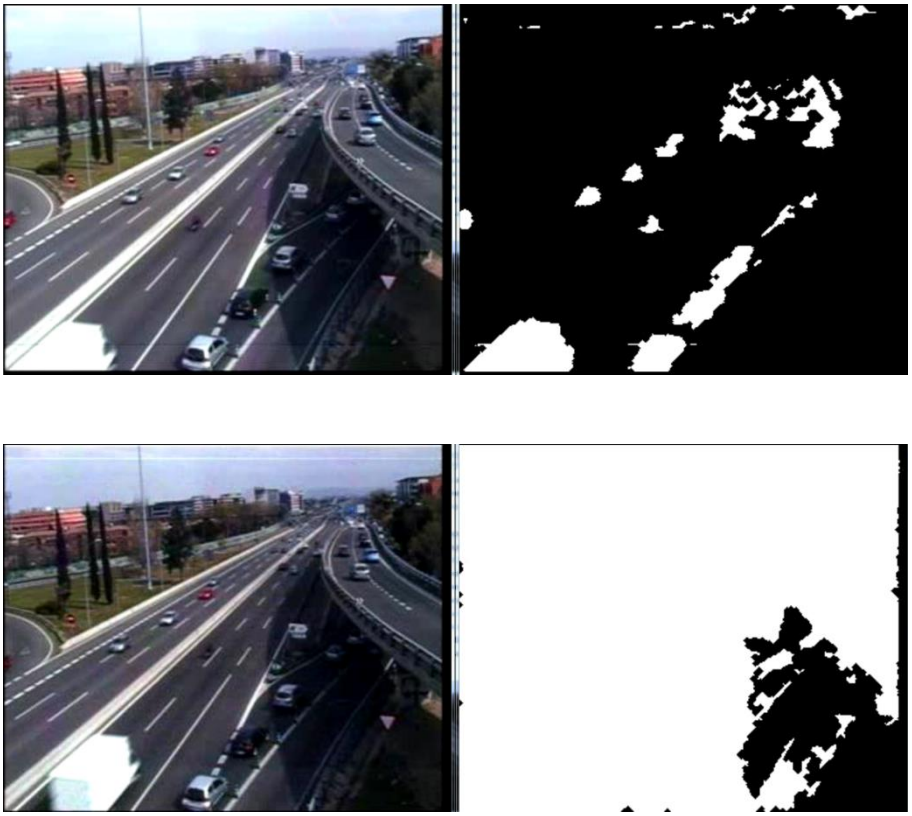

Fig. I.5: two consecutive frames whose *foreground* mask is overexposed due to an illumination change on the scene, that can be better observed on the top part of the image.

- Then, a prevention system has been developed to avoid cases in which *foreground* mask is overexposed (see Figure I.5). If this occurs, system remains in standby until overexposure disappears. If there is no overexposure, anomaly detection module starts.

Once it has been verified that the received frame is correct, anomaly detection starts:

- First, motion vectors belonging to the *foreground* mask are evaluated at its corresponding GMM. This way, depending on the value obtained when evaluating the likelihood function, $p_u(\mathbf{u})$, it can be decided whether the estimated vector is abnormal for that pixel or not. To obtain softer values of the likelihood function, it is applied the operation $e^{-\gamma \cdot p_u(\mathbf{u})}$, where ɣ=0.1. The result is an image whose values go from 0 to 1, where 1 means 'anomaly' and '0' means 'normality'. This image is converted to a binary mask by applying a threshold of 0.9, so that an anomaly binary mask is obtained.
- Then, slow or stopped vehicles are detected. For this purpose, another binary mask is built. This mask is the result of the logical AND operation of the actual *foreground* mask, the previous *foreground* mask, the 25th previous *foreground*

mask (since it is assumed that, if vehicles are slow, *foreground* mask one second before must be quite similar to actual *foreground* mask), the road mask and the slow vehicles mask. This slow vehicles mask is defined as:

$$||\mathbf{u}|| < \max\left(\min\left(\left|||\boldsymbol{\mu}|\right| - 30\left|||\boldsymbol{\sigma}|\right|, 0.015\right), 0\right) \quad (29)$$

where **u** is the motion vector at a certain pixel, $\boldsymbol{\mu}$ is the mean of the Gaussian of the pixel, and $\boldsymbol{\sigma}$ is the standard deviation of the Gaussian of the pixel.

