Universidad Autónoma de Madrid
Escuela politécnica superior

Proyecto fin de carrera

"DETECCIÓN Y SEGUIMIENTO DE OBJETOS CON CÁMARAS EN MOVIMIENTO"

Ingeniería de Telecomunicación

Héctor López Paredes
SEPTIEMBRE 2011

"DETECCIÓN Y SEGUIMIENTO DE OBJETOS CON CÁMARAS EN MOVIMIENTO"

AUTOR: Héctor López Paredes
TUTOR: Manuel Sánchez Montañés

Grupo de Neurocomputación Biológica (GNB)
Dpto. de Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
SEPTIEMBRE 2011

Resumen

Resumen

Este trabajo ha sido realizado en el contexto de INTEGRA, proyecto financiado por el CDTI dentro del programa CENIT. El programa CENIT, cuyas siglas corresponden a "Consorcios Estratégicos Nacionales en Investigación Técnica", contempla la financiación de grandes proyectos integrados de investigación industrial de carácter estratégico, gran dimensión y largo alcance científico-técnico orientados a una investigación planificada en áreas tecnológicas de futuro y con potencial proyección internacional.

El objetivo principal será conseguir la detección de objetos en movimiento en sistemas que usen cámaras móviles y reducir las falsas alarmas que puedan ocasionar los cambios en el fondo introducidos por el movimiento de la cámara. La empresa ha contactado con nosotros requiriendo una solución que se integre con su solución de cámara estática limitando las posibles soluciones. Para conseguir el objetivo principal ajustándonos a estas restricciones hemos realizado un estudio del estado del arte identificando las dos estrategias principales que resuelven el problema. Una de las soluciones propone alinear las imágenes para eliminar el movimiento de la cámara y reducir el problema de detectar movimiento al de cámara estática. La otra solución propone detectar el movimiento directamente sin alinear las imágenes utilizando el flujo óptico de las imágenes.

La solución que mejor se adapta a nuestras restricciones es realizar el alineamiento de las imágenes para posteriormente detectar las regiones de movimiento mediante el modulo de segmentación para cámaras estáticas testeado y validado por la empresa. Las regiones de movimiento detectadas por el segmentador son analizadas por un algoritmo de tracking que trata de establecer una coherencia espacial y temporal de las distintas regiones detectadas para validarlas como objetos y reducir los errores que puedan aparecer por el ruido en el alineamiento.

El algoritmo se ha probado con un banco de pruebas elaborado por nosotros compuesto por un total de 109 vídeos. Los resultados en las clasificaciones realizadas por el algoritmo son idóneos cuando alineamos las imágenes por puntos y utilizamos el algoritmo de tracking para que decida cuántos objetos son detectados en la secuencia. Además de esto hemos realizado una prueba adicional que consiste en contar los objetos móviles que aparecen en los vídeos. Los resultados obtenidos para esta prueba son relativamente buenos cuando aparecen pocos objetos en el vídeo y muy malos cuando los objetos que aparecen en los vídeos superan la decena. La conclusión a las que hemos llegado es que el algoritmo cumple con su principal objetivo detectando si hay movimiento real correctamente en el 100 % de los vídeos. El algoritmo cuenta correctamente el número de objetos móviles cuando este número es pequeño, degradándose este conteo cuando el número de objetos móviles simultáneos aumenta.

Palabras Clave

Videovigilancia, Alineamiento de imágenes, Alineamiento global, Alineamiento por puntos, Segmentación, Tracking, Detección de movimiento, Movimiento propio de la cámara, Reducción de la tasa de falsos positivos.

Abstract

This work was carried out in the context of INTEGRA (Research in technologiefor migration management), a project funded by the CDTI, which is part of the CENIT program. The CENIT (National Strategic Consortiums for Technological Research) program seeks to fund large-scale, broad-based strategic industrial research projects in technological areas of the future with potential international projection.

The main goal will be to detect object in moving cameras environments and reduce false alarms that can be caused because of changes in background introduced by ego-motion camera. The company has contacted us requesting a solution that it can be integrated with his module for static cameras. To achieve the main objective limited by these restrictions, we have studied the state of the art finding two available solutions to solve the problem. The first solution suggests to register images to remove ego-motion camera and thus, to reduce the problem of moving detection to the static camera enviroments. The other solution suggests to directly detect moving without register images using the optical flow of the images.

The solution that best fits with our restrictions is one that realize the alignment of subsequent images in order to detect motion regions throw segmentation supplied by the company. The regions detected in the phase of segmentation are analyzed by a tracking algorithm that tries to establish a spatial and temporal coherence between the diferents regions to validate them as objects and to reduce errors that may appear from noise in the alignment.

The algorithm has been tested using a datatest composed of 109 videos recorded by us. The results in the ratings given by the algorithm is suitable when we register images using a point extractor together with tracking algorithm. In addition, we have performed additional test to count moving objetcts that appear in videos. The algorithm keeps track correctly the number of moving objects when this number is small, although the track is degraded when the number of simultaneously moving objects increases. The conclusion we have reached is that the algorithm fulfills it main objective by detecting the movement correctly in 100 % of the videos. However, the algorithm is rather vague to acurrately count the number of objects in the videos.

Key words

Videovigilance, Image Registration, Global Alignment, Feature Alignment, Segmentation, Tracking Algorithm, Moving Detectión, Ego-motion camera, False positive rate reduction.

Agradecimientos

Quiero dar las gracias en primer lugar a mi tutor **Manuel Sánchez-Montañés** por haberme ofrecido la oportunidad de realizar este proyecto y agradecerte toda la ayuda y el apoyo que he recibido durante todos estos meses. También quiero agradecer a **Luis Fernando Lago** por haberme ayudado también a realizar este proyecto y haber aportado todas esas opiniones que han sido de gran ayuda para finalizar este proyecto.

Quiero dar las gracias a todos los profesores que he tenido a lo largo de la carrera por haber contribuido con sus enseñanzas a mi formación como ingeniero de telecomunicaciones.

Y por último agradecer también a todos esos compañeros de clase (vosotros ya sabéis quienes sois) que tan buenos momentos he vivido con vosotros y de verdad que ha sido un verdadero placer haberos conocido y haber compartido mi vida universitaria con vosotros. Gracias.

Índice general

Índice de figuras

1

Introducción

1.1. Motivación

Actualmente los sistemas automáticos de videovigilancia se utilizan para múltiples aplicaciones entre las que destacan intentar detectar, reconocer o seguir objetos de manera autónoma a partir de secuencias de imágenes obtenidas por una cámara [4]. La mayoría de estos sistemas admiten como solución configuraciones con cámaras estáticas pero existen determinadas situaciones en las que se necesita una solución con cámara en movimiento, debido por ejemplo a posibles vibraciones en el emplazamiento de la cámara o porque queramos detectar objetos en áreas de gran tamaño utilizando cámaras en vehículos móviles.

Los problemas que pueden surgir cuando tratamos de detectar y seguir objetos con cámaras en movimiento son de diversa índole. Por ejemplo, cuando tenemos cámaras que se mueven, las técnicas que habitualmente se usan con cámaras estáticas para detectar movimiento ya no sirven para cámaras móviles porque de utilizar éstas, detectaríamos además de los objetos, los cambios en el fondo originados por el movimiento de la cámara, provocando errores significativos en la detección y el seguimiento de los objetos [5].

Figura 1.1: Ejemplo de detección de movimiento originado por cambios en el fondo.

Una posible solución pasa por implementar técnicas que sean capaces de deshacer el movimiento de la cámara reduciendo el problema al de cámara estática. Este es el motivo por el que la empresa ha contactado con nosotros, para que realicemos un módulo que sea capaz de deshacer el movimiento y así puedan utilizar la salida de ese módulo como entrada a su módulo de segmentación que ya está implementado y validado para cámaras estáticas.

El problema que ocasionan los cambios aparentes del fondo debido al movimiento de la cámara, así como la demanda de la empresa de implementar un módulo capaz de deshacer el movimiento compatible con su módulo para cámaras estáticas ha sido lo que ha motivado la intención de proponer una solución eficiente, que consiga detectar objetos en movimiento en sistemas con cámaras móviles.

1.2. Objetivos y enfoque

El principal objetivo de este PFC es el de realizar un sistema automático que a través de algoritmos de análisis de imagen sea capaz de detectar objetos en movimiento en vídeos tomados por cámaras móviles. Para construir este sistema se analizarán en primer lugar los algoritmos y técnicas que existen en la literatura para resolver el problema. Las técnicas serán analizadas críticamente para extraer sus ventajas e inconvenientes y escoger aquellas que sean más apropiadas y eficaces para conseguir los objetivos de este proyecto.

Este PFC utilizará estas técnicas como punto de partida para conformar el sistema global que resuelva el problema de la detección de movimiento. Uno de los objetivos de este PFC será enlazar dichas técnicas, mejorando su eficacia con algoritmos adicionales y adaptar estos al problema de la detección de movimiento para maximizar el número de clasificaciones correctas.

La parte final de este proyecto se centrará en evaluar la calidad del sistema utilizando un banco de pruebas de vídeos. El objetivo será extraer las conclusiones que nos permitan juzgar con precisión cómo de bueno es este trabajo para resolver los problemas que se proponen en él.

Para alcanzar los objetivos que se plantean en este proyecto, las tareas que se llevarán a cabo son las siguientes:

- **Estudio del estado del arte:** En primer lugar haremos un breve resumen sobre la historia de los sistemas de vídeovigilancia, desde sus comienzos hasta los sistemas actuales. Se realizará un estudio del estado del arte de las diferentes técnicas que corrigen el movimiento propio de la cámara y por otra parte de las técnicas que detectan los objetos móviles sin necesidad de corregir el movimiento de la cámara. Se estudiará el enfoque actual propuesto por diferentes autores para extraer las técnicas que resuelvan más eficazmente el problema de la detección con cámaras en movimiento.

- **Diseño e implementación de algoritmos y técnicas:** Una vez conocida la problemática, realizaremos el diseño que permita integrar las técnicas anteriormente elegidas y conformar un sistema que sea capaz de resolver los problemas a los que se enfrenta este PFC.

- **Elaboración de banco de vídeos y pruebas:** Después de haber diseñado el sistema se grabará un conjunto de vídeos que servirá para analizar o medir la eficacia del diseño elaborado en el paso anterior. En esta parte se describirán las diferentes pruebas que vamos a realizar para probar el conjunto de vídeos para extraer los resultados que determinen las conclusiones de este trabajo.

- **Análisis de resultados y conclusiones:** Por último se analizarán los resultados utilizando curvas ROC y tablas de clasificación que usarán el método de validación cruzada *leave one out* para estimar el error de generalización que tendrá el sistema en nuevos vídeos. El análisis de los resultados nos permitirá extraer las conclusiones finales de este proyecto y determinar si este trabajo es apropiado para su integración en sistemas comerciales.

1.3. Organización de la memoria

La memoria está compuesta por los siguientes capítulos:

- **Capítulo 1:** Introducción, motivación y objetivos del proyecto.
- **Capítulo 2:** Breve resumen sobre la historia de los sistemas de vídeovigilancia y tipos más utilizados. Estudio del estado del arte de las diferentes técnicas propuestas para resolver el problema de la detección de objetos móviles con cámaras en movimiento. Elección de las técnicas más adecuadas para este PFC analizando los puntos fuertes y débiles de éstas.
- **Capítulo 3:** Diseño e implementación de algoritmos que permitan resolver la detección de movimiento en entornos con cámara móvil. Análisis de posibles problemas de las diferentes técnicas utilizadas para la elaboración del diseño.
- **Capítulo 4:** Elaboración del banco de vídeos e implementación de pruebas que permitan cuantificar la eficacia de los sistemas implementados.
- **Capítulo 5:** Conclusiones y trabajo futuro.

2

Estado del arte

2.1. Visión global de los sistemas de videovigilancia

Se entiende por sistema de videovigilancia a toda aquella actividad que implique la colocación de una o varias cámaras de grabación, fijas o móviles, que tengan la finalidad de monitorizar el comportamiento y actividades de un espacio o personas [4]. Los sistemas de videovigilancia pueden estar compuestos, simplemente, por una o más cámaras de vigilancia conectadas a uno o más monitores o televisores que reproducen las imágenes capturadas por las cámaras. Los sistemas de videovigilancia pueden clasificarse por el tipo de sensor (infrarojo, audio y vídeo), por la multiplicidad del sensor (monocular o estéreo) y por el emplazamiento del sensor (centralizado o distribuido) [4].

Según Valera y Velastin [4] la forma más apropiada de clasificar los sistemas de videovigilancia es clasificarlos en tres generaciones que se distinguen por la tecnología empleada en ellos. La primera generación la forman los sistemas de circuito cerrado de televisión o CCTV. Se les conoce por el nombre de circuito cerrado porque todos los componentes que lo forman están enlazados entre sí. Este conjunto de sistemas están formados por un grupo de cámaras distribuidas que se sitúan en la zona de vigilancia y se conectan a un conjunto de monitores que normalmente se ubican en una sala central. Este tipo de sistemas son completamente analógicos y necesita de la supervisación de personas para hacer posible la detección de sucesos de interés. A pesar de sus limitaciones proporcionan una baja tasa de errores y son sistemas ampliamente utilizados debido en gran parte a la madurez de la tecnología que emplean. Sin embargo utilizan técnicas analógicas para el tratamiento, distribución y almacenamiento de las imágenes, y dependen demasiado de la actividad humana para detectar las anomalías que suceden en el entorno.

Los sistemas de segunda generación combinan la tecnología de los sistemas CCTV analógicos con los sistemas digitales de vigilancia-IP. El objetivo es intentar reducir la dependencia que existe de la actividad humana interpretando automáticamente los eventos y comportamientos que se producen en el entorno monitorizado. Estos sistemas permiten incrementar la eficiencia de los sistemas de seguridad, observando y analizando un mayor número de situaciones al mismo tiempo. También reducen la presencia del factor humano para detectar situaciones anómalas. Actualmente no existe una solución que permita realizar un razonamiento general sobre cualquier situación, solo existen soluciones parciales que interpretan soluciones muy concretas. Además existen situaciones anómalas imprevisibles que podrían no ser detectadas, y algunas de

las soluciones propuestas no son demasiado robustas, dando lugar a un número elevado de falsas alarmas.

Figura 2.1: VideoWall para un centro de videovigilancia.

Por último los sistemas de tercera generación se caracterizan por ser altamente distribuidos. Estos sistemas emplean dispositivos más modernos que mejoran o complementan los aparatos utilizados en anteriores generaciones. Al ser sistemas más distribuidos que los anteriores, la carga de procesamiento ya no se encuentra centralizada, reduciendo así la cantidad de datos que hay que procesar en cada dispositivo y ofreciendo tiempos de respuesta más cercanos al tiempo real. Además este último tipo de sistemas son más sólidos dado que si algún equipo se daña el sistema puede seguir trabajando con normalidad. Sin embargo existen problemas para distribuir eficazmente la información, aparte de que la comunicación entre dispositivos puede ser complicada debido a la heterogeneidad de sus componentes.

Actualmente existen muchas aplicaciones con cámaras estáticas y hay algoritmos desarrollados que trabajan eficazmente en entornos de cámaras fijas. Sin embargo hay otras aplicaciones donde las cámaras son móviles porque se necesitan cubrir zonas muy amplias o porque la cámara va montada en un dispositivo móvil. Esta situaciones son más complejas de abordar y requieren del uso de otros algoritmos para conseguir una tasa de error similar al obtenido con cámaras estáticas.

2.2. Aplicaciones de sistemas de videovigilancia

Existen diferentes tipos de sistemas de videovigilancia entre los que destacan:

- **Vídeo detección de intrusos**

  Un sistema de vídeo detección de intrusos [6], es una tecnología capaz de detectar accesos no autorizados por personas en áreas de seguridad o restringidas como aeropuertos, centros de detención de máxima seguridad o instalaciones nucleares . La función principal de este tipo de sistemas es la de detectar anomalías que puedan ser indicios de intrusión o alarmas. Para hacer posible la detección de intrusos, se recopila información de múltiples fuentes del sistema, analizando los datos obtenidos de diferentes maneras. Algunas de estas consisten en hacer un análisis estadístico sobre la información, buscando patrones que indiquen actividad anormal.

Figura 2.2: Detección de intrusos en exteriores.

- **Vídeo Tracking**

  Se entiende por Video Tracking [7] al proceso de localización de un objeto en movimiento (o varios) y su posterior seguimiento en el tiempo mediante el uso de cámaras de vídeo. Se utiliza en gran cantidad de aplicaciones, por ejemplo: interacción hombre-máquina; seguridad y vigilancia; compresión y comunicación de vídeo; realidad aumentada; control de trafico, imágenes médicas y edición de vídeo [7]. Los sistemas visuales de seguimiento tienen dos principales componentes: representación y localización del objetivo, y filtrado de la información asociada. El primera utiliza una gran variedad de algoritmos para identificar el objeto en movimiento, por ejemplo: “Blob tracking”, “Kernel-based tracking”, “Contour Tracking”, “Feature matching” [8]. El segunda necesita conocer información a priori de la escena o del objeto para poder hacer su seguimiento . Este algoritmo permite seguir objetos complejos incrementado notablemente la complejidad computacional. Los algoritmos más comunes son: filtro de Kalman [9] y "Particle filter"[10].

- **Detección de objetos abandonados**

  Esta tarea consiste en localizar automáticamente objetos que fueron abandonados en una escena [1]. Normalmente este tipo de objetos son bastante pequeños comparados con las personas y frecuentemente son ocultados por otras personas o vehículos que se mueven alrededor de la escena. Su funcionamiento se basa en utilizar algoritmos de segmentación que actúan en el primer plano de la secuencia. Los píxeles que no son parte del fondo y no se mueven se utilizan para encontrar regiones que contienen los posibles objetos abandonados. Sin embargo este tipo de sistemas son vulnerables a los problemas causados por fluctuaciones de iluminación, sombras y niveles de contraste de la escena [1].

Figura 2.3: Imágenes tomadas de [1].

- **Conteo de objetos**

  Los sistemas de conteo de objetos en entornos de vídeo son aquellos que son capaces de llevar la cuenta del número de objetos similares que aparecen en una escena en un determinado espacio de tiempo [4]. Estos sistemas se basan en las características del objeto

para su identificación. Existe un amplio campo de aplicaciones para este tipo de sistemas entre las que destacan el análisis de tráfico, el conteo de personas en zonas videovigiladas y procesos de fabricación.

2.3. Estado actual de sistemas que detectan movimiento desde cámaras en movimiento

La mayoría de los algoritmos de detección automática y de seguimiento están diseñados para configuraciones con cámara fija. Las técnicas basadas en sustracción de fondo o los algoritmos de diferenciación temporal no pueden usarse directamente para detectar movimiento con cámara activa. En los últimos años ha habido intentos para desarrollar sistemas de videovigilancia que sean capaces de operar con cámaras en movimiento dando como resultado una serie de algoritmos que, con restricciones, son capaces de resolver el problema.

