Universidad de Alcalá

Campus Universitario
Dpto. de Teoría de la Señal y Comunicaciones
Ctra. Madrid-Barcelona, Km. 36,6
28805 Alcalá de Henares (Madrid)
Telf: +34 91 885 88 99
Fax: +34 91 885 66 99

D. SATURNINO MALDONADO BASCÓN, Catedrático de Escuela Universitaria del Área de Conocimiento de Teoría de la Señal y Comunicaciones de la Universidad de Alcalá,

CERTIFICA

Que la tesis "**Estudio, diseño y optimización de técnicas de visión artificial para su aplicación a los sistemas de videovigilancia**", presentada por D. Pedro Gil Jiménez, realizada en el Departamento de Teoría de la Señal y Comunicaciones bajo mi dirección, reúne méritos suficientes para optar al grado de Doctor, por lo que puede procederse a su depósito y lectura.

Alcalá de Henares, 21 de octubre de 2009.

Fdo: Dr. D. Saturnino Maldonado Bascón

Universidad de Alcalá

Campus Universitario
Dpto. de Teoría de la Señal y Comunicaciones
Ctra. Madrid-Barcelona, Km. 36,6
28805 Alcalá de Henares (Madrid)
Telf: +34 91 885 88 99
Fax: +34 91 885 66 99

D. Pedro Gil Jiménez ha realizado en el Departamento de Teoría de la Señal y Comunicaciones y bajo la dirección del Doctor D. Saturnino Maldonado Bascón, la tesis doctoral titulada "**Estudio, diseño y optimización de técnicas de visión artificial para su aplicación a los sistemas de videovigilancia**", cumpliéndose todos los requisitos para la tramitación que conduce a su posterior lectura.

Alcalá de Henares, 21 de octubre de 2009.

EL DIRECTOR DEL DEPARTAMENTO

Fdo: Dr. D. Manuel Rosa Zurera.

Universidad de Alcalá

Escuela Politécnica

Tesis Doctoral

Estudio, diseño y optimización de técnicas de visión artificial para su aplicación a los sistemas de videovigilancia.

Autor: Pedro Gil Jiménez
Director: Dr. Saturnino Maldonado Bascón

21 de octubre de 2009

Resumen

En esta tesis doctoral se realiza un estudio sobre distintas técnicas y algoritmos existentes en la actualidad en el campo de la visión artificial aplicadas a la videovigilancia, y se han desarrollado nuevos algoritmos, aplicando técnicas de tratamiento de señales, y más concretamente de tratamiento de imágenes, que son de aplicación en sistema de videovigilancia de segunda generación.

Con dichos algoritmos se pretende aumentar la fiabilidad de este tipo de sistemas, tratando de reducir la probabilidad de error del conjunto, especialmente en sistemas de videovigilancia donde los equipos empleados, especialmente los de captación de imágenes, no son de gran calidad. Al mismo tiempo, se ha tratado de reducir la complejidad computacional de los algoritmos, lo que nos permitirá cumplir con las restricciones de funcionamiento en tiempo real de un sistema de segunda generación. En concreto, la tesis doctoral se ha centrado en el estudio y desarrollo de nuevos esquemas de procesado en los siguientes aspectos relacionados con la vigilancia:

- Esquemas de estimación de fondo a partir de una serie de secuencias de vídeo captadas por una videocámara estándar. En este caso, se ha profundizado en el análisis de escenas complejas, que incluyen movimiento constante del fondo, y cambios de iluminación, lo que permite obtener una estimación de comportamiento del fondo para cada zona de la imagen, y con ello poder plantear esquemas de detección de movimiento con mayor robustez frente a las falsas alarmas.

- Esquemas de estimación de distancia a partir de una secuencia estéreo captada por dos cámaras de vídeo independientes trabajando simultáneamente. El algoritmo determina, a partir del análisis de dos imágenes simultáneas de la misma escena captada desde distintos puntos de vista, la distancia a la que se encuentran los objetos en la escena del sistema de captación de imágenes. Como el algoritmo únicamente se centra en los objetos detectados, el tiempo de cálculo se ve notablemente reducido, a la vez que se reduce el número de falsas correspondencias y errores por oclusiones.

- Esquemas de cálculo de profundidades a partir del desenfoque de la imagen. Respecto al apartado anterior, las técnicas de estimación de distancias mediante el desenfoque tienen la ventaja de que sólo necesitan una imagen de la escena para poder calcular la profundidad de los objetos. De esta forma, es posible reducir el

precio del sistema, a costa de reducir la precisión de la medida que, en general, suele ser peor que en sistemas estéreo.

- Esquemas de clasificación de formas. A partir de los objetos detectados, ya sea mediante detección de movimiento o cualquier otro esquema, se ha desarrollado un algoritmo capaz de clasificar dichos objetos dentro de un grupo de formas determinado.

Abstract

In this thesis, we carry out a study about different techniques and existing algorithms in the field of computer vision applied to videosurveillance systems. Furthermore, new algorithms have been developed, using signal processing techniques, and specifically image processing ones, for second generation videosurveillance systems.

With such algorithms, we aim to increase the reliability of this kind of system, focusing specially in those cases where the quality of the devices concerned, mainly the videocameras and image capturing systems, is low. Computational complexity of the algorithms has been another aim to follow. Complexity reduction could allow the system to cope with real time constrains, a requirement normally needed in common videosurveillance systems. In particular, the work is focused in the study and development of new signal processing schemes dealing with the following topics:

- Background estimation techniques computed from videosequences using standard videocameras. In this thesis, we have focused on the analysis of complex sequences that include constant background motion and illumination changes. Thus, we have developed an algorithm which obtains an estimation of background behaviour, and thus, allowing the design of further motion detection schemes that improves its performance against false alarm probability.

- Distance estimation from stereo image pair, using two independent cameras. The algorithm computes the distance from the object to the camera system analysing two images of the same scene captured from different points of view. Since the algorithm only takes into account the objects detected, reduction in computation time is straightforward, as well as in false alarms and occlusion errors.

- Depth estimation from image defocus. Compared with stereo techniques, depth from defocus has the advantage of using only one camera, instead of two or more needed in stereo algorithms. Thus, it is possible to reduce the cost of the system, at the expense of some lost of accuracy, which, in general, is lower compared with stereo systems.

- Shape classification techniques. From the detected objects, whether they are obtained using a motion detection algorithm, or any other technique, an algorithm able to classify each object into one of some predefined shapes has been developed.

Índice general

Lista de figuras

Lista de tablas

Capítulo 1

Introducción

1.1. Introducción a la videovigilancia

El objetivo principal de un sistema de vigilancia es la prevención de actuaciones que puedan violar la seguridad de un lugar supervisado, como podrían ser el vandalismo o intrusión, o la monitorización y registro de determinados espacios, como puede ser por ejemplo, una carretera donde interese hacer un recuento de automóviles que circulan por dicha vía.

Un sistema de vigilancia genérico (véase [Foresti00]) consiste en una serie de sensores, situados en posiciones estratégicas del espacio a vigilar y conectados, a través de medios de transmisión adecuados, al correspondiente receptor. Aunque el sistema de sensores puede estar formado por distintos tipos de elementos, en esta tesis únicamente se considerarán aquellos que proporcionan imágenes para su análisis. Esto no implica, sin embargo, que este tipo de sensores no pueda ser combinado con otros para mejorar el comportamiento del sistema. Por tanto, el sistema de sensores estará formado generalmente por cámaras de vídeo, aunque puede incluir en ocasiones sensores de infrarrojos para permitir su funcionamiento en condiciones de baja visibilidad, típicamente por la noche.

Antes de la popularización de los sistemas y algoritmos de inteligencia artificial y equipos de procesado, los proveedores de servicios de videovigilancia recurrían a la implantación de servicios donde la información era procesada por operadores humanos. Esto supuso la aparición de lo que se conoce como *sistemas de vigilancia de primera generación*. El procesado de la información en este tipo de sistemas corresponde exclusivamente a operadores humanos, que son los que deben evaluar en todo momento la actividad dentro del espacio vigilado, y tomar las actuaciones correspondientes.

Este tipo de sistema, aunque sencillo en su implementación, trae consigo una serie de problemas y limitaciones en aplicaciones reales. En primer lugar, al ser la información procesada por operadores humanos, existe obviamente un problema en cuanto a la fiabilidad, ya que por regla general, el rendimiento de un operador humano decrece con el paso del tiempo, el cansancio y la cantidad de información que éste debe procesar. Además, en

el caso de que el sistema de vigilancia deba operar en espacios públicos, el hecho de que la información deba ser procesada por operadores humanos puede hacer sentir a los usuarios de dichos espacios que su privacidad está siendo violada, con el consiguiente rechazo por parte del público a estos sistemas.

En este contexto, según [Foresti00], los *sistemas de vigilancia de segunda generación*, o avanzados, basados en técnicas de procesado de la información, son una alternativa ideal a los sistemas de primera generación. La característica principal de esta tecnología es la explotación de las técnicas de procesado para seleccionar automáticamente un pequeño porcentaje de la información disponible, el cual será probablemente el más interesante para la tarea de vigilancia requerida y para ser finalmente valoradas por el personal de vigilancia. Esto permite alcanzar tres objetivos importantes:

- Disminuir los costes del sistema, disminuyendo la dedicación de operadores humanos y reduciendo los costes de almacenamiento de información no necesaria. Solamente se almacenarán los eventos más importantes.
- Mejorar las condiciones de trabajo de los operadores, liberándolos del trabajo más monótono y repetitivo.
- Aumento de la aceptación del servicio de vigilancia de ámbito público, gracias a la transparencia de los sistemas de vigilancia y la reducida intervención de los operadores humanos dentro de la privacidad de los usuarios.

Dentro de los sistemas de vigilancia de segunda generación, podemos distinguir dos grandes grupos:

- Sistemas basados en sensores activos.
- Sistemas basados en sensores pasivos.

Los sensores activos son aquellos que actúan sobre el entorno, al emitir algún tipo de señal que es reflejada por los objetos de la escena, proporcionando información sobre la actividad en el escenario vigilado. Los más comunes que podemos encontrar son:

- Sensores acústicos: Emiten una señal específica y miden sus reflexiones y alteraciones. Son de bajo coste pero, desafortunadamente, tienen un rango de detección muy pequeño.
- Sensores mediante láser: Basándose en el efecto Doppler, los radares que utilizan el láser detectan la distancia a los objetos midiendo el tiempo de viaje de una señal emitida por los propios sensores y reflejada en el objeto. Las principales desventajas de estos son la baja velocidad de escaneo y la baja resolución espacial, además de razones de seguridad, pues pueden dañar la retina de las personas afectadas por su zona de influencia.

- Sensores basados en radar: Los radares de onda milimétrica siguen el mismo principio de funcionamiento que los sensores basados en láser. Su coste es mayor, pero son más robustos ante inclemencias del tiempo, como la lluvia o la niebla, que los sensores láser. Desafortunadamente siguen teniendo resolución espacial baja y velocidad lenta de escaneo.

Por otro lado, tenemos los sensores pasivos, los cuales no interfieren el entorno al adquirir los datos y, por tanto, incorporan ventajas con respecto a los sensores activos. Una de las principales ventajas de su uso en la visión por computador es que permite detectar información visual sin hacer casi ninguna modificación en las infraestructuras actuales. Pero, desafortunadamente, los sensores pasivos (sensores de visión en nuestro caso) no miden cantidades de manera directa, requiriendo una etapa de procesado complejo. Incluso son menos robustos que los radares de onda milimétrica en presencia de condiciones climatológicas adversas de niebla, nieve, lluvia o luz solar directa.

El uso de sensores activos, empleando las medidas de la alteración de las señales emitidas por los propios sensores, tiene algunas ventajas respecto al uso de la visión por computador. En concreto, los sensores activos miden las cantidades de interés de manera directa. Por ejemplo, un dispositivo radar devuelve directamente la distancia o la velocidad relativa de un objeto. Por contra, los sensores de visión pueden detectar la distancia o la velocidad relativa a través de un complejo procesado de secuencias de imágenes. Además, debido a este complejo sistema de procesado, los sensores pasivos van a requerir, en general, el uso de un mayor número de recursos computacionales que los activos.

Pero existe un gran inconveniente que desaconseja el uso de sensores activos en favor de los pasivos, y es la interferencia potencial existente entre sensores en un escenario donde puede haber varios sensores del mismo tipo. Así pues, los sensores activos pueden interferir unos con otros, decrementando así su fiabilidad y utilidad. Este problema llega a ser mucho mayor en entornos de exterior no estructurados. Además, el máximo nivel de señal debe cumplir con algunas normas de seguridad y por supuesto debe ser inferior al umbral de seguridad. Por tanto, teniendo en cuenta todo esto, el uso de sensores pasivos obtiene importantes ventajas sobre los activos.

Dentro del campo que nos interesa, la visión por computador, los sensores pasivos van a ser las cámaras. Obviamente, mientras que el uso de otro tipo de sensores podría tener la ventaja de extender las capacidades sensibles más allá de las posibilidades del ser humano, la visión por computador falla en las mismas situaciones en las que los humanos no pueden ver, por ejemplo condiciones de lluvia intensa, niebla, oscuridad sin iluminación específica, etc. No obstante, éste último inconveniente puede solventarse con el uso de otro tipo de sensores. Por ejemplo, puede mejorarse la visión en entornos con poca luz o incluso de noche con un sistema de iluminación, o con el uso de cámaras térmicas.

Para cumplir estos objetivos, un sistema de videovigilancia de segunda generación o avanzado debe incorporar, además de los dispositivos de captación de imágenes y los

medios de transmisión adecuados, los equipos y los algoritmos de visión artificial que realicen las funciones encomendadas. Normalmente, los equipos empleados para realizar el procesado de vídeo suele consistir en un ordenador con recursos suficientes, o bien sistemas de microprocesadores diseñados específicamente para dicho fin, mientras que los algoritmos son en realidad, los programas o rutinas que están siendo ejecutados por dichos equipos. Las características principales que dichos algoritmos deben cumplir son:

- Sencillez y velocidad: Como en todo sistema informático, la complejidad computacional de un programa está ligada con el precio del equipo, ya que cuanto más complejo sea y mayor cantidad de recursos necesite el algoritmo, más caro será el equipo empleado en el sistema. Como una de las características principales de un sistema de videovigilancia es la operación en tiempo real para poder evaluar la actividad en la zona vigilada en cada momento, cuanto más simple sea el algoritmo, menor velocidad se requerirá del equipo que va a ejecutarlo, con la consiguiente reducción del precio final del sistema.

- Fiabilidad: El objetivo final de un sistema de videovigilancia es determinar, en cada instante, si la actividad dentro del espacio vigilado pone en riesgo la seguridad de dicho espacio. En este sentido, y suponiendo que el sistema, como cualquier otro sistema, no va a ser perfecto, existirán ciertas probabilidades de error que deben ser evaluadas. Para un sistema de videovigilancia se definen las siguientes probabilidades:

  - Probabilidad de no detección: Cuando existe un objeto o situación dentro de la escena que debería ser detectada, y éste no es detectado por el sistema, se habla de no detección, o pérdida. La probabilidad de no detección será, por tanto, la probabilidad de que un objeto que debería ser detectado finalmente no lo sea.
  - Probabilidad de falsa alarma: Al contrario que lo anterior, cuando en la escena no existe ningún objeto extraño o situación anómala, pero debido a causas ajenas, como puede ser el ruido o errores en los algoritmos, el sistema indica una detección, se habla de falsa alarma. Aunque las consecuencias de las falsas alarmas no son tan graves como en el caso anterior, una probabilidad de falsa alarma demasiado alta puede llegar a saturar el sistema, haciendo que éste sea inútil.

- Precio: Si uno de los objetivos finales del diseño de un sistema de videovigilancia es conseguir la mayor fiabilidad posible, otro no menos importante se refiere a la reducción, al máximo posible, del precio total del sistema. La complejidad, o lo que es lo mismo, el precio total, no sólo depende de los equipos y algoritmos de procesado de las imágenes de vídeo empleados, sino también del sistema de captación de imágenes. Se buscará que tanto los sensores de captación de imágenes como los

medios de transmisión empleados sean lo más sencillos y baratos posibles, por lo que los algoritmos empleados deberían poder corregir estos defectos buscando de nuevo la mayor fiabilidad posible.

1.2. Descripción de un sistema de videovigilancia típico

Los sistemas de videovigilancia automáticos basados en técnicas de procesado de imágenes tienen distintas aplicaciones, como pueden ser el control de intrusismo en un recinto privado, el recuento de personas o vehículos que atraviesan una determinada zona, el control de plazas libres en un parking, y muchas otras. Aunque en principio distintas, generalmente son tres las tareas básicas que deben implementar los sistemas de videovigilancia. En primer lugar, la detección de los objetos de interés, lo cual puede incluir el indicar la posición donde se encuentra el objeto dentro de la imagen, la distancia a la que se encuentra del sensor, o incluso el posicionar dicho objeto en el escenario vigilado. Por otro lado, una vez detectado el objeto, en general será necesario seguirlo a lo largo de las sucesivas imágenes de la escena que vayan siendo capturadas, y finalmente, reconocer y clasificar dicho objeto dentro de un grupo de clases previamente definido. En la figura 1.1 se muestra el diagrama de bloques general para un sistema de videovigilancia.

Como es lógico, el primer paso a realizar sería la captación de la imagen, y si fuera necesario, su transmisión hacia el centro de procesado. El problema más importante a resolver en esta etapa es la colocación de las videocámaras en la posición óptima, de tal forma que se eviten zonas ocultas, prestando mayor atención a los puntos de acceso al recinto vigilado. Aunque suele ser una de las partes más complejas durante el diseño y la instalación del sistema, este trabajo no tiene ninguna relación con el procesado de imágenes, y no será tratado en este trabajo. Sin embargo, cuando el sistema de captación es móvil, e incluso la videocámara añade funciones de zoom para acercar o alejar la imagen, si que es necesario añadir al bloque de adquisición de imágenes las funciones que permitan calcular el desplazamiento relativo entre la imagen actual y la anterior. Este proceso suele ser conocido en la literatura como alineamiento de la imagen, o registro de la imagen (*image registration*).

El bloque más importante en casi cualquier sistema de procesado de imágenes en general, y de videovigilancia en particular, es la extracción de las zonas de la imagen más interesantes, lo que es conocido como proceso de segmentación ([Gonzalez93]). La salida de dicho bloque es una máscara binaria, donde se indica qué píxeles de la imagen son de interés, en nuestro caso suele denominarse como primer plano, y cuales no, lo que constituye el fondo. Existen muchas técnicas de segmentación, aunque en el caso de la videovigilancia, la técnica más extendida es la detección de movimiento.

Una vez realizada la segmentación, se dispone de una máscara binaria donde los píxeles correspondientes a los objetos de interés estarán dispuestos como masas de píxeles adyacentes o próximas entre sí. En el bloque de detección de objetos se obtiene una lista

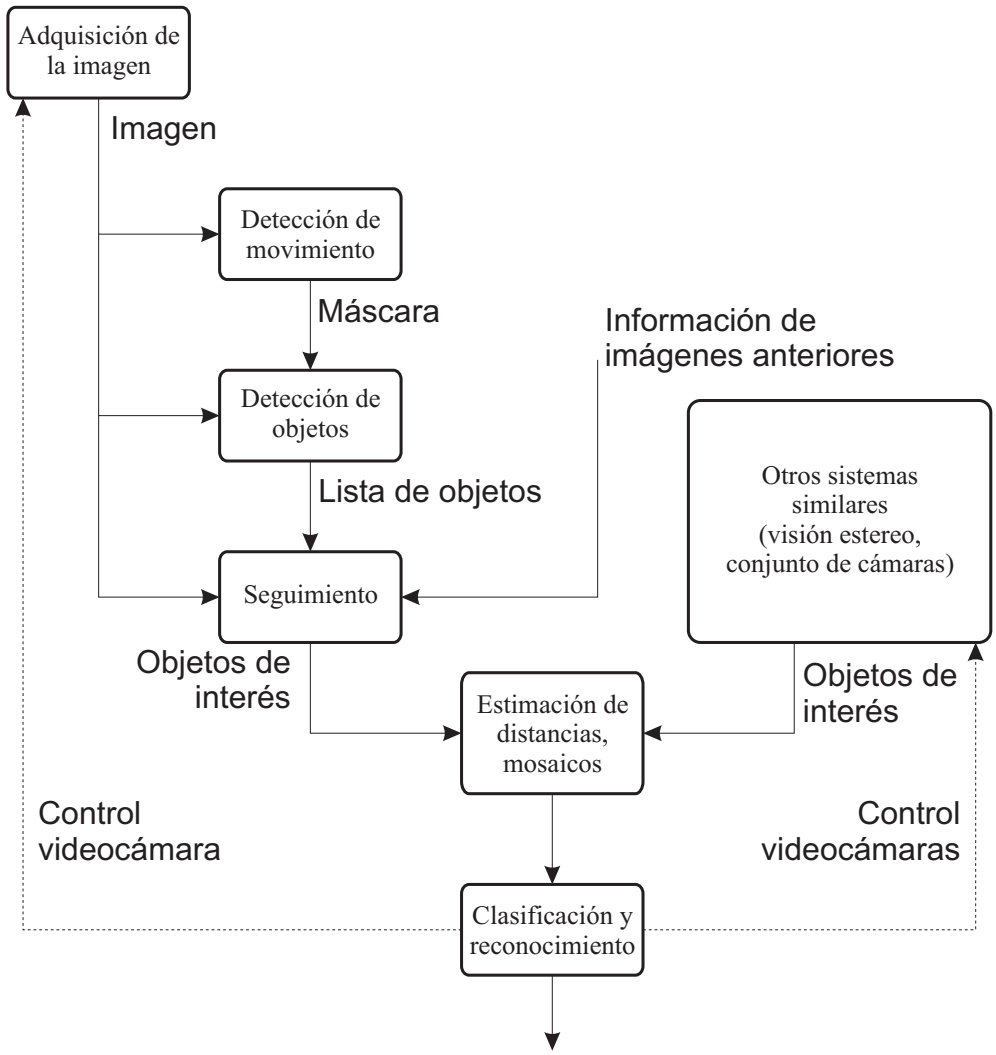


Figura 1.1: Esquema de bloques general de un sistema de videovigilancia

de objetos a partir de grupos de píxeles próximos entre sí. Aunque típicamente se realiza el cálculo de las componentes conectadas, tal y como se describe por ejemplo en [Gonzalez93], otros esquemas también pueden ser implementados. Por ejemplo, en [Boult01] la agrupación se realiza mediante la técnica denominada *Quasi-connected Components*. En este bloque suele realizarse un filtrado de objetos, a partir de las características de estos, tales como área, tamaño, relación de aspecto, etc.

Una vez obtenida la lista de objetos, el siguiente paso suele ser la búsqueda de la correspondencia de cada objeto en la imagen actual con los objetos detectados en imágenes previas. Este proceso se denomina seguimiento, o *tracking*. Este bloque permite, además de tener un conocimiento continuo de la trayectoria que ha seguido el objeto durante el tiempo que fue visible en la imagen, eliminar muchas falsas alarmas, ya que para aquellos objetos detectados en una imagen concreta que no han sido localizados en ningún otro marco, ya sea anterior o posterior, se puede suponer que se trata de una falsa alarma. Además, el conocimiento de la trayectoria completa seguida por el objeto puede ayudar también a determinar si dicho objeto es de interés o no. Por ejemplo, en un recinto, se pueden determinar que ciertas trayectorias están permitidas, mientras que otras no. Dependiendo de la trayectoria seguida por un objeto podemos determinar si se trata de un objeto a tener en cuenta.

Ciertos sistemas de videovigilancia, generalmente cuando el recinto a vigilar es amplio, suelen contar con más de una cámara de vídeo. En estas circunstancias es posible, y en general recomendable, combinar toda la información proporcionada por el conjunto de videocámaras. Cuando las cámaras están colocadas de tal manera que un mismo punto de la escena es captado por más de una cámara desde diferentes puntos de vista, toda la información puede emplearse para estimar la distancia a la que se encuentra el objeto de la cámara, o incluso cuando existe un número suficiente de cámaras, estimar la situación en tres dimensiones del objeto dentro del entorno vigilado. En otras ocasiones, también es posible combinar todas las imágenes recibidas en un único mosaico, para poder presentar una única imagen formada por la combinación de todas las imágenes recibidas, en lugar de un conjunto de imágenes independientes.

Una vez que los objetos han sido detectados, y se dispone de la mayor cantidad posible de información respecto a dicho objeto, el último paso suele ser el reconocimiento y la clasificación del objeto de acuerdo a los patrones previamente establecidos. Dicha clasificación puede ser, por ejemplo, el determinar si existe intrusismo en el recinto vigilado, si se trata de una persona más en aplicaciones que hacen recuentos de personas que entran o salen de un lugar determinado, que existe un vehículo ocupando una plaza determinada de un parking, o cualquier otra clasificación.

La salida del reconocimiento puede ser empleada en ocasiones para realizar el control del sistema de sensores. Por ejemplo, desplazar la cámara hacia la izquierda, derecha, arriba o abajo para seguir un objeto de interés, acercar o alejar el zoom, o activar un sistema de grabación que permita almacenar la imagen o imágenes actuales para su posterior supervisión por personal humano.

Para que el sistema de procesado de vídeo pueda cumplir sus objetivos de detección de eventos extraños dentro de la escena deberá incluir algoritmos de procesado de vídeo y visión artificial diseñados específicamente para dicho fin. Aunque existen numerosas aplicaciones de videovigilancia, las aplicaciones típicas pueden incluir las siguientes funciones:

- **Detección de movimiento:** Ésta suele ser una de las funciones básicas de un sistema típico de videovigilancia cuando la aplicación está destinada a vigilar la actividad dentro de un espacio determinado y el sistema de sensores de captación de imágenes es estático, es decir, está montado sobre estructuras inmóviles, como puede ser una pared o un poste. En este tipo de aplicaciones, la actividad que puede violar las condiciones de seguridad de la zona son los elementos que están en movimiento ([DiStefano01]). Mientras que el fondo, o zonas de no interés para el sistema, es en cierto modo estático, los elementos nuevos dentro de la escena, que serán normalmente los de interés para el sistema, están en movimiento. Con esta simple suposición, el sistema puede determinar en todo momento si la actividad dentro de la escena vigilada es de interés o no.

- **Estimación de distancia:** En cierto tipo de aplicaciones no sólo es necesario determinar la presencia de movimiento o no dentro de la escena vigilada, sino que también puede ser imprescindible estimar la distancia a la que se encuentra dicho objeto del sensor. Al igual que el sistema visual humano y el de muchos otros animales, es posible estimar la distancia al sensor mediante procesado de una imagen estereoscópica captada mediante dos, o en ocasiones más de dos, sensores enfocando a la misma escena con una cierta separación entre ellas (ver [Szelisky94] ó [Benosman98]). Si el sistema dispone de esta información podrá estimar, para todos aquellos objetos detectados mediante la función de detección de movimiento anterior, cuales de ellos son realmente de interés para el sistema, discriminándolos por la distancia a la que estos se encuentran del sensor. Otros sistemas también pueden aprovecharse del desenfoque producido por la lente de la cámara para estimar dicha distancia ([Pentland87]).

- **Cálculo de desplazamiento de cámara:** Existen aplicaciones de videovigilancia en el que las cámaras, aun siendo fijas, éstas pueden girar alrededor de su eje de sujeción impulsadas por un motor o cualquier otro sistema electromecánico, para poder analizar un campo de visión mucho más amplio que el que se analizaría si el sensor estuviera completamente fijo. En estos casos, puede ser necesario que el sistema tenga conocimiento, en todo momento, de la posición exacta de la cámara, o al menos, del desplazamiento relativo entre una imagen y la siguiente (ver [Benosman98]). Aplicaciones más avanzadas pueden incluir funciones de acercamiento / alejamiento mediante manejo del zoom óptico de la cámara. En estos casos, estas funciones deben no sólo evaluar el giro de la cámara respecto a su eje, sino también el grado de ampliación o reducción de la imagen actual.

- **Composición de mosaicos:** Cuando el sistema anterior no es viable, pero sin embargo, sigue siendo necesario el ampliar el campo de visión cubierto por una única cámara, una solución más completa puede ser el proveer al sistema con más de una vídeo cámara. Si todo el conjunto de vídeo cámaras están situadas en la misma posición, con la única diferencia de que cada una de ellas capta una zona distinta del espacio a vigilar, una posible estrategia sería el combinar todas las imágenes para formar una única imagen equivalente a lo que captaría una única cámara con un campo de visión mucho mayor. Una solución alternativa a la anterior sería evaluar la información aportada por cada cámara de forma independiente. En este caso sería necesario, al menos, proveer al sistema de mecanismos para la gestión de traspasos entre cámaras cuando un objeto de interés desaparece del campo de visión de una cámara para aparecer en la cámara adyacente.

- **Seguimiento de objetos:** El objetivo final de las funciones anteriores es la determinación de la presencia o no de elementos extraños en la escena actual. Como un sistema de videovigilancia exige la evaluación en tiempo real y en todo momento de la actividad en la zona evaluada, lo que implica la detección de cada uno de los objetos extraños que existen en cada instante de tiempo, puede hacerse necesario la incorporación de funciones que traten de relacionar cada uno de los objetos detectados en imágenes consecutivas, lo que se denomina seguimiento de objetos, o tracking, como por ejemplo puede verse en [DawsonHowe96]. De esta forma, se puede evaluar no sólo la presencia de objetos, sino también su comportamiento a lo largo del tiempo, lo que supone una mayor cantidad de información para ser procesada por las funciones de niveles superiores.

- **Clasificación de eventos:** En un sistema de videovigilancia manejado por operadores humanos, la misión última de dicho operador es analizar el evento ocurrido y clasificarlo de acuerdo a su experiencia y conocimientos adquiridos. Un sistema de segunda generación deberá por tanto incluir también dichas funciones, y se valdrá de la información aportada por las funciones anteriores. El objetivo de dicha clasificación puede ser muy diverso, desde determinar si el evento es de interés o no, en función de determinados parámetros del objeto detectado, como el tamaño, posición, comportamiento, etc, hasta determinar información adicional del objeto, como puede ser el tipo de objeto de que se trata.

1.3. Organización de la memoria

La enumeración de las distintas funciones que pueden ser incluidas en un sistema de videovigilancia que han sido definidas anteriormente permiten la división del trabajo realizado en esta tesis doctoral en distintos grandes bloques. Aunque un sistema de videovigilancia debería incorporar en mayor o menor medida todas las funciones comentadas

anteriormente, una buena parte de las publicaciones existentes se centran exclusivamente en una u otra función, mientras que otros trabajos si implementan un sistema completo.

Tal y como ha sido explicado anteriormente, existen distintas aplicaciones de videovigilancia, desde la monitorización de una carretera, hasta la detección de intrusismo, por lo que la implementación de un algoritmo concreto va a depender, en gran medida, de cual sea su utilidad final. Esta tesis doctoral se centra en tres de las funciones más importantes de la videovigilancia, que son la detección de movimiento, la estimación de distancias y la clasificación de objetos.

Así, en el capítulo 2 se realiza un estudio de los distintos trabajos existentes que pueden encontrarse en la literatura dentro del campo de la detección de movimiento. Posteriormente, dentro del mismo capítulo se desarrolla el algoritmo de detección de movimiento implementado en esta tesis doctoral.

El capítulo 3 está dedicado a la estimación de distancias. Como en esta tesis desarrollaremos dos métodos distintos de estimación de distancias, es decir, la visión estereoscópica y mediante el desenfoque de la imagen, este capítulo ha quedado dividido en dos grandes bloques, donde se trata cada uno de los dos métodos anteriores.

Así, tras una breve introducción sobre la estimación de distancias mediante visión por computador, en la primera parte se analizan los trabajos más importantes dentro la visión estereoscópica. Así mismo, y dada la gran importancia que presenta para este tipo de sistemas, también se ha incluido un resumen tratando el tema de la calibración de la cámara y de parejas estéreos de cámaras. Por último, se desarrolla el algoritmo de visión estereoscópica propuesto.

En la segunda parte del capítulo 3 se realiza un desarrollo similar para el caso de la estimación de distancias mediante el desenfoque de la imagen. En este caso, se comienza con una introducción que cubre los conceptos de la física óptica que serán utilizados en el resto del capítulo. Posteriormente, se analizan los trabajos más importantes en este campo, y finalmente, se analiza el funcionamiento del algoritmo desarrollado.

En el capítulo 4 nos centraremos en la clasificación de objetos. Así, en la revisión del estado del arte, se comenzará con un repaso de las técnicas de clasificación más importantes, tanto en la clasificación de objetos arbitrarios en general, como en la clasificación de formas geométricas en particular. Será para estas últimas para las que desarrollaremos el algoritmo de clasificación que será descrito en la última parte de este capítulo.

Finalmente, en el capítulo 5 se resumen las principales aportaciones originales de esta tesis, y se presentan las líneas futuras de investigación.

Capítulo 2

Detección de movimiento

La detección de movimiento suele ser una de las funciones fundamentales de cualquier sistema de videovigilancia, y se sitúa en las etapas más tempranas de todo el conjunto de procesado de imagen, de tal forma que, en general, la entrada a este sistema son las imágenes originales captadas por la cámara de vídeo, o en ocasiones, con algún preprocesado previo. La misión fundamental de este bloque es la extracción, desde la imagen completa, de los objetos de primer plano, caracterizados por haber registrado algún tipo de movimiento, que serán probablemente los más interesantes para ser procesados posteriormente por las siguientes etapas del sistema de videovigilancia.

Cuando en la escena vigilada no existen objetos de interés, se puede considerar que la imagen captada por la cámara presenta unas determinadas propiedades, y cuando sobre dicha escena aparece algún objeto de interés, esa zona de la imagen deja de presentar esas propiedades. Podemos establecer, por tanto, que el mecanismo fundamental de los algoritmos de detección de movimiento es la comparación de la imagen actual, o bien cualquier otro tipo de señal obtenida a partir de ésta, con una referencia previamente obtenida.

Los algoritmos de detección de movimiento suelen ser considerados generalmente como una división jerárquica de varios niveles, lo cual facilita su análisis y diseño. Una descomposición clásica considera la división en tres niveles para un sistema de videovigilancia de propósito general, tal y como puede verse en la figura 2.1.

La función principal del primer nivel, el denominado *nivel de píxel*, es la clasificación de cada uno de los píxeles que componen la imagen de acuerdo a dos posibles valores, que son *fondo* o *primer plano*. El objetivo de esta clasificación es el hacer una primera estimación de cuales son las zonas de la imagen a las que habrá que prestar un mayor interés. Para un sistema de videovigilancia, las zonas de interés son los objetos *extraños* en la escena, y en estos casos, la separación suele realizarse generalmente mediante funciones de detección de movimiento. Sin embargo, es necesario indicar aquí que esta primera clasificación no va a ser en general suficientemente precisa, y el refinamiento de la máscara de objetos de interés va a ser realizado por las funciones de nivel superior. No obstante, existen aplicaciones de vigilancia básicas, como puede ser la monitorización de

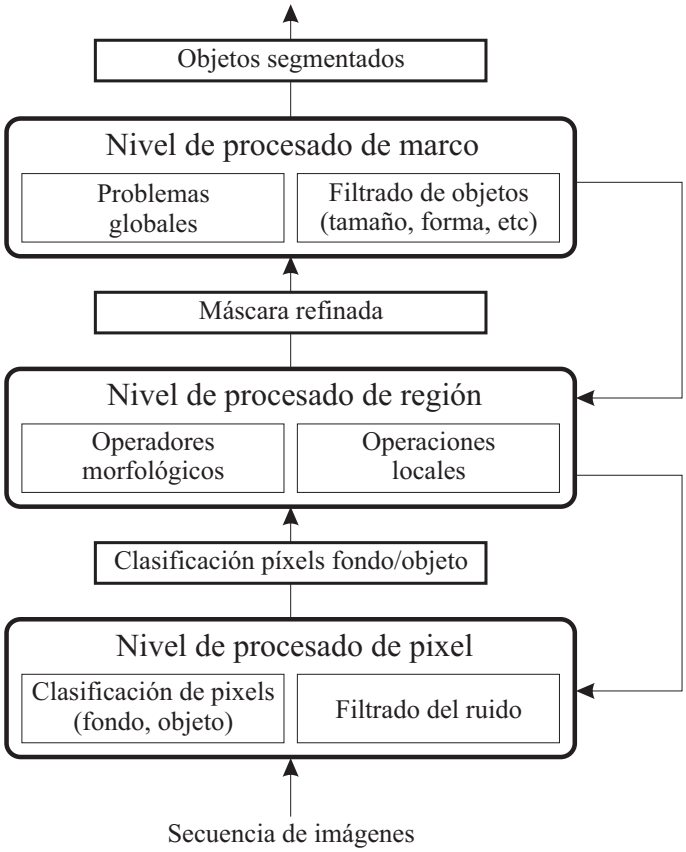


Figura 2.1: Estructura jerárquica de un sistema de videovigilancia de propósito general

una zona completamente estática, típicamente en interiores de edificios, con el objetivo de disparar una alarma cuando cualquier objeto es detectado en la imagen y así avisar al correspondiente operador humano para la evaluación de la secuencia actual. Este tipo de sistemas tan sencillos únicamente emplean funciones de procesado a nivel de píxel, aunque cualquier otra aplicación ligeramente más avanzada va a necesitar de la ayuda de funciones de niveles superiores para mejorar, en la medida de lo posible, la salida de las funciones de nivel de píxel.

Como suele ocurrir en la mayoría de los sistemas de visión artificial, el funcionamiento de las primeras etapas condiciona en gran medida el funcionamiento de todo el sistema. En el caso de un sistema de videovigilancia, el módulo de detección de movimiento, y más concretamente el nivel de procesado de píxel será el que condicione fundamentalmente el rendimiento global del sistema. El procesado a nivel de píxel suele estar bastante influenciado por el ruido u otras situaciones incontrolables, de tal forma que la misión principal de los siguientes niveles es corregir en la medida de lo posible estos inconvenientes.

El segundo nivel, el considerado como nivel de región, es un bloque opcional, y su misión principal es el refinamiento de la máscara de salida proporcionada por el nivel de píxel. Como se ha comentado anteriormente, el nivel de píxel considera cada píxel de

forma independiente, sin tener en cuenta la información observada en los píxeles vecinos. El nivel de región considera dicha información, haciendo uso de funciones de procesado de imagen locales que tienen en cuenta la información de los vecinos, para refinar el resultado final. Así mismo, también puede hacer uso de operadores morfológicos, como son la dilatación, la erosión, la apertura o el cierre para mejorar el aspecto de los contornos de los objetos detectados. También puede incorporarse a este nivel las funciones de detección y eliminación de sombras.

Entre las funciones del nivel de procesado de marco está la eliminación de píxeles ruidosos, así como unir objetos fracturados o separar objetos que quedaron unidos. La función fundamental de este bloque sería el cálculo de las componentes conectadas, para así tener constancia de los objetos presentes en la escena, y poder además realizar un filtrado de objetos, a partir de su tamaño, forma, etc.

En el módulo de procesado a nivel de marco también se intentan eliminar aquellos problemas que afectan a la imagen entera. Un caso típico podría ser un cambio de iluminación en la escena. Ante un cambio de iluminación, el valor de todos los píxeles de la imagen cambia de acuerdo al cambio de iluminación, con lo que el nivel de procesado de píxel marcará prácticamente todos los píxeles de la imagen como objetos de interés, lo que es claramente erróneo. El nivel de procesado de marco debe ser capaz de detectar estas situaciones y evitar el funcionamiento erróneo del sistema. Otros problemas típicos a solucionar en este nivel sería el sabotaje, que consiste en la modificación voluntaria del sistema de sensores, ya sea desenfocando la imagen, ocluyendo la zona de visión o destruyendo la cámara. Aunque en esta situación, el sistema de vigilancia poco podría hacer, si al menos puede disparar la alarma para avisar de que algo anormal está ocurriendo en esa zona.

La salida del sistema de detección de movimiento será un listado de los objetos que forman el primer plano, es decir, la lista de objetos en movimiento. Dependiendo de la aplicación considerada, esta salida puede ser la salida de todo el sistema, o bien, puede convertirse en la entrada a otras etapas del sistema, como puede ser el de seguimiento, o el de reconocimiento de objetos.

El rendimiento global de un sistema de videovigilancia en general, y del módulo de detección de movimiento en particular, va a ser muy dependiente del entorno donde esté trabajando. Podemos considerar desde entornos completamente controlados, como pueden ser por ejemplo zonas interiores con una iluminación constante y sin ningún tipo de movimiento del fondo, hasta entornos con cambios de iluminación, con movimientos impredecibles del fondo, ruido, etc. En este apartado se enumeran los problemas típicos que podemos encontrarnos en la detección de movimiento.

- **Objetos de fondo desplazados:** Los objetos pertenecientes al fondo, y que no son de interés para la aplicación de videovigilancia también pueden ser desplazados, lo que podría provocar la aparición de una falsa alarma. El sistema debería ser capaz

de diferenciar los objetos pertenecientes al fondo respecto de los que no, ya que estos van a ser los únicos de importancia para la aplicación.

- **Hora del día:** Para aplicaciones de videovigilancia en entornos exteriores, y cuya fuente de iluminación principal sea la luz del sol, la iluminación de la escena cambia gradualmente, modificando la apariencia del fondo y provocando la aparición de falsas alarmas. El sistema debería de ser capaz de adaptarse a dichos cambios de iluminación, incluyendo su funcionamiento por la noche, donde la única fuente de iluminación disponible sea la luz artificial.

- **Cambio brusco de iluminación:** Los cambios bruscos de iluminación, como por ejemplo el encendido o apagado de las luces de una habitación, pueden provocar la aparición de falsas alarmas, al menos en las imágenes posteriores al cambio de iluminación.

- **Movimiento continuo del fondo:** Ciertos objetos pertenecientes al fondo, como pueden ser las ramas de un árbol o una superficie llena de agua, pueden tener un movimiento continuo y aleatorio, y provocar la aparición de falsas alarmas. Al ser el movimiento continuo, este efecto provocará la activación de la alarma en todo momento, causando la saturación del sistema.

- **Camuflaje:** Los objetos de interés pueden llegar a tener características similares a los del fondo, como puede ser el color o la textura, dificultando por tanto la detección de dicho objeto.

- **Ausencia de periodo de entrenamiento:** Ciertos algoritmos de videovigilancia requieren de un periodo de entrenamiento previo a su funcionamiento normal para poder realizar una estimación del comportamiento del fondo. En ciertas entornos, no es posible disponer de un periodo de tiempo suficientemente largo y por tanto el sistema debe ser capaz de realizar el entrenamiento correspondiente en presencia de objetos en primer plano.

- **Apertura del primer plano:** Para un objeto de interés coloreado de manera uniforme, a medida que se desplaza, los cambios en los píxeles interiores del objeto no pueden ser detectados. Así, es posible que no sea detectado todo el objeto, sino sólo una parte de él.

- **Parada de un objeto:** Un objeto de primer plano que llega a detenerse puede ser confundido con el fondo. En este caso, si el objeto permanece detenido un tiempo suficientemente grande puede llegar a ser incorporado al modelo de fondo estimado.

- **Arranque de un objeto:** Cuando un objeto del fondo comienza a moverse, dicho objeto y la parte del fondo que quedaba oculta por el objeto y que ahora aparece visible a la cámara aparecerán como dos nuevos objetos del primer plano.

- **Sombras:** Las sombras proyectadas por los objetos del primer plano modifican la apariencia del fondo inicialmente estimado sin la presencia de objetos de primer plano. En este caso, el sistema debería ser capaz de diferenciar entre objetos de primer plano y sus correspondientes sombras. El principal problema provocado por las sombras es el hecho de que pueden llegar a unir objetos que a simple vista están completamente separados.

- **Objetos fracturados:** La salida del nivel de procesado de píxel suele ser aplicada a un módulo de cálculo de componentes conectadas, donde en teoría cada una de las componentes conectadas obtenidas debería pertenecer a un único objeto. Es posible que varias componentes conectadas pertenezcan a un único objeto, habiéndose provocado por tanto la fractura de dicho objeto.

- **Fusión de objetos:** Al contrario que el punto anterior, es posible que dos objetos aparezcan juntos y sean considerados una única componente conectada.

Los algoritmos de detección de movimiento tratan de resolver todos o algunos de estos problemas mediante diferentes técnicas. Los problemas con los que nos vamos a encontrar en una determinada aplicación van a depender de la aplicación en si, y del entorno donde el sistema de vigilancia va a estar operando. Por ejemplo, los problemas de cambios bruscos de iluminación van a producirse principalmente en interiores, ya que en este caso, la cantidad total de luz va a depender fuertemente de la iluminación presente en cada instante en la habitación vigilada. Por contra, los problemas de cambio gradual de la iluminación ocurrirán en exteriores con mayor frecuencia.

Sea cual sea el algoritmo utilizado, varios son los parámetros a evaluar para poder determinar el rendimiento del algoritmo. En primer lugar, y debido a que no existe el algoritmo perfecto, debemos ser capaces de medir los errores cometidos por cada algoritmo. Al igual que al evaluar el sistema completo podemos hablar de falsas alarmas y de pérdidas, en el módulo de detección de movimiento debemos bajar hasta el nivel de píxel, y hablaremos de falsos positivos, que son píxeles que han sido clasificados como de primer plano cuando en realidad pertenecen al fondo, y de falsos negativos, que son los píxeles clasificados como fondo, cuando en realidad pertenecen al primer plano. La evaluación anterior implica el disponer de la información real de a qué grupo pertenece cada píxel. Llamaremos a esta información, la máscara de movimiento real de la imagen, y evidentemente, debe ser generada a mano.

En segundo lugar, otro parámetro a evaluar será la cantidad de recursos del sistema empleados. Si el sistema va a ser implementado sobre un ordenador, será importante evaluar la cantidad de memoria necesaria para la ejecución normal del algoritmo. Este parámetro va a determinar, en cierta medida, el coste total del sistema, ya que una mayor necesidad de memoria implica un mayor coste del equipo que realizará el procesado.

Por último, y especialmente fundamental en los sistemas de videovigilancia, es el tiempo medio necesario para procesar una imagen, ya que no hay que olvidar que uno de los

objetivos principales es el funcionamiento en tiempo real de todo el sistema. Un tiempo excesivamente alto implicará la utilización de equipos más rápidos y costosos para poder cumplir con las exigencias de funcionamiento en tiempo real.

También deberíamos hacer notar aquí que existe una relación entre los falsos positivos y el tiempo de ejecución del sistema completo. Es decir, un algoritmo de detección de movimiento que genere un número elevado de falsos positivos implica que las etapas de procesado posteriores pierdan más tiempo en analizar un número de falsas alarmas que en realidad no deberían haber sido analizadas.

Centrándonos en el nivel de píxel descrito anteriormente, podemos realizar una clasificación general de los algoritmos de detección de movimiento en dos grupos principalmente.

- **Algoritmos derivativos:** La extracción de los píxeles de primer plano se realiza comparando la imagen actual con la (o las) imagen anterior.
- **Algoritmos de fondo:** El sistema realiza una estimación del fondo que será empleada para compararla con la imagen actual.

Antes de comenzar con la descripción de los distintos algoritmos existentes, debemos hacer notar aquí que, en general, los algoritmos de detección de movimiento están diseñados para trabajar tanto con imágenes en blanco y negro como con imágenes en color. Aunque en las secciones siguientes las descripciones de los algoritmos se hacen fundamentalmente para imágenes en niveles de grises, la extensión de los algoritmos a imágenes en color se reduce en la mayoría de los casos a sustituir el escalar correspondiente al nivel de gris del píxel por un vector de tres dimensiones. Cuando así no sea, y la extensión a imágenes en color sea más compleja, será indicado en el caso concreto. En cuanto a la conveniencia de emplear imágenes en niveles de grises o en color, obviamente las imágenes en color van a necesitar mayor tiempo de cómputo y mayores capacidades de almacenamiento, aunque en general, los resultados obtenidos con imágenes en color serán mejores. Por otro lado, existen distintos espacios de color, tales como el *RGB*, *HSI*, *YUV*, etc. En general, y salvo ciertas excepciones, el espacio de color es indiferente, empleándose aquel que resulte más cómodo.

2.1. Algoritmos derivativos

La opción más sencilla suele ser la comparación de la imagen actual con la imagen anterior. De esta forma, la referencia se obtiene simplemente almacenando la imagen previamente procesada. Una vez realizada la detección de movimiento en una imagen, se descarta la referencia anterior y se guarda la imagen actual como imagen de referencia para la siguiente imagen. A veces, en lugar de una única imagen, pueden utilizarse como referencia varias imágenes, pero el funcionamiento de estos algoritmos es similar a aquellos que sólo utilizan una única imagen.

De entre las principales ventajas que presentan estos algoritmos nos encontramos sobre todo su sencillez, lo cual se ve reflejado en un tiempo de cómputo por imagen pequeño. Además, como la imagen actual se compara únicamente con la imagen anterior, presentan gran robustez frente a cambios de iluminación graduales, ya que la referencia, que es la imagen anterior, va modificándose a medida que la iluminación varia a lo largo del tiempo. Cuando se presenta un cambio de iluminación brusco, dicho cambio sólo afectará a los marcos posteriores al cambio.

Sin embargo, estos algoritmos traen consigo bastantes inconvenientes. Por ejemplo, estos algoritmos soy muy sensibles al problema denominado anteriormente como *apertura del primer plano*, ya que al hacer la comparación entre la imagen actual y las anteriores, todos aquellos píxeles que en todas las imágenes correspondan a un objeto de color uniforme aparecerán marcados como píxeles de fondo. Este problema tiene una estrecha relación con la velocidad de los objetos. Así, la detección se vuelve más pobre a medida que la velocidad disminuye. Además, con los algoritmos derivativos un objeto no puede ser detectado cuando cesa su movimiento. De igual forma, si en el fondo existen objetos con movimiento continuo, como pueden ser los árboles, el sistema estaría continuamente marcando alarmas en dicha zona, provocando la consiguiente saturación del sistema.

En general, los algoritmos derivativos no suelen ser empleados en aplicaciones complejas, donde sea necesario la implementación de funciones de nivel superior, como seguimiento o clasificación de eventos, recurriéndose a algoritmos de detección de movimiento más elaborados, como se verá en siguientes secciones. La utilidad principal de los algoritmos derivativos es en aplicaciones donde lo único que se requiere es un mecanismo capaz de disparar una alarma cuando algo *anormal* está ocurriendo en la escena. Dicha alarma puede servir, por ejemplo, para alertar a un operador humano, quien será el que finalmente evalúe lo que está ocurriendo en la escena. Existen ciertas aplicaciones de vigilancia en las que no se realiza la monitorización en tiempo real por parte de personal humano de lo que está sucediendo en la escena, sino que toda la secuencia es almacenada, para ser posteriormente analizada si fuera preciso. En este caso, los algoritmos derivativos se emplean para comenzar la grabación cuando existe movimiento, y finalizarla cuando deja de haberlo, y así ahorrar espacio en el equipo de grabación, evitando grabar imágenes de una escena estática.

Dentro de los algoritmos derivativos, nos encontramos dos grupos. Los de *simple diferencia*, que trabajan únicamente con la imagen actual y la imagen anterior, y los de *doble diferencia*, que trabajan con más de una imagen anterior (lo habitual son la imagen actual y las dos imágenes anteriores).

2.1.1. Algoritmos de simple diferencia

El algoritmo más sencillo para realizar la detección de movimiento que se podría implementar es el denominado como de *simple diferencia*, que consiste en extraer los objetos de primer plano umbralizando la diferencia absoluta entre dos marcos consecutivos. Es

decir, para cada píxel se observa si la diferencia entre dos imágenes consecutivas es suficientemente grande como para considerar que ese píxel corresponde a un objeto del primer plano, de acuerdo a la siguiente expresión:

$$M_t(x,y) = \begin{cases} \text{Objeto si } |I_t(x,y) - I_{t-1}(x,y)| > U \\ \text{Fondo si } |I_t(x,y) - I_{t-1}(x,y)| < U \end{cases} \tag{2.1}$$

donde U es el umbral global para toda la imagen, y que en general será elegido de forma manual. Cuando el umbral es muy grande, objetos con niveles de gris similares a los del fondo donde se sitúa dicho objeto pueden pasar desapercibidos para el sistema, produciéndose lo que hemos denominado como *camuflaje*. Mientras, umbrales muy pequeños pueden hacer que el propio ruido introducido por la cámara provoque la aparición de falsos positivos. Es decir, a medida que aumenta el umbral, se reduce la probabilidad de falsos positivos mientras que aumenta la probabilidad de falsos negativos. El algoritmo así definido se denomina SAD (*Sum of Absolute Differences*), descrito por [Huang99]. Notemos aquí que a partir de ahora, para cualquier función de operación píxel a píxel sobre la imagen, como la definida anteriormente, se omitirá la inclusión de los índices (x, y), dejando claro que dicha función realiza la operación indicada sobre todos los píxeles de la imagen.

Aunque el cálculo de la máscara de movimiento mediante la umbralización de la diferencia de dos imágenes consecutivas es bastante simple, dicho algoritmo ha sido empleado en numerosas aplicaciones. Por ejemplo, en [Kamnoonwatana06], se aplica el algoritmo SAD sobre la componente de luminancia Y de una secuencia en color perteneciente a una transmisión de vídeo en el sistema DVTS (*Digital Video Transfer System*). El resultado de la detección de movimiento permite ajustar la velocidad de transmisión del flujo de vídeo.

Para reducir la sensibilidad del algoritmo con los cambios bruscos de iluminación, el algoritmo MR-SAD (*Mean-Reduced SAD*), implementado por [Huang99] calcula en primer lugar el valor medio de la diferencia entre ambas imágenes, restando posteriormente dicha media a la diferencia, y calculando por último el valor absoluto de la nueva diferencia, igual que en el algoritmo SAD.

En [Yang07] se combina el algoritmo de substracción de imágenes con un algoritmo de partición de imágenes (*image partition*), denominado *watershed*. En este trabajo, la máscara de movimiento obtenida a partir de la diferencia de dos marcos consecutivos es utilizada para determinar si ha existido un movimiento global en la imagen, o no. Si ha existido movimiento global, un algoritmo denominado GME&GMC (*Global Motion Estimation and Global Motion Compensation*) se encarga de estimar, mediante un algoritmo de redución de mínimos cuadrados, el movimiento global de la imagen. Si no ha existido movimiento global, el movimiento se calcula únicamente sobre las particiones en las que ha sido dividida la imagen mediante el algoritmo *watershed*.

Los algoritmos de simple diferencia presentan un nuevo problema que va a poder ser reducido considerablemente con los de doble diferencia. Este problema se denomina

ghosting, y consiste en la aparición de una región del fondo temporalmente oculta por un objeto en movimiento, y que es detectada como un objeto de primer plano cuando el objeto que lo ocultaba deja dicha zona a la vista.

2.1.2. Algoritmos de doble diferencia

Para mejorar ciertos problemas presentes en los algoritmos de simple diferencia, podemos tomar como referencia más de una imagen. Aunque el número de imágenes puede ser cualquiera, en general, el número de imágenes de referencia suele ser de dos, y de aquí surge el nombre de algoritmos de doble diferencia.

Dependiendo de como combinemos la información de las tres imágenes podemos obtener distintos algoritmos. La solución propuesta por [Kameda96] umbraliza la diferencia absoluta entre las imágenes I_t e I_{t-1} y las imágenes I_t e I_{t-2}. La información de estas matrices binarias se combinan realizando la operación lógica AND:

$$M_t = M'_{t,t-1} \text{ AND } M'_{t,t-2} \tag{2.2}$$

donde $M'_{t,t-1}$ es la máscara binaria obtenida de umbralizar la imagen actual y la imagen inmediatamente anterior, y $M'_{t,t-2}$ la correspondiente entre la imagen actual y la antepenúltima imagen. La principal mejora de este algoritmo respecto a los de simple diferencia es la reducción del *ghosting*, eliminando de manera más efectiva la máscara perteneciente a la porción de fondo que el objeto del primer plano deja visible al desplazarse por la escena. Además, este algoritmo incrementa la inmunidad frente al ruido introducido por la cámara ya que dicho ruido va a provocar la aparición de falsos positivos distribuidos de forma aleatoria por toda la imagen. En general, si consideramos que el ruido es gaussiano, será muy difícil que dos falsos positivos aparezcan en la misma posición, y por tanto, la operación AND sobre estos píxeles va a provocar la desaparición de la gran mayoría de falsos positivos.

Como inconveniente principal, podemos indicar su mayor sensibilidad ante el fenómeno de la apertura del primer plano, ya que en este caso, se combina el efecto de las dos simples diferencias, siendo por tanto el número de falsos negativos mayor en este caso que en el caso de la simple diferencia.

Otras posibles soluciones para el caso de la doble diferencia dan lugar a algoritmos a medio camino entre los algoritmos derivativos y los de fondo. Se pueden utilizar las tres imágenes disponibles para hacer una estimación del fondo, calculando la media para cada píxel y posteriormente umbralizar la diferencia entre dicha estimación y la imagen actual:

$$M_t = \mathrm{U}\left(\frac{I_t + I_{t-1} + I_{t-2}}{3} - I_t\right) \tag{2.3}$$

donde $\mathrm{U}()$ es la función umbralización, tal y como fue definida en 2.1. Otra posible implementación sería realizando el cálculo de la mediana de cada una de las tres imágenes en lugar de la media.

En [Chiu08], se realizan distintas combinaciones con los tres últimos marcos para obtener la máscara de movimiento. En un primer paso, se calcula la diferencia d_{n-1} entre la imagen $n-1$ y n, y la diferencia d_{n+1} entre la imagen n y la $n+1$. Además, se calcula una última diferencia d_{B} entre la imagen n y un fondo calculado previamente. En un segundo paso, las diferencias d_{n-1} y d_{B} son combinadas mediante una operación AND, para obtener la máscara de movimiento M_{n-1}. La misma operación se realiza con las diferencias d_{B} y d_{n+1} para obtener la máscara M_{n+1}. Una última operación OR combina ambas máscaras para obtener la máscara final de movimiento.

2.2. Algoritmos de fondo

Una opción más elaborada y que suele presentar mejores resultados cuando el sistema trabaja en condiciones adversas, como puede ser imágenes con ruido o con movimiento de objetos del fondo, consiste en generar una estimación del fondo a partir de las imágenes anteriores. De esta forma, el sistema puede aprender el comportamiento de la escena en condiciones normales, y determinar la presencia de objetos de interés mediante decisiones más complejas que las que se podrían realizar con los derivativos. A estas técnicas se las denominan como algoritmos de fondo.

Dentro de los algoritmos de fondo, es posible establecer una nueva clasificación a partir de qué es lo que constituye el fondo que se utilizará como referencia para realizar la detección de movimiento. El caso más sencillo es que el fondo lo constituya una imagen generada a partir de las imágenes anteriores, utilizando algún método de filtrado temporal, o cualquier otro tipo de procesado de señales lineal o no lineal. La referencia es, por tanto, una única imagen, de las mismas dimensiones que la imagen original, y que se asemeja, en gran medida, al fondo captado por la cámara cuando sobre dicho fondo no existen objetos de primer plano. Llamaremos a estos sistemas como *sistemas de módulo de imagen de referencia.*

Sin embargo, ya que disponemos de un histórico del valor que ha tomado cada uno de los píxeles de la imagen en un determinado intervalo de tiempo, es posible obtener una serie de estadísticos, u otra información acerca del comportamiento de cada píxel. En este caso, la referencia del fondo no es una imagen en sí, sino cierta información sobre el comportamiento de cada uno de los píxeles que forman la imagen. Estos sistemas suelen denominarse como *sistemas de módulo estadístico.*

Independientemente de qué constituye el fondo que se utilizará como referencia, estos algoritmos van a estar necesariamente compuestos por dos módulos:

- **Módulo de mantenimiento del fondo:** Es el encargado de generar la referencia del fondo empleando las imágenes iniciales y de actualizar constantemente dicha referencia a partir de las nuevas imágenes que van llegando al sistema. Gracias a esta actualización, el sistema es capaz de ir adaptándose a los cambios dinámicos

que puedan darse en la escena (cambios graduales de iluminación, nuevos objetos de fondo, etc).

- **Módulo de detección:** Es el encargado de realizar la clasificación de los píxeles de la imagen entre fondo y primer plano. Es decir, su función es similar a la de los algoritmos derivativos, con la diferencia de que en esta ocasión, la cantidad de información de la referencia puede ser mucho mayor, y por tanto, las decisiones a la hora de realizar la clasificación pueden ser mucho más complejas.

La mayoría de estos algoritmos necesitan de un periodo de entrenamiento para poder crear la imagen de referencia u obtener los estadísticos que rijan el funcionamiento de cada píxel de la imagen. Sería interesante que este entrenamiento pudiera realizarse a partir de una serie de marcos en los que no aparezcan objetos del primer plano, y por tanto, los cambios en el fondo sólo se deberán a las fuentes de ruido (ruido térmico, cambios de iluminación, movimiento continuo del fondo, etc).

Sin embargo, en algunas situaciones, como pueden ser las áreas públicas, es difícil o prácticamente imposible controlar el área que va a ser monitorizada y, por tanto, la situación ideal en la que la secuencia de entrenamiento está libre de objetos de primer plano no va a ser posible. Ante esta situación, ciertos algoritmos permiten generar la referencia de fondo en presencia de objetos de primer plano. Para que esto sea posible, es necesario por lo menos que cada píxel de la imagen muestre el fondo durante al menos un intervalo de tiempo determinado en la secuencia.

2.2.1. Métodos de imagen de referencia

Dentro de los trabajos previos basados en los algoritmos de fondo, aquellos que integran métodos de imagen de referencia probablemente sean los más sencillos de implementar. En este caso, la detección de movimiento se basa en la comparación de la imagen actual con la imagen de referencia. Así pues, la cuestión principal que plantean este tipo de algoritmos es la generación y mantenimiento de una referencia del fondo adecuada, que sea capaz de seguir las variaciones graduales de la iluminación de la escena, y que evite la incorporación de objetos del primer plano en la referencia.

Módulo de mantenimiento del fondo

El módulo de mantenimiento del fondo será el encargado de generar y actualizar la referencia del fondo de forma adecuada para que el módulo de detección de movimiento pueda adaptarse a los posibles cambios dinámicos en la escena. En general, la imagen de referencia es construida a partir de una combinación, ya sea lineal o no, de los últimos marcos procesados por el sistema, donde el número de marcos considerados va a ser un parámetro del sistema. En [Gao00] se establecen dos aproximaciones distintas para mantener y actualizar la referencia del fondo, los denominados *procesamiento multi-muestra* y *procesamiento por marco*.

En el primer caso, cada vez que sea necesario actualizar el valor de un píxel de la referencia, ese nuevo valor se calcula a partir de las últimas N muestras, descartando el valor anterior de la referencia. Dentro de los algoritmos de procesado lineal multi-muestra, el más extendido es el filtrado temporal. Las técnicas de actualización de la imagen de referencia deben conseguir reflejar las variaciones suaves de la imagen, principalmente los cambios de iluminación graduales, y evitar en la medida de lo posible la incorporación de los objetos de primer plano a la referencia. Analizando las variaciones temporales que sufre cada píxel de la imagen, las variaciones graduales de iluminación se traducen en una señal de baja frecuencia, mientras que la aparición de un objeto de primer plano implica la aparición de altas frecuencias en el espectro de dicha señal. Por tanto, un filtro paso bajo sería capaz de seguir los cambios graduales de iluminación, mientras que podría rechazar los objetos de primer plano, siempre y cuando estos no permanezcan durante mucho tiempo en la misma posición, ya que de otra forma, estos finalmente serían incorporados al modelo de fondo.

Los métodos más sencillos basados en la teoría de filtrado que podemos encontrar son aquellos que calculan la media para cada píxel de la imagen, considerando las N últimas muestras para cada píxel:

$$F_t = \frac{1}{N} \sum_{i=1}^{N} I_{t-i} \tag{2.4}$$

Si en lugar de ponderar por el mismo valor $(1/N)$ todas las muestras del píxel, ponderamos por valores distintos cada muestra tendríamos un filtro FIR genérico, que puede ser diseñado con una respuesta en frecuencia determinada:

$$F_t = \sum_{i=1}^{N} \alpha_i I_{t-i} \tag{2.5}$$

donde α_i son las muestras de la respuesta al impulso del filtro FIR considerado. Empleando las técnicas convencionales de diseño de filtros FIR, podremos diseñar aquel que se adapte a las necesidades, teniendo en cuenta que, en general, el filtro debe tener una respuesta paso bajo para cumplir los objetivos comentados anteriormente. En cualquiera de las dos aproximaciones anteriores, es necesario observar que siempre será necesario que el sistema mantenga almacenadas en todo momento las últimas N imágenes, lo cual redundará en un aumento de la cantidad total de memoria necesaria para el sistema. Por este motivo, esta solución no ha sido ampliamente adoptada.

Siendo ésta la solución más natural, la solución más comúnmente implementada es la de la actualización mediante procesado por marco. En este modelo, para el mismo caso que arriba, la actualización de un píxel determinado se realizaría de acuerdo a [Koller94, Ridder95]:

$$F_t = (1 - \alpha)\, F_{t-1} + \alpha I_t \tag{2.6}$$

Puede observarse en este caso que ya no es necesario almacenar ninguna imagen anterior, pues las únicas muestras que entran en juego son la de la imagen actual y la referencia del fondo. Además, también puede comprobarse que el número de operaciones es mucho menor. Estas dos ventajas, es decir, menor necesidad de almacenamiento y menor coste computacional, hacen que la actualización mediante procesado por marco sea la opción más comúnmente implementada.

En la práctica, la generación y actualización de la referencia va a depender de cada algoritmo particular, pero conceptualmente podemos distinguir dos métodos de actualización, según describe [DiStefano01]:

- **Actualización ciega:** En este caso no se tiene en cuenta la salida del módulo de detección de movimiento, es decir, la clasificación de los píxeles en fondo u objeto. Esto implica que todos los píxeles son actualizados de la misma forma, y no existe por tanto, realimentación desde el módulo de detección al módulo de actualización del fondo. El principal inconveniente de este método es que los objetos de primer plano que se desplazan lentamente por la imagen pueden llegar a ser incluidos erróneamente dentro del modelo del fondo. Esto va a depender lógicamente de la velocidad con la que el objeto se desplaza a lo largo de la imagen, y de la rapidez con la que el módulo de actualización se adapte a los cambios de la escena.

- **Actualización selectiva:** Está basado en una realimentación desde el módulo de detección de movimiento al módulo de actualización de fondo, de tal manera que la adaptación sólo se realiza en los píxeles que fueron clasificados como píxeles de fondo, mientras que los píxeles pertenecientes al primer plano no son actualizados. El principal problema de este método es que una decisión incorrecta, principalmente en los falsos positivos (píxeles de fondo clasificados como primer plano) puede desencadenar en una decisión incorrecta permanente (denominada *abrazo mortal*, o *deadlock*). Es decir, si en una determinada zona de la imagen la estimación del fondo es incorrecta, ésta será incorrecta siempre, ya que el fondo no puede ser actualizado en ninguna ocasión, y por tanto, será siempre erróneo.

Una solución intermedia entre ambos extremos sería la implementación de un sistema de actualización de dos pesos. Teniendo en cuenta que el valor del peso va a indicar la velocidad con la que un píxel se adapta a los cambios de la escena, se fijan pesos de valor alto, es decir velocidad de adaptación alta, para los píxeles de fondo, y pesos de valor bajo, es decir, velocidad de adaptación baja, para los píxeles del primer plano.

Por ejemplo, si la actualización del fondo se hace mediante procesado por marco, como en (2.6), el valor de α es un número entre 0 y 1, y va a establecer la influencia relativa del marco actual y del fondo antiguo en la determinación de la nueva referencia del fondo. Si α se aproxima a 1, y por tanto $(1 - \alpha)$ se aproxima a 0, la influencia de la imagen actual va a ser mucho mayor que la de la referencia, y por tanto, la nueva referencia del fondo se va a asemejar más a la nueva imagen que a la referencia antigua. Esto implica que la

velocidad de adaptación a los cambios de la escena es rápida. Por contra, si α tiende a 0 ocurrirá el caso contrario, es decir, la influencia de la imagen actual es muy pequeña, y la nueva referencia se asemejará más a la referencia antigua que a la imagen actual. Por tanto, para evitar los problemas que se presentan con la *actualización ciega* o con la *actualización selectiva*, podemos establecer valores distintos de α para cada píxel de la imagen, de acuerdo a la clasificación fondo - objeto realizado por el módulo de detección:

$$\alpha(x,y) = \begin{cases} \alpha_1 \text{ si } M(x,y) \in \text{ Fondo} \\ \alpha_2 \text{ si } M(x,y) \in \text{ Objeto} \end{cases} \tag{2.7}$$

Como la velocidad de actualización va a ser mayor cuanto mayor sea el valor de α, está claro que $\alpha_1 > \alpha_2$ para conseguir así que los píxeles de fondo sigan las variaciones de la escena, mientras que los píxeles del primer plano no lleguen a incorporarse al modelo del fondo. No obstante, independiente del valor de α_2, si un objeto del primer plano se detiene completamente y permanece detenido durante un periodo suficiente de tiempo, éste será finalmente incorporado al modelo del fondo, lo cual puede ser un inconveniente o no, dependiendo del tipo de aplicación en cuestión.