Finally, a series of operations that can be summarized in Figure I.6 are performed:

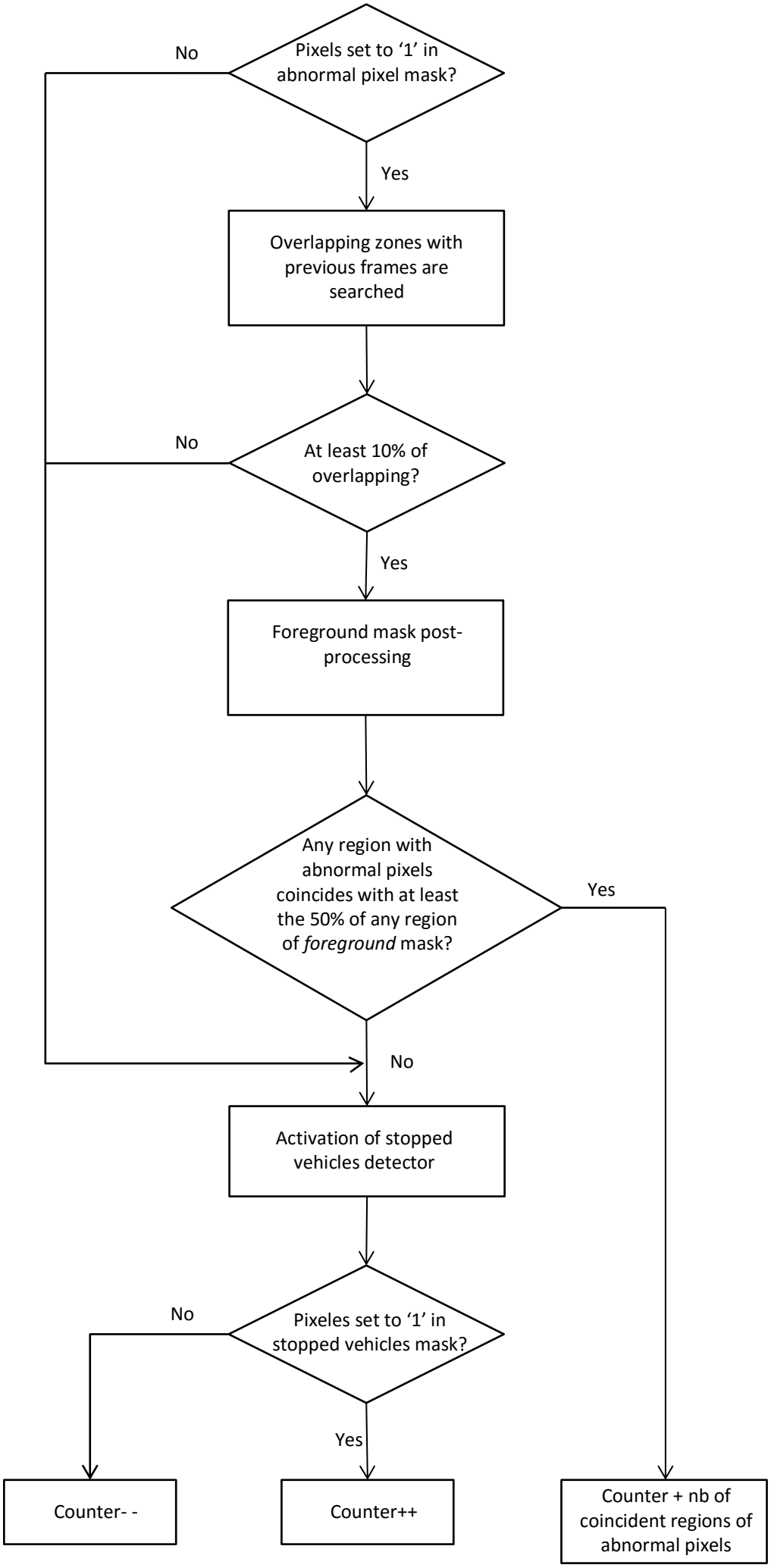

Fig. I.6: flow diagram of how anomaly detection module works

I.2. Final results

Below are presented some of the final results, where it can be seen that anomaly detection is properly performed. Both test segments, where accidents occur, and A-2 PK 4.5 incident between 8 and 9 AM, where there is some traffic jam, are detected, obtaining very similar results to those presented in the project developed by GPM at UC3M. Moreover, computing time is considerably reduced.

In addition, it supposes an innovation in anomalies detection, since until now, most of the approaches had been performed by means of vehicle tracking. This opens new investigation lines on the field, such as extending the system to night hours or urban roads.

