

*Universidad de Granada*

**DEPARTAMENTO DE ARQUITECTURA Y TECNOLOGÍA DE  
COMPUTADORES**



*ugr* | **Universidad  
de Granada**

**Circuitos bio-inspirados para la evaluación de  
movimiento en tiempo real y sus aplicaciones**

**TESIS DOCTORAL**

**M<sup>a</sup> Sonia Mota Fernández**

**Granada, 2007**





Dr. Eduardo Ros Vidal, Profesor Titular de Universidad, y Dr. Francisco José Pelayo Valle, Catedrático de Universidad, ambos del Departamento de Arquitectura y Tecnología de Computadores de la Universidad de Granada,

CERTIFICAN:

Que la memoria titulada **“Circuitos bio-inspirados para la evaluación de movimiento en tiempo real y sus aplicaciones”**, ha sido realizada por Dña. M<sup>a</sup> Sonia Mota Fernández bajo nuestra dirección en el Departamento de Arquitectura y Tecnología de Computadores de la Universidad de Granada para optar al grado de Doctor por la Universidad de Granada.

**Granada, a 19 de Enero de 2007**

Fdo.: Dr. Eduardo Ros  
Vidal

Fdo.: Dr. Francisco José Pelayo  
Valle

Director de la Tesis

Director de la Tesis



*Universidad de Granada*

**DEPARTAMENTO DE ARQUITECTURA Y TECNOLOGÍA DE  
COMPUTADORES**



*ugr*

**Universidad  
de Granada**

**Circuitos bio-inspirados para la evaluación de  
movimiento en tiempo real y sus aplicaciones**

**Memoria presentada por  
M<sup>a</sup> Sonia Mota Fernández**

Para optar al grado de  
**DOCTOR POR LA UNIVERSIDAD DE GRANADA**

Fdo.: M<sup>a</sup> Sonia Mota Fernández



## Agradecimientos

La maduración de este trabajo se ha producido a fuego lento, de manera que son muchas las personas que, de una u otra forma, han intervenido en este lento proceso.

Quiero empezar agradeciendo a toda mi familia su apoyo y cariño. Especialmente a mis padres, que han hecho posible con su amor y esfuerzo que yo me pueda dedicar a este proyecto, y que se han ilusionado con él tanto como yo.

Tampoco quiero olvidar a mis amigos, especialmente a Eduardo, Victoria, Fernando, M<sup>a</sup> José y Billy. Gracias a todos por acompañarme en la carrera de la vida.

Por supuesto existe mucha otra gente que me ha ayudado de múltiples maneras. Quiero agradecer a todos los miembros del Departamento de Arquitectura de computadores de la Universidad de Granada su consejo y ayuda en numerosas ocasiones. Me gustaría destacar a:

Eduardo Ros y Francisco Pelayo. Gracias por abrirme la puerta y por permitirme aprender tanto a vuestro lado.

Mis compañer@s del "ático de ciencias": Javier, Eva, Rodrigo, Richard, Rafa, Samuel, Antonio y Christian. Gracias por vuestros consejos y amistad.

Finalmente, quiero agradecer a mis compañeros de Córdoba, especialmente a Paco, Luisma y Jorge, su ayuda y su cálida acogida, que ha facilitado la finalización de esta tesis.





"La inteligencia consiste no sólo en el conocimiento, sino también en la destreza de aplicar los conocimientos en la práctica."

Aristóteles

"Y sin embargo, se mueve." (*Eppur si muove*)

Galileo Galilei

"Si he logrado ver más lejos, ha sido porque he subido a hombros de gigantes".

Isaac Newton

¿Cómo puede recorrer un campo que, debido al retroceso del último glaciar, se halla salpicado de incontables piedras y encontrar las puntas de flechas? ¿Por qué el ojo humano puede detectar una pequeña forma artificial perdida en el cosmos rasgado y turbulento de la naturaleza, una aguja de datos en un pajar de ruido?...

CRIPTONOMICON  
Neal Stephenson



---

## RESUMEN

---

Esta tesis doctoral trata sobre el diseño de circuitos bio-inspirados para la visión en tiempo real. Está estructurada en tres partes. La primera parte describe un nuevo modelo para la estimación de primitivas visuales de movimiento basado en los principios biológicos del funcionamiento del sistema visual de la mosca. En esta parte también se introduce una metodología de evaluación con secuencias reales, y se acredita el rendimiento del modelo.

La segunda parte de la memoria trata sobre la implementación del modelo en hardware reconfigurable. Para el diseño de este modelo se usan caminos de datos específicos. La arquitectura adopta técnicas de diseño basadas en un uso intensivo del paralelismo disponible en dispositivos basados en FPGAs. El uso intensivo de la segmentación de cauce de grano fino así como la utilización de múltiples unidades escalares permite alcanzar un rendimiento final de una estimación por ciclo de reloj del sistema. Esto representa una alta potencia computacional y abre la puerta para nuevas aplicaciones en tiempo real.