El proceso de actualización de dos pesos descrito hasta aquí sería el correspondiente a aquellos algoritmos que implementan la actualización con procesamiento por marco. Para aquellos algoritmos con procesamiento multi-muestra, el proceso sería ligeramente distinto. En estos algoritmos, el valor de un píxel determinado es calculado teniendo en cuenta un cierto número de muestras pasadas. Cuanto menor sea este número de muestras, mayor será el impacto de la nueva muestra en los parámetros calculados, y por tanto mayor la velocidad del proceso de actualización. Así pues, la velocidad de actualización viene definida por el número de muestras anteriores utilizadas para actualizar el fondo.

Por tanto, para el caso de actualización ciega, siempre se utilizarán el mismo número de muestras de cada píxel para actualizar los estadísticos de dicho píxel. Con la actualización selectiva, la diferencia será que únicamente se actualizarán los estadísticos de los píxeles clasificados como fondo por el módulo de detección. Ambas estrategias sufrirán los mismos inconvenientes que para los algoritmos con procesamiento por marco.

Así pues, para evitar los inconvenientes anteriores, una estrategia similar a la actualización con dos pesos puede ser implementada. Por tanto, si el píxel en cuestión ha sido clasificado como fondo, el número de muestras empleadas para recalcular los estadísticos será pequeña, para que así los nuevos valores obtenidos estén muy influenciados por la nueva muestra, es decir, para conseguir una actualización rápida de la referencia. Sin embargo, si el píxel pertenece al primer plano, el número de muestras deberá ser grande, para que la influencia de la nueva muestra no sea muy grande, y así evitar que el objeto del primer plano sea incorporado al modelo del fondo rápidamente.

Una estrategia ligeramente distinta puede ser seguida para intentar evitar los problemas anteriores. Siguiendo el método de actualización selectiva, para todos los píxeles de la imagen se establece un contador individual, que se utiliza para contar las veces que un píxel determinado ha sido marcado como objeto de primer plano. Para evitar las situa-

ciones de *abrazo mortal* en los píxeles erróneos del primer plano, cuando un píxel ha sido marcado como primer plano un número determinado de veces, éste pasará ya a formar parte de la referencia del fondo para su actualización, de acuerdo con cada algoritmo de actualización concreto.

La actualización de la imagen de referencia también puede ser realizada mediante un sistema no lineal. En [DawsonHowe96], la imagen de referencia se obtiene de acuerdo a la siguiente expresión:

$$F_t = \begin{cases} I_0 & t \leq 2 \\ I_t & I_t \neq F_t \text{ y } I_t \approx I_{t-1} \approx I_{t-2} \\ F_{t-1} & \text{resto} \end{cases} \tag{2.8}$$

Es decir, al arrancar la aplicación ($t \leq 2$), el fondo lo constituye la primera imagen (I_0), y un píxel determinado es actualizado con el valor en la imagen actual si éste pertenece al primer plano ($I_t \neq F_t$), y ha tomado un valor similar en los dos últimos marcos ($I_t \approx I_{t-1} \approx I_{t-2}$). En cualquier otro caso, el valor de dicho píxel no se actualiza.

Uno de los filtros no lineales más conocidos es el filtro de mediana, y se utiliza por ejemplo en [Rosin95] para obtener la imagen de referencia. Los filtros de mediana pueden ser más robustos frente al ruido no gaussiano. En nuestro caso, si consideramos un objeto de primer plano que oculta brevemente el fondo, la muestra correspondiente al objeto de primer plano generalmente no se corresponderá con la muestra central de la secuencia ordenada, es decir, la mediana, sino más bien con las muestras de los extremos. De esta forma, si el número de veces que el píxel en cuestión toma los valores del objeto de primer plano no son muchas, dicho objeto nunca será añadido al modelo de fondo. De esta forma, con este tipo de filtros, no es necesario implementar el método de actualización selectiva, sino que sería suficiente con el método de actualización ciega. No obstante, si el objeto de primer plano aparece un número suficiente de veces consecutivas en la misma zona de la imagen, o bien llega a detenerse por completo, finalmente el objeto sería incorporado al modelo del fondo. El principal problema de este método es que para poder calcular la mediana de un conjunto de muestras es necesario disponer de todas esas muestras. Esto se traduce en nuestro caso en que es necesario mantener almacenadas las últimas imágenes previas, de acuerdo al modo de funcionamiento multi-muestra definido anteriormente.

Módulo de detección

En general, la complejidad principal de los algoritmos de detección de movimiento que emplean algoritmos de fondo suele residir en el módulo de mantenimiento del fondo, siendo el módulo de detección muy similar para todos los algoritmos, y además, también muy similar al empleado con los algoritmos derivativos.

Si en los algoritmos derivativos de simple diferencia la detección de movimiento se realizaba umbralizando el valor absoluto de la diferencia entre la imagen actual y la imagen anterior, en los algoritmos de fondo con módulo de imagen de referencia, el proceso es

exactamente el mismo. La única diferencia es, por tanto, que mientras en los algoritmos derivativos la referencia la constituía la imagen anterior, en los algoritmos de fondo con imagen de referencia, la referencia es una imagen estimada a partir de las imágenes previas.

Tanto en los algoritmos derivativos como en los algoritmos con imagen de referencia, la elección del umbral es uno de los puntos claves del sistema. Aunque la mayoría de los sistemas emplean un umbral fijo, determinado a priori por el usuario en función de las características del sistema, y la relación deseada entre la probabilidad de falsos positivos y falsos negativos, algunos sistemas disponen de un umbral dinámico que se actualiza continuamente. En [Rosin95] se propone el cálculo de un umbral global para cada nueva imagen, de acuerdo a las siguientes expresiones:

$$MED_t = \text{med}\, D_t \tag{2.9}$$

$$MAD_t = \text{med}\, |D_t - MED_t| \tag{2.10}$$

donde:

$$D_t = |I_t - F_t| \tag{2.11}$$

y la función *med* indica el cálculo de la mediana de todos los coeficientes de la matriz. Suponiendo que en una escena normal, menos de la mitad de los píxeles estarán en movimiento, entonces el valor de la mediana deberá corresponderse con los valores típicos debidos al ruido introducido por el sensor. A partir de aquí, el umbral es calculado de acuerdo a:

$$U = MED + 3 \times 1{,}4826 \times MAD \tag{2.12}$$

donde el valor 1,4826 es un factor de normalización para una distribución gaussiana. De esta forma, el umbral se ajusta automáticamente a las características de cada sistema, principalmente el ruido gaussiano introducido por la cámara de vídeo, y que incluso puede variar a lo largo del tiempo, según varía, entre otras cosas, la iluminación del sistema.

Aunque la umbralización del valor absoluto de la diferencia entre la imagen actual y la imagen de referencia (ya sea la imagen de referencia la imagen anterior, o bien una imagen obtenida a partir de una combinación de las últimas imágenes) es la técnica más comúnmente empleada con los algoritmos derivativos, ésta no es la única solución propuesta para los algoritmos de imagen de referencia. Cuando los cambios de iluminación locales pueden llegar a ser un grave problema para la detección, es posible basarse en la detección de cambios de texturas de zonas de la imagen para realizar la detección de movimiento. En [DiStefano99] o [DawsonHowe96], la umbralización no se realiza sobre el valor absoluto de la diferencia, sino mediante la función de correlación cruzada normalizada (NCCF):

$$NCCF = \frac{\sum_{x,y} \left(I_t \left(x, y \right) \cdot F_t \left(x, y \right) \right)}{\sqrt{\left(\sum_{x,y} I_t^2 \left(x, y \right) \right) \cdot \left(\sum_{x,y} F_t^2 \left(x, y \right) \right)}} \tag{2.13}$$

donde los sumatorios en x e y se extienden en una ventana cuadrada de tamaño N centrada en cada uno de los píxel de la imagen. Como puede verse en la expresión anterior, cuanto mayor sea el valor de N, mayor será el número de operaciones a realizar, y por tanto, mayor el tiempo de cómputo. La ventaja principal de la función de correlación cruzada respecto al valor absoluto de la diferencia es que el resultado es más robusto ante los cambios de iluminación. Si consideramos todos los píxeles incluidos en la ventana de tamaño N como un vector en un hiperespacio de dimensión N^2, podemos comprobar que el valor de la *NCCF* tenderá a 1 cuando el ángulo entre ambos vectores sea pequeño, y tenderá a 0 cuando el ángulo sea elevado. Suponiendo que un cambio de iluminación en una zona de la imagen supone un cambio en el módulo del vector anterior, pero no en su ángulo, queda claro que ante un cambio de iluminación sin movimiento el valor de la *NCCF* será próximo a 1, y por tanto, no se habrá detectado movimiento. Sin embargo, en una zona de la imagen donde sí exista movimiento, de tal forma que un objeto del primer plano oculte el fondo, el ángulo entre ambos vectores (el de la imagen de referencia y el de la imagen actual) será en general muy distinto, por lo que el valor de la *NCCF* en dicha zona será próximo a 0, y por tanto, se habrá detectado movimiento.

Aunque la suposición anterior no es completamente correcta, la función de correlación cruzada normalizada se comporta, en general, de manera más robusta ante cambios de iluminación. Es decir, podemos suponer que dicha función es sensible a cambios de textura en lugar de cambios de niveles de intensidad de los píxeles. Sin embargo, si el objeto del primer plano tuviera la misma textura que el fondo que oculta (típicamente cuando ambos no tienen información de textura, es decir, cuando ambos son completamente lisos), la detección de cambio de la textura no sería posible, y en este caso la diferencia absoluta funcionaría mejor que la *NCCF*. Para evitar este problema en la medida de lo posible, sería fundamental que el tamaño de la ventana (N) sea suficientemente grande.

Si bien la expresión anterior puede ser aplicada a cada uno de los píxeles de la imagen, esto podría suponer un tiempo de cómputo bastante alto. La reducción del tiempo de computo puede conseguirse dividiendo la imagen en bloques y aplicando la expresión de la *NCCF* a cada uno de los bloques, y la clasificación en fondo - primer plano se realiza para cada bloque, y no para cada píxel. Así, podemos dividir esta técnica de detección de movimiento en dos grupos:

- **NCCF a nivel de píxel:** En este caso, la *NCCF* se calcula para cada píxel, de tal forma que la ventana de correlación se va desplazando a lo largo de toda la imagen, tal y como propone [DiStefano99].

- **NCCF a nivel de bloque:** En este caso, la imagen se divide en pequeños bloques, y la *NCCF* se calcula para cada uno de esos bloques, implementado por [DawsonHowe96].

Claramente, el algoritmo a nivel de píxel conduce a una resolución mucho mayor en la detección de regiones de primer plano que el algoritmo a nivel de bloque. Sin embargo, el primero será bastante más costoso computacionalmente que el segundo.

Una última alternativa para realizar la detección de movimiento mediante algoritmos de imagen de referencia, y siempre y cuando el sensor de captación de imágenes devuelva imágenes en color, sería el emplear distancias de color. En [Bing01] el flujo de vídeo en color se presenta en el espacio de color YUV, y un píxel de la imagen actual se considerará que pertenece al primer plano si se cumplen las siguientes condiciones:

$$|Y_{\mathrm{I}} - Y_{\mathrm{F}}| > T_{\mathrm{Y}} \tag{2.14}$$

$$(U_{\mathrm{I}} - U_{\mathrm{F}})^2 + (V_{\mathrm{I}} - V_{\mathrm{F}})^2 > T_{\mathrm{UV}} \tag{2.15}$$

donde el subíndice I indica el valor correspondiente en la imagen actual, y el subíndice F el valor en la imagen de referencia, e Y, U y V representan la luminancia y las dos componentes de crominancia del espacio de color YUV. Por último, T_{Y} corresponde al umbral global para la luminancia, y T_{UV} el umbral global para la crominancia. Como puede deducirse de la expresión anterior, un píxel de la imagen será clasificado como píxel del primer plano si el cambio en su nivel de gris es suficientemente grande ($> T_{\mathrm{Y}}$) y también lo es su cambio en su información de color ($> T_{\mathrm{UV}}$). Esto hace el sistema más robusto ante cambios de iluminación, ya que, aunque ante dichos cambios el nivel de luminancia puede cambiar notablemente, teóricamente los valores de las componentes de crominancia no van a verse afectadas. Esto último hará que, ante un cambio de iluminación, o bien ante sombras proyectadas por los objetos, los píxeles afectados no sean marcados como objetos de primer plano.

2.2.2. Métodos estadísticos

Si bien todos los métodos descritos hasta el momento, tanto los denominados derivativos, como los de fondo con imagen de referencia, han sido ampliamente utilizados, todos ellos presentan el mismo problema. A la hora de realizar la umbralización, ya sea del valor absoluto de la diferencia, o de cualquier otra medida definida anteriormente, entre la imagen actual y la imagen de referencia, el umbral empleado será siempre global, ya sea éste fijo, o bien calculado dinámicamente. Sin embargo, en una secuencia de vídeo el ruido presente en cada parte de la imagen puede ser distinto. En general, aquellas partes de la imagen que pertenezcan a objetos fijos, como puede ser una pared, las variaciones del nivel de gris de los píxeles en esa zona de la imagen básicamente son debidas al ruido introducido por el sensor. Sin embargo, en zonas donde existan objetos móviles, como

pueden ser árboles o zonas de agua, las variaciones serán principalmente debidas a este movimiento, y serán, en general, mucho mayores que las introducidas por el sensor. Este razonamiento nos sugiere que el umbral empleado en la detección de movimiento no debería ser constante en toda la imagen, sino más bien dependiente del nivel de variación de cada píxel de la imagen.

En general, podemos decir que un sistema con módulo estadístico dispone de mayor cantidad de información del fondo que los de imagen de referencia, que sólo disponen de la imagen estimada del fondo. Esto permite diseñar el módulo de detección de movimiento más complejo que el que se podría si sólo se dispone de una imagen de referencia. Podemos decir entonces que los sistemas con módulo estadístico van a funcionar, en general, mejor que los sistemas con imagen de referencia y los sistemas derivativos.

El inconveniente principal de estos sistemas es que van a necesitar un mayor tiempo de cómputo para obtener todos los estadísticos necesarios que si sólo hubiera que estimar la imagen del fondo. Además, como también es necesario almacenar dichos estadísticos, las necesidades de almacenamiento serán mayores que las necesarias para los sistemas de imagen de referencia, o los derivativos. Por otro lado, para poder calcular los estadísticos de forma precisa sería necesario que la escena estuviera libre de objetos de primer plano, al menos durante los primeros instantes de funcionamiento del sistema. Si esto no fuera posible, principalmente en zonas públicas, donde es casi inevitable que existan en todo momento objetos de primer plano en movimiento, el diseño del sistema debe tener esto en cuenta.

Cualquier sistema con módulo de detección estadístico va a modelar el fondo como un proceso de un conjunto de variables aleatorias con una función de densidad de probabilidad asociada. A la hora de representar dicha función de densidad, tenemos dos posibilidades:

- **Representación paramétrica:** En este caso, se emplea una distribución estadística específica, que asumimos se aproxima a la distribución real. Suponiendo esto, lo que habrá que estimar son los parámetros de dicha distribución, a partir de las muestras de la secuencia de imágenes.

- **Representación no paramétrica:** En este caso, estimamos la función de densidad de probabilidad directamente a partir de las muestras de la secuencia de entrada, sin hacer ninguna suposición sobre la distribución subyacente. Este método evita tener que elegir un modelo y calcular sus parámetros característicos.

Dentro de los métodos estadísticos paramétricos, también es posible el determinar cuantas distribuciones se asumen para modelar el comportamiento de un píxel. Así, tendríamos:

- **Distribuciones unimodales:** Son aquellas que asumen una única distribución por píxel. Son más propensas a crear zonas de baja sensibilidad cuando el nivel de un píxel varía significativamente.

- **Distribuciones multimodales:** En este caso, se asume que la distribución de probabilidad de cada píxel debe ser modelada por más de una distribución. Estas distribuciones son aconsejables cuando el valor de un píxel puede variar significativamente en el tiempo.

Métodos paramétricos unimodales

Dentro de las distribuciones unimodales, la tendencia habitual es elegir una distribución cuyas características incluyan una medida de la tendencia central, como puede ser la media, y una medida de la dispersión para cada píxel. Aunque existen distintas distribuciones, la distribución preferida es la distribución gaussiana, la cual se basa en la suposición de que las variaciones ruidosas de intensidad en cada píxel de la imagen siguen una distribución gaussiana de media cero [Fujiyoshi98], es decir, la función de densidad de probabilidad para cada píxel de la imagen estaría descrita por:

$$\eta\left(x,\mu,\sigma\right)=\frac{1}{\sqrt{2\pi\sigma^2}}\mathrm{e}^{\frac{(x-\mu)^2}{2\sigma^2}} \tag{2.16}$$

siendo μ la media y σ^2 la varianza de la distribución. Por tanto, con este método calculamos, para cada píxel de la imagen, su media, como medida de la tendencia central, y la desviación típica, como medida de dispersión. Una vez estimados los parámetros de la distribución, un píxel será clasificado como perteneciente al primer plano si su valor cae fuera de un intervalo centrado en la media, y anchura proporcional a la desviación típica estimada para dicho píxel. Es decir, la clasificación entre fondo - objeto se realiza de acuerdo a:

$$M\left(x,y\right)=\begin{cases}\text{Obj} & \text{si} \quad I\left(x,y\right)\notin\left[\mu\left(x,y\right)-k\sigma\left(x,y\right);\mu\left(x,y\right)+k\sigma\left(x,y\right)\right]\\ \text{Fnd} & \text{resto}\end{cases} \tag{2.17}$$

donde los índices x, y han sido añadidos para hacer notar que la media y la desviación típica son estimados para cada píxel independientemente. La estimación de dichos parámetros puede ser realizada de distintas formas. La forma ideal sería, a partir de las últimas N muestras de cada píxel, obtener dichos parámetros, de acuerdo a las expresiones para el cálculo de la media y la desviación típica de una serie de valores:

$$\mu_t=\frac{1}{N}\sum_{i=0}^{N-1}I_{t-i} \tag{2.18}$$

$$\sigma_t=\frac{1}{N}\sqrt{\sum_{i=0}^{N-1}\left(I_{t-i}-\mu_t\right)^2} \tag{2.19}$$

Con la llegada de un nuevo marco, habrá que actualizar los parámetros anteriores. Para ello, descartamos el último marco, añadimos el nuevo, y volvemos a calcular los

parámetros. Siendo este método el más intuitivo y más preciso, queda claro que tanto el tiempo de cómputo como la necesidad de almacenamiento de memoria serán elevados, por lo que esta estrategia no suele ser la elegida. La estrategia más comúnmente seguida es la actualización de los parámetros mediante la denominada *actualización recursiva*. Mediante este método, la media μ de la distribución gaussiana se actualiza de acuerdo a:

$$\mu_t = (1 - \alpha)\,\mu_{t-1} + \alpha I_t \tag{2.20}$$

donde μ_t es la estimación de la media en el instante actual, μ_{t-1} la estimación de la media en el instante anterior, e I_t la muestra actual. El valor de α determina la velocidad con la que el sistema olvida los valores anteriores, y se centra en los valores actuales. Como puede verse, la expresión anterior es similar a la forma de actualizar el modelo del fondo en los algoritmos de imagen de referencia que emplean filtrado de Kalman, o filtros IIR. Por otro lado, existen distintas estrategias para realizar la actualización de la varianza. En [Stauffer99], por ejemplo, la expresión elegida es la siguiente:

$$\sigma_t^2 = (1 - \alpha)\,\sigma_{t-1}^2 + \alpha\,(I_t - \mu_t)^2 \tag{2.21}$$

donde σ_t es la desviación típica en el instante actual, y σ_{t-1} la desviación típica en el instante anterior. En [Kanade98], sin embargo, la actualización de la varianza responde a la siguiente expresión:

$$\sigma_t = (1 - \alpha)\,\sigma_{t-1} + \alpha\,|I_t - \mu_t| \tag{2.22}$$

Por último, en [McKenna00], la varianza se actualiza según:

$$\sigma_t^2 = (1 - \alpha)\left(\sigma_{t-1}^2 + (\mu_t - \mu_{t-1})^2\right) + \alpha\,(I_t - \mu_t)^2 \tag{2.23}$$

La principal ventaja de la actualización recursiva es que no es necesario el almacenamiento de ninguna imagen anterior, ya que sólo intervienen en el cálculo la imagen actual y los parámetros estimados en el instante anterior. La principal desventaja es que de esta forma, los parámetros estimados son más inexactos que si se calcularan empleando las N últimas muestras. No obstante, dado que la precisión conseguida con este método es suficiente, está estrategia de actualización suele ser la seguida por todos los sistemas que emplean una distribución unimodal gaussiana.

A la hora de actualizar los estadísticos de cada uno de los píxeles de la imagen, también es posible seguir los dos tipos distintos de actualizaciones definidas anteriormente para los algoritmos de imagen de fondo de referencia. Es decir:

- **Actualización ciega:** La actualización ciega actualiza los estadísticos de todos los píxeles de la imagen, de acuerdo a (2.20) y (2.21), (2.22) ó (2.23) sin tener en cuenta la clasificación en fondo o primer plano realizado por el módulo de detección de movimiento.

- **Actualización selectiva:** En este caso sí que tenemos en cuenta la clasificación en fondo o primer plano llevada a cabo por el módulo de detección. En general, se consideran dos pesos, α_1 y α_2, de tal forma que si el píxel ha sido clasificado como fondo por el módulo de detección, la influencia de esta nueva muestra sobre los estadísticos actuales sea grande, mientras que si el píxel es clasificado como primer plano, la influencia sea lo menor posible.

Si el sistema va a operar con imágenes en color, la estimación de los parámetros puede complicarse, ya que en este caso, la función de densidad de probabilidad gaussiana se transforma en:

$$\eta\left(\mathbf{x}, \boldsymbol{\mu}, \Sigma\right) = \frac{1}{(2\pi)^{3/2}\left|\Sigma\right|^{1/2}} \mathrm{e}^{-\frac{1}{2}(\mathbf{x}-\mu)^{\mathbf{T}}\Sigma^{-1}(\mathbf{x}-\mu)} \tag{2.24}$$

donde en este caso, $\mathbf{x}$ y $\boldsymbol{\mu}$ son vectores de 3 dimensiones. Si el espacio de color fuese por ejemplo, el *RGB*, dichos vectores serían $\mathbf{x} = (x_{\mathrm{R}}, x_{\mathrm{G}}, x_{\mathrm{B}})^T$ y $\boldsymbol{\mu} = (\mu_R, \mu_G, \mu_B)^T$, aunque cualquier otro espacio de color podría haber sido empleado. Σ se corresponde con la matriz de covarianza, que en este caso es una matriz cuadrada de 3×3 elementos, y Σ^{-1} implica por tanto el cálculo de la matriz inversa. Para evitar un coste computacional elevado, se suele suponer que los tres canales son independientes entre sí, de tal forma que la matriz de covarianza se transforma en una matriz diagonal:

$$\Sigma = \begin{pmatrix} \sigma_{\mathrm{R}}^2 & 0 & 0 \\ 0 & \sigma_{\mathrm{G}}^2 & 0 \\ 0 & 0 & \sigma_{\mathrm{B}}^2 \end{pmatrix} \tag{2.25}$$

y la función de densidad de probabilidad se transforma en:

$$\eta\left(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\sigma}\ \right) = \prod_{i=R,G,B} \frac{1}{\sqrt{2\pi\sigma_i^2}} \mathrm{e}^{-\frac{(x_i-\mu_i)^2}{2\sigma_i^2}} \tag{2.26}$$

Bajo esta suposición, que puede ser asumida sin cometer un error excesivo, el cálculo de los parámetros de la distribución se reduce al cálculo de los parámetros de manera independiente para cada uno de los canales.

El principal inconveniente que presenta la distribución unimodal con función de distribución gaussiana es la baja sensibilidad que presenta en zonas de la imagen donde la actividad es elevada, pudiendo llegar a producir zonas de ceguera total (es decir, no es posible detectar absolutamente nada en dicha zona), si la varianza en dicha zona es elevada. Cuando esto ocurre, otros algoritmos, que asumen otras distribuciones, pueden tener mejores comportamientos que el modelo gaussiano.

Métodos paramétricos bimodales

Entre otros algoritmos, el algoritmo W^4 (*Who? When? Where? What?*) propuesto por [Haritaoglu98] establece una umbralización menos selectiva basada en la hipótesis de una

distribución bimodal para el ruido. El algoritmo computa tres parámetros estadísticos para cada píxel durante el periodo de entrenamiento, y serán posteriormente actualizados tras cada nuevo marco. Estos tres parámetros caracterizan bien la variabilidad del píxel mientras que las condiciones de trabajo sean relativamente similares a las de entrenamiento. En concreto, este algoritmo calcula, para cada píxel:

- La máxima intensidad (N)
- La mínima intensidad (M)
- La máxima diferencia absoluta entre marcos consecutivos (D)

Para la clasificación entre fondo y primer plano, un píxel será considerado como perteneciente al primer plano si se cumple una de estas dos condiciones:

$$|M - I| > f \cdot D \tag{2.27}$$

$$|N - I| > f \cdot D \tag{2.28}$$

donde f es un parámetro global que debe ser ajustado por el usuario. Tras la llegada de cada nuevo marco, el marco actual sustituye al marco más antiguo, y los tres parámetros anteriores son actualizados con el valor de la nueva muestra. Nuevamente, disponemos de los dos tipos de actualización comentado anteriormente, es decir, actualización ciega o selectiva. Sin embargo, en este caso, la actualización selectiva trae consigo el problema definido anteriormente como *abrazo mortal*. En este caso, no existen dos pesos distintos, uno rápido para seguir las variaciones del fondo, y otro lento para que los objetos de primer plano no sean incorporados rápidamente al fondo. Para corregir este problema, la solución incorporada al sistema es establecer, para cada píxel de la imagen, un contador, y contar cuantas veces el píxel ha sido clasificado como primer plano. Si la cuenta alcanza un umbral determinado, las sucesivas muestras de ese píxel son utilizados ya para realizar la actualización de los tres parámetros del algoritmo W^4.

Métodos paramétricos multimodales

Siguiendo con los algoritmos con representación paramétrica de la distribución de probabilidad de los píxeles de la imagen, una opción distinta a los esquemas que asumen una única distribución por píxel, sería suponer que cada píxel puede ser modelado por más de una distribución. Estos algoritmos son denominados algoritmos paramétricos multimodales.

La opción preferida, y la única que consideraremos aquí, es aquella que modela cada píxel como una mezcla de gaussianas (*Mixture of Gaussians, MoG*), como en [Stauffer99]. Cuando un píxel se corresponde con una superficie particular bajo ciertas condiciones de luz, una simple gaussiana sería suficiente para modelarlo, tal y como se asumía en los

modelos unimodales. Si la única perturbación fueran cambios graduales de iluminación, bastaría con que la gaussiana fuese adaptativa, es decir, se actualice con la llegada de cada nuevo marco. No obstante, en la práctica, en un mismo píxel pueden aparecer distintos niveles de gris correspondientes a diversas superficies bajo condiciones cambiantes de luz, por lo que sería mucho más conveniente modelarlo con una mezcla de distintas gaussianas, caracterizadas por su valor medio y su desviación típica. En base a la persistencia y a la variación de cada una de las gaussianas se determina cual o cuales de dichas distribuciones pertenecen al modelo del fondo. En general, se mantendrán K distribuciones gaussianas (K vendrá determinado básicamente por la capacidad de memoria del equipo y del poder computacional del sistema, normalmente oscila entre 3 y 5 distribuciones), pero solamente las N distribuciones más importantes serán las que formen parte del fondo. Los valores de los píxeles que no caigan dentro de los márgenes establecidos por dichas gaussianas se consideran como primer plano.

Si denominamos la historia más reciente de un píxel como $\{x_{t-T}, ..., x_t\}$, y modelamos esta secuencia como una mezcla de K gaussianas, la función de densidad de probabilidad de la mezcla de gaussianas será:

$$\mathrm{P}\left(x_t\right) = \sum_{i=1}^{K} w_{i,t} \cdot \eta\left(x_t, \mu_{i,t}, \sigma_{i,t}\right) \tag{2.29}$$

donde K es el número de gaussianas consideras, tal y como fue discutido anteriormente, $w_{i,t}$ es el peso estimado (que porción de los datos son tenidos en cuenta por esta gaussiana) de la i-ésima gaussiana en el instante t, $\mu_{i,t}$ y $\sigma_{i,t}$ son la media y la varianza respectivamente de la i-ésima gaussiana en el instante t, y η es la función de densidad de probabilidad de la gaussiana, definida como:

$$\eta\left(x, \mu_i, \sigma_i\right) = \frac{1}{\sqrt{2\pi\sigma_i}} \mathrm{e}^{\frac{(x_t-\mu_i)^2}{2\sigma_i^2}} \tag{2.30}$$

Ante la llegada de un nuevo marco, para cada píxel de la imagen debe determinarse si pertenece al fondo o al primer plano. Para ello, se comprueba si el valor de dicho píxel cae dentro de los márgenes establecidos por alguna de las K gaussianas que forman el modelo estadístico. Si existe alguna gaussiana entre cuyos márgenes está comprendido el valor del nuevo píxel, y además dicha gaussiana es una de las N primeras, es decir, las que forman parte del modelo del fondo, el píxel será clasificado como fondo. Si dicha gaussiana se corresponde con las $K - N$ últimas distribuciones, el píxel es clasificado como primer plano. Si no se ha encontrado ninguna gaussiana que asuma el valor de dicho píxel, el píxel también es clasificado como primer plano.

En cualquier caso, una vez realiza la detección de movimiento, las distribuciones para cada píxel deben ser actualizadas. Para ello, consideramos dos situaciones distintas. Si ha sido encontrada alguna gaussiana que asuma el valor del nuevo píxel, los parámetros de esta gaussiana son actualizados de acuerdo a:

$$\mu_{i,t} = (1-\rho)\,\mu_{i,t-1} + \rho x_t \tag{2.31}$$

$$\sigma^2_{i,t} = (1-\rho)\,\sigma^2_{i,t-1} + \rho\,(x_t - \mu_{i,t})^2 \tag{2.32}$$

donde

$$\rho = \alpha\eta\,(x_t, \mu_{i,t}, \sigma_{i,t}) \tag{2.33}$$

mientras que el resto de gaussianas no son modificadas. Si no existe ninguna gaussiana que asuma el valor del nuevo píxel, la última gaussiana es sustituida por una nueva con el valor del nuevo píxel como media, con una varianza inicial relativamente elevada, y un peso prioritario $w_{i,t}$ pequeño. En cualquier caso, haya habido coincidencia o no, el peso de cada una de las gaussianas, w_i, se actualiza de la siguiente manera:

$$w_{i,t} = (1-\alpha)\,w_{i,t-1} + \alpha M_{i,t} \tag{2.34}$$

donde α es el factor de aprendizaje, y $M_{i,t}$ será 1 para la distribución gaussiana que asuma el valor del nuevo píxel, y 0 para el resto de distribuciones. Después de la actualización se vuelven a normalizar para que la expresión (2.29) se corresponda con una función de densidad de probabilidad válida. Si consideramos imágenes en color, y asumiendo que las bandas de color son independientes entre sí, la extensión de este algoritmo para imágenes en color es similar a la realizada para los algoritmos unimodales gaussianos, discutido anteriormente, por lo que no se volverá a tratar aquí.

Una de las ventajas más significativas de este método es que cuando a algo que es nuevo en la escena se le permite empezar a formar parte del modelo, esta acción no destruye el modelo de fondo existente. De hecho, aunque la nueva distribución es incorporada al modelo estadístico, el peso de esta nueva gaussiana será muy pequeño, de tal forma que no estará comprendida entre las N primeras, que son las que realmente conforman el modelo del fondo. Para que un objeto nuevo en la escena comience a formar parte del fondo, es necesario que permanezca estacionario el número de veces suficiente para que el peso de su distribución supere a la N-ésima distribución. Además, aunque un objeto del primer plano haya sido incorporado al modelo del fondo, cuando este desaparece, la distribución que reflejaba el fondo anterior todavía permanece con los mismos valores de media y varianza, aunque con un valor de w menor, pero que serán rápidamente actualizados para recobrar sus valores originales.

El último criterio a considerar sería el método a seguir para ordenar las gaussianas, de tal forma que sean las N primeras las que modelen el fondo. Aunque cabría pensar en ordenar las gaussianas a partir de sus pesos w_i, la opción seguida por [Stauffer99] es ordenarlas en función del valor w_i/σ_i. Para entender esta elección, consideremos en primer lugar las distribuciones correspondientes al fondo. Como los valores correspondientes a estas distribuciones son las que más aparecerán, sus pesos serán elevados. Del mismo

modo, también se puede suponer que las varianzas de estas gaussianas no serán muy elevadas, y su valor vendrá determinado básicamente por el ruido del sensor. Por contra, cuando un nuevo objeto aparece en la escena, éste se inicializa con un valor de w bajo y una varianza σ^2 elevada, que harán que el parámetro w/σ sea muy pequeño. Además, se espera que la varianza de un objeto en movimiento será generalmente mayor que la del fondo, al menos mientras éste esté en movimiento. Con todo esto, se realiza la reordenación de las gaussianas en función del parámetro anterior, y se eligen las N primeras distribuciones, de tal forma que:

$$N = \arg\min_{b} \left(\sum_{j=1}^{b} w_j > T \right) \tag{2.35}$$

donde T viene a ser una medida de cuantas muestras de la secuencia de entrada van a ser tenidas en cuenta para modelar el fondo. Si T es pequeño, solamente la primera distribución será la que pase a formar parte del modelo del fondo, como si sólo tuviésemos en cuenta un número reducido de las muestras de la secuencia de entrada. Según T aumenta su valor, más gaussianas entrarán a formar parte del modelo del fondo, es decir, se tiene en cuenta la información aportada por un mayor número de muestras de la secuencia de entrada.

Una variante de este algoritmo es la presentada por [Utasi07]. En este trabajo, el modelado del fondo paramétrico multimodal sigue los mismos criterios que los vistos anteriormente. Sin embargo, en este caso, el primer plano también tiene su propio modelo. La única diferencia es que el primer plano es modelado con una distribución gaussiana unimodal. Además, el algoritmo realiza el análisis de las distribuciones de los píxeles vecinos, con la idea de reducir el problema de la apertura del primer plano. Es decir, si dos píxeles tienen aproximadamente la misma distribución, y uno de ellos ha sido clasificado como objeto de primer plano, lo más seguro es que el otro píxel también lo sea, reducienda así en parte el citado problema.

Métodos no paramétricos

Los métodos no-paramétricos realizan la estimación de la función de densidad de probabilidad de la secuencia de entrada teniendo en cuenta las últimas muestras recibidas, de acuerdo a la siguiente expresión:

$$\hat{f}(x) = \sum_{i} \alpha_i \mathrm{K}_\sigma (x - x_i) \tag{2.36}$$

donde el sumatorio en i se extiende desde la muestra actual hasta las N últimas muestras de la secuencia y α_i son los coeficientes de ponderación, generalmente uniformes, es decir, $\alpha = 1/N$. K_σ es la función *Kernel* centrada en cada uno de los puntos de los datos de entrada, y el parámetro σ se suele conocer como ancho de banda, o escala, del Kernel, y debe cumplir que:

$$\mathrm{K}_{\sigma}(t) = \frac{1}{\sigma}\mathrm{K}\left(\frac{t}{\sigma}\right) \tag{2.37}$$

Típicamente, la función Kernel suele ser elegida con forma gaussiana, de acuerdo a:

$$\mathrm{K}_{\sigma}(t) = \frac{1}{\sqrt{2\pi\sigma^2}}\mathrm{e}^{-\frac{t^2}{2\sigma^2}} \tag{2.38}$$

Dada una secuencia de entrada $x_{t-1}, x_{t-2}, ..., x_{t-N}$, las expresiones anteriores nos permiten calcular la probabilidad de que la muestra actual, x_t se corresponda con el fondo o con un objeto del primer plano. Ya que (2.36) estima la función densidad de probabilidad del modelo del fondo observado por la cámara, un píxel será considerado como perteneciente al primer plano si la probabilidad de que ese píxel pertenezca al fondo sea menor de un cierto umbral, es decir si:

$$\Pr(x_t) = \frac{1}{N}\sum_{i=1}^{N}\mathrm{K}_{\sigma}(x_i - x_t) < th \tag{2.39}$$

donde th es el umbral global para toda la imagen, que puede ser ajustado para alcanzar un compromiso entre el número de falsos positivos y falsos negativos. Si consideramos imágenes en color, la probabilidad puede ser calculada mediante el producto de Kernels de una dimensión, como sigue:

$$\Pr(x) = \frac{1}{N}\sum_{i=1}^{N}\prod_{j=\mathrm{R,G,B}}\mathrm{K}_{\sigma_j}(x_j - x_{ij}) \tag{2.40}$$

donde j es el índice de la banda de color. En principio, cualquier espacio de color podría emplearse aquí, aunque posiblemente el espacio RGB sea el más empleado. Computacionalmente hablando, la expresión (2.36) puede evaluarse empleando tablas de búsqueda precalculadas que tomen como entrada la diferencia de valores $(x_t - x_i)$ y el ancho de banda σ. Además, para la mayoría de los píxeles de la imagen no va a ser necesario realizar el cálculo para las N muestras, ya que, teniendo en cuenta que la mayor parte de la imagen va a pertenecer al fondo, dichos píxeles superarán el umbral global th tras unas pocas iteraciones del sumatorio.

Uno de los problemas que presenta la implementación de este algoritmo es que siempre va a ser necesario el disponer de las N últimas muestras de cada píxel, es decir, es necesario almacenar los N últimos marcos, con el consiguiente requerimiento de capacidad de almacenamiento. La llegada de un nuevo marco hará que el marco más antiguo sea sustituido por este nuevo. Una última cuestión a tener en cuenta es la determinación del ancho de banda σ de la función Kernel. Teóricamente, si el número de muestras fuese infinito, el ancho de banda no tendría importancia, y la función estimada se correspondería con la función de densidad de probabilidad real. Sin embargo, a medida que el número de muestras tiende a ser pequeño, la elección de un ancho de banda adecuado se vuelve más importante. Además, ya que el comportamiento del fondo va a ser diferente en cada parte

de la escena, cada píxel de la imagen tendrá, en general, un ancho de banda distinto. La solución propuesta por [Elgammal02] calcula la mediana del valor absoluto de la diferencia entre píxeles consecutivos, es decir:

$$m = \text{MED}\,|x_{t-i} - x_{t-i-1}| \quad i = [1...N] \tag{2.41}$$

La razón de emplear la mediana se debe a que la secuencia de valores para la señal analizada va a tener saltos grandes si el píxel en cuestión es ocupado por distintos objetos. En general, estos saltos, que son los que podrían distorsionar la estimación del ancho de banda, no van a ser numerosos, y por tanto, nunca se corresponderán con la mediana de la secuencia. Asumiendo que la distribución local en el tiempo es gaussiana $\eta(\mu, \sigma)$, entonces la distribución de la diferencia $x_i - x_{i-1}$ es también gaussiana $\eta(0, 2\sigma^2)$. A partir de esta suposición, el ancho de banda es calculado de acuerdo a:

$$\sigma = \frac{m}{0{,}68\sqrt{2}} \tag{2.42}$$

Por último, la extensión para imágenes en color, asumiendo independencia entre canales de color, es similar a como se realizó para las modelos paramétricos, y por tanto, no se volverá a discutir aquí.

2.3. Otros métodos de detección de movimiento

Los métodos vistos hasta ahora, basados, de una forma muy genérica, en la comparación del marco actual con una estimación obtenida a partir de las imágenes anteriores, no son los únicos métodos que permiten realizar la estimación de movimiento en una escena de vídeo. En [Meier98] la detección de movimiento se obtiene a partir de un bloque de reconocimiento y seguimiento de objetos. En este trabajo, el sistema crea un modelo del objeto en un marco determinado, y emplea la distancia de Hausdorff para realizar el seguimiento del objeto en los siguientes marcos. Este modelo de funcionamiento permite ser más inmune al movimiento del fondo y a los cambios de iluminación, pero requiere mayor carga computacional.

En [Iwata08] se emplea una implementación eficiente mediante instrucciones SIMD (*Single Instruction Multiple Data*) del algoritmo CHLAC (*Cubic Higher-order Local Auto-Correlation*) para realizar la detección de patrones de movimiento. El algoritmo CHLAC fue definido por [Kobayashi04], y es una extensión del algoritmo HLAC definido por [Otsu88]. HLAC realiza una auto-correlación de orden N entre píxeles vecinos, pero su motivación principal era para el reconocimiento de objetos en imágenes fijas. CHLAC incluye a esa correlación las imágenes previas, y lo que realiza en este caso es la identificación de patrones de movimiento en secuencias de imágenes.

2.4. Sistema implementado

El principal inconveniente de los algoritmos de detección de movimiento descritos en la literatura es su poca sensibilidad en escenas con movimiento del fondo, por ejemplo, árboles agitados por el viento, vehículos en movimiento o cambios de iluminación. El efecto que este ruido provoca puede verse en función de los tipos de algoritmos descritos en este capítulo.

Para los algoritmos derivativos y los de fondo con imagen de referencia, donde el umbral de decisión es fijo para toda la imagen y éste suele ser pequeño (ligeramente superior al nivel de ruido de la cámara para conseguir que tenga la mayor sensibilidad posible), cualquier movimiento del fondo suele generar una falsa alarma en el sistema. Si elegimos un umbral mayor, entonces lo que aumenta es la probabilidad de pérdida, que es si cabe, mucho más grave que la probabilidad de falsa alarma.

En las figuras 2.2, 2.3, 2.4 y 2.5 se representan estos efectos. La figura 2.3 corresponde a la máscara de movimiento obtenida al aplicar el algoritmo *SAD* a una secuencia de vídeo mientras que en la figura 2.2 se muestra uno de sus marcos. Como puede observarse, las zonas de la imagen que se corresponden con zonas de fondo dinámico (árboles principalmente en este caso) presentan una gran probabilidad de falsa alarma, aun cuando el umbral ha sido establecido por encima del nivel de ruido de la cámara, como puede verse por la ausencia de falsas alarmas en las zonas estáticas (por ejemplo, la parte superior de la imagen, que se corresponde con el cielo).

Si, por el contrario, aumentamos deliberadamente el umbral para evitar que en las zonas de fondo dinámico aumente la probabilidad de falsa alarma, ocurrirá que aumenta peligrosamente la probabilidad de pérdida. Esto puede verse por ejemplo en la figura 2.5, que se corresponde con la máscara de movimiento para la misma secuencia anterior, en donde se ha aumentado el umbral de tal forma que en las zonas de fondo dinámico la probabilidad de falsa alarma sea lo más pequeña posible. En este caso, lo que conseguimos es que no sea posible la detección de objetos de interés, como puede ser el coche que aparece en la parte inferior derecha de la figura 2.4.

Los algoritmos estadísticos son capaces de corregir este problema adaptándose a las características de cada parte de la imagen. Así, para las zonas poco ruidosas el sistema establecerá un umbral pequeño, y según el ruido se hace mayor, el umbral también. El principal problema estriba en que estos algoritmos suponen que el ruido presente en la imagen es gaussiano, pero esto no es cierto cuando la variación del nivel de un píxel está debida principalmente al movimiento de los objetos del fondo o cambios de iluminación, y no al ruido. En estos casos, y para evitar una probabilidad de falsa alarma elevada, los algoritmos de detección de movimiento establecen para esta zona un umbral elevado, que hará que estas zonas sean menos sensibles que las zonas sin movimiento de fondo, aumentando por tanto, la probabilidad de pérdida.

Por ejemplo, si sobre la secuencia de la imagen 2.6 aplicamos el algoritmo de detección de movimiento unimodal gaussiano obtendríamos, a partir de la secuencia, la media y la

Figura 2.2: Primer marco de la secuencia empleada para la ilustración de los problemas más importantes de un sistema de videovigilancia típico.

Figura 2.3: Efecto de un umbral demasiado bajo en el cálculo de la máscara de movimiento.

Figura 2.4: Marco de la secuencia empleada para comprobar el efecto de un umbral demasiado alto.

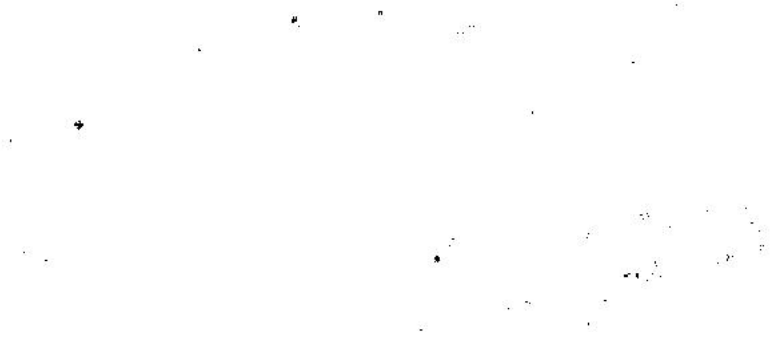


Figura 2.5: Efecto de un umbral demasiado grande sobre la máscara de movimiento.

Figura 2.6: Marco de la secuencia empleada para ilustrar el funcionamiento del algoritmo unimodal gaussiano.

desviación típica para cada uno de los píxeles de la imagen. Si representamos sobre una imagen en niveles de grises el valor de la desviación típica, teniendo en cuenta que el color negro corresponde a un valor de la desviación igual a cero, y el color blanco al nivel máximo de dicha desviación, obtendríamos la imagen representada en la figura 2.7. Teniendo en cuenta el funcionamiento del bloque de detección de movimiento en este algoritmo, queda claro que aquellas partes de la escena que aparezcan con color blanco, o gris claro, van a ser menos sensibles al movimiento de los objetos de primer plano, por tanto, aumentando la probabilidad de pérdida en estas zonas.

En este trabajo vamos a profundizar en el análisis de escenas con alto contenido de movimiento de fondo, tratando de reducir la probabilidad de falsa alarma y de pérdida lo máximo posible. Para conseguir este objetivo, se propone desarrollar un sistema que clasifique cada zona de la imagen en función de su comportamiento más reciente, de tal forma que las siguientes etapas, como puede ser la detección de movimiento, analicen cada zona de la imagen de acuerdo a dicha clasificación. Para ello el sistema va a disponer, al igual que lo hacen la mayoría de los algoritmos de imagen de referencia o estadísticos, de un periodo de entrenamiento, y posteriormente, tras la llegada de cada nueva imagen, dicha clasificación será actualizada.

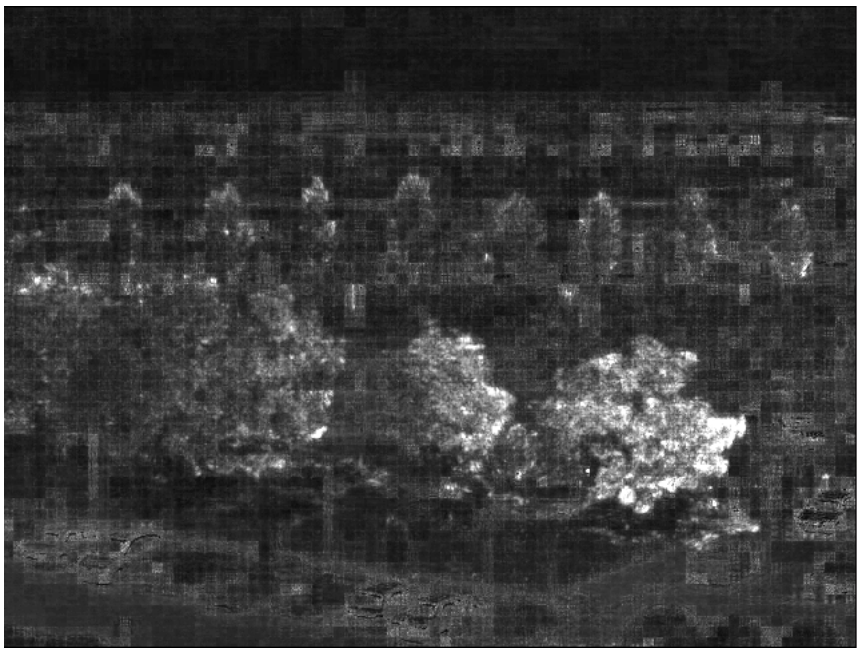


Figura 2.7: Visualización de la desviación típica calculada sobre la secuencia de ejemplo de la figura 2.6

2.4.1. Categorías de comportamiento del fondo

Para realizar la clasificación de cada zona de la escena, se analizará el comportamiento de cada píxel de la imagen, y se asignará a uno de los siguientes grupos. Dichos grupos han sido creados a partir del análisis de distintas secuencias de imágenes, en particular aquellas que presentan mayores problemas a la hora de realizar la detección de movimiento. Por lo tanto, podemos decir que el comportamiento de un píxel debe identificarse en uno de los siguientes tipos:

- **Fondo estático:** A este grupo pertenecen aquellos píxeles cuyo valor sea casi constante a lo largo de la secuencia. En imágenes reales, este grupo corresponde a las zonas de la escena que no se ven afectadas por movimiento del fondo o cambios de iluminación. En la figura 2.9(a) puede verse un ejemplo de la evolución temporal de un píxel que ha sido clasificado en este grupo. Dicho píxel ha sido obtenido de la secuencia *parking1*, de la cual, en la figura 2.8 se muestra su primera imagen, y en concreto, corresponde al píxel cuya posición está señalada con la marca "A".

- **Fondo dinámico:** Los píxeles que pertenecen a este grupo se caracterizan porque su valor varía notablemente a lo largo del tiempo, y su varianza es tan elevada que dicho píxel sería insensible a la detección de movimiento si empleáramos algoritmos clásicos de detección de movimiento como los vistos anteriormente. Por ejemplo, en la figura 2.9(b) se muestra un ejemplo de evolución temporal de un píxel pertene-

ciente a este grupo, obtenido de la secuencia *Parking1*, en concreto, el píxel señalado con la marca "B" en la figura 2.8.

- **Fondo impulsivo:** En ciertas ocasiones, el fondo puede ser completamente estático durante la mayor parte del tiempo, pero aleatoriamente, su nivel cambia rápidamente y vuelve a su valor original, como si fuera una especie de impulso. Este comportamiento puede corresponder, por ejemplo, a una carretera en la que la imagen que predomina durante la mayor parte del tiempo es la carretera, que correspondería a un fondo estático, y aleatoriamente un vehículo puede aparecer en la escena y desaparecer en la siguiente. Por ejemplo, en la figura 2.9(c) se muestra un ejemplo de evolución temporal de un píxel perteneciente a este grupo, obtenido de la secuencia *Parking1*. En este caso, la posición de este píxel está indicado con la marca "C" en la imagen 2.8.

- **Fondo cambiante:** En una escena, ciertas partes de una imagen pueden comportarse de forma estática durante una buena parte del tiempo, y de repente, en un breve espacio de tiempo, cambian el valor de los píxeles y se mantienen en dicho valor nuevamente durante un tiempo relativamente largo. En la figura 2.9(d) se muestra un ejemplo de evolución temporal de un píxel cambiante, obtenido nuevamente de la figura *parking1*, e indicado en la figura 2.8 con la marca "D". Esta categoría modela principalmente el comportamiento del cielo, donde, si hay presencia de nubes por ejemplo, dicha zona de la imagen puede estar alternando entre distintos valores, pero a un ritmo bastante lento. También modela las zonas de la imagen donde puede aparecer un objeto nuevo, o bien desaparecer un objeto existe, principalmente pensando en la detección de objetos abandonados o robo de objetos.

A diferencia de los algoritmos de detección de movimiento vistos hasta ahora, donde la clasificación entre fondo y primer plano se hacía de forma idéntica para todos los píxeles de la imagen, en nuestro caso, el método de clasificación será función del grupo al que pertenezca cada píxel, como se describirá posteriormente. Dos son por tanto las tareas a realizar por el sistema de detección de movimiento a implementar. Por un lado, habrá que establecer el criterio para realizar la clasificación de los píxeles de la imagen en cada uno de los grupos descritos anteriormente, y una vez clasificados, se deberá aplicar a cada píxel el algoritmo de detección de movimiento correspondiente, en función de la clase a la que pertenece.

2.4.2. Clasificación del fondo

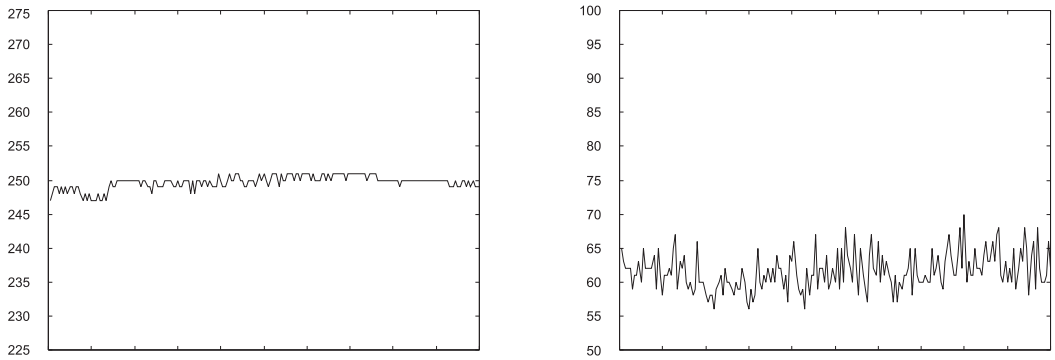
La diferencia principal entre los cuatro grandes grupos en los cuales vamos a clasificar cada píxel de la imagen es la variación a lo largo del tiempo de su valor. Así, para los píxeles que pertenecen al fondo estático, su varianza es debida únicamente al ruido de la cámara, que en general será suficientemente pequeño. Para los píxeles que pertenecen

Figura 2.8: Primera imagen de la secuencia Parking1

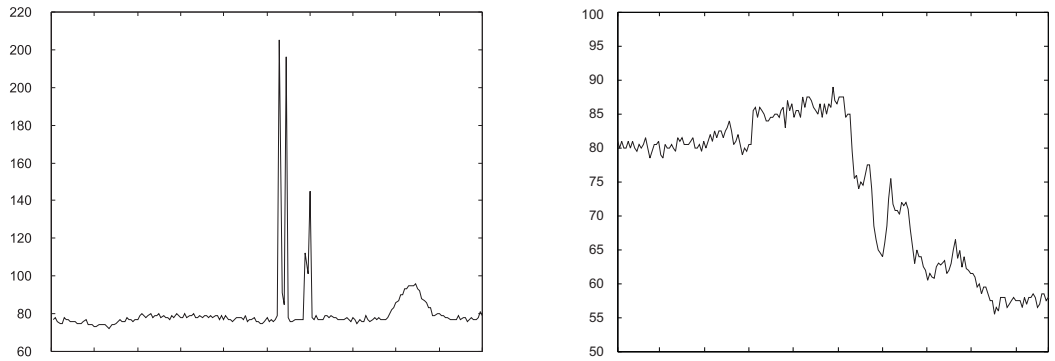
al fondo impulsivo, durante el tiempo que la imagen corresponde al fondo estático, la varianza de éste también será pequeña, del mismo nivel que para el grupo anterior. Sin embargo, para los píxeles del fondo dinámico, como la variación de sus niveles viene motivada principalmente por el movimiento del fondo, que será de un nivel mucho mayor que el ruido de la cámara, la varianza temporal de estos píxeles será mucho mayor que la de los otros dos grupos. Por último, para el fondo cambiante tendremos una varianza pequeña y una media que se modifica bruscamente.

De esta forma, para diferenciar entre píxeles de fondo estático, impulsivo o cambiante, y píxeles de fondo dinámico, vamos a umbralizar la varianza temporal de cada píxel, de tal forma que aquellos que superen el umbral pertenecerán al fondo dinámico, mientras que los que no lo superen corresponderán a cualquiera de los otros grupos. Para evitar que en los píxeles que pertenezcan al fondo impulsivo las muestras que no corresponden al fondo estático aumenten indeseadamente la varianza de estos píxeles, trataremos de eliminar dichos valores para que en el cómputo de la varianza únicamente entren en juego los que corresponden al periodo estático. Así, el proceso para realizar la clasificación de cada píxel sería el siguiente:

- Cálculo de la media y la varianza temporal para cada píxel de la imagen. Ya que se trata de un cálculo continuo, los métodos idóneos son aquellos que permiten su actualización marco a marco, como los descritos en la página 31 (2.20, 2.21), (2.22) ó (2.23). No obstante, en esta tesis propondremos un nuevo método de cálculo para la estimación del fondo y la varianza con el que hemos obtenido mejores resultados



(a) Píxel estático (b) Píxel dinámico



(c) Píxel impulsivo (d) Píxel cambiante

Figura 2.9: Ejemplos de evoluciones temporales de distintos píxeles de la secuencia de la figura 2.8

cuando el sistema trabaja bajo condiciones de iluminación adversas, o la calidad de las imágenes es baja.

- Detección y recuento de impulsos para cada píxel. En este punto, deberemos ser capaces de determinar, para cada nuevo marco, si el nuevo valor del píxel se corresponde con un impulso. Cada vez que se detecta un impulso, no se realiza el cálculo de la media ni la varianza para dicho píxel. De esta forma, evitamos que para píxeles pertenecientes al fondo impulsivo, las muestras que no corresponden al fondo estático modifiquen erróneamente los estadísticos de dicho píxel. El algoritmo de detección de impulsos será explicado más adelante.

- La clasificación entre píxeles pertenecientes al fondo impulsivo y píxeles pertenecientes al fondo estático, cambiante o dinámico se realiza analizando el número de impulsos detectados. Está claro que las falsas alarmas y los objetos de primer plano provocarán que se detecten impulsos en píxeles que no pertenecen a este tipo de fondo, pero el análisis del número de impulsos detectados nos ayudará a discriminar entre clases.

- la clasificación entre píxeles pertenecientes al fondo estático o cambiante, y píxeles pertenecientes al fondo dinámico se realiza umbralizando la varianza del píxel.

- Finalmente, la clasificación entre píxeles estáticos y cambiantes es más difícil, ya que los píxeles cambiantes serán vistos como píxeles estáticos hasta que no cambie su valor. En este caso no se trata de una clasificación permanente, sino más bien cada vez que se detecte un pixel cambiante, esto queda registrado, indicando al sistema que dicha zona de la imagen puede volver a sufrir nuevos cambios. Los píxeles cambiantes serán aquellos que presenten una variación brusca de su media sin variación apreciable de su varianza.

Para poder realizar la clasificación anteriormente expuesta, es necesario en primer lugar obtener los estadísticos de la escena. Como veremos a continuación, uno de los estadísticos más difíciles de estimar, pero que es fundamental para un funcionamiento adecuado del sistema, es el valor medio. Ya que el valor medio de la escena puede verse como una imagen del fondo cuando en ésta no hay objetos de primer plano, a partir de ahora llamaremos a este estadístico como el *fondo* de la escena. Aunque existen muchas técnicas de estimación y mantenimiento de este estadístico, en la siguiente sección vamos a proponer un nuevo algoritmo que se adapta mejor a escenas con gran cantidad de movimiento, como es el caso de nuestro trabajo. Además, también propondremos un método alternativo para el cálculo de la varianza, y finalmente, se expondrán los algoritmos que permitirán a la etapa final poder clasificar cada píxel en su categoría correspondiente.

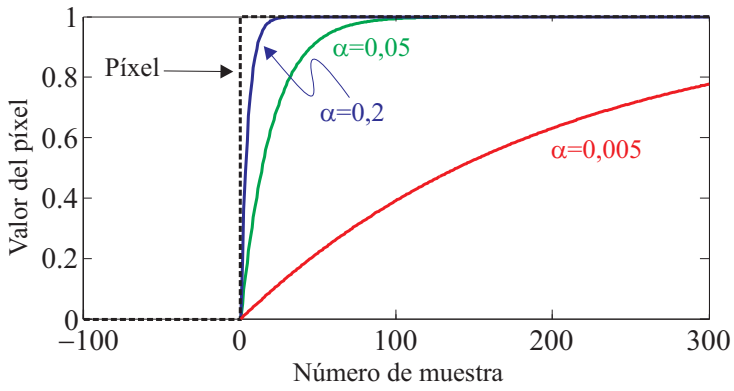


Figura 2.10: Simulación de la evolución de la estimación continua del fondo para distintos valores del parámetro α

Estimación y mantenimiento del fondo

Aunque el método más comúnmente empleado para la estimación del fondo, tal y como se vio en la revisión bibliográfica expuesta anteriormente, es la de la actualización de una imagen de referencia con la nueva imagen mediante un parámetro de actualización α, de acuerdo a la siguiente expresión:

$$F_t = (1 - \alpha)\, F_{t-1} + \alpha I_t \tag{2.43}$$

dicha expresión debe ser revisada más detenidamente cuando la escena a analizar tenga gran cantidad de movimiento del fondo. En la expresión anterior, el parámetro α (que toma valores entre 0 y 1) determina la velocidad con la que se actualiza el fondo. Un valor de α elevado hará que el fondo se actualice rápidamente y siga las variaciones de la escena. Esto es útil cuando la imagen del fondo tiene variaciones continuas, como puede ser cambios de iluminación más o menos bruscos, modificaciones en los parámetros de la cámara, etc. Sin embargo, esta característica hará que los objetos en movimiento se incorporen rápidamente al fondo, lo cual no es deseable. Un valor de α demasiado bajo hará que la estimación del fondo no siga perfectamente las variaciones de éste, obteniendo por tanto una estimación que no va a ser de utilidad para el sistema. Antes de exponer el método empleado en esta tesis, vamos a analizar entre qué rango de valores puede variar el valor de α. Ayudándonos de la figura 2.10, podemos obtener una estimación de este rango. Dicha figura simula la evolución del fondo estimado ante una variación brusca del nivel del píxel desde 0 hasta 1 alcanzado en el transcurso de una única imagen. Aunque la figura muestra que, independientemente del valor de α, la estimación del fondo alcanza el valor final del píxel, dependiendo del valor de α, éste se alcanzará antes o después.

Por ejemplo, para un valor de α igual a 0,05, vemos que se alcanza un valor próximo al valor final en aproximadamente 20 imágenes. Si consideramos un sistema de videovigilancia típico, trabajando a 2 imágenes pos segundo, tenemos que el tiempo de estabilización

es de unos 10 segundos, lo que a todos los efectos puede considerarse como suficientemente rápido. Sin embargo, si bajamos hasta 0,005, este periodo ya llega a ser de unas 400 imágenes, lo que equivale a más de tres minutos, lo que es en general inaceptable para un sistema de videovigilancia. Para valores inferiores el periodo es tan grande que en un sistema real sería imposible seguir las variaciones del fondo. En escenas estáticas, donde la única modificación fuera la variación gradual de la iluminación, un valor de $\alpha = 0{,}005$ podría ser más que suficiente, ya que el ritmo de variación del fondo sería suficientemente lento como para que el sistema pudiera seguir sus variaciones. Sin embargo, en escenas reales como las que analizaremos en este trabajo, las variaciones del fondo se deben no solo a la variación gradual de la iluminación, sino que también pueden existir variaciones más bruscas de ésta, debido por ejemplo a las nubes, alteración de los parámetros de la cámara, sombras, etc, por lo que valores más próximos a 0,05 serían más deseables. En la misma figura, podemos ver que para un valor igual a 0,2, el sistema tarda únicamente unas 10 imágenes en aproximarse al valor final.

También es importante tener en cuenta el número de imágenes que procesa el sistema. En la estimación anterior consideramos que el sistema trabajaba a 2 imágenes por segundo. Sin embargo, también se pueden considerar sistemas trabajando a menor velocidad, o incluyo a velocidades mayores. Por ejemplo, un sistema típico puede procesar 25 imágenes por segundo. En este caso, para un valor de $\alpha = 0{,}005$, el tiempo de estabilización pasaría a ser de 16 segundos, lo cual podría empezar a ser aceptable. En cualquier caso, la conclusión es que el tiempo de estabilización del fondo depende tanto del valor del parámetro α como del número de imágenes que procesa el sistema por segundo. En esta tesis, vamos a considerar siempre un sistema trabajando a dos imágenes por segundo, así que son válidas las consideraciones sobre el valor de α comentadas en el párrafo anterior.

En la figuras 2.12, 2.13 (a) y 2.14 (a) podemos comprobar la precisión en el seguimiento de las variaciones del fondo en una secuencia real, en concreto, en la secuencia *NII*, cuya primera imagen puede verse en la figura 2.11 para distintos valores del parámetro α.

La figura 2.12 corresponde a un píxel estático, que apenas varía su valor a lo largo de la secuencia. En este caso puede comprobarse que cualquier valor de α podría seguir las ligeras variaciones del fondo sin ningún problema.

Sin embargo, la figura 2.13 (a) corresponde a un píxel impulsivo, donde cada uno de los impulsos que se pueden ver en la figura corresponde al paso de un vehículo. En este caso vemos que tanto la estimación con $\alpha = 0{,}2$ o $\alpha = 0{,}05$ incorporan rápidamente el nuevo objeto al fondo, cuando sería deseable que esto no ocurriera. También podemos ver como la estimación con $\alpha = 0{,}005$ es incapaz de seguir las ligeras variaciones del fondo.

Por último, en la figura 2.14 (a) corresponde a un píxel que se comporta de manera estática durante una buena parte de la secuencia, pero que a partir de la imagen 90 aproximadamente, cambia su valor para permanecer en este nuevo valor indefinidamente. En este caso, vemos como únicamente la estimación del fondo con un valor de α grande (en este caso 0,2) es capaz de seguir este variación, y por otro lado, vemos como la estimación empleando $\alpha = 0{,}005$ es incapaz de seguir en ningún momento las variaciones del fondo,

Figura 2.11: Primera imagen de la secuencia *NII*. En la imagen se encuentran marcados con los números 1, 2 y 3 la posición de los píxeles cuya estimación del fondo se representa en las figuras 2.12, 2.13 y 2.14

por lo que en este punto podríamos determinar ya que valores de $\alpha = 0{,}005$ o menores nunca serán recomendables, al menos en este tipo de escenas. Para facilitar la visualización de este efecto, en la figura 2.15 (a) se muestra el mismo conjunto de señales anteriores ampliada.

A partir de los resultados anteriores, podemos concluir que el principal problema a la hora de estimar y mantener el fondo de la escena lo vamos a encontrar, como sería de esperar, en las zonas donde existe movimiento del fondo. Otra conclusión, que nos servirá para desarrollar el método de estimación del fondo propuesto, es la estimación del valor del parámetro α ideal. Aunque esto va a depender de las condiciones particulares de la aplicación, como el tipo de escena, tipo de cámara, etc, en unas condiciones genéricas, como pueden ser las analizadas en la escena anterior, podemos ver que el valor ideal se encuentra entorno a 0,05, siempre y cuando le aportemos una *ayuda extra* para poder seguir al fondo en casos excepcionales, como impulsos, cambios bruscos del fondo, etc.