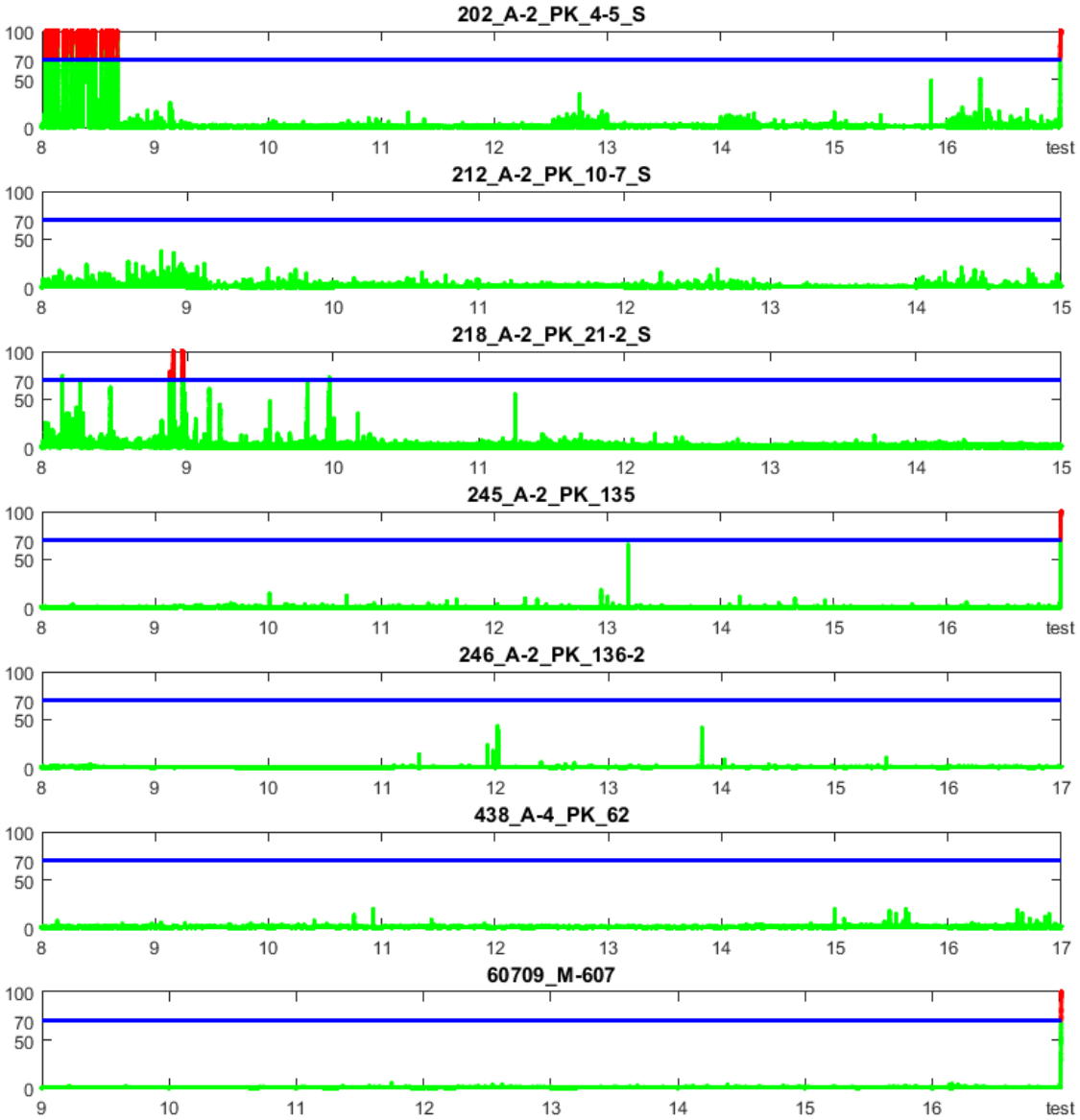

Fig. I.7: final results obtained over all available video segments. A threshold of 70 has been considered and in red are shown those incidents detected as anomalies.

Anexo II: Almacenamiento de vectores de movimiento estimados

Adicionalmente se ha diseñado un módulo de almacenamiento eficiente de vectores de movimiento estimados para reducir los tiempos de trabajo. Este módulo se ubicaría entre el módulo de la estimación de vectores de movimiento y el modelado estadístico del flujo de movimiento a nivel de píxel. Si bien no es necesario para el funcionamiento del sistema, se explica brevemente a continuación, puesto que ha sido empleado durante el trabajo de investigación.

El funcionamiento de este módulo se basa en el almacenamiento persistente de los vectores de movimiento estimados con una cuantificación no homogénea, ya que en las zonas de carretera más alejadas, con vectores de movimiento más pequeños, se requerirá una precisión mucho mayor que en las zonas más cercanas, con vectores mayores. Para ello, durante los primeros 5 minutos de vídeo se realiza un aprendizaje de la media y la varianza del módulo de los vectores de movimiento de cada píxel. En aquellos píxeles en los que no se hayan obtenido datos durante esos 5 minutos, se les asigna la mayor desviación típica de entre sus vecinos más próximos. Así, el módulo de todos los vectores de movimiento se limitará a 5σ y el escalón de cuantificación será de 10σ/256, siendo éste diferente para cada píxel.