En la tercera parte se evalúan dos aplicaciones para el sistema de detección de movimiento, en las que el procesamiento en tiempo real es una necesidad patente.



---

---

# ÍNDICE GENERAL

---

---

	PÁGINA
<b>CAPÍTULO 1: Introducción</b>	1
1.1 Introducción	2
1.1.1 Motivación	3
1.1.2 Sistemas electrónicos en tiempo real	5
1.1.3 El estado de la cuestión	6
1.2 Objetivos del trabajo realizado	7
1.3 El sistema visual de los insectos	8
<b>CAPÍTULO 2: Algoritmo de detección de movimiento</b>	17
2.1 Introducción	18
2.2 Extracción de características	18
2.2.1 Detección de bordes mediante Sobel: convolución con una máscara cuadrada de 3x3	20
2.2.2 Detección de bordes mediante el detector de Sobel en el espacio logarítmico	23
2.2.3 Principio de 'non maximum suppression'	25
2.2.4 Umbralización de los bordes	26
2.3 Detección de movimiento: Correladores de Reichardt	30

2.4 Filtrado de la imagen	34
2.4.1 Coherencia espacial: Movimiento del sólido rígido	36
2.4.2 Coherencia temporal	38
2.5 Conclusiones	40
<b>CAPÍTULO 3: Implementación hardware del algoritmo de detección de movimiento</b>	<b>41</b>
3.1 Introducción	42
3.2 Arquitectura de procesamiento con segmentación de grano grueso	44
3.3 Arquitectura de procesamiento con segmentación de grano fino	45
3.3.1 Captación de la imagen (S1)	46
3.3.2 Extracción de bordes (S2)	48
3.3.3 Selección de la dirección del píxel mediante correlación con diferentes constantes de tiempo (S3)	50
3.3.4 Etapa de cómputo del máximo (“Winner takes all”) o etapa de estimación de la velocidad (S4)	51
3.3.5 Selección de las características dominantes (S5)	54
3.3.6 Módulo de gestión de memoria	55
3.4 Flujo de los datos y comunicación entre las etapas de procesamiento	57
3.5 Recursos de Hardware consumidos	60
3.6 Conclusiones	64
<b>CAPÍTULO 4: Evaluación del algoritmo de detección de movimiento</b>	<b>67</b>

4.1	Introducción	68
4.2	Evaluación I	70
4.3	Evaluación II	85
4.4	Evaluación III	95
4.5	Conclusiones	105
	<b>CAPÍTULO 5: Aplicación a la monitorización de adelantamientos: Vigilancia del ángulo muerto</b>	107
5.1	Introducción	108
5.2	Evaluación del algoritmo de detección de movimiento en tareas de monitorización de adelantamientos bajo diferentes condiciones de trabajo	113
5.2.1	Estudio del rango de velocidades detectadas	114
5.2.2	Estudio bajo diferentes condiciones luminosas	118
5.2.3	Estudio bajo diferentes condiciones climatológicas	119
5.2.4	Estudio respecto a la detección de distintos tipos de vehículos objetivo	122
5.3	Sistema de alarma basado en el seguimiento del vehículo objetivo	124
5.3.1	Algoritmo de seguimiento	124
5.3.2	Implementación hardware	126
5.3.3	Resultados	126
5.4	Mejoras en la detección de movimiento: reducción del efecto de la perspectiva	129
5.4.1	Métodos de reducción de la deformación: Canales de Velocidad frente a Remapeo Espacial de la imagen	129
5.4.2	Implementación en hardware específico	133
5.5	Conclusiones	134
	<b>CAPÍTULO 6: Conclusiones</b>	137



6.1 Discusión	138
6.2 Resultados publicados	141
6.2.1 Publicaciones internacionales con índice de impacto (SCI)	141
6.2.2 Publicaciones en Lecture Notes in Computer Science (también con Índice de Impacto SCI)	142
6.2.3 Conferencias internacionales	142
6.2.4 Conferencias nacionales	143
6.3 Conclusiones	144
<b>APENDICE: Sistema de realidad aumentada para la ayuda a pacientes con Visión de Túnel</b>	145
A.1 Introducción	146
A.2 La estrategia de realidad aumentada para dispositivos de ayuda a la baja visión	147
A.3 Dispositivos de realidad aumentada para ayuda a la Visión de Túnel con el movimiento como descriptor de la escena	149
<b>REFERENCIAS</b>	153