Es decir, si tenemos en cuenta que en la escena van a aparecer objetos pertenecientes al fondo en movimiento, especialmente en los píxeles impulsivos, un valor de α demasiado alto hará que los objetos del fondo se incorporen rápidamente a la estimación, como puede verse por ejemplo en la figura 2.13 (a). Aunque la solución más popular para este problema

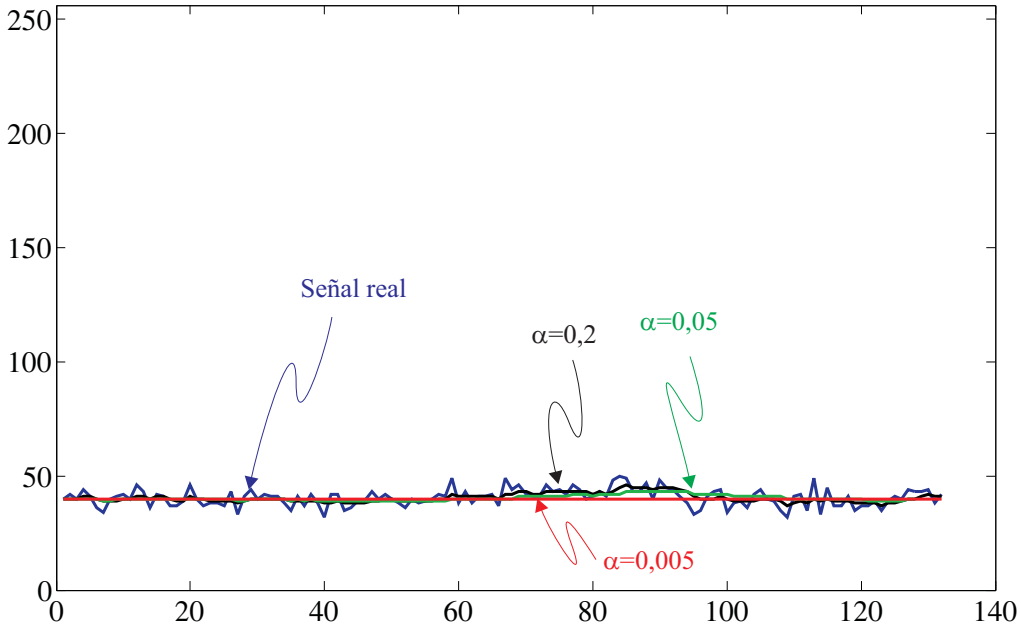
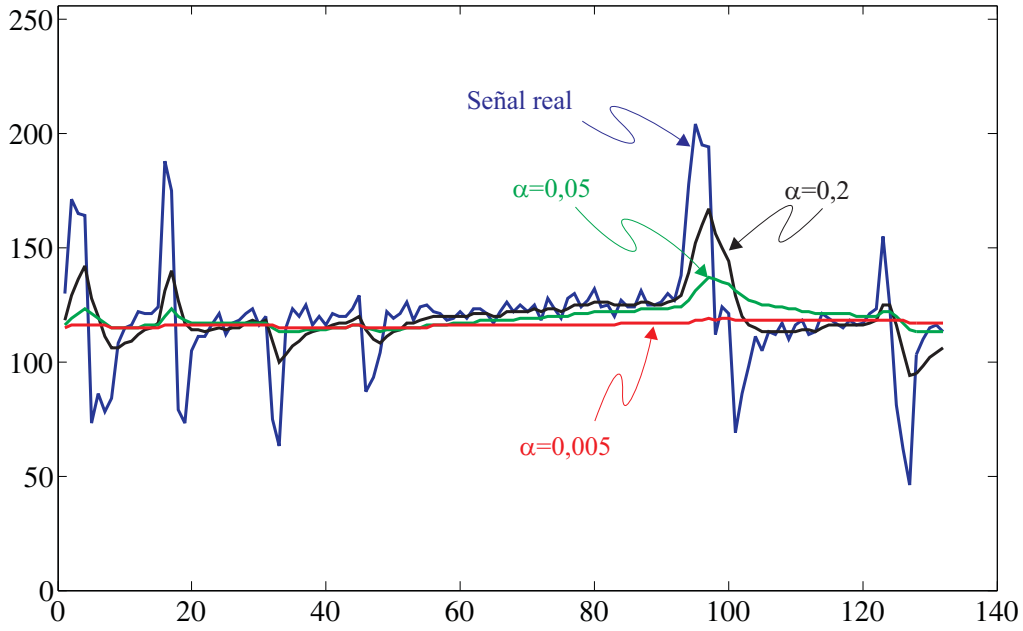


Figura 2.12: Evolución de la estimación continua del fondo para distintos valores del parámetro α para un píxel estático (ver marca '1' en la figura 2.11.)

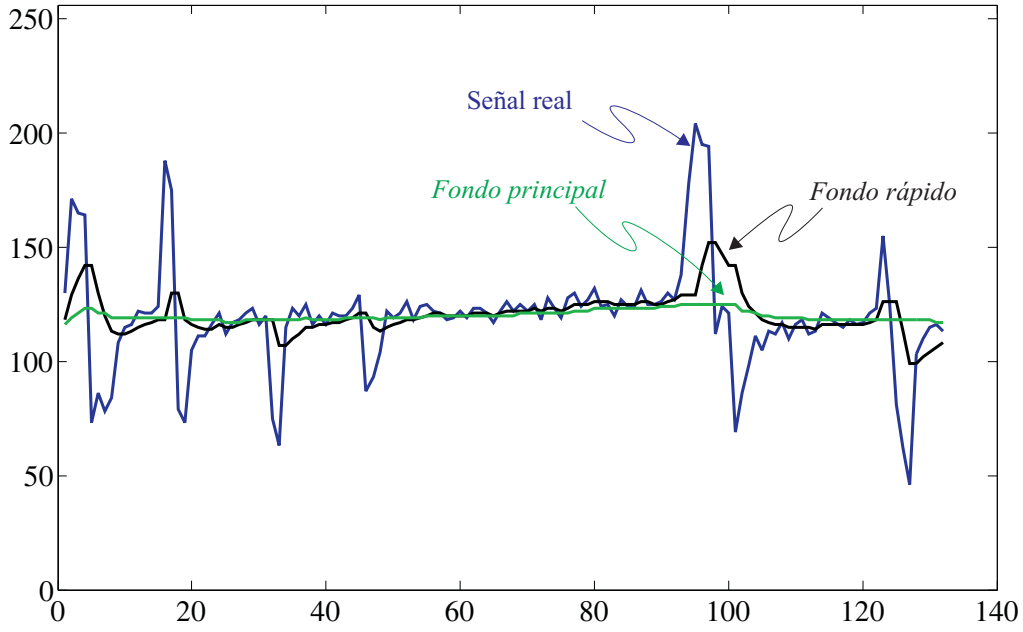
es la llamada actualización selectiva, que implica la actualización del fondo únicamente cuando no se ha detectado movimiento en la escena, o bien la actualización con distintos valores de α dependiendo de si se detectó o no movimiento, en esta tesis vamos a proponer un método algo más elaborado que nos a reportado mejores resultados, especialmente en zonas de la imagen pertenecientes al fondo impulsivo y cambiante.

Para ello, aparte de la estimación del fondo principal, que se calcula con un valor de α igual o próximo a 0,05, vamos a estimar otro fondo que se actualice mucho más rápido, con un valor de $\alpha = 0{,}2$. Este nuevo fondo, que vamos a denominar a partir de ahora como *fondo rápido*, no va a constituir un nuevo estadístico para el sistema, únicamente servirá para controlar la forma en que se actualiza el fondo principal. Tal y como podemos ver en la figura 2.13 y 2.14 (o 2.15 para una versión ampliada de la anterior figura), sería interesante que la estimación sea capaz de seguir las variaciones del fondo cuando éstas son graduales, y no actualizarse, o actualizarse lentamente cuando las variaciones son bruscas. Sin embargo, hay que tener en cuenta que existirán variaciones bruscas que también deben ser seguidas, como por ejemplo, las variaciones del color en el cielo, o en general, de los píxeles que hemos definido como cambiantes.

De esta forma, la propuesta es calcular la diferencia entre la imagen actual y el fondo rápido, y si ésta es superior a un determinado umbral, que puede ser el mismo empleado para la detección de movimiento, el fondo no se actualiza, tal y como se recoge en la siguiente expresión:



(a)



(b)

Figura 2.13: Evolución de la estimación continua del fondo para un píxel impulsivo (ver marca '2' en la figura 2.11). (a) Estimación según el método clásico para distintos valores del parámetro α. (b) Estimación según el método propuesto en esta tesis doctoral.

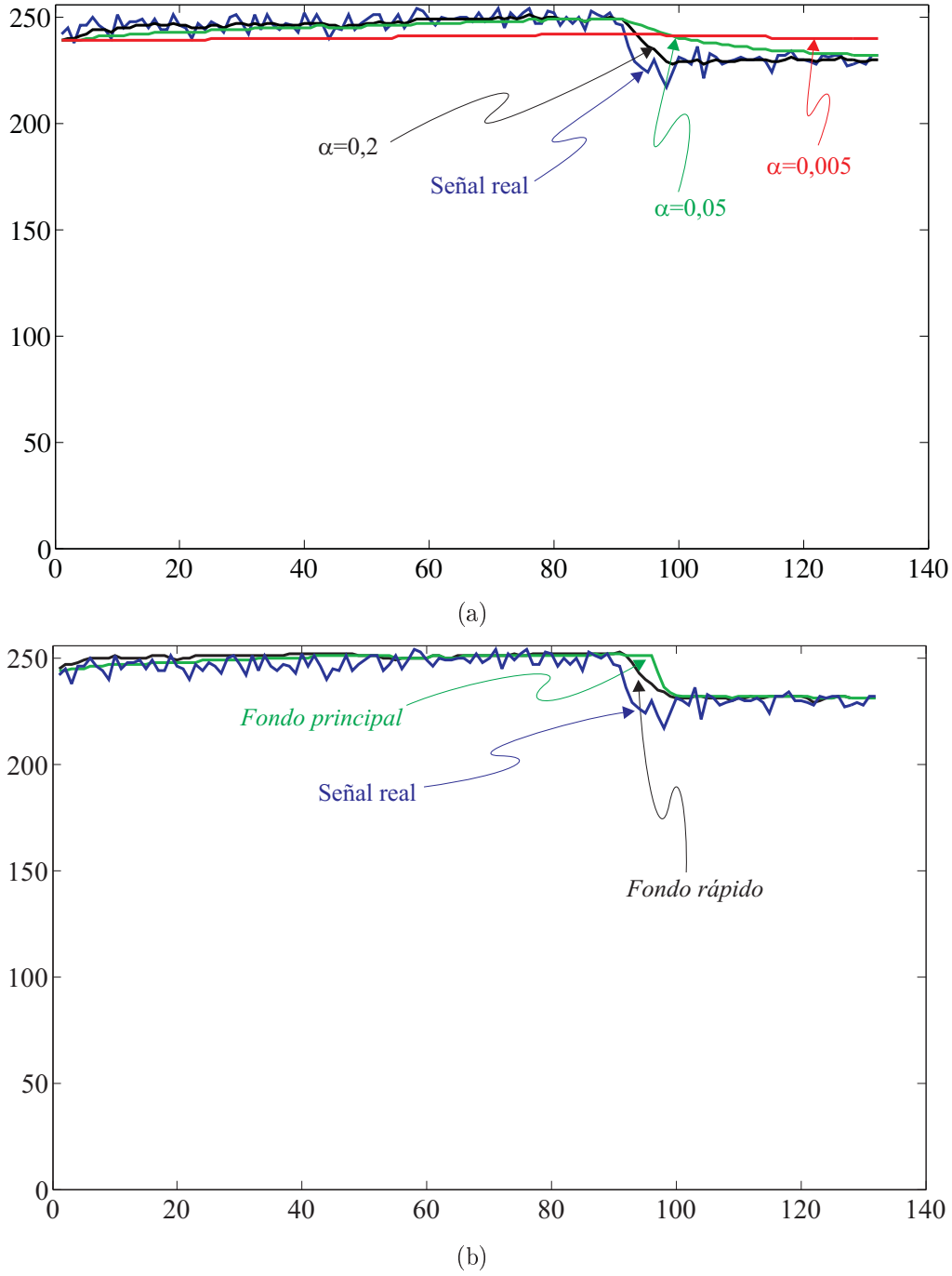


Figura 2.14: Evolución de la estimación continua del fondo para un píxel cambiante (ver marca '3' en la figura 2.11). (a) Estimación según el método clásico para distintos valores del parámetro α. (b) Estimación según el método propuesto en esta tesis doctoral.

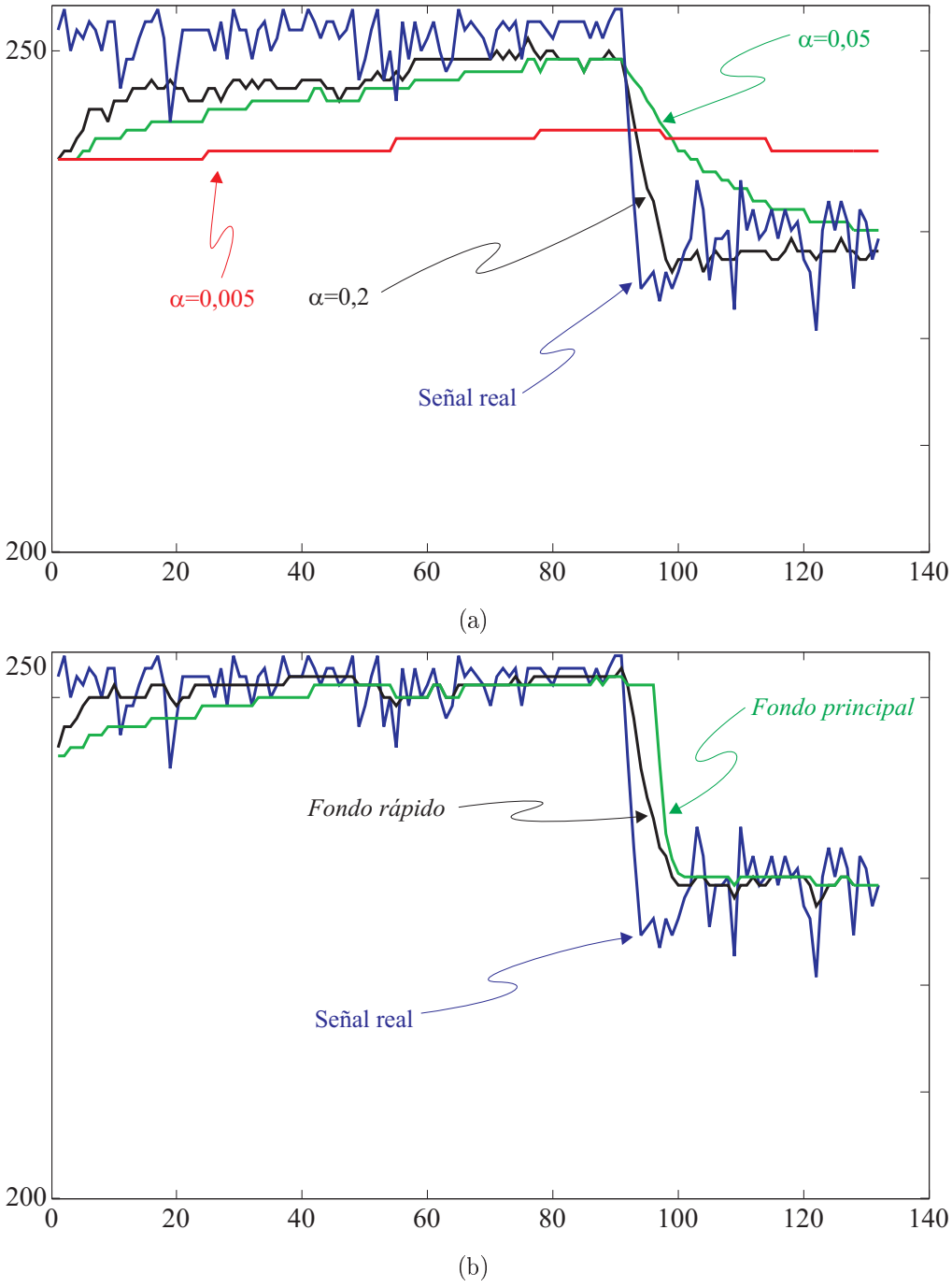


Figura 2.15: Zoom de la figura 2.14 (a) y (b) sobre la zona donde se produce el cambio del valor del píxel.

$$F_t^{\mathrm{P}} = \begin{cases} F_{t-1}^{\mathrm{P}} & \text{si} \quad \left|I_t - F_t^{\mathrm{R}}\right| > \sigma_t \\ (1-\alpha_{\mathrm{P}})\, F_{t-1}^{\mathrm{P}} + \alpha_{\mathrm{P}} I_t & \text{resto} \end{cases} \tag{2.44}$$

donde F^{P} es el fondo principal, mientras que F^{R} es el fondo rápido. Así mismo, como ahora existen dos valores para la constante de actualización α, se ha identificado la de actualización del fondo principal con la notación α_{P}. Por último, σ_t se corresponde con la desviación típica del píxel en la imagen actual. El cálculo de su valor será detallado en el siguiente apartado.

Con la expresión anterior tendríamos resuelto el problema del mantenimiento del fondo en presencia de movimiento. Sin embargo, no resolvería el problema de los píxeles cambiantes. Para ello, se propone la siguiente modificación de la expresión anterior:

$$F_t^{\mathrm{P}} = \begin{cases} F_{t-1}^{\mathrm{P}} & \text{si} \quad \mathrm{Nv}\left(\left|I_t - F_t^{\mathrm{R}}\right| > \sigma_t\right) < n \\ (1-\alpha_{\mathrm{R}})\, F_{t-1}^{\mathrm{P}} + \alpha_{\mathrm{R}} I_t & \text{si} \quad \mathrm{Nv}\left(\left|I_t - F_t^{\mathrm{R}}\right| > \sigma_t\right) \geqslant n \\ (1-\alpha_{\mathrm{P}})\, F_{t-1}^{\mathrm{P}} + \alpha_{\mathrm{P}} I_t & \text{resto} \end{cases} \tag{2.45}$$

La expresión $\mathrm{Nv}\,()$ es una función que cuenta el número de veces consecutivas que la expresión interior es cierta. De esta forma, la expresión $\mathrm{Nv}\left(\left|I_t - F_t^{\mathrm{R}}\right| > \sigma_t\right)$ va a contar el número de veces consecutivas que el fondo rápido se desvía notablemente de la imagen actual.

Como con un valor de $\alpha = 0{,}2$ el fondo tarda entre 5 y 10 marcos en alcanzar las variaciones bruscas del fondo, cuando esta variación ha sido debida a un impulso o un objeto del primer plano en movimiento, la función $\mathrm{Nv}\,()$ nunca llegará a ser mayor de un determinado valor. Si elegimos para n un valor ligeramente superior, la función anterior nunca se activará en presencia de movimiento, y por tanto el fondo principal no se verá afectado por el movimiento de dicho objeto.

Por otro lado, si el fondo cambia de nivel, para permanecer en este nuevo nivel por un tiempo indefinido, el fondo rápido tardará entre 5 y 10 marcos en alcanzar este nuevo valor, haciendo que el fondo principal también se actualice rápidamente, alcanzando un valor próximo al del fondo en menos de 10 marcos. La constante n en la expresión anterior nos permitirá discriminar entre ambos fenómenos, es decir, un objeto de primer plano, o un cambio en el fondo. Se ha elegido, para el tipo de escenas con las que estamos experimentando, un valor igual a 5. Como el fondo rápido alcanzará antes que el fondo principal el nuevo valor del fondo, la función $\mathrm{Nv}\,()$ dejará de estar activa en ese momento, y de nuevo el fondo principal volverá a actualizarse con su valor de α_{P} correspondiente.

En las figuras 2.13 (b) y 2.14 (ó 2.15) (b) se muestra el resultado de la estimación del fondo de los mismos píxeles analizados en las figuras 2.13 (a) y 2.14 (a), en este caso empleando las variaciones en el algoritmo de mantenimiento del fondo vistas anteriormente. Podemos ver como, por un lado, en la figura 2.14 (b) el fondo principal se adapta con mucha mayor rapidez a la variación brusca del fondo, tardando únicamente 6 marcos (exactamente desde el marco 91 hasta el 96) en alcanzar el nuevo valor final del fondo.

De igual forma, en la figura 2.13 (b) se puede comprobar cómo el fondo principal no incorpora en ninguna ocasión ninguno de los objetos en movimiento que aparecen en la escena, al mismo tiempo que es capaz de seguir las variaciones graduales del fondo. La mejora producida por el nuevo algoritmo propuesto puede verse con mayor claridad si comparamos las figuras 2.15 (a) y (b).

Cálculo de la varianza

Tal y como se vio en la revisión del estado del arte sobre los distintos métodos estadísticos descritos en la literatura para la detección de movimiento (pág. 31), una de las expresiones más comúnmente empleadas para el cálculo y actualización continua de la varianza es la propuesta por [Stauffer99]:

$$\bar{\sigma}_t^2 = (1-\alpha)\,\bar{\sigma}_{t-1}^2 + \alpha\,(I_t - \bar{\mu}_t)^2 \tag{2.46}$$

También vimos que se han propuesto distintas variantes de la expresión anterior, como por ejemplo las ecuaciones (2.22) [Kanade98] ó (2.23) [McKenna00], aunque la filosofía de funcionamiento de todas ellas es muy similar. Aunque los resultados obtenidos con cualquiera de ellas son en general muy aceptables, en el caso de trabajar con secuencias de vídeo con gran componente de movimiento del fondo, o variaciones importantes de la iluminación, es decir del tipo de secuencias con las que estamos trabajando en esta tesis, las expresiones anteriores suelen presentar problemas importantes. Para comprender este problema, basta con analizar la esperanza del estimador anterior. En presencia de señales sin variación de la iluminación, o al menos aquellas cuya iluminación varíe tan lentamente que la estimación de la media se acerque con bastante precisión al valor de la media real ($\mathrm{E}\{\bar{\mu}_t\} = \mu$), la esperanza de la expresión (2.46) sería:

$$\mathrm{E}\left\{\bar{\sigma}_t^2\right\} = (1-\alpha)\,\mathrm{E}\left\{\bar{\sigma}_{t-1}^2\right\} + \alpha\mathrm{E}\left\{(I_t - \bar{\mu}_t)^2\right\} \tag{2.47}$$

Teniendo en cuenta que por definición $\mathrm{E}\left\{(I_t - \mu)^2\right\} = \sigma^2$, tenemos que:

$$\mathrm{E}\left\{\bar{\sigma}_t^2\right\} \;=\; (1-\alpha)\,\mathrm{E}\left\{\bar{\sigma}_{t-1}^2\right\} + \alpha\sigma^2 = \sigma^2 \tag{2.48}$$

es decir, la ecuación (2.46) es un buen estimador de la varianza de la señal cuando ésta no presenta variaciones de la iluminación significativas. Sin embargo, supongamos una señal donde la iluminación varíe rápidamente, de tal forma que la media estimada difiera de la media real en:

$$\Delta_t = \bar{\mu}_t - \mu \tag{2.49}$$

Por simplicidad, supongamos que el valor de Δ_t es constante, es decir, $\Delta_t = \Delta$. En este caso, la esperanza del segundo término de la expresión (2.46) sería:

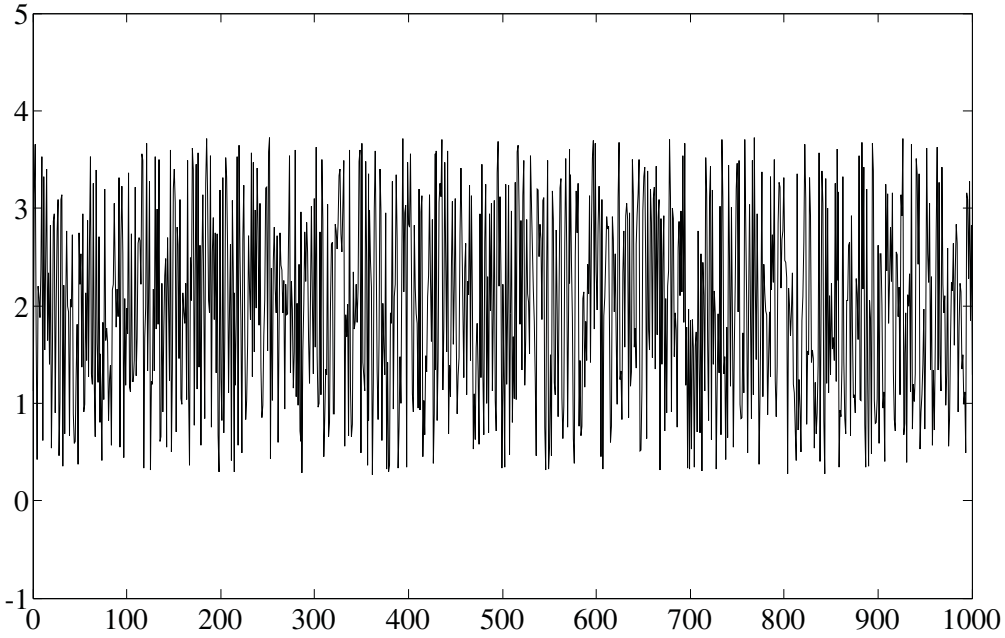


Figura 2.16: Señal estacionaria de media 2 y varianza 1.

$$\begin{aligned} \mathrm{E}\left\{(I_t-\bar{\mu}_t)^2\right\} &= \mathrm{E}\left\{(I_t-(\mu+\Delta))^2\right\} \\ &= \mathrm{E}\left\{I_t^2-2I_t\mu+\mu^2+\Delta^2-2I_t\Delta+2\mu\Delta\right\} \\ &= \mathrm{E}\left\{I_t^2-2I_t\mu+\mu^2\right\}+\Delta^2-2\Delta\mathrm{E}\left\{I_t\right\}+2\mu\Delta \end{aligned} \tag{2.50}$$

El primer término de la expresión anterior se correspondería con la varianza de la señal (σ^2), mientras que los dos últimos términos se anulan entre sí, dando como resultado que la esperanza del estimador anterior sería:

$$\mathrm{E}\left\{(I_t-\bar{\mu}_t)^2\right\}=\sigma^2+\Delta^2 \tag{2.51}$$

es decir, el estimador se encuentra sesgado por el término Δ^2, siendo por tanto el sesgo mayor cuanto mayor sea la diferencia entre la media estimada $\bar{\mu}_t$ y la media real μ. Esto conlleva principalmente que la varianza estimada sea siempre superior a la varianza real, y sobre todo, dado que el valor de Δ no es conocido, y generalmente variable a lo largo de la secuencia de vídeo, no sea posible emplear dicha estimación para nuestro propósito de clasificación de píxeles. Un ejemplo puede ayudarnos a verlo más claramente. Para ello se simula una señal aleatoria, de media 2 y varianza 1, como se muestra en la figura 2.16.

En la figura 2.17 se muestra la evolución de la estimación de la media y la varianza para la señal anterior. Para el caso de la media, simplemente se ha utilizado la expresión básica de cálculo continuo de la media, tal y como fue descrito en la página 31 con la ecuación (2.20). Sin embargo, para el caso de la varianza se han utilizado dos métodos

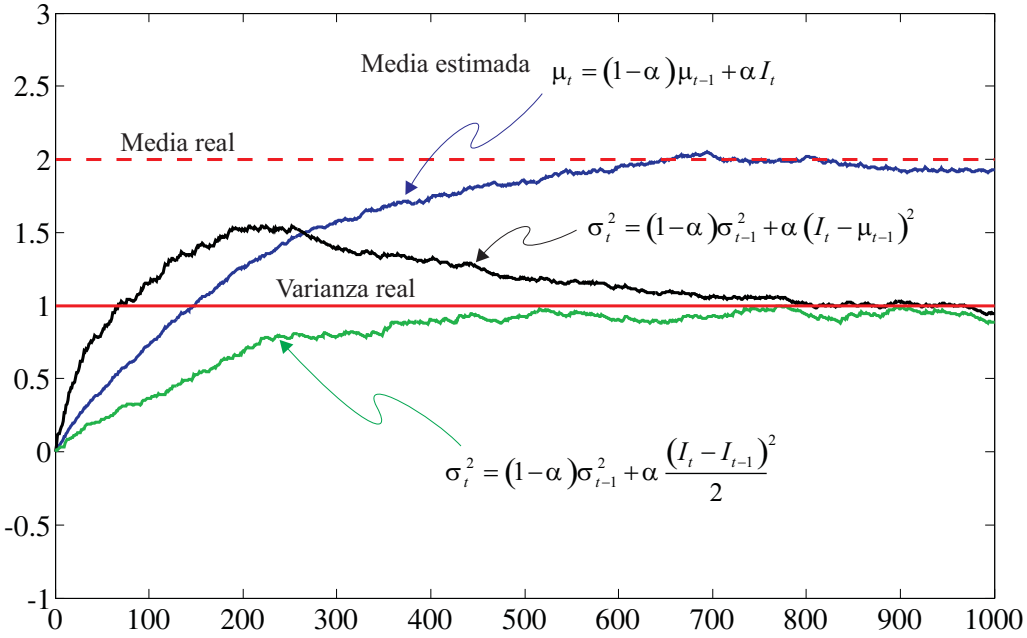


Figura 2.17: Estimación de la media y la varianza de la señal aleatoria mostrada en la figura 2.16.

distintos. El primero de ellos, representado en negro, sería la estimación siguiendo la expresión propuesta por [Stauffer99], y comentada anteriormente en la ecuación (2.46). Obviando el transitorio inicial, que es debido a la lentitud por parte de la estimación de la media en alcanzar su valor final, y que termina cuando la media estimada se aproxima a su valor teórico, podemos comprobar como la varianza estimada se aproxima perfectamente a su valor esperado. En este punto debemos indicar que se han elegido deliberadamente valores para el parámetro α muy pequeños (evolución lenta de las señales estimadas) con el único propósito de hacer más clara esta demostración. Los resultados que obtendríamos con valores de α más *normales* (es decir, más grandes) serían los mismos, pero a la vez serían más difíciles de poder ser visualizados en una gráfica como la representada en la figura anterior.

Sin embargo, cuando la iluminación de la escena no es fija, sino que sufre variaciones más o menos importantes y a una velocidad tal que la estimación de la media no puede seguir completamente dichas variaciones, tal y como se ha demostrado anteriormente, la estimación de la varianza puede devolver resultandos poco precisos. Para visualizar este problema, vamos a simular de nuevo una señal aleatoria de varianza 1, y cuya media tome el valor 0 durante las primeras muestras, y que cambie brúscamente a valer 2 a partir de la muestra 1000. En la figura 2.18 podemos ver dicha señal. Esta señal constituye un modelo aproximado de cambio de iluminación típico de ciertos escenarios de videovigilancia.

En este caso, si empleamos las mismas expresiones para el cálculo de la media y la varianza, obtendríamos las respuestas mostradas en la figura 2.19. Si nos fijamos en el

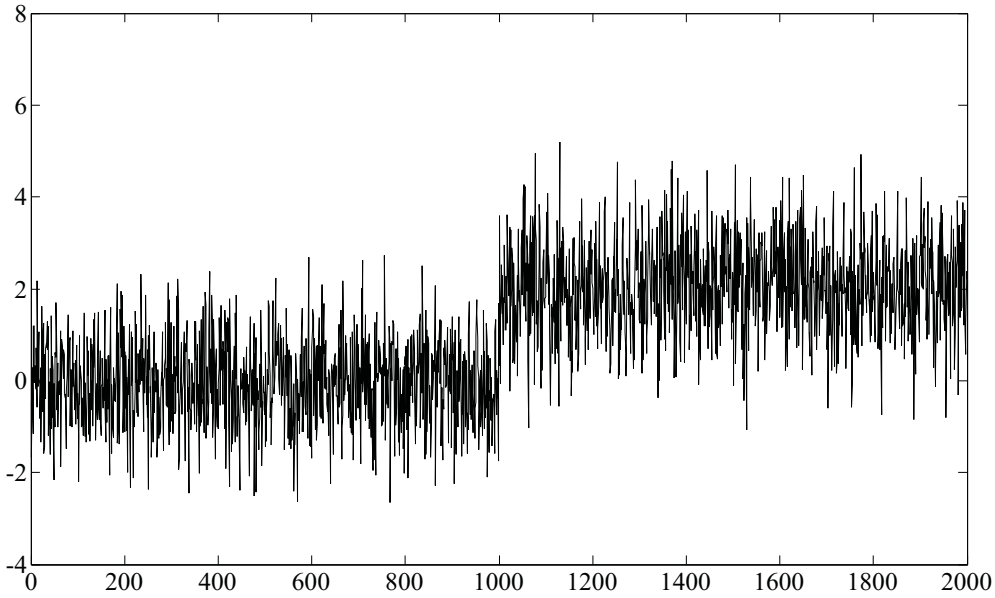


Figura 2.18: Señal no estacionaria de varianza 1.

valor de la varianza estimada (línea negra), podemos comprobar como, mientras no se produce ningún cambio en la iluminación, ésta es bantante precisa. Sin embargo, a partir del instante del cambio de iluminación, la varianza estimada difiere en casi un 100 % de su valor original, y esta discrepancia se mantiene durante un periodo de tiempo considerable.

En conclusión, el estudio anterior nos ha servido para comprobar que, si bien las expresiones de estimación continua de la varianza de una señal más comúnmente empleadas en trabajos de videovigilancia, como pueden ser las propuestas por [Stauffer99] (eq. 2.21), [Kanade98] (eq. 2.22) ó [McKenna00] (eq. 2.23), ofrecen resultados bastante aceptables cuando la escena es estable, cuando ésta deja de serlo, debido a fluctuaciones en la iluminación, sombras, etc, las expresiones anteriores ofrecen resultados poco precisos.

Con la idea de intentar mejorar la precisión en la estimación de la varianza de cada uno de los píxeles, en esta tesis propondremos el cálculo de dicha varianza mediante una expresión alternativa. La característica principal de esta expresión es que debe evitar el uso de la media estimada, ya que, como vimos en la sección anterior, esta estimación puede desviarse del valor real cuando varía a una velocidad suficientemente rápida. Por tanto, en lugar de comparar la imagen actual con la media estimada, proponemos la comparación con la imagen anterior, de acuerdo a:

$$\bar{\sigma}_t^2 = (1-\alpha)\,\bar{\sigma}_{t-1}^2 + \frac{\alpha}{2}\,(I_t - I_{t-1})^2 \tag{2.52}$$

Obviando por el momento la constante 1/2 que multiplica a α en el segundo término de la ecuación, podemos comprobar la validez de esta expresión, calculando la esperanza de dicho estimador, igual que hicimos con la expresión anterior:

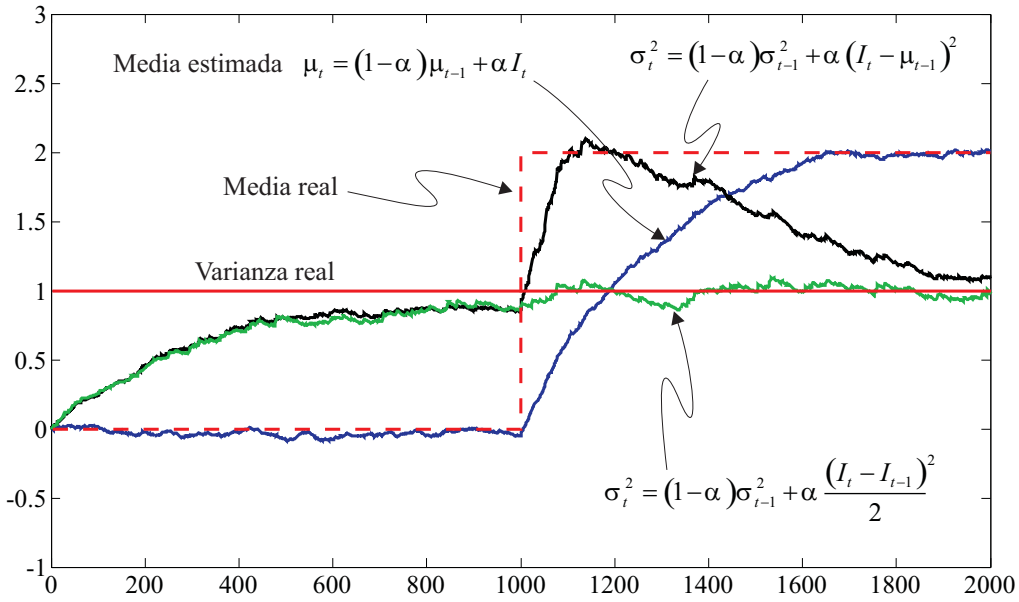


Figura 2.19: Estimación de la media y la varianza de la señal aleatoria mostrada en la figura 2.18.

$$\mathrm{E}\left\{(I_t - I_{t-1})^2\right\} = \mathrm{E}\left\{I_t^2\right\} + \mathrm{E}\left\{I_{t-1}^2\right\} - 2\mathrm{E}\left\{I_t I_{t-1}\right\} \tag{2.53}$$

Para el último miembro de la ecuación, cuando las muestras están incorreladas entre sí, tendríamos:

$$\mathrm{E}\left\{I_t^2 I_{t-1}^2\right\} = \mathrm{E}\left\{I_t\right\}\mathrm{E}\left\{I_{t-1}\right\} = \mu^2 \tag{2.54}$$

por lo que la esperanza del estimador anterior sería:

$$\mathrm{E}\left\{(I_t - I_{t-1})^2\right\} = 2\left(m_2 - \mu^2\right) = 2\sigma^2 \tag{2.55}$$

lo cual demuestra que, en presencia de señales con ruido blanco, la expresión (2.52) permite estimar la varianza de la señal. Como no aparece la media estimada $\bar{\mu}_t$, queda claro que, para el tipo de secuencias de vídeo que estamos analizando en esta tesis, la expresión (2.52) nos va a reportar, en general, resultados más precisos que la expresión (2.46).

En las figuras 2.17 y 2.19 podemos comprobar los resultados obtenidos con el método propuesto y compararlos con los obtenidos con el método anterior. Cuando el fondo no presenta variaciones de la iluminación importantes, como ocurría en la figura 2.17, podemos ver como tanto el método que emplea la expresión (2.46) como el que emplea la expresión (2.52) convergen al valor real de la varianza de la señal. Sin embargo, en la figura 2.19 vemos que cuando la iluminación sufre un cambio de iluminación brusco, la varianza estimada con el método propuesto se comporta de forma más estable.

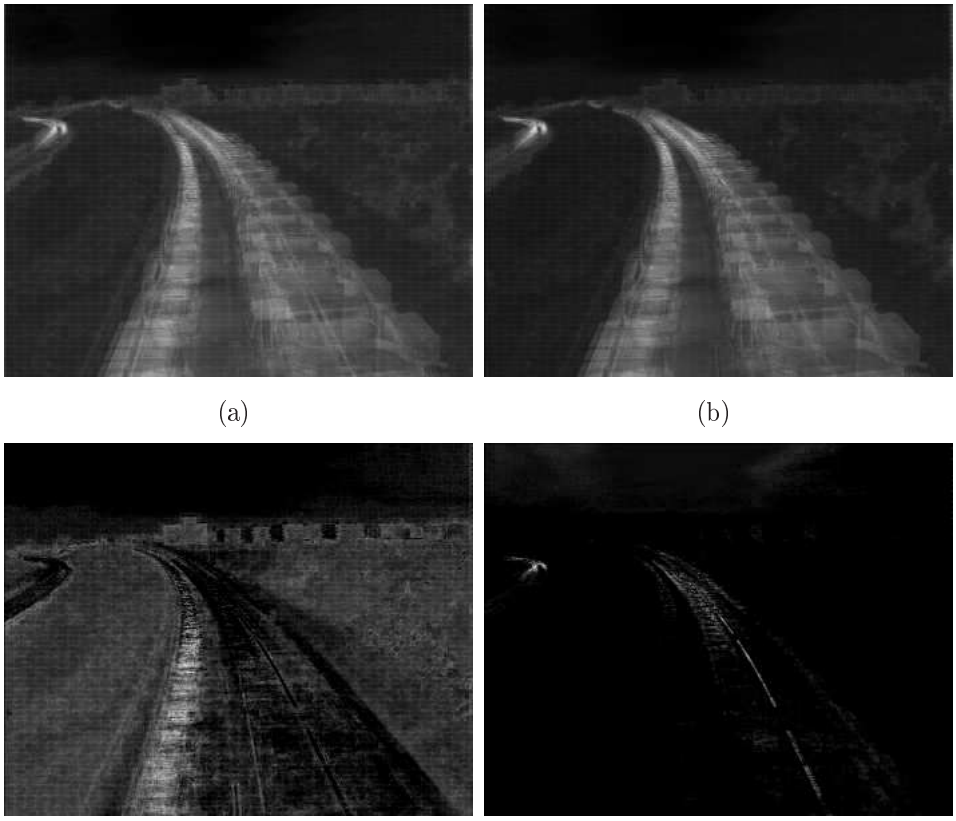
(a) (b)

Figura 2.20: (a): Imagen de la secuencia de vídeo empleada para ilustrar el cálculo de la varianza. (b): Media estimada.

También es posible comprobar la validez de los resultados obtenidos del método anterior aplicado a entornos reales de videovigilancia. En la figura 2.20 (a) se muestra una imagen de una secuencia de videovigilancia, correspondiente a una carretera. Sobre esta secuencia, podemos realizar la estimación de la media, empleando la expresión (2.20). La figura 2.20 (b) muestra dicha estimación para un instante determinado. Finalmente, si realizamos la estimación de la varianza mediante las expresiones (2.46) y (2.52), obtenemos los resultados mostrados en las figuras 2.21 (a) y (b) respectivamente. Ambos resultados corresponden a un instante posterior a un cambio de iluminación que ocurre en el escenario debido al paso de una nube.

Si observamos detenidamente ambas figuras, se puede apreciar ligéramente como, en la parte correspondiente a la zona arbolada, principalmente la zona a la derecha de la carretera, la varianza en esta zona es mayor cuando ésta ha sido estimada mediante la expresión (2.46). Para poder apreciar con mayor precisión dicho detalle, las figuras 2.21 (c) y (d) muestran las diferencias entre ambas figuras. En concreto, en (c) se muestra la diferencia entre las figuras (a) menos (b) cuando dicha diferencia es positiva, y toma el valor 0 (color negro) cuando la diferencia es negativa. En (d) se muestra la operación contraria, es decir (b) menos (a) cuando es positiva, y cero cuando resulta negativa. Aquí se puede apreciar con mayor claridad como, para la zona arbolada descrita anteriormente, la varianza es siempre mayor cuando ésta es calculada mediante (2.46). Para el resto de la escena, es decir aquellas partes que tienen un comportamiento estático (parte superior de la imagen, correspondiente al cielo), o impulsivo (parte correspondiente a la carretera), los resultados finales no son tan importantes, ya que para nuestro propósito de clasificación del fondo, los errores en la estimación de la varianza no son tan importantes.

Antes de concluir esta sección, deberíamos también elegir el valor del parámetro de actualización α a emplear para la estimación y mantenimiento de la varianza. Aunque podríamos optar por elegir alguno de los valores empleados en el cálculo de la media,



(a) (b)

(c) (d)

Figura 2.21: (a,b): Varianza estimada de la secuencia de la figura 2.20, utilizando la expresión: (a): Expresión (2.46), (b): Expresión (2.52). En ambas visualiaciones, el color negro corresponde a una varianza de valor 0, mientras que el color blanco a 1000. (c): Diferencia entre las figuras (a) menos (b) cuando la diferencia es positiva, 0 si negativa. (d): Diferencia entre las figuras (b) menos (a) cuando la diferencia es positiva, 0 si negativa.

estos no serían los más recomendables, debido a la siguiente razón. Como se dedujo en la sección anterior, el valor de α debe ser lo suficientemente grande como para que pueda seguir, en cierto modo, las variaciones *rápidas* del fondo, ya que la velocidad de variación del fondo en general será tal que haga esto necesario. Sin embargo, aunque el fondo varíe más o menos rápido, es de esperar que la varianza no lo haga así. Es decir, si en una parte de la escena el nivel de ruido es elevado (por ejemplo, por que hay un árbol en movimiento) el ruido será elevado siempre, al menos hasta que el árbol deje de moverse, y viceversa, por lo que un valor del parámetro de actualización α rápido para la varianza no es necesario. Con todo esto, el valor elegido para dicho parámetro fue $\alpha = 0{,}005$.

Periodo de entrenamiento

En las figuras 2.17 y 2.19 podemos comprobar que, independientemente del método elegido para la estimación de los estadísticos, existe un periodo transitorio desde que el sistema entra en funcionamiento hasta que éste se estabiliza. Por ejemplo, en la figura 2.17 podemos ver que este periodo dura unas 500 muestras. No hay que olvidar que en este ejemplo, el parámetro de actualización se eligió deliberadamente lento para la mejor visualización del efecto analizado. En general, en los sistemas de videovigilancia, la velocidad de actualización es mucho más rápida, pero independientemente de esto, este periodo transitorio siempre existirá.

En numerosas aplicaciones, este periodo no es aceptable, y se necesita que el sistema alcance su regimen estable lo antes posible. Para solucionar este problema, la solución adoptada consiste en establecer un periodo de entrenamiento, al comienzo de la ejecución del algoritmo, en el que se calculen los estadísticos con el método *multi-muestra*, que nos servirá para establecer un valor inicial para los parámetros a estimar. Si este periodo es suficientemente grande, los estadísticos estimados estarán más cerca de sus valores reales, por lo que se tardará bastante menos en alcanzar la estabilidad del sistema. Por ejemplo, en la figura 2.22 se muestra una comparación entre la estimación de la media y la varianza de una señal de media 2 y varianza 1, por un lado sin entrenamiento, partiendo de un valor inicial igual a 0, y por el otro lado con entrenamiento, utilizando un periodo de 50 muestras. Podemos ver como, para la estimación con entrenamiento, aun a pesar de empezar la estimación en la muestra 50 (ya que durante las anteriores se han estado calculando los estadísticos) alcanza el régimen estable mucho antes.

El problema principal consiste en establecer la duración de ese periodo de entrenamiento. Una longitud excesivamente larga trae como consecuencia dos problemas. El primero de ellos es bastante evidente. Si hacemos el periodo de entrenamiento más largo que el propio periodo que tarda un sistema sin entrenamiento en alcanzar la estabilidad, hemos generado un sistema más lento, en lugar de ser más rápido. Por otro lado, la inestabilidad del fondo puede hacer que un cálculo global sobre un número elevado de muestras sea más inexacto que el cálculo local sobre un número reducido de muestras. Por ello, queda claro que el periodo de entrenamiento debe ser muy reducido. Incluso para el cálculo de la

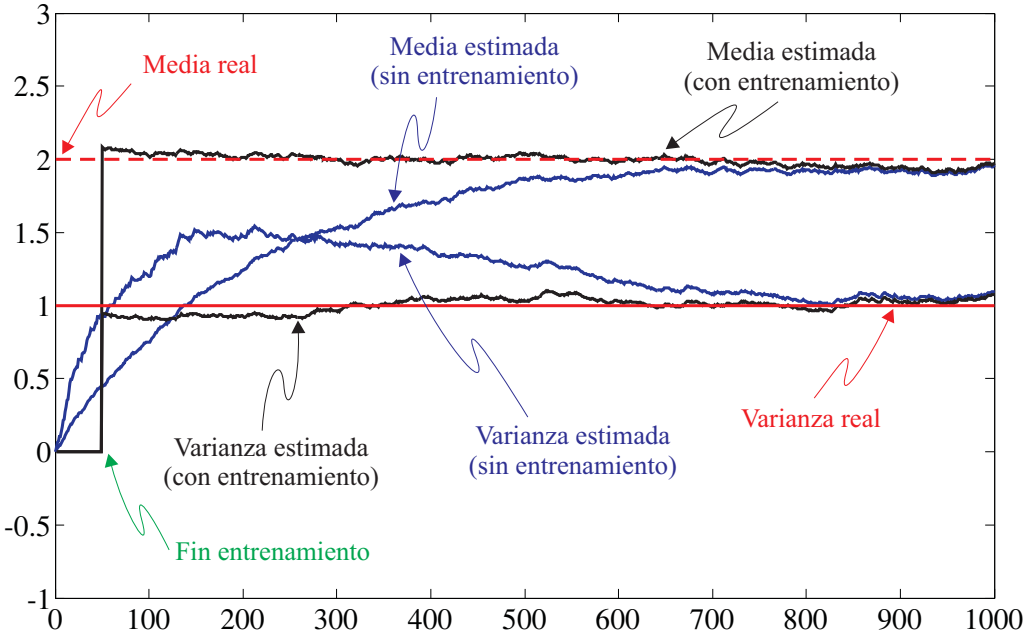


Figura 2.22: Reducción del periodo transitorio mediante el establecimiento de un periodo de entrenamiento de 50 muestras.

media, se podría evitar el cálculo de la media y tomar como primera estimación, el valor de la primera muestra. No obstante, podemos comprobar de manera experimental que un número superior, de aproximadamente 15 imágenes, es necesario para obtener unos valores más adecuados.

Clasificación del fondo

Una vez estabilizados los estadísticos de la secuencia de vídeo, llega el turno de la clasificación de cada píxel de la imagen en uno de los cuatro grupos definidos anteriormente. Según la descripción de dichos grupos, la distinción entre los distintos grupos la vamos a realizar como sigue. La distinción entre píxeles pertenecientes al fondo dinámico y al fondo estático podrá realizarse a partir de la umbralización de la varianza. Valores por encima de un determinado umbral corresponderán al fondo dinámico, mientras que aquellos que estén por debajo corresponderán al fondo estático. En este punto no deberíamos tener en cuenta los píxeles impulsivos, ya que en general la varianza se va a encontrar cerca del umbral anterior. Efectivamente, en el cálculo de la varianza de estos píxeles, la mayor contribución va a ser la debida al fondo estático que se visualiza cuando no hay ningún objeto atravesando dicha zona, pero la pequeña contribución debida a los objetos que atraviesan dicha zona hará que la varianza se eleve, en ocasiones por encima del umbral.

Para evitar errores en estas zonas, será necesario discriminar en primer lugar los píxeles impulsivos de los estáticos y dinámicos. Para ello, lo que haremos será contar el número de veces que un píxel ha superado un porcentaje determinado de su varianza, pongamos

por ejemplo el 200 %. Si éste es superado sólo una vez en un tiempo determinado, lo más seguro es que haya sido debido más bien a un objeto de primer plano desplazándose por una zona no impulsiva. Sin embargo, si ese número es elevado, estaría indicando que ese píxel corresponde a una zona impulsiva.

Igual que ocurre con la estimación de estadísticos, tales como la media o la varianza, si necesitamos establecer un contador, volvemos a tener dos opciones, es decir, el *procesado multi-muestra* y el procesado *por marco* (pag. 21). Siguiendo el esquema de procesado multi-muestra se plantea el mismo problema de necesidades de almacenamiento. Por ejemplo si quisiéramos mantener el recuento de cuantos impulsos han ocurrido en el transcurso de las últimas N imágenes, necesitaríamos disponer de un array de N matrices donde almacenaríamos en cada una de ellas los eventos ocurridos (impulso o no impulso en este caso) para cada píxel de las últimas N imágenes. A la llegada de una nueva imagen, le añadiríamos el evento actual (+1 si hay impulso, 0 si no), le restaríamos el evento N (-1 si hubo impulso, 0 si no), y desplazamos todas las matrices una unidad, eliminado la matriz N para dejar hueco a la matriz actual. Cuando N es suficientemente grande, y en general lo va a ser, porque necesitamos tener un registro de los impulsos ocurridos en un intervalo grande de tiempo para poder discriminar entre movimientos aislados y zonas con movimiento continuo, la necesidad de almacenamiento será prohibitivamente grande.

Por tanto, la solución, al igual que ocurría con el cálculo y mantenimiento de los estadísticos principales de la escena, es recurrir al procesado por marco, teniendo en cuenta en este caso que el método será ligeramente distinto. Si para un píxel determinado de la imagen construimos una señal z_t, que toma el valor 1 cuando en dicho píxel ocurrió un impulso, y 0 si no, la expresión que permitiría realizar la cuenta del número de impulsos ocurridos en un determinado intervalo de tiempo sería:

$$C_t = \frac{1}{N} \sum_{\tau=t}^{t-N+1} z_\tau \tag{2.56}$$

Desde el punto de vista del procesado de señal, la función anterior se corresponde con un filtro FIR donde la respuesta al impulso sería una señal constante, de valor $1/N$ y longitud N. Si queremos aproximar la función anterior mediante la técnica de procesado por marco, de acuerdo a la siguiente expresión:

$$C'_t = \begin{cases} (1-\alpha)\, C'_{t-1} + \alpha k & \text{si} \quad z_t = 1 \text{ (impulso)} \\ (1-\alpha)\, C'_{t-1} & \text{si} \quad z_t = 0 \text{ (no impulso)} \end{cases} \tag{2.57}$$

que puede expresarse de forma más compacta como $C'_t = (1-\alpha)\, C'_{t-1} + \alpha k z_t$. El valor de k se corresponde con el valor al que tendería la señal C'_t si todas las muestras de z_t fuesen 1. Como muestro interés se centra en aproximar esta función al contador de impulsos visto anteriormente, k debe valer 1, ya que C_t también tiende a 1 cuando todas las muestras de z_t tienden a ser 1. El parámetro α, que como ya hemos visto hasta ahora, tiene que ver con la velocidad de variación de la señal, en esta ocasión vamos a encontrar que también tiene relación con el valor de N de la expresión anterior.

Para poder encontrar la relación entre α y el intervalo de tiempo que consideramos para el recuento de impulsos, podemos ver los sistemas anteriores como sendos filtros lineales, de los cuales vamos a analizar su respuesta en frecuencia. Para el primero de ellos (2.56), podemos comprobar que se trata de un filtro FIR, con una respuesta al impulso de N muestras, todas de tamaño $1/N$. La respuesta en frecuencia de este filtro sería del tipo:

$$|H_1(\Omega)| = \frac{\sin\left(\Omega\frac{N}{2}\right)}{N\sin\left(\frac{\Omega}{2}\right)} \tag{2.58}$$

teniendo su primer paso por cero a la pulsación discreta $\Omega_0 = 2\pi/N$. Para el segundo filtro (2.57), su ecuación en diferencias sería:

$$y[n] = (1-\alpha)\, y[n-1] + \alpha x[n] \tag{2.59}$$

siendo $x[n]$ la entrada e $y[n]$ su salida. Su función de transferencia puede expresarse como:

$$H_2(\Omega) = \frac{\alpha}{1-(1-\alpha)\,\mathrm{e}^{-\mathrm{j}\Omega}} \tag{2.60}$$

Del análisis de ambas funciones de transferencia, se puede comprobar que ambos constituyen filtros paso bajo. En la figura 2.23 podemos ver la representación del módulo de la respuesta en frecuencia de ambos filtros para un valor concreto de N y α. Aunque la respuesta en frecuencia de ambos filtros es bien distinta, para nuestro propósito de sustituir el método de cómputo de impulsos de la ecuación (2.56) por uno más sencillo, según la ecuación (2.57), es más que suficiente. Para establecer una relación entre α y N, vamos a diseñar ambos filtros de tal forma que coincida la frecuencia a la que se produce el primer cruce por cero del primero, $\Omega_0 = \frac{2\pi}{N}$, con la frecuencia de corte del segundo, Ω_c, tomando como frecuencia de corte aquella a la cual el módulo de la respuesta en frecuencia se hace $1/5$ de su valor máximo.

En primer lugar, vamos a establecer la relación entre el parámetro α y la frecuencia de corte del segundo filtro como:

$$|H_2(\Omega_\mathrm{c})| = \frac{\alpha}{|1-(1-\alpha)\,\mathrm{e}^{\mathrm{j}\Omega_\mathrm{c}}|} = \frac{1}{5} \tag{2.61}$$

de donde obtenemos que el valor de α en función de la frecuencia del corte del filtro sería:

$$\alpha = \rho\left(\cos\Omega_\mathrm{c} - 1 + \sqrt{49 - 50\cos\Omega_\mathrm{c} + \cos^2\Omega_\mathrm{c}}\right) \tag{2.62}$$

siendo $\rho = 0{,}041667$. Teniendo en cuenta que, para el primer filtro, $N = \frac{2\pi}{\Omega_0}$, en la figura 2.24 podemos ver la relación existente entre α y N. A partir de las expresiones anteriores, podemos comprobar que para valores de N por encima de 50, que por otro lado será lo normal en nuestro sistema de recuento de impulsos, el valor de α aproximadamente

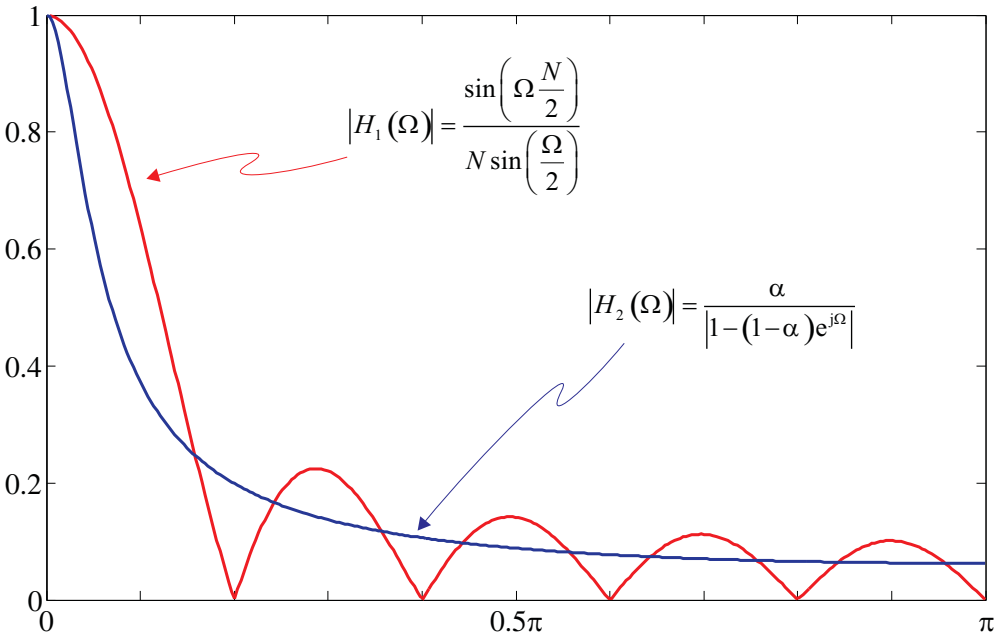


Figura 2.23: Respuesta en frecuencia de los filtros descritos por las ecuaciones 2.56 ($N = 10$) y 2.57 ($\alpha = 0{,}118$)

es inversamente proporcional al valor de N, de tal forma que podemos establecer una relación definitiva entre ambos parámetros de acuerdo a:

$$\alpha = \frac{1{,}28123}{N} \tag{2.63}$$

Por último, como comprobación del desarrollo teórico expuesto, en la figura 2.25 podemos ver un ejemplo práctico. Para ello, se ha generado una señal aleatoria z_t, tal y como fue definida anteriormente, que simula la llegada de impulsos al sistema, con una probabilidad de un 20 %. En la figura se muestra la salida de ambos contadores, teniendo como parámetros $N = 10$ para el primer contador y, de acuerdo con (2.62) ya que $N \ll 50$ en este ejemplo, $\alpha = 0{,}118$ para el segundo contador. A la vista de esta gráfica podemos ver como la respuesta de ambos sistemas es muy similar, y por tanto, la opción propuesta para el cálculo de impulsos recibidos mediante actualización continua es más que suficiente para nuestro propósito de detección de zonas impulsivas de la imagen.

Por tanto, del análisis anterior, podemos determinar que la expresión (2.57) nos permite mantener una estimación del número de impulsos producidos en cada píxel de la imagen. Finalmente, para poder determinar si un píxel pertenece al grupo impulsivo o no, es necesario umbralizar la función anterior. Aunque este umbral debe ser parametrizable, su valor debería ajustarse de acuerdo al siguiente razonamiento. Un valor de C'_t cercano a 0 indica que no se ha producido prácticamente ningún impulso en ese píxel. Un valor de 0,5 indicaría que la mitad de las veces se han detectado impulsos, lo cual no sería muy lógico que ocurriera, pero puede darnos una idea de por dónde debe situarse el umbral

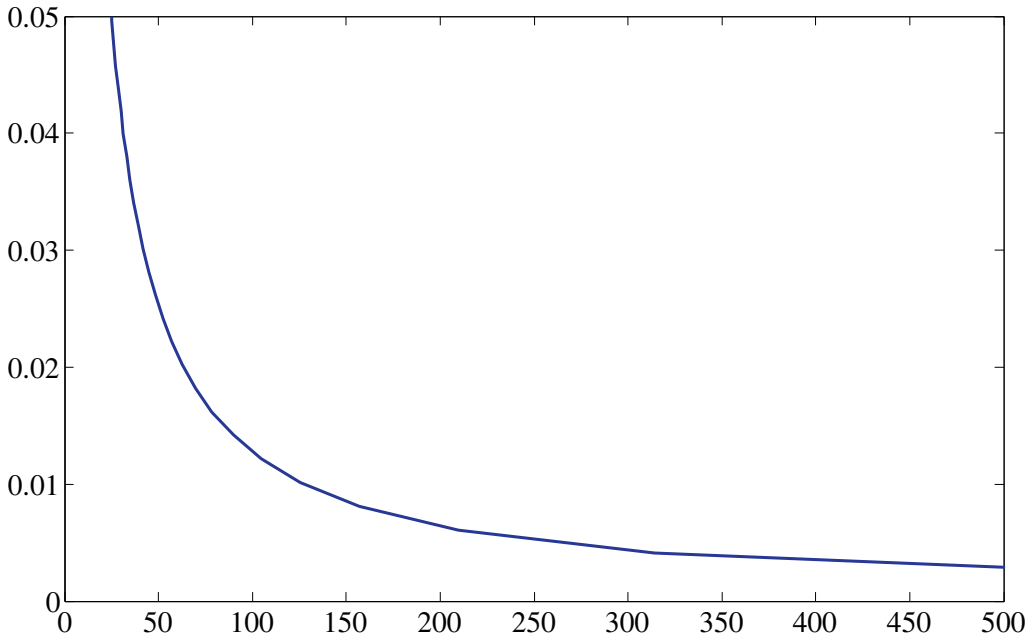


Figura 2.24: Relación entre el valor de α en la expresión 2.57 y el número de muestras N tomadas en cuenta en la expresión 2.56 para el recuento del número de impulsos.

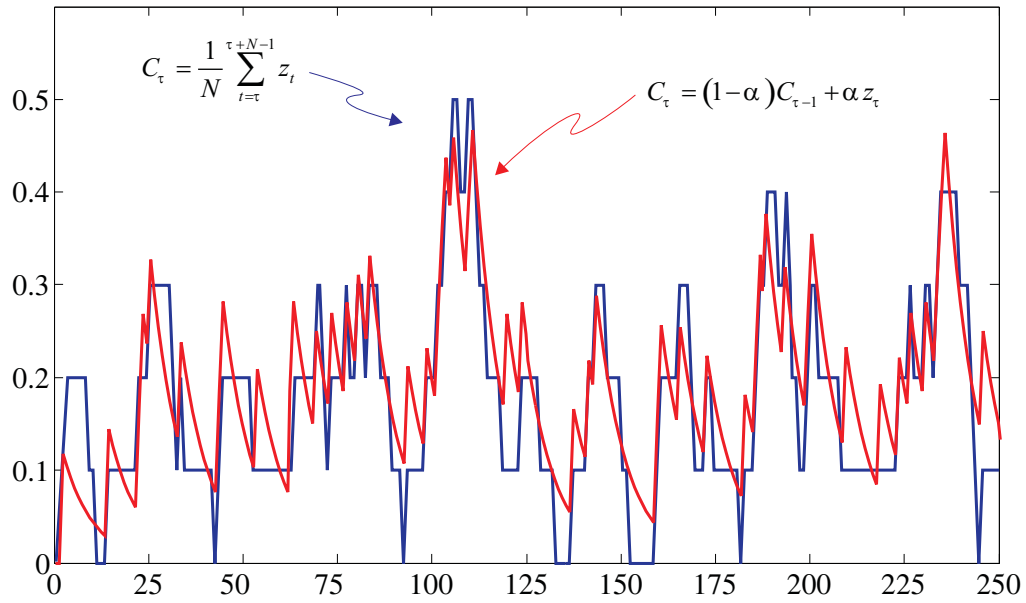


Figura 2.25: Comparación entre contadores de impulsos implementados según la expresión (2.56) con N igual a 10 (linea azul), y la expresión (2.57) con $\alpha = 0{,}118$ (linea roja).

para esta función. Por ejemplo, si el valor baja a 0,1 indica que el 10 % del tiempo aparecen objetos que provocan impulsos en la escena. En las secuencias analizadas en esta tesis, se han tomado valores inferiores, del orden de 0,01 a 0,05.

De esta forma, una vez clasificados los píxeles impulsivos, los restantes píxeles deberán ser divididos entre los que pertenecen al fondo estático y los que pertenecen al fondo dinámico y, como ya se ha comentado anteriormente, esta división se realizará umbralizando la varianza de dichos píxeles. El cálculo del umbral que emplearemos para dicha distinción está descrito en la siguiente sección.

Por último, según se describió anteriormente, los píxeles cambiantes son más bien píxeles del fondo estático que bruscamente cambian a un nuevo valor, y permanecen en dicho valor un tiempo considerable. En este caso, el mejor indicador que nos permita detectar este tipo de eventos es la diferencia entre la imagen actual y el fondo rápido. Cuando el fondo es estático o dinámico y no ha cambiado bruscamente, esta diferencia no será mayor que un porcentaje determinado de la varianza de dicho píxel. Cuando el fondo es impulsivo, la presencia de un nuevo objeto hará que la diferencia entre la imagen actual y el fondo rápido sí sea mayor, pero como se trata de un objeto en movimiento, el número de veces que esto ocurrirá será pequeño. Finalmente, cuando se trate de un cambio brusco en un píxel estático, la diferencia anterior será mayor que un porcentaje de la varianza durante un número mayor de veces. Si recordamos que el parámetro α_{R} fue elegido de tal forma que haga que en estos casos el fondo rápido converja al nuevo valor del fondo en aproximadamente 10 a 15 muestras, tendremos que la diferencia anterior estará activa durante unas 10 muestras, tiempo más que suficiente para que el sistema se dé cuenta de que el fondo estático ha cambiado.

En este caso, la implementación de este contador es bastante más sencilla, ya que lo que realmente necesitamos es contar el número de veces consecutivas que la diferencia anterior está activa. Por tanto, si en un momento determinado, la diferencia se encuentra activa, se aumenta uno el contador de ese píxel. En el momento en el que la diferencia se desactiva, se resetea a 0 el contador. Si en algún momento el contador supera un determinado valor, el sistema determina que dicho píxel pertenece al fondo cambiante. Es necesario recordar aquí de nuevo que el fondo cambiante no es una categoría en sí, sino más bien un indicador que advierte al sistema de que esa zona es susceptible de volver a cambiar nuevamente.

Antes de terminar con la clasificación, es necesario tener en cuenta un problema típico en sistemas de decisión mediante umbrales, y es el hecho de considerar la clasificación en sí cuando el valor de la señal se encuentra muy próximo al umbral. En este caso, ligeros cambios en el valor de la señal por encima y por debajo del umbral hará que el valor de la clasificación alterne entre un estado y otro de manera continua. En nuestro caso, el estado de un píxel, y más generalmente el estado de una zona concreta de la imagen no va a cambiar frecuentemente. Es decir, por ejemplo, una zona que pertenezca al fondo dinámico se supone que seguirá comportándose como fondo dinámico durante bastante tiempo. Sin embargo, si el valor de la varianza en dicha zona se encuentra muy próxima al umbral,

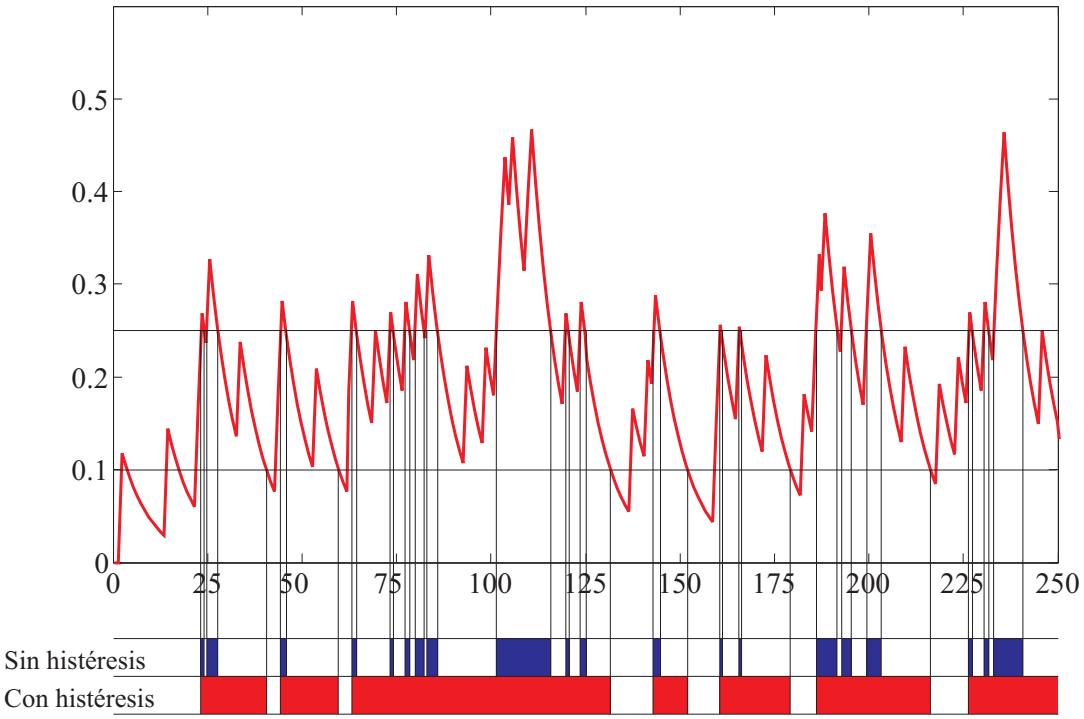


Figura 2.26: Comparación entre umbralización sin histéresis y con histéresis

es posible que la clasificación de dicha zona se encuentre continuamente alternando entre fondo dinámico y digamos por ejemplo, fondo estático.