A. **Métodos que diferencian imágenes consecutivas alineadas.**

Estos métodos consisten en alinear previamente las imágenes para después hacer una diferenciación entre ellas. Básicamente la idea consiste en eliminar el movimiento aparente de la cámara para que en la diferenciación se descubran realmente los objetos que se mueven. Para que este tipo de algoritmos tenga éxito la mayor parte del movimiento aparente entre las dos imágenes debe ser causado por el movimiento de la cámara. De lo contrario se estaría intentando alinear con respecto a los objetos móviles que queremos detectar causando importantes errores en la detección del verdadero movimiento. El alineamiento de las imágenes puede realizarse de tres formas distintas:

- **Alineamiento global:** Este tipo de alineamiento se caracteriza por tener en cuenta toda la información de la imagen para realizar el alineamiento. A cada píxel de la imagen se le aplica una transformación global que permite poner en correspondencia las imágenes que se traten de alinear. La forma de obtener dicha transformación varía según el método que se aplique:
  - *Template Matching:* Este método consiste en aplicar una transformación directa a una imagen objetivo (Target) para adecuarla a una imagen de "plantilla"(Template) [11]. Las imágenes template y objetivo suelen ser consecutivas para facilitar la transformación que se produce con el movimiento de la cámara. Para transformar la imagen se parte de unas restricciones iniciales acerca del tipo de transformación que se desea hacer. Existen muchas transformaciones diferentes entre las que destacan las transformaciones simples (tres parámetros: traslaciones y rotaciones), transformaciones afines (seis parámetros) y transformaciones proyectivas (nueve parámetros). Una vez transformada la imagen se necesita medir cómo de bueno ha sido el alineamiento. Para ello se utilizan unas medidas de similitud que nos darán información acerca de lo parecida que es la imagen alineada con respecto al Template. Existen muchas medidas de similitud entre las que destacan la correlación, el error cuadrático medio y la información mutua [11]. Finalmente necesitamos un algoritmo de búsqueda que encuentre el valor de los parámetros de la transformación que hace que la similitud entre las dos imágenes sea máxima. El problema de este método suele ser el tiempo que conlleva el realizar la búsqueda de los parámetros que optimizan la medida. Conforme queremos transformaciones más complejas, mayor número de parámetros se ven implicados en ella y por tanto más costosa es la búsqueda ya que se incrementan exponencialmente las combinaciones entre ellos. Sin embargo esta técnica

puede ser adecuada para imágenes con baja resolución, debido a la ventaja que se obtiene al utilizar toda la información para el alineamiento.

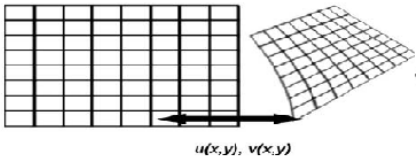
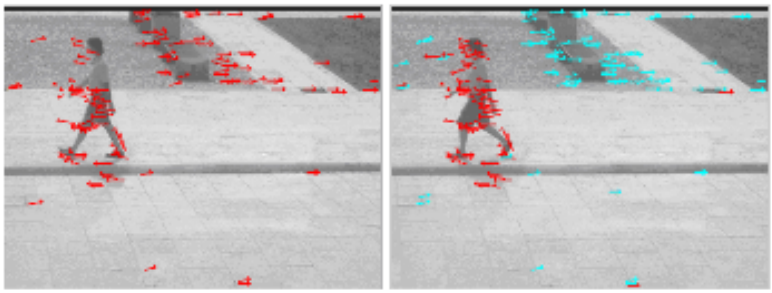


Figura 2.4: Ejemplo de transformación de la imagen en Template Matching [2].

- *Derivada Temporal:* El método más utilizado en este tipo de solución es el de obtener el flujo óptico de la imagen para que a través de técnicas de agrupamiento [3] detectemos la parte de la imagen correspondiente al fondo para su posterior alineamiento. El flujo óptico es el patrón de movimiento que existe entre una imagen y otra. Con él estimamos las velocidades instantáneas asumiendo que la variación de la intensidad entre las imágenes es pequeña. Una vez obtenido el flujo dominante (fondo) se utiliza la velocidad de dicho flujo para alinear el fondo y para a continuación detectar con la diferencia entre las imágenes los objetos de interés. Para obtener las velocidades se realiza el desarrollo de Taylor y se deja el resultado en función de las velocidades Vx y Vy. Obtenemos una ecuación en la que se deben buscar las velocidades Vx y Vy que hacen que esa ecuación sea igual a cero. A partir de este punto tendremos una ecuación (velocidades) por cada píxel, es decir, la variación de intensidad entre una imagen y otra en las dimensiones x e y pueden relacionarse con sus componentes de velocidad Vx y Vy, por tanto tenemos un sistema de ecuaciones que no necesariamente tienen que tener solución única. Este sistema se puede resolver de diferentes formas. Entre los métodos que destacan en la literatura los más utilizados son los métodos que utilizan aproximaciones por mínimos cuadrados, o el método de Lucas-Kanade [12], que para resolver el sistema impone unas restricciones considerando que el flujo situado en la vecindad del píxel que estamos considerando es constante.



(a) Motion vectors with the Lucas-Kanade algorithm (b) Clustered motion vectors

Figura 2.5: Ejemplo de extracción y clustering de vectores de movimiento [3].

- *Modelo Elástico:* Los anteriores métodos se basan en transformar directamente toda la imagen para así poder alinearla con su antecesora. El método del modelo elástico propone asumir que la imagen puede ser un material elástico, es decir, las deformaciones de una zona de la imagen no tienen por qué afectar a otras, por lo que la imagen que pretendemos alinear puede adaptarse completamente a la imagen previa deformando, según convenga, unas zonas u otras con independencia de las deformaciones en el resto de la imagen [2]. La imagen se va a flexionar y

estirar y su cantidad estará caracterizada por la energía del material elástico. El objetivo no será otro que el de minimizar esta energía. Esto conllleva una gran carga computacional volviendo de nuevo al problema del coste que supone utilizar un algoritmo que deformando la imagen converja a un óptimo proporcionado por algún tipo de medida.

- **Alineamiento por características:** A diferencia del alineamiento global, el alineamiento por características ya no utiliza toda la información de la imagen para realizar el alineamiento. En su lugar, se utilizan algoritmos que buscan puntos claves en las imágenes que queremos alinear. Los puntos deben ser lo más robustos posible frente a las variaciones entre imágenes para que la puesta en correspondencia sea más eficaz. Tal y como aparece en la literatura este método se puede dividir en tres fases:

  1) Extracción de puntos característicos.

     Existen diversos métodos en la literatura que extraen puntos de la imagen. Entre ellos destaca el algoritmo de Harris Stephens [13]. Este algoritmo se basa en el principio de que en una esquina la intensidad de una imagen cambiará en gran medida en múltiples direcciones. Para detectar ese cambio de intensidad se utiliza una ventana que se desplaza alrededor de la esquina para detectar la dirección de donde proviene el cambio. Otro algoritmo muy utilizado es el detector de SUSAN [14]. La idea consiste en suponer que en torno a la esquina existen dos áreas bien definidas: un área correspondiente a los píxeles que pertenecen a la esquina y otro área correspondiente a los píxeles que están fuera de la esquina. Se define un círculo que va situando su centro en los píxeles de la imagen y cuando el área que intersecciona con el círculo compuesta por los píxeles parecidos a ese centro es mínima se detecta esa zona como una esquina. Los algoritmos que más se suelen utilizar son los detectores SIFT/SURF [15]. En el algoritmo de SIFT los puntos característicos son los máximos y los mínimos de las diferencias entre funciones gausianas aplicadas a distintas escalas en la imagen suavizada. Los puntos candidatos de bajo contraste y los de borde son descartados para finalmente asignar un descriptor para cada punto con su orientación y escala.

  2) Correspondencia entre puntos (Matching).

     Una vez extraídos los puntos en las dos imágenes, para que puedan alinearse se necesita identificar el par de puntos/esquinas que son iguales en las dos imágenes. Como solución se proponen fundamentalmente dos tipos de métodos [16]:
     Correspondencia basada en flujo óptico: consiste en utilizar la idea del flujo óptico para realizar el "matching" de los puntos. En lugar de obtener una ecuación para cada píxel de la imagen, se obtiene una ecuación por cada punto característico de la imagen para después resolver el sistema utilizando algún método resolutivo como el de Lucas-Kanade o mínimos cuadrados.
     Correspondencia basada en similitud de descriptores: para que funcione este método cada punto característico obtenido debe tener asociado un descriptor. Este descriptor contiene información acerca de la escala y orientación. Seguidamente se calculan las distancias entre descriptores de un punto característico en la imagen fuente frente a todos los puntos característicos de la imagen destino. Aquella distancia mínima de entre todas servirá para identificar el par de puntos correspondidos entre la imagen fuente y la de destino. El proceso se repite para el resto de puntos característicos de la imagen fuente.

3) Transformación de la imagen

Establecida la correspondencia entre puntos, buscaremos una transformación que ponga en correspondencia unos puntos con otros. Según el tipo de transformación que queramos hacer (simple, afín y proyectiva) necesitaremos más o menos puntos para que la calidad de la transformación sea buena y así tener un buen alineamiento.

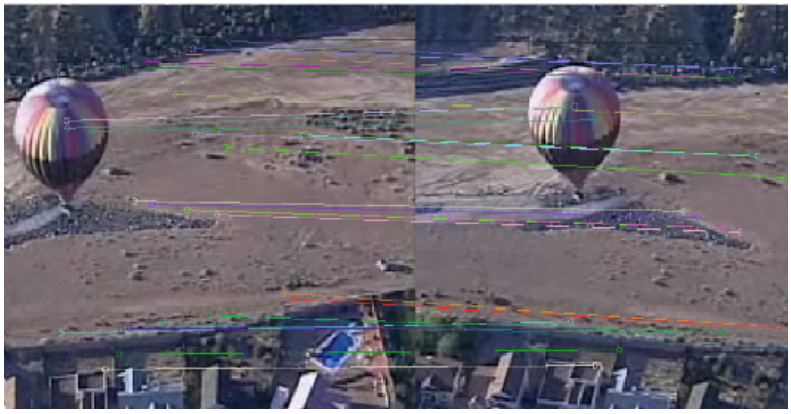


Figura 2.6: Figura de ejemplo: Extracción y matching de puntos.

- Alineamiento por parámetros intrínsecos de la cámara: La última técnica que utiliza alineamiento de imágenes como paso previo para detectar movimiento consiste en utilizar parámetros intrínsecos de la cámara para estimar el movimiento que se produce en ella. La relación que existe entre cada posición de los píxeles en las dos imágenes se puede obtener mediante la fórmula de Kanatani [17]. Para pequeños desplazamientos de la lente central desde el centro de rotación de la cámara, la fórmula de Kanatani relaciona con éxito los píxeles de las dos imágenes. Dada una inclinación inicial θ del sistema de la cámara, desplazamiento α e inclinación de rotación γ :

$$x_{t-1} = f\frac{x_t + \alpha sin\theta y_t + f\alpha cos(\theta)}{-\alpha cos\theta x_t + \gamma y_t + f}$$
$$y_{t-1} = f\frac{-\alpha sin\theta x_t + y_t - f\gamma}{-\alpha cos\theta x_t + \gamma y_t + f}$$

donde f es la distancia focal. Sabiendo f, θ,γ, y α para cada posición (x_t, y_t) en la imagen actual, podemos calcular la posición (x_{t-1}, y_{t-1}) del correspondiente píxel en la imagen previa.

B. **Métodos que detectan movimiento directamente en la imagen.** Existe una segunda clase de método que no necesita alinear las imágenes para detectar los objetos en movimiento en cámaras móviles. Esta técnica intenta encontrar entre dos imágenes movidas los píxeles que no encajan con el patrón del movimiento general de la imagen. A esta técnica se la conoce como **Detección basada en distorsión de flujo** [18] y sirve para distinguir entre objetos estáticos y en movimiento. En las imágenes capturadas con la cámara en movimiento, ambos tipos de objetos generan flujo óptico. La idea es aprovechar el flujo para obtener el movimiento de los objetos directamente. Este método asume que los píxeles que se corresponden con el flujo global pertenecen al fondo de la imagen. Este flujo global se extrae ajustando un sistema de ecuaciones que relaciona los flujos de las dos imágenes. La parte de flujo no ajustado se considera el movimiento real de la escena de modo que la detección del movimiento es directa sin ningún tipo de alineación previa.

Según las técnicas analizadas hemos visto que existen múltiples métodos para detectar movimiento con cámaras activas.

La primera estrategia que hemos encontrado en la literatura consiste en deshacer el movimiento de la cámara para eliminar el movimiento aparente (Métodos A). Si la cámara nos ofrece parámetros acerca de la rotación, desplazamiento e inclinación, nos será mas fácil y rápido alinear las imágenes utilizando dichos parámetros. Sin embargo en la mayoría de las situaciones esto no ocurre, por lo que deberemos analizar la imagen en busca de parámetros externos a la cámara. Las técnicas de alineamiento global utilizan todos los píxeles de la imagen para obtener una transformación que alinee la imagen con su antecesora. Por otro lado las técnicas de alineamiento por puntos utilizan zonas aisladas de la imagen con determinadas propiedades que facilitan la correspondencia entre ellas y permiten aplicar posteriormente una transformación que alinee las imágenes. Según la literatura las dos técnicas de alineamiento son igualmente validas para alinear las imágenes y corregir de este modo el movimiento de la propia cámara.

La tercera estrategia que hemos encontrado en la literatura abandona la idea de alinear las imágenes y se centra en la detección directa del objeto en movimiento (Métodos B). Aprovecha la distorsión del flujo óptico para identificar el movimiento real que ocurre en la secuencia grabada. Aunque no necesita alineamiento, el algoritmo es muy costoso debido a la cantidad de ecuaciones que deben resolverse simultáneamente. Además esta técnica introduce ruido en los píxeles de la imagen, ruido que deberá ser corregido por algún método adicional.

En este proyecto utilizaremos dos técnicas que están dentro del grupo de métodos A: el alineamiento global y el alineamiento por puntos con correspondencia basada en similitud de descriptores. La razón por la que hemos escogido estos dos métodos es porque las técnicas que utilizan flujo óptico requieren de algoritmos muy costosos para resolver los sistemas de ecuaciones. Además el flujo óptico introduce ruido que necesariamente tendría que ser corregido en fases posteriores. El método basado en alinear las imágenes utilizando parámetros intrínsecos de la cámara fue descartado porque uno de los objetivos de este trabajo es que el sistema sea compatible para todo tipo de cámaras. Las técnicas elegidas plantean resolver el problema del alineamiento desde dos puntos de vista diferentes ofreciendo ventajas e inconvenientes que exploraremos para resolver el problema de la detección de movimiento en entornos de cámaras móviles.

3
Diseño

A lo largo de este capítulo se van a describir con detalle las técnicas implementadas para la realización con éxito de la detección de movimiento con cámara activa. Se van a explorar fundamentalmente dos técnicas que alinean las imágenes consecutivas (sección 2.1.2, punto A). Este tipo de técnicas son computacionalmente más sencillas y en principio menos sensibles al ruido en contraposición a las técnicas basadas en flujo óptico [19] . Las técnicas que vamos a poner en práctica son el alineamiento global utilizando Template Matching y el alineamiento por características con correspondencia basada en similitud de descriptores.

En las siguientes subsecciones se abordará paso a paso cómo realizar un alineamiento eficaz entre imágenes consecutivas y cómo distinguir entre el movimiento aparente y el movimiento real de los objetos que aparecen en la escena.

Posteriormente se realizará una comparativa entre los distintos métodos implementados utilizando una base de vídeos creada por nosotros, haciendo un estudio sobre que métodos son más apropiados para según qué situaciones y cuáles de ellos son computacionalmente más eficaces para ser candidatos a algoritmos que funcionen en tiempo real.

3.1. Corrección del movimiento de la cámara usando alineación

La idea principal que vamos a utilizar para detectar los objetos móviles es corregir el movimiento de la cámara alineando las imágenes para después aplicar métodos de substracción y segmentación que se utilizan en cámaras fijas para extraer los objetos detectados. En el anterior capítulo vimos que podíamos detectar movimiento directamente utilizando flujo óptico, pero debido al coste que implica la resolución de las ecuaciones de flujo y el ruido que introduce, se optó por la idea del alineamiento previo y la detección a posteriori.

Esta idea es más adecuada para este PFC porque tiene una implementación más sencilla y existen varios tipos de técnicas que ofrecen más versatilidad para la detección de objetos en entornos de cámara activa. Sin embargo la calidad de la detección va a depender en gran parte de la correcta alineación de las imágenes. Un mal alineamiento producirá en la detección objetos no deseados, normalmente esquinas o bordes que pertenecen al fondo de la imagen.

Para realizar correctamente el alineamiento optaremos por dos enfoques distintos:

- Alinear utilizando toda la información de los píxeles que proporciona la imagen (Alineamiento Global).
- Alinear utilizando sólo unos puntos característicos de la imagen (Alineamiento por Características basado en similitud de descriptores).

3.1.1. Alineamiento Global (*Template Matching*)

El Template Matching consiste en buscar de manera iterativa la transformación que hay que aplicar a una de las imágenes (imagen "flotante") para que esté lo más alineada con la otra imagen ("template"). La calidad del alineamiento se cuantifica mediante una medida que compara los píxeles de la imagen template con los de la imagen flotante transformada. El proceso completo se detalla en la siguiente imagen:

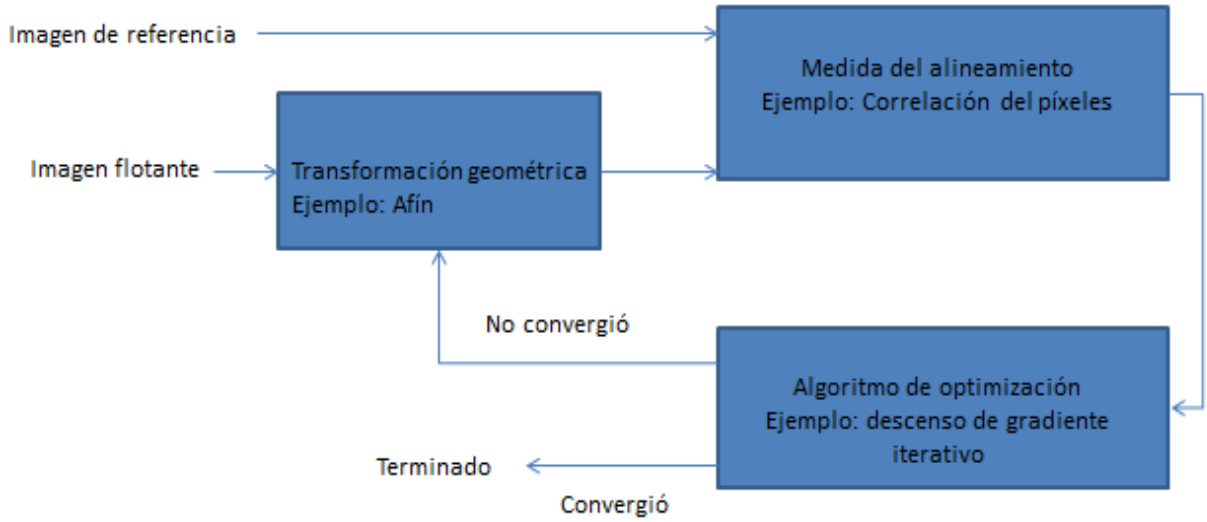


Figura 3.1: Esquema *Template Matching*

En la Figura 3.1 podemos observar que el algoritmo parte de dos imágenes: La primera imagen, es la imagen de referencia o "template", y la segunda es la imagen "flotante". Esta última imagen va a sufrir una serie de transformaciones hasta que el alineamiento con la imagen de referencia se complete. Por ejemplo en el caso de un descenso por gradiente se siguen los siguientes pasos:

1. Se compara la imagen transformada (Floating Image) con la imagen de referencia, utilizando algún tipo de medida que proporcione un valor acerca de la calidad que tiene el alineamiento.
2. Tras evaluar la calidad del alineamiento, el algoritmo de optimización reajusta los parámetros de transformación para que las dos imágenes sean más parecidas. Se pasa al paso 1 hasta que el proceso converja.