El almacenamiento de los vectores de movimiento se realiza mediante la creación de un vídeo, asignando la coordenada *u* del vector al canal R del vídeo y la coordenada *v* del vector al canal G. El canal B se mantiene siempre a 128. Para realizar el almacenamiento es necesario aplicar previamente la cuantificación a los vectores, puesto que éstos toman valores reales y en un vídeo todos los píxeles han de tener un valor sin decimales entre 0 y 255. Como las coordenadas de los vectores pueden ser positivas o negativas, se ha decidido emplear los valores entre 0 y 127 para las coordenadas con negativas, el 128 para los vectores nulos y los valores entre 129 y 255 para las coordenadas positivas. Así, se aplica a cada coordenada la siguiente ecuación:

$$output = \frac{(128 - input \geq 0) \cdot input}{5\sigma} + 128 \qquad (30)$$

donde *output* es el valor a almacenar en el vídeo; *input* es el valor de la coordenada; e *input ≥ 0* será '1' cuando la desigualdad se cumpla y '0' cuando no. En la Figura II.1 se puede apreciar el resultado final.

Para recuperar los vectores de movimiento, basta con aplicar la operación inversa:

$$input = \frac{(output - 128) \cdot 5\sigma}{(128 - output \geq 128)} \quad (31)$$

y añadirle un pequeño ruido blanco centrado en el valor recuperado que lo haga oscilar ±1/3 el escalón de cuantificación. De esta forma se consigue compensar en cierto modo la perdida de realismo de los datos causada por la cuantificación, aunque no sean exactamente los mismos valores.

Fig. II.1: fragmento de vídeo y vídeo con los vectores de movimiento asociados almacenados.