---

---

## ÍNDICE DE FIGURAS

---

---

	PÁGINA
<b>Figura 1.1.</b> Anatomía de una omatidia del ojo compuesto de los insectos.	10
<b>Figura 1.2.</b> Comparativa en el número de píxeles en sistemas visuales biológicos y artificiales.	11
<b>Figura 1.3.</b> Flujo de información desde la retina hasta la lámina lobular.	12
<b>Figura 1.4.</b> Modelo de respuesta de las células a la selectividad de la dirección basado en detectores de movimiento elementales.	14
<b>Figura 2.1.</b> Procedimiento de convolución para la obtención de los bordes de la imagen.	21
<b>Figura 2.2.</b> Obtención de los bordes verticales de la imagen.	22
<b>Figura 2.3.</b> Bordes verticales en el caso de una imagen contaminada con ruido de tipo 'sal y pimienta'.	23
<b>Figura 2.4.</b> Comparativa de los bordes obtenidos mediante Sobel y mediante el procesamiento de LIP-Sobel.	24
<b>Figura 2.5.</b> Principio de non-maximum suppression.	26
<b>Figura 2.6.</b> Umbralización y binarización de la imagen de bordes obtenida mediante el operador de Sobel lineal.	28
<b>Figura 2.7.</b> Umbralización y binarización de la imagen de bordes obtenida mediante el operador de LIP-Sobel.	28

<b>Figura 2.8.</b> Variabilidad del número de bordes que superan un umbral fijo.	29
<b>Figura 2.9.</b> Partiendo de un umbral igual a 30, mediante la introducción de un umbral variable controlamos que el número de bordes a lo largo de la secuencia permanezca constante.	29
<b>Figura 2.10.</b> Variabilidad del número de bordes detectados con umbral igual a 30 y con umbral variable.	29
<b>Figura 2.11.</b> Funcionamiento del modelo de correladores de Reichardt desarrollado para el algoritmo.	31
<b>Figura 2.12.</b> Extracción del movimiento mediante el algoritmo de correlación de Reichardt.	33
<b>Figura 2.13.</b> Conexiones sinápticas de tres neuronas colectoras que integran la actividad de los correladores de Reichardt en un área local de la imagen.	35
<b>Figura 2.14.</b> Descripción del proceso de filtrado mediante el criterio de movimiento del sólido rígido.	38
<b>Figura 2.15.</b> Proceso de filtrado del movimiento a lo largo de distintas imágenes de la secuencia.	39
<b>Figura 3.1.</b> Plataforma de prototipado RC200 de Celoxica empleada en la implementación del algoritmo.	43
<b>Figura 3.2.</b> Estructura de la segmentación de cauce de grano grueso del sistema, en el que vemos la distribución de los diferentes módulos de procesamiento.	44
<b>Figura 3.3.</b> Proceso de renovación cíclico de píxeles en los buffers generados por el circuito de captación de imágenes.	47
<b>Figura 3.4.</b> Segmentación de cauce para realizar las operaciones de extracción de bordes mediante LIP-Sobel.	49
<b>Figura 3.5.</b> Esquema de la estructura microsegmentada en las etapas de correlación y estimación de velocidad.	53

<b>Figura 3.6.</b> Flujo de Datos entre los distintos módulos de procesamiento dentro de la FPGA y con los bloques de memoria externa SRAM, la cámara y la salida VGA.	59
<b>Figura 3.7.</b> Caminos de datos para los distintos módulos de procesamiento.	60
<b>Figura 3.8.</b> Degradación de las características del circuito cuando aumenta el número de ciclos empleados en procesar un píxel en la etapa de estimación de velocidad (S4).	63
<b>Figura 4.1.</b> Selección de algunas imágenes correspondientes a la película “Tiempos Modernos” de Charles Chaplin.	70
<b>Figura 4.2.</b> Ejemplos de segmentación manual sobre los objetos que se desplazan por la escena.	71
<b>Figura 4.3.</b> Promedio en la secuencia de $SE_D$ y $SE_I$ frente al umbral que se aplica en la etapa de extracción de bordes y promedio en la secuencia del número de bordes.	73
<b>Figura 4.4.</b> Comparativa de NC, $SE_D$ y $SE_I$ cuando usamos una máscara de Sobel de 3x3 píxeles y de 7x7 píxeles.	74
<b>Figura 4.5.</b> Comparativa de NC, $SE_D$ y $SE_I$ cuando la correlación se realiza píxel a píxel y cuando usamos un conjunto de 11x11 píxeles.	75
<b>Figura 4.6.</b> Comparativa de los resultados de detección de movimiento cuando la correlación se realiza píxel a píxel y cuando usamos un conjunto de 11x11 píxeles.	76
<b>Figura 4.7. (a)</b> Relación entre NC, $SE_D$ y $SE_I$ y el tamaño del área de influencia para un umbral igual a 25; <b>(b)</b> Relación entre el NC y el umbral para distintas áreas de influencia.	77
<b>Figura 4.8.</b> Relación entre el porcentaje de ruido de fondo y el umbral para distintos tamaños de área de influencia.	79
<b>Figura 4.9.</b> Resultados de Nivel de Clasificación, la Sensibilidad Derecha y la Sensibilidad Izquierda antes y después de la etapa de filtrado.	80