La transformación de la imagen, restringida a transformaciones lineales, se consigue multiplicando la imagen flotante por una matriz de transformación que está definida por parámetros libres que determinan el tipo de transformación. El tipo de transformación va a caracterizar la capacidad, la complejidad y el tiempo de alineamiento. Las transformaciones que utilicen un

mayor número de parámetros podrán deshacer movimientos más complejos, como el zoom o efectos no deseados de la lente de la cámara. Sin embargo el tiempo en la búsqueda del alineamiento será mayor porque el algoritmo de optimización tiene más parámetros que ajustar. En ocasiones, la búsqueda puede no converger, debido a que el algoritmo no es capaz de superar mínimos locales proporcionados por la medida del alineamiento. Por ello el tipo de transformación es clave para marcar el tipo de movimientos que estamos dispuestos a tolerar, y el tiempo que vamos a necesitar para corregir con éxito el movimiento inducido por la cámara.

3.1.1.1. Tipos de transformaciones

Existen muchos tipos de transformaciones geométricas en la literatura clasificándose en transformaciones lineales y no lineales. Las transformaciones lineales son más simples que las no lineales y suelen utilizarse más cuando tratamos de alinear dos imágenes [11]. Las transformaciones más usuales son las traslaciones, rotaciones, simetrías y las homotecias.

Las transformaciones geométricas se pueden dividir en tres grupos:

- Transformación euclídea.
- Transformación afín.
- Transformación proyectiva

A continuación se describen en detalle estas transformaciones:

Transformaciones euclídeas. Las transformaciones euclídeas permiten que los objetos se trasladen, roten o se reflejen y conservan la longitud y el ángulo de la forma del objeto, es decir, las líneas se trasforman en líneas, los círculos en círculos y los ángulos se conservan. En nuestro estudio con transformaciones euclídeas sólo se han probado traslaciones y rotaciones puesto que los reflejos en secuencias de vídeo no se suelen dar. Sin embargo cuando la cámara que se mueve está suficientemente alta, enfoca al suelo y el ángulo que forma con respecto al suelo es fijo (por ejemplo videovigilancia en UAVS Unmmaned Aerial Vehicle) las únicas transformaciones que se dan son traslaciones y rotaciones.

Cuando intentamos trasladar a un nuevo sitio un punto en el plano-xy añadiendo un vector de traslación (t_1, t_2), podemos expresar el nuevo punto (x', y') como $x' = x+t_1$ y $y' = y+t_2$. Por conveniencia, añadimos una tercera componente, de esa forma podemos expresar las traslaciones en dos dimensiones como matrices. Por lo tanto, el punto en 2D (x, y) puede expresarse de la siguiente forma:

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

La relación que existe entre (x, y) y (x', y') puede expresarse mediante la siguiente ecuación:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & t_1 \\ 0 & 1 & t_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Paralelamente, si el punto (x, y) es rotado un ángulo θ desde su coordenada de origen para convertirse en un nuevo punto (x', y'), la relación entre ambos puntos es de la siguiente forma:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

La matriz que relaciona ambos puntos se conoce como matriz de rotación R. Poniendo en una misma ecuación la traslación y rotación la ecuación anterior puede expresarse como:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} cos(\theta) & -sin(\theta) & t_1 \\ sin(\theta) & cos(\theta) & t_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

La matriz que relaciona el nuevo punto (x', y') con el punto (x, y) se denomina matriz de transformación y se puede observar que tiene tres parámetros libres que la definen: traslación en horizontal t_1, traslación en vertical t_2 y rotación de ángulo θ. Estos tres parámetros serán modificados por el algoritmo de búsqueda hasta encontrar la traslación (t_1, t_2) y la rotación θ que pone en correspondencia la imagen flotante con la imagen de referencia.

Normalmente este tipo de transformaciones se observan en vídeos grabados por cámaras situadas en aviones que vuelan a una altura constante y a gran altitud con la cámara fija hacia abajo. Los objetos observados desde esta perspectiva carecen de tercera dimensión y parecen moverse en un plano de dos dimensiones, de modo que las únicas transformaciones necesarias para describir estos movimientos de cámara son la traslación y la rotación.

Transformaciones afines. Las trasformaciones afines son una generalización de las euclídeas y son las transformaciones más comúnmente utilizadas para registrar imágenes [11]. Bajo una transformación afín, las líneas se transforman en líneas pero los círculos se convierten en elipses. Por tanto la longitud y el ángulo dejan de conservarse. Las principales diferencias de las transformaciones afines con respecto a las euclídeas son el escalado y la cizalla (deformación).

La ecuación matricial de la transformación afín es muy parecida al de la euclídea. Los puntos (x, y) y (x', y') se relacionan mediante la expresión:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & t_1 \\ a_{21} & a_{22} & t_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

siendo

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$$

La forma de obtener A cambia ligeramente con respecto a las transformaciones euclídeas. La matriz A se obtiene multiplicando dos matrices de rotación con ángulos diferentes y una matriz de escalado D, es decir:

$$A = R(\theta)R(-\phi)DR(\phi)$$

donde $R(\theta)$ y D son respectivamente:

$$R(\theta) = \begin{pmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{pmatrix}, D = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}$$

Los parámetros λ_1 y λ_2 corresponden al escalado, el ángulo θ a la rotación y ϕ al ángulo de deformación:

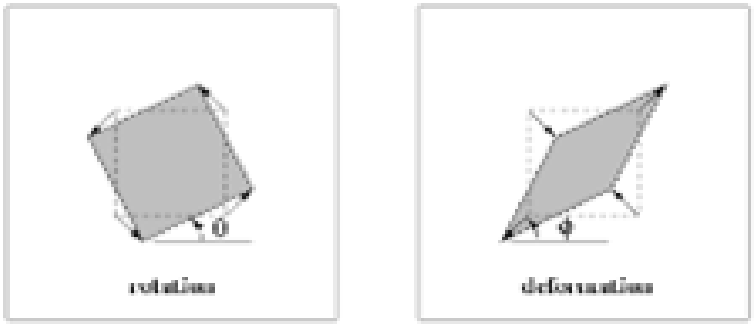


Figura 3.2: Rotación y deformación o cizalla

Para este tipo de transformaciones tenemos un total de seis parámetros libres que hay que ajustar: traslaciones (t_1, t_2), rotación θ, escalado (λ_1, λ_2) y deformación ϕ. Las transformaciones afines suelen utilizarse cuando la cámara en movimiento se encuentra más próxima al suelo. El algoritmo de búsqueda que trate de corresponder dos imágenes en una secuencia de vídeo en la que pueda haber transformaciones afines tendrá que ajustar estos seis parámetros para cada par de imágenes.

Transformaciones proyectivas. Las transformaciones proyectivas son una generalización de las transformaciones afines, y están compuestas por una transformación lineal general y de una traslación. La notación compacta de una transformación proyectiva es:

$$x' = Hx = \begin{pmatrix} A & t \\ \mathbf{v^T} & v \end{pmatrix} x$$

donde $v = (v_1, v_2)^T$. La principal diferencia entre la transformación proyectiva y la afín es el vector $v = (v_1, v_2)^T$ que define la tercer fila de la matriz de transformación H. En la transformación afín este vector es fijo e igual a $(0, 0)$. En cambio en la proyectiva puede ser cualquiera del espacio siendo este vector el responsable de los efectos de la perspectiva. La transformación proyectiva puede descomponerse en el siguiente conjunto de matrices:

$$H = H_S H_A H_P = \begin{pmatrix} s\mathbf{R} & \mathbf{t} \\ 0^T & 1 \end{pmatrix} \begin{pmatrix} \mathbf{K} & \mathbf{t} \\ 0^T & 1 \end{pmatrix} \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{v^T} & v \end{pmatrix} = \begin{pmatrix} A & t \\ \mathbf{v^T} & v \end{pmatrix}$$

donde A es una matriz no singular dada por $\mathbf{A} = \mathbf{sRK} + \mathbf{tv^T}$ y $\mathbf{K}$ es una matriz triangular superior con determinante igual a uno. La matriz A tiene nueve elementos pero sólo ocho de ellos son independientes. Esto significa que el algoritmo de búsqueda tiene que ajustar 8 parámetros

para encontrar la transformación de la imagen que hace que se corresponda con la imagen de referencia. Las transformaciones proyectivas permiten transformaciones más complejas y suelen aparecer en secuencias de vídeo donde la cámara se mueve cerca del suelo. La búsqueda en este rango de transformaciones va a ser más costosa en tiempo debido a la enorme cantidad de combinaciones que se pueden dar en 8 parámetros independientes. Sin embargo este tipo de transformaciones son las que más versatilidad ofrecen dentro de las estudiadas.

3.1.1.2. Medidas de similitud

Una vez transformada la imagen flotante, el algoritmo necesita saber cómo de buena es la transformación. Para averiguarlo, el algoritmo necesita que se calcule algún tipo de medida de similitud entre la imagen flotante transformada y la imagen de referencia. Esta medida normalmente suele ser una distancia que nos va a decir cómo de parecidas son las imágenes. Las medidas que se han tenido en cuenta en este PFC son:

- Error absoluto.
- Correlación.
- Correlación + Error absoluto.

Estas medidas son unos indicadores que determinan el grado de similitud que existe entre dos imágenes. El algoritmo de minimización utilizará la medida para dar nuevos valores a los parámetros e intentar que la búsqueda converja a un óptimo absoluto. La convergencia estará determinada por la medida y el tipo de imagen que estemos intentando alinear existiendo la posibilidad de alineamientos erróneos por convergencias a mínimos locales.

Error absoluto La forma más sencilla de establecer una medida para determinar el alineamiento entre dos imágenes, es restar la imagen transformada de la imagen de referencia. Al restar las imágenes obtenemos con el valor de la resta, información precisa de si los píxeles son iguales o no. Dadas dos imágenes f y g, la medida local de alineamiento $m(p)$ donde $p = (x, y)$ determina la similitud entre el conjunto de píxeles $f(x, y)$ y los transformados $Tg(x, y)$:

$$E_A(T) = \sum_i |Tg(p_i) - f(p_i)|$$

El objetivo será buscar la transformación T que minimice la suma de cada píxel restado de entre las imágenes de referencia y transformada. Al contrario de la correlación el error absoluto no contempla dependencias de escalado y sesgo entre los valores de los píxeles de las dos imágenes. Entre una imagen y otra podría haber un simple cambio de iluminación y el E_A daría un error muy alto cuando las imágenes están completamente alineadas. Por eso decidimos probar como medida de similitud la correlación, ya que es invariante frente a traslaciones en la iluminación y cambios de escalas.

Correlación La correlación es un método rápido para estimar el grado de dependencia que existe entre dos variables. La correlación maximiza el producto (correlación cruzada) entre dos imágenes alineadas,

$$E_C = \sum_i f(p_i)Tg(p_i)$$

A primera vista esta fórmula tiene demasiado en cuenta el valor absoluto del producto de los píxeles de ambas imágenes. Nuestro objetivo es buscar la transformación T que maximice el producto de las dos imágenes. Si una determinada zona de la imagen $g(p)$ está demasiado iluminada, esa zona aumentará mucho el valor de la medida mintiendo acerca del alineamiento global de las imágenes. Por esta razón preferimos utilizar la correlación normalizada. La correlación normalizada se define como:

$$E_{CN} = \frac{\sum_i [f(p_i) - \overline{f(p_i)}][Tg(p_i) - \overline{Tg(p_i)}]}{\sqrt{\sum_i [f(p_i) - \overline{f(p_i)}]^2} \sqrt{\sum_i [Tg(p_i) - \overline{Tg(p_i)}]^2}}$$

donde

$$\overline{f(p_i)} = \frac{1}{N} \sum_i f(p_i)$$

$$\overline{Tg(p_i)} = \frac{1}{N} \sum_i Tg(p_i)$$

son el valor medio de las imágenes y N es el número total de píxeles de la imagen. La correlación normalizada está acotada entre los valores $[-1, 1]$. Un valor de 1 significará que las imágenes están muy correlacionadas y el valor de 0 corresponde a dos imágenes totalmente no correlacionadas. Como el algoritmo intenta minimizar el valor obtenido por la función de medida deberemos reescribir la fórmula anterior a,

$$VB(T) = 1 - E_{CN}(T)$$

de este modo el algoritmo está maximizando el valor de la correlación normalizada.

Correlación + Error Absoluto La medida que finalmente utilizamos tras realizar pruebas no exitosas con las medidas anteriores fue realizar una combinación de ambas ya que los objetos que se mueven interfieren en la alineación. Esta solución aprovecha las ventajas de la correlación y el error absoluto haciendo que el algoritmo busque píxeles parecidos entre las dos imágenes sin que se introduzcan demasiados errores por cambios de iluminación o ruido entre las imágenes. El objetivo de la reordenación es quitar el porcentaje de píxeles que fueron más distintos entre sí ya que no nos interesa una alineación completa asumiendo que son los píxeles de los objetos que se mueven. La finalidad de esta técnica es alinear el fondo de las imágenes para que una vez alineado se distingan los objetos que se mueven. Por tanto es importante que la medida no se vea interferida por aquella resta de píxeles que dieron un mayor valor, porque tal vez esos píxeles pertenecen a los objetos que realmente se mueven. El proceso sigue los siguientes pasos:

1. Resta de píxeles. Se guardan en un vector el valor absoluto de la resta píxel a píxel de la imagen transformada con la imagen de referencia, eso es:

   $$V(p_i) = |F(p_i) - T(p_i)|$$

   donde $F(p_i)$ es la imagen transformada y $T(p_i)$ es la imagen de referencia o template.

2. Ordenación. Se ordenan los valores del vector de menor a mayor para quitar los píxeles con mayor valor.

3. Correlación. Una vez realizada la ordenación obtenemos los indices de la imagen correspondientes a los píxeles reordenados y utilizamos estos índices para realizar la correlación normalizada entre las dos imágenes. Finalmente se utiliza la fórmula $VB(T)$ para pasar el valor de la medida al algoritmo.

3.1.1.3. Algoritmo de optimización

Una vez transformadas las imágenes y obtenida una medida acerca del grado de alineamiento que hay entre ellas, se utiliza un algoritmo para buscar unos nuevos valores de transformación que den como resultado un alineamiento mejor. El alineamiento será mejor cuando la medida de similitud sea mejor por tanto necesitamos un algoritmo que optimice la medida del alineamiento. El algoritmo que hemos escogido se basa en el método de Nelder-Mead [20]. El método busca de modo aproximado una solución óptima local a un problema con N variables cuando la función a minimizar varía suavemente. Para realizar la búsqueda parte del concepto de un simplex que es un politopo de $N + 1$ vértices donde N son las dimensiones o parámetros. Así que para un problema de dos parámetros tendríamos tres puntos, un triángulo. El algoritmo sigue los siguientes pasos:

1. Calcula las medidas para los $N + 1$ puntos iniciales.
2. Refleja el punto que tiene el valor de la medida más alto utilizando como espejo el centroide del simplex.
3. Si produce un punto con menor medida el simplex se expande hacia la dirección del reflejo.
4. Si resulta que tras el reflejo el punto no es bueno el simplex reflejado se comprime haciendo el nuevo reflejo más cercano al del punto de partida.

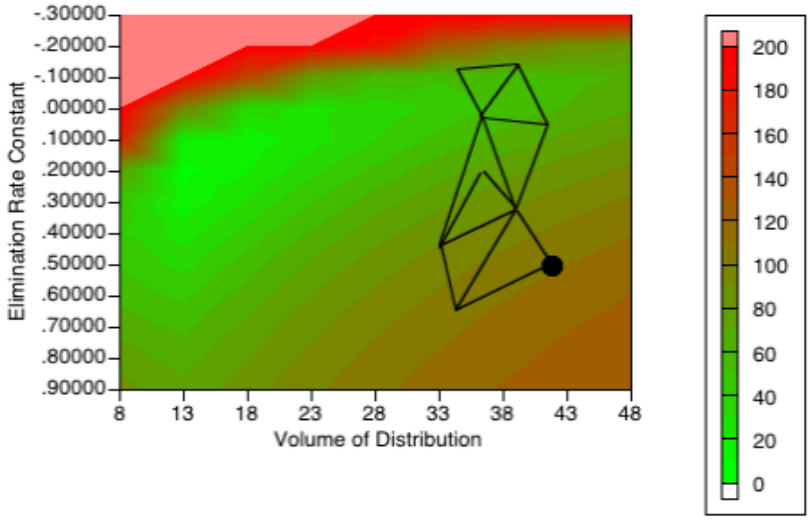


Figura 3.3: Algoritmo Nelder-Mead (método Simplex). Imagen obtenida de la página `http://www.boomer.org/c/p3/c11/c1106.html`.

Estos pasos se repiten una y otra vez hasta que la convergencia se alcanza. Este método es relativamente robusto y numéricamente no muy complicado pero puede ser ineficiente por problemas asociados a la inicialización de los parámetros y a que la función de medida no sea suave pudiendo estancarse el algoritmo en mínimos locales.

Parámetros del algoritmo A lo largo de las anteriores secciones se ha destacado el impacto en eficiencia que tiene el hacer una búsqueda con transformaciones complejas por culpa de un elevado número de parámetros de transformación. Sin embargo existen entradas adicionales en el algoritmo que pueden ayudar a que la búsqueda del mínimo no sea tan costosa. Un valor de entrada mal elegido puede dar como resultado que el algoritmo no converja. Algunas de estos parámetros son:

- Función objetivo. Una función objetivo suave y que no varíe bruscamente será clave para que la búsqueda sea rápida.
- Intervalo de búsqueda. El intervalo de búsqueda es el espacio que deja entre los distintos saltos que hace el algoritmo para realizar la búsqueda. Intervalos de búsqueda excesivamente grandes serán menos finos en la búsqueda pudiendo hacer que se pase por alto mínimo buscado. Sin embargo, un intervalo de búsqueda demasiado pequeño hará que el algoritmo sea demasiado vulnerable a mínimos locales porque la búsqueda se explorara en entornos mas cercanos a los puntos que dieron un valor aceptable.
- Valores iniciales de los parámetros de transformación. Inicializar los parámetros libres de los valores que se ajustan para encontrar el mínimo objetivo es de vital importancia. Una correcta inicialización situará al algoritmo en el entorno adecuado para la búsqueda del mínimo absoluto evitándonos la convergencia a posibles mínimos locales y realizando la búsqueda en un menor tiempo. Por defecto, la inicialización de los valores de los parámetros de transformación es cero. Tras realizar muchas pruebas se observó que esta inicialización suponía demasiado coste en tiempo de búsqueda por lo que se optó por una inicialización alternativa. Esta inicialización que va a suponer un menor tiempo de convergencia, consiste en realizar una búsqueda previa del mínimo con resoluciones de imágenes inferiores. Estas búsquedas se van a realizar en muchísimo menos tiempo al estar trabajando con imágenes de mucho menos tamaño. Cuando finalice la búsqueda se habrán elegido unos valores de los parámetros que aplicando correcciones de escalado valdrán como valores iniciales de las imágenes de mayor tamaño. Aunque se llama al algoritmo un número mayor de veces la eficiencia es mucho mayor utilizando multiresolución porque la búsqueda converge mucho más rápido que cuando buscamos el mínimo sólo en las imágenes de tamaño inicial.
- Condición de parada. La condición de parada es el criterio que utiliza el algoritmo para determinar si el mínimo encontrado es el adecuado. El algoritmo resta las medidas actual y anterior y si la diferencia es menor que la condición de parada elegida entonces el algoritmo detiene la búsqueda. Existe un compromiso entre condiciones de parada demasiado flexibles y eficiencia del algoritmo. A mayor flexibilidad menos restrictivos somos con el mínimo y menos tiempo se necesita para su búsqueda y al contrario. Cuanto menor sea la flexibilidad de la condición de parada mejor será el mínimo encontrado pero se invertirá mucho más tiempo en su búsqueda.