Anexo III: Algoritmo EM

El algoritmo EM permite hallar la máxima verosimilitud de un modelo de Mezcla de Gaussianas de una forma relativamente sencilla. A continuación se muestra el funcionamiento del algoritmo y el desarrollo de unas ecuaciones básicas para su comprensión.

Se ha de tener en cuenta en primer lugar que en el máximo de la función de verosimilitud ha de verificarse la condición:

$$\frac{\partial \ln(p(\mathbf{U}|\boldsymbol{\mu}_k,\sum_k))}{\partial \boldsymbol{\mu}_k} = 0 \tag{32}$$

$$\frac{\partial}{\partial \boldsymbol{\mu}_k}(\sum_{n=1}^{N} ln\{\sum_{k=1}^{K} \pi_k N(\mathbf{u}_n|\boldsymbol{\mu}_k,\sum_k)\}) = 0 \tag{33}$$

$$\sum_{n=1}^{N} \frac{\pi_k N(\mathbf{u}_n|\boldsymbol{\mu}_k,\sum_k)}{\sum_j \pi_j N(\mathbf{u}_n|\boldsymbol{\mu}_j,\sum_j)} \sum_k (\mathbf{u}_n - \boldsymbol{\mu}_k) = 0 \tag{34}$$

$$\sum_{n=1}^{N} \gamma(z_{nk}) \sum_k (\mathbf{u}_n - \boldsymbol{\mu}_k) = 0 \tag{35}$$

siendo **U** una matriz en la que en la n-ésima fila se encuentra el vector de observaciones $\mathbf{u}_n^T$, y $\boldsymbol{\mu}_k$ la media, $\sum_k$ la matriz de covarianza y π_k, los coeficientes de mezcla de la Gaussiana asociada.

Si se multiplica ahora por $\sum_k^{-1}$ y se trata de reorganizar la ecuación, se obtiene:

$$\sum_{n=1}^{N} \gamma(z_{nk}) \boldsymbol{u}_n - \boldsymbol{\mu}_k \sum_{n=1}^{N} \gamma(z_{nk}) = 0 \tag{36}$$

$$\boldsymbol{\mu}_k = \frac{1}{\sum_{n=1}^{N} \gamma(z_{nk})} \sum_{n=1}^{N} \gamma(z_{nk}) \boldsymbol{u}_n \tag{37}$$

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \boldsymbol{x}_n \tag{38}$$

siendo

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk}) \tag{39}$$

De estas ecuaciones se pueden extraer dos aspectos relevantes:

- Que N_k hace referencia al número de puntos asignados a cada cluster *k*,

- Y que la media μ_k de la componente *k* se corresponde con la media de todos los puntos del conjunto de datos, como en el caso de una única Gaussiana, pero ponderada por la probabilidad *a posteriori* $\gamma(z_{nk})$ de que $\mathbf{x}_n$ pertenezca a la componente *k*.

De forma similar, y derivando $\ln(p(\mathbf{U}|\boldsymbol{\pi}, \boldsymbol{\mu}, \sum))$ con respecto de $\sum_k$, se obtiene:

$$\sum_k = \frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_{nk})(\boldsymbol{u}_n - \boldsymbol{\mu}_k)(\boldsymbol{u}_n - \boldsymbol{\mu}_k)^T \tag{40}$$

en la que, de nuevo, se tiene la covarianza de una única Gaussiana pero ponderada por la correspondiente probabilidad *a posteriori* y dividida por el número de puntos asociados a cada componente.