<b>Figura 4.10.</b> Salida del algoritmo de detección de movimiento usando los parámetros que optimizan la eficiencia.	82
<b>Figura 4.11.</b> Superposición en una única imagen de los bordes principales pertenecientes a los objetos en movimiento de la secuencia.	83
<b>Figura 4.12.</b> Valor promedio y desviación estándar del módulo de la velocidad obtenido mediante el algoritmo de detección de movimiento y de forma manual.	84
<b>Figura 4.13.</b> Selección de algunas imágenes correspondientes a la película “Tiempos Modernos” de Charles Chaplin.	86
<b>Figura 4.14.</b> Segmentación manual del personaje para los distintos tramos en que se divide la película.	87
<b>Figura 4.15.</b> Representación del Nivel de Clasificación obtenida para cada fotograma de la secuencia, y para distintas umbralizaciones.	88
<b>Figura 4.16.</b> Número de bordes en movimiento a lo largo de la secuencia.	89
<b>Figura 4.17.</b> Efecto sobre los valores del Nivel de Clasificación ( $SE_D$ y $SE_I$ ) de la correlación mediante estructuras de 1x1 o de 11x11 píxeles para toda la secuencia en estudio.	90
<b>Figura 4.18.</b> Muestra de los resultados del algoritmo de detección de movimiento sobre la escena bajo estudio.	92
<b>Figura 4.19.</b> Resultados de la segmentación sobre la secuencia de evaluación.	94
<b>Figura 4.20.</b> Valor promedio y desviación estándar del módulo de la velocidad obtenido mediante el algoritmo de detección de movimiento y de forma manual.	95
<b>Figura 4.21.</b> Selección de algunos fotogramas correspondientes a la película “El increíble hombre menguante” de Jack Arnold.	96
<b>Figura 4.22.</b> Segmentación manual del personaje en movimiento en la secuencia bajo estudio.	97
<b>Figura 4.23.</b> Las gráficas de la izquierda representan el promedio en la secuencia de $SE_D$ y $SE_I$ frente al umbral y el número de bordes detectados que	99

forman parte de los objetos en movimiento.

<b>Figura 4.24.</b> Valor promedio del porcentaje de la Sensibilidad Derecha, Sensibilidad Izquierda y Nivel de Clasificación para distintos tamaños de las áreas de influencia y del umbral.	100
<b>Figura 4.25.</b> Imagen original y salida de la detección de movimiento de la secuencia bajo estudio.	102
<b>Figura 4.26.</b> Detecciones anómalas debidas al movimiento de la cámara.	103
<b>Figura 4.27.</b> Valor promedio y desviación estándar del módulo de la velocidad obtenido mediante el algoritmo de detección de movimiento y de forma manual.	104
<b>Figura 4.28.</b> Degradación en el número de píxeles activos en cada etapa del procesamiento para la secuencia actual.	104
<b>Figura 5.1.</b> Distintas situaciones peligrosas que se dan durante la conducción debido a la existencia de ángulos muertos o zonas ciegas en distintos vehículos.	111
<b>Figura 5.2.</b> Planteamiento del problema al que aplicaremos en sistema de detección de movimiento.	112
<b>Figura 5.3.</b> Selección de algunas imágenes correspondientes a una de las secuencias.	115
<b>Figura 5.4.</b> Distintos tamaños de vehículo objetivo considerados.	116
<b>Figura 5.5.</b> Media y Desviación Estándar de la Sensibilidad Derecha ( $SE_D$ ) y Rendimiento de la Segmentación ( $R_S$ ) para los tamaños A, B, C, D y E cuando el vehículo objetivo se aproxima a: <b>(a)</b> 1-3 m/s, <b>(b)</b> 5-10 m/s, <b>(c)</b> 15-20 m/s y <b>(d)</b> 25-30 m/s.	117
<b>Figura 5.6.</b> Secuencias tomadas al anoecer.	118
<b>Figura 5.7.</b> Media y Desviación Estándar de la Sensibilidad Derecha y Rendimiento de la Segmentación para los distintos tamaños del vehículo objetivo en secuencias tomadas al anoecer.	119
<b>Figura 5.8.</b> Secuencias tomadas al bajo diferentes condiciones meteorológicas:	120

(a) Niebla suave; (b) Niebla densa y lluvia; (c) Día muy nublado y (d) Lluvia.