El Template-Matching aprovecha la ventaja de trabajar con toda la información de la imagen para encontrar el alineamiento correcto. Para ello es necesario elegir primero el tipo de transformación que vamos a necesitar para alinear las imágenes. Seguidamente deberemos elegir una medida que se adapte lo mejor posible a las condiciones de alineamiento que queremos y finalmente deberemos dar valores a los parámetros del algoritmo de búsqueda. Todo este proceso puede ser demasiado lento porque las búsquedas pueden no converger, porque no se haya elegido el tipo de transformación adecuado o porque la inicialización de los parámetros esté demasiado alejada del mínimo buscado. Sin embargo para transformaciones simples (traslaciones y rotaciones) el algoritmo suele converger rápidamente ya que no tenemos muchos parámetros en la búsqueda y se pueden efectuar muchas transformaciones en poco tiempo hasta encontrar el valor

de los parámetros que alinean las imágenes. Si queremos alinear imágenes con transformaciones mas complejas en un tiempo menor debemos utilizar una técnica alternativa que no pierda tanto tiempo analizando toda la información de la imagen. Deberemos entonces utilizar técnicas que extraigan sólo una pequeña información relevante de la imagen.

3.1.2. Técnicas de alineamiento usando puntos característicos

El alineamiento por puntos característicos consiste en relacionar los puntos característicos extraídos de dos imágenes y utilizar esta información de relación (matching) para alinear una imagen con otra por medio de una transformación geométrica. El esquema completo del proceso se observa en la siguiente imagen:

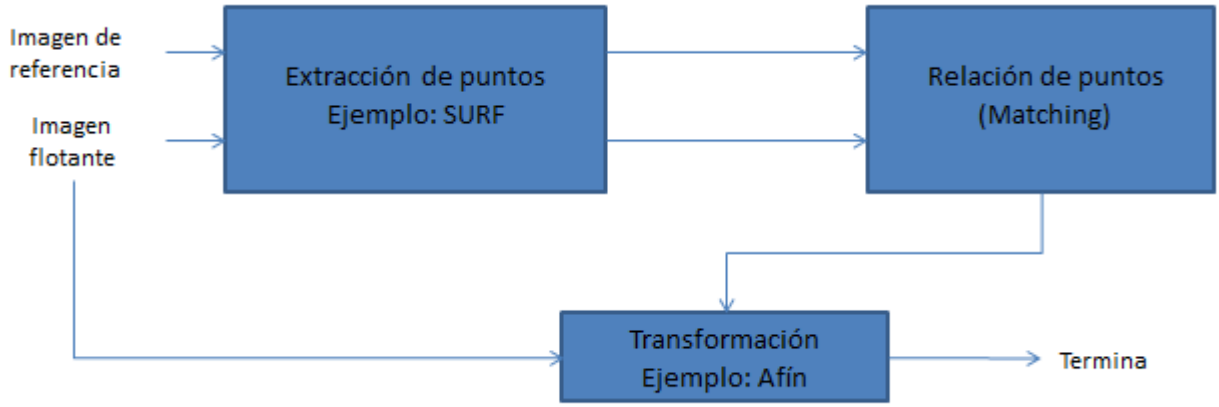


Figura 3.4: Esquema de alineamiento por puntos característicos

En el esquema podemos observar que primeramente se analizan las imágenes para extraer puntos con unas determinadas características. Seguidamente se intentan relacionar puntos comunes entre las dos imágenes y finalmente se utiliza una transformación geométrica que ponga en correspondencia los puntos de la imagen flotante con los puntos de la imagen de referencia. A primera vista podemos observar que, al contrario del alineamiento global, este algoritmo no es iterativo, disminuyendo considerablemente el tiempo que conlleva la búsqueda del alineamiento. El proceso que más tiempo va a llevar es precisamente elegir esa parte de la información de la imagen que es relevante y que se va a conservar lo suficientemente en tiempo para que pueda relacionarse con su imagen sucesora.

3.1.2.1. Extracción de puntos

La extracción de puntos es la parte más importante de esta técnica. Esta fase del algoritmo es la que más tiempo consume y es la que va a determinar los puntos de la imagen que se utilizarán para obtener la transformación que alinee las imágenes.

En nuestra implementación hemos utilizado un algoritmo de uso muy extendido en la literatura. El algoritmo que hemos utilizado se llama SURF [21] que es una mejora varias veces más rápida que su antecesora SIFT (Scale Invariant Feature Transform). Gran parte de la implementación de SURF está basada en el descriptor de SIFT por lo que comenzaremos describiendo qué hace SIFT y qué pasos sigue el algoritmo para extraer los puntos más característicos de la

imagen. SIFT es un algoritmo de análisis de imagen que sirve para detectar y describir características locales en las imágenes. Los descriptores de puntos que proporciona SIFT son invariantes a la escala, orientación, distorsión afín y parcialmente invariantes a los cambios de iluminación [15]. Gracias a estas propiedades, los puntos de las dos imágenes pueden relacionarse con éxito en la fase posterior del matching. Los pasos que sigue el algoritmo para extraer los puntos son:

1. **Detección de extremos**

   En esta fase del algoritmo son detectados los puntos de interés, para ello la imagen es convolucionada con filtros Gaussianos de diferentes escalas. Los puntos característicos se detectan en los máximos y los mínimos de la resta de las imágenes convolucionadas que tienen lugar a múltiples escalas. Dada la resta de Gaussianas $R(x, y, \sigma)$ viene dada por:

   $$R(x, y, \sigma) = P(x, y, k_i\sigma) - P(x, y, k_j\sigma)$$

   donde $P(x, y, k\sigma)$ es la convolución de la imagen original $T(x, y)$ con la Gaussiana $G(x, y, k\sigma)$ a la escala $k\sigma$, es decir:

   $$P(x, y, k\sigma) = G(x, y, k\sigma) * I(x, y)$$

   Por lo tanto la imagen $R(x, y, \sigma)$ entre la escala $k_i\sigma$ y $k_j\sigma$ es exactamente la diferenencia de las imágenes convolucionadas con sus respectivas imágenes a escalas $k_i\sigma$ y $k_j\sigma$. Una vez se han obtenido las imágenes $R(x, y, \sigma)$, los puntos característicos son identificados como mínimos y máximos de las imágenes $R(x, y, \sigma)$ a diferentes escalas. Esto se consigue comparando cada píxel en las imágenes con sus ocho vecinos en la misma escala y con nueve píxeles de su vecindad a diferentes escalas. Si el valor del píxel es máximo o mínimo entre todos los píxeles comparados entonces el píxel es candidato.

2. **Descarte de puntos**

   Primero se hace un descarte de puntos de bajo contraste, es decir, puntos cuyos píxeles vecinos tienen niveles de intensidad parecidos al del punto candidato para quedarnos con los puntos de mayor calidad. Para descartarlos se utiliza el valor de segundo orden del desarrollo de Taylor $T(x)$ evaluado cerca del píxel. Si este valor es menor que 0,03 entonces el punto candidato es descartado. El siguiente paso es eliminar puntos que están muy próximos al verdadero punto identificado como una esquina para definir bien la posición del punto-esquina e incrementar la estabilidad del punto. Para encontrar las curvaturas principales que identificarían el punto-esquina buscado, se deben resolver los autovalores de segundo orden de la matriz Hessiana porque esos autovalores son proporcionales a las curvaturas principales. Una vez obtenidos los autovalores, se eliminan los candidatos con un cociente de curvaturas principales elevado.

3. **Asignación de orientación**

   Para cada punto se asignan una o más orientaciones basadas en direcciones de gradiente locales. Este paso es clave para conseguir puntos invariantes a la rotación porque una vez obtenida la orientación de un punto las demás orientaciones se pueden representar relativas a ese punto consiguiendo así invarianza a la rotación. Para una determinada convolución $P(x, y, \sigma)$ de la imagen original I tenemos que la magnitud de su gradiente $m(x, y)$ y su orientación, $\theta(x, y)$ se obtienen diferenciando píxeles:

   $$m(x, y) = \sqrt{(P(x+1, y) - P(x-1, y))^2 + (P(x, y+1) - P(x, y-1))^2}$$

   $$\theta(x, y) = \arctan \frac{L(x, y+1) - L(x, y-1)}{L(x+1, y) - L(x-1, y)}$$

La magnitud y la dirección se calculan para cada píxel situado en la región vecina alrededor del punto característico en la imagen convolucionada. Con esta información se forma un histograma de orientaciones de 36 columnas, cubriendo cada columna 10 grados. Una vez rellenado el histograma, las orientaciones correspondientes al 80 % de las columnas más altas son asignadas al punto característico. Finalmente cuando se han terminado de obtener las orientaciones y las escalas, se forma para cada punto un descriptor que estará compuesto por 128 valores permitiendo así la identificación de los puntos en la posterior fase del matching.

El descriptor de SIFT es uno de los descriptores más utilizados para aplicaciones prácticas pero la alta dimensionalidad de su descriptor es una desventaja para la posterior relación de puntos en la fase del matching, porque el tiempo consumido por el conjunto detección + matching puede limitar a las aplicaciones de tiempo real. Sin embargo el detector de SURF está basado en la matriz Hessiana. El algoritmo integra las imágenes para reducir el tiempo de procesamiento y el descriptor está compuesto por una distribución de respuestas de Haar-wavelet [21] encontradas dentro de la vecindad de los puntos característicos. Nuevamente se utilizan la integración para construir los descriptores aumentando la velocidad y obteniendo tan solo un descriptor de 64 dimensiones reduciendo así el tiempo en la construcción del descriptor y en el posterior matching. La calidad de los puntos obtenidos por SURF es similar a la de SIFT [21] y el tiempo que emplea para conseguirlos es menor. Todas estas ventajas convierten a SURF en el algoritmo más apropiado para este proyecto.

3.1.2.2. Relación de puntos (Matching)

La relación de puntos consiste en identificar los pares de puntos extraídos en las dos imágenes que son iguales entre sí. Una vez extraídos los valores de las posiciones de los puntos emparejados podemos calcular la transformación que al aplicarla a la imagen se corresponda con la que se pretende alinear. Este paso previo es fundamental para registrar las imágenes y es el motivo por el cual se justifica la extracción de puntos en las dos imágenes. Para conseguir relacionar los puntos con éxito se siguen los siguientes pasos:

1. Se crean dos matrices $D1, D2$ para los puntos de la imagen de referencia y la imagen flotante respectivamente. El número de filas (64) es igual al número de descriptores en SURF, y el número de columnas es igual al número de puntos. Como puede haber diferente número de puntos extraídos en las dos imágenes, llamamos al número de columnas en la imagen de referencia y la imagen flotante $n1$ y $n2$ respectivamente

2. Se encuentran las mejores relaciones entre puntos. Para ello, por cada punto extraído en la imagen de referencia, se calcula la distancia euclídea a todos los puntos de la imagen flotante, y se le relaciona con el que menor distancia tenga. Este proceso se repite por cada uno de los $n1$ puntos extraídos en la imagen de referencia:

$$R = D2 - D1_i$$

donde $D1_i$ es la matriz de n columnas repetidas correspondientes al descriptor situado en la columna i de la matriz $D1$. Seguidamente se calcula para cada columna de la matriz R su norma y finalmente se obtienen n distancias sumando para cada columna todos los valores contenidos en ella. Con estos datos construiremos dos vectores: El vector ps contendrá las posiciones de la columnas (descriptores) que dieron menor distancia de la matriz $D2$ y el vector e estará compuesto por los errores que dieron distancias mínimas.

3. El vector ps contiene ya la información de relación de puntos entre las dos imágenes es decir, el valor de $ps(1)$ relaciona el punto 1 de la imagen de referencia con el punto de la imagen flotante que de $ps(1)$. Para encontrar los mejores puntos (aquellos que dieron menor error) se ordena de menor a mayor los valores del vector e desechando un porcentaje de puntos que haya dado mayor error. Este porcentaje va a variar según la resolución de la imagen y el tipo de transformación que queramos hacer. Cuanto mayor resolución tengan las imágenes más puntos necesitaremos para que la calidad del alineamiento y cuanto menos compleja sea la transformación menos puntos necesitaremos porque menos variables tendremos de la matriz de transformación. Tampoco es recomendable escoger todos los puntos porque puede que los puntos con mayor error no estén bien relacionados por lo que existe un compromiso entre elegir demasiados puntos para realizar una transformación compleja y el error que puedan proporcionar a esa transformación los puntos que tienen un mayor error de relación.

3.1.2.3. Transformación

Una vez identificados los pares de puntos que relacionan los puntos de las dos imágenes necesitamos realizar la transformación que pondrá en correspondencia las imágenes. La transformación se realizará a la imagen flotante para que se corresponda con la imagen de referencia. Para realizarla se debe resolver un sistema de ecuaciones donde las incógnitas son los parámetros de la transformación que elijamos y los datos conocidos el conjunto del par de puntos de las dos imágenes. Dado un conjunto de puntos conocidos U de la imagen de referencia relacionados con otro conjunto de puntos X de la imagen flotante y un conjunto de parámetros de transformación P no conocidos, la ecuación que los relaciona es:

$$U = XP$$

Multiplicando las matrices queda un sistema de ecuaciones que habrá que resolver para obtener los parámetros de transformación que buscamos. Este conjunto de parámetros varía según el tipo de transformación elegida (euclídea, afín o proyectiva). El modelo de transformaciones se puede describir como:

$$\begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} = P \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}$$

i va desde $1...n$ donde n son las correspondencias que relacionan los pares de puntos $(x', y')(x, y)$. La función de Matlab llamada $cp2tform$ busca una transformación proyectiva P tal que:

$$\left| \begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} - P \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} \right|$$

sea lo menor posible.

3.2. Segmentación de la imagen

Una vez alineada las imágenes deberemos extraer los objetos o las áreas de interés. El proceso por el cual se extraen objetos en movimiento o regiones de interés de la secuencia analizada es la segmentación [4]. Las áreas detectadas pueden ser realmente los objetos que se mueven o ruido causado por una mala alineación. El segmentador deberá elegir aquellas zonas que con mayor

probabilidad sean objetos y descartar las zonas que parezcan ruido o zonas pequeñas inconexas que no forman conjuntos compactos. Existen muchas técnicas de segmentación según el tipo de aplicación, en el caso que confiere a este PFC, una vez las imágenes están alineadas el problema de la detección de objetos se simplifica al de cámara estática. En este ámbito, nosotros hemos utilizado un segmentador impuesto por la empresa que contactó con nosotros y adaptado para funcionar para la detección de objetos con cámara fija. El segmentador está compuesto por las siguientes etapas:

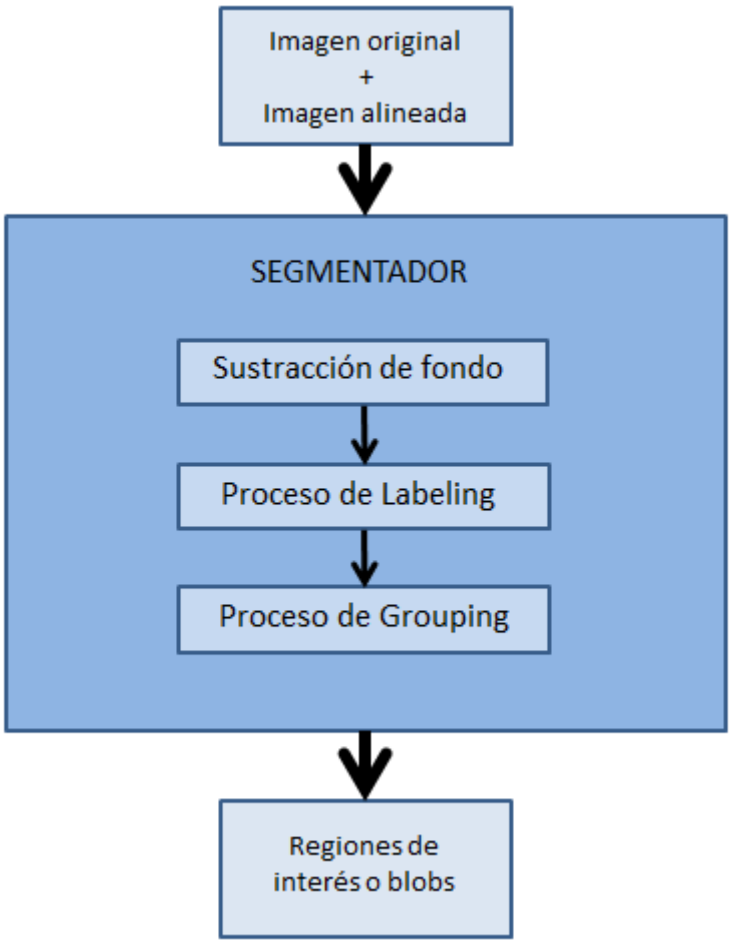


Figura 3.5: Etapas del segmentador

Según se puede apreciar en la imagen, el segmentador necesita dos imágenes, la imágen original o previa y la imagen siguiente alineada a la previa. Tras pasarle como entradas las imágenes, el segmentador realiza una primera etapa de preprocesado de las imágenes que consiste en realizar una **sustracción de fondo**. Al restar o substraer el fondo de la imagen conseguimos dejar al descubierto las zonas de interés. Para conseguir este fondo se debe realizar un modelado de fondo que en una situación de cámara estática tendría en cuenta muchas imágenes previas para construirlo. Pero en nuestro caso el modelado de fondo es inmediato puesto que el fondo está cambiando continuamente así que no interesa construir un modelo de algo que está en continuo cambio. Las siguientes dos etapas son de postprocesado ayudan a delimitar y conectar las zonas que quedaron al descubierto en la primera etapa. La primera de ellas se conoce como **etiquetado o labeling**, su función principal consiste en aplicar un umbral que marcará o etiquetará una zona como región de interés o no, es decir:

$$|frame_i - frame_{i-1}| \geq Th$$

La inecuación nos indica que se marcarán como regiones de interés las zonas que superen un determinado umbral Th después de que se haya restado el fondo de la imagen. Un umbral muy pequeño permitirá la aparición de ruido que no nos interesa para la detección de movimiento, mientras que un umbral demasiado grande puede hacer que perdamos los objetos de interés buscados.