Igualmente, derivando $\ln(p(\mathbf{U}|\boldsymbol{\pi}, \boldsymbol{\mu}, \sum))$ respecto de los coeficientes de mezcla π_k y teniendo en cuenta, a través de un multiplicador de Lagrange, que la suma de dichos coeficientes ha de ser igual a 1, se obtiene:

$$\frac{\partial}{\partial \boldsymbol{\pi}_k}\left(\ln(p(\boldsymbol{U}|\boldsymbol{\pi}, \boldsymbol{\mu}, \sum) + \lambda(\sum_{k=1}^{K}\pi_k - 1)\right) = 0 \tag{41}$$

$$\sum_{n=1}^{N}\frac{N(\boldsymbol{u}_n|\boldsymbol{\mu}_k, \sum_k)}{\sum_j \pi_j N(\boldsymbol{u}_n|\boldsymbol{\mu}_j, \sum_j)} + \lambda = 0 \tag{42}$$

$$\sum_{n=1}^{N}\gamma(z_{nk}) + \lambda = 0 \tag{43}$$

Multiplicando por π_k a ambos lados:

$$\lambda\pi_k = -\sum_{n=1}^{N}\gamma(z_{nk}) \tag{44}$$

$$\lambda\pi_k = -N_k \tag{45}$$

Sumando ahora en k y teniendo en cuenta que la suma de los coeficientes de mezcla es igual a 1:

$$\lambda = -N \tag{46}$$

Por tanto:

$$\pi_k = \frac{-N_k}{\lambda} \tag{47}$$

$$\pi_k = \frac{N_k}{N} \tag{48}$$

Así pues, el coeficiente de mezcla para la componente *k* viene dado por la probabilidad media de que los puntos pertenezcan a dicha componente.

A partir de las ecuaciones obtenidas para $\boldsymbol{\mu}_k$, $\sum_k$ y $\boldsymbol{\pi}_k$, es posible implementar un sistema iterativo que permita encontrar una solución al problema de máxima verosimilitud. Basta con definir unos valores iniciales para las medias, las covarianzas y los coeficientes de mezcla y aplicar de forma iterativa los dos pasos del algoritmo EM que actualizan estos parámetros.

El primer paso es el E: esperanza (*expectation*). En este paso, el algoritmo trata de evaluar las probabilidades *a posteriori* $\gamma(z_k)$ en base a los valores $\boldsymbol{\mu}_k$, $\sum_k$ y $\boldsymbol{\pi}_k$ actuales.

El segundo paso es el M: maximización (*maximization*). Aquí se busca recalcular las medias, las covarianzas y los coeficientes de mezcla en base a las nuevas probabilidades $\gamma(z_k)$, haciendo uso de las ecuaciones definidas anteriormente. El objetivo es que, con cada conjunto de pasos EM, se aumente la función de log-verosimilitud hasta que la variación de un conjunto de pasos EM al siguiente caiga por debajo de un cierto umbral. Es entonces cuando el algoritmo habrá convergido hacia el máximo de la función.

Como se puede observar en la Figura III.1., este algoritmo requiere un número de pasos relativamente elevado. Es por ello que con frecuencia se emplea el algoritmo *k*-

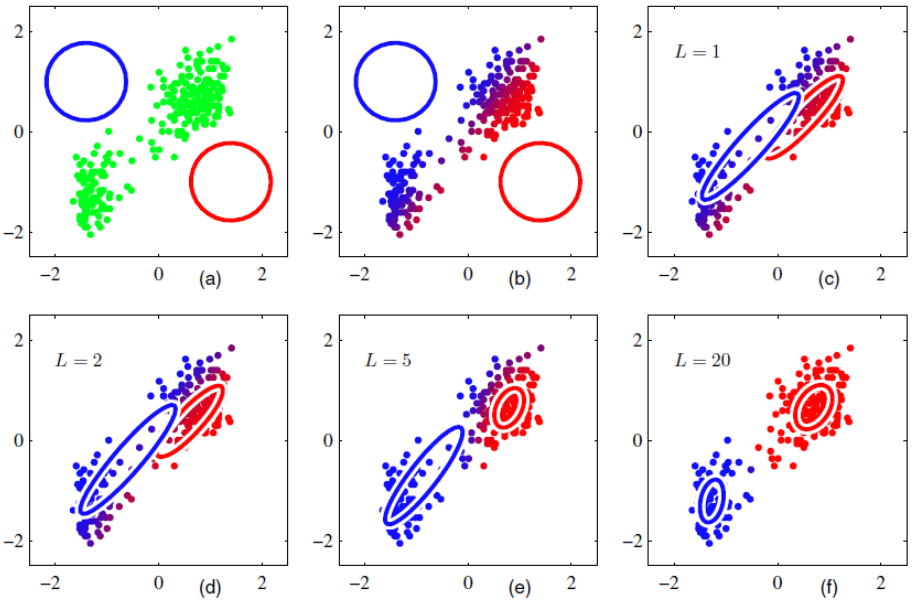

Fig. III.1: inicialización y evolución del algoritmo EM tras 1, 2, 5 y 20 pasos. Extraído de [17].