**Figura 5.9.** Media y la Desviación Estándar de la Sensibilidad Derecha y Rendimiento de la Segmentación para los distintos tamaños del vehículo objetivo en las secuencias: (a) Niebla suave; (b) Niebla densa y lluvia; (c) Día muy nublado y (d) Lluvia. 121

**Figura 5.10.** Secuencias con distinto tipo de vehículo objetivo: (a) Furgoneta con caravana; (b) Camión; (c) Moto. 122

**Figura 5.11.** Media y la Desviación Estándar de la Sensibilidad Derecha y Rendimiento de la Segmentación para los distintos tamaños del vehículo objetivo en: (a) Furgoneta con remolque; (b) Camión; (c) Moto. 123

**Figura 5.12.** Proceso iterativo realizado para la obtención de la posición del vehículo objetivo en cada imagen de la secuencia. 125

**Figura 5.13.** Resultados de  $C_s$  cuando se utiliza el cálculo del centro de masas basado en densidades. 127

**Figura 5.14.** Resultados del seguimiento sobre la secuencia real. 127

**Figura 5.15.** Resultados de la Calidad del Seguimiento para: (a) Velocidad entre 5-10 m/s; (b) Anochecer; (c) Día nublado; (d) Mucha niebla y lluvia; (e) Remolque; (f) Moto. 128

**Figura 5.16.** (a) Transformación de las coordenadas; (b) Resultado del remapeo sobre diferentes imágenes de una de las secuencias de adelantamiento. 130

**Figura 5.17.** Transformación sobre las coordenadas de los píxeles de la imagen. 131

**Figura 5.18.** Transformación de la velocidad por la perspectiva. 132

**Figura 5.19.** Calidad del Seguimiento usando: (a) Canales de velocidad; (b) Remapeo Espacial de la imagen. 132

**Figura A.1.** Escena real e imagen de bordes que contiene el resumen de la información ambiental; Dispositivo completo de ayuda a la baja visión. 148

**Figura A.2.** Superposición de la información dinámica sobre la visión residual. 150

---

---

## ÍNDICE DE TABLAS

---

---

	PÁGINA
<b>Tabla 3.1.</b> Comparativa en el consumo de recursos en la extracción de bordes mediante LIP-Sobel con y sin el uso de un <i>core</i> para realizar la división, y con y sin el uso de los multiplicadores embebidos.	61
<b>Tabla 3.2.</b> Aumento en el consumo de recursos con el incremento en el rango de velocidades en la etapa de selección de la velocidad.	61
<b>Tabla 3.3.</b> Coste del procesamiento de las diferentes etapas descritas.	64
<b>Tabla 3.4.</b> Coste del procesamiento de los dos sistemas propuestos.	64
<b>Tabla 4.1.</b> Porcentaje de Nivel de Clasificación (NC) para los distintos valores del tamaño de las áreas de influencia y el umbral, para toda la secuencia bajo estudio.	92
<b>Tabla 4.2.</b> Porcentaje de ruido de fondo debido a la vibración de la cámara, que se percibe como movimiento en objetos estáticos del entorno.	93
<b>Tabla 5.1.</b> Recursos de hardware consumidos por el sistema de seguimiento cuando se implementa en el chip Virtex-II XC2V3000.	126
<b>Tabla 5.2.</b> Resultados comparados de la aplicación de ambos métodos en cuatro secuencias distintas	133
<b>Tabla 5.3.</b> Recursos de hardware consumidos por el módulo de Remapeo Espacial de la imagen cuando el diseño se implementa en un chip Virtex-II XC2V1000.	134





---

## **CAPÍTULO 1:**

# **Introducción**

---

En este capítulo se introduce la motivación y los objetivos perseguidos con este trabajo, que constituye un ejemplo de la tendencia actual de la Ingeniería Neuromórfica, cuyos mayores elementos motivadores son la necesidad de procesamiento en tiempo real para ciertas aplicaciones. En ese contexto se están desarrollando dos proyectos Europeos y otro Nacional en los que se persigue la extracción de características visuales de bajo nivel (como movimiento) para su utilización en tareas que requieren procesamiento en tiempo real. Este trabajo se ha realizado en el marco de dichos proyectos. En este capítulo también se incluye un esquema de la estructura de esta memoria. Finalmente se introducen los principios biológicos del procesamiento del sistema visual de los insectos, en el que fundamentalmente se ha centrado este trabajo.

## 1.1. Introducción

Existe un interés creciente por construir máquinas inteligentes, y aunque no todos los sistemas inteligentes se basan en principios biológicos, la rapidez, robustez y precisión de éstos ha llevado a la necesidad de estudiar el cerebro para determinar los procesos biológicos que permiten la eficiente extracción y procesamiento de la información. La *Ingeniería Neuromórfica* es un campo de investigación que trata del diseño de sistemas artificiales de computación que utilizan propiedades físicas, estructuras o representaciones de la información basadas en el sistema nervioso biológico.