La segunda etapa de postprocesado es el **agrupamiento o grouping**. Esta última etapa consiste en juntar o separar las regiones etiquetadas de acuerdo a la aplicacion de tres subprocesos:

- Delimitación de regiones: Se delimitan las regiones con un recuadro (blob) asegurándose de que este recuadro rodea a la región con un anchura y altura mínimas.
- Pre-combinación de regiones: En este subproceso se examina a que distancia están las regiones unas de otras y se juntan aquellas que sean menor de un determinado valor.
- Post-combinación de regiones: En este subproceso se observa si la combinación de regiones llevado en la pre-combinación tiene un tamaño mínimo, si no, se descarta la región.

Figura 3.6: Imagen con blobs a la salida del segmentador

El resultado se guarda en un archivo de texto indicando cuantos recuadros o blobs hay en una imagen en que posición se encuentran y cual es su anchura y altura a partir de esa posición. Estos procesos ayudarán a que la tarea de tracking sea mucho más sencilla de llevar a cabo tal y como indican Stauffer y Grimson en [22] .

3.3. Algoritmo de Tracking

El algoritmo de seguimiento o tracking es la última etapa que interviene en la detección de los objetos. El algoritmo determina, sigue y contabiliza las trayectorias de cada una de las regiones recuadradas marcadas por el segmentador. El tracking es útil para descartar aquellas regiones que marcó el segmentador por causa del ruido. Este tipo de regiones suele aparecer

en frames muy dispersos o si aparecen seguidamente no tienen coherencia espacial entre las imágenes consecutivas por lo que se descartan. El algoritmo de tracking también es capaz de seguir objetos que cruzan sus trayectorias y producen oclusiones en la imagen. Para el caso de este PFC la finalidad del algoritmo de tracking es determinar si existe movimiento o no y contar el número de objetos que aparecieron en la secuencia.

El algoritmo que hemos utilizado para seguir las trayectorias de los objetos se basa en utilizar el filtro de Kalman y obtener las propiedades estadísticas de cada una de las trayectorias que se siguen para descartar o validar una trayectoria como objeto [23]. El algoritmo está implementado por Johanna Broddfelt y el código se puede obtener directamente desde su memoria `http://www.geintra-uah.org/system/files/Thesis_-_Johanna.pdf`. En él se definen un conjunto de parámetros de entrada que permiten ajustar la sensibilidad del algoritmo para agrupar los puntos como objetos y contabilizar las trayectorias de éstos como objetos detectados. Los parámetros que intervienen en el algoritmo son:

- **Radio umbral:** El radio umbral es el valor que permite decidir cuántos de los puntos (centroides de blobs) que aparecen en la imagen pertenecen a un mismo objeto. Para poder establecer un área circular es necesario aparte del radio decidir cuáles de los puntos totales que aparecen en la imagen serán los centros de las áreas circulares. Los centros se determinan calculando todas las distancias para cada uno de los puntos y eligiendo para cada uno de ellos la distancia mínima. Si esta distancia mínima es menor que el radio se define ese centro como un nuevo objeto. Todos los puntos que caigan alrededor de ese centro con una distancia menor que el radio R se agruparán como un único objeto.

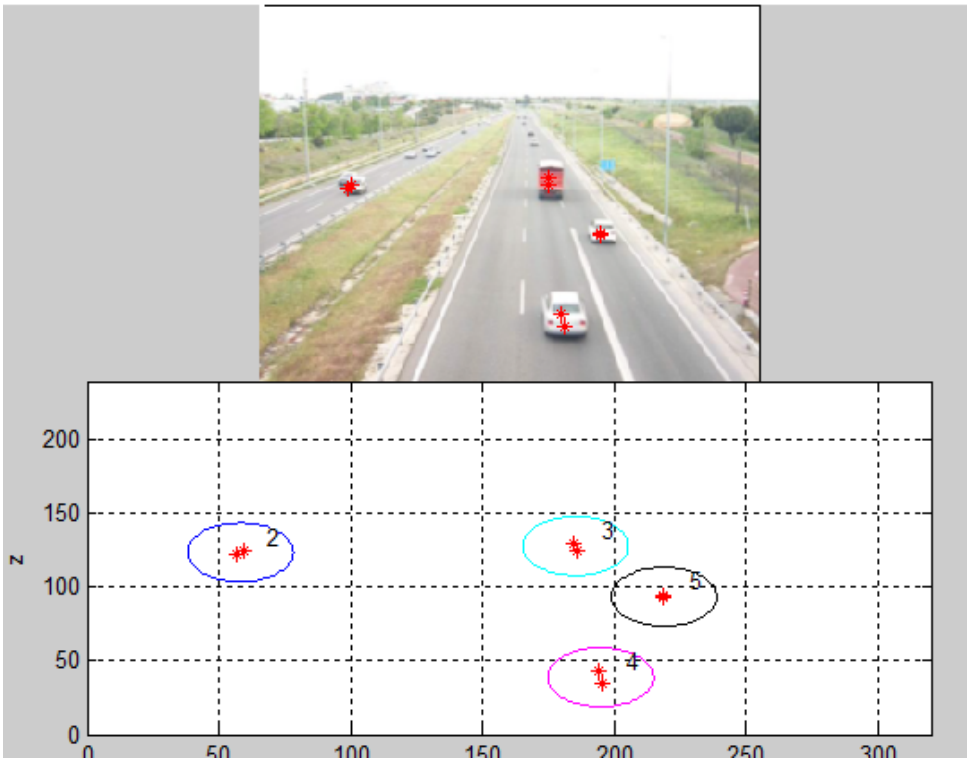


Figura 3.7: Los asteriscos rojos corresponden a los objetos móviles marcados. Los círculos que se muestran en la imagen se construyen utilizando el radio que hayamos definido. El centro de los círculos se corresponde con el punto que esté a menor e igual distancia de los objetos qué caen bajo el área del círculo. Por último los números que aparecen al lado de los círculos se corresponden con la cuenta de los objetos móviles que ha ido contabilizando el algoritmo de tracking.

- **Flag de borrado:** El flag de borrado decide cuántas imágenes como máximo pueden estar sin aparecer puntos en torno al centro del posible objeto etiquetado antes de que la trayectoria que describe sea eliminada. Los puntos que aparezcan deben estar a una distancia menor que el radio para que el contador de la bandera de borrado sea reseteado. Si decidimos elegir un valor de flag pequeño, los posibles objetos se desestimarán rápido ocasionándose errores al descartar objetos que se mueven lentamente, ya que estos tardan más frames en aparecer como detectados tras el segmentador. Por el contrario si elegimos un valor alto se dará mucho tiempo a que una trayectoria que empezó desde un punto ruidoso se marque como objeto.
- **Flag de validación:** El flag de validación determina durante cuántas imágenes como mínimo debe aparecer un posible objeto para que se determine como tal. Nuevamente existe un compromiso en la elección del valor de este flag. Un valor demasiado pequeño podrá ocasionar errores en el conteo de objetos al tener en cuenta trayectorias que comenzaron con puntos ruidosos. Un valor demasiado grande podrá dar lugar a que se desechen objetos que aparecen en un corto espacio de tiempo en la secuencia de vídeo.

Las salidas que produce el algoritmo tras su análisis son el número mínimo y máximo de objetos que aparecieron en un determinado momento de la secuencia y el número de objetos totales que aparecen en la secuencia completa. Estas salidas proporcionan un refuerzo a la tarea principal de este PFC que es detectar los objetos que se mueven y se utilizarán para ser comparadas con los datos reales que se etiquetaron a mano para cada vídeo.

4

Pruebas y resultados experimentales

A lo largo de este capítulo se van a presentar los diferentes resultados experimentales de las técnicas y algoritmos descritos en el anterior capítulo. El conjunto de pruebas se enfocará principalmente en determinar si existe movimiento real y se realizará una prueba adicional para determinar la capacidad de este algoritmo para contar el número de objetos que aparecen en la secuencia. Los datos que proporcionen estas pruebas se utilizarán para extraer las diferentes conclusiones de las soluciones propuestas a los problemas que plantea este proyecto.

4.1. Bases de datos utilizadas y modelos generales de prueba

La primera parte de esta sección se centra en describir con detalle el banco de pruebas utilizado así como sus características técnicas. La segunda parte se va a centrar en explicar el procedimiento general que sigue el algoritmo para la obtención de los resultados.

4.1.1. Bases de datos

La base de datos la hemos elaborado nosotros y comprende un total de 109 vídeos grabados. Cada vídeo tiene una duración media de 8 segundos y un framerate de 25 imágenes por segundo. Los vídeos los hemos grabado a una resolución de $640x480$ píxeles pero las imágenes que analiza el algoritmo tienen la mitad de resolución para que el tiempo de análisis no sea demasiado elevado. Los vídeos los hemos dividido en dos grupos: vídeos sin movimiento de cámara y vídeos con movimiento de cámara. Cada tipo de vídeo lo hemos subdivido a su vez en dos grupos: uno compuesto por secuencias donde aparecen objetos que se mueven y otro en las que no existe objeto alguno moviéndose. La siguiente tabla esquematiza el total de vídeos junto con sus grupos:

	Sin movimiento de cámara	Con movimiento de cámara
Sin Objetos	14	21
Con Objetos	32	42
Total	46	63

Figura 4.1: Tabla con número de vídeos por tipo.

Como puede observarse en la tabla, existen más vídeos grabados cuando la cámara está en movimiento porque el objetivo de este proyecto es detectar objetos cuando la cámara se mueve. No obstante hemos grabado vídeos sin movimiento de cámara para poder obtener una comparativa. Los 109 vídeos los hemos grabado en 14 escenarios diferentes, intentando para cada escenario grabar el mismo número de vídeos. Los escenarios filmados contienen todo tipo de entornos abiertos: autopistas, parques, calles de ciudad, etcétera. Las condiciones de iluminación de los vídeos son altas porque los vídeos fueron grabados en días despejados. En el conjunto de pruebas obtenido hemos grabado los escenarios desde cierta distancia para evitar errores de movimiento aparente ocasionado por la aparición del parallax. El parallax es el desplazamiento o cambio de posición aparente de los objetos que son observados desde puntos de vista diferentes y que en nuestro caso puede aparecer por el movimiento de la cámara. Por último los objetos móviles más habituales que aparecen en las grabaciones son coches y personas.

Cada vídeo tiene una etiqueta elaborada a mano que contiene información acerca del número de objetos totales que aparecen en el vídeo. Este dato se utilizará para realizar las oportunas comprobaciones con los resultados que se obtengan de las pruebas y así obtener conclusiones.

4.1.2. Modelos generales de prueba

Antes de analizar en detalle las formas de evaluación debemos explicar cómo funciona todo en conjunto y qué partes van a ser comunes a todas las pruebas. La imagen que aparece a continuación describe todos los procesos que sigue el algoritmo para detectar los objetos móviles:

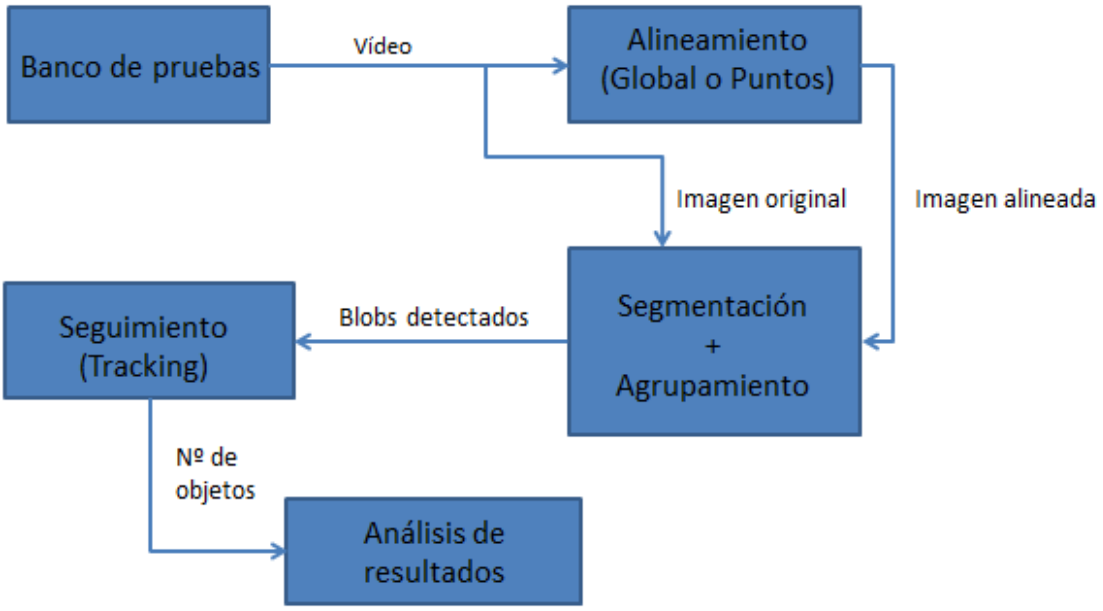


Figura 4.2: Algoritmo completo.

Según se observa en la imagen, se extrae un vídeo del banco de pruebas, seguidamente se alinean las imágenes cada una con su antecesora. La imagen original y la alineada se pasan al segmentador con el fin de detectar regiones de movimiento. Cada una se marca con su posición en la imagen y las dimensiones en ancho y alto que rodean a la región de movimiento (Blob). El algoritmo va realizar pruebas para cada uno de lo siguientes procedimientos que definimos ahora:

- Procedimiento 1: Modelo **sin alineamiento**. El modelo realiza todos los pasos descritos antes pero se salta el paso de alinear las imágenes. Este modelo pretende analizar la complejidad de la secuencia y comparar el impacto que tiene en la detección de objetos con otros modelos que sí alinean las imágenes.
- Procedimiento 2: Modelo **alineamiento global.** Este modelo utiliza la configuración de alinear las imágenes basándose en toda la información global de la imagen. Para realizar la alineación se ha optado por utilizar transformaciones que admiten sólo traslaciones en horizontal y vertical. El objetivo es que el algoritmo de búsqueda (Nelder-Mead) no invierta demasiado tiempo buscando la alineación ideal de las imágenes a cambio de no alinear de forma totalmente precisa.
- Procedimiento 3: Modelo **alineamiento por puntos.** Este modelo alinea las imágenes extrayendo puntos en ambas con el extractor de *SURF* estableciendo la relación o matching entre los puntos para finalmente realizar la transformación que ponga en correspondencia una con la otra. Para este modelo se admiten las transformaciones lineales más complejas ya que la mayor parte del tiempo se consume en la extracción de puntos.

Para cualquiera de estos modelos necesitaremos una cantidad o índice que sea mayor cuanto mayor sea la probabilidad de existencia de movimiento real, permitiéndonos decidir si hay movimiento real o no. Los índices que se van a utilizar son los siguientes:

- Número de blobs promedio. Para cada par de imágenes analizadas el segmentador proporciona cuántas regiones de movimientos se detectaron en ese momento de la secuencia. Si

promediamos el número de esas regiones por el total de las imágenes de la secuencia obtenemos el número de blobs promedio. El objetivo del número de blobs promedio es detectar si existe movimiento en una secuencia considerando algún tipo de umbral y ofrecer unos resultados comparativos con la salida del algoritmo de tracking.

- Número de objetos por vídeo. Esta salida se obtiene directamente después de pasar toda la secuencia del vídeo por el algoritmo de tracking. El algoritmo nos dirá cuantos objetos se detectaron y utilizaremos esta salida con dos propósitos: Comprobar si existe movimiento de objetos $nobjetos > 0$ y comparar en una prueba adicional su grado de concordancia con el valor real de los objetos que se mueven en la secuencia.

4.1.3. Pruebas específicas

Las pruebas que a continuación se van a describir se van a realizar para cada grupo de vídeos (con o sin movimiento de cámara) y para cada tipo de modelo. El objetivo es tener un escenario global de comparación con el que poder establecer cual de los modelos es más apropiado y que prueba arroja mejores resultados. Las pruebas que va incluir este PFC son:

1. Detección de objetos en movimiento. La primera prueba analiza si existen objetos en movimiento o no en el vídeo analizado. Para esta prueba utilizaremos el número de blobs promedio por video y la salida del número de objetos del algoritmo de tracking comparando estos resultados con el valor real que está etiquetado en el vídeo .
2. Contar el número de objetos en movimiento. La segunda prueba examina la veracidad que existe entre el número de objetos reales y el número de objetos que se obtiene de la función de tracking o seguimiento.

Detección de movimiento

La primera prueba intenta deducir si en el vídeo que se está analizando existe movimiento real de objetos o no. Para decidir esto es necesario previamente establecer algún tipo de umbral. Si el número de blobs promedio para ese vídeo es mayor que el umbral el vídeo es clasificado con movimiento de objetos y si está por debajo del umbral se clasificaría sin movimiento de objetos. Finalmente este resultado se compara con el real para ver si ha habido un acierto o un fallo. En este PFC se ha optado por utilizar la técnica leave one out para estimar el error que obtendrá el sistema con vídeos nuevos. Esta técnica consiste en dejar un vídeo fuera y establecer un umbral con los vídeos restantes y anotar como éxito o fallo si el vídeo excluido se cataloga bien o mal. Esto se realiza una y otra vez dejando cada vez un vídeo diferente como test y el resto como conjunto de entrenamiento, y se calcula el acierto promedio como el número de aciertos dividido entre el número de vídeos totales. El umbral se puede establecer de varias maneras. Nosotros hemos elegido dos:

- **Umbral tipo 1.** Este umbral se elige intentando que el umbral esté siempre por debajo de los vídeos catalogados con movimiento de objetos. El objetivo es no pasar por alto ninguna verdadera alarma porque interesa que si de verdad existe movimiento de objetos el algoritmo lo indique. A cambio obtendremos un aumento en el número de falsas alarmas debido a la elección de un umbral tan sensible.
- **Umbral tipo 2.** Este umbral es más equitativo que el anterior ya que va a intentar minimizar la suma de los errores por falsos negativos y falsos positivos. Para la elección de este umbral se establecen $n-1$ umbrales posibles donde n corresponde al número de vídeos totales menos el que se dejo fuera. Los $n-1$ umbrales se obtienen del vector del

número de blobs promedio para cada vídeo. Cada umbral se establece como sigue en la siguiente fórmula:

$$umbral_i = \frac{nblobs_i + nblobs_{i+1}}{2}, i = 1..n-1$$

siendo $nblobs_i$ la posición i del vector blobs promedio. Para cada $umbral_i$ se determina cuantos fallos se cometen entre los $n-1$ vídeos. Finalmente se elige aquel umbral que obtuvo menos fallos.

Finalmente realizaremos la clasificación de la detección de movimiento de objetos sin usar como dato el número de blobs promedio. En lugar de utilizar como dato el número de blobs promedio utilizaremos los datos que obtenga el algoritmo de tracking. El algoritmo de tracking proporciona para cada vídeo el número de objetos móviles que aparecen en el vídeo de modo que los resultados pueden clasificarse en dos clases de la siguiente forma: Los vídeos que den un resultado igual a cero se clasificarán como vídeos sin movimiento de objetos y los vídeos que den un resultado mayor que cero se clasificarán como con movimiento de objetos.