Para evitar estas alternancias entre clases, y conseguir que la clasificación permanezca estable durante un tiempo considerable, la umbralización, tanto para el contador de impulsos analizado en la figura 2.25, como para la clasificación entre fondo dinámico y estático, se realiza mediante la técnica de umbralizado con histéresis. Este técnica consiste en tener dos umbrales, pongamos α_1 y α_2, con $\alpha_1 > \alpha_2$. Supongamos que para que la clasificación cambie de un estado a otro, el valor de la señal debe ser superior a α_1. Entonces, para que vuelva a cambiar al otro estado, no basta con que la señal sea inferior a α_1, sino que no cambiará hasta que descienda del valor de α_2.

En la figura 2.26 podemos ver un ejemplo de su funcionamiento. En esta figura se muestra, en la parte superior, la señal contador analizada en la figura 2.25. En la parte inferior se muestra el resultado de la clasificación mediante una umbralización sin histéresis (resultado superior) y con histéresis (resultado inferior). Claramente puede comprobarse como la frecuencia de transición entre estados es menor cuando se emplea la umbralización con histéresis, y además el tiempo que permanece en cada estado es mucho mayor.

Aunque lo anterior es una simulación de una variable generada aleatoriamente, cuando está técnica se aplica a imágenes reales dentro del algoritmo de clasificación de fondo implementado, se consigue que aquellos píxeles cuyos valores se encuentran muy cerca de

los umbrales no permanezcan alternando entre distintos estados continuamente, sino que permanecen durante un tiempo suficientemente largo.

Umbralización automática

Como ocurre en un gran número de algoritmos, la parte más crítica es la elección del valor del umbral. Para la tarea de clasificación entre fondo dinámico y fondo estático, el umbral debe estar por encima del nivel de ruido de los píxeles estáticos, y al mismo tiempo ser inferior a la desviación típica que presentan los píxeles del fondo dinámico, y por supuesto, de los impulsos debidos al movimiento continuo del fondo. Dicho umbral puede ser determinado manualmente, como suele hacerse en muchos de los algoritmos de detección de movimiento, o mediante técnicas de umbralizado automático. Una de ellas es la propuesta por [Rosin95], pag. 26, donde se realiza la suposición de que menos de la mitad de los píxeles de la imagen estarán en movimiento. En nuestro caso, no queremos dejar tan clara esa suposición, ya que, aparte de los objetos en movimiento del primer plano, debemos considerar también el movimiento continuo del fondo dinámico e impulsivo.

Para la implementación del algoritmo de cálculo automático del umbral vamos a considerar la suposición de que, si bien en la misma escena van a existir píxeles dinámicos y píxeles estáticos, las varianzas de los píxeles estáticos van a ser muy parecidas entre sí, mientras que las varianzas de los píxeles dinámicos van a ser muy diferentes. Si calculamos el histograma de la varianza, o más bien, de la desviación típica, podremos comprobar, si se cumple la suposición anterior, que dicho histograma presentará un máximo a la altura del nivel de ruido de la cámara.

Por ejemplo, sobre una secuencia de vídeo perteneciente a una escena exterior con movimiento del fondo, como puede ser la mostrada en la figura 2.6, calculamos su varianza, tal y como fue mostrado anteriormente en la figura 2.7. Si sobre esta matriz calculamos su histograma, obtendríamos la gráfica mostrada en la figura 2.27. Claramente, podemos comprobar como la posición del máximo del histograma, que toma un valor alrededor de 4, se corresponde con el nivel de ruido de la cámara, mientras que el ruido debido al movimiento del fondo, que se corresponde con varianzas más grandes, queda distribuido a lo largo del eje de abscisas, y por tanto, el histograma nunca va a tomar un valor tal elevado como lo toma para el ruido debido a la cámara.

En escenas sin una gran proporción de fondo dinámico el análisis es mucho más sencillo. De todas formas puede verse un ejemplo similar al anterior en estas condiciones. Por ejemplo, para una secuencia de interior, como puede ser la mostrada en la figura 2.28, obtendríamos la desviación típica de la figura 2.29, donde la normalización para la visualización se ha realizado entre 0 y 80. El histograma de esta nueva matriz sería el que puede observarse en la figura 2.30. En este caso, queda más claramente reflejado, si cabe, cual es el nivel de ruido de la cámara, donde podemos comprobar, además, que existe muy poco fondo dinámico, ya que a partir de un valor de la desviación típica de aproximadamente 15, el valor del histograma es prácticamente 0. Además, estos dos ejemplos demuestran

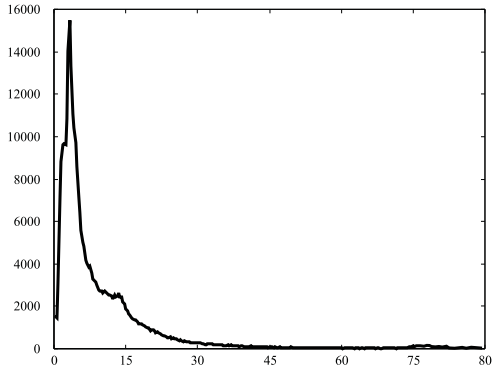


Figura 2.27: Histograma de la matriz de varianzas mostrada en la figura 2.7

como es posible el ajuste automático del umbral, ya que para ambos casos el donde se localiza el máximo del histograma es distinto, tomando para la primera escena un valor aproximado de 3,9, mientras que para la segunda es de 1,3.

Por último, la elección del umbral que permite la clasificación entre píxeles pertenecientes al fondo dinámico y pertenecientes al fondo estático, es tan sencillo como tomar un valor proporcional y ligeramente superior al máximo obtenido en el cálculo del histograma. En nuestro caso, hemos decidido tomar de forma empírica como umbral un valor igual a 3 veces el máximo del histograma. Por ejemplo, en las secuencias anteriores, estos serían igual a 11,7 y 3,9 respectivamente.

Teniendo en cuenta, por un lado, que el nivel de ruido de la cámara no va a cambiar rápidamente, y por otro, que el cálculo del histograma de una señal del tamaño de una imagen es bastante costoso computacionalmente, podemos determinar que el cálculo automático del umbral no se haga para cada nueva imagen, sino que puede hacerse una vez cada M imágenes. Por ejemplo, si el sistema procesa dos imágenes por segundo, se podría actualizar el umbral cada minuto, es decir, tomar un valor de $M = 60\,\text{seg} \times 2 = 120$ imágenes.

2.4.3. Detección de movimiento

El objetivo de cualquier algoritmo de detección de movimiento es la obtención de la máscara de movimiento correspondiente a cada marco de la secuencia. En nuestro caso, tal y como se ha comentado anteriormente, la obtención de dicha máscara dependerá del grupo al que pertenezca cada uno de los píxeles. Aunque el alcance de esta tesis no incluye el desarrollo de los algoritmos de detección de movimiento, en este punto sí se hace necesario exponer aquí la estrategia a seguir en futuras líneas de investigación que tengan como objetivo la implementación de un algoritmo de detección de movimiento basado en los resultados aquí obtenidos.

Así, de acuerdo a la clasificación anterior, la detección de movimiento en cada grupo se realizará de la siguiente forma:

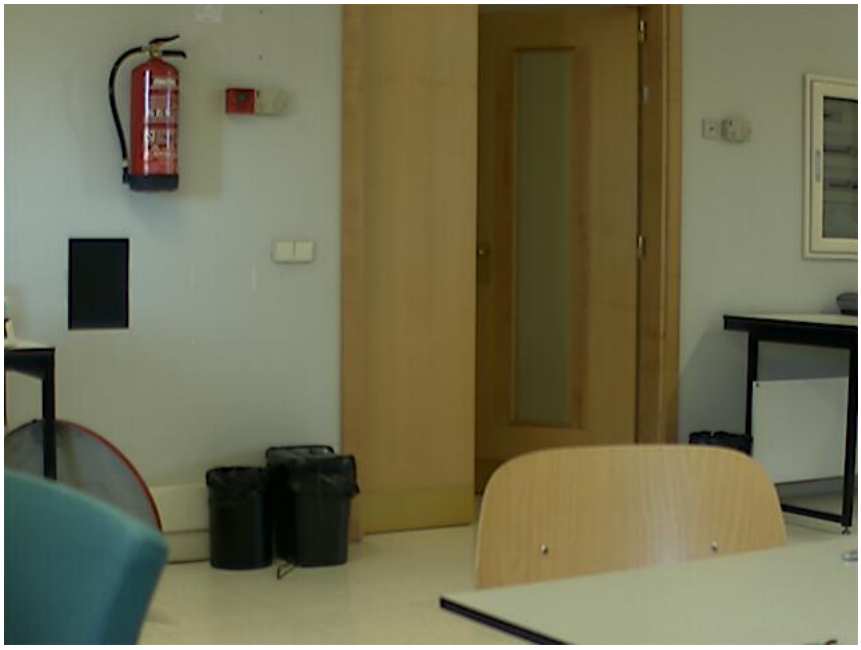


Figura 2.28: Secuencia con poca proporción de fondo dinámico

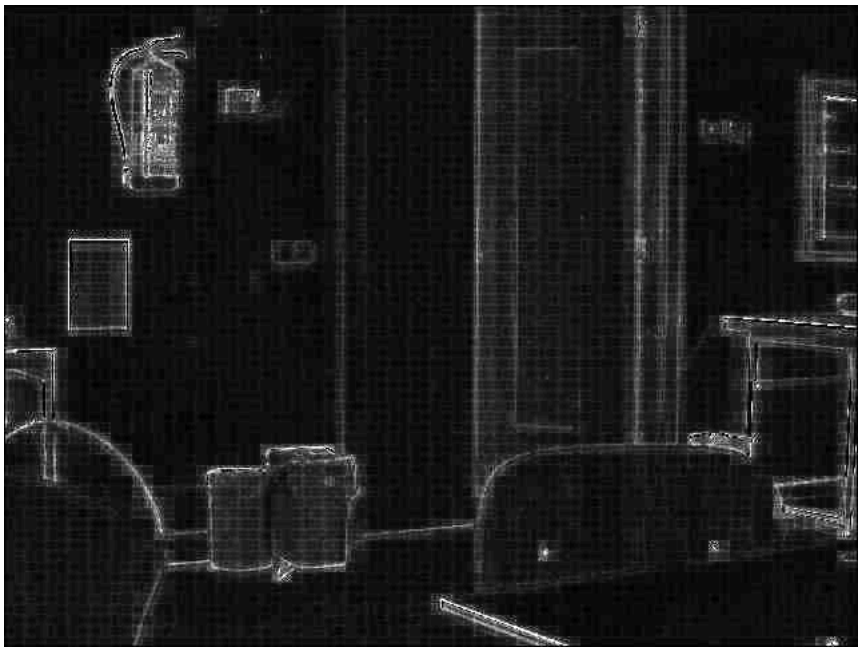


Figura 2.29: Desviación típica calculada para la secuencia de la figura 2.28

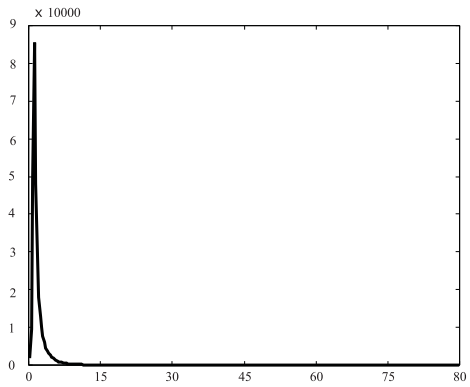


Figura 2.30: Histograma de la matriz de varianzas mostrada en la figura 2.29

- **Fondo estático:** Éste es el caso más sencillo de todos, ya que la única variación del nivel del píxel será debido al ruido de la cámara, y en ningún caso al movimiento continuo del fondo. Por ello, una simple umbralización del valor absoluto de la media, de acuerdo a los esquemas vistos para los algoritmos derivativos o algoritmos de imagen de referencia sería más que suficiente.

- **Fondo dinámico:** Como ya hemos concluido anteriormente, aquellas zonas de la imagen cuyos píxeles presentan una variación en sus niveles excesiva, no son propicias para aplicarles los mismos métodos de detección de movimiento que para aquellas zonas más estáticas. Éste es uno de los escenarios donde más difícil es realizar la detección de movimiento, ya que deberemos ser capaces de distinguir entre el movimiento normal del fondo y el movimiento de los objetos de primer plano. En este caso, se puede realizar una doble detección de movimiento de forma complementaria. El primero sería comprobar el nivel de variación de dichos píxeles a lo largo del tiempo, y si en algún momento ésta descendiera, puede querer indicar que algún objeto nuevo se ha detenido o ha sido depositado en dicha zona, lo cual puede ser de interés para el sistema. Por otro lado, también vamos a evaluar la variación de los píxeles vecinos, y si en algún momento ésta se elevase indicará un cambio en el comportamiento de dicha zona, lo cual puede volver a ser de interés para el sistema.

- **Fondo impulsivo:** Ya que este tipo de fondo se corresponde con zonas de la imagen con movimiento aleatorio y poco frecuente de objetos, como pueden ser carreteras o lugares de paso de personas, los eventos extraños a detectar en esta zona serían justamente los contrarios, es decir, cuando deja de haber movimiento porque un objeto se ha detenido en dicha zona. La detección en este caso sería muy simple, bastaría con comprobar si en algún momento el nivel de un píxel de este grupo permanece fuera de su nivel estacionario durante un número determinado de imágenes.

- **Fondo cambiante:** En este caso, dado que la mayor parte del tiempo los píxeles de esta zona se comportan como estáticos, la detección de movimiento se realizará de

la misma forma que para los píxeles estáticos. La información de que estos píxeles pertenecen al fondo cambiante puede ser utilizada por el sistema para evitar falsas alarmas cuando se producen cambios bruscos en el color de esa zona del fondo.

De esta forma, conseguimos solucionar dos de los problemas más típicos que suelen presentarse en los sistemas de detección de movimiento. Por un lado, la aparición de falsas alarmas continuas en las zonas de la imagen donde existe movimiento permanente de los objetos de fondo, básicamente aquellas zonas de la imagen que se corresponden con el fondo dinámico, y de falsas alarmas esporádicas en el fondo cambiante. En segundo lugar, también suele ser un problema bastante común la aparición de falsas alarmas intermitentes en zonas de la imagen donde existe un tránsito esporádico de objetos, como vehículos, personas, etc, es decir en aquellas zonas que en esta tesis hemos definido como impulsivas. Además, debido a la forma de realizar la detección de movimiento en zonas de la imagen que corresponden al fondo impulsivo, tenemos también la opción de realizar la detección de objetos que se detienen en zonas donde está prohibido. Éste es un caso muy común en aplicaciones de vigilancia de carreteras, donde un vehículo detenido puede suponer un peligro para el resto de usuarios de la carretera.

2.4.4. Esquema de bloques del sistema implementado

Para finalizar la descripción del sistema, una vez definido cada uno de los algoritmos individuales que lo forman, se describirá en esta sección la implementación global del mismo, para poder obtener así una idea general de su funcionamiento.

En la figura 2.31 se muestra el esquema de bloques completo del sistema. En primer lugar, puede verse que ha sido dividido en dos bloques generales. El primero de ellos sería el que se ejecutaría nada más arrancar la aplicación, y se encarga de encontrar el valor inicial para la media empleando la expresión:

$$\mu_{t_0} = \frac{1}{N} \sum_{i=0}^{N-1} I_{t_0-i} \tag{2.64}$$

donde t_0 es el instante de tiempo correspondiente a la primera imagen que es procesada por el sistema una vez transcurrido el periodo de entrenamiento. El valor así obtenido es el que se utiliza para inicializar los valores del principal ($F^{\mathrm{P}}_{t_0}$) y el fondo rápido ($F^{\mathrm{R}}_{t_0}$):

$$F^{\mathrm{R}}_{t_0} = F^{\mathrm{P}}_{t_0} = \mu_{t_0} \tag{2.65}$$

En el mismo bloque también se calcula la varianza de acuerdo a:

$$\sigma_{t_0} = \frac{1}{N} \sqrt{\sum_{i=0}^{N-1} (I_{t_0-i} - \mu_{t_0})^2} \tag{2.66}$$

Por último, en el bloque también se calcula el umbral global (th_{G}) que se corresponde con la posición donde se encuentra el máximo del histograma de la varianza, como se describió anteriormente (pag. 71), para la clasificación de los píxeles en dinámicos, estáticos o impulsivos.

Una vez terminada la función anterior, que hemos denominado como periodo de entrenamiento, el sistema entraría a ejecutar el otro gran bloque, una vez por cada nueva imagen. El procedimiento, tal y como se ha comentado a lo largo de este capítulo, sería el siguiente. En primer lugar, se actualizarían los estadísticos de la imagen, esto es, el fondo, tanto el que definimos como fondo rápido, de acuerdo a:

$$F_t^{\mathrm{R}} = (1-\alpha_{\mathrm{R}})\, F_{t-1}^{\mathrm{R}} + \alpha_{\mathrm{R}} I_t \tag{2.67}$$

como el principal:

$$F_t^{\mathrm{P}} = \begin{cases} F_{t-1}^{\mathrm{P}} & \text{si} \quad \mathrm{Nv}\left(\left|I_t - F_t^{\mathrm{R}}\right| > \sigma_t\right) < n \\ (1-\alpha_{\mathrm{R}})\, F_{t-1}^{\mathrm{P}} + \alpha_{\mathrm{R}} I_t & \text{si} \quad \mathrm{Nv}\left(\left|I_t - F_t^{\mathrm{R}}\right| > \sigma_t\right) \geqslant n \\ (1-\alpha_{\mathrm{P}})\, F_{t-1}^{\mathrm{P}} + \alpha_{\mathrm{P}} I_t & \text{resto} \end{cases} \tag{2.68}$$

y la varianza, a partir de:

$$\bar{\sigma}_t^2 = (1-\alpha_{\mathrm{V}})\, \bar{\sigma}_{t-1}^2 + \frac{\alpha_{\mathrm{V}}}{2}\left(I_t - I_{t-1}\right)^2 \tag{2.69}$$

donde $\alpha_{\mathrm{R}} > \alpha_{\mathrm{P}} > \alpha_{\mathrm{V}}$, y la función $\mathrm{Nv}\,()$ de la expresión anterior cuenta el número de veces que la expresión interior es cierta. Con la varianza ya actualizada, es posible calcular nuevamente el umbral global th_{G}. Sin embargo, como es de suponer que el ruido de la cámara, que es básicamente de lo que depende dicho umbral, no va a variar con cada nueva imagen, esta actualización no se hace para cada nueva imagen, sino una vez cada cierto número de imágenes. No hay que olvidar que el cálculo de este umbral es costoso desde el punto de vista computacional.

Posteriormente se pasaría a realizar la detección de impulsos en la escena. Para ello, se considera que un impulso se produce cuando el valor del píxel supera un umbral determinado (th_{I}), y no se produce impulso en caso contrario. El umbral se calcula a partir de la varianza del píxel de acuerdo a:

$$th_{\mathrm{I}} = p\bar{\sigma}_t \tag{2.70}$$

donde p se encuentra aproximadamente entre 2 y 4. Para todos aquellos impulsos detectados, se actualiza su contador de impulsos, de acuerdo a:

$$C'_t = \begin{cases} (1-\alpha)\, C'_{t-1} + \alpha k & \text{si} \quad z_t = 1 \text{ (impulso)} \\ (1-\alpha)\, C'_{t-1} & \text{si} \quad z_t = 0 \text{ (no impulso)} \end{cases} \tag{2.71}$$

donde $z_t = 1$ si el valor del píxel corresponde a un impulso, es decir, $I_t > th_{\mathrm{I}}$, y 0 en caso contrario. Finalmente el sistema determina si el píxel pertenece a la clase impulsiva

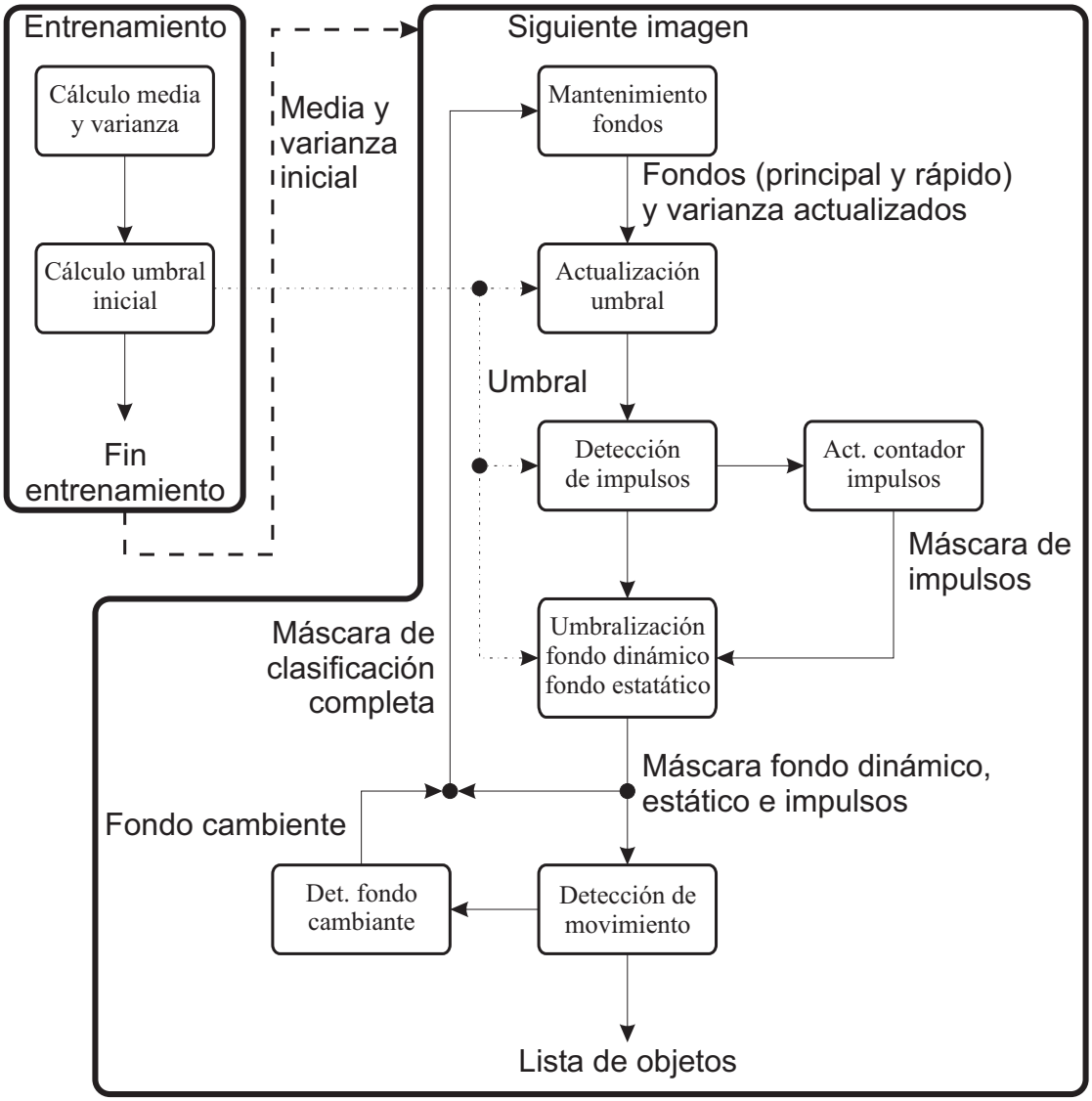


Figura 2.31: Esquema de bloques del sistema implementado. Dentro de cada bloque se especifica la función que realiza dicho bloque. Fuera de los bloques, las etiquetas indican la información de la que se dispone en el punto donde se sitúa la etiqueta.

si el valor de la expresión anterior se encuentra por encima de un determinado umbral th_{C}, cuyo valor se sitúa entre 0,01 y 0,05. El resultado de esta umbralización generará la correspondiente máscara de impulsos:

$$M_{\mathrm{I}} = \begin{cases} 1 & \text{si} \quad C'_t > th_{\mathrm{C}} \\ 0 & \text{si} \quad C'_t \leq th_{\mathrm{C}} \end{cases} \tag{2.72}$$

donde M_{I} es la máscara de píxeles impulsivos, y toma el valor 1 para los píxeles pertenecientes al fondo impulsivo, y 0 para el resto. En el siguiente bloque se realiza la clasificación del resto de píxeles en dinámicos o estáticos, mediante la umbralización de la varianza de estos píxeles. Para esta clasificación, se emplea el umbral th_{G} calculado durante el periodo de entrenamiento, y actualizado cada cierto tiempo, a partir de la posición del máximo del histograma de la varianza. En resumen, la máscara de clasificación del fondo, M_{F}, se obtiene de acuerdo a:

$$M_{\mathrm{F}} = \begin{cases} \text{Impulsivo} & \text{si} \quad M_{\mathrm{I}} = 1 \\ \text{Dinámico} & \text{si} \quad M_{\mathrm{I}} = 0 \text{ y } I_t > th_{\mathrm{G}} \\ \text{Estático} & \text{si} \quad M_{\mathrm{I}} = 0 \text{ y } I_t \leq th_{\mathrm{G}} \end{cases} \tag{2.73}$$

La máscara aquí obtenida será la empleada por el módulo de detección de movimiento para extraer los objetos de primer plano. Al mismo tiempo, la máscara de movimiento será también la empleada para detectar las zonas del fondo estático que pertenecen al denominado fondo cambiante.

La salida de este último bloque, junto con la clasificación anterior de la escena en fondo dinámico, estático e impulsivo forma la máscara completa de la clasificación de la escena. Esta máscara será utilizada como realimentación por el bloque de mantenimiento del fondo, especialmente con la información de píxeles impulsivos y cambiantes, para actualizar correctamente los nuevos valores del fondo.

2.4.5. Ejemplo práctico 1: control de intrusión

Una vez analizado todo el sistema de clasificación del fondo, podemos ejecutar el sistema con imágenes reales para comprobar su funcionamiento. Para ello elegimos una secuencia de vídeo que incluya la mayor parte de adversidades que se han tratado de solventar en este trabajo, es decir, imágenes exteriores preferiblemente, con un fondo que incluya objetos en continuo movimiento. Las imágenes en exteriores permite comprobar la adaptación gradual del sistema a los cambios de iluminación, conforme avanza el día y se modifica la iluminación solar. Para incluir objetos de fondo en movimiento, una opción, que suele ser la que más problemas plantea a los sistemas de videovigilancia, suelen ser los árboles, ya que incluso un ligero movimiento de las hojas de estos hace que la variación del nivel de gris de los píxeles que corresponden a esta zona va a ser muy grande. El fondo impulsivo está pensado principalmente para el análisis de carreteras o zonas de paso de

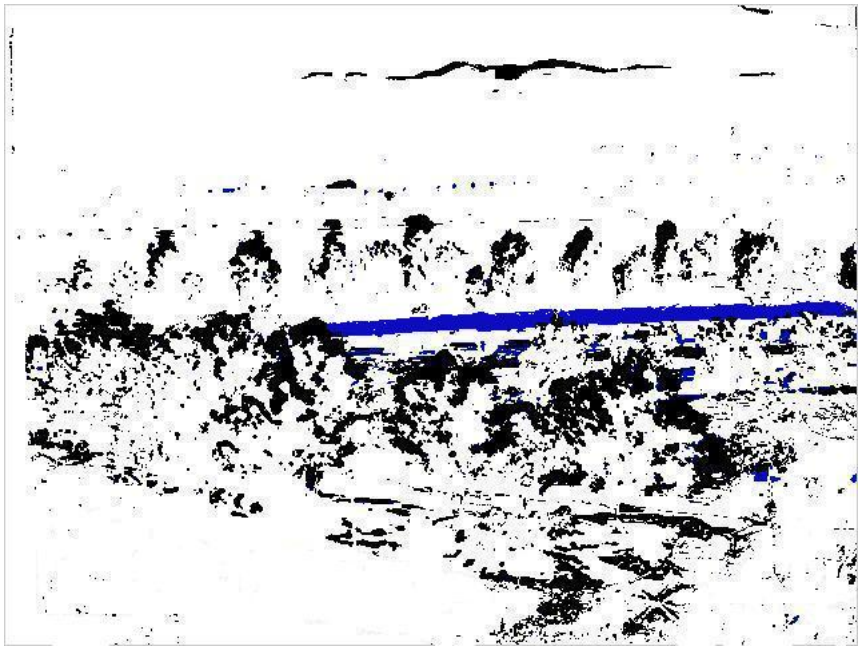


Figura 2.32: Resultado de la clasificación de los píxeles de la escena en la secuencia *Parking*

peatones. Por tanto, la secuencia también incluye una carretera, para poder comprobar el comportamiento del sistema en estos casos.

En la figura 2.8 se muestra la primera imagen de la secuencia a analizar. Como se puede ver, la secuencia incluye zonas ruidosas, formadas por árboles, y una carretera. Si ejecutamos el algoritmo descrito en este apartado, después de transcurrido unos segundos, el mapa de clasificación de píxeles se estabiliza, resultando en una imagen como la mostrada en la figura 2.32. En esta imagen, los píxeles clasificados como pertenecientes al fondo dinámico están marcados en negro, los pertenecientes al fondo impulsivo en azul, y finalmente, los pertenecientes al fondo estático en blanco. Puede comprobarse fácilmente como el fondo dinámico corresponde principalmente a los árboles de la imagen, mientras que el fondo impulsivo corresponde únicamente a la zona de la carretera, habiendo sido marcado el resto como fondo estático.

2.4.6. Ejemplo práctico 2: sistema de control de la congestión en autovías

En este nuevo ejemplo, vamos a desarrollar una aplicación ideada para controlar el nivel de tráfico en carreteras. Esta aplicación puede ser usada por los agentes encargados de la vigilancia de carreteras para controlar y prevenir, de forma sencilla, la congestión. Para ello, la aplicación básicamente realiza la clasificación de los píxeles de la imagen, tal y como ha sido descrito anteriormente. Sin embargo, en este caso, ya que lo que nos interesa controlar es la carretera, nos centraremos básicamente en los píxeles clasificados

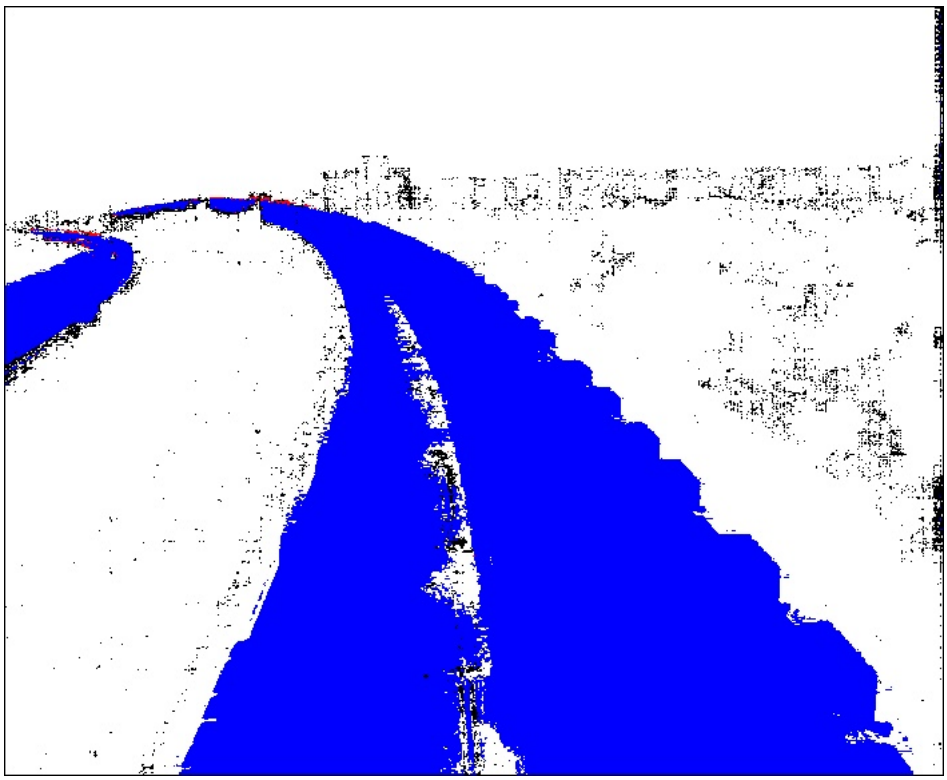


Figura 2.33: Detección de la carretera para control de la congestión en la secuencia *NII*. Azul: Píxeles impulsivos. Blanco: Píxeles estáticos. Negro: Píxeles ruidosos. Rojo: Píxeles cambiantes.

como impulsivos. La figura 2.33 muestra el resultado de aplicar el sistema desarrollado en este tema a la secuencia *NII*. En este caso, siguiendo el mismo código de colores que el utilizado en el ejemplo anterior, los píxeles impulsivos han sido marcados en color azul, los píxeles estáticos en blanco y los dinámicos en negro. Aquí podemos comprobar principalmente cómo la mayoría de los píxeles pertenecientes a la carretera han sido coloreados de color azul, es decir, han sido clasificados como píxeles impulsivos, que era lo que andábamos persiguiendo. Por otro lado, es interesante indicar que en esta escena no existe gran cantidad de fondo dinámico, únicamente el correspondiente principalmente a ciertas zonas de la vegetación que son agitadas ligeramente por el viento.

Una vez realizada la clasificación de la imagen, especialmente la clasificación de la zona impulsiva, y por lo tanto, la detección de la zona de la escena que corresponde a la carretera, se puede hacer una estimación del nivel de congestión a partir de la cantidad de movimiento detectada en esa zona. Aunque sería posible realizar dicha estimación con métodos más complejos, el método propuesto en este caso sería tan sencillo como hacer un recuento del total de píxeles pertenecientes a la carretera donde existe movimiento. Este método tan sencillo nos ha aportado resultados bastante aceptables.

2.4.7. Ejemplo práctico 3: detección de abandono de objetos

Una aplicación de los sistemas de videovigilancia que está cobrando una importancia cada vez mayor es la detección de abandono o robo de objetos. Desde el punto de vista de la videovigilancia, esto implica el ser capaz de interpretar como píxeles pertenecientes a un objeto abandonado aquellos píxeles que cambian bruscamente a un valor distinto al que había tomado durante el periodo anterior, y se mantiene en dicho valor durante un tiempo suficiente. Para la detección de objetos robados, el razonamiento es similar.

En nuestro caso, con el sistema desarrollado en este tema, basta con detectar aquellos píxeles marcados como cambiantes. De acuerdo al algoritmo descrito, los píxeles cambiantes son aquellos que, tras haber sido clasificados como píxeles estáticos, en un momento determinado su valor cambia bruscamente para quedarse en ese nuevo valor durante un tiempo suficientemente grande. Este comportamiento, aparte de corresponder a aquellas zonas de la imagen que cambian bruscamente, como pueden ser las nubes de la escena, también se corresponde con las situaciones de abandono, y por extensión, con las de robo.

En las siguientes figuras puede verse un ejemplo de todo esto. La figura 2.34 corresponde con el fondo estimado en una secuencia de exterior. En un momento determinado de dicha secuencia, se accede al escenario para dejar un objeto abandonado (figura 2.35), y dicho objeto permanece estático en dicha posición durante un tiempo suficiente, como se puede ver en la figura 2.36. En la figura 2.37 se muestra la máscara de clasificación del fondo obtenida varios marcos después de haber dejado el objeto. Como se puede ver, el objeto aparece perfectamente localizado en color rojo. Aunque, tal y como ha sido desarrollado el sistema, si este objeto permanece en dicha posición un tiempo suficiente, volverá a ser clasificado como fondo, lo que queda claro es que la marca de objeto abandonad permanecerá activa durante un tiempo suficientemente grande como para que el bloque superior puede interpretar dicha actividad como un abandono de objeto.

Figura 2.34: Fondo estimado de la secuencia de abandono de objetos.

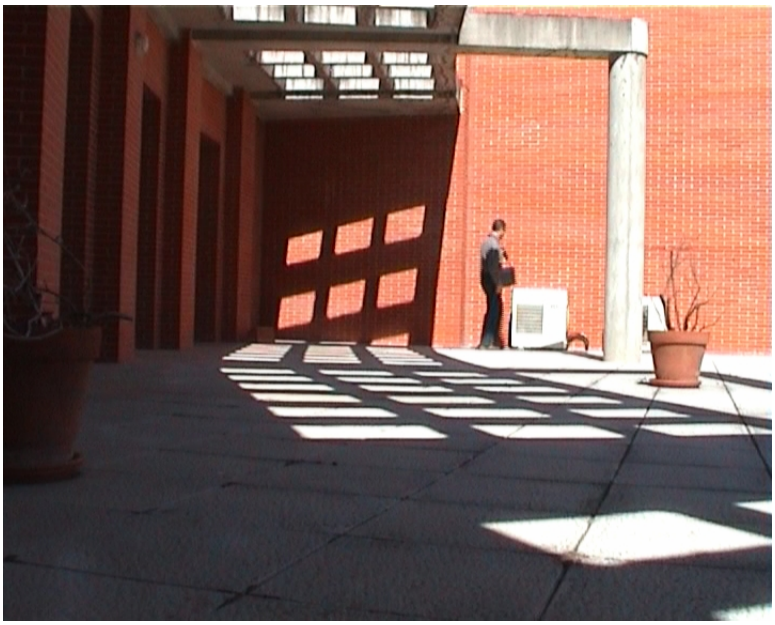


Figura 2.35: Instante en el que se abandona el objeto en la escena vigilada.

Figura 2.36: Objeto abandonado

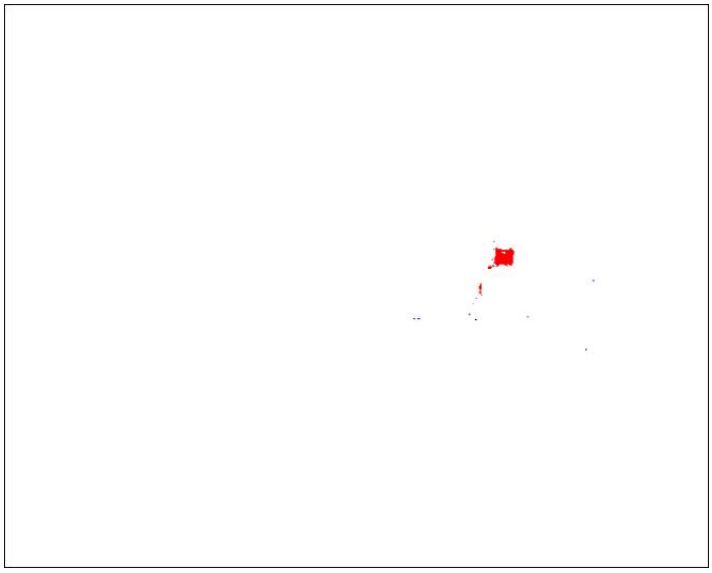


Figura 2.37: Máscara de clasificación de fondo. En blanco se representa el fondo estático, mientras que en rojo se representa el fondo cambiante. En este caso, no existe fondo dinámico ni impulsivo.

Capítulo 3

Estimación de distancias

En muchas aplicaciones de videovigilancia, el determinar la distancia a la que se encuentra un objeto de interés respecto del sistema de cámaras, o incluso posicionar dicho objeto dentro del entorno vigilado, suele ser una función a tener en cuenta dentro del sistema completo. Tradicionalmente, los esquemas propuestos han tratado de imitar la forma en la que el sistema visual humano obtiene la forma de los objetos a partir de las imágenes capturadas por sus retinas. De esta forma, varias son las características a partir de las cuales un sistema de visión artificial puede obtener una representación 3D de una escena [Pajares00]:

- Sombras.
- Contornos.
- Texturas.
- Enfoque.
- Movimiento.
- Visión estereoscópica.

Si bien todos los métodos anteriores pueden ser empleados por un sistema de visión artificial para obtener la forma, y a partir de ahí la distancia, de los objetos, para el sistema de videovigilancia que nos ocupa no todos los métodos anteriores podrían ser utilizados.

Por ejemplo, los métodos que obtienen la representación 3D a partir de las sombras se basan en el hecho de que el brillo con el que se visualiza una superficie de un objeto depende de la orientación de dicha superficie con respecto al punto de donde proviene la luz que ilumina dicho objeto. Es por tanto imprescindible que la iluminación sea uniforme, o al menos conocida. Los métodos que se basan en los contornos de los objetos son capaces de interpretar la posición en tres dimensiones de las líneas visualizadas en las imágenes

teniendo en cuenta la forma en que éstas han sido proyectadas por el sistema visual. Estos métodos necesitan tener conocimiento de la forma de los objetos para poder interpretar correctamente dichas líneas. Los métodos para obtener la forma a partir de la textura se basan en la distorsión que presentan las texturas en función de la orientación de la superficie con respecto al sistema de captación visual. En este caso, será necesario tener conocimiento al menos de cómo es la textura de dichos objetos.

Por todo lo expuesto en el párrafo anterior, los tres primeros métodos no son convenientes para nuestro propósito de estimar la distancia a la cual se encuentran los objetos de interés.

Continuando con los métodos enumerados anteriormente, los tres últimos métodos no necesitan un conocimiento a priori de la escena observada, ni operar sobre entornos controlados. Los métodos basados en el enfoque se basan en la propiedad de que una cámara de vídeo que emplea un sistema óptico real, solamente quedan perfectamente enfocados los objetos que se encuentren a una distancia determinada. La distancia de un punto de la escena se obtiene por tanto buscando la posición de enfoque óptimo, es decir, la distancia focal en la cual el punto de la imagen está perfectamente enfocado [Pentland87].

Por último, los métodos basados en el movimiento o en visión estereoscópica realizan la captura de dos o más imágenes de una misma escena desde puntos de vista distintos. La imágenes, aun procediendo de la misma escena, se presentan ligeramente desplazadas entre sí. Este desplazamiento de los objetos dependerá de las posiciones desde las cuales se capturan las imágenes y de la situación en el espacio de los objetos considerados. Dicho desplazamiento es la característica empleada por estos sistemas para poder calcular la distancia. La diferencia principal entre ambos es que, mientras en el caso estéreo existen varias cámaras, y la posición relativa de las cámaras es conocida a priori, en el caso del movimiento una única cámara es la que captura todas las imágenes mediante un desplazamiento de ésta. Para este segundo caso, por tanto, aparte de las funciones propias de la visión estéreo, el sistema debe implementar las funciones que permitan obtener la posición relativa entre cada uno de los puntos de vista con los cuales fueron tomadas las sucesivas imágenes. Aunque ambos métodos permiten obtener resultados similares, hay que tener en cuenta que para el sistema de videovigilancia que nos ocupa, las cámaras en general siempre van a estar fijas, al menos en lo que respecta a su centro de proyección, por lo que la estimación de distancias basada en movimiento no va a ser de aplicación en nuestro caso.

Existen otras técnicas para la obtención de distancias distintas a las enumeradas anteriormente, como pueden ser el uso de luz estructurada, láseres de distancia o radares. Estas técnicas, aunque pueden llegar a presentar mejores resultados que los comentados anteriormente, necesitan del uso de elementos activos (láser, rádar, etc), que para el caso de la videovigilancia no suelen ser convenientes, bien sea porque el sistema opera en entornos públicos, con el consiguiente peligro para la salud de las personas, bien porque el rango de distancias suele ser elevado. Aunque en algunas aplicaciones estos sistemas

pueden ser empleados como complemento a los sistemas de vídeo procesado, en esta tesis solamente se tendrán en cuenta aquellos basados en visión artificial.

A partir de todo esto, queda claro que los dos únicos métodos que pueden ser utilizados para la estimación de distancias son aquellos basados en el desenfoque y los basados en visión estéreo, y por tanto, los únicos que contemplaremos.

3.1. Visión estereoscópica

La visión estereoscópica, o estereovisión, si bien no es la única, si que es la herramienta más importante para la obtención de información de distancia en los humanos. En el campo de la visión artificial, la visión estéreo es el método preferido para la obtención de la representación en tres dimensiones de una escena. El principal inconveniente para su utilización práctica ha sido la gran cantidad de computación necesaria para calcular la correspondencia de puntos entre las dos imágenes. Actualmente, el empleo de procesadores con cada vez mayor capacidad de cálculo puede reducir en gran manera este inconveniente.

Un sistema de visión estereoscópica consiste en un conjunto de dos, o en ocasiones más de dos, cámaras de vídeo separadas entre sí una determinada distancia, en general conocida, capturando la misma escena desde puntos de vista distintos. Este conjunto de dos o más imágenes son la entrada a un bloque de procesado de imagen donde se trata de establecer correspondencias entre características de la escena, por ejemplo, puntos, bordes o regiones, en las distintas imágenes capturadas. Aunque el número de cámaras que se pueden emplear no está limitado, en general suele utilizarse un sistema binocular, por su sencillez y menor coste respecto al uso de un número más elevado de cámaras. Una vez establecida la correspondencia, mediante una sencilla triangulación puede hallarse la distancia de esa característica de la escena a las cámaras. Este planteamiento presenta los siguientes grandes problemas:

- **Definición del modelo de las cámaras y calibración de las mismas:** La elección del modelo de cámara y su calibración es siempre uno de los primeros problemas a solventar en muchas aplicaciones de visión artificial. En el caso de la visión estéreo, no solo es necesario realizar la calibración de cada una de las cámaras, sino también la de todo el conjunto, es decir, la obtención de la posición relativa de todas las cámaras entre sí. Aunque existen algoritmos de visión estéreo que funcionan con sistemas de cámaras no calibradas, estos en general van a presentar una mayor necesidad de cómputo, y los resultados obtenidos van a ser menos precisos que con sistemas calibrados.

- **Obtención de las correspondencias entre imágenes (matching):** Consiste en determinar a qué punto de una imagen corresponde un punto en otra imagen, para obtener lo que se denomina un mapa de disparidades. Si esto se realiza para todos los píxeles de la imagen se denomina mapa de disparidades denso, o bien si

dicho mapa no incluye todos los píxeles, si no más bien una porción reducida de estos, se denomina mapa de disparidades disperso. Éste es el mayor problema de la estereovisión, a causa de la ambigüedad que se presenta a la hora de establecer las correspondencias entre puntos de las imágenes, y del elevado número de veces que hay que realizar dicha estimación. En principio, un punto de una imagen puede corresponder a cualquier punto de la otra, por lo que se utilizarán una serie de restricciones geométricas y físicas para tratar de reducir la ambigüedad de la correspondencia, así como la zona de búsqueda de correspondencias.

- **Reconstrucción 3D:** Una vez realizada la correspondencia de los píxeles de la imagen, si el sistema está bien calibrado, el cálculo de la profundidad o distancias es inmediato. El inconveniente puede ser en algunos métodos que el número de puntos para los cuales se halló correspondencia sea escaso, lo que obliga a realizar una interpolación para la obtención de un mapa de profundidades denso.

Para el caso de las aplicaciones de videovigilancia, el cálculo de distancias puede realizarse únicamente sobre los objetos de interés, lo que reduce considerablemente el coste computacional, ya que el cálculo de disparidades se realiza sobre una porción muy pequeña de la imagen. También puede ser necesario, no obstante, la obtención de la estructura en tres dimensiones del escenario vigilado, aunque teniendo en cuenta que éste es estático, el cálculo puede ser realizado a priori. Es por tanto interesante resaltar aquí que, aunque sigue siendo necesario el cumplir las exigencias de funcionamiento en tiempo real, éstas pueden conseguirse fácilmente incluso con los algoritmos más complejos, debido a que el número de puntos a analizar será, en general, pequeño.

3.1.1. Modelo de cámara y calibración

La calibración de la cámara consiste en determinar los elementos que gobiernan la relación o transformación entre la información en tres dimensiones del objeto real y la imagen en dos dimensiones de dicho objeto percibida por la cámara. Para poder evaluar esta relación, se hace necesario establecer un modelo de cámara que nos permita medir estos parámetros. El modelo de cámara más sencillo, y que suele ser el empleado en las aplicaciones de visión artificial, es el modelo *pin-hole*. El modelo se denomina *pin-hole* porque refleja la situación que se produciría si el plano focal fuera un plano físico real con un orificio infinitesimal, en el centro de proyección C, como se representa en la figura 3.1. Un modelo de cámara tan sencillo puede modelar con gran exactitud la geometría y óptica de la mayor parte de las cámaras modernas. En algunos casos, sin embargo, es preciso considerar la distorsión radial respecto del modelo debida a las lentes ópticas empleadas.

En una cámara *pin-hole*, el plano imagen representa el plano donde se registra la imagen percibida por la cámara. Por ejemplo, en una cámara CCD, el propio CCD sería el plano imagen. Aunque de todos es sabido que en una cámara CCD el plano imagen se encuentra por detrás del centro de proyección (el centro de proyección se sitúa en

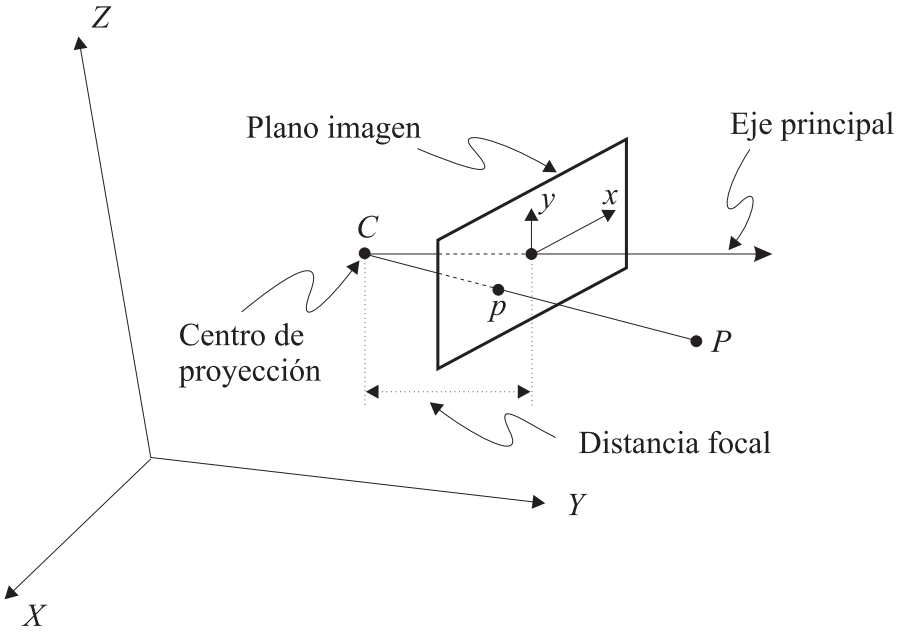


Figura 3.1: Geometría del modelo de cámara *pin-hole*

algún punto próximo a las lentes que forman la óptica de la cámara), en general, en los desarrollos teóricos el plano imagen suele situarse por delante, para así evitar la inversión que se produce en la imagen en una cámara real. La línea perpendicular al plano imagen que pasa por el centro de proyección se denomina *eje principal*, aunque en ocasiones también podemos denominarlo como *eje óptico* de la cámara.

Para el modelo de cámara *pin-hole*, las leyes que gobiernan la proyección del mundo real en tres dimensiones a la imagen en dos dimensiones puede expresarse simplemente por una transformación lineal de las coordenadas de un punto en el espacio de tres dimensiones P a otro punto en la imagen p de dos dimensiones [Hartley03]. Esta transformación lineal depende de una serie de parámetros físicos y geométricos, cuya determinación es el objetivo de la calibración. Estos parámetros se dividen en intrínsecos, que no dependen de la posición y orientación de la cámara, sino que son propios de la misma, y extrínsecos, que vienen dados por la posición y orientación de la cámara en el espacio.

Los parámetros intrínsecos son los siguientes:

- Posición del centro de la imagen respecto del sistema de referencia del plano imagen.
- Distancia focal f.
- Factores de escala para los ejes x e y.

Existe un parámetro más, que es el ángulo que forman los ejes x e y de la imagen, aunque generalmente se considera que dicho ángulo es de 90 grados. Los parámetros

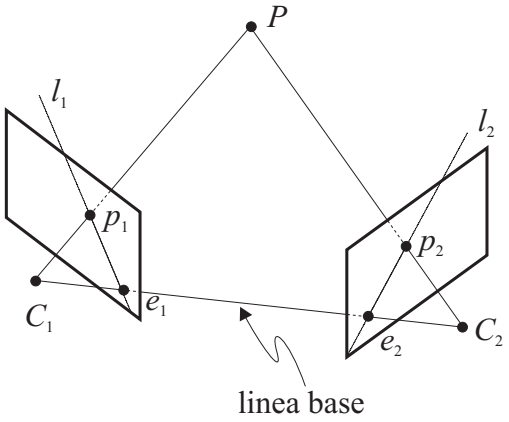


Figura 3.2: Geometría epipolar

extrínsecos definen la posición y orientación de la cámara en el espacio. Éstos quedan claramente definidos si se conoce:

- Coordenadas del centro óptico C respecto del sistema de coordenadas del espacio de tres dimensiones.
- Orientación de la cámara en el espacio.

A partir de los parámetros anteriores, es posible obtener la transformación lineal que relaciona las coordenadas del mundo real con las coordenadas de la imagen. El objetivo de la calibración es hallar los parámetros anteriores mediante medidas geométricas, o en su defecto, obtener la expresión lineal que gobierna la transformación directamente. El primer método se denomina explícito, y requiere para su obtención de aparatos de medición de magnitudes físicas complejos, y por tanto, no suele ser recomendado a la hora de realizar la calibración de la cámara. El segundo método se denomina implícito, y emplea la propia cámara como elemento de medida. Es decir, los parámetros de la transformación lineal pueden obtenerse a partir de correspondencias entre puntos reales de coordenadas conocidas, y sus correspondientes puntos en la imagen capturada. Este método suele ser el preferido, ya que no requiere de ninguna instrumentación adicional.

Lo descrito hasta este punto sería el esquema de calibración de una cámara de vídeo para cualquier aplicación de visión artificial. En el caso de visión estéreo binocular, hay que tener en cuenta que también es necesario conocer la posición relativa entre ambas cámaras. La disposición relativa de las cámaras entre sí, y todas las relaciones geométricas que de dicha disposición se derivan se denomina *geometría epipolar*.

En la figura 3.2 puede visualizarse la disposición de dos cámaras estéreo, junto con los elementos que componen la geometría epipolar. Si consideramos C_1 y C_2 los centros de proyección de las cámaras 1 y 2 respectivamente, la línea que une dichos centros se denomina *línea base*. Por otro lado, la imagen del centro de proyección de la cámara 2

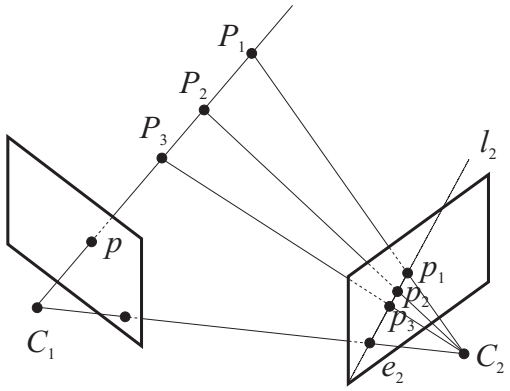


Figura 3.3: Correspondencia entre puntos y línea epipolar

en la cámara 1, e_1, y la imagen del centro de proyección de la cámara 1 en la cámara 2, e_2 se denominan *epipolos*. Si suponemos un punto real P, cuya imagen en las cámaras 1 y 2 sean p_1 y p_2 respectivamente, podemos comprobar en la figura 3.2 que los centros de proyección C_1 y C_2, el punto P y las imágenes de dicho punto p_1 y p_2 pertenecen al mismo plano. Cada uno de los distintos planos que podemos formar se denominan *planos epipolares*, con la característica de que todos estos planos contienen a la línea base.

Consideremos ahora la imagen de un punto p en la cámara 1, y deseamos saber qué puntos de la imagen 2 pueden ser su correspondiente en la otra imagen. A la vista de la figura 3.3, es fácil deducir que dicho punto deberá pertenecer, obligatoriamente, a la línea l_2. Dicha línea se obtiene geométricamente como la intersección del plano epipolar formado por los centros de proyección C_1 y C_2 y el punto P con el plano imagen de la cámara 2. La línea así formada se denomina *línea epipolar para p*. El concepto de línea epipolar es uno de los más importantes, ya que restringirá la búsqueda de correspondencias para un punto a una única línea, en lugar de ser una búsqueda bidimensional por toda la imagen.

Para una disposición genérica de las cámaras, como la mostrada en la figura 3.2, las líneas epipolares de una imagen forman un conjunto de líneas con la propiedad de que todas pasan por el epipolo de dicha imagen. Por tanto, las líneas epipolares de una imagen no van a ser, en general, paralelas entre sí, sino que cada una tendrá una inclinación determinada. Existe una configuración, no obstante en la que es posible conseguir que todas las líneas sean paralelas. Tal y como se muestra en la figura 3.4, donde los ejes de ambas cámaras son paralelos, todos los planos epipolares intersectan en el plano imagen con la misma orientación. Además, si uno de los ejes de ambas cámaras, por ejemplo el x, es paralelo a la línea base, las líneas epipolares de ambas imágenes serán paralelas, en este caso al eje x. Esto, que teóricamente no tiene ninguna implicación más allá de la sencillez en su representación, computacionalmente tiene una gran importancia, ya que con esta configuración simplificamos la búsqueda de correspondencias a una línea, horizontal o vertical, de la imagen. Es decir, que para un píxel de coordenadas (x_1, y_1) determinadas,

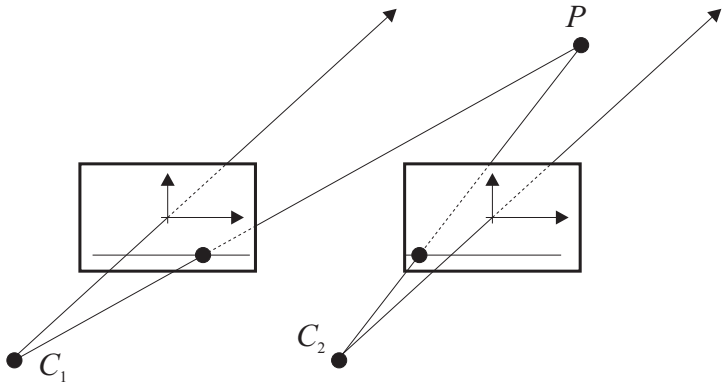


Figura 3.4: Geometría epipolar con ejes alineados

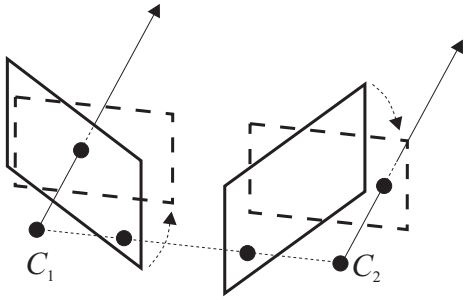


Figura 3.5: Rectificación epipolar de las imágenes

su píxel correspondiente debe tener, obligatoriamente, coordenadas (x_1, y), donde y será la única variable a determinar con el algoritmo de correspondencia estereoscópica empleado.

Para una configuración de ejes no alineados, las líneas epipolares ya no son horizontales ni verticales, sino que tendrán una pendiente determinada. Esto implica que, a la hora de realizar la búsqueda de correspondencias, es necesario calcular que píxeles de la imagen pertenecen a la línea epipolar considerada, con la consiguiente carga computacional.

Para este tipo de configuración, también es posible crear, a partir de las imágenes capturadas, una disposición virtual en la que ambas cámaras tengan ejes alineados, como en la figura 3.5. Para una imagen capturada con una cámara con su centro óptico en una posición determinada (X, Y, Z), se puede generar una imagen sintética a partir de dicha imagen con una orientación distinta, eso si, sin modificar la posición de su centro óptico [Hartley03]. De esta forma, se pueden generar vistas sintéticas a partir del par estéreo capturado para que la disposición virtual de las cámaras sea la de ejes alineados vista en la figura 3.4. Este procedimiento se denomina *rectificación de la imagen*, y debe ser aplicado a todos los píxel de ambas imágenes.

La elección de uno u otro método debe verse desde el punto de vista computacional. El primero implica que cada vez que queremos acceder a una línea epipolar, debemos calcular sus parámetros. El segundo método implica la generación de la imagen sintética

una única vez para cada imagen. Por tanto, si vamos a acceder a una línea epipolar más de una vez, sería recomendable rectificar las imágenes, ya que los sucesivos accesos a la línea epipolar no van a requerir ningún cálculo adicional. Como desventaja, el segundo método implica el cálculo para todos los píxeles de la imagen. Existen algoritmos de correspondencia estereoscópica que no acceden a todos los píxeles de la imagen, sino a un número muy reducido de estos. En este caso, podría ser más conveniente el no rectificar las imágenes y calcular las líneas epipolares para cada punto considerado. En lo sucesivo, se supondrá que las cámaras están perfectamente alineadas, ya sea porque las cámaras están perfectamente dispuestas, como indica la figura 3.4, ya sea porque se ha conseguido realizar la rectificación con cualquiera de los métodos anteriores.

Para finalizar, es necesario indicar que la discusión anterior sólo tiene implicación en el coste computacional del algoritmo, y por tanto, en el tiempo de ejecución por cada par de imágenes. No tiene ninguna implicación en el resultado final, ya que ambos deberían obtener el mismo resultado.

En una configuración de cámaras alineadas como la mostrada en la figura 3.4, donde las dos cámaras se suponen iguales y con los sistemas de referencia idénticos, diferenciándose únicamente en la posición de sus orígenes, el cálculo de la profundidad se puede obtener de forma sencilla e inmediata.

Se conoce como disparidad a la diferencia en la posición entre ambas imágenes de dos puntos correspondientes. Puede verse intuitivamente que la disparidad es inversamente proporcional a la distancia entre el punto de la escena y el sistema de cámaras. Si los puntos estuvieran en el infinito la disparidad sería cero, y aumenta al acercarse a las cámaras.

En el caso de cámaras paralelas horizontales, la disparidad sólo tiene componente horizontal, debido a que los puntos correspondientes de ambas imágenes se sitúan sobre la misma línea horizontal. Supongamos que las cámaras se encuentran situadas con el eje óptico en la dirección del eje Z positivo y separadas una distancia d_x en el eje x, tal y como se muestra en la figura 3.6. Tomando el triángulo formado entre el punto observado P y los centros ópticos del sistema de observación C_{L} y C_{R}, la información de profundidad Z del punto P vendrá dada por:

$$x_{\mathrm{L}} = \frac{x_p f}{Z_p} \tag{3.1}$$

$$x_{\mathrm{R}} = \frac{(x_p - d_x)\, f}{Z_p} \tag{3.2}$$

De donde podemos obtener que:

$$Z_p = \frac{d_x f}{x_{\mathrm{L}} - x_{\mathrm{R}}} \tag{3.3}$$

A raíz de la expresión anterior, se observa que la profundidad depende inversamente de la disparidad. Es decir, objetos que se encuentran en el infinito, o al menos muy

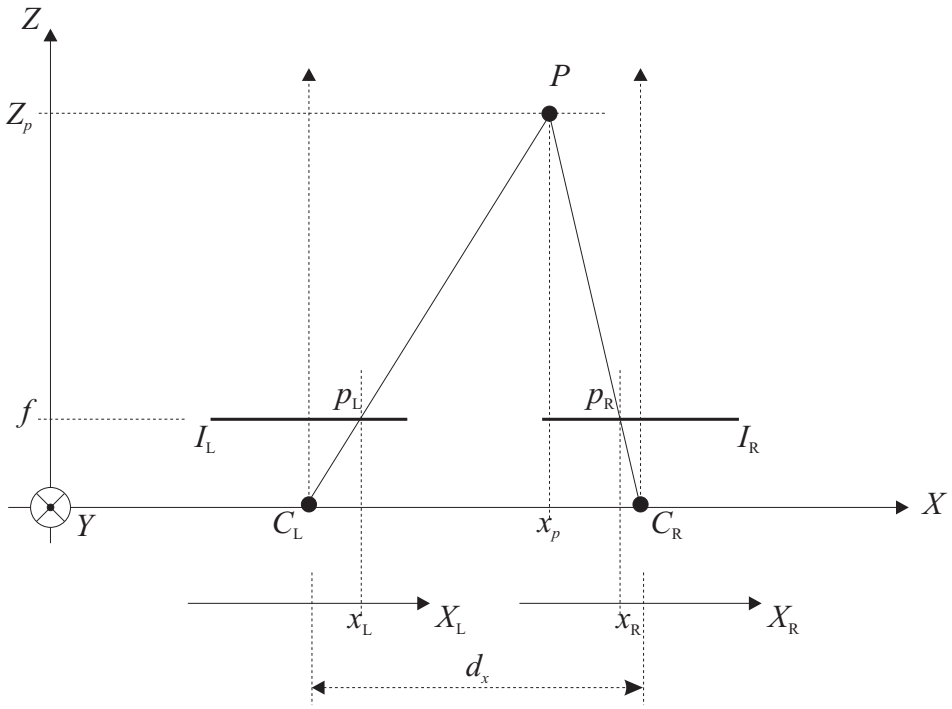


Figura 3.6: Cálculo de la profundidad en un sistema estéreo.

alejados del sistema de cámaras, tendrán una disparidad nula, mientras que la máxima disparidad se alcanzará con objetos que se encuentren muy próximos al sistema. Otro aspecto a tener en cuenta aquí, aunque se insistirá en el apartado dedicado a la búsqueda de correspondencias, es que la disparidad nunca puede ser menor de cero. Esta propiedad reducirá el rango de búsqueda en la estimación de las correspondencias estereoscópicas.

También es importante notar que la distancia es proporcional a la separación d_x de las cámaras. Si tenemos en cuenta que la disparidad es una magnitud discreta, medida en píxeles, y por supuesto, en en sistema real vendrá alterada por el ruido, la precisión del sistema va a depender de la separación entre las cámaras, y para una separación determinada, la precisión será diferente para distintas distancias. Por ejemplo, si representamos la profundidad del objeto para un par de cámaras con distancia focal de 1000 píxeles en función de la disparidad en la imagen para dos valores de la distancia d_x distintas, $d_x = 10$ cm y $d_x = 1$ m, podemos obtener una gráfica similar a la que se muestra en la figura 3.7. Si nos fijamos en la curva con distancia de separación entre cámaras de 1 metro, podemos apreciar que la longitud máxima que puede medir con cierta precisión es de unos 300 metros. Sin embargo, si nos fijamos en la gráfica correspondiente a una separación de 10 centímetros, podemos apreciar que, aunque la distancia máxima es mucho menor, también se puede apreciar que la precisión alcanzada para distancias pequeñas es mucho mayor. Para aclarar esto, podemos hacer un símil con el sistema visual humano. Los ojos izquierdo y derecho de una persona están separados unos pocos centímetros. Según estu-

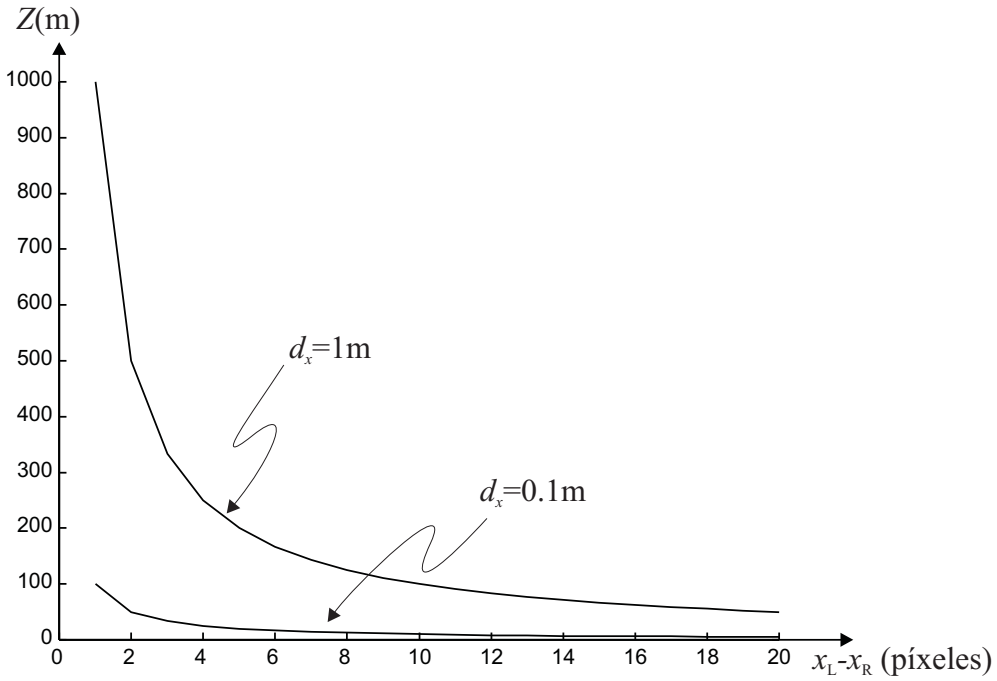


Figura 3.7: Profundidad en función de la disparidad.

dios científicos, se puede decir que una persona no es capaz de distinguir profundidades claramente a partir de unos 20 ó 30 metros. Si la separación hubiese sido mayor, seríamos capaces de distinguir profundidades a distancias más lejanas, pero tendríamos menor capacidad de distinguir con precisión profundidades a distancias cercanas.