El cerebro humano es uno de los sistemas más complicados que se conocen. Contiene en torno a  $10^{10}$  neuronas en el cerebro, cada una recibiendo conexiones de cientos de otras neuronas. Hans Moravec [MOR90], basándose en consideraciones sobre las células nerviosas del ojo humano, y extrapolándolas según los datos conocidos sobre anatomía cerebral, estableció que una máquina artificial capaz de ejecutar 10 billones de operaciones por segundo (ops) sería lo suficientemente potente como para albergar una mente parecida a la humana. También que esta máquina debería tener una memoria de  $10^{15}$  bits para ser capaz de almacenar la información contenida en las  $10^{14}$  sinapsis del cerebro humano. Y predijo que antes del 2010 se podría construir una máquina de 10 Teraops por 10 Millones de Dólares USA.

Las células de la retina del ojo procesan la información luminosa que les llega, esta es transportada al cerebro a través del millón de fibras que componen el nervio óptico. En concreto la fóvea, aunque su extensión es menor del 1% del campo visual, genera información que ocupa la cuarta parte de la capacidad del nervio óptico. La resolución máxima de la fóvea se puede comparar a un sistema artificial con 500x500 píxeles, pero su capacidad de procesamiento es relativamente lenta, aproximadamente de 10 imágenes por segundo. Basándose en la capacidad de cómputo que requieren algunos programas de detección de moviendo y reconocimiento existentes, que necesitan de unos 100 cálculos por píxel, una imagen de 500x500 píxeles requeriría de unos 25 millones de operaciones. Por tanto, Moravec calcula que para el millón de fibras del nervio óptico esto supone una capacidad de millones de cálculos por segundo ( $10^9$  ops). Extrapolando estos resultados al cerebro, que posee 1.000 veces más neuronas que la retina, con un grado de complejidad 10.000 veces mayor, la capacidad de cómputo de este debe ser del orden de  $10^{13}$  ops (10 Teraops).

Con respecto a la memoria, Moravec se apoya en los resultados de ciertos experimentos biológicos que demuestran que el aprendizaje se debe a cambios químicos que alteran de forma duradera la capacidad de sinapsis de las neuronas. Si suponemos que la capacidad de memoria de cada sinapsis es equivalente a una palabra de 10 bits, un ordenador con una memoria de  $10^{15}$  bits tendría una capacidad equivalente a las  $10^{14}$  sinapsis del cerebro humano. Esto supone una capacidad de  $1,25 \cdot 10^{14}$  Bytes (125 Terabytes).

Con la tecnología actual, un macroordenador con tales características será en realidad un conjunto de 196 ordenadores interconectados, cada uno del tamaño de un frigorífico.

Ocupará una superficie de 827 m<sup>2</sup> (casi el tamaño de dos pistas de baloncesto) con un peso total de 197 toneladas. Por su parte, el cerebro humano medio apenas ocupa unos 900 cm<sup>3</sup>, con un peso de 1.5 Kg. Aparte de su movilidad, otras ventajas, tales como su habilidad para "pensar" y "crear" son cuestiones largamente debatidas por los partidarios y detractores de la Inteligencia Artificial.

Con anterioridad, Turing [TUR50] había realizado una aproximación al tema utilizando el mismo criterio de capacidad neuronal del cerebro: *"Estimates of the storage capacity of the brain vary from 10<sup>10</sup> to 10<sup>15</sup> binary digits. I incline to the lower values and believe that only a very small fraction is used for the higher types of thinking. Most of it is probably used for the retention of visual impressions, I should be surprised if more than 10<sup>9</sup> was required for satisfactory playing of the imitation game, at any rate against a blind man. (Note: The capacity of the Encyclopaedia Britannica, 11th edition, is 2X10<sup>9</sup>) A storage capacity of 10<sup>7</sup>, would be a very practicable possibility even by present techniques"* (1950).

A la hora de comparar los sistemas biológicos con los artificiales es necesario realizar ciertas consideraciones adicionales. Por un lado, los sistemas biológicos procesan la información de forma relativamente lenta, sin embargo el procesamiento que realizan es masivamente paralelo, y las conexiones neuronales se establecen de forma tridimensional. Sin embargo los sistemas artificiales basados en silicio son dispositivos muy rápidos (la frecuencia de procesamiento de un ordenador de sobremesa ya está por encima de 1GHz), pero el procesamiento se realiza de forma secuencial, y las conexiones sólo se pueden realizar de forma bidimensional y muy limitada.

### 1.1.1. Motivación

De entre todos los sentidos, la visión es uno de los más importantes para la supervivencia, y una gran proporción de nuestro cerebro se dedica al procesamiento de la información visual. Aunque son muchos los avances de las últimas décadas, los sistemas computacionales inteligentes aún no son capaces de "ver" (realizar la "percepción visual"), entendiendo por ver el proceso extracción de información de una imagen o conjunto de imágenes para realizar actividades tales como identificación y localización de objetos, navegación, etc. En definitiva, traducir la imagen a información útil.