Contar el número de objetos en movimiento

La segunda prueba pretende analizar la capacidad que tiene este algoritmo para contar los objetos que aparecen en una escena. Una vez se realicen las pruebas de detección de movimiento se escogerá aquel método que dió mejores resultados para detectar objetos y se aplicará sin realizar cambios al conteo de objetos de la secuencia.

El total de experimentos de los procedimientos y pruebas que se van a realizar se muestran a modo de resumen en la siguiente tabla:

	Cámara estática	Cámara móvil
Procedimiento 1 Sin alineamiento	Detección de movimiento Umbrales tipo 1, 2 + tracking Conteo de objetos	Detección de movimiento Umbrales tipo 1, 2 + tracking Conteo de objetos
Procedimiento 2 Alineamiento global	Detección de movimiento Umbrales tipo 1, 2 + tracking Conteo de objetos	Detección de movimiento Umbrales tipo 1, 2 + tracking Conteo de objetos
Procedimiento 3 Alineamiento por puntos	Detección de movimiento Umbrales tipo 1, 2 + tracking Conteo de objetos	Detección de movimiento Umbrales tipo 1, 2 + tracking Conteo de objetos

Figura 4.3: Tabla resumen del conjunto total de pruebas

4.2. Diferentes pruebas realizadas

En la primera parte de esta sección se van a explicar las herramientas o medidas que utilizaremos para representar los resultados y se mostrará con ejemplos cuándo esos resultados son satisfactorios o no. En la segunda parte mostraremos todos los resultados obtenidos de las pruebas comentándolos con breves explicaciones.

4.2.1. Curvas ROC y modelos de clasificación

Algunos de los resultados globales que vamos a obtener en las pruebas se van a representar mediante una **curva ROC**. Una curva ROC es una representación gráfica que sirve para mostrar el equilibrio entre tasas de éxito y tasas de falsa alarma en clasificadores que etiquetan dos tipos de clases. La curva ROC proporciona herramientas para seleccionar los modelos posiblemente óptimos y descartar modelos subóptimos variando el umbral de decisión que determina la frontera entre los miembros de una clase y de otra.

Toda curva ROC para su representación necesita la información que proporciona un **modelo de clasificación**. Un modelo de clasificación es una función que permite determinar cuales de un conjunto de experimentos pertenecen a un grupo u otro, donde la frontera del clasificador entre cada clase debe establecerse por un valor umbral. Para el caso que confiere a este PFC el clasificador tendrá dos tipos de clases: Vídeo con movimiento de objetos o Vídeo sin movimiento de objetos. A este tipo de clasificadores se les conoce como clasificadores binarios porque solo podemos clasificar cierta instancia en una de las dos clases.

Según hemos visto un clasificador binario se puede ver como una función con dos posibles entradas y dos posibles salidas. Las entradas corresponderían a la clasificación predicha por el algoritmo y las salidas a la verdadera clasificación a la que pertenece una instancia. Tenemos en total cuatro posibles resultados que mostramos en la siguiente tabla también llamada matriz de confusión:

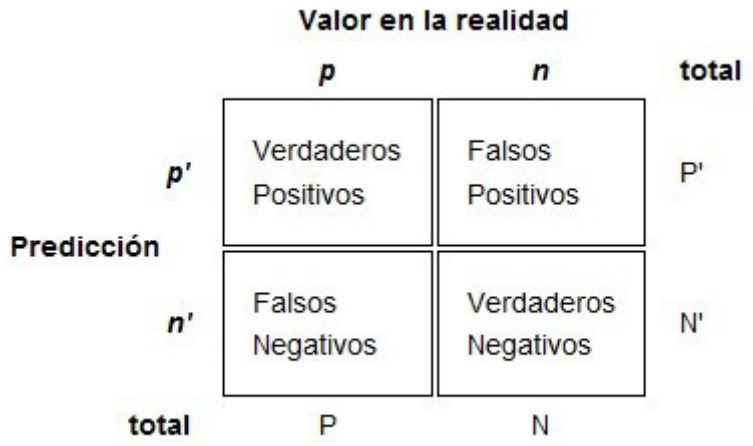


Figura 4.4: Matriz de confusión de un clasificador binario

Según se observa en la tabla podemos describir cuatro tipos de términos:

- Verdadero Positivo (VP): Si la predicción es p' y se corresponde con el valor p tendremos una clasificación con éxito.
- Verdadero Negativo (VN): Si la predicción es n' y se corresponde con el valor n tendremos un rechazo con éxito.
- Falso positivo (FP) : Si la predicción es p' y el verdadero resultado es n tenemos entonces una falsa alarma puesto que el algoritmo marcaría detección cuando realmente no hay.
- Falso negativo (FN) : Si la predicción es n' y el verdadero resultado es p tenemos un falso rechazo, es decir, el algoritmo no marcaría detección cuando realmente existe.

De estos términos se pueden derivar otros dos que tienen especial importancia para la curva ROC:

- Razón de Verdaderos Positivos (VPR): Este término nos revela el porcentaje de instancias positivas clasificadas correctamente, se le suele denominar sensibilidad y sigue la siguiente expresion:

$$VPR = \frac{VP}{VP + FN}$$

- Razón de falsos positivos (FPR): Este término nos dice el porcentaje de instancias negativas clasificadas erróneamente, se expresa mediante la siguiente formula:

$$FPR = \frac{FP}{FP + VN}$$

Estos dos últimos términos son los únicos necesarios para poder representar la curva ROC. La VPR nos dice hasta que punto un clasificador o prueba diagnóstica es capaz de detectar o clasificar los casos positivos correctamente de entre todos los casos positivos disponibles durante la prueba. La FPR define cuantos resultados positivos son incorrectos de entre todos los casos negativos disponibles durante la clasificación. La curva ROC se dibujará con FPR y VPR siendo respectivamente el eje de abscisas y eje de ordenadas reflejando en la representación los intercambios que existen entre verdaderos positivos y falsos positivos. Según estas especificaciones la mejor gráfica que podremos obtener será un punto en la esquina superior izquierda $(0, 1)$ ya que este punto representa por un lado que no tuvimos ningún falso negativo $FN = 0 \rightarrow VPR = 1$ y ningún falso positivo $FP = 0 \rightarrow FPR = 0$ a este caso se le conoce como clasificación perfecta. El caso opuesto seria una clasificación totalmente aleatoria y la curva ROC lo representaría como una línea recta que une dos puntos el $(0, 0)$ y el $(1, 1)$. Esta última representación significaría que si queremos obtener ningún falso negativo $FN = 0 \rightarrow VPR = 1$ asumiremos también ningún verdadero negativo ya que según esa recta para $VPR = 1$ tenemos que $FPR = 1$ dando $VN = 0$.

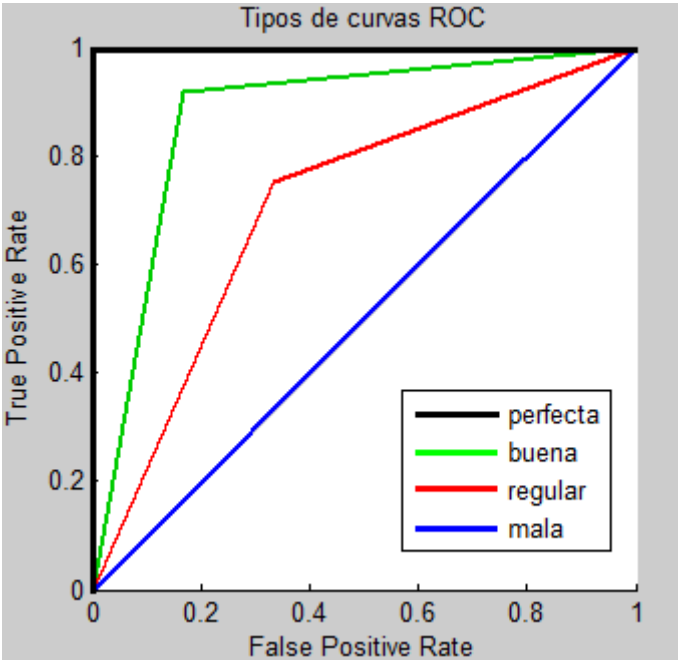


Figura 4.5: Ejemplos de curvas ROC

Existe un último término de especial importancia para caracterizar como de buenas han sido las clasificaciones. A este término se le conoce como Probabilidad de Acierto (PA) y sigue la siguiente expresión:

$$PA = \frac{VP + VN}{Total}$$

El término mide cuantas clasificaciones se hicieron correctamente sobre el total dándonos una medida aproximada de cuál será la probabilidad de clasificar correctamente un nuevo vídeo.

4.2.2. Pruebas obtenidas para cámara estática

A continuación se van a presentar todos los resultados relativos a los vídeos que fueron filmados sin mover la cámara. La presentación de las pruebas sigue en primer lugar una visión general utilizando curvas ROC, una tabla de clasificación para cada tipo de prueba conteniendo los tres tipos de procedimientos utilizados: Sin alineamiento, alineamiento global, alineamiento por puntos.

4.2.2.1. Comportamiento general del sistema

En este subapartado se pretende visualizar mediante la curva ROC el comportamiento global de nuestro sistema cuando la cámara no está en movimiento:

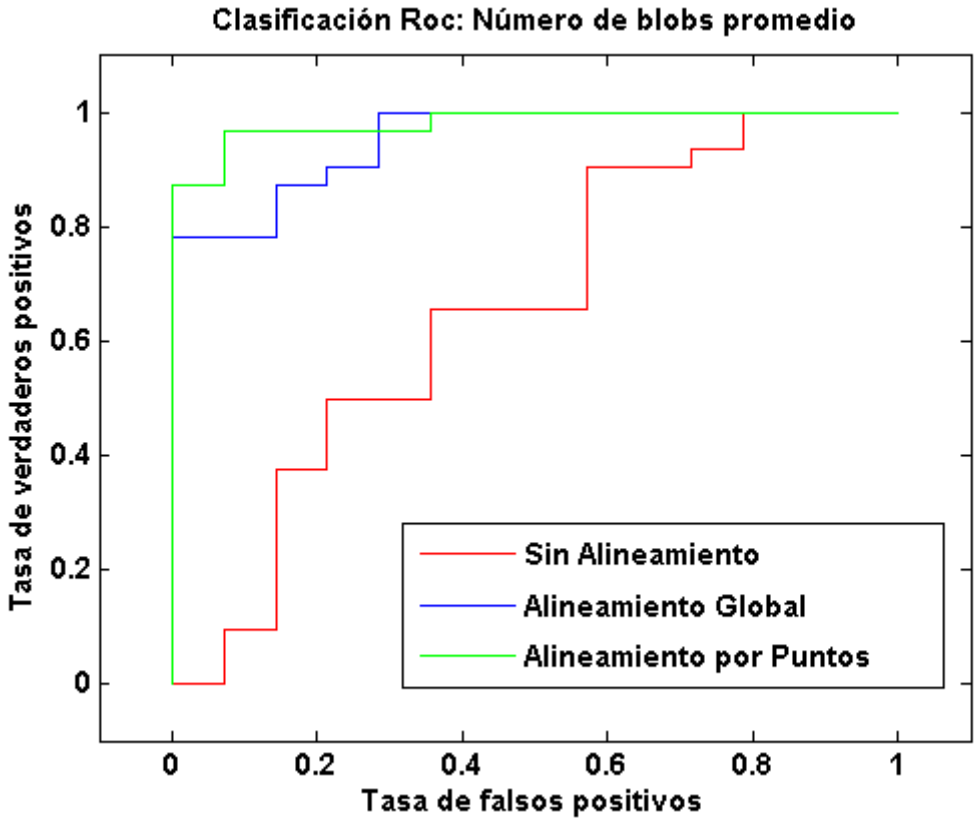


Figura 4.6: ROCs para el número de blobs promedio sin movimiento de cámara

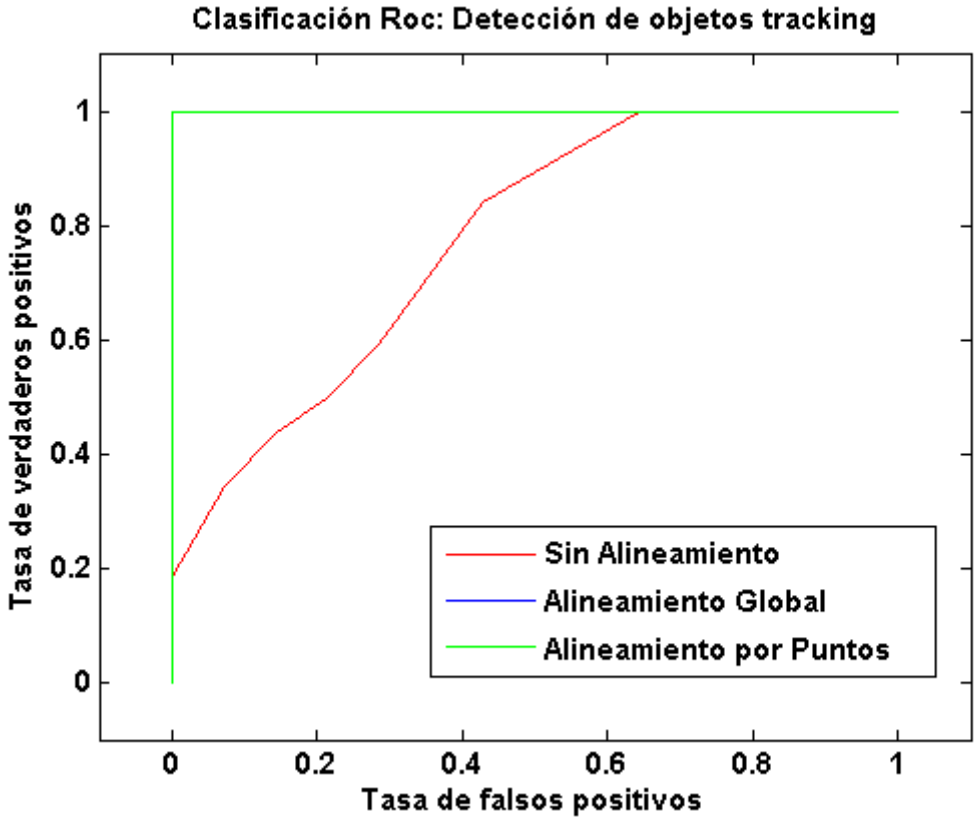


Figura 4.7: ROCs utilizando el tracking de objetos sin movimiento de cámara

Como puede observarse los resultados de las gráficas no parecen ser consistentes con los vídeos etiquetados como sin movimiento. Si los vídeos fueron grabados por una cámara completamente inmóvil todas las rocs deberían de ser ideales (área 1). Este hecho se investigó y se descubrió que tras analizar en detalle estos vídeos observamos que en algunos de ellos la cámara introducía pequeñas vibraciones. Este hecho inducía al segmentador a marcar regiones de movimiento originadas por cambios en el fondo y no por los objetos que se mueven aumentando el porcentaje de falsas alarmas especialmente cuando no alineamos las imágenes. Cuando alineamos las imágenes ya sea global o por puntos los resultados cambian bastante. Las ROCs comienzan a parecerse a esa roc ideal donde la clasificación de si existe movimiento es perfecta. Por tanto, según estos resultados podemos afirmar que el algoritmo también es capaz de corregir las vibraciones de la cámara. Finalmente parece existir una mejora cuando en lugar de analizar el número de blobs promedio analizamos si existe movimiento o no con el algoritmo de tracking.

4.2.2.2. Detección de movimiento: Prueba con umbral tipo 1

	# VP	# FP	# FN	% PA
Sin Alineamiento	31	11	1	73.9
Alineamiento Global	32	5	0	89.1
Alineamiento Puntos	31	5	1	87.0

Figura 4.8: Tabla con resultados del umbral tipo 1

La tabla muestra los distintos aciertos, fallos y probabilidad de acierto en la clasificación que han tenido lugar en los vídeos catalogados como sin movimiento. Los resultados se refieren a la prueba tipo uno que fue descrita en el anterior apartado. En la tabla puede observarse que a pesar de que los vídeos fueron filmados sin movimiento de cámara los peores resultados se consiguen cuando no alineamos las imágenes. Este hecho se explica por los efectos de las vibraciones que introduce la cámara para algunos vídeos. A pesar de este problema los resultados son bastante buenos, sin alinear la cámara obtenemos un 74 % de acierto y cuando utilizamos la alineación se obtienen resultados entorno al 90 %.

4.2.2.3. Detección de movimiento: Prueba con umbral tipo 2

	# VP	# FP	# FN	% PA
Sin Alineamiento	29	11	3	69.6
Alineamiento Global	32	5	0	89.1
Alineamiento Puntos	31	1	1	95.7

Figura 4.9: Tabla con resultados del umbral tipo 2

Con el umbral tipo 2 la probabilidad de acierto mejora casi en un 10 % cuando la alineación se hace con puntos característicos. Esta mejora se consigue porque la prueba con umbral tipo 2 elije un umbral menos restrictivo. El anterior umbral se centra en intentar no dar falsos negativos o alarmas que ocurrieron en realidad pero que no fueron detectadas por el algoritmo. En este caso el umbral se relaja obteniendo en los resultados menos falsos positivos y sacrificando en su lugar algunas verdaderas alarmas no detectadas.

4.2.2.4. Detección de movimiento: Prueba con algoritmo de tracking

	# VP	# FP	# FN	% PA
Sin Alineamiento	32	9	0	80.4
Alineamiento Global	32	0	0	100
Alineamiento Puntos	32	0	0	100

Figura 4.10: Tabla con resultados con el algoritmo de tracking

En este último tipo de prueba se consiguen los mejores resultados y se puede apreciar también la importancia de alinear las imágenes aunque el movimiento de cámara tenga unas ligeras vibraciones. Los resultados que refleja la tabla son casi perfectos excepto por el 80 % de acierto que se obtiene cuando no se alinean las imágenes. La mejora que se obtiene en este tipo de prueba se debe a que utilizamos el algoritmo de tracking para detectar si ha existido movimiento. No olvidemos que en las anteriores pruebas el dato que utilizábamos para detectar en un vídeo si existe movimiento o no era calcular el número de blobs promedio. Con esta última técnica la detección de movimiento es mas precisa porque el algoritmo analiza la relación espacial y temporal de los blobs detectados en toda la secuencia.

4.2.3. Pruebas obtenidas para cámara móvil

A continuación se van a presentar los resultados obtenidos para los vídeos grabados con cámara en movimiento. Las grabaciones se han tratado de realizar para todo tipo de escenarios intentando que la velocidad del movimiento de la cámara sea constante. El movimiento de la cámara se ha realizado para todo tipo de direcciones, de izquierda derecha, de arriba abajo y en círculos.