means para inicializar los parámetros $\boldsymbol{\mu}_k$, $\sum_k$ y $\boldsymbol{\pi}_k$ a unos valores óptimos antes de comenzar el algoritmo EM.

A pesar de las ventajas que presenta este procedimiento, no es capaz de paliar uno de los principales problemas del modelo de Mezcla de Gaussianas. Como se muestra en la Figura III.2, la presencia de singularidades puede hacer que alguna de las componentes del modelo "colapse" hacia un punto específico cuando se trata de maximizar la función de log-verosimilitud. Este comportamiento se ha de evitar con métodos heurísticos, reseteando su media a un valor aleatorio y su covarianza a un valor grande cuando esto ocurra, antes de continuar con la optimización.

Por último, se ha de tener en cuenta que en ocasiones la función de log-verosimilitud puede presentar diversos máximos locales y que en dichos casos el algoritmo EM no garantiza converger hacia el mayor de ellos.

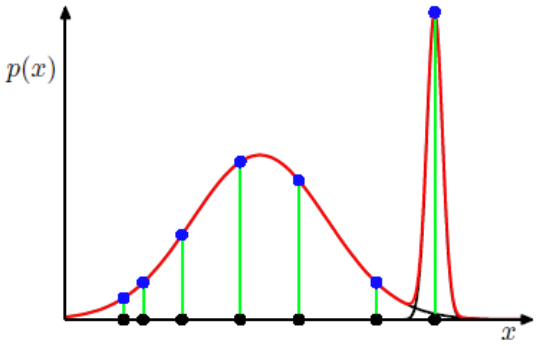

Fig. III.2: representación del problema de las singularidades en GMMs. Extraída de [17].

Referencias

[1] Balance de Seguridad Vial 2015 emitido por el Ministerio del Interior a 4 de enero de 2016.

[2] Piciarelli, C. (Noviembre de 2008). Trajectory-Based Anomalous Event Detection. *IEEE Transactions on Circuits and Systems for Video Technology, 18*(11), 1544-1554.

[3] Johnson, N., & Hogg, D. (2002). Representation and synthesis of behavior using Gaussian mixtures. *Image and Vision Computing, 20,* 889-894.

[4] Stauffer, C., & Grimson, W. (Agosto de 2000). Learning patterns of activity using real-time tracking. *IEEE Trans. Pattern Anal. Mach. Intell., 22(8)*, 852-872.

[5] Lou, J., Liu, Q., Tan, T., & Hu, W. Semantic interpretation of object activities in a surveillance system. *Proc. Int. Conf. Pattern Recognit*. Quebec, Canadá. III: 777-780.

[6] Robertson, N., & Reid, I. (2006). A General Method for Human Activity Recognition in Video. *Journal of Computer Vision and Image Understanding, 104*(2), 232-248.

[7] Lucas, B. D., & Kanade, T. (1981). An Iterative Image Registration Technique with an Application to Stereo Vision. *Proceedings DARPA Image Understanding Workshop*, 121-130.

[8] Morris, B. T., & Trivedi, M. M. (2008). A Survey of Vision-Based Trajectory Learning and Analysis for Surveillance. *IEEE Transactions on Circuits and Systems for Video Technology, 18*(8), 1114-1127.

[9] Morris, B. T., & Trivedi, M. M. (2008). Learning, Modeling, and Classification of Vehicle Track Patterns from Live Video. *IEEE Transactions on Intelligent Transportation Systems, 9*(3), 425-437.