El proceso de conversión de imágenes a información es complejo y requiere la utilización de algoritmos de alto nivel hasta llegar a la etapa de "interpretación de la escena". No obstante, estos algoritmos se basan en características de bajo nivel (como movimiento o profundidad) que son computacionalmente muy costosas.

El movimiento es un aspecto particularmente importante en el entorno visual, y puede proporcionar información de la detección de objetos, la segmentación de la imagen, la percepción de la profundidad y la percepción del propio movimiento o ego-movimiento.

Existen distintos modelos que permiten la extracción a bajo nivel de primitivas de movimiento en secuencias de imágenes, inspirándose en mayor o menor grado en los sistemas biológicos.

Los modelos de detección de movimiento *basados en energía* se inspiran en los mamíferos [ADE85]. En estos modelos, se combinan filtros paso-alto espaciales y temporales para crear campos receptivos espacio-temporales no separables. Las salidas de dichos filtros son elevadas al cuadrado y sumadas para generar las respuestas selectivas a la dirección.

Otro esquema de detección de movimiento son los llamados *modelos de gradiente*, que calculan la velocidad en la imagen mediante el cociente de las derivadas temporales y espaciales en cada punto de la imagen [MAR81, HIL87]. En su forma teórica, los modelos de gradiente proporcionan una medida inequívoca y directa de la velocidad en la imagen, en particular, la salida es independiente de la estructura espacial de la imagen y del contraste local en la misma. Sin embargo, existen evidencias experimentales muy débiles de la presencia de los modelos de detección basados en gradiente en los sistemas biológicos [BUC84b, HIL87].

Aunque se los considera animales con un cerebro “simple”, los insectos exhiben un comportamiento visual que rivaliza con los vertebrados en su sofisticación [LAN97b] y es un buen ejemplo de un sistema visual sencillo pero muy eficiente. En el cerebro de las moscas hay unas  $10^5$  neuronas, de las cuales aproximadamente el **70%** se dedica al procesamiento de la información visual [STR76]. En la mosca existe un subconjunto de neuronas que son fácilmente accesibles a través de microelectrodos, y las mejor caracterizadas como neuronas sensibles al movimiento en cualquier sistema visual, por ello, en los últimos 40 años, la mosca se ha convertido en un modelo popular para el estudio del procesamiento visual del movimiento. El procesamiento visual temprano en los insectos, y en la mosca en particular, son bien conocidos, y siguen un modelo basado en correlación (detector de Reichardt) para la detección de movimiento local.

Los detalles de la detección de movimiento basado en modelos de energía son muy distintos de los esquemas basados en correlación, sin embargo, se ha demostrado [VAN85] que ambos modelos son funcionalmente equivalentes al *Detector Elaborado de Reichardt* (un detector algo más complejo que el Detector Elemental de Reichardt), esto es, con los filtros espaciales y temporales apropiados la salida de ambos algoritmos para una entrada dada es idéntica. Por ello, los modelos basados en energía se consideran a veces una variación del esquema basado en correlación. La mayor diferencia entre ambas aproximaciones es la secuenciación y naturaleza de las operaciones que se proponen en ambos modelos.

Así mismo, trabajos recientes sugieren que los modelos de correlación y de gradiente se pueden considerar como extremos opuestos de un continuo de estrategias óptimas de detección de movimiento. Esta aproximación sugiere que el sistema visual puede alterar el algoritmo de detección que utiliza dependiendo de la relación señal-ruido de las señales de entrada [POT94].

En este contexto se han desarrollado en nuestro grupo de investigación dos proyectos europeos y otro de ámbito nacional, en el marco de los cuales se ha realizado este trabajo:

- ECOVISION: Artificial vision system based on early cognitive cortical processing (FP5-IST-2001-32114). (01-01-2002 hasta 30-12-2004).
- DRIVSCO: Learning to emulate perception action cycles in a driving school scenario (FP6-IST-2006-016276-2). (01-02-2006 hasta 31-01-2009).
- DEPROVI: Diseño de sistemas empotrados para procesamiento de visión en tiempo real. Aplicaciones en medicina, vehículos y robots. (DPI2004-07032). (01-12-1004 hasta 30-11-2007).

Uno de los principales objetivos de los tres proyectos es utilizar el conocimiento del modo de funcionamiento de sistemas biológicos de visión para construir un sistema de extracción de primitivas visuales en tiempo real. Se han explorado diferentes modelos. Este trabajo se ha centrado en la exploración de modelos basados en el sistema visual de los insectos.