Por tanto, en la fase de puesta a punto y calibración del sistema estéreo, es necesario hacer una reflexión acerca del rango de distancias que queremos medir. Cuanto mayor sea el rango de distancias a medir más separadas deberían estar las cámaras entre sí, aunque perderemos precisión a distancias cercanas. En los sistemas de videovigilancia que nos ocupan, nos interesa más cubrir un rango amplio, que al menos cubra la zona vigilada, que la precisión de la medida. Parece, por tanto, que sería ideal que las cámaras estén suficientemente separadas entre sí para cubrir un rango amplio. Sin embargo, como se verá al analizar el cálculo de correspondencias, cuanto mayor sea la separación habrá una mayor probabilidad de oclusiones (regiones que sólo se ven desde una de las dos cámaras).

3.1.2. Correspondencia estereoscópica

El problema de la correspondencia es el de más difícil solución dentro de la visión estereoscópica. Se trata de decidir que puntos de dos imágenes estéreo son la imagen de un mismo punto real. Una vez establecido los puntos que son imagen de un mismo punto real, es posible ya obtener las coordenadas de dicho punto, mediante un proceso de triangulación como el mencionado en la sección anterior. El problema de la correspondencia es a priori

ambiguo. En principio, cualquier punto en el plano imagen I_2 podría corresponder a cualquier punto en el plano imagen I_1. Para resolver esta ambigüedad, podemos emplear ciertas restricciones, que pueden clasificarse en tres grupos:

- **Restricciones geométricas debidas a la configuración del sistema de imagen (posición relativa entre las cámaras):** La más importante es la restricción epipolar, que nos permite convertir una búsqueda bidimensional en unidimensional, con la consiguiente reducción en el coste computacional. También se engloba aquí el límite de disparidad, que permite reducir la búsqueda de correspondencias sobre la línea epipolar a un margen más pequeño. En ciertas ocasiones es posible que esta restricción no pueda aplicarse, porque no estén determinados los márgenes máximo y mínimo de las posibles profundidades de los objetos. Normalmente, la mayor restricción la impone la mínima distancia admisible, ya que si aceptamos distancias muy próximas a la cámara, la disparidad correspondiente a dicha distancia puede ser muy grande, obligándonos a evaluar un segmento de línea epipolar bastante extenso.

- **Restricciones geométricas propias de los objetos de la escena:** En este caso, se realizan suposiciones sobre los objetos de la escena, que en ocasiones pueden no ser ciertas. La más importante, y que se tiene en cuenta en todos los algoritmos de correspondencia estereoscópica, es la de unicidad, que establece que un punto de una imagen tiene un solo punto correspondiente en la otra. Otra suposición, que suele ser empleada por los algoritmos que no devuelven un mapa de disparidades denso, es el de continuidad, que establece que las profundidades varían suavemente, al menos en las superficies de los objetos, presentando discontinuidades únicamente en los bordes.

- **Restricciones físicas o fotométricas:** Ésta es la restricción principal para poder realizar la búsqueda de correspondencias entre imágenes. Por ejemplo, si lo que vamos a realizar es una búsqueda píxel a píxel podemos hacer la suposición de que la imagen generada por ese píxel y sus vecinos en ambas imágenes es la misma. Esto supone que el modelo empleado para la escena es el lambertiano (las superficies lambertianas reflejan igual en todas direcciones), lo que supone que las intensidades correspondientes a puntos correspondientes serían similares, independientemente del punto de vista.

De las restricciones anteriores, el primer grupo estaría ligado con la geometría epipolar descrita anteriormente, y con el proceso de calibración de las cámaras, y por tanto, son independientes de la escena. Por tanto, si la calibración se realizó correctamente, dichas restricciones serán ciertas siempre. El segundo grupo de restricciones dependen de la escena, y por tanto, no podemos asegurar que se cumplan en todo momento, aunque sí serán ciertas para la mayor parte de la escena capturada.

El tercer grupo de restricciones están basadas en modelos de interacción de los objetos con la iluminación. Estas restricciones son las empleadas por la mayor parte de algoritmos

para realizar la búsqueda de puntos correspondientes. Las restricciones deben aplicarse sobre entornos de vecindad, ya que los valores puntuales de intensidad en un píxel están sujetos al ruido.

Los algoritmos que no utilizan, o no utilizan únicamente la información de intensidad de los píxeles, establecen otras restricciones para primitivas de mayor nivel. Por ejemplo, los algoritmos que realizan la correspondencia entre los bordes de los objetos, deben imponer las restricciones de compatibilidad de orientación, longitud y gradiente del borde en ambas imágenes. Los algoritmos que establecen correspondencias entre regiones de la imagen emplean las restricciones de dimensión y forma de la región considerada.

3.1.3. Métodos de correspondencia

Como ya hemos visto, el objetivo de la correspondencia estereoscópica es el buscar parejas de píxeles, u otras primitivas de más alto nivel, que son la imagen de una misma primitiva en ambas imágenes. También se ha visto que la búsqueda de dichas correspondencias se ve simplificada enormemente gracias a una serie de restricciones impuestas tanto al sistema de cámaras como a la propia escena bajo vigilancia. Sin embargo, existen también algunos problemas que dificultan el proceso de correspondencia o *matching*. A continuación se describen algunos de estos problemas:

- **Variación fotométrica:** Aunque al tratar las restricciones fotométricas asumíamos ciertos modelos y propiedades, lo cierto es que la luz que recoge la cámara depende de la posición de la cámara respecto a la escena, del ruido que pueda introducirse durante la captura de la imagen y de las características de no linealidad en la cámara. Por tanto, si dos cámaras ven una escena desde distinto punto de vista, la intensidad en puntos correspondientes puede ser distinta, al contrario de lo que se asumía en el modelo de reflectancia lanbertiano. Para solucionar los problemas de ruido, como ya se vio, deben aplicarse las restricciones en entornos de vecindad. Por último, el tema de distorsiones que introduce la cámara puede reducirse escogiendo una cámara adecuada. En este sentido, debe tenerse en cuenta que en cámaras con una distancia focal muy pequeña (por debajo de 8.5 mm) la distorsión radial puede ser excesiva. No obstante, es necesario indicar que existen métodos que permiten realizar la correción de este tipo de distorsiones.

- **Oclusiones:** En escenas donde la diferencia de profundidades es elevada, es posible que parte de la escena sea vista por una única cámara. Esto implica que no todos los puntos de una imagen tienen su correspondiente en la otra imagen. El número de oclusiones aumenta al aumentar la distancia de separación entre las cámaras. Sin embargo, no hay que perder de vista que la distancia de separación debe ser grande si el rango de profundidades a medir es elevado.

- **Textura repetitiva:** La presencia en muchas escenas de texturas repetitivas hace que aparezcan múltiples puntos similares susceptibles de ser correspondientes. El

escenario, por tanto, tiene un papel importante y sobre el que no vamos a poder actuar. Los puntos erróneos habrá que descartarlos utilizando las restricciones vistas anteriormente, si bien el hecho de tener muchos puntos candidatos aumenta la probabilidad de obtener una correspondencia errónea.

- **Falta de textura:** En una zona de la escena sin ningún tipo de textura no será posible aplicar ningún tipo de correspondencia, y por tanto esas zonas quedarán sin correspondencia, o bien la correspondencia obtenida tendrá una gran probabilidad de ser errónea.

Para resolver todos estos problemas, existen distintos tipos de algoritmos, que pueden ser clasificados en las siguientes categorías [Pajares00]:

- **Basados en área:** Se lleva a cabo la correlación de los niveles de gris en ventanas de las distintas imágenes, considerando que en puntos correspondientes existirá una similaridad de niveles de intensidad luminosa, tal y como exponía la restricción fotométrica. Este tipo de métodos permite obtener mapas de profundidades densos, ya que evalúan todos los puntos de la escena. Sin embargo son difíciles de aplicar en zonas de poca textura y en bordes o en oclusiones, donde el número de correspondencias erróneas puede dispararse. También presentan como inconveniente la velocidad, ya que el coste computacional aumenta debido al gran número de puntos a evaluar.

- **Basados en características:** Extraen características de interés de la imagen (bordes, contornos, etc.) con las que realiza la correspondencia. En principio son más rápidos, ya que sólo usan un pequeño subconjunto de los píxeles de la imagen. También tienen la ventaja de presentar un porcentaje de correspondencias erróneas menor que otros métodos. Por contra, solamente proporcionan mapas de profundidad dispersos, pudiendo quedar áreas de interés sin evaluar. Además, pueden fallar si no se encuentran primitivas de interés en la imagen, con lo que su funcionamiento depende en gran parte de la escena.

3.1.4. Métodos basados en área

Los métodos de correspondencia estereoscópica basados en área consideran el par de imágenes estéreo como la misma señal con un ligero desplazamiento entre ambas. Por tanto, lo que persiguen estos algoritmos es buscar, para cada píxel de una imagen, la traslación que ha sufrido en la otra. Para ello, se minimiza un cierto criterio de semejanza, de modo que para cada píxel de una imagen se calcula la semejanza entre la distribución de intensidades de una ventana centrada en dicho píxel y una ventana del mismo tamaño centrada en el píxel a analizar de la otra, como se muestra en la figura 3.8. Aunque la función de semejanza puede tratarse de una correlación o una diferencia entre señales, en general, dicha ventana suele identificarse como la *ventana de correlación.*

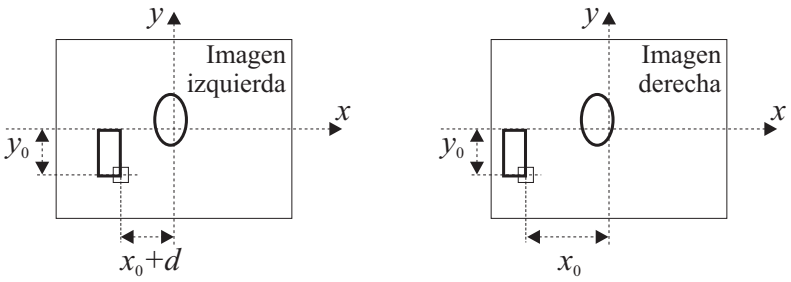


Figura 3.8: Técnica de búsqueda de correspondencias basadas en área.

Para el correcto funcionamiento de estos métodos, se deberá asumir que la intensidad reflejada por un objeto no depende del punto de vista, es decir, que las superficies son lambertianas, como suponía la restricción fotométrica vista anteriormente. De este modo, supondremos que las dos cámaras, situadas en lugares ligeramente distintos o desplazados, verán el mismo punto de la misma forma, con lo que al hacer la correlación encontraremos la semejanza máxima entre los dos píxeles correspondientes al mismo punto del escenario real.

Para que el cálculo de la función de similitud sea correcto, debe suponerse que todos los píxeles dentro de la ventana de correlación han sufrido el mismo desplazamiento. Esto solamente es cierto si la superficie es paralela a los planos imagen de las cámaras, es decir, toda la superficie tiene la misma profundidad. Si no se cumple esta condición, los resultados podrían ser erróneos. De hecho, en los bordes de los objetos con diferencias de profundidades elevadas, este tipo de algoritmos generalmente devolverá resultados incorrectos.

Para evaluar la similitud de la ventana de correlación, las funciones más comúnmente empleadas son las siguientes [Roma02]:

- **Suma de las diferencias de cuadrados (*SSD*):**

$$\mathrm{SSD}\,(d) = \sum_{x=1}^{N}\sum_{y=1}^{M}\left(I_{\mathrm{L}}\,(x,y) - I_{\mathrm{R}}\,(x+d,y)\right)^2$$

  donde N y M son respectivamente las filas y las columnas de la ventana de correlación.

- **Suma de las diferencias absolutas (*SAD*):**

$$\mathrm{SAD}\,(d) = \sum_{x=1}^{N}\sum_{y=1}^{M}\left|I_{\mathrm{L}}\,(x,y) - I_{\mathrm{R}}\,(x+d,y)\right|$$

- **Suma normalizada de las diferencias de cuadrados centradas respecto a la media (*ZNSSD*):**

$$\mathrm{ZNSSD}\,(d) = -\frac{\sum_{x=1}^{N}\sum_{y=1}^{M}\left(\left(I_{\mathrm{L}}\,(x,y)-\mu_{\mathrm{L}}\right)-\left(I_{\mathrm{R}}\,(x+d,y)-\mu_{\mathrm{R}}\,(d)\right)\right)}{\sigma_{\mathrm{L}}\sigma_{\mathrm{R}}\,(d)}$$

Estos tres primeros casos son una medida de diferencia entre señales. Por tanto, la correspondencia deberá buscarse entre los valores mínimos de la función. Los problemas que presentan las dos primeras funciones (*SSD* y *SAD*) es que sus valores se incrementan notablemente si el contraste de intensidades dentro de la ventana de correlación es alto. Por este motivo, y para compensar este efecto, la función *ZNSSD* normaliza con las varianzas de ambas ventanas. Otra ventaja de la función *ZNSSD* es que presenta mejores resultados cuando se emplean dos cámaras no idénticas con una diferencia importante de intensidades entre ambas cámaras. Al normalizar con respecto a la media de la ventana de correlación se consigue cierta invariancia a este problema. Si las cámaras son idénticas, o se emplea la misma cámara para tomar las dos imágenes, quizás no sería necesario normalizar respecto a la media, y únicamente normalizar con la varianza.

Para evaluar la similitud en la ventana de correlación las siguientes funciones están basadas en la correlación cruzada:

- **Correlación cruzada simple (*SCC*):**

$$\mathrm{SCC}\,(d) = \sum_{x=1}^{N}\sum_{y=1}^{M} I_{\mathrm{L}}\,(x,y)\,I_{\mathrm{R}}\,(x+d,y)$$

- **Correlación cruzada normalizada centrada en la media (*ZNCC*):**

$$\mathrm{ZNCC}\,(d) = \frac{\sum_{x=1}^{N}\sum_{y=1}^{M}\left(I_{\mathrm{L}}\,(x,y)-\mu_{\mathrm{L}}\right)\left(I_{\mathrm{R}}\,(x+d,y)-\mu_{\mathrm{R}}\,(d)\right)}{\sigma_{\mathrm{L}}\sigma_{\mathrm{R}}\,(d)}$$

En este caso, al tratarse de una correlación, la correspondencia deberá buscarse entre los valores máximos de la función. En general, las funciones normalizadas son menos sensibles a las variaciones de intensidad media y a los distintos parámetros de las cámaras cuando éstas son distintas. Por contra, al tratarse de expresiones más complejas, el tiempo de cómputo será mayor que para las funciones no normalizadas.

Tamaño de la ventana de correlación

Además del criterio de semejanza elegido, también el tamaño escogido para la ventana de correlación tiene gran importancia. La ventana de correlación debe ser lo suficientemente grande como para que la variación de intensidad que se produzca dentro de ésta,

y por tanto la energía de las señales que vamos a comparar, sea grande y no se vea por tanto afectada por el ruido. Por otro lado, debemos garantizar la restricción de continuidad para que todos los puntos dentro de la ventana de correlación se encuentren a la misma distancia. Si bien habrá lugares dentro de la imagen donde esto no vaya a cumplirse, especialmente en los bordes de los objetos, independiente del tamaño de la ventana, cuanto mayor sea la ventana de correlación mayor será la probabilidad de que la propia ventana incluya cambios de disparidad, por lo que mayor será la probabilidad de no cumplir con la restricción de continuidad. La consecuencia de esto último es un ensanchamiento de los bordes de los objetos, haciendo parecer más grande de lo que en realidad son [Hirschmuller01]. Por este motivo, es necesario que la ventana sea lo suficientemente pequeña para poder cumplir con la restricción de continuidad en la mayor parte posible de los píxeles.

La elección del tamaño de la ventana de correlación es por tanto, una relación de compromiso entre la relación señal a ruido y la distorsión. Si queremos tener una relación señal a ruido muy grande deberemos escoger una ventana de correlación grande, pero de esta forma la posición de los bordes quedará poco definida e introduciremos un error sistemático en las zonas de cambio de disparidad. Sin embargo, si escogemos una ventana pequeña, en las zonas donde no haya mucha textura, la relación señal a ruido será muy baja y tendremos multitud de errores debidos al ruido.

El problema es complejo, ya que en unas zonas nos interesará utilizar ventanas de correlación pequeñas y en otras zonas interesará utilizar ventanas de correlación más grandes. En las zonas donde se produzcan cambios de disparidad convendrá utilizar ventanas de correlación muy pequeñas, para así respetar la restricción de continuidad. Además, estas zonas son bordes, donde las variaciones de intensidad son grandes, y por tanto podemos emplear una ventana pequeña y mantener una relación señal a ruido aceptable.

En las zonas donde la disparidad no varía, o varía muy poco, es más adecuado utilizar ventanas de correlación grandes. En estas zonas es posible que haya bordes debido a la naturaleza del material que no tiene por qué ser uniforme, pero como no se producen cambios de disparidad podemos usar una ventana grande y garantizar que se cumple la restricción de continuidad. Sin embargo, la ventana de correlación no puede ser todo lo grande que se quiera, ya que si ésta es muy grande, el lóbulo de la función de correlación se ensancha, y será más complicado evaluar el punto exacto en el que se produce el máximo, o el mínimo.

En principio, el tamaño óptimo de la ventana depende de cada aplicación y debe seleccionarse empíricamente, pues su valor debería ser función de la variación de intensidad y la disparidad dentro de la ventana, valor que desconocemos. Sin embargo, existen varios algoritmos que utilizan distintos tamaños de la ventana de correlación para conseguir mejores resultados que los obtenidos con una ventana fija. La mayoría de los algoritmos proponen el uso simultáneo de ventanas de distinto tamaño. De hecho, algunos autores, como [Mansson98] proponen que incluso el sistema visual humano está formado células

que responden a distintas frecuencias espaciales, es decir, como si utilizara ventanas de distinto tamaño para analizar la imagen visualizada con distintos niveles de resolución.

Como se ha comentado anteriormente, las ventanas de tamaño más grande permiten obtener resultados menos precisos, pero más fiables. Por contra, las ventanas de tamaño pequeño obtienen resultados más precisos, pero con mayor probabilidad de error. En general, la técnica más empleada utiliza los resultados obtenidos con ventanas más grandes para guiar la búsqueda con ventanas más pequeñas. En [Lee97], [Chen00] o [Roma02] esto se lleva a cabo mediante la descomposición piramidal de la imagen, de tal forma que en un nivel determinado la imagen es una versión diezmada de la imagen del nivel inferior. De esta forma, en el primer nivel disponemos de la imagen original, y las imágenes de niveles superiores se obtienen mediante un diezmado por 2 de la imagen anterior. El valor de cada píxel de una imagen diezmada suele calcularse como el valor medio de los cuatro píxeles correspondientes del nivel anterior. Una vez obtenidos todos los niveles de la descomposición, los algoritmos calculan la disparidad en el nivel más alto, donde se obtendrán valores más fiables, aunque menos precisos. Los resultados obtenidos en un nivel son utilizados en la imagen del nivel inferior como una estimación inicial de la disparidad. Este refinamiento de la disparidad se realiza en sucesivas iteraciones hasta llegar al último nivel, que se corresponderá con la imagen original.

Otros algoritmos proponen el uso de ventanas no rectangulares. Por ejemplo, en [Cord98] o [Giraudon94] las ventanas de correlación tienen la forma de los objetos presentes en la imagen. Para ello, realiza una detección de los objetos presentes en una de las imágenes, y para cada uno de estos objetos, busca su correspondencia en la otra imagen.

En [Tang02] la imagen es dividida en distintas regiones mediante el diagrama de Voronoi. Las semillas empleadas para la construcción del diagrama de Voronoi lo forman el conjunto de correspondencias fiables de características obtenidas previamente. En [Hirschmuller01], se emplea una ventana rectangular central, y varias ventanas auxiliares adyacentes o solapadas con la ventana central. En el cálculo de la correlación se considera la contribución de la ventana central, y de las N *mejores* ventanas auxiliares, con la intención de que éstas se adapten a la forma de los objetos. De esta forma, aunque las ventanas son rectangulares, en realidad, la ventana total empleada al calcular la correspondencia no es en general rectangular.

Optimizaciones para el cálculo de la función de similitud

Uno de los principales problemas que plantean los métodos basados en área es su gran conste computacional, ya que supone el cálculo de la función de similitud para todos los píxeles de la imagen. Existen distintos trabajos que intentan reducir el tiempo de cómputo con distintos algoritmos. [Chen00] propone la existencia de tres estrategias distintas para conseguir dicha reducción. Las dos primeras categorías consiguen disminuir el tiempo de cómputo mediante la reducción del número de posiciones donde calcular la función de similitud. Los algoritmos de la primera categoría utilizan las técnicas del gradiente de la

función de similitud de una versión diezmada de la señal original para dirigir la búsqueda en la siguiente versión de mayor resolución. Los algoritmos de la segunda categoría reducen el tiempo de cómputo simplemente diezmando la imagen. Los algoritmos de estas dos primeras categorías no garantizan, sin embargo, la búsqueda del mínimo absoluto, ya que no son analizadas todas las posiciones posibles.

Los algoritmos de la tercera categoría utilizan algún criterio para eliminar aquellas posiciones que no deben ser evaluadas, a la vez que garantizan que la posición obtenida va a ser el mínimo o máximo absoluto. Por ejemplo, [Chen00], en la búsqueda del mínimo de la función de correlación, anula aquellas posiciones cuya suma parcial, después de cada iteración, supera al mínimo absoluto en cada momento. Este algoritmo, aun siendo bastante complejo, reduce el número de operaciones en el cálculo de la función de similitud, aunque sólo puede ser empleado con funciones que sean medida de la diferencia entre dos señales, como son las funciones *SSD*, *SAD* o *ZNSSD* descritas anteriormente.

La estrategia de descomposición piramidal descrita anteriormente también permite la reducción del tiempo de cómputo, ya que como se comentó, las correspondencias obtenidas en niveles superiores, aparte de guiar la búsqueda en los niveles inmediatamente inferiores, evita realizar la búsqueda en una parte importante de la señal.

Sin embargo, la estrategia más comúnmente empleada para la reducción de tiempos es la técnica de *Box-Filtering* [DiStefano04]. En este caso, el tiempo de cálculo se reduce gracias a que esta técnica evita la realización de operaciones redundantes. Es decir, si la ventana de correlación es de tamaño $(N+1) \times (N+1)$, para cada píxel y disparidad de la imagen, el número de operaciones a realizar sería de $(N+1)^2$. Con la técnica *Box-Filtering*, para cada posición, el número de operaciones se reduce a únicamente 4, independientemente del tamaño de la ventana, con lo que la reducción del tiempo de cómputo es claramente evidente, aunque requiere mayor capacidad de almacenamiento para guardar resultados temporales.

Aparte de todas las posibles optimizaciones de los algoritmos de correspondencias comentadas anteriormente, diversos autores han dedicado sus esfuerzos a la redución del tiempo de cálculo aprovechando las posibilidades que ofrece la tecnología actual, como pueden ser la programación en paralelo mediante instrucción SIMD (*Single Instruction Multiple Data*), el uso de la tarjeta gráfica del ordenador, a través de la tecnología conocida como GP-GPU (*General Purpose computing on Graphics Processing Units*), o diversas otras opciones. Por ejemplo, en [Cuadrado06], se implementa un sistema de visión estereoscópica, basado en el algoritmo SAD, sobre una FPGA (*Field Programmable Gate Arrays*).

Chequeo de consistencia del mapa de disparidad

Un último problema a tener en cuenta con los algoritmos de área es su probabilidad de error, especialmente alta cuando el ruido es elevado, o la escena no es rica en texturas. En este caso, es posible que la función de similitud evaluada tenga más de un pico, y es

posible que la posición correcta no se corresponda con el máximo, o mínimo, absoluto, y por tanto se produzca un error al establecer la correspondencia de dicho píxel. Este tipo de errores también ocurrirá en las oclusiones, aun cuando la relación señal a ruido sea elevada.

Para comprobar la validez del mapa de disparidades obtenido, suele emplearse el chequeo de la consistencia izquierda - derecha. Hasta el momento, se ha considerado que una de las imágenes, en general la izquierda, es la imagen de referencia, y se calcula la función de correlación desplazando la ventana de correlación en la imagen derecha. El chequeo de la consistencia izquierda - derecha calcula tanto la correspondencia directa (izquierda - derecha) como la correspondencia inversa (derecha - izquierda). La denominación directa e inversa es la utilizada por la mayoría de los autores, ya que tradicionalmente la imagen izquierda era la que se utilizaba como referencia.

Aunque pueda parecer que el número de operaciones se dobla, ya que supuestamente hay que obtener la función de similitud para el doble de píxeles, en realidad el número de operaciones es el mismo. Supóngase que realizamos el cálculo de la función de similitud directa para el píxel 0 (imagen izquierda) y, en lugar de obtener el máximo y descartar toda la señal, la almacenamos en un array, como en la figura 3.9, que se correspondería con la primera línea de puntos horizontal empezando por abajo. Para simplificar la demostración, consideremos por ejemplo que el rango de disparidades se encuentra entre 0 y 5, es decir hay que desplazar la ventana de correlación únicamente 6 posiciones. Posteriormente, calculamos la función de similitud para el píxel 1, y lo volvemos a almacenar, y así sucesivamente hasta obtener la función de similitud para todos los píxeles de la imagen izquierda. La tabla así obtenida se denomina *Tabla de correspondencias* [DiStefano04]. De la figura 3.9 podemos comprobar fácilmente que las líneas horizontales se corresponden con la función de similitud directa (izquierda - derecha), mientras que las líneas verticales se corresponden con la función de similitud inversa (derecha - izquierda).

Para realizar el chequeo del mapa de disparidades obtenido, se puede seguir un esquema como el mostrado por [DiStefano04]. De acuerdo con el teorema de unicidad, cada píxel de la línea izquierda debe tener su correspondiente en la línea derecha, que si por ejemplo la función de similitud es la función SAD, se corresponderán con los mínimos de cada una de las líneas horizontales de la tabla de correspondencias. Esta primera fase se denomina fase directa. Posteriormente hacemos lo mismo para cada uno de los píxeles de la línea derecha buscando su correspondiente en la línea izquierda, en lo que se denomina la fase inversa. Finalmente, tomamos como correspondencias válidas únicamente aquellas que fueron marcadas tanto en la fase directa como en la inversa. Este procedimiento nos va a devolver un mapa de disparidades disperso, ya que algunas correspondencias van a ser eliminadas tras el chequeo de consistencia. Esto no es del todo un inconveniente, ya que las oclusiones, ya sean en la imagen izquierda o en la derecha, que evidentemente no tienen su correspondiente en la otra imagen, teóricamente deberían ser eliminadas. Si se desea obtener un mapa de disparidades denso, se puede emplear cualquier algoritmo de relajación, como se verá posteriormente.

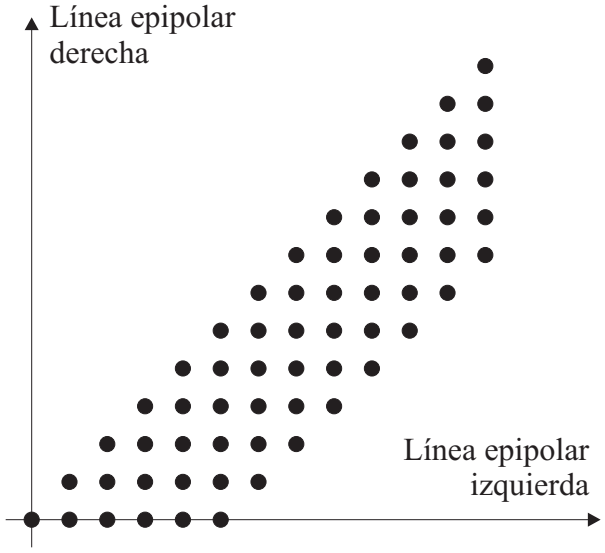


Figura 3.9: Tabla de correspondencias

Además de lo anterior, la *Tabla de Correspondencias* permite también realizar un refinamiento de las correspondencias finales, para poder alcanzar una precisión subpixel. Una de estas técnicas se conoce como SPF (*Symmetric Parabola Fitting*), que permite obtener la posición subpíxel del máximo a partir de los valores de las funciones de similitud alrededor del máximo. En [Lim08], se propone una variante, denominada APF (*Asymmetric PF*), empleando como función de similitud la correlación cruzada normalizada (*ZNCC*).

3.1.5. Métodos basados en características

Los problemas que plantean los algoritmos de búsqueda de correspondencia basados en área son principalmente dos. Por un lado, su gran coste computacional, aun a pesar de las optimizaciones comentadas anteriormente. Por otro lado, los algoritmos basados en área obtienen resultados incorrectos en zonas donde hay cambios de disparidad grandes u oclusiones. Esto último se produce generalmente en los bordes de los objetos, lo que suele resultar, en general, en un aumento del grosor de los objetos. Es necesario resaltar que este aumento del grosor sólo se produce en la dirección de la línea base que une las cámaras. Es decir, si por ejemplo las cámaras están separadas horizontalmente, los objetos sólo se ensanchan horizontalmente, pero nunca lo harían verticalmente.

En la figura 3.10 se encuentra ilustrado este problema. Si consideramos la imagen del objeto O en las imágenes izquierda y derecha, el resultado que obtendríamos sería parecido al mostrado en la figura 3.11. Al calcular ahora la función de correlación para el punto P de la imagen izquierda en las posiciones P_1 y P_2, parece claro que, en la situación actual, el mínimo se producirá en la posición P_2. Sin embargo, si observamos la figura 3.10 nuevamente, podemos comprobar como la posición correcta debería ser P_1. Este error en

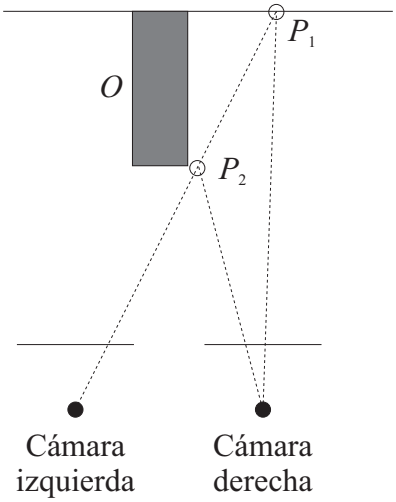


Figura 3.10: Problema típico ante oclusiones de objetos

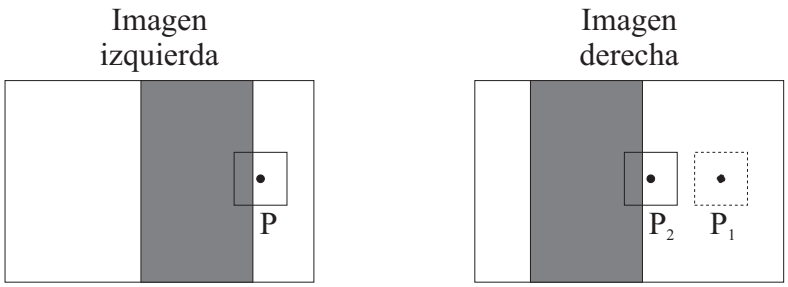


Figura 3.11: Error en el cálculo de disparidad ante oclusiones de objetos

la decisión de la correspondencia va a provocar, por tanto, el ensanchamiento aparente del objeto.

Los algoritmos de búsqueda de correspondencias basados en características van a tratar de resolver ambos problemas. El problema del tiempo de cómputo se resuelve gracias a que el número de posiciones a evaluar va a ser mucho menor (sólo aquellas características obtenidas en la imagen). La precisión en la estimación de los bordes de los objetos se consigue gracias a que estos algoritmos obtienen los bordes, esquinas, u otras primitivas de mayor nivel, que pueden aportar una información mucho más precisa que los niveles de intensidad de las imágenes.

El principal problema de estos algoritmos es que no proporciona mapas de disparidad densos, sólo proporciona la disparidad de aquellas características obtenidas en la imagen. Si se desearan mapas densos, se necesitaría del empleo de técnicas de relajación, que en general suelen ser procesos iterativos, que podrían hacer que el tiempo de cómputo fuera mayor incluso que el necesitado por los algoritmos basados en área.

Además, como estos algoritmos funcionan únicamente con características, es necesario que dichas características existan en la imagen, y por tanto, es necesario un conocimiento previo de la escena. Por ejemplo, si el algoritmo establece correspondencias únicamente entre líneas rectas, extraídas de los bordes de los objetos, es necesario que en la escena existan figuras geométricas, o algún otro tipo de objeto que presente este tipo de característica.

En general, cuanto más complejas sean las características empleadas, mayor va a ser la precisión del sistema, y menor la probabilidad de error. Supongamos por ejemplo un sistema que trata de detectar señales de tráfico, y mediante un sistema de cámaras binocular como los vistos hasta ahora, determinar la distancia a la que se encuentra la señal. Si el sistema de detección de señales de tráfico funciona correctamente, el establecer la correspondencia para dicha señal entre ambas imágenes es casi inmediato. Los algoritmos de correspondencia estereoscópica basados en características complejas van a depender de la aplicación en sí.

Los algoritmos basados en características más genéricas suelen emplear principalmente los bordes o las esquinas de los objetos para realizar la correspondencia. Quizás uno de los algoritmos más conocidos de correspondencia estereoscópica es el desarrollado por Marr y Poggio [Ghasemlou01]. Marr presentó uno de los estudios más importantes sobre el sistema visual humano. Dentro de esta teoría se incluye el proceso de visión estereoscópica. Lo que Marr sugiere es que el sistema visual humano utiliza detectores de paso por cero basados en filtros de Laplaciana de la Gaussiana (LoG, *Laplacian of Gaussian*), con diferentes escalas. En un intento de imitar el funcionamiento del sistema visual humano, el algoritmo de Marr realiza una detección de bordes mediante un filtro LoG. Posteriormente, trata de establecer correspondencias entre píxeles de borde de la imagen izquierda con los píxeles de borde de la imagen derecha [Ghasemlou01]. El mapa de profundidades así obtenido será un mapa disperso, ya que sólo proporciona información de disparidad en los puntos detectados como bordes. Para obtener un mapa de disparidades denso será necesario el uso de técnicas de relajación, como se verá posteriormente.

En [Boufama02], se realiza una detección de bordes para la imagen izquierda, y se busca la correspondencia de cada píxel de borde con su correspondiente en la imagen derecha. Como los bordes de los objetos son ricos en textura, es poco probable que existan errores de correspondencia en este proceso. Dentro de cada línea epipolar, los píxeles que no fueron marcados como bordes, y que por tanto todavía no tienen correspondencia, estarán siempre comprendidos entre dos píxeles de borde, que si tienen su correspondencia establecida. Por tanto, la búsqueda de correspondencias para estos segmentos se reduce al rango de píxeles establecido por los píxeles de borde entre los que se encuentra dicho segmento.

Los algoritmos de correspondencia basados en caracteristicas han sido ampliamente utilizados en el mundo de la robótica para guiar elementos móviles a través de un entorno más o menos estructurado. Una de las técnicas más populares consiste en el guiado a partir de las imágenes capturadas por un sistema binocular de cámaras que lleva implantado

el robot. Para la navegación correcta del robot, éste necesita de una implementación denominada SLAM (*Simultaneous Localization and Mapping*. En [Gao07] y [Dailey06] las características empleadas son los bordes o líneas obtenidas de la imagen mediante un detector de Canny. En [Herath06], se emplean distintas características, que deben ser validadas mediante un algoritmo que analiza el histograma de los píxeles en una ventana centrada en la posición de dicha característica. Sólo aquellas características que superen este proceso de validación son incluidas para obtener la estructura en tres dimensiones del entorno, y permitir por tanto la navegación del robot. La visión estereo también puede ser utilizada para el guiado de robots industriales. En [Oh07] se emplea para el guiado y posicionamiento de un brazo mecánico. En este caso, al encontrarse el sistema de visión en un entorno totalmente controlado, no es ningún problema la búsqueda de características.

3.1.6. Técnicas de relajación

Como se ha visto en las secciones anteriores, existen algoritmos que no devuelven un mapa de disparidades denso, por lo que se hace necesario un procesado posterior que nos permita refinar o mejorar el resultado. Éste es el objetivo de las técnicas de relajación.

La idea básica de funcionamiento de las técnicas de relajación es que, a partir de una hipótesis inicial para las correspondencias, permitimos que éstas se reorganicen propagando alguna restricción, como la de continuidad, la de orden o la de unicidad. La restricción de unicidad establece que un píxel en una imagen sólo tiene un píxel correspondiente en la otra. Si observamos la figura 3.12, y consideramos el píxel p, si ese píxel fuese una correspondencia correcta, ninguno de los píxeles situados en la misma línea horizontal o vertical podrían ser una correspondencia válida, ya que haría que un mismo píxel en una imagen tuviera dos correspondencias en la otra imagen. La zona formada por la línea horizontal y vertical suele denominarse *Zona X*, debido a su forma [Sara02].

La restricción de orden establece que los puntos correspondientes aparecen en ambas imágenes en el mismo orden al desplazarnos de un lado al otro de la línea epipolar. Esto no siempre es cierto, ya que en presencia de objetos estrechos, el orden de aparición de los objetos puede cambiar. Por tanto, para la restricción de orden puede ser o no necesaria su aplicación. Si consideramos de nuevo en punto p de la figura 3.12, los puntos que no cumplirían la restricción de orden serían todos los situados en el cuadrante inferior derecho, y los situados en el cuadrante superior izquierdo de la zona X. Esta zona suele ser denominada como *Zona F* (de *Forbidden*).

Por último, la restricción de continuidad establece que puntos adyacentes deberán presentar una disparidad similar. Si consideramos nuevamente el punto p de la figura 3.12, los puntos que cumplirían la restricción de continuidad serían los puntos más cercanos al punto p situados en el cuadrante superior derecho, y en el cuadrante inferior izquierdo. Esta zona suele denominarse como *Área de soporte*. Por ejemplo, los puntos que están en la misma diagonal que el punto p serían puntos con la misma disparidad, o que están a la misma distancia, que el punto p.

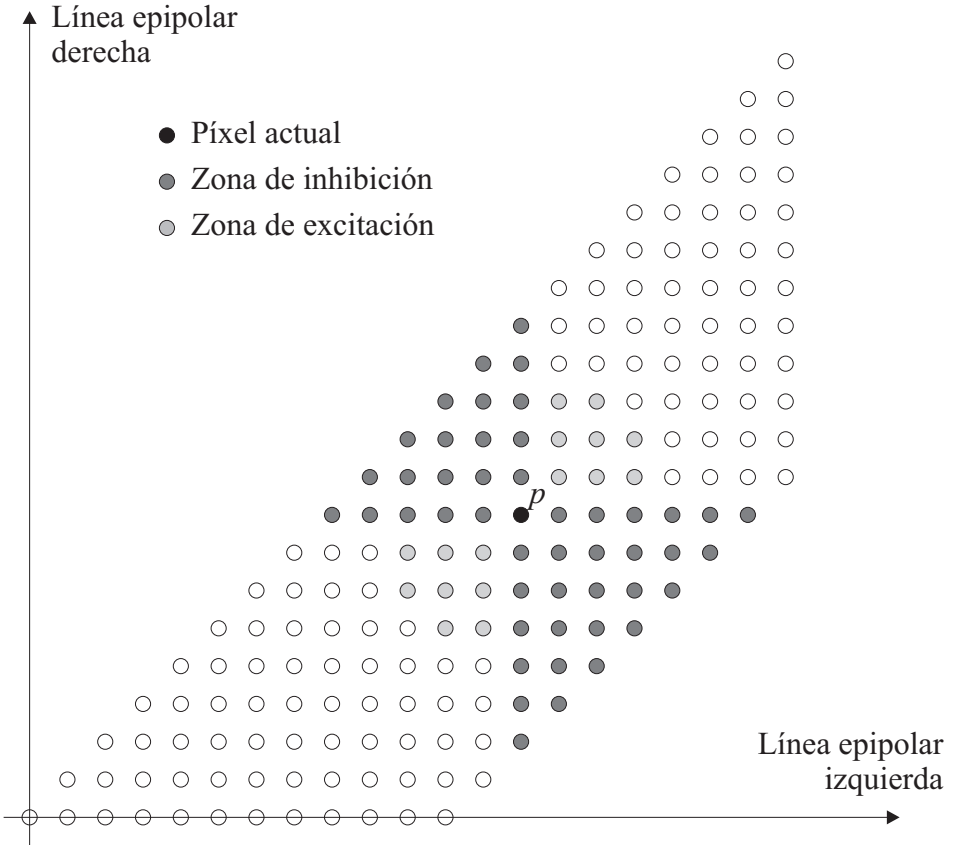


Figura 3.12: Técnicas de relajación: Zonas de inhibición y de excitación

A partir de todo lo anterior, podemos crear una zona de inhibición, formada por la *zona X* y la *zona F*, y una *zona de excitación*, formada por el área de soporte. Estas zonas así definidas pueden ser utilizadas para el desarrollo de algoritmos, en general iterativos, que mejoran o refinan el mapa de disparidad inicial.

Por ejemplo, en [Ghasemlou01], la tabla de disparidades es actualizada de la siguiente forma:

$$C_{x,y,d}^{(n+1)} = \sigma\left(\sum_{x',y',d' \in E}\left(C_{x',y',d'}^{(n)}\right) - \varepsilon \sum_{x',y',d' \in I}\left(C_{x',y',d'}^{(n)}\right) + C_{x,y,d}^{(n)}\right)$$

donde $C_{x,y,d}^{(n)}$ representa el estado de la correspondencia del píxel (x, y) con el píxel $(x+d, y)$ en la iteración n. E se corresponde con la zona de excitación para el punto p de la tabla de correspondencias considerado, e I la zona de inhibición. σ es una función de umbralización, que toma los valores 0 ó 1, y ε es la constante *de inhibición*. El estado inicial de la tabla de correspondencias toma los valores 1 ó 0. En concreto, $C_{x,y,d} = 1$ si $L(x, y) = R(x + d, y)$, es decir, si el valor de la intensidad de las imágenes coincide, y 0 si no coinciden. Este algoritmo no funciona correctamente con imágenes naturales, donde existan gran cantidad de disparidades, pero históricamente fue una de las primeras propuestas de este tipo de algoritmo, y suele ser considerado como clásico.

En [Zitnick99] se propone un nuevo algoritmo iterativo que mejora el funcionamiento del anterior con imágenes reales. En concreto, el proceso de actualización, siguiendo la nomenclatura utilizada antes, sería:

$$C_{x,y,d}^{(n+1)} = C_{x,y,d}^{(0)}\left(\frac{S_n(x, y, d)}{\sum_{x',y',d' \in I} S_n(x', y', d')}\right)^{\alpha} \tag{3.4}$$

donde

$$S_n(x, y, d) = \sum_{x',y,d' \in E} C_{x',y',d'}^{(n)} \tag{3.5}$$

indica la cantidad de excitación para el punto (x, y, d). El denominador de la expresión anterior indica la cantidad de inhibición para dicho punto. Por tanto, para un punto determinado, si la cantidad de excitación es alta, mientras que la cantidad de inhibición es baja, el valor de la correlación para dicho punto $C_{x,y,d}$ aumentará respecto a su estimación inicial $C_{x,y,d}^{(0)}$, o viceversa. El parámetro α controla la cantidad de inhibición en cada iteración. Para garantizar que el cálculo anterior converja a 1, α debe ser mayor de 1.

En [Sara02], la tabla de correspondencias $C_{x,y,d}^{(0)}$ se obtiene mediante el cálculo de la correlación cruzada normalizada empleando una ventana de 5×5 píxeles. Un algoritmo iterativo va eliminando los puntos de la tabla de correspondencias menos probables, y añadiendo a otra tabla las correspondencias fiables, empleando únicamente la zona de inhibición. El algoritmo termina cuando cada punto ha sido eliminado o bien ha sido

añadido a la tabla de correspondencias fiables. Este algoritmo no devuelve un mapa de disparidad denso, sino que su objetivo es encontrar el máximo número de correspondencias no ambiguas, o fiables.

Otra técnica empleada en la literatura es el uso de la programación dinámica. Supongamos que sobre una línea epipolar se ha obtenido su tabla de correspondencias, tal y como se muestra en la figura 3.13. El mapa de disparidades puede obtenerse a partir de la búsqueda del camino más corto entre los dos extremos de la línea epipolar (ver coeficientes marcados en negro en la figura anterior). Dicha búsqueda está ilustrada en la figura 3.14. En la tabla original, está representado el coste que supone pasar por cada nodo. El objetivo es buscar el camino de menor coste para ir desde la parte baja de la tabla hasta llegar a la parte alta de la tabla, teniendo en cuenta que sólo podemos desplazarnos en sentido vertical hacia arriba, o en sentido diagonal hacia arriba a la derecha o hacia arriba a la izquierda. Para ello se construye la tabla de costes totales de la derecha, donde el valor representado en cada celda comprende el coste de dicha celda más el coste mínimo para llegar desde dicha celda hasta la parte baja de la tabla. Las técnicas de programación dinámica permiten no solo obtener dicho camino, sino obtenerlo de una forma computacionalmente eficiente. Detalles sobre la implementación puede obtenerse por ejemplo en [Wagner95]. En [Cox96] se propone utilizar la programación dinámica para obtener, para cada línea epipolar, el camino de menor coste. La tabla de correspondencias es calculada a partir de únicamente el nivel de intensidad de cada píxel, en lugar de emplear la ventana de correlación vista hasta ahora. El principal problema de este algoritmo es que los bordes horizontales no quedan perfectamente definidos.

En [Sun02], la tabla de correspondencias se calcula a partir de la función de correlación cruzada normalizada *ZNCC*. Posteriormente, la aplicación del algoritmo de programación dinámica aplicado en primer lugar en sentido vertical, y posteriormente en sentido horizontal, permite obtener la superficie de menor coste. Este algoritmo presenta mejores resultados en los bordes horizontales que el algoritmo anterior. En [Zhao06], la programación dinámica se emplea para mejorar el mapa de disparidades en los bordes de los objetos, especialmente en aquellos puntos donde existan oclusiones.

3.1.7. Sistema implementado

Como ya se ha comentado anteriormente, existen dos métodos fundamentales para la obtención de correspondencias entre imágenes estéreo, las basadas en área, y las basadas en características. En esta tesis, vamos a centrarnos únicamente en la búsqueda de correspondencias basadas en características, ya que es fácilmente comprobable que presentan una mayor precisión y robustez frente a los métodos basados en área. Si bien el principal inconveniente de los algoritmos basados en características es precisamente la búsqueda de dichas características, en nuestro caso las características van a ser los objetos de primer plano detectados, por lo que si, la detección ha sido correcta, no vamos a tener ningún problema de falta de características. De hecho, si no existen características, significa que

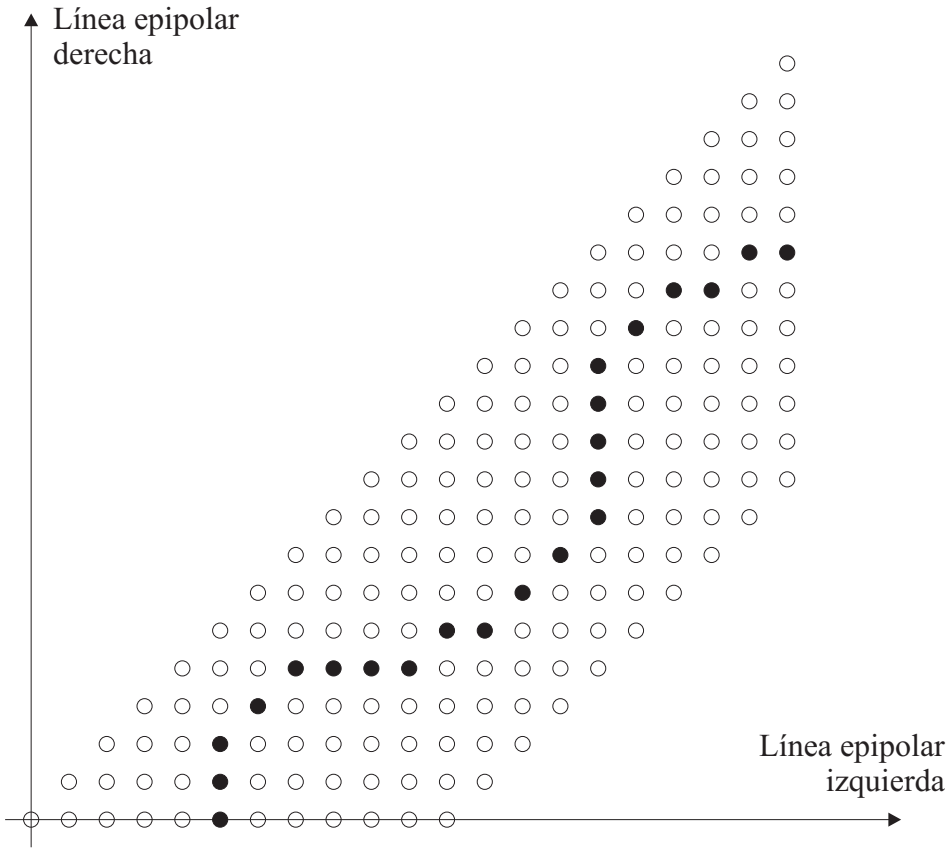


Figura 3.13: Superficie 3D de menor coste obtenida mediante programación dinámica

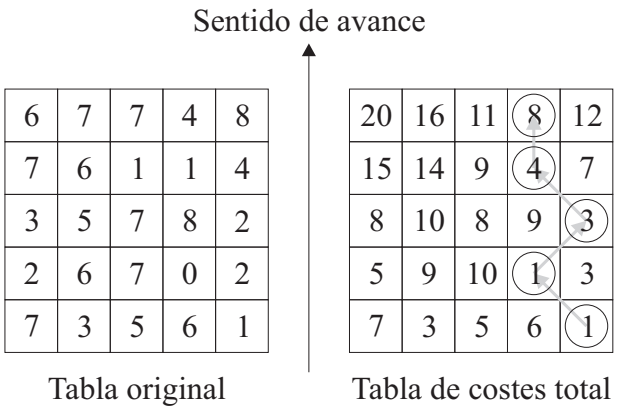


Figura 3.14: Búsqueda del camino de menor coste mediante programación dinámica

no hay objetos de primer plano, y por tanto, no sería necesario realizar ninguna estimación de distancias.

De esta forma, nuestro algoritmo va a realizar la búsqueda de correspondencias únicamente para los objetos detectados, ignorando por tanto el resto de la imagen. La principal ventaja de este método es que podemos centrarnos únicamente en los píxeles pertenecientes a los objetos, por lo que, por un lado, el tiempo de procesado va a ser mucho menor, y por el otro, ciertas imprecisiones que presentan los algoritmos de área van a poder ser corregidos al evaluar únicamente los píxeles de los objetos de primer plano. Hay que tener en cuenta que, como en los sistemas de videovigilancia el interés principal suele residir en los objetos de primer plano, no va a haber ningún problema en que el algoritmo de visión estereoscópica devuelva un mapa de profundidades dispersa, ya que aquellos puntos donde no haya información de profundidad corresponderán al fondo.

El principal inconveniente de los algoritmos de búsqueda de correspondencias, tal y como se ha comentado anteriormente, es el ruido de las imágenes, que puede provocar imprecisiones en la distancia estimada, o incluso errores de correspondencias. Si bien en algunas de las aplicaciones de la visión estereoscópica existentes esto no ha sido considerado un problema importante, ya que dichos errores pueden ser corregidos, en nuestro caso, un error de correspondencia para un objeto determinado puede ser fatal. Por este motivo, a la hora de diseñar el algoritmos de búsqueda de correspondencias, debemos asegurarnos principalmente de que disponga de una gran robustez, aun a costa de aumentar la complejidad computacional.

Así pues, el sistema a diseñar básicamente tratará de establecer relaciones entre objetos detectados en ambas imágenes. Teniendo en cuenta que el algoritmo de detección de los objetos de primer plano también puede presentar errores, se nos pueden plantear varios posibles casos:

- El objeto ha sido detectado en ambas imágenes. En este caso, se podrá realizar la búsqueda directa e inversa, y ambas informaciones podrán servirnos para refinar la estimación de la distancia.

- El objeto ha sido detectado sólo en una imagen. Sin embargo, será de esperar que el objeto también sea visible en la otra imagen, aunque no haya sido detectado. Si es así, sólo podrá realizarse la búsqueda de correspondencia directa.

- El objeto detectado sólo se ve en una imagen. Si el objeto no se ve en la otra imagen, entonces se trataría de una oclusión, y será imposible encontrar la correspondencia en la otra imagen. Si esto ocurriera, el sistema no podría determinar la distancia, pero al menos el objeto sí sería detectado.

El tema de las oclusiones es un problema bastante frecuente en los algoritmos genéricos de visión estereoscópica, donde no se tiene ningún conocimiento, y sobre todo, ningún control sobre la escena observada. En videovigilancia, aunque seguimos sin tener ningún

control sobre la escena, lo que sí es posible hacer es colocar las cámaras en los puntos que permitan evitar las oclusiones. Como ya se ha comentado anteriormente en esta tesis, éste es uno de los pasos más importantes en el diseño del sistema de videovigilancia, es decir, el colocar las cámaras en las posiciones idóneas. Además, como las cámaras son fijas, y el fondo puede ser conocido a priori, el sistema puede tener constancia en todo momento de cuales son los posibles puntos de oclusión, y esto puede ser indicado manualmente por el operador durante la configuración del sistema. Además, esta taréa solo debe ser realizada una única vez.

Búsqueda de correspondencias

Tal y como se ha comentado anteriormente, la búsqueda de correspondencias dentro de la escena vigilada se reduce únicamente a aquellos puntos donde ha sido detectado algún objeto. Además de reducir el tiempo de cálculo, esto nos permite cumplir con la restricción de continuidad enumerada anteriormente (pág. 96), ya que en este caso sí que podemos asegurar que las disparidades van a variar suavemente a lo largo de todo el objeto.

Por otro lado, como un sistema de videovigilancia va a trabajar principalmente en entornos exteriores, donde la iluminación de la escena no va a estar completamente controlada, es importante tener en cuenta aquí que un mismo objeto puede verse con distinto nivel de gris, o color si trabajamos con escenas en color, por cada una de las cámaras. Por tanto, el esquema general de búsqueda de correspondencias, que tiene en cuenta principalmente la restricción fotométrica (pág. 96) descrita anteriormente, no va a ser de gran utilidad aquí.

Una de las principales técnicas de procesado de imágenes que permite ser más inmune a estos cambios en los niveles de iluminación es la evaluación únicamente en los bordes de los objetos, en lugar de en todo el objeto. Aunque el objeto se vea con un nivel distinto en una y otra imagen, lo que en general no suele cambiar son los bordes de estos. Así pues, nuestro algoritmo va a realizar la búsqueda de correspondencias teniendo en cuenta únicamente los bordes de los objetos detectados. En este caso, no es necesario realizar una detección de bordes como se hacia en algún otro algoritmo descrito anteriormente, ya que los bordes de los objetos aquí se corresponderán con los contornos de las máscaras detectadas por el algoritmo de detección de movimiento.

Además, para conseguir mayor robustez frente a posibles diferencias de intensidad en el caso de que se utilicen cámaras distintas, o con distintos parámetros de configuración, las funciones de similitud ideales son aquellas que normalizan respecto a la media, tales como la función *ZNSSD*. En concreto, en esta tesis vamos a utilizar la correlación cruzada normalizada centrada en la media *ZNCC*, de acuerdo a:

$$\mathrm{ZNCC}\,(d) = \frac{\sum_{x,y \in V} \left(I_{\mathrm{L}}\,(x,y) - \mu_{\mathrm{L}}\right)\left(I_{\mathrm{R}}\,(x+d,y) - \mu_{\mathrm{R}}\,(d)\right)}{\sigma_{\mathrm{L}}\sigma_{\mathrm{R}}\,(d)} \tag{3.6}$$

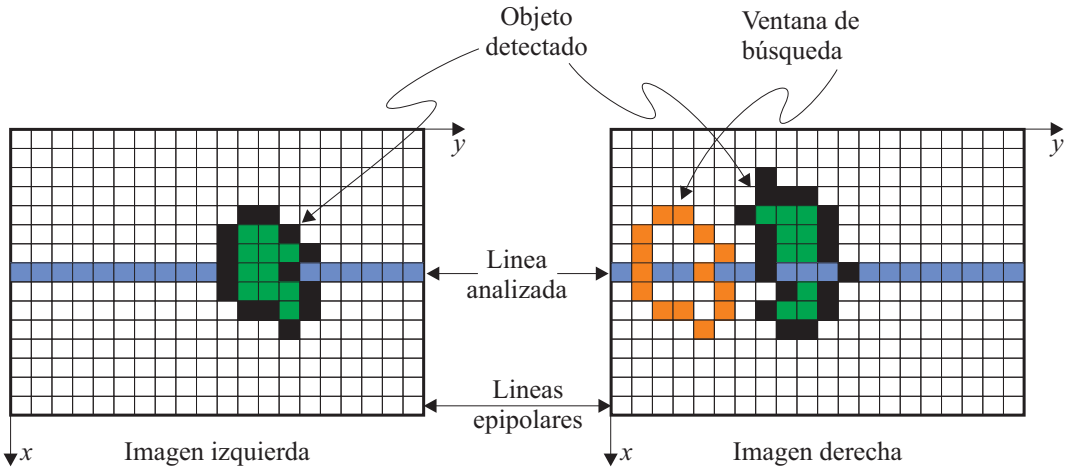


Figura 3.15: Búsqueda de correspondencias con el algoritmo propuesto. En este caso, el objeto ha sido detectado en ambas imágenes. En la imagen se muestra el cálculo de correspondencias para el objeto de la imagen izquierda. Sobre la imagen derecha, la ventana de búsqueda, que tiene la forma del contorno del objeto de la imagen izquierda, se desplaza horizontalmente para buscar el pico de la función de similitud empleada.

donde en este caso, el sumatorio se extiende únicamente a los píxeles pertenecientes al contorno del objeto detectado. Supongamos por ejemplo, que el sistema ha detectado los objetos que se muestran en la figura 3.15 para la imagen izquierda y la derecha, tal y como se indica. En verde estaría representado el cuerpo del objeto, y en negro sus bordes, que se corresponden con el contorno de la máscara. De esta forma, para calcular la correspondencia del objeto de la izquierda en la imagen derecha desplazamos una ventana con la misma forma que el contorno del objeto sobre la otra imagen, como se muestra con la ventana naranja en la imagen derecha, en la figura 3.15.

Una vez desplazada la ventana a lo largo de la línea epipolar, tendríamos calculada una de las columnas de la tabla de correspondencias para dicha línea, como puede verse en la figura 3.16. De hecho, ésta será la única columna a calcular, ya que sólo existe un objeto en la imagen izquierda. Sin embargo, como el objeto tiene un tamaño de varios píxeles, y es de esperar que la disparidad de todos sus píxeles sea la misma, podemos concluir que las funciones de similitud de las sucesivas columnas a la derecha de la columna anterior sean las mismas. Por ejemplo, en la figura 3.16 se muestra en naranja los coeficientes de la tabla de correspondencias que realmente se han calculado con la disposición de la ventana de búsqueda mostrada en la figura 3.15. Nótese que los coeficientes con el mismo valor para la función de similitud están dispuestos en diagonal, ya que en la tabla de correspondencias los coeficientes con el mismo valor de disparidad se sitúan en diagonal.

De igual forma, se podría calcular la función de similitud para el objeto de la imagen derecha, y así obtener la fila de la tabla de correspondencias de la figura 3.16. Es interesante destacar aquí que las funciones de similitud directa e inversa aquí no toman el mismo valor,

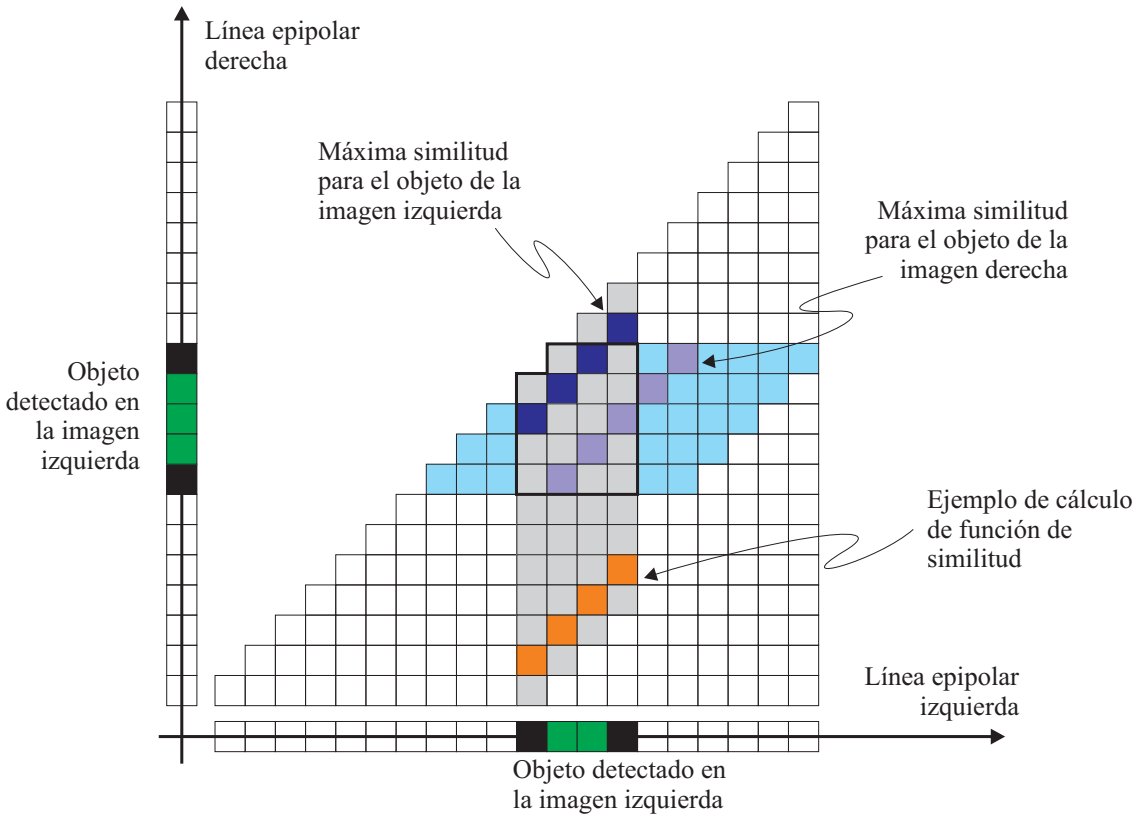


Figura 3.16: Tabla de correspondencias para el ejemplo de la figura 3.15. En la imagen, se muestran en color los coeficientes de los cuales se ha calculado su función de similitud. El resto de coeficientes no se calculan, ya que no pertenecen a ningún objeto detectado. En naranja se muestran los coeficientes que se están calculando en la figura 3.15.

ya que el cálculo para las filas y las columnas se realizará, en general, con distintos píxeles. De hecho, la expresión (3.6) no sería correcta para el cálculo de la función de similitud de los objetos en la imagen derecha, ya que ésta sólo permite calcular la disparidad tomando como referencia la imagen izquierda. En concreto, la función correspondiente sería la misma, únicamente cambiando los subíndices L (izquierda) por R (derecha), y viceversa.

Una vez calculadas todas las filas y columnas necesarias de la tabla de correspondencias, es decir, tantas como objetos hayan sido detectados en esa línea, se realiza la búsqueda de los picos de dicha función, máximos si la función es una correlación, y mínimos si se trata de una medida de diferencia entre señales. Por ejemplo, si nos fijamos nuevamente en la figura 3.16, se muestra en diferentes colores los picos para los objetos detectados en cada una de las imágenes.

Si ambos objetos en las imágenes izquierda y derecha son la proyección del mismo objeto de la escena, sería de esperar que, sino con la misma disparidad, ambos picos queden con disparidades muy próximas entre sí. Si esto es así, entonces ya podemos suponer finalmente que ambos objetos pertenecen al mismo objeto real. Para determinar si ambos objetos tienen la misma disparidad, fijémonos de nuevo en la figura 3.16, donde ha sido definido el área I como la intersección de la proyección de los objetos en la imagen izquierda y derecha. En general, este área será rectangular, aunque en ocasiones, como puede ser este ejemplo, desaparece la esquina superior izquierda cuando la disparidad entre ambos objetos es muy pequeña (objeto muy alejado del sistema de cámaras). De esta forma, consideraremos que dos objetos son la imagen del mismo objeto real si el número de coeficientes totales que quedan dentro de I es mayor de la mitad. En el caso del ejemplo, serían seis los coeficientes que quedan dentro, y tres fuera, por lo que el sistema determina que ambos objetos pertenecen al mismo objeto real.

Cuando el objeto sólo ha sido detectado en una de las imágenes, la estimación se complica, ya que el chequeo de la consistencia izquierda - derecha no va a poder realizarse. En este caso, lo único que podemos hacer es calcular la función de similitud para el objeto detectado, y obtener el pico de dicha función. Si dicho pico supera un determinado umbral, indica que esa posición, en la imagen donde no fue detectado el objeto, tiene una gran probabilidad de ser la imagen del objeto detectado en la otra imagen. Si es así, podemos estimar la posición del objeto detectado en una de las imágenes proyectándolo en la otra, como se muestra en la figura 3.17. Esta información, que evidentemente no puede tener la misma fiabilidad que en el caso de que el objeto haya sido detectado en ambas imágenes, puede ser utilizado, por ejemplo, como realimentación para el módulo de detección.

Si el máximo de la función de similitud no supera un determinado umbral, probablemente indique una falsa correspondencia. Esta circunstancia podrá entenderse, generalmente, como una oclusión. Si es así, niveles superiores del sistema, que no serán analizados en esta tesis, podrán utilizar esta información para obtener información más precisa sobre el funcionamiento del fondo, en concreto, sobre donde es posible que se produzcan futuras oclusiones.

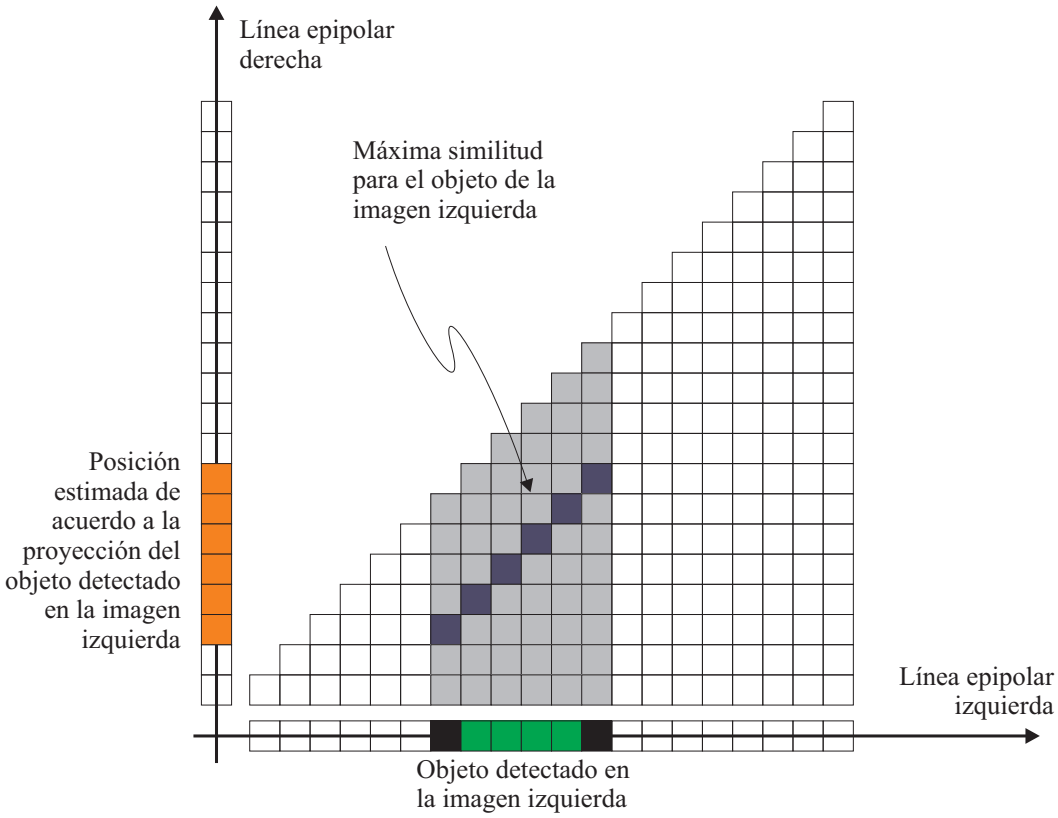


Figura 3.17: Tabla de correspondencias el caso de que el objeto sólo se detecte en una de las imágenes. Si la estimación es válida, se proyecta el objeto sobre la otra imagen.