[10] Utasi, Á., & Czúni, L. (2008). Visual Analysis of Urban Road Traffic. *15th International Conference on Systems, Signals and Image Processing*. 445-448.

[11] Monteiro, G., Ribeiro, M., Marcos, J., & Batista, J. Wrong way drivers detection based on optical flow. *IEEE International Conference on Image Processing, 5*. V: 141-144.

[12] Manuel Guizar-Sicairos, Samuel T. Thurman, and James R. Fienup, "Efficient subpixel image registration algorithms," Opt. Lett. 33, 156-158 (2008).

[13] N. J. B. McFarlance, and C. P. Schofield, Segmentation and tracking of piglets in images, Journal of Machine Vision and Applications , vol. 8, no. 3, pp. 187-193, 1995.

[14] Fischler, M.A., Bolles, R.C.: “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography.”

[15] Bagdanov, A., Del Bimbo, A., Dini, F., & Nunziati, W. (2007). Improving the robustness of particle filter-based visual trackers using online parameter adaptation. *IEEE Conference on Advanced Video and Signal Based Surveillance, AVSS*, 218-223.

[16] González, I. (2014). Diapositivas sobre Estimación densa de movimiento para la asignatura Ingeniería de Sistemas de Vídeo del Grado de Ingeniería de Sistemas Audiovisuales de la UC3M.

[17] Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Cambridge: Springer.

[18] Schulte, O. (s.f.). Latent Variable Models and Expectation Maximization slides from Machine Learning course at Simon Fraser University.

[19] Proyecto de Detección automática de incidentes en cámaras de la DGT basado en el tracking de vehículos realizado por el grupo GPM de la Universidad Carlos III de Madrid.

[20] Galvin, B., McCane, B., Novins, K., Mason, D., & Mills, S. (1998). Recovering Motion Fields: An Evaluation of Eight Optical Flow Algorithms. *British Machine Vision Conference*. 195-204.

[21] Calinon, S., & Billard, A. (2007). Incremental Learning of Gestures by Imitation in a Humanoid Robot. *2nd ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. 255-262.

[22] Calinon, S. (2009). Robot Programming by Demonstration: A Probabilistic Approach. EPFL/CRC Press.

[23] Nielsen, F. (2011). The Burbea-Rao and Bhattacharyya centroids. *IEEE Transactions on Information Theory, 57*(8). 5455-5466.

Bibliografía adicional

A continuación se listan otros documentos que, si bien no han sido referenciados explícitamente en la memoria, han sido consultados durante la realización del proyecto:

- Bagdanov, A., Del Bimbo, A., Dini, F., & Nunziati, W. (2007). Improving the robustness of particle filter–based visual trackers using online. *Advanced Video and Signal Based Surveillance, 2007. AVSS 2007. IEEE Conference on advanced video and signal based surveillance*, 218-223.
- Bloomenthal, J., & Rokne, J. (s.f.). *Unchained Geometry. The Future of Animation & Geometric Design.* Recuperado en 2016, de http://www.unchainedgeometry.com/jbloom/pdf/homog-coords.pdf

- De la Calle, F. (2012). *TFG Detección de eventos en secuencias con multitudes.* Leganés.
- Ley Orgánica 4/1997, de 4 de agosto, por la que se regula la utilización de videocámaras por las Fuerzas y Cuerpos de Seguridad en lugares públicos.
- Liu, C. (2013). Mixture Models and EM slides from Microsoft Research New England at Boston University.
- McCormick, C. (4 de Agosto de 2014). Recuperado el 5 de Junio de 2016, de http://mccormickml.com/2014/08/04/gaussian-mixture-models-tutorial-and-matlab-code/
- Rodríguez, Á. (2009). *PFC Estudio del filtro de partículas aplicado al seguimiento de objetos en secuencias de imágenes.* Leganés.
- Tomasi, C. (2004). Estimating Gaussian Mixture Densities with EM - A Tutorial. Durham, Carolina del Norte, EEUU.