4.2.3.1. Comportamiento general del sistema

En este subapartado vamos a representar mediante la curva ROC el comportamiento general de nuestro sistema cuando la cámara está en movimiento:

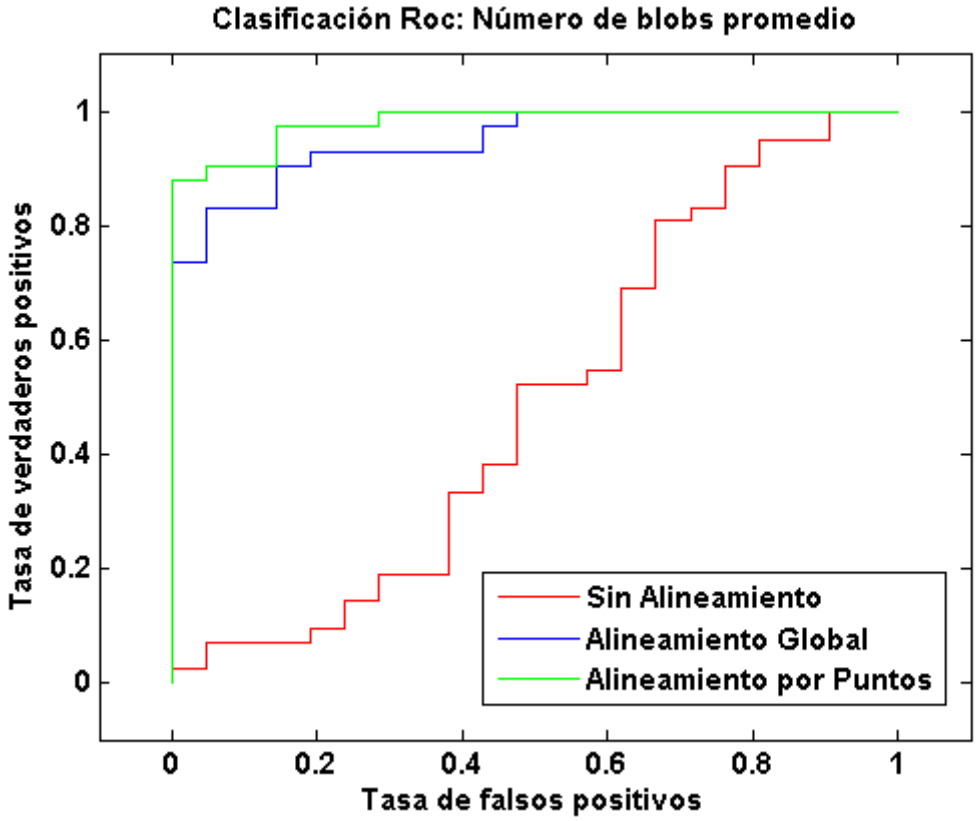


Figura 4.11: ROCs para el número de blobs promedio con movimiento de cámara

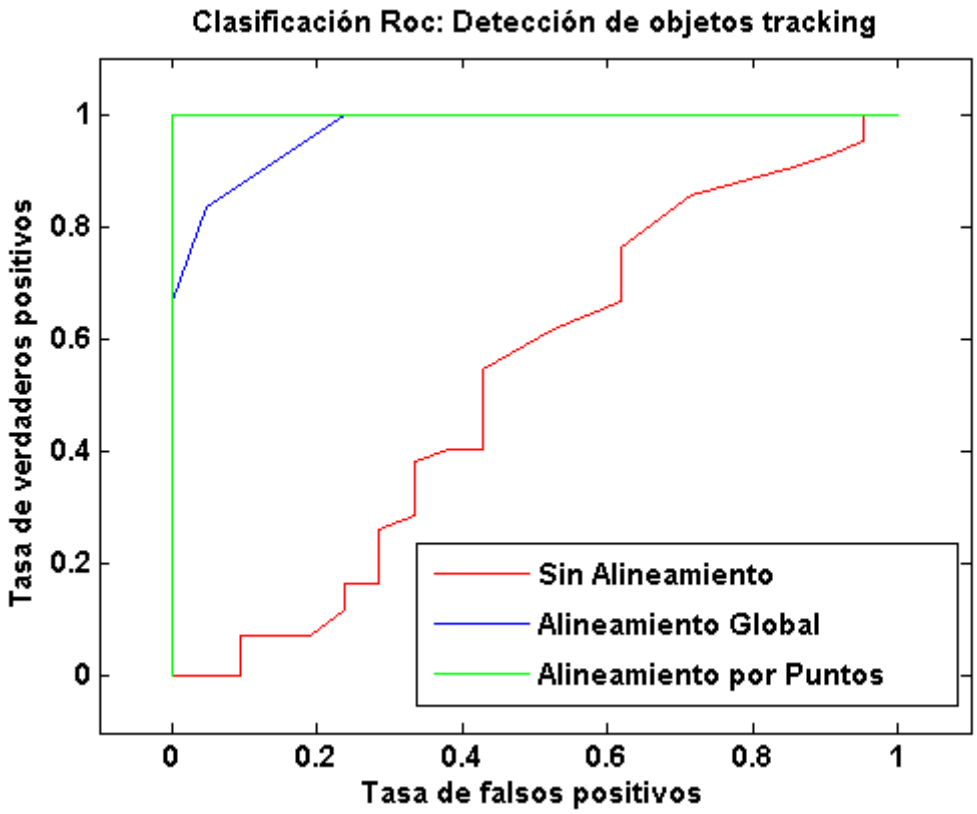


Figura 4.12: ROCs utilizando el tracking de objetos con movimiento de cámara

Como era de esperar los resultados obtenidos son algo peores cuando la cámara está en movimiento. La mayor diferencia se aprecia cuando utilizamos como dato para detectar movimiento el número de blobs promedio. Utilizando este dato la ROC muestra los peores resultados cuando no alineamos las imágenes. La curva roc dibujada en rojo zigzaguea en torno a la peor curva roc posible que es una recta que une los puntos $(0, 0)$ y $(1, 1)$. Cuando alineamos las imágenes la mejora es bastante notable. Las curvas roc se acercan bastante a la roc ideal cubriendo un área de casi 1. Cuando dejamos que el algoritmo de tracking decida si en una secuencia hubo movimiento o no, los resultados parecen mejorar notablemente en los tres procedimientos con respecto al anterior criterio. Cuando utilizamos el alineamiento por puntos la detección de movimiento es perfecta para los 63 vídeos mientras que el alineamiento global se acerca algo más a la curva ideal que utilizando el dato del número de blobs promedio para dar alarmas. Donde se aprecia mayor diferencia en los resultados es entre alinear las imágenes y no. Los resultados para cámara estática reflejan una proximidad mayor entre los resultados de alineamiento y sin alineamiento. Cuando la cámara esta en movimiento los resultados entre alinear o no parecen estar más alejados.

4.2.3.2. Detección de movimiento: Prueba con umbral tipo 1

	# VP	# FP	# FN	% PA
Sin Alineamiento	42	21	0	66.7
Alineamiento Global	41	11	1	80.9
Alineamiento Puntos	41	6	1	88.9

Figura 4.13: Tabla con resultados para el umbral tipo 1

La tabla para este tipo de umbral que prioriza la obtención de verdaderas alarmas sin importarle demasiado si da falsas alarmas (falsos positivos) detecta casi perfectamente todas las secuencias donde aparecen objetos moviéndose. Sin embargo el coste debido a lo restrictivo que es el umbral para no obtener alarmas no detectadas es bastante alto provocando una bajada considerable de la probabilidad de acierto con respecto al mismo tipo de prueba que para cámara estática. Cuando no alineamos las imágenes el algoritmo marca la existencia de movimiento en todos los vídeos analizados, es decir, el algoritmo detecta siempre movimiento aunque no haya realmente objetos moviéndose. Cuando alineamos las imágenes las falsas alarmas bajan casi a la mitad cuando alineamos globalmente y un 70 % cuando alineamos por puntos. A pesar de que los resultados no son demasiado buenos la introducción del alineamiento mejora bastante los resultados globales del algoritmo.

4.2.3.3. Detección de movimiento: Prueba con umbral tipo 2

	# VP	# FP	# FN	% PA
Sin Alineamiento	42	21	0	66.7
Alineamiento Global	38	5	4	85.7
Alineamiento Puntos	41	3	1	93.7

Figura 4.14: Tabla con resultados para el umbral tipo 2

Cuando utilizamos el umbral tipo 2 el intercambio entre falsos negativos y falsos positivos se iguala. Para este caso el mayor coste en alarmas no detectadas se lo lleva el alineamiento global. El alineamiento por puntos es el que introduce mayores mejoras cobrándose solo una verdadera alarma y reduciendo en tres los falsos positivos. Sin embargo, si no alineamos las imágenes, el algoritmo sigue detectando en todas las secuencias movimiento de objetos. Estos resultados se reflejan en la probabilidad de acierto obteniéndose una mejora global de casi el 5 % exceptuando el método de no alinear las imágenes.

4.2.3.4. Detección de movimiento: Prueba con algoritmo de tracking

	# VP	# FP	# FN	% PA
Sin Alineamiento	42	20	0	68.4
Alineamiento Global	42	5	0	92.1
Alineamiento Puntos	42	0	0	100

Figura 4.15: Tabla con resultados para algoritmo de tracking

El algoritmo de tracking obtiene nuevamente los mejores resultados. Cuando alineamos por puntos se consigue una clasificación perfecta y lo más importante es que para ninguno de los tres métodos se obtiene alguna verdadera alarma no detectada. La mejora en la probabilidad de acierto con el alineamiento global es casi del 7 % con respecto al anterior umbral e incluso conseguimos acertar 2 secuencias de vídeo en las que no aparecen objetos cuando no alineamos las imágenes. A pesar de esta leve mejora, los resultados del procedimiento de no alinear las imágenes siguen tendiendo a marcar todos los vídeos como con movimiento de objetos.

4.2.4. Prueba de conteo de objetos

Hasta ahora hemos visto que algoritmo detecta muy bien si hay movimiento o no cuando utilizamos sobretodo el procedimiento de alinear imágenes usando el alineamiento por puntos y aplicando a los resultados del segmentador un algoritmo de tracking basado en filtros de Kalman. La siguiente prueba tiene como objetivo mostrar hasta que punto el algoritmo es efectivo para contar los objetos que se mueven en las secuencias de cámara móvil. Para ello utilizaremos la configuración que mejores resultados dió para la detección de movimiento. La configuración utiliza el procedimiento de alinear las imágenes por puntos manteniendo las variables de entrada del segmentador y del algoritmo de tracking. Los resultados se van a representar mostrando dos tablas:

- La primera es una matriz de acierto en el conteo de objetos que muestra directamente los fallos o aciertos en el número de objetos. La matriz se va a mostrar a través de una tabla de 15 filas x 13 columnas. Cada columna corresponde a los objetos contados manualmente y que realmente aparecieron en la secuencia y las filas de la matriz corresponden a los objetos que detectó del algoritmo. A modo de ejemplo y para que se entienda si un vídeo ha sido catalogado con 3 objetos y el algoritmo ha detectado 4, en la casilla correspondiente a la fila 4 y columna 3 se le sumará un uno al resultado que ya hubiese ahí.
- La segunda tabla nos muestra el error absoluto entre los objetos detectados y los etiquetados y el coeficiente de correlación que existe entre los dos vectores de números.

4.2.4.1. Matriz de acierto

A continuación vamos a mostrar la matriz de acierto que se ha descrito al comienzo de esta sección. Debido a que no se ha grabado un vídeo para cada número posible de objetos,

la representación de la matriz ha sido acortada de tal forma que se puedan mostrar todos los sucesos ocurridos:

NÚMERO ESTIMADO \ NÚMERO REAL	0	1	2	3	4	5	...	12	14	16	19	22
0	21	0	0	0	0	0	...	0	0	0	0	0
1	0	10	1	0	0	0	...	0	0	0	0	0
2	0	3	1	0	0	1	...	0	0	0	0	0
3	0	2	3	4	1	1	...	0	0	0	0	0
4	0	0	0	0	1	1	...	0	0	0	0	0
5	0	0	3	1	0	1	...	0	0	0	0	0
6	0	0	0	0	0	0	...	0	0	0	0	0
7	0	0	0	0	0	1	...	0	0	0	0	0
8	0	0	1	0	0	0	...	1	0	0	0	0
9	0	0	0	0	0	0	...	1	0	0	0	0
...	...	...	...	...	...	...	...	....	...	...	...	...
13	0	0	0	0	0	0	...	0	0	1	0	0
14	0	0	0	0	0	0	...	0	0	0	0	0
15	0	0	0	0	0	0	...	0	1	0	1	1

Figura 4.16: Matriz de acierto para el conteo de objetos

Según está representada la tabla podemos apreciar que hay algunas filas y columnas que han sido omitidas debido a que no ocurrieron sucesos para esos casos y así facilitar la presentación de la matriz de acierto. A primera vista podemos ver que cuantos más objetos tratamos de acertar más alejado está el error en esos fallos. La desviación en el error de los vídeos etiquetados hasta cinco objetos no es demasiado elevado incluso podemos ver que se aciertan casi todos los vídeos etiquetados con 3 objetos. Sin embargo en los vídeos que fueron etiquetados con mas de 8 objetos la desviación se dispara. Estos últimos vídeos corresponden a grabaciones de autopistas donde aparecen una gran cantidad de coches en cortos periodos de tiempo lo que incrementa la dificultad en el conteo de objetos. Por último añadir que los resultados son coherentes con la detección de movimiento, 21 vídeos acertados cuando no hay objetos moviéndose y 42 con algún objeto moviéndose. También se puede observar que en una secuencia de vídeo etiquetada con 2 objetos móviles el algoritmo marca hasta 8 objetos diferentes moviéndose. Esto se debe a que los dos objetos que se movían en esa secuencia eran demasiado grandes para la elección del radio elegido. Al ser un radio bastante más pequeño que el tamaño de los objetos móviles, el algoritmo de tracking identifica un mismo objeto como varios (tantos como círculos de radio R quepan en ese objeto) introduciendo un error por exceso en el conteo de objetos.

4.2.4.2. Medidas en la calidad del conteo

De la anterior matriz de acierto se pueden obtener tres medidas importantes para mostrar hasta que punto es bueno el algoritmo contando objetos. La primera de las medidas es el error medio medio. El error medio la obtenemos sumando para cada uno de los valores absolutos obtenidos en la resta entre el vector de objetos etiquetados y el vector de objetos obtenidos por el algoritmo y dividiendo el resultado de la suma por el total de secuencias:

$$Media = \frac{\sum_i |Etiquetados_i - Detectados_i|}{N_{secuencias}}$$

La segunda medida es la desviación típica para ver en que cantidad se aleja el error de la media. La última es el coeficiente de correlación y nos sirve para calcular la relación de dependencia que existe entre los dos vectores de números. Un valor cercano a uno significará una buena dependencia entre el vector de etiquetas y el detectado y un valor cercano a 0 lo contrario. Los resultados obtenidos se muestran en la siguiente tabla:

Error Absoluto medio	Desviación estándar	Coeficiente de correlación
0.94	1.5	0.93

Figura 4.17: Tabla de medidas para el conteo de objetos

En la tabla puede apreciarse que los resultados obtenidos en las medidas son bastante buenos. La media del error en el conteo de objetos por vídeo es menor que uno y su desviación es algo mayor que uno. Esto se debe a que en los vídeos en los que aparecen muchos objetos el error está bastante por encima de uno mientras que los vídeos con pocos objetos está por debajo de uno. El coeficiente de correlación está muy próximo a 1 lo que indica que hay una relación de dependencia muy buena entre el número de objetos que aparecen en los vídeos que fueron etiquetados a mano y los detectados por el algoritmo.

5

Conclusiones y trabajo futuro

En este último capítulo se expondrán las conclusiones del trabajo realizado en este proyecto, y se propondrán los diferentes pasos a seguir para extender este trabajo en un futuro relacionado con la detección de objetos móviles con cámaras en movimiento y otras nuevas aplicaciones dentro del ámbito de la videovigilancia. La mayor parte de las conclusiones se extraen de los resultados globales del sistema mostrados en el anterior capítulo. También se van a proporcionar conclusiones relativas a la elección de variables o datos de entrada dentro de los subsistemas que componen el algoritmo general, así como la viabilidad de este trabajo para posibles aplicaciones de tiempo real.

5.1. Conclusiones

A lo largo de este proyecto se han desarrollado diferentes algoritmos con el fin de conseguir detectar objetos en movimiento en filmaciones con cámara móvil. Para ello se hizo un análisis de las diferentes propuestas que existen en la literatura para detectar objetos en movimiento con cámaras móviles. Mediante el estudio del estado del arte, se comprobó que existen principalmente dos formas diferentes para resolver el problema: los métodos basados en alinear las imágenes frente a los métodos que detectan el movimiento directamente en la secuencia. Se eligió el método de deshacer el movimiento de la cámara para detectar en fases posteriores a través de técnicas de segmentación para cámaras estáticas y tracking los objetos que aparecían en la secuencia. Este método fue el elegido para resolver el problema en lugar de utilizar los métodos basados en flujo óptico más vulnerables al ruido y con mayor coste computacional.

La técnica fundamental que utiliza el método de deshacer el movimiento de la cámara es alinear las imágenes mediante algún tipo de transformación lineal. El alineamiento de imágenes nos condujo a dos procedimientos o formas de realizarlo: El alineamiento global basado en utilizar toda la información de las imágenes para su correspondencia, y el alineamiento por puntos que extrae puntos de las imágenes con determinadas propiedades sobre las que se basa el paso posterior de alineamiento. Una vez alineadas las imágenes utilizamos un segmentador para poder substraer el fondo y etiquetar las regiones de interés detectadas. Una forma de validar las regiones detectadas como objetos, es utilizar un algoritmo de tracking que estableciera la correspondencia espacial y temporal de las regiones entre los distintos frames para dar un positivo a favor de la detección de movimiento.

Para probar todo el sistema se grabo un banco de pruebas con un total de 109 vídeos que fueron divididos en dos grupos. Unos vídeos se grabaron sin mover la cámara y otros con movimiento de cámara. Los objetivos de hacer esta división de grupos eran chequear que el algoritmo de cámaras móviles funciona para cámaras estáticas y comprobar que los algoritmos diseñados para cámaras estáticas fallan para cámaras móviles. Este hecho se muestra en las curvas ROC al comienzo de la subsecciones **4.2.2.1** y **4.2.3.1**. Las curvas correspondientes al procedimiento de no alinear las imágenes demuestran que existe un mayor número de aciertos en los vídeos de cámara estática que en los vídeos de cámara móvil. Cuando utilizamos vídeos con cámara móvil y no alineamos las imágenes el algoritmo detecta siempre movimiento de objetos. Esto ocurre porque al no alinear las imágenes, el algoritmo marca como movimiento el propio movimiento de la cámara haciendo imposible distinguir las secuencias con objetos de las de sin objetos. La siguiente implicación importante que se deriva de esta conclusión es que el solo hecho de alinear (deshacer el movimiento de la cámara) consigue filtrar la mayor parte de los falsos positivos de la no alineación tal y como muestran las gráficas. El trabajo realizado en este PFC consigue también mejorar los aciertos en los vídeos de cámara estática. Pero los vídeos catalogados como con cámara estática no deberían fallar en la detección del movimiento porque se supone que para este tipo de vídeos alinear las imágenes no es necesario porque no hay movimiento propio de la cámara. Pero tras comprobar estos vídeos se observó que existían leves vibraciones en la cámara que provocaban falsos positivos cuando no se alineaban las imágenes. Por tanto llegamos a la conclusión de que el algoritmo también sirve para eliminar los falsos positivos debidos a vibraciones de la cámara.