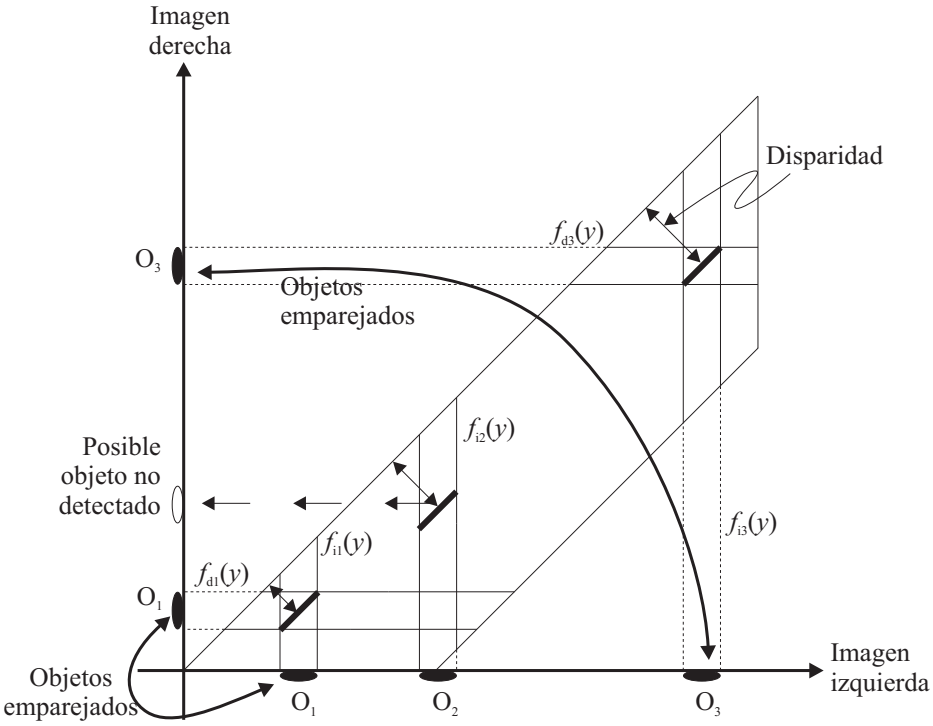
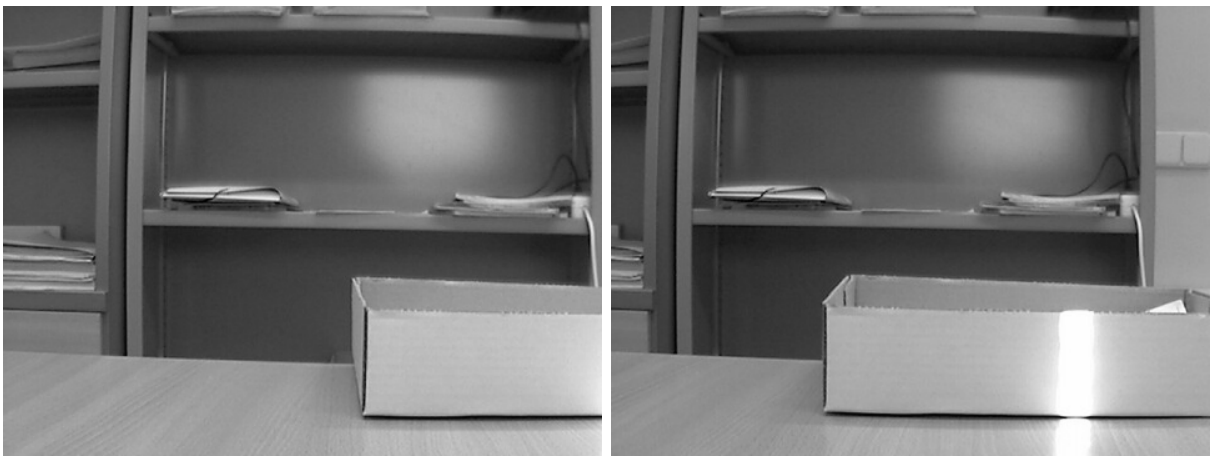


Figura 3.18: Visión global del funcionamiento del sistema en presencia de más de un objeto en cada imagen.

En la figura 3.18 podemos comprobar el funcionamiento global del sistema, cuando en cada imagen han sido detectados más de un objeto. En este caso concreto, en la imagen izquierda han aparecido tres objetos (O_1, O_2 y O_3), mientras que en la derecha sólo dos (O_1 y O_3). Esto podría haber sido debido, por ejemplo, a una oclusión, o bien a un error en el proceso de detección en la imagen derecha. Si el algoritmo funciona correctamente, sería de esperar que el sistema empareje los objetos tal y como se muestra en la figura, es decir, O_1 y O_3 entre sí, mientras que, dependiendo del resultado de la función de similitud, para O_2 podría suponer la existencia de una oclusión, o bien, un error en la detección del objeto en la imagen derecha.

Por último, y como optimización computacional del sistema, podremos comprobar ahora cual es la dependencia entre la información de las distintas líneas epipolares. Hasta ahora únicamente hemos analizado el sistema a nivel de línea epipolar. Sin embargo, no hay que olvidar que la ventana de búsqueda para un objeto determinado es compartido por varias líneas epipolares, en concreto, tantas como la altura del objeto. Por ejemplo, en la figura 3.15 podemos ver que la función de similitud calculada con la ventana representada en color naranja en la imagen derecha es la misma para las líneas 4 a 10 (comenzando por arriba con la línea 0).

De esta forma, la función de similitud calculada una única vez puede propagarse a lo largo de las sucesivas líneas que contienen al objeto. De hecho, computacionalmente ha-



(a) Imagen izquierda (b) Imagen derecha

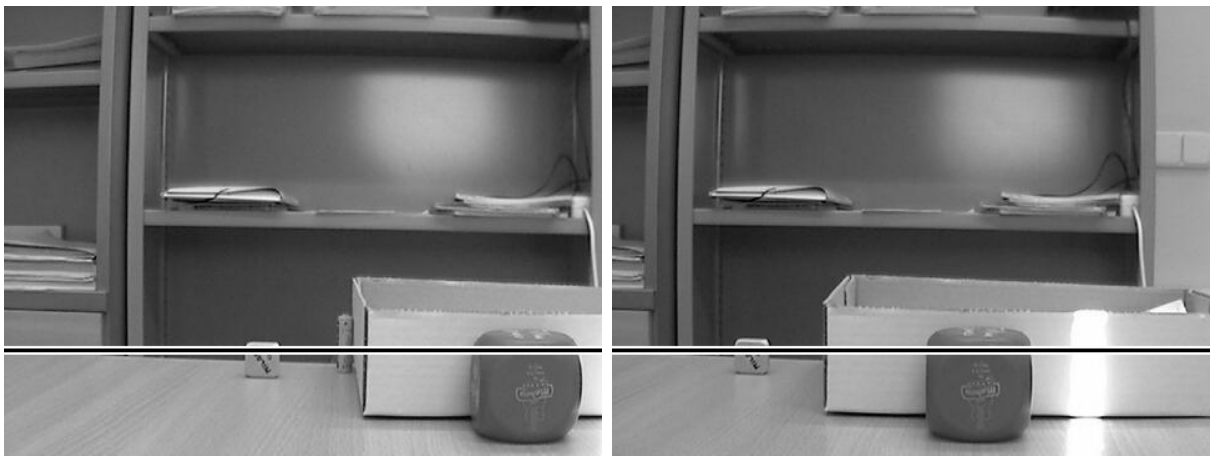
Figura 3.19: Imágenes izquierda y derecha del fondo estimado.

blando, resulta más efectivo calcular la función de similitud para cada objeto al comienzo de la ejecución del algoritmo, y luego ir combinando los distintos resultados en función de los píxeles que ocupan cada una de las máscaras de los objetos. Además, no es necesario evaluar la disparidad para cada nueva línea epipolar, ya que mientras la situación no cambie al evaluar la siguiente línea epipolar, es decir, sigan existiendo los mismos objetos, el resultado será el mismo.

Ejemplo práctico

Para demostrar el funcionamiento del algoritmo propuesto, mostraremos en este apartado los resultados prácticos obtenidos mediante el procesado, de acuerdo al algoritmo anterior, de imágenes reales. Para ello, se han grabado, mediante dos cámaras similares separadas una de la otra una distancia determinada, la misma secuencia. En concreto, se ha grabado una secuencia con un fondo estacionario, en la que en un momento determinado aparecen distintos objetos del primer plano. En la figura 3.19 se muestra el fondo estimado a partir de las primeras imágenes de la secuencia. Es interesante observar que las imágenes están perfectamente calibradas, lo cual quiere decir básicamente que las líneas epipolares de ambas imágenes coinciden con las filas de las imágenes. Aunque éste no hubiese sido el caso, siempre podríamos haber rectificado las imágenes para obtener dos nuevas imágenes calibradas (ver figura 3.5).

En la figura 3.20 se muestran dos imágenes de la misma secuencia en la que aparecen varios objetos de primer plano. Si aplicamos sobre estas imágenes un algoritmo de detección de movimiento para extraer los objetos de primer plano, el resultado son las máscaras mostradas en la figura 3.21. Con el fin de eliminar el ruido de segmentación y mejorar el aspecto del contorno de los objetos, dichas máscaras han sido sometidas a procesos de erosión y dilatación. Además, se ha realizado un filtrado por tamaño para



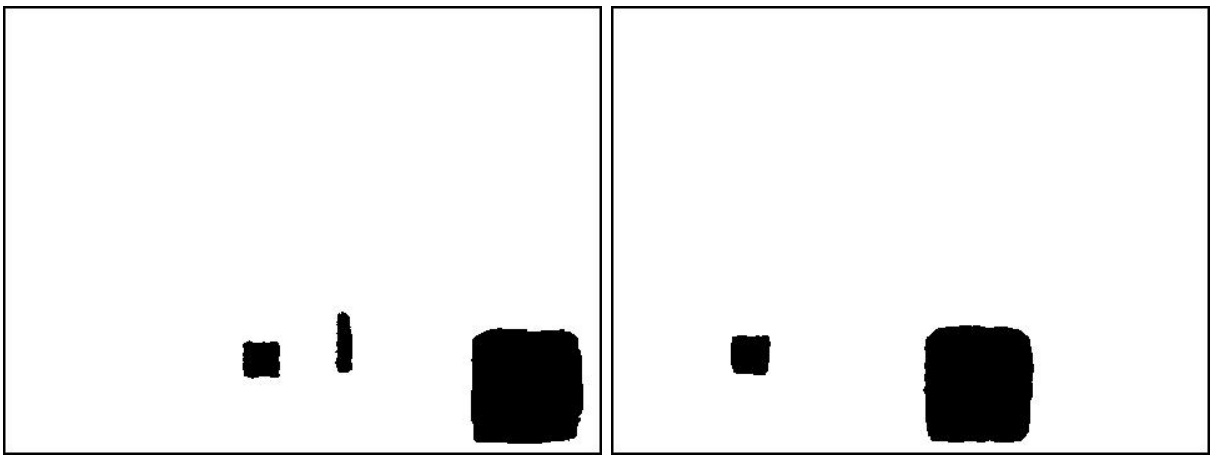
(a) Imagen izquierda (b) Imagen derecha

Figura 3.20: Imágenes izquierda y derecha con objetos de primer plano. Obsérvese que el objeto del centro sólo aparece en la imagen izquierda.

eliminar falsas alarmas, que generalmente suelen presentarse como conjuntos de píxeles aislados, o blobs de pequeño tamaño. Por último, las máscaras resultantes se someten a un proceso de rellenado de sus huecos internos, para así conseguir que el contorno del objeto solamente corresponda a los bordes de éste, y no a zonas interiores del objeto, que debido a errores de segmentación puedan haber quedado sin marcar.

Para comprobar el funcionamiento del algoritmo de cálculo de disparidades de objetos entre ambas imágenes, vamos a analizar el resultado sobre la línea $x = 275$, que puede verse marcada en negro en la figura 3.20. En esta línea epipolar, los objetos detectados serían los indicados en la tabla 3.1. En esta tabla se indica la posición, en píxeles, de los bordes izquierdo y derecho de cada objeto en cada una de las imágenes. Además, y con el objeto de poder comparar con los resultados prácticos mostrados más adelante, se ha calculado también la disparidad del objeto, como la media entre la disparidad para el borde derecho y el borde izquierdo, ya que el tamaño del objeto en cada una de las imágenes puede no ser idéntica, como ocurre por ejemplo con el objeto 1. Es importante observar en esta tabla, al igual que en la figura 3.20, que uno de los objetos sólo es visible en la imagen de la izquierda, es decir, que ha ocurrido una oclusión en la imagen derecha.

Por tanto, el algoritmo desarrollado en esta sección calculará las funciones de similitud, en este caso, la correlación cruzada normalizada respecto a la media, para los tres objetos detectados en la imagen izquierda, y los dos detectados en la derecha. El caso aquí analizado es el mismo que el visualizado en la figura 3.18, por lo que podemos observar de nuevo esta imagen para comprender mejor qué funciones han sido calculadas. En concreto, las funciones calculadas serían las marcadas en la figura como $f_{\mathrm{d}1}(y)$ y $f_{\mathrm{d}3}(y)$ para los objetos de la imagen derecha, y $f_{\mathrm{i}1}(y)$, $f_{\mathrm{i}2}(y)$ y $f_{\mathrm{i}3}(y)$ para los objetos de la izquierda.



(a) Imagen izquierda (b) Imagen derecha

Figura 3.21: Máscaras de movimiento para las imágenes de la figura 3.20.

Tabla 3.1: Posición máxima y mínima, en píxeles, de los objetos detectados en la imagen izquierda y derecha, en la línea epipolar $x = 275$, y disparidad del objeto.

Objeto	Izquierda	Derecha	Disparidad
1	192 - 222	95 - 127	96
2	267 - 280	Oclusión	-
3	375 - 462	250 - 337	125

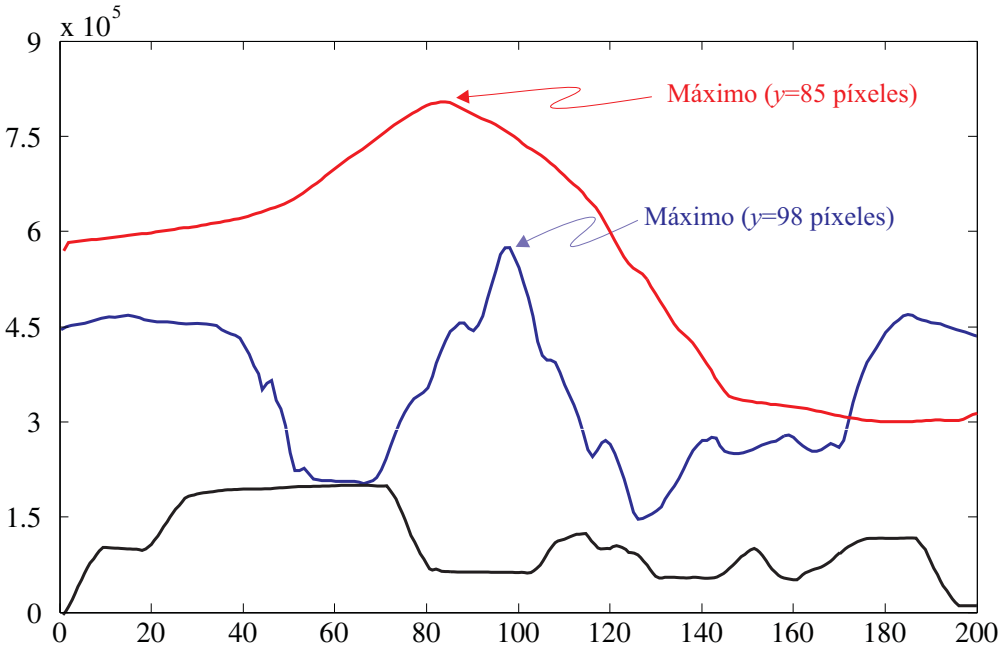


Figura 3.22: Ejemplos de correlación cruzada normalizada para los objetos mostrados en la escena de la imagen 3.20. (Rojo): Correlación cruzada normalizada ($ZNCC$) para el objeto 1 en la imagen derecha. (Azul): Correlación cruzada normalizada ($ZNCC$) para el objeto 1 en la imagen izquierda. (Negro): Correlación cruzada normalizada ($ZNCC$) para el objeto 2 en la imagen izquierda (objeto ocluido en la imagen derecha).

La imagen 3.22 muestra una representación gráfica de tres de las funciones anteriores. En concreto, en rojo y en azul se muestra el cálculo de la correlación cruzada normalizada para el objeto 1 detectado en la imagen izquierda sobre la imagen derecha, y viceversa respectivamente. Para evitar un cálculo innecesario de muestras de la correlación, se ha limitado los posibles valores de disparidad desde 0 hasta 200 píxeles. En la figura se puede comprobar fácilmente cómo el valor del máximo para la función de similitud ocurre a 98 y 85 píxeles, lo cual puede entenderse como una estimación buena respecto a su valor real de 96 píxeles (ver de nuevo la tabla 3.1).

En la figura 3.22 se muestra también el cálculo de la correlación para el caso del objeto ocluido en la imagen derecha. En este caso, podemos comprobar como el máximo para la función de correlación alcanza valores muy por debajo de los valores que alcanza en condiciones de no oclusión. Este hecho puede alertar al sistema de que, probablemente, se trate de una oclusión.

Como sería de esperar, el algoritmo descrito también puede trabajar con escenas de exteriores, que es quizás el lugar donde más utilidad tienen el sistema aquí descrito. Para ello, se ha grabado una secuencia de vídeo con dos cámaras. En la figura 3.23 se muestra el fondo estimado de la escena para cada una de las cámaras. También podemos observar aquí

(a) Imagen izquierda (b) Imagen derecha

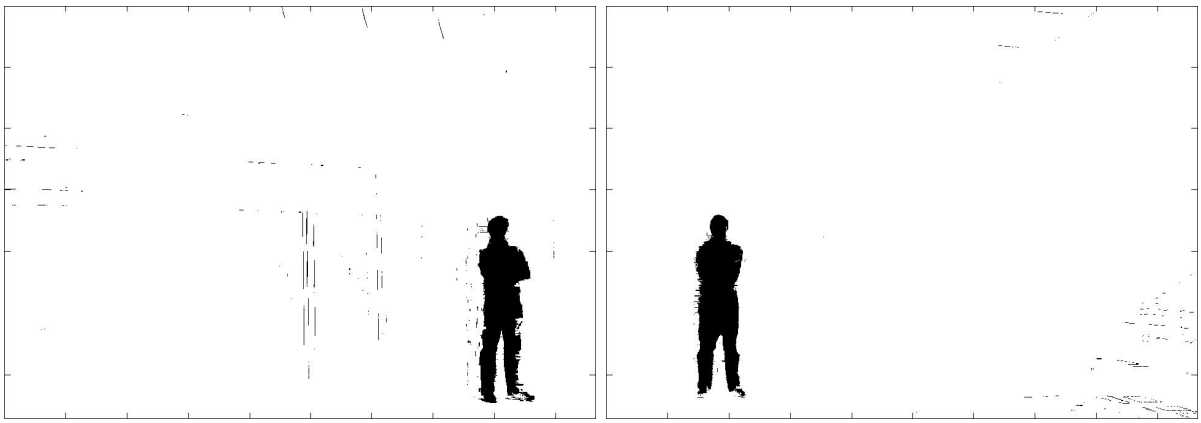
Figura 3.23: Imágenes izquierda y derecha del fondo estimado.

(a) Imagen izquierda (b) Imagen derecha

Figura 3.24: Imágenes izquierda y derecha con objetos de primer plano.

que, con el fin de reducir la complejidad computacional, las cámaras están perfectamente calibradas, es decir las líneas epipolares coinciden con las filas de las imágenes.

En la figura 3.24 se muestra un instante concreto de la escena anterior, donde aparece una persona sobre el fondo. Tras realizar la detección de movimiento sobre ambas imágenes, las máscaras resultantes se muestran en la figura 3.25, donde, a pesar de aparecer bastantes objetos, un simple filtrado por tamaño deja únicamente un objeto en cada imagen, el correspondiente a la persona que realmente queríamos detectar. En este caso, al ser la forma del objeto más compleja que las obtenidas en el ejemplo anterior, la obtención de la disparidad real es bastante más complicado. Para obtener una aproximación bastante fiable de dicho valor, se ha procedido a calcular manualmente la disparidad en distintos puntos del objeto, especialmente en los bordes de éste, y obtener la disparidad como la media de todos esos valores. Dicho valor resultó ser igual a 627 píxeles.



(a) Imagen izquierda (b) Imagen derecha

Figura 3.25: Máscaras de movimiento para las imágenes de la figura 3.24.

Como solamente existe un objeto en cada imagen, el cálculo de la función de correlación cruzada sólo deberá ejecutarse una vez por cada imagen, una para realizar la búsqueda del objeto en la imagen derecha sobre la imagen izquierda, y otra para el objeto en la imagen izquierda sobre la imagen derecha. La figura 3.26 muestra la función de correlación cruzada para ambos casos, donde podemos comprobar como, para ambos casos, el máximo de la función de correlación cruzada se encuentra a unos 615 píxeles, que es un valor muy próximo a la disparidad de 627 píxeles obtenida de forma manual.

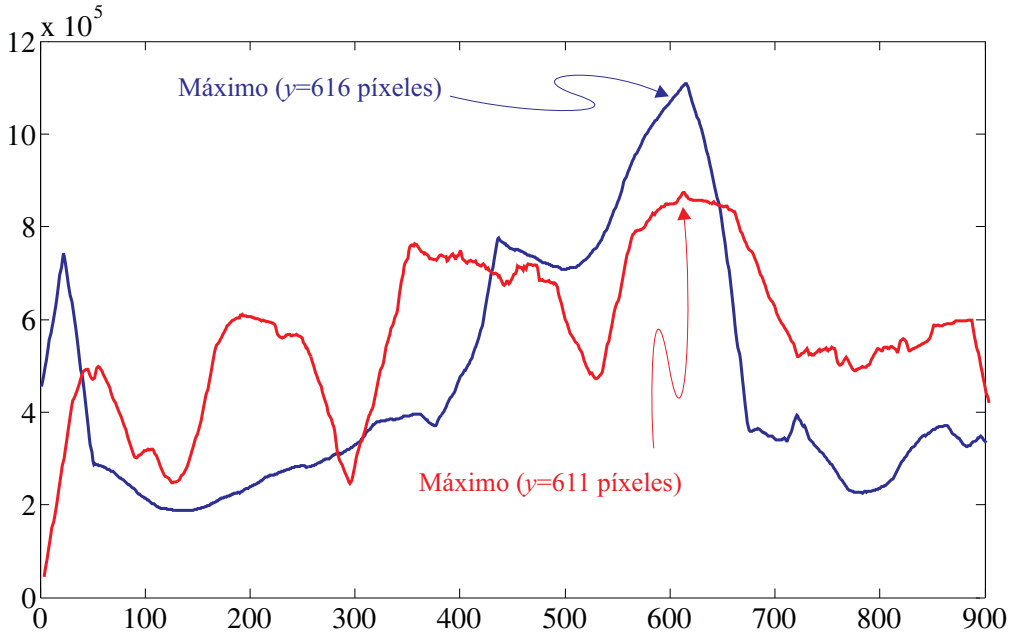


Figura 3.26: Cálculo de la correlación cruzada normalizada para los objetos mostrados en la escena de la imagen 3.24. (Rojo): Correlación cruzada normalizada ($ZNCC$) para el objeto en la imagen derecha. (Azul): Correlación cruzada normalizada ($ZNCC$) para el objeto en la imagen izquierda.

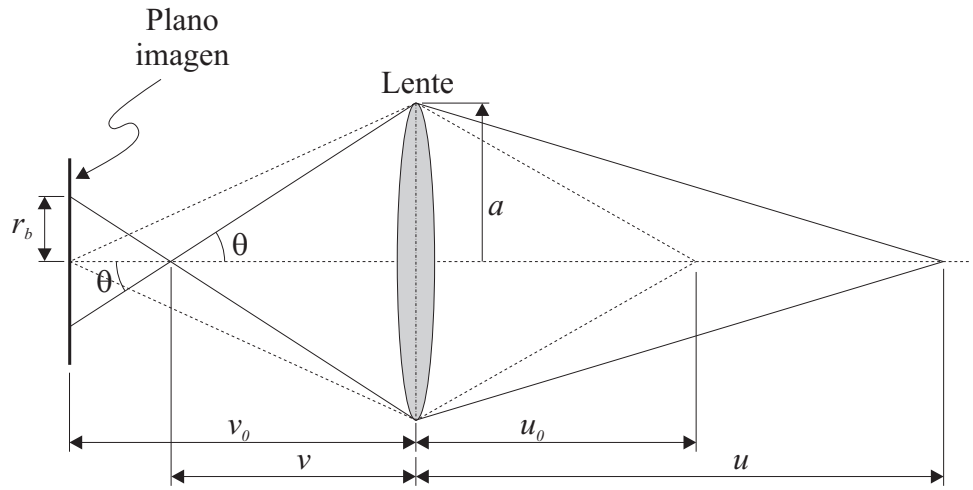


Figura 3.27: Geometría del sistema de captura de imágenes. Desenfoque de la imagen

3.2. Profundidad por desenfoque

De todos los mecanismos que el sistema visual humano emplea para realizar la reconstrucción en tres dimensiones de una escena, el desenfoque, junto con la visión estéreo, son los dos únicos métodos que no necesitan de un conocimiento previo de la escena, y por lo tanto, idóneo para poder realizar la estimación de distancias en un sistema de videovigilancia. Las técnicas de reconstrucción en tres dimensiones por desenfoque se basan en el hecho de que el desenfoque producido por una lente real depende de la distancia a la que se encuentra el objeto. En la figura 3.27 se muestra la geometría de un sistema de captura de imágenes que emplea una lente real de radio a. Un sistema óptico como el representado sólo es capaz de enfocar los puntos situados en un plano a una distancia determinada. Supongamos que la distancia focal (distancia entre el eje de la lente y el plano imagen, por ejemplo el CCD de una videocámara) es v_0, y que la longitud focal de la lente es f. En estas condiciones, sólo los objetos situados a la distancia u_0 del eje de la lente estarían enfocados perfectamente. Para una lente delgada, la relación entre las tres magnitudes anteriores viene dada por la *ley de la lente delgada*:

$$\frac{1}{u_0} + \frac{1}{v_0} = \frac{1}{f} \tag{3.7}$$

Un objeto situado a cualquier otra distancia será visualizado en la imagen de forma borrosa, y ese emborronamiento será tanto mayor cuanto más alejado se encuentre el objeto del punto de enfoque perfecto u_0. Como puede verse en la figura 3.27, ese emborranamiento se produce por el hecho de que un punto situado a una distancia u distinta de u_0 no es visualizado en un único punto del plano imagen, sino que se extiende a lo largo de un círculo de radio r_b.

Antes de continuar, es necesario observar que dos objetos situados a dos distancias, u_{f} más lejana (*far*) de la lente que el punto de enfoque perfecto u_0, y u_{n} más cercana (*near*) van a provocar el mismo radio r_b en la imagen desenfocada, como se puede ver en la figura 3.28. En general, este hecho debe tenerse en cuenta, y muchos algoritmos de cálculo de profundidad por desenfoque necesitan que los objetos de interés estén todos situados o bien más lejos que el punto de enfoque perfecto, o bien todos más cerca del punto de enfoque perfecto. El cálculo de dicho radio r_b sería:

$$\tan\theta = \frac{a}{v} = \frac{r_b}{|v_0 - v|} \tag{3.8}$$

donde el cálculo del valor absoluto de la diferencia entre v_0 y v permite contemplar ambos casos, es decir, el hecho de que el punto se encuentre más cerca o más lejos del punto de enfoque perfecto. Por otro lado, para un punto situado a una distancia de la lente u arbitraria tendríamos, a partir de la ley de la lente delgada:

$$\frac{1}{u} + \frac{1}{v} = \frac{1}{f} \tag{3.9}$$

Combinando las dos expresiones anteriores, llegamos a:

$$u = \frac{f v_0 a}{v_0 a - f\,(a \pm r_b)} \tag{3.10}$$

o más comúnmente:

$$u = \frac{f v_0}{v_0 - f - r_b F} \tag{3.11}$$

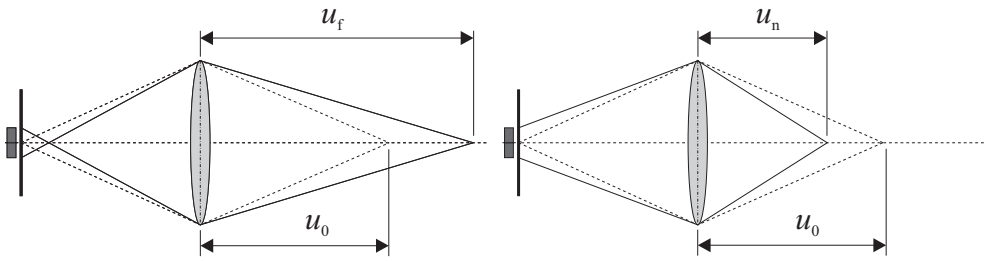
para objetos situados más allá del punto de enfoque perfecto (figura 3.28(a)) y:

$$u = \frac{f v_0}{v_0 - f + r_b F} \tag{3.12}$$

para objetos situados más cerca que el punto de enfoque perfecto (figura 3.28(b)). En las expresiones anteriores, $F = f/a$ es el denominado *número F* de la lente, que depende exclusivamente de las características geométricas y ópticas de la lente empleada.

En las expresiones anteriores se puede comprobar que la cantidad de emborronamiento para un punto desenfocado, que es proporcional al radio r_b, depende del tamaño de la apertura a. Es decir, cuanto más pequeña sea la lente de la cámara menor será el emborronamiento para los puntos que no se encuentren perfectamente enfocados. En el límite, cuando el tamaño de la apertura fuese infinitesimal, todos los puntos estarían perfectamente enfocados, independientemente de la distancia a la que se encuentren de la cámara. Este tipo de cámaras, donde toda la escena se encuentra perfectamente enfocada, se denominan cámaras *pinhole*. Una imagen tomada con este tipo de cámara, donde todos los puntos están perfectamente enfocados, suele denominarse como la *imagen enfocada*.

El efecto que el desenfoque produce sobre la imagen enfocada puede verse como una convolución bidimensional de dicha imagen con una función de dispersión PSF (*Point*



(a) Objeto lejano (b) Objeto cercano

Figura 3.28: Distancias que producen el mismo desenfoque.

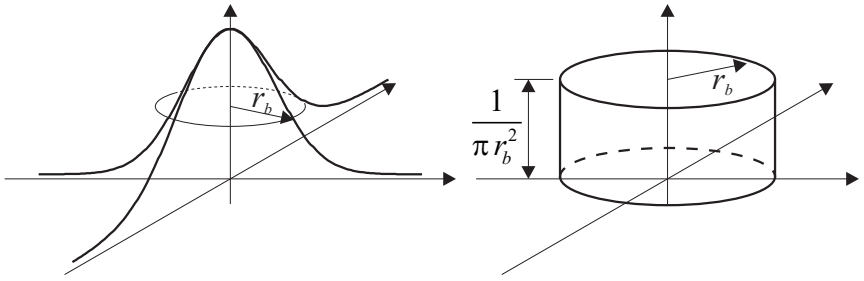
Spread Function). Aunque la forma de esta función depende de la lente empleada, y debe ser analizada desde el punto de vista de las leyes ópticas que rigen su funcionamiento, distintos autores la han aproximado por funciones más simples que pueden ser analizadas matemáticamente de forma más sencilla. A la hora de elegir la función de dispersión, hay que tener en cuenta que la luz que atraviesa la lente e incide sobre el plano imagen es la misma, independientemente de cual sea el punto de enfoque perfecto. Quiere esto decir que la energía total de la imagen no debería verse afectada por un cambio en la posición del enfoque, o dicho de otra forma:

$$\int_{x,y=-\infty}^{\infty} \mathrm{PSF}\,(x,y)\,\mathrm{d}x\,\mathrm{d}y = \int_{r=0}^{\infty}\int_{\theta=-\pi}^{\pi} \mathrm{PSF}\,(r,\theta)\,\mathrm{d}r\,\mathrm{d}\theta = 1 \tag{3.13}$$

En general, en la descripción de las funciones de dispersión es preferible el uso de coordenadas polares debido a la simetría circular del sistema óptico. Entre las funciones de dispersión más utilizadas podemos destacar la función gaussiana bidimensional, como se puede ver en la figura 3.29, empleada entre otros por [Pentland87], que fue uno de los primeros autores en trabajar en el campo de la estimación de profundidad por desenfoque. En este caso se trata de una función gaussiana bidimensional, de media 0 y desviación típica σ igual al radio r_b de la figura 3.27. La función gaussiana fue la preferida inicialmente, ya que su transformada de Fourier en coordenadas polares sigue siendo una función gaussiana.

La segunda función, empleada por ejemplo por [Watanabe98], es la denominada *Pill-box*. Se trata de un cilindro cuyo radio es igual al parámetro r_b en la figura 3.27. Otras funciones también han sido descritas y utilizadas por distintos autores, aunque estas dos han sido en general las más empleadas.

Es importante ver que, independientemente de la función de dispersión empleada, la escala de la función de dispersión varía en función de la posición del punto observado respecto del punto de enfoque perfecto. En este sentido, cuanto más cerca esté el punto observado al punto de enfoque perfecto, más se parecerá la función de dispersión a una función delta bidimensional, tal y como se muestra en la figura 3.30. En el límite, cuando



(a) Función gaussiana bidimensional (b) Función *Pillbox*

Figura 3.29: Funciones de dispersión *Point Spread Function* más comunes.

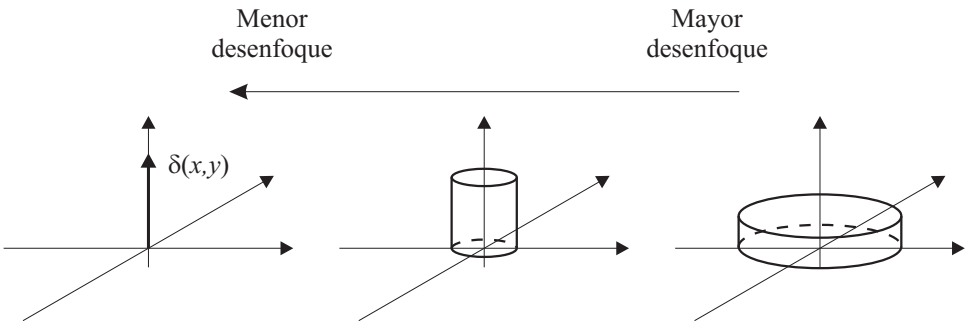


Figura 3.30: Función de dispersión *Pillbox* para distintas distancias del punto observado.

el enfoque es perfecto, la función de dispersión es una delta, y no se produce ningún emborronamiento de la imagen:

$$I(x,y) = I_{\mathrm{f}}(x,y) * \delta(x,y) = I_{\mathrm{f}}(x,y) \tag{3.14}$$

El principal problema que se plantea es que la función de dispersión varía en función de la distancia a la que se encuentra cada punto de la escena de la lente. Por tanto, el emborronamiento que se produce en la imagen no puede verse como una convolución, ya que la respuesta al impulso depende de la distancia, que es justamente lo que queremos calcular. Aplicando la restricción de continuidad enunciada anteriormente, el desenfoque deberá evaluarse, por tanto, de forma local.

Básicamente, existen dos técnicas para calcular profundidades a partir del desenfoque producido por la lente, que son:

- **Profundidad por enfoque (*Depth Form Focus* DFF):** En este caso un sistema de enfoque variable toma un número de imágenes elevado (algunos autores proponen el uso de unas 30 o más imágenes), cada una de ellas con distinto enfoque. Para cada punto de la imagen se evalúa la posición del enfoque con la que se obtiene una imagen más nítida. La profundidad se obtiene a partir de las expresiones derivadas de la figura 3.27.

- **Profundidad por desenfoque (*Depth Form Defocus* DFD):** En este caso, sólo dos imágenes, una enfocando a un punto cercano (imagen cercana o *Near-focused image*), y otra enfocando a un punto lejano (imagen lejana o *Far-focused image*) son suficientes para obtener la profundidad de cada punto de la imagen. La ventaja principal de este método es el menor número de imágenes a tomar, mientras que el principal inconveniente es que la complejidad computacional aumenta.

En ambos casos, hay que tener en cuenta que todos los objetos de la escena deben estar comprendidos entre la distancia del enfoque más cercano y el enfoque más lejano.

3.2.1. Técnicas de cálculo de profundidad por enfoque

Las técnicas basadas en DFF necesitan encontrar, para cada punto de la imagen, la posición de la lente que hace que dicho punto se vea con mayor nitidez. De entre los autores que proponen el uso de las técnicas DFF se encuentra [Jarvis83]. En este caso, se realiza la medida local de las siguientes funciones:

- Entropía:

$$E = -\sum_{x} P\left(x\right) \ln P\left(x\right)$$

- Varianza:

$$V = \frac{1}{N} \sum_{i=1}^{N} \left(x_i - \bar{x}\right)^2$$

- Suma del módulo de la diferencia:

$$\mathrm{SMD} = \sum_{i=2}^{N} \left|x_i - x_{i-1}\right|$$

Para cada punto de la imagen, sólo es necesario buscar la posición de la lente que maximiza las funciones anteriores. Una vez conocida dicha posición, el cálculo de la profundidad es inmediato empleando las expresiones anteriores. La toma de las imágenes suele realizarse con un sistema de lentes motorizado, de tal forma que la cámara toma una imagen para cada una de las distintas posiciones del enfoque. El número de imágenes tomadas es variable, siendo necesario un mayor número de imágenes cuanto mayor sea la precisión requerida. Hay que tener en cuenta que el enfoque debe barrer todo el rango de profundidades que se desea cubrir.

Si bien la complejidad algorítmica de los esquemas DFF es pequeña, el inconveniente principal es el complejo sistema de captación de imágenes requerido, ya que es necesario un sistema óptico con enfoque motorizado suficientemente preciso. Además, como el tiempo

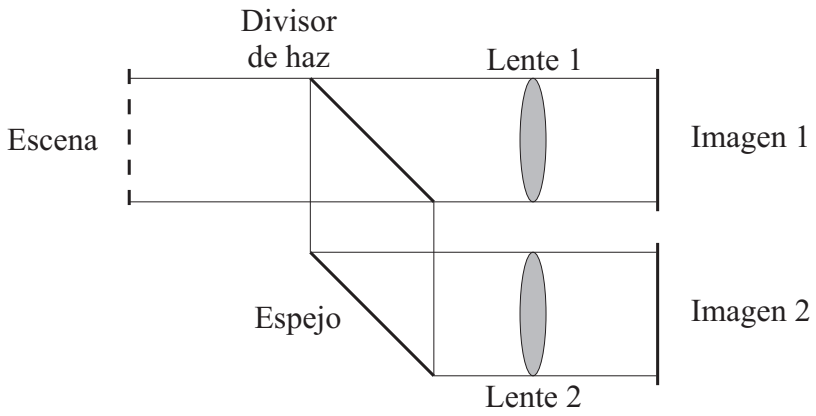


Figura 3.31: Sistema de captura de imágenes con dos enfoques distintos.

que necesita el sistema óptico para barrer todo el rango de profundidades es elevado, aun siendo la complejidad computacional pequeña, el tiempo necesario para evaluar la profundidad de una escena será elevado. Además, durante el tiempo de grabación de las distintas imágenes de la escena con enfoques distintos, la escena deberá permanecer inmóvil.

3.2.2. Técnicas de cálculo de profundidad por desenfoque

Aunque computacionalmente más complejos, los algoritmos basados en DFD presentan la ventaja de que el sistema de captación es más sencillo, ya que sólo se requieren dos imágenes, aunque se pueden incluir más para aumentar la fiabilidad del algoritmo. La captura de las imágenes puede realizarse de dos formas. Bien empleando un sistema óptico motorizado que permita variar el enfoque para poder tomar la imagen cercana y la imagen lejana, o bien utilizar un sistema como el de la figura 3.31, propuesto por ejemplo por [Pentland87]. Este sistema emplea un divisor de haz, es decir, un dispositivo que divide la imagen original en dos imágenes idénticas, que posteriormente son registradas por dos cámaras independientes, cada una de ellas con un enfoque determinado. De esta forma, el tiempo que se tardaría en registrar el conjunto de dos imágenes es el mismo que se tardaría en tomar una única imagen. Otra ventaja de este sistema es que no es necesario que la escena sea estática, ya que ambas imágenes se graban simultáneamente. Independientemente del modo de captación de las imágenes desenfocadas, los algoritmos más conocidos en el campo del cálculo de profundidad por desenfoque son los que se describen a continuación:

DFD basado en gradientes focales

Este algoritmo, propuesto en [Pentland87] en 1987, es uno de los primeros sistemas de cálculo de profundidades por desenfoque propuestos por la comunidad científica. En este

algoritmo, una vez realizada la toma de dos imágenes de la misma escena con dos enfoques distintos, el siguiente paso es el cálculo de la cantidad de emborronamiento de cada píxel de la imagen. Si somos capaces de relacionar dicho emborronamiento con el radio r_b de la función de dispersión vista anteriormente, el cálculo de la profundidad sería inmediato a partir de (3.12) ó (3.11). El problema principal radica, por tanto, en el cálculo del radio de la función de dispersión para cada píxel.

Asumiendo que la función de dispersión es una función gaussiana, su desviación típica será igual al radio r_b ilustrado en la figura 3.27. Como se obtienen dos imágenes con enfoques distintos, dicho radio, y por tanto la varianza, serán en general distintas entre sí. Si suponemos que la imagen enfocada es $I_{\mathrm{f}}(x, y)$, y tomamos una región circular centrada en el punto (x_0, y_0) dentro de esta imagen:

$$f_{\mathrm{f}}(r, \theta) = I_{\mathrm{f}}(x_0 + r\cos\theta, y_0 + r\sin\theta) \tag{3.15}$$

las imágenes desenfocadas pueden obtenerse como la convolución de dicha región circular con la función de dispersión gaussiana. La relación entre ambas imágenes sería:

$$\frac{f_1(r, \theta)}{f_2(r, \theta)} = \frac{f_{\mathrm{f}}(r, \theta) * g(r, \sigma_1)}{f_{\mathrm{f}}(r, \theta) * g(r, \sigma_2)} \tag{3.16}$$

donde $g(r, \sigma)$ es la función de dispersión gaussiana de media 0 y varianza σ, ó r_b:

$$g(r, \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{r^2}{2\sigma^2}\right) \tag{3.17}$$

Teniendo en cuenta que para la función de dispersion anterior, la transformada de Fourier-Bessel, o alternativamente transformada de Hankel de orden cero, viene dada por [Goodman96]:

$$G(\rho, \sigma) = \exp\left(-2(\pi\sigma\rho)^2\right) \tag{3.18}$$

la relación anterior se convierte en:

$$\frac{F_1(\rho)}{F_2(\rho)} = \frac{F_{\mathrm{f}}(\rho) \cdot G(\rho, \sigma_1)}{F_{\mathrm{f}}(\rho) \cdot G(\rho, \sigma_2)} = \frac{G(\rho, \sigma_1)}{G(\rho, \sigma_2)} = \exp\left(\rho^2 2\pi^2\left(\sigma_2^2 - \sigma_1^2\right)\right) \tag{3.19}$$

Calculando el logaritmo de las expresiones anteriores, obtendremos que:

$$\rho^2 2\pi^2\left(\sigma_2^2 - \sigma_1^2\right) = \ln F_1(\rho) - \ln F_2(\rho) \tag{3.20}$$

que es una regresión lineal en ρ^2, con la cual podremos obtener el valor de $\sigma_2^2 - \sigma_1^2$. Haciendo $\sigma_1 = 0$, es decir, si la primera imagen es una imagen tomada con una cámara *pinhole*, la función de dispersión ya estaría calculada para cada punto de la escena, y por tanto, mediante (3.12) ó (3.11) podríamos obtener la profundidad de cada punto.

Una variante de este algoritmo es el propuesto en [Wei05] para realizar la reconstrucción en tres dimensiones de imágenes tomadas con microscopio. Para ello se obtienen

distintas señales a partir de la imagen desenfocada utilizando operadores de gradiente de primer y segundo orden. En el mencionado trabajo, se emplean los operadores de Rober, Prewitt, Sobel y el Laplaciano [Gonzalez93]. El cálculo del desenfoque se obtiene exclusivamente en los bordes de los objetos colocados bajo el microscopio. Una aproximación más simple puede ser la presentada en [Jong07]. En este caso, y considerando que la función de dispersión PSF es una función gaussiana bidimensional, los autores proponen el uso de una red neuronal con funciones de base radial (RBF) para la estimación de la desviación típica σ de la función de dispersión. La precisión de los algoritmos de cálculo de profundidad por desenfoque puede mejorarse empleando técnicas multirresolución, como por ejemplo en [Asif05] o en [Qiang01]. En ambos casos, se emplea la transformada *Wavelet* para mejorar la precisión en los resultados obtenidos.

DFD mediante filtros racionales

En este algoritmo, propuesto en [Watanabe98], la función de dispersión es sin embargo la función *pillbox*, que consiste en un cilindro de radio el parámetro r_b calculado de acuerdo a la figura 3.27, y de altura:

$$h = \frac{1}{\pi r_b^2} \tag{3.21}$$

para conseguir que el volumen de la función de dispersión sea igual a 1. De esta forma, la función de dispersión *pillbox* se identificará con la siguiente expresión:

$$h\left(r,\theta\right) = h\left(r\right) = \frac{1}{\pi r_b^2} \prod\left(\frac{r}{r_b}\right) \tag{3.22}$$

donde la coordenada θ desaparece, ya que la función tiene simetría circular. La función Π se corresponde con la siguiente expresión:

$$\prod\left(u\right) = \begin{cases} 1 & u < 1 \\ 0 & u \geq 1 \end{cases} \tag{3.23}$$

es decir, define un círculo de radio r_b. Aprovechando la simetría circular de la función anterior, la transformada de Fourier-Bessel en coordenadas polares correspondería con la expresión [Goodman96]:

$$\mathrm{H}\left(\rho,\phi\right) = \mathrm{H}\left(\rho\right) = \frac{1}{\pi r_b \rho} \mathrm{J}_1\left(2\pi\rho r_b\right) \tag{3.24}$$

donde ρ y ϕ son las coordenadas frecuenciales radial y angular respectivamente empleando un sistema de coordenadas polares, y J_1 la función de Bessel de orden uno de primera especie. Además, como la función de dispersión tiene simetría circular, también tendrá la misma simetría su transformada de Fourier, por lo que la coordenada ϕ desaparece.

En la figura 3.32 se muestra la respuesta en frecuencia de dos funciones de dispersión *Pillbox* con dos radios r_b distintos. Para simplificar la visualización, y teniendo en cuenta

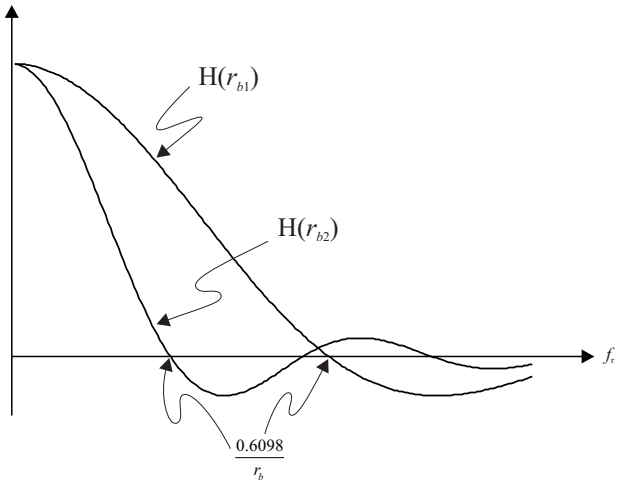


Figura 3.32: Respuesta en frecuencia de la función de dispersión *Pillbox*.

que la señal bidimensional tiene simetría circular, sólo se ha representado la respuesta en frecuencia en función de la componente radial ρ. Hay que tener en cuenta que, al igual que ocurre con señales de una dimensión, cuanto más ancha sea la señal en el tiempo, más estrecha es su respuesta en frecuencia (la señal tiene más componentes de baja que de alta frecuencia) y al revés. De hecho, considerando que el primer paso por cero de la función de Bessel de primera especie de orden 1 se produce aproximadamente en 3,8317, el primer paso por cero de la respuesta en frecuencia se producirá en:

$$\begin{aligned} 2\pi\rho r_b &= 3{,}8317 \\ \rho &= \frac{0{,}6098}{r_b} \end{aligned} \tag{3.25}$$

Por tanto, en la figura 3.32, r_{b_1} corresponde a un radio más pequeño que r_{b_2}. En el límite, para un punto que se encuentra perfectamente enfocado, para el cual el radio r_b sería 0, es decir, la función de dispersión correspondiese a una delta en el origen, la respuesta en frecuencia sería una línea horizontal.

La filosofía de funcionamiento de este algoritmo es, en líneas generales la siguiente. Como puede comprobarse de la figura 3.27, para un objeto situado a la distancia del punto de enfoque lejano, este objeto se visualizaría perfectamente enfocado en la imagen lejana, es decir, la función de dispersión tendría un radio de valor 0. Sin embargo, la función de dispersión para este objeto en la imagen cercana tendría un radio de valor elevado. A medida que el objeto se desplazara del punto de enfoque lejano hacia el punto de enfoque cercano el radio de la función de dispersión en la imagen lejana irá aumentando, mientras que en la imagen cercana tenderá a cero. Esto, traducido a la componente frecuencial de cada imagen tal y como se vio en la figura 3.32 resultaría en que, para la imagen lejana, el primer paso por cero se iría desplazando desde ∞ hasta $0{,}6098/r_{\mathrm{bm}}$, donde r_{bm} sería el

radio máximo, cuyo valor dependerá de los distintos parámetros de la geometría óptica del sistema en cuestión, mientras que para la imagen cercana, el proceso sería el inverso.

Teniendo en cuenta que las imágenes lejana y cercana, localmente pueden considerarse como la convolución en el espacio de la imagen perfectamente enfocada con la función de dispersión correspondiente, de igual forma, la componente espectral de cada una de las imágenes sería el producto de la componente espectral de la imagen original con la correspondiente función de Bessel. Definimos la relación M/P como:

$$\frac{M}{P} = \frac{I_{\mathrm{F}}(\rho,\theta) - I_{\mathrm{N}}(\rho,\theta)}{I_{\mathrm{F}}(\rho,\theta) + I_{\mathrm{N}}(\rho,\theta)} = \frac{H_{\mathrm{F}}(\rho) - H_{\mathrm{N}}(\rho)}{H_{\mathrm{F}}(\rho) + H_{\mathrm{N}}(\rho)} \tag{3.26}$$

ya que el espectro de la imagen original $I(\rho,\theta)$ se cancela. Si representamos la relación M/P anterior para distintos valores de la frecuencia radial f_{r} en función del radio r_b de la imagen lejana, obtendríamos las gráficas representadas en la figura 3.33. En esta gráfica se puede comprobar que, para un rango determinado de frecuencias radiales, la función M/P es monótona creciente, y por tanto, se puede establecer de forma unívoca una relación entre el valor de la función M/P y el radio r_b, y por tanto, de acuerdo con (3.11) ó (3.12), podríamos obtener la profundidad de dicho punto.

Por tanto, el único problema que restaría para poder establecer la profundidad de los objetos sería el calcular la relación M/P a una frecuencia radial determinada. El problema estriba en que la componente frecuencial depende de la escena, y por tanto, no podemos determinar a priori a qué frecuencia radial calcular la relación anterior. La estrategia seguida es utilizar una serie de bancos de filtros que puedan estimar la densidad espectral para un elevado número de frecuencias individuales.

DFD mediante la transformada S

En [Subbarao94], la profundidad se determina mediante la estimación de la función de dispersión que originó la imagen desenfocada. Para ello se desarrolla la Transformada S (*Spatial-Domain Convolution/Deconvolution Transform*). En este caso, la imagen original se modela como una función polinómica bidimensional cúbica, de acuerdo a:

$$i(x,y) = \sum_{m=0}^{3} \sum_{n=0}^{3-m} a_{mn} x^m y^n \tag{3.27}$$

donde a_{mn} son los coeficientes del polinomio. Al ser un polinomio cúbico, la función puede ser expresada mediante las series de Taylor:

$$i(x,y) = \sum_{0 \leqslant m+n \leqslant 3} \frac{(x-s)^m}{m!} \frac{(y-t)^n}{n!} i^{mn)}(s,t) \tag{3.28}$$

donde:

$$i^{mn)}(x,y) = \frac{\partial^m}{\partial x^m} \frac{\partial^n}{\partial y^n} i(x,y) \tag{3.29}$$

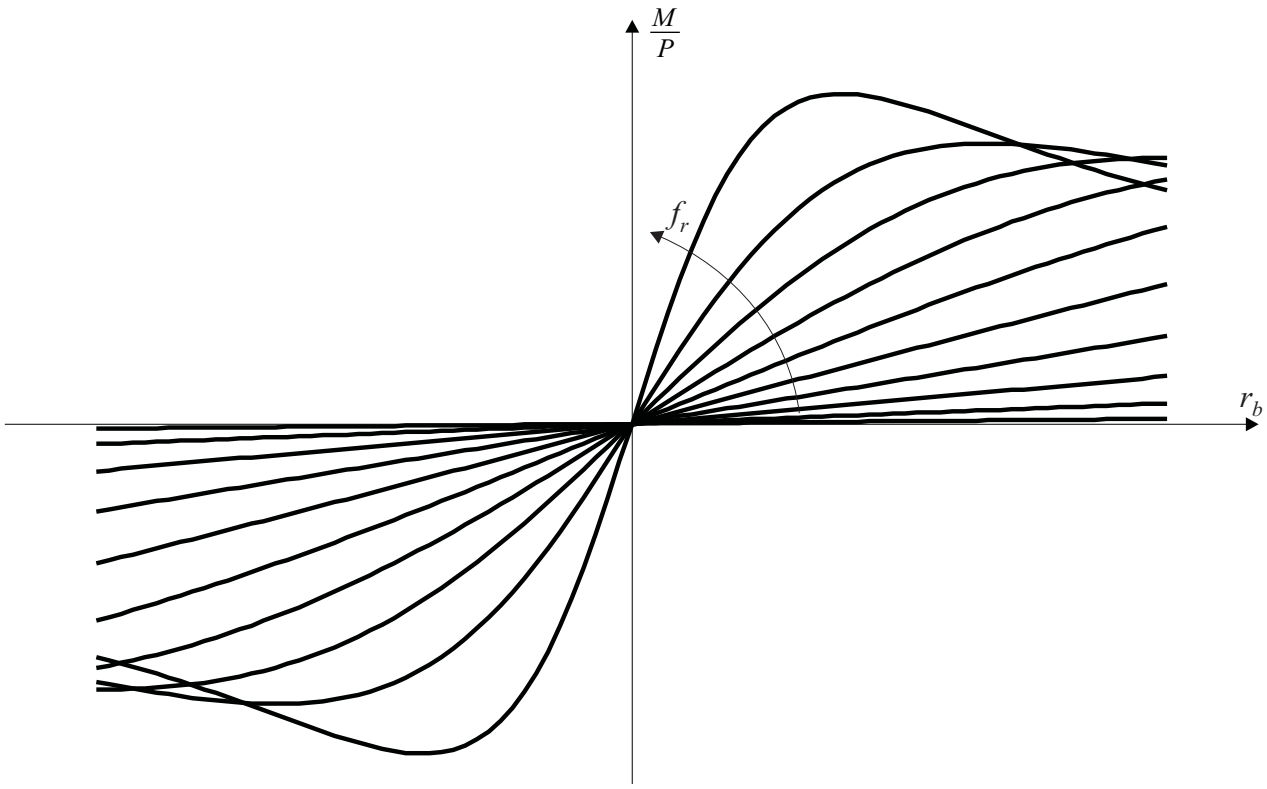


Figura 3.33: Relación entre la función M/P y el radio r_b para distintos valores de la frecuencia radial f_r.

La función de dispersión $h(x,y)$ es, como en el resto de casos, una función con simetría circular. Los momentos de dicha función se definen como:

$$h_{mn} = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} x^m y^n h(x,y)\,\mathrm{d}x\,\mathrm{d}y \tag{3.30}$$

Si consideramos ahora la convolución de la imagen enfocada con la función de dispersión $h(x,y)$:

$$i_1(x,y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} i(x-s,y-t)\,h(s,t)\,\mathrm{d}s\,\mathrm{d}t \tag{3.31}$$

$$i_1(x,y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} \sum_{0\leqslant n+m\leqslant 3} \frac{(-1)^{m+n}}{m!n!} i^{mn)}(x,y)\,s^m t^n h(s,t)\,\mathrm{d}s\,\mathrm{d}t \tag{3.32}$$

Sacando fuera de la integral los términos que no dependen de s o t:

$$i_1(x,y) = \sum_{0\leqslant n+m\leqslant 3} \frac{(-1)^{m+n}}{m!n!} i^{mn)}(x,y) \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} s^m t^n h(s,t)\,\mathrm{d}s\,\mathrm{d}t \tag{3.33}$$

donde la integral doble se corresponde con los momentos de orden m,n de la función de dispersión $h(x,y)$ descrita en (3.30), con lo que obtenemos finalmente:

$$i_1(x,y) = \sum_{0\leqslant n+m\leqslant 3} \frac{(-1)^{m+n}}{m!n!} i^{mn)}(x,y) h_{mn} \tag{3.34}$$

La última expresión corresponde a la transformada S, e implica la convolución de dos funciones en términos de derivadas parciales de i (imagen enfocada) con los momentos de $h(x,y)$ (función de dispersión). Debido a que la función de dispersión tiene simetría circular, tenemos que [Subbarao94]:

$$h_{01} = h_{10} = h_{11} = h_{30} = h_{03} = h_{21} = h_{12} = 0 \tag{3.35}$$

No es necesario evaluar momentos de orden superior, ya que según (3.34), no afectan en la convolución. También tenemos que:

$$h_{02} = h_{20} \tag{3.36}$$

y de acuerdo a (3.13), $h_{00} = 1$. Por tanto, podemos expresar la convolución entre la imagen enfocada y la función de dispersión como:

$$i_1(x,y) = i(x,y) + \frac{h_2}{2}\left(i^{20)}(x,y) + i^{02)}(x,y)\right) \tag{3.37}$$

donde $h_2 = h_{20} = h_{02}$ se corresponde con la varianza de la función de dispersión. Aplicando la segunda derivada a la expresión anterior respecto a x o respecto a y, y teniendo en cuenta que las derivadas de orden mayor de 3 son nulas, al ser i una función cúbica, obtenemos:

$$i^{20)}(x,y) = i_1^{20)}(x,y) \tag{3.38}$$

y

$$i^{02)}(x,y) = i_1^{02)}(x,y) \tag{3.39}$$

Por tanto, tenemos que:

$$i(x,y) = i_1(x,y) - \frac{h_2}{2}\nabla^2 i_1(x,y) \tag{3.40}$$

La expresión anterior relaciona la imagen enfocada i en función de la imagen desenfocada, sus derivadas y los momentos de la función de dispersión. Por tanto, si la imagen es conocida (obtenida a través de una cámara *pinhole* por ejemplo), la expresión anterior nos permite calcular el valor de la desviación típica de la función de dispersión. La relación entre la desviación típica y el radio del círculo de emborronamiento depende de la función de dispersión elegida, como pueden ser la función *Pillbox* o la función gaussiana bidimensional.

3.2.3. Sistema implementado

De acuerdo con la bibliografía existente en el campo de la estimación de distancias mediante procesado de imágenes, y en concreto, la visión estéreo y la estimación de distancias por desenfoque, las principales ventajas de esta última técnica respecto a la visión estéreo es su menor complejidad computacional y el uso de una única cámara, en lugar de dos o más. Sin embargo, si tratamos de emplear, al igual que hicimos con la visión estéreo, estos algoritmos a la estimación de distancias en sistemas de video vigilancia, vemos que nos encontramos con distintos problemas. El principal es el debido al hecho de que, al necesitar la captura de dos o más imágenes con distinto nivel de enfoque, si empleamos una única cámara, las imágenes no pueden ser capturadas al mismo tiempo. Si tenemos en cuenta que en las imágenes a procesar van a existir objetos en movimiento, que es precisamente sobre los que queremos calcular su distancia, nos damos cuenta de que este método no va a ser válido. Si empleamos la opción propuesta por [Pentland87] que utiliza dos cámaras enfocando a la misma escena con enfoques distintos, tal y como se pudo ver en la figura 3.31, la captura al mismo tiempo de ambas imágenes sí que sería posible. Sin embargo, no hay que perder de vista que con esta solución sería necesario el uso de dos cámaras, con lo cual, el coste del sistema de captura de imágenes sería, como poco, igual al de un sistema estéreo de dos cámaras.

Además, tal y como se verá en la siguiente sección, la precisión en la distancia estimada por un sistema estéreo es, con gran diferencia, mucho mayor que la alcanzada por

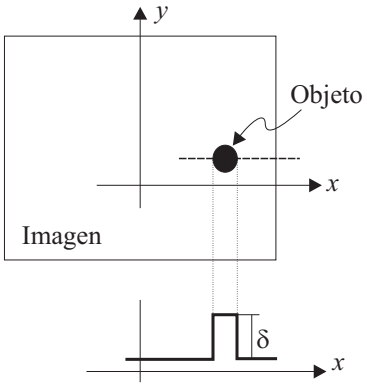


Figura 3.34: Imagen de un objeto sobre el que se calculará su distancia a la cámara. La diferencia entre el nivel de gris del objeto y el fondo es igual a δ.

un sistema de desenfoque. Por último, y debido a que en el sistema de videovigilancia implementado en este trabajo, la estimación de distancias sólo se realiza sobre los objetos detectados, que constituyen una porción muy pequeña de la imagen, la carga computacional se reduce en gran medida, no debiendo ser por tanto la carga computacional un factor discriminante a la hora de determinar las ventajas e inconvenientes de ambos sistemas. Así pues, el único caso en el que el sistema de estimación de distancias por desenfoque sea preferible a un sistema estéreo es cuando se utiliza una única cámara con enfoque fijo, tal y como se describe a continuación. La ventaja proviene del hecho de que solamente empleamos una cámara, que puede ser de las mismas características de las que emplearíamos en estéreo, en lugar de dos o más.

Aunque, tal y como se ha comentado en la sección anterior, se necesitarían al menos dos imágenes con distinto desenfoque para poder realizar el cálculo de distancias, también sería posible realizar dicho cálculo con una única imagen si se tienen en cuenta ciertas restricciones sobre la escena vigilada. Si consideramos, igual que se hizo para el sistema de visión estéreo, que la salida del bloque de detección de movimiento se corresponde con los objetos que están en movimiento en ese instante, y que constituyen los elementos sobre los que queremos calcular su distancia, la estrategia consiste en centrarnos únicamente en los bordes de dichos objetos, que pueden ser calculados fácilmente a partir de la máscara de movimiento. Por tanto, consideremos el borde del objeto como un escalón de valor δ, como se muestra en la figura 3.34.

Tal y como se ha comentado anteriormente, la imagen resultante puede entenderse como la convolución de la imagen enfocada con la función de dispersión producida por la lente.

$$i\left(x,y\right) = i_{\mathrm{f}}\left(x,y\right) * h\left(x,y\right) \tag{3.41}$$

donde $i\left(x,y\right)$ es la imagen capturada, e $i_{\mathrm{f}}\left(x,y\right)$ es la imagen enfocada. Aunque existen muchas funciones de dispersión $h\left(x,y\right)$ para modelar el efecto de desenfoque producido

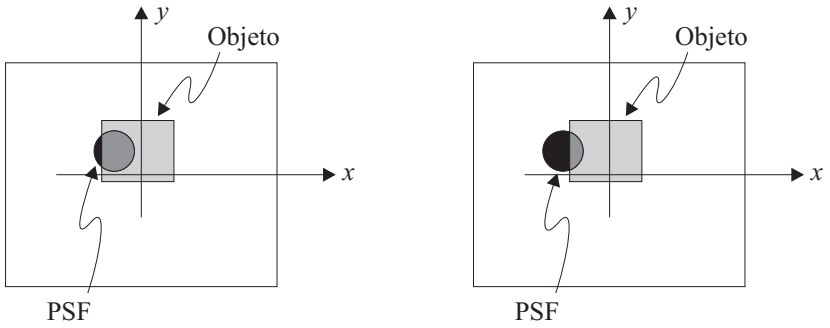


Figura 3.35: Cálculo de la señal de salida como la convolución de la función PSF con el objeto de primer plano.

por la lente, en este trabajo emplearemos una de las más populares, la denominada *pillbox*, debido a que, aparte de ser una aproximación bastante buena para la mayoría de ópticas existentes en el mercado, nos va a simplificar en gran medida los cálculos a realizar, y por tanto, la complejidad computacional del algoritmo a implementar. En nuestro caso, vamos a definir la función *pillbox* según la siguiente expresión:

$$h\left(x,y\right)=\frac{1}{\pi r_b^2}\prod\left(\frac{\sqrt{x^2+y^2}}{r_b}\right) \tag{3.42}$$

donde $\Pi\left(r\right)$ es la función rectangular que toma el valor 1 para $r<1$, y 0 en el resto, y r_b es el radio de desenfoque, (ver figura 3.27), que está relacionado con la distancia a la que se encuentra el objeto de acuerdo a (3.11) ó (3.12). Dependiendo de si el objeto se encuentra delante o detrás del punto de enfoque perfecto podríamos tener dos soluciones distintas para un mismo radio r_b. Para evitar esta ambigüedad, es imprescindible que todos los objetos de interés estén situados, o bien más allá del punto de enfoque perfecto, o bien más cerca de dicho punto.

Como r_b varía con la distancia al objeto, la función de dispersión también varía, y por tanto, (3.41) corresponde a un sistema variante en el espacio (x,y). Sin embargo, como sólo estamos interesados en los pequeños objetos del primer plano, y más concretamente en sus bordes, podemos considerar, sin gran error, que la distancia de todos los puntos del contorno del objeto están a la misma distancia, y por tanto, (3.41) sí puede considerarse como un sistema invariante en el espacio, al menos en el entorno del objeto.

Consideremos un borde vertical de un objeto de primer plano. Como la señal $i_{\mathrm{f}}\left(x,y\right)$ es constante a lo largo del eje y, también lo será la señal $i\left(x,y\right)$. Además, como la convolución implica el desplazamiento de la función *pillbox* sobre el borde vertical, la salida $i\left(x,y\right)$ puede calcularse fácilmente como la superficie del círculo correspondiente a la función PSF que se solapa, para cada posición en el eje x, con el escalón vertical, tal y como muestra la figura 3.35.

Por tanto, la salida puede expresarse como:

$$i(x,y) = \begin{cases} 0 & x < -r_b \\ S(x) & -r_b < x < r_b \\ \delta & x > r_b \end{cases} \tag{3.43}$$

donde $S(x)$ se corresponde con la superficie del segmento circular de la PSF que se solapa con el escalón vertical del objeto de primer plano. Este cálculo puede verse de forma gráfica en la figura 3.36. Si calculamos la primera derivada de (3.43), obtendremos:

$$\frac{\mathrm{d}i(x,y)}{\mathrm{d}x} = \begin{cases} 0 & x < -r_b \\ \frac{2\delta}{\pi r_b^2}\sqrt{r_b^2 - x^2} & -r_b < x < r_b \\ 0 & x > r_b \end{cases} \tag{3.44}$$

que se corresponde con una semi-elipse con un eje horizontal de valor $2r_b$, y un semieje vertical de valor $2\delta/\pi r_b^2$. A partir de (3.44), podemos estimar los parámetros de la semi-elipse obtenida de la primera derivada en los bordes de los objetos, y de acuerdo con (3.11) ó (3.12) y los parámetros de la cámara utilizada, obtener la distancia del objeto al sensor, que es precisamente lo que andábamos buscando.

Ejemplo práctico

Para comprobar el funcionamiento del algoritmo anterior, se ha realizado el experimento mostrado en la figura 3.37. Este experimento ha consistido en la grabación de una secuencia de video con una cámara cuya óptica ha sido ajustada para enfocar los puntos situados a una distancia de 1,4 metros. En 3.37(a) se muestra la estimación del fondo realizada a partir de las primeras imágenes de la secuencia. En (b), (c) y (d) se muestran tres marcos de dicha secuencia, donde aparece un objeto de primer plano situado a distintas distancias.

Una vez realizada la detección de movimiento y localizado el objeto de primer plano, o al menos sus bordes, el sistema puede obtener una señal unidimensional a partir de un corte de la imagen alrededor del borde del objeto, siguiendo una trayectoria normal a dicho borde. Para reducir al máximo posible el ruido introducido por la cámara, se pueden obtener un número considerable de éstas, y promediar su valor. Sin embargo, no hay que perder de vista que uno de los parámetros fundamentales de las ecuaciones (3.43) ó (3.44) es el valor del escalón δ calculado como la diferencia entre el nivel de gris del objeto y el nivel de gris del fondo alrededor de éste, que puede ser distinto en cada uno de los puntos del objeto pertenecientes a su borde exterior. Para evitar este problema, previo al promediado de todas las señales obtenidas, el sistema normaliza éstas, de tal forma que el fondo siempre tenga un valor determinado, mientras que el objeto otro distinto, y la diferencia entre ambas sea un valor constante δ. Para ello, dado que conocemos el punto se separación entre fondo y objeto, devuelto por la función de detección de movimiento, se calcula el nivel medio de gris del objeto, y el nivel medio de gris del fondo, en las inmediaciones del borde, y se transforma dicha señal para que en la señal normalizada, el

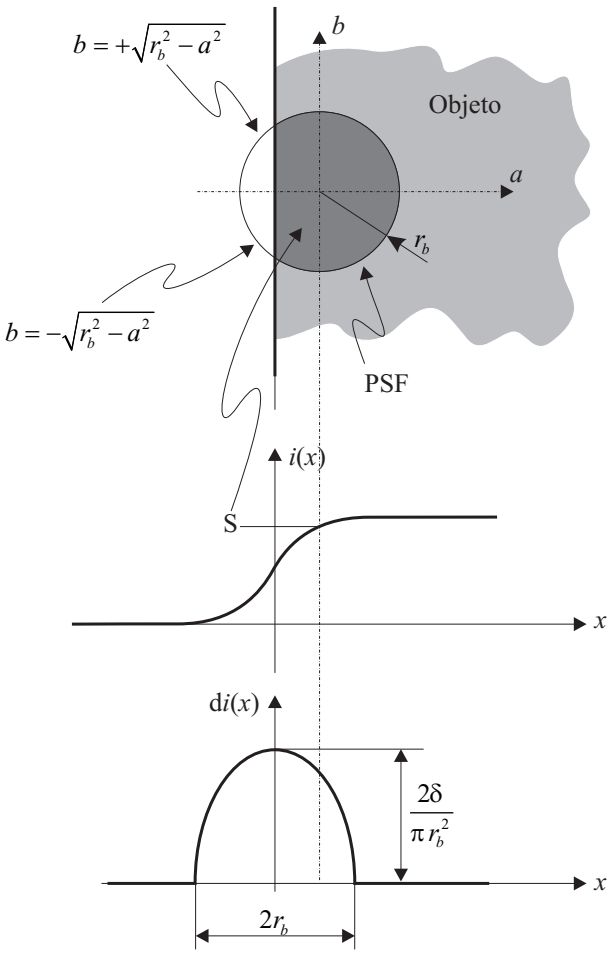


Figura 3.36: Demostración gráfica del cálculo de la convolución entre la función PSF y el objeto de primer plano.

fondo tenga un valor de 5, mientras que el objeto un valor de 0. Para evitar una influencia excesiva del desenfoque en el cálculo de los niveles medios, en dichos cálculos se prescinden de los 5 píxeles más próximos al borde, tanto en la parte del objeto, como en la parte del fondo, y el cálculo se hace con los 10 píxeles siguientes.

En la figura 3.38 se muestra la primera derivada de la señal obtenida a partir del promedio de unos 20 cortes de la imagen normales al borde del objeto, para tres distancias distintas. En esta figura podemos comprobar en primer lugar como la forma de la señal así obtenida se asemeja a la semi-elipse teórica indicada en (3.44), y en la figura 3.36. En los tres casos, debido a que el escalón es negativo (el nivel del objeto es 0 mientras que el de el fondo es 5), la primera derivada se corresponderá con la parte negativa de la semi-elipse de (3.44). El hecho de tomar un escalón positivo o negativo no va a influir en el resultado final. Además, como era de esperar, a medida que el desenfoque es mayor (en este caso al disminuir la distancia con respecto a la cámara, ya que la distancia focal es mayor que cualquiera de las distancias consideradas) la altura de la semi-elipse disminuye, mientras que su anchura aumenta, tal y como indican las expresiones en la figura 3.36.

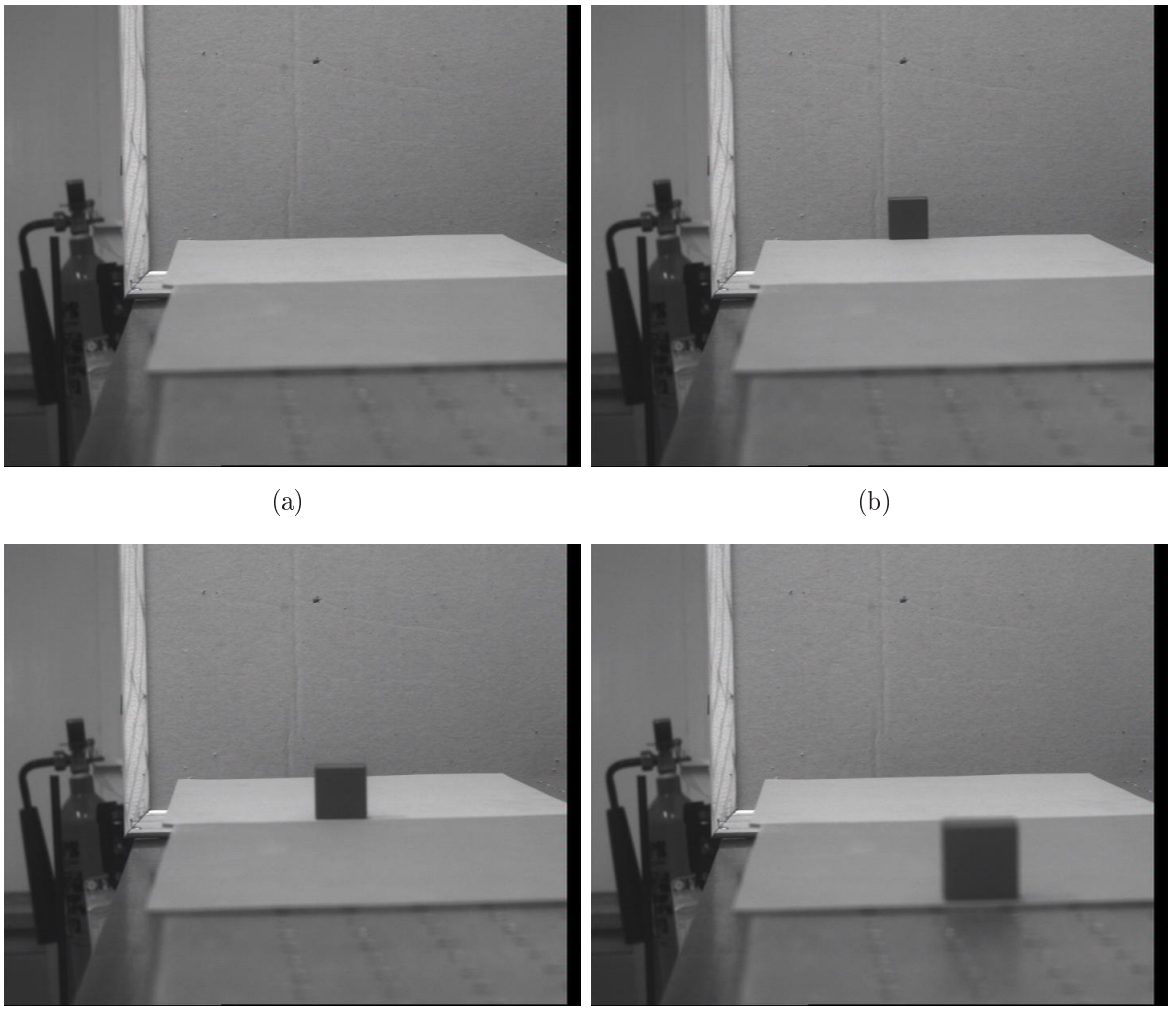
A partir de la altura de la elipse, que puede obtenerse fácilmente como el máximo de la primera derivada, podemos obtener el valor del radio r_b de la función de dispersión. Finalmente, la distancia al objeto, que es el objetivo de este algoritmo, podrá calcularse a partir de (3.12), ya que en este experimento los objetos están situados todos por delante del punto de enfoque perfecto.

Sin embargo, para poder obtener el valor de u en metros, las variables anteriores deberán estar expresadas también en metros. Sin embargo, r_b, el radio de la función de dispersión, según se vio anteriormente, vendrá expresada en píxeles. De esta forma, se hace necesario transformar dicho valor a metros, siendo imprescindible por tanto, conocer el tamaño de cada píxel del sensor CCD de la cámara. Esta transformación nos dará cuál es el tamaño del radio r_b proyectado sobre el sensor CCD.

Además, según puede verse en la expresión (3.12), así como también en (3.11), son necesarios los valores de f (longitud focal de la lente), y v_0 (distancia entre el eje de la lente y el plano imagen, el CCD en este caso). Si bien el parámetro f es un valor que es generalmente aportado por el fabricante de la lente, el caso del parámetro v_0 no lo es, ya que, entre otras cosas, dicho valor varía al variar la distancia focal, es decir, cuando se ajusta la óptica de la lente para enfocar a una distancia concreta. Ahora bien, una vez enfocada la lente, es decir, conocida la distancia de enfoque perfecto, el valor de v_0 ya no varía, por lo que puede ser obtenido mediante la siguiente expresión, derivada a partir de la ley de la lente delgada (3.7):

$$v_0 = \frac{f u_0}{u_0 - f} \tag{3.45}$$

Todos los parámetros anteriores, que se derivan de la geometría óptica de la cámara, y que son fijos para una configuración concreta de la cámara, suelen denominarse como parámetros intrínsecos, a diferencia de los extrínsecos, que dependen de la escena, y de



(a) (b)

(c) (d)

Figura 3.37: (a) Estimación del fondo empleada por el algoritmo de cálculo de distancias mediante el desenfoque de la imagen. La secuencia ha sido grabada con una distancia focal de 1,4 metros. (b,c,d) Objetos del primer plano situados a (a) 1,25 m, (b) 1,10 y (c) 0,95 m. Nótese como el desenfoque aumenta conforme el objeto se acerca a la cámara (téngase en cuenta que la distancia focal es de 1,4 m)

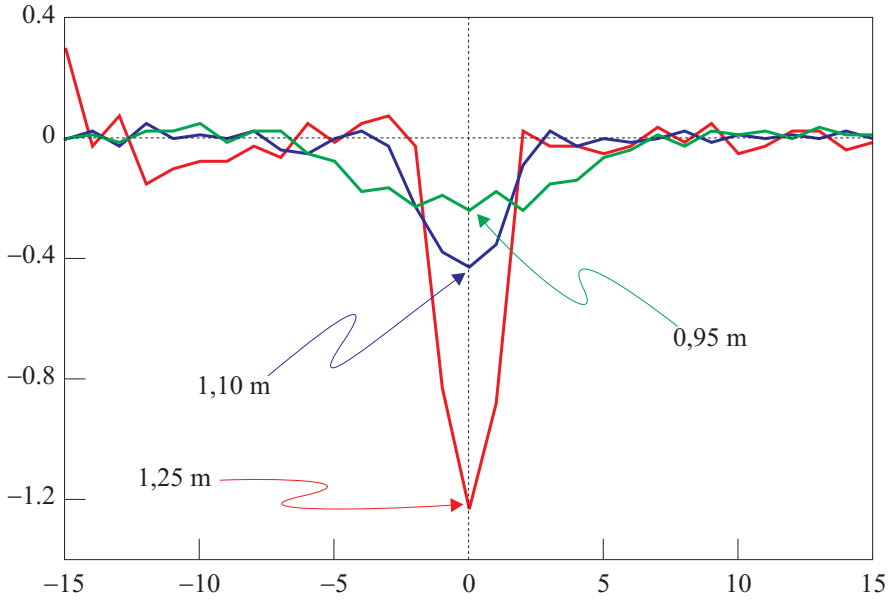


Figura 3.38: Primera derivada de la normal al borde del objeto para tres distancias distintas entre el objeto y la cámara.

Tabla 3.2: Valores de los parámetros intrínsecos de la cámara necesarios para realizar el cálculo de distancias de acuerdo al algoritmo propuesto

Parámetro	**Descripción**	**Valor**
a	Radio de la apertura (ver figura 3.27)	10 mm
f	Longitud focal de la lente	18 mm
u_0	Distancia de enfoque perfecto	1.4 m
v_0	Distancia entre la lente y el plano imagen	18.23 mm
Δ	Tamaño de un píxel del CCD de la cámara	6μm

la posición de la cámara respecto a ésta. En la tabla 3.2 están recogidos todos estos parámetros intrínsecos de la cámara empleada en este experimento. En esta tabla se incluyen los valores de los parámetros aportados por el fabricante (longitud focal, radio de la apertura y tamaño del píxel en el CCD), y los parámetros calculados a partir de medidas indirectas (distancia entre la lente y el plano imagen). Con estos parámetros, junto con los resultados obtenidos del análisis de las imágenes anteriores, podemos realizar una estimación de la distancia a la cual se encuentran los objetos detectados. En la tabla 3.3 se pueden comprobar los resultados obtenidos, y compararlos con los datos reales.

A simple vista puede comprobarse que el error cometido para todas las distancias analizadas (columna e de la tabla 3.3) es bastante elevado. Esto es debido entre otras cosas, a todas las aproximaciones realizadas. Además, si observamos más detenidamente la expresión (3.10), podemos comprobar como dicha expresión es muy sensible a ligeras

Tabla 3.3: Distancia estimada para los tres objetos detectados en la imagen 3.37. h representa la altura de la semi-elipse calculada a partir de la derivada de la señal indicada por la expresión (3.44). r_b es el radio de la función de dispersión, medida en píxeles sobre la imagen, y en micrometros sobre el CCD. d es la distancia real a la que se encontraban los objetos. $\overline{d}$ es la distancia estimada, a partir de la expresión (3.12) sin calibración y e es el porcentaje de error en la medida estimada. Por último $\overline{d'}$ y e' son las mismas medidas anteriores obtenidas con el sistema calibrado.

h	r_b		d	$\overline{d}$	e	$\overline{d'}$	e'
	píx.	μm	m	m	%	m	%
0.253	15.86	95.16	0.95	0.8177	13.93	0.9670	1.79
0.425	12.24	73.44	1.10	0.9060	17.64	1.0719	2.55
1.22	7.22	43.32	1.25	1.0655	14.76	1.2619	0.95

variaciones en los valores de los parámetros. Un razonamiento más comprensible de todo esto, analizándolo desde el punto de vista de la geometría de la cámara, será desarrollado en la siguiente sección. La conclusión de esta primera evaluación es que, el cálculo de distancias utilizando los valores de los parámetros aportados por el fabricante de la cámara es bastante impreciso.

No obstante a lo anterior, también hemos podido comprobar como las imágenes procesadas, en particular, las señales obtenidas en la figura 3.37, se comportan tal y como era de esperar del análisis teórico realizado anteriormente. Gracias a esto, podemos pensar en desarrollar un sistema de calibración que permita estimar, a partir de medidas indirectas en las imágenes, los parámetros anteriores. Es decir, si reescribimos la expresión (3.10) como sigue:

$$u = \frac{f v_0 a}{(v_0 - f)\, a \pm f r_b} = \frac{c_1}{c_2 \pm c_3 r_b} = \frac{1}{k_2 \pm k_3 r_b} \tag{3.46}$$

donde vamos a denominar las constantes c_1, c_2 y c_3, ó k_2 y k_3 como *constantes de calibración*. Además, dichas constantes pueden identificarse con los parámetros intrínsecos de la cámara como sigue:

$$\begin{cases} \dfrac{c_2}{c_1} = k_2 = \dfrac{v_0 - f}{f v_0} \\ \dfrac{c_3}{c_1} = k_3 = \dfrac{1}{v_0 a} \end{cases} \tag{3.47}$$

Las expresiones anteriores forman ecuaciones lineales en las constantes de calibración. De las dos ecuaciones, es preferible la última (k_2 y k_3), ya que resulta en una ecuación inhomogénea en los valores de las constantes de calibración, mientras que la primera sería una ecuación homogénea, que como es sabido, presenta mayor dificultad para su resolución.

Si disponemos de varias medidas donde los valores de u y r_b sean conocidas, podemos formar un sistema de ecuaciones lineales inhomogéneo que nos permitan obtener, ya no los valores individuales de los parámetros de la cámara, que dejan de ser necesarios, sino los valores de las constantes que nos permitan tener el sistema calibrado:

$$\begin{pmatrix} u_1 & \pm u_1 r_{b1} \\ \vdots & \vdots \\ u_N & \pm u_N r_{bN} \end{pmatrix} \begin{pmatrix} k_2 \\ k_3 \end{pmatrix} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \tag{3.48}$$

siendo la expresión con + para sistemas donde los objetos están más cerca de la cámara que el punto de enfoque perfecto, y el − para el caso contrario. Cómo solo existen dos parámetros (k_2 y k_3) solo es necesario un mínimo de dos medidas. Si disponemos de más medidas, podremos formar un sistema lineal sobredeterminado que permitirá al sistema ser más inmune al ruido debido a errores en cada medida individual.

Para ilustrar este razonamiento, se tomaron 6 nuevas medidas, para distancias comprendidas entre 0,80 y 1,25 metros. Es necesario recordar que la distancia de enfoque perfecto está a 1,4 metros, y por tanto, dado que estamos considerando sólo distancias por delante, ninguna medida puede ser superior a dichos 1,4 metros. A partir de estas medidas, podemos calcular los valores de los parámetros de calibración k_2 y k_3 resolviendo el sistema de ecuaciones dado por (3.48). Finalmente, con estos dos valores, y empleando la expresión (3.46), ya sería posible realizar la estimación de distancias una vez conocido el valor del radio r_b producido por un objeto determinado. Por ejemplo, en la tabla 3.3 se muestran los resultados del cálculo de distancias para los tres ejemplos anteriores, en las columnas $\overline{d'}$ y e' empleando este nuevo método, para poder compararlos con los resultados que obtuvimos sin realizar el proceso de calibración.

Principalmente podemos comprobar como la precisión a aumentado notablemente con respecto al primer cálculo, a pesar de que en el primer caso se estaban empleando los valores reales de los parámetros dados por el fabricante. No obstante, hay que tener en cuenta que, igual que en muchas de los trabajos relacionados con el cálculo de distancias mediante desenfoque, se han realizado muchas aproximaciones que son las que provocan los errores en la precisión del sistema. De hecho, a partir de (3.47), y con los valores para las constantes de calibración k_2 y k_3 obtenidas de la resolución del sistema de ecuaciones (3.48), se podrían obtener los parámetros intrínsecos de la cámara, de acuerdo a:

$$\begin{cases} v_0 = \dfrac{f}{1 - k_2 f} \\ a = \dfrac{k_3}{v_0} \end{cases} \tag{3.49}$$

En la tabla 3.4 se muestran los resultados obtenidos, tanto para las constantes de calibración k_2 y k_3, como para los parámetros v_0 (distancia de separación entre la lente y el sensor CCD) y a (radio de la apertura), tomando como valor conocido la distancia focal, $f = 18$mm, de la lente, que, en general, suele ser el parámetro dado con mayor

Tabla 3.4: Parámetros intrínsecos de la cámara calculados a partir de medidas realizadas en el desenfoque de los objetos.

Parámetro	Valor
k_2	0,5905 m^{-1}
k_3	$4{,}6622 \cdot 10^3$ m^{-2}
a	11,79 mm
v_0	18,19 mm

precisión por el fabricante. La principal conclusión que podemos obtener de esta tabla es que, aunque ligeramente distintos, los valores obtenidos son cercanos a los datos físicos reales dados por el fabricante, lo cual nos da una pista de la validez de esta segunda aproximación.

3.3. Comparación entre visión estereoscópica y profundidad por desenfoque

Una vez vistas las características principales de las técnicas de visión estereoscópica y las técnicas de desenfoque, realizaremos en este punto una enumeración de las ventajas e inconvenientes de ambas técnicas.

La principal ventaja de las técnicas de desenfoque con respecto a la visión estéreo es la ausencia del problema de la ambigüedad en la correspondencia entre puntos y de oclusiones. Las oclusiones van a provocar en general errores en la visión estéreo, y el problema de la correspondencia provoca entre otras cosas, una complejidad computacional elevada, y en muchas ocasiones, sobre todo cuando la textura de la imagen no es elevada, errores. La ausencia de estos problemas en las técnicas de desenfoque van a mejorar, por tanto, el rendimiento de estos últimos. El principal inconveniente de las técnicas de desenfoque con respecto a la visión estereoscópica es su baja resolución y sensibilidad.

Estas características, que inicialmente fueron observadas de forma empírica, pueden ser analizadas de una forma más teórica, llegando a la conclusión de que ambos sistemas no son realmente tan diferentes [Schechner98]. En la figura 3.39 (a) se muestra una cámara con una lente real de diámetro D, y una separación entre la lente y el plano imagen de valor v. Si consideramos que la distancia de enfoque perfecto es u, un objeto situado a una distancia u_1 se visualizará en el plano imagen con un desenfoque producido por una función de dispersión de radio r_b. Consideremos ahora un sistema estéreo con las mismas dimensiones físicas que el sistema anterior. Esto significa que la distancia focal para ambas cámaras del par estéreo es v y que la distancia entre los centros de proyección es igual al diámetro de la lente, es decir, D. En este caso, se puede comprobar que la diferencia

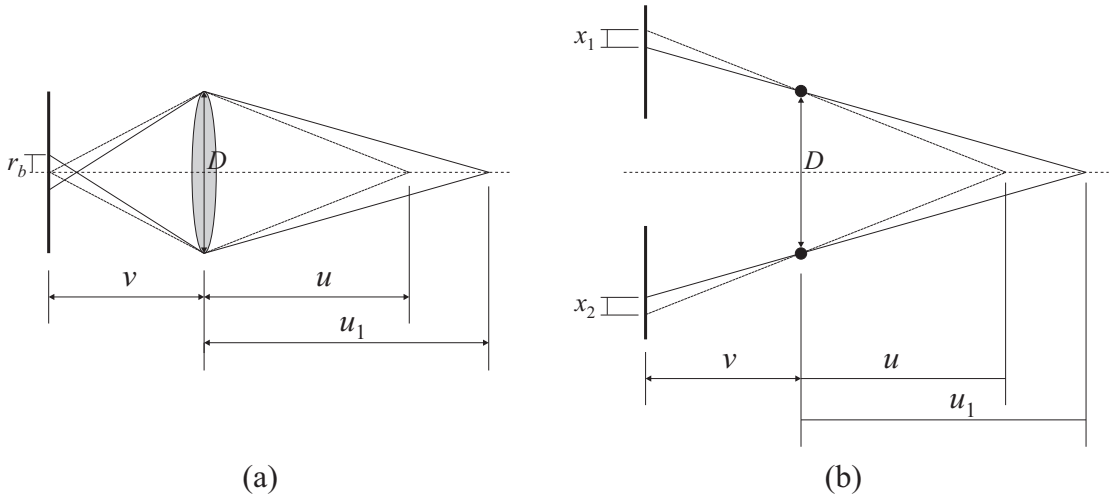


Figura 3.39: Comparación geométrica de la resolución entre un sistema de desenfoque y un sistema estéreo.

entre la disparidad que presentaría un punto situado a la distancia u y otro situado a la distancia u_1 sería:

$$x_1 - x_2 = 2r_b \tag{3.50}$$

Lo que indica que un sistema estéreo con las mismas dimensiones físicas que un sistema DFF o DFD debería presentar resoluciones similares. En la práctica, sin embargo, la diferencia estriba en que, mientras en sistemas de desenfoque el tamaño de la apertura suele ser del orden de uno o dos centímetros (tamaños mayores implican lentes más grandes, lo cual es económicamente más caro, o incluso físicamente irrealizables), en un sistema estéreo, la distancia entre los centros de proyección de las cámaras puede aumentarse a voluntad sin necesidad de aumentar el coste del sistema.

Siguiendo el razonamiento anterior, también podemos comprobar fácilmente que un sistema de desenfoque tampoco está libre del problema de las oclusiones. Si observamos la figura 3.40, podemos comprobar como un objeto de dimensiones comparables al sistema óptico provoca una oclusión en la imagen visualizada. El efecto que produce es que reduce el tamaño de la función de dispersión. En realidad, la función de dispersión dejaría de tener simetría circular. Por ejemplo, en la figura 3.40, si el objeto es suficientemente grande en sentido horizontal, la función de dispersión sería un semicírculo. Sin embargo, el problema de la oclusión no es tan grave como en los sistemas estéreo, donde para cada píxel se debe decidir cual es su correspondiente en la otra imagen. Si esa decisión es incorrecta, dará lugar a una profundidad que no tiene nada que ver con la realidad. En los sistemas de desenfoque, la oclusión provoca que la función de dispersión sea ligeramente distinta a la teóricamente considerada, lo que provocará imprecisiones, pero en general no provocará errores tan graves.

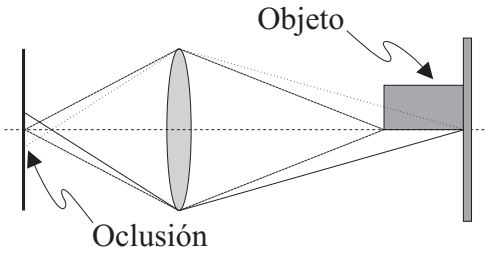


Figura 3.40: Efecto de la oclusión en sistemas de desenfoque.

Este último razonamiento puede verse como una particularización de un concepto más genérico que es la robustez. En sistemas estéreo, el cálculo de la profundidad de un punto de la escena se realiza con la información de dos puntos únicamente (rayos de luz), mientras que en sistemas de desenfoque, el cálculo está basado en mucha mayor cantidad de luz, la que atraviesa la apertura de la lente, por lo que la relación señal a ruido se verá mejorada. Además, hay que tener en cuenta que en sistemas de desenfoque la apertura es de dos dimensiones, mientras que en sistemas estéreo el desplazamiento de las cámaras se produce en una única dimensión. Como se vio al analizar los sistemas estéreo, estos tienen gran dificultad para calcular la disparidad en bordes paralelos a la línea base. En sistemas de desenfoque, al producirse el emborronamiento en dos dimensiones, la profundidad puede calcularse independientemente de la orientación de los bordes.

Capítulo 4

Clasificación de formas

Una vez detectados los objetos de interés dentro de la escena vigilada, el siguiente paso suele consistir en reconocer los objetos resultantes, y una posible opción sería comenzar con el reconocimiento de su forma. El procedimiento de clasificación de forma en sí se puede dividir en dos bloques. El primero, y generalmente más importante, se denomina *representación y descripción* y consiste en la selección de las características más representativas de los objetos, que permita un funcionamiento lo más fiable posible del segundo bloque, que es la *clasificación* propiamente dicha.

Por ejemplo, si nuestro interés radica en clasificar objetos de acuerdo a su relación de aspecto o su color, una característica bastante sencilla podría ser la relación de momentos de orden 2 del blob detectado para poder clasificar de acuerdo a su relación de aspecto, o el color de la región interior del blob para clasificar por colores. Obviamente, a medida que las formas a clasificar son más complejas, y parecidas entre sí, el problema se vuelve más complicado y se hace necesario el realizar la selección de características de una forma más cuidadosa. Al conjunto de todos los descriptores del objeto suele denominarse como *vector de clasificación*. Por otro lado, el bloque de clasificación determina, a partir del vector de clasificación de un objeto determinado, la clase a la que pertenece, en este caso, su forma, dentro del conjunto de formas consideradas. Existen numerosos algoritmos para la implementación de este bloque, desde un simple cálculo del error cuadrático medio entre las muestras del vector de clasificación del objeto analizado y los patrones de referencia, hasta esquemas más evolucionados, como pueden ser las Redes Neuronales (NN) [Haykin99] o las Máquinas de Vectores Soporte (SVMs) [Vapnik00]. En este capítulo nos centraremos principalmente en el estudio del bloque de representación y descripción, mientras que la clasificación se realizará con el algoritmo conocido como *Vecino más cercano* (NN: Nearest Neighbour) [Theodoridis03].

La representación de una región implica el uso de dos posibilidades, o ambas a la vez, hacerlo en términos de sus características externas (su contorno), o en términos de sus características internas (los píxeles que comprenden la región). Generalmente, la representación externa se elige cuando el objetivo principal se centra en las características de forma, y la representación interna cuando el principal interés se centra en propiedades

tales como el color y textura. Por tanto, si nuestro objetivo es la clasificación de la forma detectada, está claro que únicamente será necesario el uso del contorno del objeto, prescindiendo completamente del interior de éste.

El siguiente paso es la elección de los descriptores del objeto. Independientemente del objetivo final, es interesante que las características seleccionadas como descriptores sean tan insensibles como fuera posible a variaciones tales como cambios de tamaño, traslación o rotación. Además, también es recomendable que dichas características sean lo más diferentes posibles entre las distintas clases consideradas. Por ejemplo, si quisiéramos distinguir entre cuadrados y triángulos equiláteros, una característica interesante sería el número de vértices presentes en el contorno, mientras que una no recomendable podría ser la excentricidad del objeto.

4.1. Esquemas de descripción

Las técnicas de detección de movimiento vistas hasta ahora, o de segmentación de imágenes en general, generan conjuntos de píxeles, o blobs, del cual puede extraerse su contorno fácilmente [Gonzalez93]. A partir de aquí, la selección de los descriptores del objeto es totalmente dependiente del objetivo final y de las características de las formas a clasificar. Es decir, no es lo mismo realizar la clasificación de formas geométricas sencillas, donde sería interesante centrarse en la descripción mediante rectas, vértices, etc, que clasificar formas más complejas, como vehículos, donde otros esquemas de descripción serían necesarios. A continuación se describen los descriptores de contorno más comunes en la literatura, junto con ejemplos de uso.

4.1.1. Descriptores de Fourier

Partiendo del contorno de un objeto, éste puede ser recorrido, ya sea en sentido horario o antihorario, desde un punto arbitrario para llegar al mismo punto. Los pares de coordenadas correspondientes a cada punto $(x_0, y_0), (x_1, y_1), (x_2, y_2), \ldots, (x_{N-1}, y_{N-1})$ que se encuentran al recorrer el contorno, pueden expresarse en la forma $x(k) = x_k$ y $y(k) = y_k$. Con esta notación, el propio contorno se puede representar como la serie de coordenadas $s(k) = [x(k), y(k)]$, para $k = 0, 1, 2, \ldots, N-1$. Más aun, cada par de coordenadas puede ser tratado como un número complejo tal que:

$$s\left(k\right) = x\left(k\right) + \mathrm{j}y\left(k\right) \tag{4.1}$$

para $k = 0, 1, 2, \ldots, N-1$. Esto es, se toman el eje x como eje real y el eje y como eje imaginario de una serie de números complejos. La transformada discreta de Fourier (DFT) de $s(k)$ es:

$$a\left(u\right) = \frac{1}{N}\sum_{k=0}^{N-1} s\left(k\right)\exp\left(-\frac{2\pi\mathrm{j}uk}{N}\right) \tag{4.2}$$

para $u = 0, 1, 2, \ldots, N-1$. Los coeficientes complejos $a(u)$ se denominan *descriptores de Fourier* del contorno. La transformada inversa de Fourier de los descriptores de Fourier restaura el contorno original. Es decir:

$$s(k) = \sum_{k=0}^{N-1} a(u) \exp\left(\frac{2\pi \mathrm{j} uk}{N}\right) \tag{4.3}$$

para $k = 0, 1, 2, \ldots, N-1$. Supongamos sin embargo, que en vez de todos los coeficientes de $a(u)$, solamente se utilizan los primeros M coeficientes. Esto es equivalente a hacer $a(u) = 0$ para $u > M-1$ en (4.3). El resultado es la siguiente aproximación a $s(k)$:

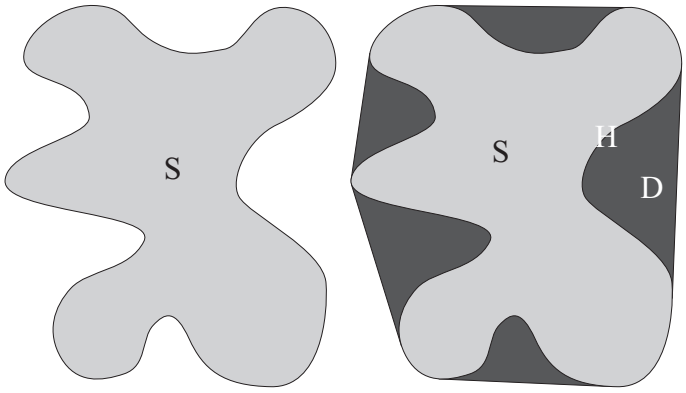
$$\hat{s}(k) = \sum_{k=0}^{M-1} a(u) \exp\left(\frac{2\pi \mathrm{j} uk}{N}\right) \tag{4.4}$$

para $k = 0, 1, 2, \ldots, N-1$. Aunque sólo se utilizan M términos para obtener cada componente de $\hat{s}(k)$, k permanece en el intervalo de 0 a $N-1$, esto es, existe el mismo número de puntos en el contorno aproximado, pero no se utilizan tantos términos en la reconstrucción de cada punto. Si el número de puntos del contorno es grande, generalmente se selecciona N como una potencia de 2 de tal forma que se pueda utilizar un algoritmo FFT para realizar el cálculo de los descriptores.

Los descriptores de Fourier representan la forma del objeto en el dominio de la frecuencia. Los coeficientes de baja frecuencia contienen la información sobre la forma general del objeto, y los coeficientes de alta frecuencia los detalles de ésta. En general, en las aplicaciones de clasificación, suele ser suficiente con emplear un conjunto reducido de coeficientes para poder discriminar entre formas. Sin embargo, hay que tener en cuenta que, en este caso, al ser la señal $s(k)$ compleja, su transformada de Fourier no es simétrica, y por tanto, no se puede explotar la propiedad de simetría que caracteriza la transformada de Fourier de una señal real.

Para conseguir que el sistema sea invariante al desplazamiento, basta con reescribir las coordenadas de los píxeles del contorno $x(k)$ e $y(k)$ con respecto al centro de masas del objeto, en lugar de con respecto al origen de coordenadas de la imagen. La invarianza respecto a la rotación se consigue aprovechando la propiedad de desplazamiento de la transformada de Fourier. De esta forma, descartando la información de fase, y guardando únicamente el módulo de los descriptores de Fourier conseguimos dicha invarianza. Por último, la invarianza al escalado puede conseguirse dividiendo los valores absolutos de los descriptores por el valor absoluto de la muestra en cero, $|a(0)|$ [Gonzalez93].

Los descriptores de Fourier han sido utilizados ampliamente en distintos trabajos. Por ejemplo, en [Kauppinen95], los descriptores de Fourier son utilizados, junto con otros métodos, para realizar la clasificación de letras del alfabeto inglés y modelos de aviones militares. En [Kunttu03], los descriptores de Fourier son empleados para clasificar defectos en el proceso de clasificación de papel, como agujeros, arrugas o suciedades, y defectos en superficies metálicas.



(a) (b)

Figura 4.1: (a): Contorno de una región. (b): Cerco convexo (negro) de la región mostrada en (a).

4.1.2. Cerco y deficiencia convexos

Cuando el contorno de la figura presenta una o más concavidades significativas que contienen información sobre la forma, puede resultar interesante emplear el *cerco convexo (convex hull)*. El cerco convexo H de una región S es el conjunto convexo más pequeño que contiene a S. El conjunto diferencia $H - S$ se denomina *deficiencia convexa D (convex deficiency)*. Estos conceptos están representados en la figura 4.1. Considerando la región S mostrada en 4.1(a), si ampliamos la región trazando rectas en aquellos puntos de S donde se produce una transición de afuera hacia adentro para eliminar todas las partes cóncavas, obtenemos el cerco convexo H, como se representa en 4.1(b). La diferencia de ambas regiones, representada en negro en la figura anterior, sería la deficiencia convexa D.

La principal ventaja de estos descriptores es su invarianza a diferentes transformaciones geométricas. En concreto, características como el número de componentes de la deficiencia convexa o la situación relativa entre éstas no cambia bajo ningún tipo de transformación. Un parámetro empleado frecuentemente es la relación entre la deficiencia convexa y el cerco convexo [Dionisio04]:

$$R = \frac{\text{Area}\,(D)}{\text{Area}\,(H)} \tag{4.5}$$

ya que esta relación no cambia ante una transformación afín. Podemos comprobar esto fácilmente si suponemos T una transformación afín arbitraria [Hartley03] y $\mathbf{A}$ la matriz que representa dicha transformación. Bajo dicha transformación, el área de una región S cualquiera pasa a valer:

$$\text{Area}\,(T\,(S)) = |\mathbf{A}|\,\text{Area}\,(S) \tag{4.6}$$

donde $|\mathbf{A}|$ representa el determinante de $\mathbf{A}$. Es decir, el área de la superficie queda multiplicada por el determinante de la matriz de transformación, y al ser dicha transformación afín, la matriz será no singular, y por tanto, el determinante distinto de 0. De esta forma, en la expresión (4.5), al quedar ambos términos multiplicados por la misma constante, estos se anulan entre sí, lo cual demuestra que dicho parámetro es invariante ante una transformación afín. Este descriptor es utilizado por ejemplo en [Dionisio04], junto con otros descriptores, para realizar la clasificación de gestos estáticos de la mano, entre otras, como aplicación para la interpretación del lenguaje de sordomudos.

4.1.3. Funciones de curvatura

La curvatura del contorno de un objeto en un punto determinado p_i se calcula como la diferencia entre ángulos de las tangentes al contorno entre dicho punto y el siguiente, p_{i+1}, como muestra la figura 4.2. Es decir, el valor de la curvatura en dicho punto puede calcularse como:

$$c_i = \tan^{-1}\frac{y_{i+W} - y_i}{x_{i+W} - x_i} - \tan^{-1}\frac{y_{i+W+1} - y_{i+1}}{x_{i+W+1} - x_{i+1}} \tag{4.7}$$

La función $c(i)$, donde $i = 0, 1, 2, \ldots, N-1$, y N es el número de puntos del contorno, se denomina *función de curvatura*. El termino W permite evitar los errores debidos a la naturaleza discreta de una imagen digital, que se verían reflejados en la función de curvatura como componentes de alta frecuencia. Así, un valor para W pequeño hace que el ruido de alta frecuencia debido a la discretización de la imagen aumente, mientras que un valor grande hará que dichas componentes se reduzcan, pero si éste es excesivamente grande también podrían hacerlo las componentes de baja frecuencia, que son las que llevan la información fundamental de la forma del objeto. Por tanto, se debe optar por un valor de compromiso para W, que evite los problemas anteriormente mencionados. En general, dicho valor va a depender, entre otras cosas, del tamaño del objeto. A mayor tamaño del objeto, dicho valor debería aumentar de forma proporcional [Kauppinen95].

La principal ventaja de la función de curvatura es que reduce un problema bidimensional a uno unidimensional, que suele ser más sencillo de tratar mediante las técnicas tradicionales de procesado de señal. Además, por el hecho de emplear diferencias entre ángulos en lugar de ángulos directamente conseguimos que la función sea invariante a la rotación, aunque no al punto de comienzo. Esto quiere decir que, dependiendo de cual sea el punto del contorno por el que empecemos a obtener la función de curvatura, ésta va a ser distinta, y se verá reflejado como un desplazamiento circular en dicha función. La invarianza al escalado puede conseguirse muestreando la función de curvatura a un número de muestras determinado, para que así todas las funciones de curvatura tengan un tamaño fijo. Por último, la propia naturaleza de la expresión para la obtención de la curvatura según (4.7), hace que sea invariante al desplazamiento.

La función de curvatura es utilizada por ejemplo en [Kauppinen95] para la clasificación de distintos contornos, como letras y aviones. Para evitar la dependencia del punto de

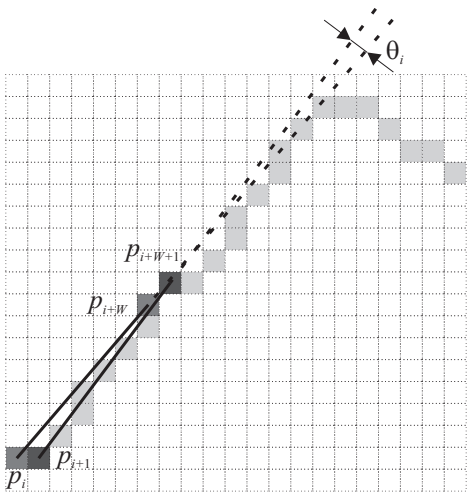


Figura 4.2: Cálculo de la curvatura del contorno en el punto p_i.

comienzo comentado anteriormente, la clasificación se realiza empleando la *Curvatura de Fourier* que es básicamente el módulo de la transformada de Fourier de la función de curvatura. La función de curvatura es muestreada a un valor potencia de 2, de tal forma que, aparte de hacer el algoritmo invariante al escalado, como se vio anteriormente, también permite realizar el cálculo mediante el uso de la FFT.

4.1.4. Firmas

La firma de un objeto es una representación unidimensional del contorno del objeto, y se puede obtener de varias formas. Una de las más simples es representar la distancia desde el centro del objeto al contorno como una función del ángulo. La firma generada por este procedimiento no varía con la traslación, pero dependen de la escala y la rotación. La normalización con respecto a la escala se puede conseguir normalizando la firma obtenida, ya que los cambios de tamaño de un objeto producen cambios en los valores de amplitud de la firma correspondiente. Normalizando la energía o la amplitud de la señal resultante conseguimos que el método sea invariante a la escala.

La principal ventaja de la firma con respecto a los descriptores de Fourier o la función de curvatura es que la longitud de la señal resultante es siempre la misma, igual a 360^o entre el valor del incremento del ángulo entre muestras consecutivas. En los métodos anteriores, y cualquier otro método que recorra el contorno completo del objeto, la longitud total depende de la forma en sí, por lo que necesita ser muestreada para obtener una longitud homogénea para todos los objetos. Además, para una misma forma, errores en la segmentación que hagan que desaparezca un trozo del objeto, puede hacer que la longitud sea diferente para distintos objetos con la misma forma, lo que dificulta el proceso de clasificación posterior.

Sin embargo, el principal inconveniente de la firma respecto a los métodos anteriores es que sólo funciona aceptablemente cuando las formas consideradas no tienen concavidades, o éstas no son muy pronunciadas. Si esto no fuese así, la firma no sería una representación muy exacta del contorno del objeto.

Este método de obtención de la firma no es desde luego el único. En [Becker00] por ejemplo, la firma no se obtiene como la distancia desde el centro del objeto. En este caso, la elección del centro de coordenadas es crítico, ya que es necesario que la firma resultante sea lo más invariante posible a la rotación, a la vez que debe capturar la mayor cantidad de información posible de la forma del objeto. Para ello, el centro de coordenadas se posiciona en el punto del contorno más próximo al centro de masas del objeto.

Debido a las ventajas que presenta el uso de la firma en la clasificación cuando las formas consideradas carecen de concavidades, y dado que las formas consideradas en esta tesis presentan dichas características, la firma, tal y como fue definida al comienzo de esta sección será la base fundamental del sistema desarrollado en esta tesis para realizar la clasificación de formas.

4.1.5. Otros descriptores

Los descriptores vistos hasta este punto no son, obviamente, los únicos que podemos emplear en un determinado proceso de clasificación de formas, pero si son algunos de los más empleados. Como se ha comentado anteriormente, el uso de un descriptor u otro depende fundamentalmente del aspecto de las formas a clasificar, y por tanto, se podría decir que existen casi tantos descriptores como aplicaciones distintas hayan sido desarrolladas.

Por ejemplo, en [Antani03], se realiza la detección de *osteofitos* en las vértebras humanas a partir de radiografías mediante la detección de círculos y el cálculo del radio de dicho círculo. Además, emplea procedimientos de apertura [Gonzalez93] para la detección de *protusiones* en las mismas vértebras. De esta forma, el método puede clasificar, y mucho más importante, detectar problemas médicos en la columna vertebral del paciente analizado.

La clasificación de formas geométricas simples está bastante difundido en la literatura, especialmente debido al creciente interés por la detección y el reconocimiento de objetos, generalmente artificiales, que presentan dichas formas, como pueden ser triángulos, cuadrados o círculos en el caso de señales de tráfico, rectángulos para la interpretación de paneles informativos, etc.

Por ejemplo, en [Ishizuka04] se desarrolla un algoritmo para la detección de círculos eligiendo tres puntos arbitrarios del contorno del objeto, que son empleados para determinar los parámetros del círculo. Posteriormente, se emplean el resto de puntos del contorno para calcular el error cuadrático medio con respecto al círculo estimado, y si éste es inferior a un determinado umbral, el blob es considerado como un círculo válido.

El trabajo desarrollado en [Sandoval00] también se centra en la detección de círculos, pero en este caso mediante la implementación de un detector de bordes dependiente del ángulo. De esta forma, la salida sólo detectará objetos en aquellas partes de la imagen donde realmente aparezcan círculos, anulando la salida para el resto de la imagen.

Los algoritmos genéticos también han sido ampliamente utilizados en la detección de formas. Los trabajos descritos en [Liu02] y [Aoyagi96] son algunos ejemplos del uso de algoritmos genéticos para la localización de círculos, pero ninguno de ellos funciona correctamente en presencia de oclusiones en el objeto.

Un ejemplo de detección de rectángulos puede encontrarse en [Suau05]. En este caso, se realiza la proyección horizontal y vertical de los bordes de los objetos, y se buscan los máximos en dichas proyecciones. El principal problema de este método es que no funcionará correctamente en el caso de que el objeto rectangular se encuentre rotado.

En [Gavrila99] se amplia la búsqueda a triángulos y círculos. En este caso, se emplean plantillas predefinidas en distintas escalas de las formas a localizar. El principal problema es que, si el número de plantillas consideradas es elevada y la zona de búsqueda grande, el proceso puede ralentizarse considerablemente. En [Vitabile01] también se emplean plantillas predefinidas, pero en este caso también se incluyen versiones rotadas, para conseguir que el algoritmo sea invariante a la rotación.

En [Escalera94], se analiza el número y tipo de cada uno de los vértices del contorno del objeto. Esto permite la identificación de círculos, triángulos y rectángulos.

4.2. Sistema implementado

Aunque para un sistema de videovigilancia típico la cantidad de formas distintas que pueden aparecer en la imagen y ser detectadas, por ejemplo por el módulo de detección de movimiento, son infinitas, en este capítulo vamos a centrarnos en la clasificación y reconocimiento de formas sencillas. La principal utilidad de esta propuesta es la detección de objetos, generalmente artificiales, como pueden ser matrículas de coches, señales de tráfico, o cualquier otro objeto que presente una forma simple. Además, en numerosas ocasiones suele ser necesario no solo detectar la forma del objeto, sino también reconocer la información en su interior. Para optimizar el funcionamiento de bloques posteriores que realicen el reconocimiento o la identificación del contenido del objeto, sería interesante obtener una nueva imagen del objeto, deshaciendo las posibles distorsiones geométricas que haya sufrido debido a la proyección en el plano imagen de la cámara, como rotaciones, escalados, etc.

Para alcanzar los objetivos planteados, el algoritmo implementado se ha dividido en dos partes. Una primera que realiza la clasificación de la forma del objeto dentro de un grupo de formas determinadas. Las formas consideradas en principio en esta tesis serán el triángulo, el rectángulo, el círculo y el semicírculo, aunque no sería demasiado complejo la inclusión de alguna otra forma simple si fuese necesario. La segunda parte del algoritmo

permite realizar la reorientación, con la idea de obtener una mejor representación del objeto, para que los siguientes bloques puedan realizar el reconocimiento del interior.

4.3. Clasificación de objetos a partir de la firma

El algoritmo propuesto realiza la clasificación de la forma del objeto comparando la firma de cada blob obtenido en el proceso de detección, con las firmas de las formas de referencia. En esta tesis, vamos a definir la firma como la distancia desde el centro de masas del objeto hasta el borde de éste en función del ángulo [Gonzalez93]. En las figuras 4.3, 4.4, 4.5 y 4.6 se muestran las formas de referencia, junto con su firma correspondiente. Aunque la firma de cada nuevo blob podría ser calculada directamente, si tenemos en cuenta ciertas consideraciones respecto a posibles distorsiones geométricas que los objetos pueden sufrir, podemos mejorar el resultado del sistema realizando un preprocesado a los blob detectados antes de la obtención de la firma. Las distorsiones geométricas consideradas son las siguientes:

- **Distorsiones de proyección:** En objetos planos, las distorsiones de proyección aparecen cuando el plano imagen de la cámara no es paralelo al plano que contiene el objeto. Bajo estas circunstancias, por ejemplo un círculo será visualizado como una elipse, un triángulo equilátero como uno escaleno, o un rectángulo como un cuadrilátero.

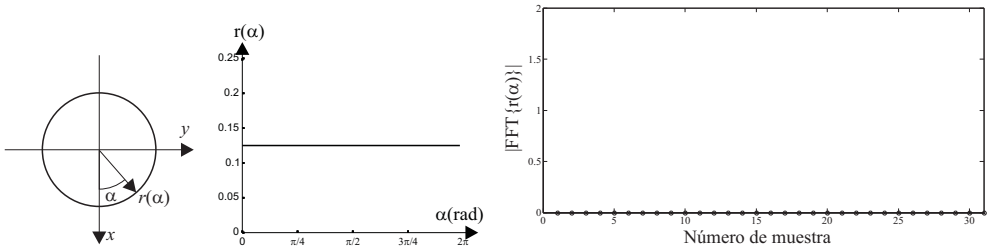
- **Escalado:** El tamaño del objeto proyectado sobre el plano imagen depende de la distancia existente entre la cámara y el objeto. Según la cámara se aproxima al objeto, su proyección se hace más grande, y por tanto, las firmas obtenidas para el mismo objeto grabado a distintas distancias tendrán distintas amplitudes.

- **Rotaciones:** Diferencias en los ángulos de inclinación entre la cámara y el objeto se reflejan en la imagen como una rotación del objeto. Dichas rotaciones en la imagen se convierten en desplazamientos circulares en la firma.

- **Oclusiones:** Las oclusiones parciales en los objetos pueden alterar su forma. Otros tipos de oclusiones pueden dividir el objeto segmentado en dos o más partes, o incluso ocultarlo completamente.

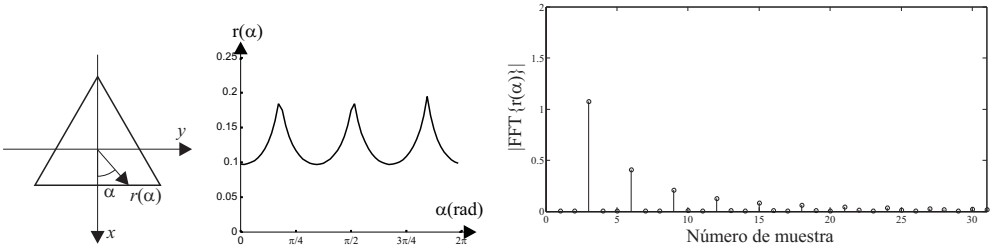
- **Ruido de la cámara:** El ruido introducido por la cámara puede provocar que la máscara binaria obtenida en la segmentación no tenga un contorno perfectamente definido, sino alterado por el ruido.

Con el objetivo de mejorar el resultado del módulo de clasificación de formas, el algoritmo propuesto en esta tesis no obtiene la firma directamente del propio blob, sino de una versión modificada que trata de evitar, en la medida de lo posible, todas las distorsiones geométricas consideradas anteriormente. En las siguientes secciones se describe la estrategia adoptada para manejar cada uno de los problemas anteriores.



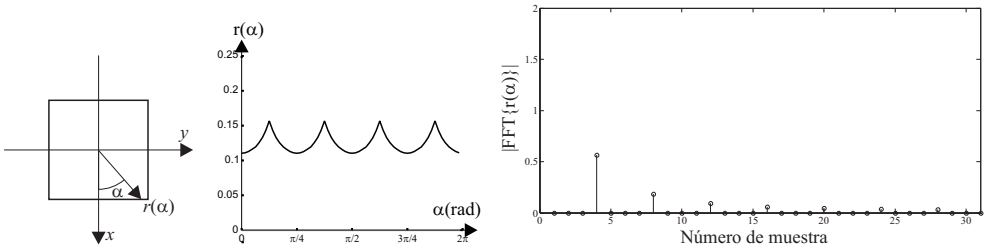
(a) Figura (b) Firma (c) FFT

Figura 4.3: Firma y módulo de la FFT para un círculo



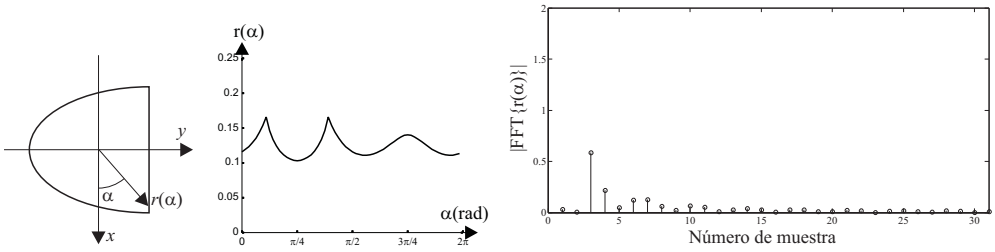
(a) Figura (b) Firma (c) FFT

Figura 4.4: Firma y módulo de la FFT para un triángulo equilátero



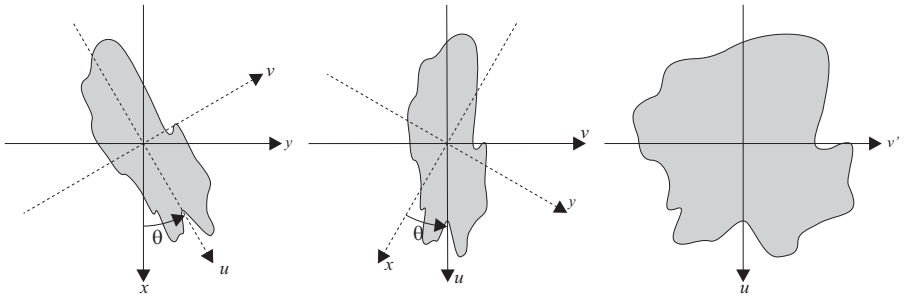
(a) Figura (b) Firma (c) FFT

Figura 4.5: Firma y módulo de la FFT para un cuadrado



(a) Figura (b) Firma (c) FFT

Figura 4.6: Firma y módulo de la FFT para una semielipse



(a) (b) (c)

Figura 4.7: Cambio de sistema de coordenadas para corregir la distorsión de proyección.

4.3.1. Distorsiones de proyección

Las distorsiones de proyección ocurren siempre que un objeto es visualizado por una cámara genérica, y el plano imagen no se encuentra paralelo al plano que contiene dicho objeto. Aunque el escalado y las rotaciones también pueden ser consideradas como distorsiones de proyección, éstas últimas son transformaciones más específicas, y serán analizadas posteriormente. Por otro lado, aunque en este punto no es posible, pero tampoco necesario, deshacer todas las posibles distorsiones que ha sufrido el objeto, sin embargo si que sería interesante realizar alguna transformación geométrica del blob con la idea de mejorar el proceso de clasificación de formas.

El inconveniente principal de la distorsión de proyección para nuestro propósito de clasificar la forma de un objeto a partir de la firma del blob, es que se produce una deformación del objeto. Dicho de otra manera, la relación de aspecto, o excentricidad, del objeto varía, lo que llevará, si no se corrige dicha distorsión, a un error en la clasificación. Por tanto, para mejorar el funcionamiento del clasificador, se realiza previamente una normalización de la relación de aspecto para cada blob analizado. Para realizar dicha normalización, en primer lugar se calcula la orientación del objeto de acuerdo a [Jain89]:

$$\theta = \frac{1}{2}\arctan\left(\frac{2\cdot\mu_{11}}{\mu_{20}-\mu_{02}}\right) \tag{4.8}$$

donde μ_{11}, μ_{20} y μ_{02} son los momentos centrales del objeto. Una vez que conocemos la orientación del objeto, creamos un nuevo sistema de coordenadas, siendo u la coordenada en la dirección de la orientación del objeto, y v la coordenada ortogonal a la coordenada u, como se muestra en la figura 4.7(a). A partir de ahora, llamaremos μ^{uv} al momento en el espacio uv, y lo mismo para los demás espacios. Para pasar de un espacio a otro, será necesaria la matriz de transformación siguiente:

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} \tag{4.9}$$

En este nuevo sistema de coordenadas (figura 4.7(b)) los momentos de segundo orden μ_{20}^{uv} and μ_{02}^{uv} pueden ser calculados a partir de los momentos en el espacio xy empleando las siguientes expresiones:

$$\mu_{20}^{uv} = m_{20}^{uv} - m_{00} \cdot (\mu_{10}^{uv})^2 \tag{4.10}$$

$$\mu_{02}^{uv} = m_{02}^{uv} - m_{00} \cdot (\mu_{01}^{uv})^2 \tag{4.11}$$

donde

$$m_{20}^{uv} = m_{20}^{xy} \cdot \cos^2\theta + m_{02}^{xy} \cdot \sin^2\theta + 2 \cdot m_{11}^{xy} \cdot \cos\theta \cdot \sin\theta \tag{4.12}$$

$$m_{02}^{uv} = m_{20}^{xy} \cdot \sin^2\theta + m_{02}^{xy} \cdot \cos^2\theta - 2 \cdot m_{11}^{xy} \cdot \cos\theta \cdot \sin\theta \tag{4.13}$$

y

$$\mu_{10}^{uv} = \mu_{10}^{xy} \cdot \cos\theta + \mu_{01}^{xy} \cdot \sin\theta \tag{4.14}$$

$$\mu_{01}^{uv} = -\mu_{10}^{xy} \cdot \sin\theta + \mu_{01}^{xy} \cdot \cos\theta \tag{4.15}$$

Con los valores anteriores, podemos definir la excentricidad del objeto como la relación entre los momentos μ_{20}^{uv} y μ_{02}^{uv} de acuerdo a:

$$k = \sqrt{\frac{\mu_{02}^{uv}}{\mu_{20}^{uv}}} \tag{4.16}$$

Esta expresión siempre devolverá un valor menor de 1, ya que queda impuesto con el cálculo de la orientación del objeto mediante (4.8). La normalización comentada al inicio de la sección puede conseguirse simplemente *estirando* del objeto $1/k$ veces en el eje v, como se muestra en la figura 4.7(c). Esta normalización puede realizarse utilizando la siguiente matriz de transformación:

$$\begin{pmatrix} u' \\ v' \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ -k\sin\theta & k\cos\theta \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} \tag{4.17}$$

La firma calculada ahora pertenecerá a un objeto con una excentricidad, tal y como fue definida anteriormente en (4.16), igual a 1, y por lo tanto, el mismo valor necesita ser impuesto a las formas de referencia. Puede comprobarse fácilmente que dicha excentricidad es siempre 1 para las tres primeras formas mostradas en las figuras 4.3, 4.4 y 4.5, es decir, la relación entre los momentos de segundo orden en el eje x y el y es siempre 1 para los círculos, los triángulos equiláteros y los cuadrados. Sin embargo, esto no es cierto para los semicírculos, de tal forma que la forma de referencia debe ser escalada para construir una semielipse que posea una relación entre sus momentos de orden 2 igual a 1. En la figura 4.6 se muestra el aspecto de dicha semielipse.

Es importante resaltar que esta transformación sólo se realiza para mejorar los resultados en el módulo de clasificación de formas del objeto analizado a partir de la firma de

éste. La reorientación y normalización del objeto no es función de esta parte del algoritmo, ya que este proceso se realiza una vez que la forma del objeto ya ha sido determinada. Además, el método para realizar dicha normalización es diferente para cada una de las distintas formas analizadas.

4.3.2. Escalado

El escalado de un objeto se traduce en la amplificación o atenuación de la firma de dicho objeto. Para hacer el algoritmo invariante al escalado, se realiza la normalización en energía de la firma obtenida antes de realizar la clasificación del objeto. Obviamente, las firmas de todos las formas de referencia, que ya están previamente calculadas y almacenadas, también deben haber sido normalizadas en energía, como puede verse en las figuras 4.3, 4.4, 4.5 y 4.6.

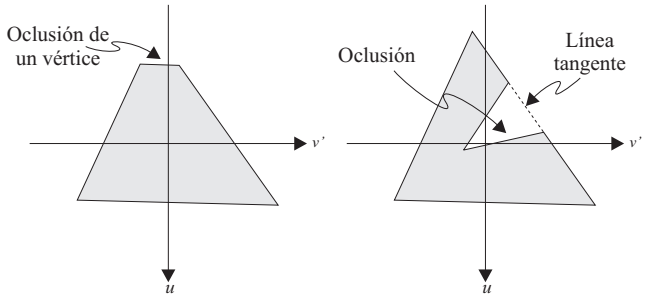
4.3.3. Rotaciones

Las rotaciones de los objetos se traducen en desplazamientos circulares en la firma. Para hacer el algoritmo de clasificación invariante a las rotaciones, en lugar de comparar directamente la propia firma, la comparación se realiza a partir del valor absoluto de la transformada discreta de Fourier (DFT, *Discrete Fourier Transform*), aprovechando la propiedad de invarianza al desplazamiento del módulo de la DFT. Por lo tanto, el valor absoluto de la DFT de firma normalizada de cada una de las formas de referencia es lo que realmente se almacena en lugar de la propia firma, y lógicamente, para cada nuevo objeto a analizar, la DFT de la firma normalizada del objeto es calculada y empleada para su clasificación.

4.3.4. Oclusiones

Las oclusiones son uno de los grandes problemas que se pueden presentar en las tareas de reconocimiento y clasificación de formas. Si la oclusión es significativa, por ejemplo, si el objeto ha quedado dividido en dos o más partes, lo más probable es que el algoritmo de clasificación falle, a no ser que se añada algún algoritmo especial para tratar con este tipo de oclusiones. Sin embargo, si la oclusión no es muy grande, el funcionamiento del bloque puede ser mejorado con ligeras modificaciones sobre el algoritmo original. En el caso que nos ocupa en esta tesis, donde estamos tratando con formas simples, podemos distinguir entre dos tipos distintos de oclusiones, especialmente para triángulos y rectángulos. Es decir, tenemos que diferenciar entre aquellas oclusiones que hacen desaparecer una de las esquinas de la figura, como se puede ver en la figura 4.8(a), y aquellas en las que la oclusión afecta a la parte recta, como se muestra en la figura 4.8(b).

Mientras que el primer tipo de oclusiones es más difícil de tratar, las segundas son bastante más sencillas. Aprovechando que todas las figuras consideradas son convexas, podremos eliminar un gran número de oclusiones simplemente eliminando todas las partes



(a) (b)

Figura 4.8: Ejemplos de oclusión.

cóncavas del objeto trazando líneas tangentes al contorno del objeto en esos puntos. Por ejemplo, en la figura 4.8(b) se puede comprobar como la oclusión ha sido eliminada, y el resultado que tenemos es el mismo que tendríamos si no hubiese habido oclusión.

Para los casos en los que una de las esquinas (o más de una) del objeto ha desaparecido como consecuencia de una oclusión, el esquema anterior no resuelve el problema, ya que la figura resultante sigue siendo convexa. En este caso, mientras que la oclusión no sea demasiado grande, la clasificación probablemente sea correcta. Si la oclusión es demasiado grande, los resultados son impredecibles.

Para el caso de los círculos, dependiendo del grado de la oclusión, el objeto analizado puede llegar a convertirse en un semicírculo, causando el error en la clasificación. Dependiendo de la aplicación en concreto, esto puede ser considerado como un error en el sistema. Sin embargo, tal y como se verá posteriormente, en nuestro caso, cada vez que se detecta un semicírculo, se procede a la detección del círculo que teóricamente originó dicho semicírculo, por que en ciertas ocasiones, este error de clasificación no provocará un error en el funcionamiento global del sistema.

4.3.5. Clasificación mediante FFT

Según lo visto hasta ahora, el vector que emplearemos para realizar la clasificación del objeto no es la firma directamente, sino el valor absoluto de la DFT de la firma normalizada, que ha sido obtenida de un objeto derivado del objeto inicial. El tiempo de cálculo puede reducirse si ajustamos el número de muestras de la firma a una potencia de 2, de tal forma que se puede realizar el cálculo mediante la FFT (*Fast Fourier Transform*) en lugar de la DFT. En esta tesis, hemos elegido dicho número igual a 64 (2^6). Este valor es un compromiso entre precisión y coste computacional. En las figuras 4.3, 4.4, 4.5 y 4.6 se muestra, para cada forma de referencia, el valor absoluto de la primera mitad de su FFT correspondiente. Con la única idea de hacer dichas figuras más claras, y teniendo en cuenta que para todas las formas, la muestra en 0 es bastante más grande que el resto

de las muestras, y además muy similar para todas ellas (alrededor de $\sqrt{64} = 8$ para las firmas consideradas), dicha muestra ha sido eliminada de las figuras.

Aunque existen un gran número de algoritmos para realizar la clasificación, un clasificador tan sencillo como la clasificación por el vecino más cercano (NN, *Nearest Neighbour*) ha resultado ser suficientemente preciso para el propósito de clasificación de formas. Por otro lado, como se puede ver en las figuras 4.3, 4.4, 4.5 y 4.6, sólo aproximadamente un cuarto de las muestras de la FFT son significativas, mientras que el valor del resto de las muestras es muy pequeño, y su valor real va a depender más del ruido que de la forma real. Para la muestra en cero también se puede decir algo similar, ya que el valor de ésta es muy similar para las formas consideradas, y por tanto no aporta ninguna información. Como resumen de la implementación del clasificador de formas, hay que indicar que la clasificación se realiza calculando la distancia euclídea empleando sólo las muestras 1 a 8 de la FFT de 64 muestras de la firma del objeto, y eligiendo aquel cuya distancia sea la menor entre las cuatro clases consideradas.

4.4. Localización de la figura

Aparte de la clasificación de la forma del objeto analizado, a veces también es interesante reorientarlo para poder analizar su contenido más fácilmente. En este caso el sistema, una vez reconocida la forma del objeto, deberá normalizar la posición el objeto deshaciendo cualquier posible rotación, traslación, escalado y deformación proyectiva para poder situar el objeto en una posición de referencia. En este punto vamos a aprovechar el hecho de que ya conocemos la forma del objeto, de tal forma que el algoritmo de normalización puede ser diseñado de forma independiente para cada una de las distintas formas.

Independientemente de la forma del objeto, la imagen de dicho objeto puede considerarse como una homografía en dos dimensiones, transformando puntos del plano que contiene el objeto, en otro plano, en este caso, el plano imagen [Hartley03]. Teniendo esto en cuenta, el problema de normalización consiste en el cálculo de una homografía, es decir, una matriz no singular de tamaño 3×3. Sin embargo, este cálculo depende de la forma del objeto en cuestión. Por ejemplo, para triángulos y rectángulos, la matriz puede ser calculada a partir de las coordenadas de los vértices de la figura, estableciendo correspondencias entre esos puntos y los vértices de la figura de referencia. Para los círculos y semicírculos, este cálculo se hará estimando los parámetros de la elipse correspondiente. Debido a que, básicamente lo que debemos realizar es estimar la localización exacta de la figura en la imagen, a partir de ahora denominaremos esta tarea como *localización* del objeto. En las siguientes secciones se describirá el algoritmo específico diseñado para localizar, y posteriormente normalizar, cada una de las formas consideradas.

4.4.1. Normalización de triángulos

Una homografía en dos dimensiones se representa mediante una matriz no singular de tamaño 3×3, con 8 grados de libertad (DOF, *Degree of Freedom*), que serían los 9 coeficientes de una matriz no singular menos uno de escala. Una forma común de calcular dicha homografía es mediante el algoritmo DLT (*Direct Linear Transformation*) [Hartley03]. El algoritmo necesita un conjunto de correspondencias entre puntos, y el número mínimo de dichas correspondencias depende del tipo de homografía a calcular.

En el caso de una homografía genérica de dos dimensiones, se tienen 8 DOFs. Como cada correspondencia de un punto en ambos planos supone dos restricciones, sería necesario un mínimo de 4 correspondencias entre puntos de ambos planos. Sin embargo, si realizamos ciertas suposiciones al proceso de grabación de la imagen, es posible reducir el número de DOFs en el sistema.