El objetivo de este PFC era desarrollar un sistema capaz de detectar movimiento en una secuencia de vídeo tomada con cámaras móviles. Para poder realizar la detección se elaboraron tres tipos de pruebas. Dos pruebas utilizando en la clasificación el número medio de regiones detectadas en el output del segmentador y otra prueba dejando al algoritmo de tracking decidir qué conjunto de regiones eran detectadas y validadas como objetos. El objetivo de las dos primeras pruebas era establecer un umbral que decidiese si las secuencias tenían movimiento o no. La diferencia entre las dos pruebas residía en la forma en que se establecían los umbrales. Una prueba hacía hincapié en proporcionar un umbral con el que no se obtuvieran alarmas no detectadas y la otra prueba proporcionaba un umbral más equitativo. Los resultados demuestran que las pruebas de establecimiento de umbral cumplen con su objetivo: cuando se utilizaba la prueba tipo uno las tablas demuestran que tan solo se comete un falso negativo pero eso contribuye notablemente a incrementar los falsos positivos bajando demasiado la probabilidad de acierto. Mientras que cuando utilizamos la prueba tipo dos, la probabilidad de acierto mejora con respecto a la anterior prueba pero se incrementan las alarmas no detectadas.

Sin embargo podemos ver que los mejores resultados se consiguen cuando dejamos que el algoritmo de tracking decida si existe movimiento o no. El algoritmo de tracking proporciona un número por cada secuencia de vídeo analizada que corresponde al número de objetos que detectó en ella. Si se decide alarma cuando el número es mayor que cero obtenemos unas muy buenas probabilidades de acierto llegando a superar a las de la prueba tipo dos. Esto es debido a que las anteriores pruebas utilizan como dato para determinar si existe movimiento o no el número medio de regiones detectadas en el segmentador. Este dato es bastante impreciso y es muy vulnerable a secuencias que tengan ruido o a vídeos en los que sea difícil alinear las imágenes debido a la complejidad del escenario que se filma. El algoritmo de tracking sin embargo trabaja mejor para este tipo de situaciones puesto que analiza la correspondencia espacial y temporal de las distintas regiones, de modo que si existen regiones que son marcadas en diferentes frames y en distintas partes del frame, el algoritmo de tracking consigue filtrar adecuadamente esas regiones reduciendo notablemente las falsas alarmas.

En cuanto a las técnicas de alineamiento, la técnica que mejores resultados obtiene es la que utiliza puntos característicos. Los resultados pueden observarse en las tablas y en las gráficas

ROC de la sección **4.2**. Todos los procedimientos que utilizaron la técnica de alinear por puntos consiguieron una mejora notable en la reducción de falsos positivos. El alineamiento por puntos combinado con el algoritmo de tracking obtiene un 100 % de aciertos en todas las pruebas, convirtiendo a este conjunto de algoritmos en los más efectivos para la detección de objetos. Por otro lado, el alineamiento global obtiene resultados bastante buenos pero no tan idóneos como la anterior técnica. Esto se debe a que el abanico de transformaciones lineales que hemos probado para el alineamiento global sólo utiliza el conjunto de las traslaciones y rotaciones para que no invirtiera demasiado tiempo en la búsqueda de la transformación ideal. Esto provoca que para algunos frames se deshaga el movimiento incorrectamente aumentando la probabilidad de que el segmentador detecte regiones falsas. Debido a la complejidad de los escenarios, la distancia a la que se graban y el movimiento de la cámara, algunas transformaciones son más complejas que una simple traslación o rotación y requieren de transformaciones afines o proyectivas. Sin embargo el alineamiento global lo probamos con vídeos aéreos grabados desde gran altura con la cámara fija mirando hacia el suelo. En estas situaciones el alineamiento global obtiene muy buenos resultados y se puede optimizar para llegar a funcionar en tiempo real. El alineamiento por puntos característicos funciona mejor para nuestro banco de pruebas porque puede abarcar transformaciones más complejas debido a que su coste computacional se centra en la extracción de puntos, mientras que el alineamiento global invierte la mayor parte de su tiempo en buscar de forma iterativa la optimización del error y la correlación de las imágenes alineadas mediante un algoritmo de búsqueda, dando lugar a una muy baja eficiencia computacional.

Durante la elaboración de este proyecto se probaron las dos técnicas de alineamiento en diferentes tipos de vídeos. Las técnicas se probaron con vídeos de baja resolución descubriéndose que para este caso en particular el alineamiento global obtiene mejores resultados que el alineamiento por puntos. La razón de esto es porque el alineamiento por puntos necesita encontrar esquinas con suficiente calidad para que los puntos puedan emparejarse con facilidad. Pero cuando tenemos imágenes con resoluciones bajas estas esquinas no están del todo definidas, lo que lleva al algoritmo a cometer errores en la fase de correspondencia de puntos. Sin embargo el alineamiento global no tiene ese problema porque intenta que toda la imagen coincida con su predecesora reduciendo la vulnerabilidad a la falta de calidad en la imagen.

El banco de pruebas elaborado para este proyecto está compuesto por vídeos con una resolución alta (640 x 480). Los vídeos se grabaron a esta resolución para evitar problemas al alineamiento por puntos. A pesar de ello encontramos dificultades en la elección del número de puntos a considerar para alinear las imágenes. Una vez extraíamos los puntos de las dos imágenes, elaborábamos una lista de emparejamientos ordenada de mayor a menor calidad de matching. Se descubrió que la elección del número de parejas de puntos para realizar la transformación era de especial importancia para el alineamiento. Cuando elegíamos una baja cantidad de parejas el alineamiento era bastante malo ya que la transformación proyectiva utilizada necesita una mayor cantidad de parejas de puntos para transformar la imagen con suficiente calidad. Sin embargo cuando elegíamos una cantidad que cubría más del 70 % de los puntos tampoco conseguíamos el alineamiento esperado, porque ese porcentaje tan alto de puntos contribuía a utilizar emparejamientos de mala calidad introduciendo errores en la transformación. La conclusión fue considerar siempre para cada alineación el 50 % de las parejas de puntos listadas alejándonos así del error de los extremos.

Las variables de entrada del segmentador también eran de especial importancia para la obtención de buenos resultados. La conclusión a la que se llegó es que había que elegir variables adaptadas al problema del movimiento de cámara y al tamaño de los objetos que esperábamos encontrar en los vídeos. De acuerdo con esto configuramos el segmentador para que no hiciera modelo de fondo y que detectara regiones como mínimo del 3 % de la imagen para evitar regiones ocasionadas por ruido o mala alineación de la imagen y ajustarnos así al tamaño mínimo esperado de los objetos. Finalmente la sensibilidad del segmentador se ajustó a un término medio

evitando los inconvenientes que surgían en la elección de extremos. La elección de una sensibilidad alta aumentaba la probabilidad del segmentador de detectar regiones falsas mientras que una sensibilidad baja podría ocasionar errores por omisión de regiones verdaderas.

La elección de las entradas del algoritmo de tracking también demostraron ser de especial importancia para la detección de objetos. El procedimiento que se siguió para calibrar las variables de entrada fue escoger 3 vídeos incluidos en el banco de pruebas y ajustarlas para conseguir un conteo de objetos similar al etiquetado manualmente. Cuando el calibrado para esos vídeos no era el correcto, el algoritmo de tracking cometía más falsas alarmas o se producían fallos si los objetos no aparecían durante suficientes frames en la secuencia. Tras realizar la calibración de las variables del algoritmo de tracking comprobamos para el resto de vídeos que la detección de movimiento era perfecta utilizando la técnica de alineamiento por puntos. Gracias a estos buenos resultados decidimos extender el algoritmo de tracking a una aplicación de videovigilancia muy utilizada, el conteo de objetos.

Los resultados del conteo de objetos se obtuvieron introduciendo las mismas variables de entrada al segmentador y al algoritmo de tracking que se utilizaron para la detección de movimiento. Los resultados se muestran en la tabla **4.16**. La tabla demuestra que en general es bastante complicado aproximarse a un conteo de objetos exacto especialmente cuando aparecen muchos objetos en la secuencia. La elección de las variables de entrada para esta aplicación parecen ser más críticas que para la detección de movimiento. Un vídeo etiquetado con pocos objetos a veces puede ser clasificado por el algoritmo como con muchos. Esto se debe a que el radio que utiliza el algoritmo de tracking es demasiado pequeño para el objeto que tratamos de seguir y por tanto divide el objeto en múltiples objetos de radio igual al indicado en el algoritmo de tracking dando lugar a error por exceso de objetos. La conclusión que podemos extraer del conteo de objetos muestra que es muy difícil obtener unos buenos resultados si no adaptamos con precisión las variables del algoritmo de tracking a las características de los objetos que aparecen en la secuencia.

5.2. Trabajo futuro

A lo largo de este capítulo se han extraído conclusiones de las diferentes técnicas de alineamiento utilizadas. Una de las conclusiones es que la técnica de alineamiento global es más robusta para alinear imágenes de baja resolución pero emplea demasiado tiempo en alinear imágenes que permitan transformaciones complejas (por ejemplo afines proyectivas). Un posible futuro trabajo podría enfocarse en encontrar o desarrollar algoritmos más modernos de búsqueda que converjan más rápidamente, más adaptados a las medidas utilizadas para evaluar la calidad del alineamiento. El objetivo sería conseguir un algoritmo más rápido y menos vulnerable a la calidad de la imagen.

El alineamiento por puntos es un algoritmo bastante rápido y ha demostrado ser bastante efectivo en detectar los objetos en movimiento. Sin embargo a pesar de su rapidez no podríamos utilizar este algoritmo para aplicaciones en tiempo real con un frame rate alto. El motivo está en la inversión de tiempo que requiere SURF para extraer los puntos. Sería interesante optimizar el algoritmo de SURF utilizando métodos que paralelizan algunas de las operaciones que intervienen en SURF [24]. De este modo el algoritmo de detección de movimiento trabajaría en tiempo real siendo muy útil sobretodo para aplicaciones militares de videovigilancia aérea. Los aviones de reconocimiento se mueven a relativa velocidad y necesitan de algoritmos rápidos que procesen los escenarios con rapidez para detectar objetos no autorizados.

En este proyecto también se exploró la posibilidad de detectar los objetos móviles directamente sin alinear las imágenes sobre vídeos que contenían fondos homogéneos como el mar. Este problema se puede resolver mediante la localización de puntos característicos de cierto tamaño ya

que en fondos homogéneos la detección de puntos característicos provienen de los objetos móviles que tratamos de detectar. Las pruebas que se realizaron para este tipo de vídeos siguiendo esta técnica fueron muy satisfactorias y podría ser un punto de partida para un trabajo relacionado con la detección de objetos móviles con cámaras en movimiento sobre fondos homogéneos.

Otro de los problemas que se podría abordar en la continuación de este trabajo es la filtración del movimiento aparente o parallax. En los bancos de pruebas elaborados para este proyecto se ha tratado de evitar el parallax grabando los objetos desde ciertas distancias. La distancia previene la aparición del parallax porque el escenario grabado es más parecido a un espacio de dos dimensiones. Este problema limita la eficacia de este algoritmo a unas determinadas condiciones, por eso sería interesante solucionar sus efectos en un trabajo futuro. Una posible solución podría estar basada en la utilización de dos cámaras que graban una misma escena desde puntos diferentes. El objetivo sería extraer información de tres dimensiones y compensar los efectos del parallax mediante métodos como los propuestos en [25].

En este proyecto hemos tratado de probar la eficacia de este algoritmo para una aplicación adicional que consiste en contar los objetos que aparecen en una escena. Los resultados obtenidos no son del todo buenos porque este trabajo se ha encaminado en resolver el problema de la detección de objetos móviles. El paso previo antes de contar los objetos es detectarlos, por eso este trabajo deja la puerta abierta a una posible nueva aplicación como puede ser el análisis y gestión de tráfico mediante mecanismos aéreos de videovigilancia.

Glosario de acrónimos

- **CCTV**: Circuito Cerrrado de Televisión
- **BLOB**: Binary Large Object
- **SUSAN**: Smallest Univalue Segment Assimilating Nucleus
- **SIFT**: Scale-invariant Feature Transform
- **SURF**: Speeded Up Robust Feature
- **UAV**: Unmmaned Aerial Vehicle
- **VP**: Verdaderos Positivos
- **VN**: Verdaderos Negativos
- **FP**: Falsos Positivos
- **FN**: Falsos Negativos
- **VPR**: Razón de Verdaderos Positivos
- **FPR**: Razón de Falsos Positivos
- **ROC**: Receiver Operating Characteristic
- **PA**: Probabilidad de Acierto

Bibliografía

[1] V.Chandran S.Denman, S.Sridharan. Abandoned object detection using multi-layer motion detection. *Proceeding of International Conference on Signal Processing and Communication Systems,*, 2007.

[2] B.Zitova J.Flusser. Image registration methods: a survey. *Image and Vision Computing*, (21):977–1000, 2003.

[3] R.Vezzani R.Cucchiara, A.Prati. Advanced video surveillance with pan tilt zoom cameras. *Proc of Workshop on Visual Surveillance VS at ECCV*, 2006.

[4] M. Valera and SA Velastin. Intelligent distributed surveillance system: a review. *Vision, Image and Signal Processing*, 152(2):192–204, 2005.

[5] Y-K. Jung, K-W. Lee, and Y-S. Ho. Feature-based object tracking with an active camera. *PCM '02 Proceedings of the Third IEEE Pacific Rim Conference on Multimedia: Advances in Multimedia Information Processing*, 2002.

[6] J.L Días Otegui. Sistemas de detección perimetral en infraestructuras penitenciarias. *Congreso Penitenciario Internacional*, 2006.

[7] T.Ellis B.James and P.Rosin. A novel method for video tracking performance evaluation. *Pattern Recognition*, pages 125–132, 2003.

[8] A. Koschan B. Abidi S. Kang, J. Paik and M. A. Abidi. Real-time video tracking using ptz cameras. *Proc. of SPIE 6th International Conference on Quality Control by Artificial Vision*, 5132:103–111, 2003.

[9] S.Tu S.Weng, C.Kuo. Video object tracking using adaptive kalman filter. *Visual Communication and Image Representation*, 2006.

[10] N. Gordon M. Arulampalam, S. Maskell and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Trans. on Signal Processing*, 2002.

[11] L.Gottesfeld Brown. A survey of image registration techniques. *ACM Computing Surveys*, 24(4):330–336, 1992.

[12] T. Kanade B.D. Lucas. An iterative image registration technique with an application to stereo vision. *7th International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[13] C.Harris and M. Stephens. A combined corner and edge detector. *Proceedings of the 4th Alvey Vision Conference*, pages 147–151, 1988.

[14] S. M. Smith and J. M. Brady. Susan - a new approach to low level image processing. *International Journal of Computer Vision*, 23(1):45–78, 1997.

[15] David G Lowe. Object recognition from local scale-invariant features. *Proceedings of the International Conference on Computer Vision.*, 2:1150–1157, 1999.

[16] I.Saleemi S.Ali S.Bhattacharya, H. Idrees and M.Shah. Moving object detection and tracking in forward looking infra-red aerial imagery. 2010.

[17] K. Kanatani. Camera rotation invariance of image characteristics. *Comput. Vision, Graphics, Image Processing*, 39(3):328–354, 1987.

[18] M.Shibata, Y.Yasuda, and M.Ito. Moving object detection for active camera based on optical flow distortion. *Proceedings of the 17th World Congress*, pages 14721–14722, 2008.

[19] M.Irani and P.Anandan. Global image alignment with any local match measure. Technical report, David Sarnoff Research Center, 2007.

[20] J.A. Nelder and R.Mead. Nelder-mead (simplex) method. *Compute Journal*, 7:308–313, 1965.

[21] H.Bay, T.Tuytelaars, and L.V.Gool. Surf: Speeded up robust features. Technical report, Katholieke Universiteit Leuven, 2006.

[22] C.Stauffer and W.Grimson. Adaptive background mixture models for real-time tracking. *IEEE Conference on Computer Vision and Pattern Recognition*, 2, 1999.

[23] Johanna Broddfelt. Tracking multiple objects with kalman filters part i. Master's thesis, MÄLRDALEN UNIVERSITY, Department of Computer Science and Electronics, 2006.

[24] B. Timothy, M. Lindley, and J.Helmsen. Gpu accelerating speeded-up robust features. *International Symposium on 3D Data Processing, Visualization and Transmission*, 2008.

[25] D. Crispell, J.Mundy, and G.Taubin. Parallax-free registration of aerial video. Technical report, Brown University, Providence, RI, USA, 2008.

A

Presupuesto

1) Ejecución Material

▪ Compra de ordenador personal (Software incluido)	1.200 €
▪ Material de oficina	150 €
▪ Total de ejecución material	1.350 €

2) Gastos generales

▪ sobre Ejecución Material	243 €

3) Beneficio Industrial

▪ sobre Ejecución Material	81 €

4) Honorarios Proyecto

▪ 960 horas a 6 €/ hora	5760 €

5) Material fungible

▪ Gastos de impresión	160 €
▪ Encuadernación	20 €

6) Subtotal del presupuesto

▪ Subtotal Presupuesto	7.614 €

7) I.V.A. aplicable

▪ 18 % Subtotal Presupuesto	1370,52 €

8) Total presupuesto

▪ Total Presupuesto	8984,52 €

Madrid, SEPTIEMBRE 2011

El Ingeniero Jefe de Proyecto

Fdo.: Héctor López Paredes

Ingeniero Superior de Telecomunicación

B

Pliego de condiciones

Pliego de condiciones

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

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

Condiciones generales.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

15. La garantía definitiva será del 4

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

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

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

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

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

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

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

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

Condiciones particulares.

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

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

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

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

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

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

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

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

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

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

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

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

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

C

Acreditación de méritos

Vaelsys
Image Recognition Software

A quien pueda interesar

Proyecto Fin de Carrera

"Detección y seguimiento de
objetos con cámaras en movimiento"

Por la presente certifico que Héctor López Paredes ha realizado el Proyecto Fin de Carrera con título "Detección y seguimiento de objetos con cámaras en movimiento" en colaboración con nuestra compañía desde Octubre 2010 a Septiembre 2011.

El proyecto en enmarca en el cuadro de colaboración con la Universidad Autónoma de Madrid por el cual nuestra empresa contrata con la misma servicios de apoyo tecnológico. Los resultados de dicha colaboración son propiedad de la empresa y pueden hacerse públicos con la autorización de ésta.

Sirva este documento como autorización para la publicación del Proyecto Fin de Carrera "Detección y seguimiento de objetos con cámaras en movimiento" basado en trabajos realizados en colaboración con personal de la empresa y profesores de la UAM. Los resultados de estos trabajos han sido validados por nuestros ingenieros e incorporados a nuestra oferta de software. Actualmente forman parte de varias propuestas comerciales a clientes para su licenciamiento e incorporación a sistemas de mando y control avanzados.

El profesor Manuel Sánchez-Montañés ha dirigido el Proyecto fin de carrera de Héctor López, como miembro del equipo de investigación de la Universidad que colabora con la empresa. Como responsable de los proyectos de I+D en colaboración con la UAM quedo a disposición para ampliar información respecto al trabajo realizado y su explotación comercial.

E. Cermeño

VAELSYS FORMACION Y DESARROLLO, S.L.
Crt. CANILLAS, 99 9º 3 MADRID
C.I.F.: B-84071976
TLF.: +34 497 34 82

Firmado: Eduardo Cermeño
Director General

Parque Tecnológico de Madrid – Edificio PCM. C/Santiago Grisolía nº2 – Tres Cantos 28760 - Tlf.: +34 91 804 62 48 fax:+34 91 497 34 71 - www.vaelsys.com info@vaelsys.com