Aunque el caso genérico de una cámara de vídeo grabando un plano puede describirse como una transformación proyectiva, que supondría los 8 DOFs mencionado arriba, en muchas ocasiones el sistema de grabación empleado en videovigilancia puede aproximarse sin mucho error a una transformación afín, lo que reduciría el número de DOFs a sólo 6. De esta forma, el número de correspondencias de puntos bajaría hasta 3. En teoría, la simplificación a una transformación afín ocurre cuando la cámara que está visualizando el plano se encuentra en el infinito. Por tanto, la transformación afín puede suponerse siempre y cuando la cámara se encuentre en el infinito, o al menos a una distancia considerablemente más grande que el tamaño del objeto visualizado, lo que es, en la mayoría de las ocasiones, lo normal en los sistemas de videovigilancia. Por tanto, la matriz `H` se convierte en:

$$\mathtt{H}_A = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ 0 & 0 & 1 \end{bmatrix} \tag{4.18}$$

Para el caso de los triángulos, esa correspondencia entre puntos puede obtenerse directamente de los vértices del triángulo, y por tanto, el problema se reduce a la identificación de esos tres puntos en la imagen original.

Como se puede ver en la figura 4.4, los vértices del triángulo pueden localizarse en la firma calculada anteriormente para el proceso de clasificación de formas, ya que los tres vértices se corresponden con los tres picos de la firma. Aunque este método sería una buena solución en el caso teórico, cuando se aplique a imágenes reales la solución sería bastante imprecisa, ya que estaríamos obteniendo un resultado basándonos únicamente en la información de un único píxel. Además, si una oclusión hace desaparecer uno de los vértices del triángulo, el sistema estimaría las coordenadas del vértice en una posición distinta a su posición real. Por último, también tenemos que tener en cuenta que el sistema no tiene precisión sub-píxel, ya que la salida del algoritmo es justamente la posición de los tres píxeles que se corresponden con los picos de la firma.

Los problemas anteriores pueden evitarse si consideramos más de un píxel para estimar las coordenadas de los vértices, en lugar de utilizar únicamente el píxel correspondiente al pico de la firma. Para ello, en primer lugar se estimarán los parámetros de las tres rectas que componen el triángulo, y a partir de los puntos de cruce de éstas se obtendrán los vértices del triángulo. La estimación de cada recta puede hacerse a partir de varios píxeles pertenecientes al contorno del objeto. Aprovechando que la firma está compuesta por píxeles pertenecientes al contorno del objeto, podemos utilizar dichos píxeles para estimar las rectas del triángulo. Suponiendo que los tres picos de la firma ya han sido localizados, todos los píxeles situados entre dos picos componen una recta. Utilizando coordenadas homogéneas, un punto $\mathbf{x_i} = (x_i, y_i, 1)^\mathtt{T}$ pertenece a una línea recta $\mathbf{l} = (l_a, l_b, l_c)^\mathtt{T}$ si:

$$\left(\begin{array}{ccc} x_i, & y_i, & 1 \end{array} \right) \left(\begin{array}{ccc} l_a, & l_b, & l_c \end{array} \right)^\mathtt{T} = 0 \, . \tag{4.19}$$

Si obtenemos la ecuación anterior para cada uno de los puntos disponibles de la firma para una recta en concreto, tenemos:

$$\mathtt{X} \cdot \mathbf{l} = \begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots \\ x_N & y_N & 1 \end{pmatrix} \left(\begin{array}{ccc} l_a & l_b & l_c \end{array} \right)^\mathtt{T} = \mathbf{0} \, , \tag{4.20}$$

lo que implica el cálculo del espacio nulo de la matriz $\mathtt{X}$. Ya que el número de filas de $\mathtt{X}$, que se corresponde con el número de puntos disponibles de la firma para una determinada recta, será, en general, mayor de dos, y los puntos estarán afectados por el ruido, (4.20) define un sistema sobredeterminado.

Al ser un sistema sobredeterminado, deberá ser resuelto utilizando alguna técnica de minimización. Aunque lo recomendable sería utilizar alguna técnica iterativa de minimización geométrica para obtener un resultado preciso, una simple minimización algebraica es suficiente para nuestro propósito, y de esta forma, aprovechamos su menor complejidad computacional para acelerar el proceso. Esto es posible ya que la minimización geométrica ofrece mejores resultados en presencia de *outliers*, es decir, puntos falsos, en este caso, puntos que no pertenecen a la recta, o bien cuando los datos están afectados fuertemente por el ruido. En nuestro caso, esto no se produce, es decir, no existen puntos falsos ni el ruido es elevado, ya que, si esto fuese cierto, la clasificación de la forma hubiese fallado, y el proceso siguiente sería incorrecto en cualquier caso.

Una vez obtenidos los parámetros de la recta, podemos estimar el grado de confianza de la recta. Si calculamos el error geométrico como la media de la suma de las distancias desde cada punto de la firma a la recta estimada mediante:

$$e = \frac{1}{N} \sum_{i=1}^{N} \left| \frac{l_a x_i + l_b y_i + l_c}{\sqrt{l_a^2 + l_b^2}} \right| , \tag{4.21}$$

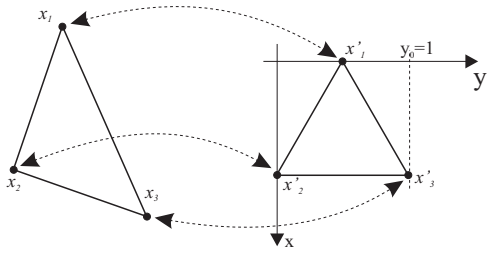


Figura 4.9: Correspondencias entre puntos en triángulos

podemos determinar hasta que grado la lista de puntos de la firma empleados para estimar la recta forman una recta. Ajustando un umbral para el error geométrico de cada recta, si el error de alguna de las rectas supera dicho umbral, podremos tratar ese objeto como una falsa alarma. Es decir, en el bloque de clasificación de formas, se clasificó dicho objeto como un triángulo pero, en realidad, no se trata de un triángulo, ya que alguna de sus rectas no es realmente una recta, de acuerdo al umbral impuesto.

Una vez que disponemos de los parámetros de las tres rectas que componen el triángulo, podemos calcular los vértices de acuerdo a:

$$\mathbf{x}_{ij} = \mathbf{l}_i \times \mathbf{l}_j = \begin{vmatrix} \mathrm{a} & \mathrm{b} & \mathrm{c} \\ l_{ia} & l_{ib} & l_{ic} \\ l_{ja} & l_{jb} & l_{jc} \end{vmatrix} = \begin{pmatrix} l_{ib}l_{jc} - l_{ic}l_{jb} \\ l_{ic}l_{ja} - l_{ia}l_{jc} \\ l_{ia}l_{jb} - l_{ib}l_{ja} \end{pmatrix} \tag{4.22}$$

donde $(i, j) = \{(1, 2), (2, 3), (3, 1)\}$, de tal forma que obtenemos las coordenadas de los tres vértices que componen el triángulo.

Una vez que tenemos las coordenadas de los tres vértices, el siguiente paso sería hacer las correspondencias entre los puntos de la imagen original y el triángulo de referencia, como se muestra en la figura 4.9, donde la referencia es un triángulo equilátero, siendo la longitud de los lados igual a 1. En este punto debemos tener en cuenta una posible restricción. Tal y como puede verse en la figura 4.9, cada vértice del triángulo original debe ser relacionado con su correspondiente vértice en el triángulo de referencia, y si no existiera ninguna otra restricción, tendríamos tres posibilidades. Si en la aplicación en la cual se estuviera empleando este algoritmo fuese posible suponer que ninguna figura sufrirá una rotación mayor de 60 grados, entonces el vértice superior del triángulo original siempre se correspondería con el vértice superior del triángulo de referencia.

Si la anterior suposición no fuese posible, y fuese necesario disponer de una imagen normalizada del objeto con su orientación real, la solución pasaría por obtener tres imágenes normalizadas distintas del objeto: una de ellas la imagen obtenida directamente con la homografía calculada por el algoritmo, y las otras dos serían versión rotadas de la primera, la primera a +120 grados, y la segunda a −120 grados.

Con estas suposiciones, el sistema dispondría de tres correspondencias entre puntos $\mathbf{x_i} \leftrightarrow \mathbf{x'_i}$ $(i = 1, 2, 3)$. Escribiendo $\mathbf{x_i} = (x_i, y_i, 1)^\intercal$ y $\mathbf{x'_i} = (x'_i, y'_i, 1)^\intercal$, el algoritmo DLT

para una transformación afín genera el siguiente sistema de ecuaciones lineales para cada correspondencia de puntos entre ambos planos [Hartley03]:

$$\begin{bmatrix} 0 & 0 & 0 & -x' & -y' & -1 \\ x' & y' & 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{pmatrix} \mathbf{h^1} \\ \mathbf{h^2} \end{pmatrix} = \begin{pmatrix} -y \\ x \end{pmatrix} \tag{4.23}$$

donde $\mathbf{h^1} = (h_{11}, h_{12}, h_{13})^\mathtt{T}$ y $\mathbf{h^2} = (h_{21}, h_{22}, h_{23})^\mathtt{T}$ son las dos primeras filas de la matriz afín $\mathtt{H}_A$ dispuestas para formar un vector de 6 elementos. La ecuación puede escribirse de forma compacta como:

$$\mathtt{X}\mathbf{h} = \mathbf{y} \tag{4.24}$$

Disponiendo las seis ecuaciones obtenidas a partir de la tres correspondencias, obtenemos un sistema de ecuaciones lineales de seis incógnitas y seis ecuaciones, que puede resolverse para $\mathbf{h}$ utilizando técnicas estándar de resolución de ecuaciones lineales.

4.4.2. Normalización de rectángulos

El cálculo de la homografía de reorientación para rectángulos puede derivarse del algoritmo desarrollado para los triángulos mediante ligeras modificaciones. En el caso de los rectángulos, existen cuatro puntos con los que hacer las correspondencias, que nos permitiría calcular la homografía sin necesidad de realizar la simplificación a una transformación afín. Sin embargo, aquí también se puede realizar la aproximación a una transformación afín, lo que nos va a permitir ahorrar tiempo en el cálculo. Por tanto, al tener 4 correspondencias, lo que equivale a ocho ecuaciones, y sólo 6 incógnitas, es decir, los seis DOFs de una transformación afín, tenemos un sistema de ecuaciones lineal sobredeterminado, que puede ser resuelto minimizando el error cuadrático medio mediante las ecuaciones normales:

$$\left(\mathtt{X}^\mathtt{T}\mathtt{X}\right)\mathbf{h} = \mathtt{X}^\mathtt{T}\mathbf{y} \tag{4.25}$$

Los cuatro vértices del rectángulo son localizados de la misma forma que los del triángulo vista anteriormente, a partir de la firma previamente calculada, y las correspondencias entre puntos realizadas como se muestra en la figura 4.10. De nuevo, aquí se presenta el mismo problema que antes, es decir, que para un punto dado, es posible establecer cuatro correspondencias distintas, y por tanto, se deberá realizar una restricción similar. En el caso del rectángulo es suficiente con suponer que el objeto no sufrirá una rotación mayor de 45 grados para asegurarnos de que la imagen del rectángulo transformado con la homografía devuelta con el algoritmo desarrollado está situado en su orientación correcta. En estas condiciones, se puede asegurar por ejemplo que el vértice inferior derecho del rectángulo de referencia corresponderá siempre con el primer pico de la firma, es decir, el primer vértice del objeto que nos encontraríamos si empezáramos en el eje x avanzando en sentido negativo. Si no fuese posible el asegurar que el objeto no sufra una rotación

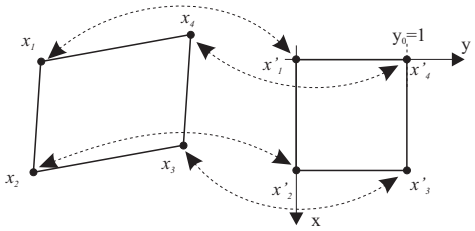


Figura 4.10: Correspondencias entre puntos en rectángulos

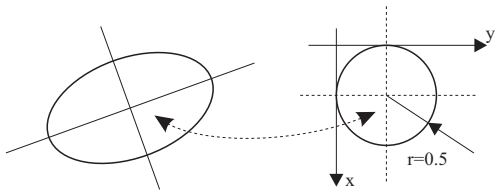


Figura 4.11: Correspondencias en círculos

superior a 45 grados, tendremos que buscar una solución similar a la diseñada para los triángulos. En este caso, serían necesarias cuatro imágenes distintas, siendo las tres últimas las correspondientes a las versiones rotadas a 90, 180 y 270 grados respecto de la primera.

4.4.3. Normalización de círculos

El principal problema que se presenta en la normalización de los círculos, con respecto a las normalizaciones vistas hasta ahora, es que el círculo no tiene vértices, y por tanto, no podemos utilizar los puntos como referencia para el cálculo de la homografía. Por ello, no es posible establecer correspondencias entre puntos, y la homografía deberá ser calculada de otra forma. Teniendo en cuenta que una transformación afín puede transformar un círculo en una elipse, sólo necesitaríamos estimar los parámetros de la elipse analizada para poder calcular la correspondiente homografía. Hay que tener en cuenta que una elipse se define con cinco parámetros, que serían el centro, que contaría como dos parámetros, los ejes mayor y menor y la orientación del eje principal, mientras que una homografía afín tiene seis DOFs. El último DOF se correspondería con la orientación del círculo, que no puede ser calculada debido a la falta de puntos de referencia, y por tanto, la homografía sólo puede ser calculada hasta una rotación alrededor de su centro.

Aunque existen distintos trabajos que permiten ajustar puntos a una elipse [Gander94], estos algoritmos se centran en el cálculo de los parámetros de la elipse. En esta tesis, sin embargo, proponemos un método que calcula directamente la homografía necesaria para reorientar el objeto sin ningún tipo de cálculo adicional. Esta transformación mapea la elipse original en la figura de referencia, en este caso un círculo de diametro 1, y centrado en $(0{,}5; 0{,}5)$, como se muestra en la figura 4.11.

Teniendo en cuenta que sólo consideraremos los píxeles del contorno o, por simplificar el algoritmo, sólo los píxeles de la firma, se puede considerar la elipse como una cónica [Hartley03], que puede ser descrita por una matriz simétrica de la forma:

$$\mathtt{C} = \begin{bmatrix} a & b/2 & d/2 \\ b/2 & c & e/2 \\ d/2 & e/2 & f \end{bmatrix} , \tag{4.26}$$

con cinco grados de libertad, que serían los seis coeficientes distintos de una matriz simétrica menos uno de escala. Por lo tanto, será necesario un mínimo de cinco puntos para definir una elipse. Cualquier punto perteneciente a la elipse debe cumplir:

$$\mathbf{x}_i^{\mathtt{T}} \mathtt{C} \mathbf{x}_i = 0 , \tag{4.27}$$

o equivalentemente:

$$\begin{pmatrix} x_i^2, & x_i y_i, & y_i^2, & x_i, & y_i, & 1 \end{pmatrix} \mathbf{c} = 0 , \tag{4.28}$$

donde $\mathbf{c} = (a, b, c, d, e, f)^{\mathtt{T}}$ es la cónica $\mathtt{C}$ representada como un vector de seis elementos. A partir de la ecuación anterior obtenida para todos los puntos disponibles de la firma del objeto, obtenemos el siguiente sistema de ecuaciones:

$$\begin{pmatrix} x_1^2 & x_1 y_1 & y_1^2 & x_1 & y_1 & 1 \\ x_2^2 & x_2 y_2 & y_2^2 & x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_N^2 & x_N y_N & y_N^2 & x_N & y_N & 1 \end{pmatrix} \mathbf{c} = \mathbf{0} . \tag{4.29}$$

Si el número de puntos en mayor de cinco (4.29) supone el cálculo del espacio nulo de un sistema sobredeterminado, que puede ser resuelto fácilmente mediante técnicas de minimización algebraica. Una vez que la cónica $\mathtt{C}$ ha sido calculada, el siguiente paso consiste en el cálculo de la homografía de dos dimensiones que transforme la elipse en un círculo auxiliar de radio 0,5 centrado en el origen de coordenadas, de acuerdo a [Hartley03]:

$$\mathtt{C}' = \mathtt{H}^{-\mathtt{T}} \mathtt{C} \mathtt{H}^{-1} , \tag{4.30}$$

donde $\mathtt{C}' = \mathrm{diag}\,(4, 4, -1)$ es el círculo auxiliar representado como una matriz diagonal de 3×3 coeficientes, siendo los valores de su diagonal principal igual a 4, 4 y −1. Haciendo la transformación inversa, y descomponiendo $\mathtt{H}$ en un producto de matrices:

$$\mathtt{C} = \mathtt{H}_R^{\mathtt{T}} \mathtt{H}_P^{\mathtt{T}} \mathtt{H}_D \mathtt{C}' \mathtt{H}_D \mathtt{H}_P \mathtt{H}_R , \tag{4.31}$$

donde $\mathtt{H}_R$ es una matriz ortogonal. Además, $\mathtt{C} = \mathtt{H}_R^{\mathtt{T}} \mathtt{C}_D \mathtt{H}_R$, donde $\mathtt{C}_D$ es una matriz diagonal, puede verse como la descomposición en valores singulares (SVD *Singular Value Decomposition*) de $\mathtt{C}$, que puede calcularse utilizando algoritmos estándar de descomposición SVD. La matriz $\mathtt{H}_P$ es una matriz de permutación para asegurar que $\mathtt{C}_D = \mathtt{H}_P^{\mathtt{T}} \mathtt{C}_E \mathtt{H}_P$,

siendo $\mathtt{C}_E$ una matriz diagonal con entradas e_{11} y e_{22} positivas, y la entrada e_{33} negativa. Por último, $\mathtt{H}_D$ es una matriz diagonal con entradas:

$$d_{11} = \sqrt{e_{11}}/2; \quad d_{22} = \sqrt{e_{22}}/2; \quad d_{33} = \sqrt{-e_{33}}\ . \tag{4.32}$$

Finalmente la matriz $\mathtt{H} = \mathtt{H}_D\mathtt{H}_P\mathtt{H}_R$ es la matriz que transforma la elipse original $\mathtt{C}$ en un círculo $\mathtt{C}'$ de radio 0,5 centrado en el origen de coordenadas. En este punto, podemos medir el error geométrico como la suma de la distancia desde cada uno de los puntos de la firma hasta la elipse estimada según:

$$e = \frac{1}{N}\sum_{i=1}^{N} \left| d\left(\mathtt{H}\mathbf{p}_i, \mathbf{0}\right) - 0{,}5 \right| , \tag{4.33}$$

donde $d\left(\mathbf{x}_1, \mathbf{x}_2\right)$ es la distancia euclídea entre los puntos $\mathbf{x}_1$ y $\mathbf{x}_2$, $\mathbf{p}'_i = \mathtt{H}\mathbf{p}_i$ son los puntos originales de la firma del objeto trasladados al plano destino, $\mathbf{0}$ son las coordenadas del centro del círculo (0;0), y 0,5 es el radio del círculo. El error geométrico obtenido puede ser umbralizado para descartar falsas alarmas. Por último, como la figura de referencia que queremos obtener es un círculo de radio 0,5 centrado en $(0{,}5; 0{,}5)$, necesitamos una matriz de traslación para desplazar el círculo anterior su la posición final:

$$\mathtt{H}_t = \begin{pmatrix} \mathtt{I} & \mathbf{t} \\ \mathbf{0}^\mathtt{T} & 1 \end{pmatrix} \tag{4.34}$$

donde $\mathtt{I}$ es una matriz identidad de tamaño 2×2, y $\mathbf{t} = (0{,}5, 0{,}5)^\mathtt{T}$ es el vector de traslación.

4.4.4. Normalización de semicírculos

Si consideramos el semicírculo como un problema de segmentación en el cual el módulo de segmentación devuelve dos semicírculos en lugar de un único círculo, una forma de localizar el semicírculo es estimando la posición del círculo que originó dichos semicírculos. Por lo tanto, el problema de la localización de semicírculos se reduce al cálculo de la homografía que transforma el círculo, del cual derivó el semicírculo analizado, en el círculo de referencia, es decir, exactamente el mismo problema analizado en la sección anterior.

Aunque, igual que para el círculo, los puntos para estimar la homografía pueden ser extraídos de la firma, el problema ahora es que hay que determinar qué puntos de la firma pertenecen a la parte circular del semicírculo, y qué puntos pertenecen a la parte recta. Si observamos la figura 4.12, podemos comprobar como la firma de una semielipse tiene tres picos, dos de ellos pertenecen a los puntos de cruce de la elipse con la recta que genera la semielipse, y el otro corresponde a aproximadamente el punto central entre los dos puntos anteriores.

Basándonos en lo anterior, podemos diseñar un algoritmo similar al diseñado para los triángulos, y emplearlo para estimar las tres rectas que unen los puntos que corresponden con los tres picos de la firma. Si calculamos el error geométrico para cada línea de acuerdo

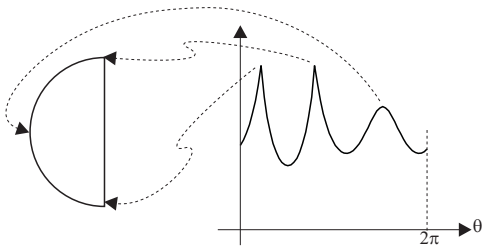


Figura 4.12: Correspondencias en semicírculos

a 4.21, deberíamos obtener una de ellas con un error pequeño, mientras que las otras dos deberían tener un error mucho más grande, lo cual nos permitirá determinar cual de las tres rectas pertenece realmente a la parte recta del semicírculo. Una vez determinado qué parte de la firma pertenece a la recta, y qué parte al círculo, solo nos queda construir un sistema de ecuaciones, similar al descrito en 4.29, empleando únicamente las muestras de la firma que corresponden a la parte circular. El resto del algoritmo en idéntico al descrito en la sección anterior para la localización de círculos.

Para el caso de semicírculos, podemos realizar dos evaluaciones para determinar hasta que grado la figura detectada se corresponde con un semicírculo. Por un lado, ya que tenemos calculado el error geométrico de cada una de las tres rectas, y sabemos cual de ellas es la que se corresponde con la parte recta del semicírculo, podemos umbralizar el error geométrico de dicha recta, de la misma forma que se hacía para triángulos y rectángulos. Por otro lado, igual que se hizo para el caso de los círculos, se puede también calcular el error geométrico de los puntos que se corresponden con la parte circular de la firma, y umbralizar este valor. Un semicírculo será considerado como una falsa alarma si cualquiera de las dos medidas anteriores es superior a su umbral correspondiente.

4.5. Evaluación del algoritmo implementado

En este apartado evaluaremos el funcionamiento del algoritmo implementado, tanto en su fiabilidad para realizar la clasificación de formas, como en su precisión para localizar y reorientar la figura una vez clasificada correctamente. Aunque el sistema puede trabajar con imágenes reales, hemos optado por testar el sistema con imágenes sintéticas generadas automáticamente. De esta forma, conseguiremos que los resultados sean independientes del bloque de segmentación, e incluso de las condiciones de iluminación o la calidad de la imagen capturada. Además, al ser las figuras generadas automáticamente, toda la información perteneciente a cada figura, como forma, posición exacta, tamaño, etc, puede ser también almacenada, lo que nos permitirá obtener los resultados buscados de forma automática.

Para ello, se ha desarrollado una aplicación adicional que es capaz de generar triángulos, rectángulos, elipses y semielipses de forma aleatoria, y guardar la información perteneciente a cada una de ellas es un archivo aparte. La imagen se genera en blanco y

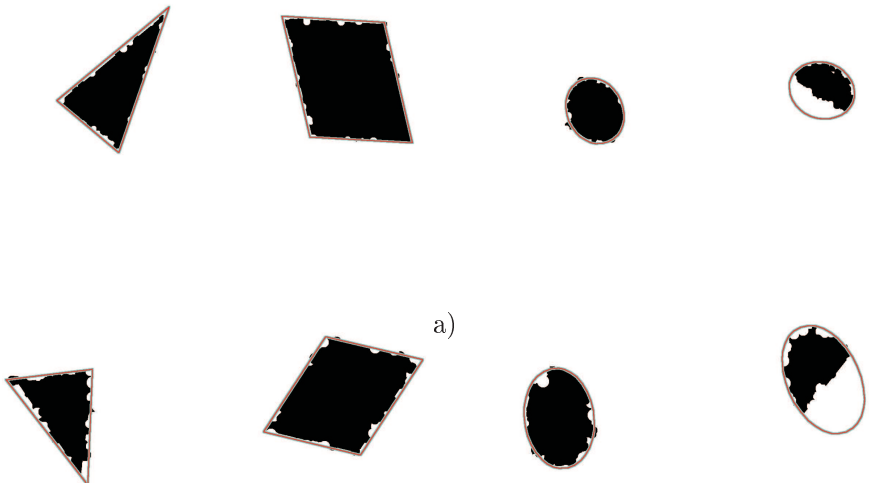
negro, por lo que no es necesario realizar ninguna segmentación adicional para ejecutar el algoritmo.

Para la construcción de triángulos, la aplicación genera tres coordenadas aleatorias y, tras validar ciertas restricciones geométricas, se genera un triángulo que tiene como vértices las tres coordenadas anteriores. Para rectángulos, el sistema es bastante similar. Sin embargo, hay que tener en cuenta que, aunque el sistema podría ser capaz de generar cualquier tipo de cuadrilátero, solamente son interesantes los cuadriláteros convexos. Además, como en la detección de rectángulos solamente se han tenido en cuenta transformaciones afines, únicamente se tendrán en cuenta los paralelogramos, por lo que sólo será necesario generar tres coordenadas. Para las elipses, serían necesarios cinco parámetros, dos para definir el centro de la elipse, uno para el tamaño del eje mayor, otro para el eje menor, y el último para la orientación del eje mayor. Para las semielipses es necesario un parámetro más, el que define la orientación de la recta que divide la elipse en dos mitades.

Para analizar el algoritmo en presencia de ruido, la aplicación desarrollada incorpora un paso más para realizar la adicción de ruido a la imagen. Para simular este efecto, en este paso la aplicación genera varias coordenadas aleatorias próximas a los bordes de la figura, y dibuja parches circulares de tamaño aleatorio, que alterarán el contorno de la figura. Como queremos evaluar el funcionamiento del algoritmo propuesto en presencia de distintos niveles de ruido, el diámetro de los parches es una variable aleatoria con distribución de probabilidad normal, y desviación típica σ configurable. De esta forma, podemos generar conjuntos de figuras con distinto nivel de ruido cada uno de ellos. Obviamente, cuanto mayor sea el valor de la desviación típica, mayores serán los parches, y mayor será la distorsión añadida a la figura original. La figura 4.13 muestra algunos ejemplos de figuras con ruido generadas con esta aplicación.

El efecto de las oclusiones parciales también será analizado aquí. Para ello, la aplicación anterior también es capaz de dibujar un parche, de tamaño mucho mayor que los empleados para el ruido, en los bordes de las figuras. Para triángulos y rectángulos, el parche es dibujado con su centro justo en uno de los vértices de la figura, ya que esto es el peor caso en el caso de una oclusión. Tal y como se vio en el apartado de manejo de las oclusiones, cuando la oclusión se produce en la parte recta de la figura, esto prácticamente no tiene efecto en el resultado final. Para las elipses, el parche es dibujado centrado en un punto perteneciente a la elipse, elegido de forma aleatoria. Para semielipses, el funcionamiento es el mismo, excepto en que el sistema debe asegurarse de que dicho punto pertenece a la parte no eliminada de la elipse original. Algunos ejemplos de figuras con oclusiones parciales pueden verse en la figura 4.14.

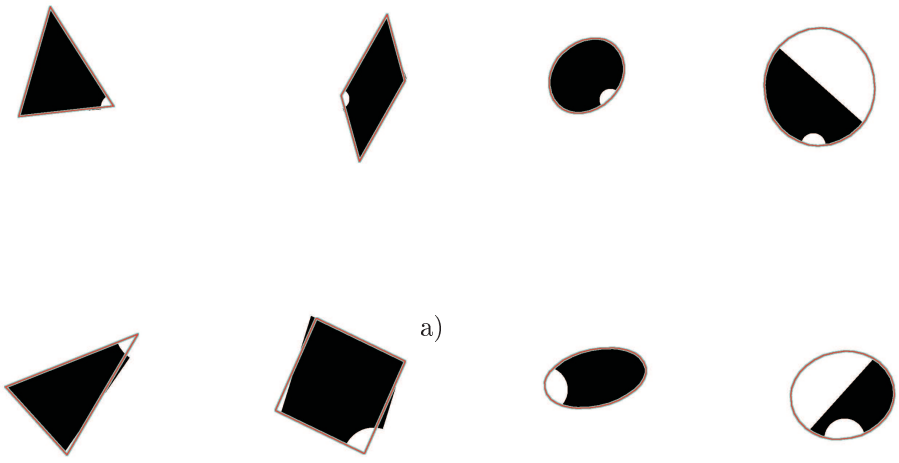
Utilizando esta herramienta adicional, vamos a evaluar el funcionamiento del algoritmo diseñado a partir de dos medidas. Por un lado, se obtendrá la probabilidad de acierto en la clasificación de la forma del objeto, medido como la relación entre el número de objetos correctamente clasificados y el total de objetos analizados. Por otro lado, también se medirá la precisión en la localización de la posición de la figura. Para ello, la figura con la posición estimada se dibuja sobre la figura original, sin ruido ni oclusión, y se contará el



a)

(b)

Figura 4.13: Ejemplo de aciertos en figuras ruidosas con σ igual a: (a) 6 píxeles; (b) 9 píxeles.



a)

(b)

Figura 4.14: Ejemplo de aciertos en figuras con oclusiones, con un porcentaje de oclusión igual a: (a) 10 %; (b) 20 %.

Tabla 4.1: Resultados numéricos:
Acc: Porcentaje de acierto en la clasificación de formas.
A. Err: Porcentaje de píxeles no coincidentes con respecto al área total en píxeles de la figura.

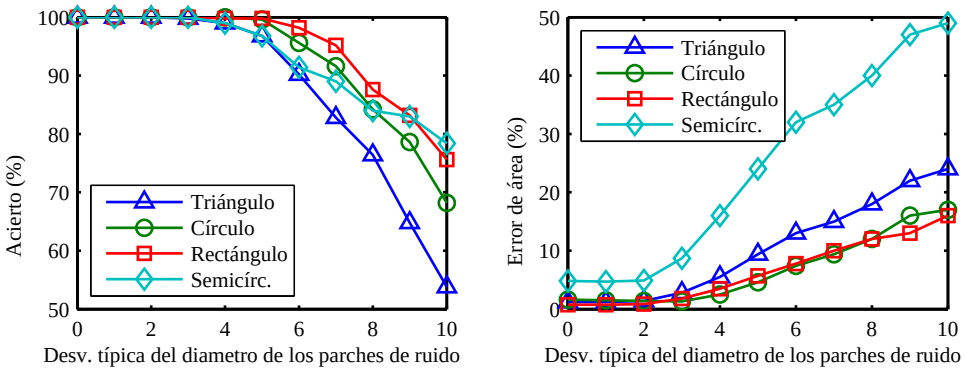
	σ=0		σ=5		σ=10	
	% Acc.	% A. Err.	% Acc.	% A. Err.	% Acc.	% A. Err.
Triángulo	100	1.20	96.80	9.40	53.80	24.00
Círculo	100	1.60	99.60	4.60	68.20	17.00
Rectángulo	100	0.74	99.80	5.70	75.60	16.00
Semicírculo	100	4.80	96.80	24.00	78.40	49.00

número de píxeles que coinciden en la figura original y la figura estimada. Así, la precisión en la localización se calcula como la relación entre el número de píxeles no coincidentes, es decir, píxeles que pertenecen a la figura original y no a la estimada, o viceversa, y el área total en píxeles de la figura original. Obviamente, esto sólo se hace para las figuras con su forma correctamente clasificadas, descartando para este segundo análisis las figuras incorrectas.

Para la obtención de resultados, se han generado conjuntos de 2000 figuras sintéticas, formado cada uno de ellos por 500 triángulos, 500 rectángulos, 500 elipses y 500 semielipses. Para la evaluación del ruido se emplearon 11 conjuntos como los anteriores, cada uno de ellos con diferente nivel de ruido, con σ variando desde 0 hasta 10 píxeles. Por otro lado, para la evaluación de las oclusiones, se emplearon otros 7 conjuntos, cada uno de ellos con diferente nivel de oclusión, variando el tamaño de la oclusión desde el 10 % del tamaño del objeto, de 5 en 5 hasta el 40 %.

En la tabla 4.1 se muestran los resultados obtenidos para valores de la desviación típica σ igual a 0 (sin ruido), 5 y 10 píxeles. Para cada nivel de ruido, se muestra el porcentaje de acierto en la clasificación y la precisión en la localización de la figura. Para completar los resultados del funcionamiento del algoritmo en función del ruido, en la figura 4.15 se muestra gráficamente los parámetros anteriores en función del nivel de ruido. En (a) se muestra el porcentaje de acierto en la clasificación, y en (b) la precisión en la localización de la figura.

De los resultados anteriores, se puede comprobar como la probabilidad de acierto en la clasificación se mantiene en un nivel elevado hasta un valor de la desviación típica de unos 7 píxeles. Para valores de σ mayores podemos ver como este porcentaje decrece rápidamente, debido a que, para niveles altos de ruido, la figura es prácticamente irreconocible, y en esas condiciones incluso una inspección visual sería también incorrecta. Con respecto a la precisión en la localización, también podemos ver que dicho valor es aceptable para valores normales de ruido, de hasta unos 7 píxeles de desviación típica. Es necesario aclarar aquí que, incluso para figuras sin ruido añadido, el error siempre es superior a aproximadamente



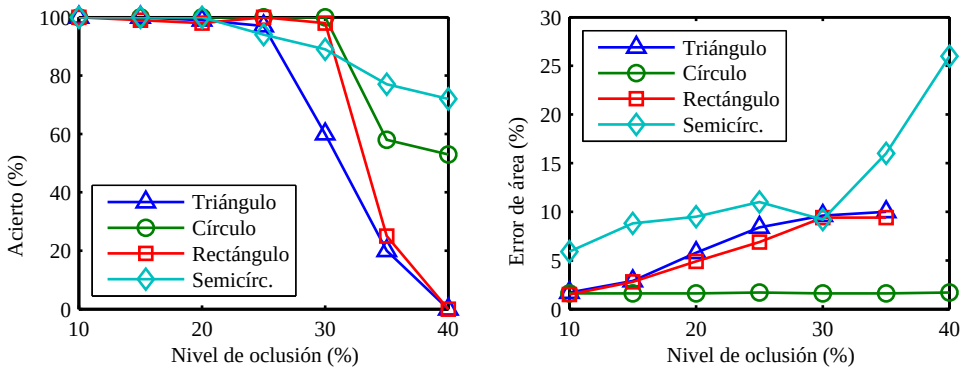
(a) (b)

Figura 4.15: Resultados en función del ruido.

1 píxel en la precisión de la localización, debido principalmente a errores de redondeo a la hora de construir la figura estimada. Para los semicírculos, en general la precisión es menor que para el resto de formas. La razón es que la comparación se realiza sobre la elipse entera que generó la semielipse, mientras que para la localización de la elipse sólo están disponibles la mitad de los puntos, ya que la otra mitad pertenecen a la parte eliminada.

El funcionamiento del algoritmo en presencia de oclusiones parciales se muestra en la figura 4.16. Igual que en la discusión anterior, en (a) se muestra el porcentaje de acierto en la clasificación de la forma en función del porcentaje de la oclusión, mientras que en (b) se muestra la precisión en la localización de la figura. En dichas figuras se puede comprobar como, hasta aproximadamente un 25 % de oclusión, ambas medidas toman un valor aceptable. Para oclusiones iguales o superiores a un 30 %, los valores decrecen rápidamente, debido a que para dichos valores de oclusión, más de un cuarto o más de la figura ha desaparecido. Esta degradación es más notoria para triángulos y rectángulos, debido sobre todo al diseño de la aplicación que genera las figuras. Como se comentó anteriormente, para triángulos y rectángulos, la aplicación siempre oculta uno de los vértices de la figura, lo que siempre será mucho más perjudicial que ocultar cualquier parte de un círculo. Incluso para oclusiones por encima del 35 % el porcentaje de acierto es tan bajo que no es posible disponer de datos válidos para el cálculo de la precisión en la localización, como se puede ver en la figura 4.16. Para semicírculos tenemos el caso intermedio.

El algoritmo también ha sido preparado para devolver, como parte de los resultados de la ejecución, la figura detectada dibujada superpuesta sobre la figura original. Las figuras son dibujadas con líneas en colores, como puede verse en la figura 4.13 y 4.14. En estos ejemplos, puede comprobarse el funcionamiento correcto del algoritmo incluso en presencia de niveles altos de ruido u oclusiones. Esta facilidad de la aplicación resulta muy útil a la hora de inspeccionar los resultados del algoritmo con imágenes reales, donde no se dispone de la información real de las figuras presentes en la imagen.



(a) (b)

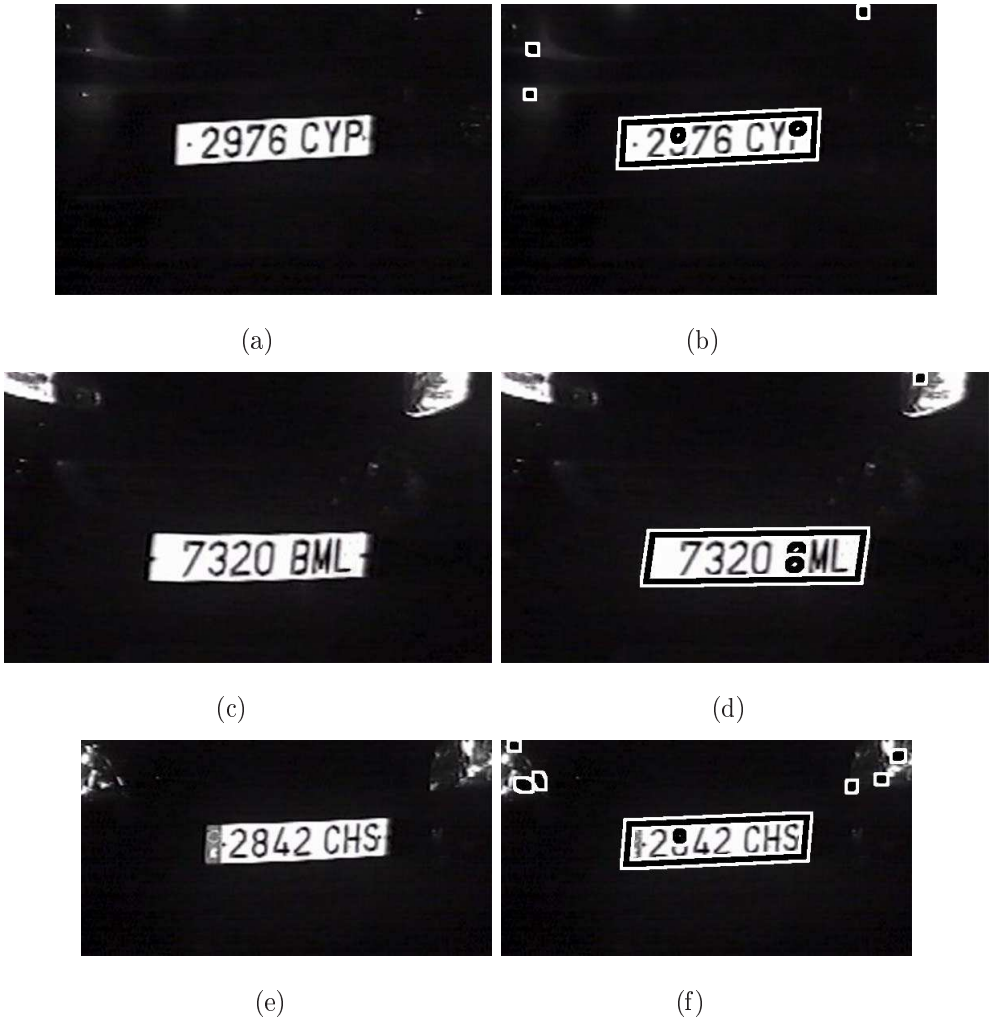
Figura 4.16: Resultados en función del nivel de oclusión

4.5.1. Ejemplo práctico 1: localización de matrículas

Como ejemplo de aplicación práctica del algoritmo implementado en esta sección, se describirá a continuación el funcionamiento de un módulo de detección y localización de matrículas para un sistema de control de acceso de vehículos mediante la lectura de su matrícula. El sistema consta de tres etapas: una primera de segmentación, donde se tratará de extraer la matrícula como una componente conectada, aparte de otros objetos, que se corresponderán con errores de segmentación, y que deberán ser eliminados por etapas posteriores. La segunda etapa se corresponde con la detección y localización de la matrícula, que comprenderá básicamente el algoritmo expuesto en esta sección, con la particularidad de que únicamente se atenderán a aquellos objetos que presenten forma rectangular. La tercera etapa realizará el reconocimiento de los caracteres de la matrícula.

Para facilitar el proceso de segmentación, se ha optado por realizar la grabación de las imágenes mediante una cámara de infrarrojo, e iluminación auxiliar, también infrarroja. De esta forma, conseguimos que el sistema sea robusto frente a cambios de iluminación de la luz visible. En las figuras 4.17 (a) y (c) podemos ver dos ejemplos de imágenes tomadas bajo estas condiciones. En estas imágenes podemos comprobar como la segmentación puede realizarse mediante una umbralización de nivel de gris.

Si la segmentación consigue aislar la matrícula del resto de objetos que aparezcan en la matriz binaria, el proceso de detección y localización de la matrícula debería funcionar correctamente, tal y como se ha visto a lo largo de esta sección. En las figuras 4.17 (b) y (d) se muestra el resultado del proceso de detección y localización para las matrículas de las figuras anteriores. El resultado se muestra con líneas negras, dentro de dos líneas blancas para mejorar el contraste de la visualización, pintadas entre cada par de vértices del rectángulo estimado. En ambas figuras, podemos comprobar como la precisión en la localización de las matrículas es bastante precisa.



(a) (b)

(c) (d)

(e) (f)

Figura 4.17: Ejemplo de detección de matrículas. (a, c, e): Imágenes originales tomadas con cámaras de infrarrojo. (b, d, f): Resultados de la detección y localización de las matrículas. Las formas adicionales que no se corresponden con la matrícula son errores de detección que serán eliminados en etapas posteriores.

Capítulo 5

Contribuciones y futuras líneas de investigación

A continuación se enumeran las principales contribuciones originales, y conclusiones obtenidas, que han sido aportadas en esta tesis. Posteriormente se comentarán las futuras líneas de investigación que quedan abiertas tras el desarrollo de la tesis.

5.1. Aportaciones originales

Dado que la parte principal del desarrollo de esta tesis doctoral ha sido dividido en los tres de los bloques más importantes que forman un sistema de videovigilancia, es decir, la extracción de los objetos de primer plano, o detección de movimiento, la estimación de distancias y el reconocimiento y clasificación de objetos, en las siguientes secciones se indicarán las aportaciones en cada uno de los campos de forma separada.

5.1.1. Aportaciones en detección de movimiento

Generalmente, en los sistemas más avanzados de videovigilancia, el bloque de detección de movimiento suele estar formado por dos bloques, que serían el bloque de mantenimiento del fondo, y el de segmentación, o detección de movimiento propiamente dicho. Respecto al bloque de mantenimiento del fondo, las principales contribuciones serían:

- La principal contribución original, de la cual no ha sido encontrada ninguna otra propuesta similar de otros investigadores en este campo, es la clasificación del fondo en función del comportamiento de éste a lo largo del tiempo. Esta división facilita el análisis de la escena por parte del módulo de detección de movimiento, e incluso módulos superiores, que no han sido desarrollados en esta tesis.

- Se ha mejorado el bloque de estimación de la imagen del fondo cuando la escena analizada presenta gran cantidad de movimiento del fondo, y en presencia de cambios de iluminación. El método desarrollado permite la estimación del fondo con la

ventaja de que evita la incorporación a éste de los objetos en movimiento, aparte de tener una buena respuesta en presencia de cambios bruscos, tales como nubes, cambios de iluminación, etc.

- Se ha propuesto un método alternativo para el cálculo de la varianza de los píxeles de la imagen. Se ha demostrado teóricamente, y comprobado experimentalmente, que funciona mejor que los métodos tradicionales propuestos en la literatura cuando las escenas presentan gran variación del nivel medio del fondo.

Respecto a las contribuciones en el bloque de detección de movimiento, la principal aportación deriva de la clasificación del fondo realizada en el módulo de mantenimiento del fondo. En concreto, la detección de movimiento se realiza en función del comportamiento de cada zona de la imagen, es decir, de la clasificación realizada por el módulo anterior.

Los algoritmos aquí desarrollados nos han permitido implementar, de una forma sencilla, aplicaciones de utilidad práctica, como fue por ejemplo, un sistema de control de congestión en carreteras, o incluso mejorar las prestaciones de un sistema típico de videovigilancia en escenas con movimiento de fondo, como árboles, agua, etc, especialmente la reducción de la probabilidad de falsas alarmas.

5.1.2. Aportaciones en estimación de distancias

En esta tesis se han analizado dos de los métodos más comúnmente empleados en la estimación de distancias mediante procesado de imagen, la visión estereoscópica y el desenfoque de la imagen.

En visión estereoscópica, se ha desarrollado un nuevo algoritmo que toma, como entrada, aparte del par de imágenes estéreo, la máscara de movimiento de la escena vigilada. Gracias a que el sistema sólo evalúa la distancia para los objetos de primer plano, hemos conseguido desarrollar un algoritmo robusto y a la vez rápido. Dado que la mayoría de los investigadores se centran o bien en la detección de movimiento o bien en la visión estereoscópica, pero no en ambas a la vez, podemos considerar este desarrollo como una aportación original, en el sentido de que se han utilizado ambos bloques conjuntamente para poder obtener un resultado más fiable.

En cuanto a la estimación de distancias mediante el desenfoque de la imagen, la principal aportación ha sido el desarrollo de un algoritmo que permite estimar distancias con el uso de una única imagen, en lugar del mínimo de dos imágenes que son requeridas por los algoritmos más populares existentes en la literatura de cálculo de profundidad por desenfoque.

5.1.3. Aportaciones en clasificación de formas

En este apartado dos son las aportaciones originales realizadas en esta tesis.

En primer lugar, se ha desarrollado un algoritmo para la clasificación de formas simples, en este caso, triángulos, rectángulos, elipses y semielipses, aunque extender el sistema para incluir alguna otra forma simple es inmediato. El algoritmo a resultado ser bastante robusto frente al escalado, desplazamiento, rotación y distorsiones de proyección en general. También se ha comprobado que presenta gran inmunidad frente a niveles de ruido elevado y a oclusiones parciales.

Por otro lado, se ha desarrollado un método de normalización y reorientación de los objetos detectados basados en homografías de dos dimensiones que permite preparar el objeto detectado para pasos posteriores, principalmente los centrados en el reconocimiento del contenido del objeto. En este caso, el cálculo es dependiente de la forma del objeto, lo que lo hace más preciso que un algoritmo de reorientación genérico que no tenga en cuenta la forma del objeto.

También podríamos enumerar como aportación original el desarrollo de una aplicación para la evaluación automática de cualquier algoritmo de clasificación de formas simples. Hasta la fecha, la mayor parte de los trabajos evaluaban el funcionamiento del algoritmo mediante inspección visual de los resultados obtenidos. Con el uso de esta aplicación, no es necesario ninguna intervención manual, ya que la propia aplicación genera las figuras a procesar, y toda la información necesaria para comparar con los resultados obtenidos. Especialmente novedoso es la inclusión del ruido a las imágenes, que nos ha permitido generar figuras muy parecidas a las que obtendríamos con imágenes reales.

5.2. Futuras líneas de investigación

Los estudios realizados, junto con los algoritmos desarrollados en esta tesis, sientan las bases de futuras líneas de investigación, entre las que cabe citar:

- En el apartado de detección de movimiento se han sentado las bases para el desarrollo de funciones de más alto nivel, entre las que cabe citar la detección de objetos de primer plano, seguimiento de objetos, etc. En general, en el desarrollo de aplicaciones de procesado de imágenes y de vídeo, las etapas claves son aquellas que trabajan a nivel de píxel, ya que son muy dependientes de la calidad de las imágenes, de las condiciones de iluminación y del ruido. En nuestro caso se han desarrollado las funciones de más bajo nivel, que simplifica el desarrollo de las funciones superiores.

- Respecto a la clasificación del fondo propiamente dicha, es posible investigar nuevos métodos de estimación de los estadísticos del fondo que mejoren los resultados obtenidos hasta ahora, o incluso si el sistema lo permite, combinar varios métodos para aumentar la fiabilidad de la clasificación. Si fuese necesario, sería posible también definir nuevas categorías de comportamiento, si en el desarrollo de funciones de niveles superiores se hiciera necesario.

- El desarrollo de la estimación de distancias mediante desenfoque se limitó a objetos que presentan un borde abrupto en la imagen enfocada del objeto. Sería necesario analizar las expresiones utilizadas en esta sección para extender la aplicación a objetos que presenten otro tipo de bordes en la imagen enfocada.

- Respecto a la investigación desarrollada en el tema de visión estereoscópica, una vez implementadas las funciones de correspondencia de objetos entre un par de imágenes, el siguiente paso sería el poder aumentar el número de imágenes para aplicaciones en las que se disponga de más de dos cámaras visualizando el mismo escenario. Para ello, sería necesario entre otras cosas, el desarrollo de funciones que permitan la calibración del conjunto de cámaras mediante el tensor trifocal o cuadrifocal, en lugar de la matriz fundamental utilizada hasta ahora.

- Conjugando las funciones de detección de movimiento, seguimiento de objetos y visión estereoscópica, sería posible construir la trayectoria seguida en el espacio por un objeto determinado. Para ciertas aplicaciones, sería interesante poder definir, a partir de dicha reconstrucción, trayectorias permitidas y no permitidas, de tal forma que sería posible aumentar la fiabilidad del sistema, y reducir el número de falsas alarmas.

- En el tema de clasificación de formas, sería interesante comprobar el funcionamiento del sistema implementado cuando las formas consideradas son más complejas que las consideradas hasta ahora. En cualquier caso, es necesario investigar el funcionamiento, en el contexto actual, de clasificadores más complejos, como pueden ser las Máquinas de Vectores Soporte (SVM) y las Redes Neuronales (NN), con respecto al clasificador empleado en esta tesis, tanto en precisión como en tiempo de ejecución.

- El módulo de normalización y reorientación de los objetos es un tema prácticamente cerrado, aunque sería posible estudiar variantes del método propuesto para comprobar si el sistema es capaz de aumentar la precisión en la localización de la figura. Por ejemplo, se puede analizar si la implementación de algoritmos iterativos de minimización geométrica para la estimación de rectas y elipses mejora la precisión del sistema, sin aumentar notablemente el tiempo de computo.

- Siguiendo con el apartado de normalización, el siguiente paso sería la implementación de las funciones capaces de reconocer el contenido del objeto. Por ejemplo, si se trata de la detección y reconocimiento de matrículas, sería necesario comprobar el funcionamiento de algoritmos de reconocimiento de caracteres. Si se trata de una aplicación de reconocimiento de señales de tráfico, sería necesario el desarrollo de un módulo que sea capaz de identificar el contenido de la señal, una vez identificada su forma geométrica y normalizada la imagen de la señal.

Bibliografía

[Antani03] Antani, Sameer, L. Rodney Long, George R. Thoma, y R. Joe Stanley. «Vertebra Shape Classification using MLP for Content-Based Image Retrieval.» *Neural Networks*, 1, (2003), 160–165.

[Aoyagi96] Aoyagi, Y., y T. Asakura. «A study on traffic sign recognition in scene image using genetic algorithms and neural networks.» En *Proc. of the 22nd. IEEE Int. Conf. Industrial Electronics, Control and Instrumentation*. Taipeh, Taiwan, 1996, tomo 3, 1838–1843.

[Asif05] Asif, M., A. S. Malik, y Tae-Sun Choi. «3D shape recovery from image defocus using wavelet analysis.» En *Proc. IEEE International Conference on Image Processing ICIP 2005*. 2005, tomo 1, I–1025–8.

[Becker00] Becker, Glenn, y Peter Bock. «Shape Classification using a Radial Feature Token.» En *29th Applied Imagery Pattern Recognition Workshop (AIPR'00)*. 2000, 47–53.

[Benosman98] Benosman, R., y S. Bing. *PANORAMIC VISION. Sensors, Theory and Applications*. Springer-Verlag, 1998.

[Bing01] Bing, Z., G. Yunhong, L. Bo, Z. Guangwei, y T. Tian. «A Practical Algorithm for Exception Event Detection for the Home Video Security Surveillance.» En *Proc. of ICII 2001*. 2001, tomo 3, 202–208.

[Boufama02] Boufama, B., y K. Jin. «Towards a fast and reliable dense matching algorithm.» En *International Conference on Vision Interface*. 2002, 178–185.

[Boult01] Boult, T., R. Micheals, X. Gao, y M. Eckmann. «Into the woods: Visual surveillance of noncooperative and camouflaged targets in complex outdoor settings.» En *Proceedings of the IEEE*. 2001, tomo 89, 1382–1402.

[Lowe07] Brown, M., y D Lowe. «Automatic Panoramic Image Stitching using Invariant Features.» *International Journal of Computer Vision*, 74, nº 1, (2007), 59–73.

[Chen00] Chen, Y.S., Y.P. Hung, y C.S. Fuh. «A Fast Block Matching Algorithm Based on the Winner-Update Strategy.» En *In Proccedings of the Fourth Asian Conference on Computer Vision*. 2000, tomo 2, 977–982.

[Huang99] Ching-Kay, Huang, y Tsuhan Chen. «Motion Activated video surveillance using TI DSP.» En *DSPS FEST'99*. 1999.

[Chiu08] Chiu, Shih-Hsuan, Che-Yen Wen, y Wei-Chieh Kao. «An effective surveillance video retrieval method based upon motion detection.» En *Proc. IEEE International Conference on Intelligence and Security Informatics ISI 2008*. 2008, 261–262.

[Cord98] Cord, M., N. Paparoditis, y M. Jordan. «Dense, reliable, and depth discontinuity preserving DEM computation from H.R.V. urban stereopairs.» En *Proceedings of the ISPRS Commission II Symposium*. Cambridge, UK, 1998, 49–56.

[Cox96] Cox, Ingemar J., Sunita L. Hingorani, y Satish B. Rao. «A Maximum Likelihood Stereo Algorithm.» *Computer Vision and Image Understanding*, 63, nº 3, (1996), 542–567.

[Cuadrado06] Cuadrado, C., A. Zuloaga, J. L. Martin, J. Lazaro, y J. Jimenez. «Real-Time Stereo Vision Processing System in a FPGA.» En *Proc. IECON 2006 - 32nd Annual Conference on IEEE Industrial Electronics*. 2006, 3455–3460.

[Dailey06] Dailey, M. N., y M. Parnichkun. «Simultaneous Localization and Mapping with Stereo Vision.» En *Proc. 9th International Conference on Control, Automation, Robotics and Vision ICARCV '06*. 2006, 1–6.

[DawsonHowe96] Dawson-Howe, K. «Active Surveillance Using Dynamic Background Subtraction.» *Inf. Téc. TCD-CS-96-06*, Dept of Computer Science, Trinity College, Dublin, IRELAND, 31 1996.

[Dionisio04] Dionisio, Carlos R. P., y Hae Yong Kim. «New Features for Affine-Invariant Shape Classification.» En *Proceedings of the International Conference on Image Processing (ICIP'04)*. 2004, 2135–2138.

[Elgammal02] Elgammal, Ahmed, Ramani Duraiswami, David Harwood, y Larry S. Davis. «Background and Foreground Modeling Using Nonparametric Kernel Density for Visual Surveillance.» En *Proceedings of the IEEE*. 2002, tomo 90, 1151–1163.

[Escalera94] de la Escalera, Arturo, L.E. Moreno, E. A. Puente, y M. A. Salichs. «Neural traffic sign recognition for autonomous vehicles.» En *IECON'94. 20th. International Conference on Industrial, Electronics, Control and Instrumentation*. IEEE, New-York, USA, 1994, tomo 2, 841–846.

[Foresti00] Foresti, G. L., P. Mähönen, y C.S. Regazzoni. *Multimedia video-based surveillance systems. Requirements, issues and solutions*. Kluwer Academic Publishers, 101 Philip Drive, Assinippi Park, Norwell, Massachusetts 02061 USA, 2000, 1 ed$^{\underline{\text{ón}}}$.

[Fujiyoshi98] Fujiyoshi, H., y A. Lipton. «Real-time human motion analysis by image skeletonization.» En *IEEE Workshop on Applications of Computer Vison*. 1998.

[Gander94] Gander, W., G. H. Golub, y R. Strebel. «Least-squares fitting of circles and ellipses.» *BIT*, 34, (1994), 558–578.

[Gao07] Gao, Lu-Fang, Yu-Xian Gai, y Sheng Fu. «Simultaneous Localization and Mapping for Autonomous Mobile Robots Using Binocular Stereo Vision System.» En *Proc. International Conference on Mechatronics and Automation ICMA 2007*. 2007, 326–330.

[Gao00] Gao, X., T.E. Boult, F. Coetzee, y V. Ramesh. «Error Analysis of Background Adaption.» En *CVPR00*. 2000, I: 503–510.

[Gavrila99] Gavrila, D. M., y V. Philomin. «Real-Time object detection for SMART vehicles.» En *Proc. of IEEE international conference on computer science*. Kerkyra, Greece, 1999, 87–93.

[Ghasemlou01] Ghasemlou, Mani. «Marr and Poggio's Cooperative Stereopsis Algorithm: An Implementation In C.» *Inf. téc.*, Fundamentals of Computer vision, April 2001.

[Gonzalez93] Gonzalez, R. C., y R. E. Woods. *Digital Image Processing*. Addison-Wesley, 1993.

[Goodman96] Goodman, J.W. *Introduction to Fourier Optics*. McGrau-Hill International Editions, 1996.

[Haritaoglu98] HARITAOGLU, Ismail, David HARWOOD, y Larry S. DAVIS. «W4: Who? When? Where? What? A Real Time System for Detecting and Tracking People.» En *FG*. 1998, 222–227.

[Hartley03] HARTLEY, R., y A. ZISERMANN. *Multiple view geometry in computer vision*. Cambridge University Press, 2003, second edón.

[Haykin99] HAYKIN, S. *Neural Networks. A comprehensive foundation*. Prentice Hall Inc., 1999, second edón.

[Herath06] HERATH, D. C., S. KODAGODA, y G. DISSANAYAKE. «Simultaneous Localisation and Mapping: A Stereo Vision Based Approach.» En *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2006, 922–927.

[Hirschmuller01] HIRSCHMÜLLER, Heiko. «Improvements in Real-Time Correlation-Based Stereo Vision.» *IEEE Workshop on Stereo and Multi-Baseline Vision*, (2001), 141–148.

[Ishizuka04] ISHIZUKA, Y., y Y. HIRAI. «Segmentation of road sign symbols using opponent-color filters.» En *ITSWC2004*. Nagoya, 2004, 8.

[Iwata08] IWATA, K., Y. SATOH, K. SAKAUE, T. KOBAYASHI, y N. OTSU. «Development of Software for Real-Time Unusual Motions Detection by Using CHLAC.» En *Proc. ECSIS Symposium on Bio-inspired Learning and Intelligent Systems for Security BLISS '08*. 2008, 34–39.

[Jain89] JAIN, A. K. *Fundamentals of Digital Image Processing*. Prentice-Hall, 1989.

[Jarvis83] JARVIS, R.A. «A perspective on Range Finding Techniques for Computer Vision.» *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5, nº 2, (1983), 122–139.

[Jong07] JONG, Shyh-Ming. «Depth from Defocus using Radial Basis Function Networks.» En *Proc. International Conference on Machine Learning and Cybernetics*. 2007, tomo 4, 1888–1893.

[Kameda96] KAMEDA, Y., y M. MINOH. «A human motion estimation method using 3-successive viedo frames.» En *In Proc. of the Int'l. Conf. on Visual Systems and Multimedia '96*. 1996, 135–140.

[Kamnoonwatana06] KAMNOONWATANA, N., A. KUPRIANOV, P. SAENGUDOMLERT, T. SANGUANKOTCHAKORN, y K. KANCHANASUT. «DVTS Video Frame Rate Adjustment Based on Motion Detection.» En *Proc.*

Second International Conference on Automated Production of Cross Media Content for Multi-Channel Distribution AXMEDIS '06. 2006, 229–235.

[Kanade98] KANADE, T., R. T. COLLONS, y A. J. LIPTON. «Advances in Cooperative Multi-Sensor Video Surveillance.» En *DARPA Image Understanding Workshop*. 1998, 3–24.

[Kauppinen95] KAUPPINEN, Hannu, Tapio SEPPANEN, y Matti PIETIKAINEN. «An Experimental Comparison of Autoregressive and Fourier-Based Descriptors in 2D Shape Classification.» *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17, (1995), 201–207.

[Kobayashi04] KOBAYASHI, Takumi, y Nobuyuki OTSU. «Action and Simultaneous Multiple-Person Identification Using Cubic Higher-order Local Auto-Correlation.» En *Proceedings of the 17th ICPR*. 2004, 741–744.

[Koller94] KOLLER, Dieter, Joseph WEBER, y Jitendra MALIK. «Robust Multiple Car Tracking with Occlusion Reasoning.» En *ECCV (1)*. 1994, 189–196.

[Kunttu03] KUNTTU, Iivari, Leena LEPISTO, Juhani RAUHAMAA, y Ari VISA. «Multiscale Fourier Descriptor for Shape Classification.» En *Proceedings of the 12th International Conference on Image Analysis and Processing (ICIAP'03)*. 2003, 1–6.

[Lee97] LEE, Chang-Hsing, y Ling-Hwei CHEN. «A Fast Motion Estimation Algorithm Based on the Block Sum Pyramid.» *IEEE Transactions on Image Processing*, 6, nº 11, (1997), 1587–1591.

[Lim08] LIM, Young-Chul, Chung-Hee LEE, Soon KWON, y Woo-Young JUNG. «Distance estimation algorithm for both long and short ranges based on stereo vision system.» En *Proc. IEEE Intelligent Vehicles Symposium*. 2008, 841–846.

[Liu02] LIU, H., D. LIU, y J. XIN. «Real-Time recognition of road traffic sign in motion image based on genetic algorithm.» *Proc. of the 1st. Int. Conference on Machine Learning and Cybernetics*, (2002), 83–86.

[Giraudon94] LOTTI, Jean-Luc, y Gérard GIRAUDON. «Adaptive Window Algorithm for Aerial Image Stereo.» En *ICVPR* (Jean-luc Lotti, ed.). Jerusalem, 1994, 701–703.

[McKenna00] McKenna, S. J., S. J. Zoran Duric, A. Rosenfeld, y H. Wechsler. «Tracking Groups of People.» *Computer Vision and Image Understanding*, 80, (2000), 42–56.

[Meier98] Meier, T., y K. N. Ngan. «Automatic segmentation of moving objects for video object plane generation.» *IEEE Transactions on Circuits and Systems for Video Technology*, 8, nº 5, (1998), 525–538.

[Mansson98] Månsson, Jens. «Stereovision: A model of human stereopsis.» *Inf. téc.*, Lund Univ. Cognitive Science, 1998.

[Oh07] Oh, Jong-Kyu, y Chan-Ho Lee. «Development of a stereo vision system for industrial robots.» En *Proc. International Conference on Control, Automation and Systems ICCAS '07*. 2007, 659–663.

[Otsu88] Otsu, Nobuyuki, y Takio Kurita. «A New Scheme for Practical Flexible and Intelligent Vision Systems.» En *IARP Workshop on CV*. 1988, 431–435.

[Pajares00] Pajares, G., y J. M. de la Cruz. «Formas a Partir de X.» *Revista Electrónica de Visión por Computador*.

[Pentland87] Pentland, A.P. «A New Sense for Depth of Field.» *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9, nº 4, (1987), 523–531.

[Qiang01] Qiang, Wang, Hu Wiping, Hu Jianping, y Hu Kai. «A wavelet Multiresolution Edge Analysis Method for Recovery of Depth from Defocus Images.» En *Wavelet Analysis and Its Applications*. Springer-Verlag, 2001, Lecture Notes in Computer Science, 217–222.

[Ridder95] Ridder, C., O. Munkelt, y H. Kirchner. «Adaptive background estimation and foreground detection using Kalman filtering.» En *ICRAM'95*. 1995, 193–199.

[Roma02] Roma, Nuno, José Santos-Victor, y José Tomé. «A Comprative Analysis of Cross-Correlation Matching Algorithms Using a Pyramidal Resolution Approach.» *World Scientific*, (2002), 1–25.

[Rosin95] Rosin, P., y T. Ellis. «Image difference threshold strategies and shadow detection.» En *Proceedings of the 6th British Machine Vision Conference*. 1995, 347–356.

[Sandoval00] Sandoval, H., T. Hattori, S. Kitagawa, y Y. Ghigusa. «Angle-dependent Edge detection for traffic signs recognition.» En *Proc.*

of the IEEE Intelligent Vehicles Symposium. IEEE, Dearbom (MI) USA, 2000, 308–313.

[Sara02] SARA, Radim. «Finding the Largest Unambiguous Component of Stereo Matching.» En *ECCV (3)*. 2002, 900–914.

[Schechner98] SCHECHNER, Y.Y., y N KIRYATI. «Depth from Defocus vs Stereo: How Different Really are They?» En *Proc. International Conference on Pattern Recognition*. 1998, 1784–1786.

[Stauffer99] STAUFFER, Chris, W. ERIC, y L. GRIMSON. «Adaptive Background Mixture Models for Real-Time Tracking.» En *CVPR*. 1999, 2246–2252.

[DiStefano04] DI STEFANO, L., M. MARCHIONNI, y S. MATTOCCIA. «A Fast Area-Based Stereo Matching Algorithm.» *Image and Vision Computing*, 22, nº 12, (2004), 983–1005.

[DiStefano01] DI STEFANO, L., G. NERI, y E. VIARANI. «Analysis of pixel-level algorithms for video surveillance applications".» En *CIAP01*. 2001, 541–546.

[DiStefano99] DI STEFANO, L., y E. VIARANI. «Vehicle Detection and Tracking Using the Block Matching Algorithm.» En *Proc. of 3rd IMACS/IEEE*. 1999, tomo 1, 4491–4496.

[Suau05] SUAU, P. «Robust artificial landmark recognition using polar histograms.» *Lecture Notes in Computer Science*, 3803, (2005), 455–461.

[Subbarao94] SUBBARAO, M., y G. SURYA. «Depth from Defocus: A Spatial Domain Approach.» *International Journal of Computer Vision*, 13, nº 3, (1994), 271–294.

[Sun02] SUN, Changming. «Fast stereo matching using rectangular subregioning and 3D maximum-surface techniques.» *International Journal of Computer Vision*, 47, nº 1/2/3, (2002), 99–117.

[Szelisky94] SZELISKI, R. «Image Mosaicing for Tele-Reality Applications.» *Inf. téc.*, Cambridge Research Lab., 1994.

[Tang02] TANG, Li, H.T. TSUI, y C.K WU. «Dense stereo matching based on propagation with a Voronoi diagram.» En *ICVGIP*. 2002.

[Utasi07] UTASI, A., y L. CZUNI. «Reducing the Foreground Aperture Problem in Mixture of Gaussians Based Motion Detection.» En *Proc. Multimedia Communications and Services Systems, Signals and*

Image Processing and 6th EURASIP Conference focused on Speech and Image Processing 14th International Workshop on. 2007, 157–160.

[Vapnik00] VAPNIK, V. *The Nature of Statistical Learning Theory.* Springer-Verlag, New York, 2000.

[Vitabile01] VITABILE, S., G. POLLACCIA, G. PILATO, y F. SORBELLO. «Road signs recognition using a dynamic pixel aggregation technique in the HSV color space.» En *Proc. of the 11th. Int. Conference on Image Analysis and Processing* (IEEE, ed.). IEEE, Palermo, Italy, 2001, 572–577.

[Wagner95] WAGNER, David B. «Dynamic Programming.» *The Mathematica Journal*, 5, nº 4, (1995), 42–51.

[Watanabe98] WATANABE, M., y S. NAYAR. «Rational filters for passive depth from defocus.» *International Journal of Computer Vision*, 27, nº 3, (1998), 203–225.

[Wei05] WEI, Yangjie, Zaili DONG, Lei MIAO, y W. J. LI. «Analysis of depth from defocus measurements for micro-imaging and 3D micro-visual reconstruction.» En *Proc. IEEE International Conference on Information Acquisition.* 2005, 6pp.

[Yang07] YANG, Wenming, Wang LU, y Naitong ZHANG. «Object Extraction Combining Image Partition with Motion Detection.» En *Proc. IEEE International Conference on Image Processing ICIP 2007.* 2007, tomo 3, III–337–III–340.

[Zhao06] ZHAO, Jun, y J. KATUPITIYA. «A Dynamic Programming Approach Based Stereo Vision Algorithm Improving Object Border Performance.» En *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems.* 2006, 5315–5320.

[Zitnick99] ZITNICK, C. Lawrence, y Takeo KANADE. «A Cooperative Algorithm for Stereo Matching and Occlusion Detection.» *Inf. téc.*, The Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213, October 1999.