### **1.1.2. Sistemas electrónicos en tiempo real**

En los modelos de detección de movimiento la estructura visual de las escenas puede ser muy compleja y es difícil extraer rasgos relevantes de forma robusta, por ello se han evaluado generalmente realizando el procesamiento de secuencias *off-line*, debido a la gran carga computacional que requieren. Sin embargo, en algunas ocasiones se precisa de computación en tiempo real, concretamente en tareas de exploración activa como la navegación guiada por un sistema de visión, o tareas de monitorización y alerta en tiempo real. En el marco de los proyectos mencionados antes se han planteado aplicaciones en las que la necesidad de procesamiento en tiempo real se hace patente.

La investigación en este campo sólo es accesible cuando se dispone de la tecnología capaz de realizar el procesamiento de imágenes en tiempo real. El *hardware reconfigurable* (FPGA – Field Programmable Gate Array) es una buena opción para el prototipado e implementación de sistemas neuromórficos en el área de la visión artificial, ya que esta tecnología hace posible definir arquitecturas de gran complejidad e implementar sistemas que requieran mecanismos de adaptación, plasticidad y aprendizaje a medio-largo plazo, además adquiere especial importancia por su capacidad de poder procesar imágenes en tiempo real. Además es una alternativa con buena perspectiva de transferencia a la industria ya que la

tecnología basada en dispositivos reconfigurables reduce el *Time-to-Market* (Tiempo de salida del producto al Mercado), ya que es fácil cambiar los diseños, y adaptar los sistemas disponibles.

Durante este trabajo desarrollaremos arquitecturas de propósito específico para la extracción robusta de movimiento y propiedades de cambios de contraste locales. Para ello diseñaremos caminos de datos finamente segmentados. En las tareas de visión de bajo nivel el tipo de procesamiento que hay que realizar es muy regular (fundamentalmente basado en operadores de convolución) y denso (a nivel de píxeles). Para cada estimación, además debe calcularse un valor de confianza (por ejemplo basado en la estructura local de la imagen). Este flujo regular de datos hace que arquitecturas de procesamiento con segmentación de grano fino sean especialmente apropiadas. Además se hace uso del paralelismo intensivo propio de los circuitos FPGA.

Para la definición de los circuitos FPGA se ha utilizado Handel-C [CEL06] como lenguaje de descripción de hardware de alto nivel. Este lenguaje está basado en ANSI C, y por tanto permite un gran nivel de abstracción y una descripción puramente algorítmica. Además, en comparación con otros lenguajes de descripción de hardware más utilizados como VHDL, se pone de manifiesto que el consumo de recursos no es excesivo [ORT06].

### **1.1.3. El estado de la cuestión**

Para reducir la gran carga computacional de los modelos de extracción de movimiento actuales muchos autores han desarrollado y utilizado modelos basados en la extracción de características dispersas [TOM91, SHI94, GUY96, KRU02].

Por otro lado, la necesidad de tiempo real ha llevado a la implementación de circuitos que lo hagan posible. Aprovechando las ventajas de la tecnología VLSI (Very Large Scale Integration), se pueden fabricar chips de silicio con millones de transistores que procesan la información en paralelo y con gran eficiencia. Por ello, en la última década se han desarrollado muchos sistemas analógicos VLSI de visión basados en sistemas biológicos [DEL93, ETI96, SAR96, MOI97, HAR98].

También en la última década muchos autores han desarrollado robots móviles con sistemas visuales bio-inspirados [FRA92, LEW98, HAR00, ITT03].

Con la tecnología actual ya es posible el procesamiento en tiempo real de modalidades visuales específicas como movimiento [DIA06] implementando arquitecturas de procesamiento de propósito específico (por ejemplo utilizando dispositivos FPGAs) [NII04, DAR06, MAR05] o utilizando las primitivas de computación paralelas de que disponen los procesadores de propósito general [FOR04, BRU05] o incluso las tarjetas gráficas de los PCs [GON05].

Nosotros hemos validado el sistema de procesamiento de movimiento en el marco de una aplicación industrial: monitorización de adelantamiento.

Sistemas de monitorización de adelantamientos similares al propuesto en este trabajo ya están siendo evaluados por distintas empresas del sector automovilístico [VOL06, FIC06]. Aunque no se han publicado datos de sus prestaciones.

## 1.2. Objetivos del trabajo realizado

El objetivo fundamental de este trabajo es la implementación de un sistema de procesamiento de movimiento en tiempo real.

El planteamiento bio-inspirado de los esquemas de procesamiento que proponemos es un elemento diferenciador. Actualmente son muy pocos los sistemas en el mercado que tienen como base una hipótesis de estudio y emulación de sistemas biológicos. Para ello se desarrollará una arquitectura de procesamiento eficiente en dispositivos FPGAs. Este trabajo plantea objetivos concretos a medio plazo (detallados abajo) dentro del objetivo más amplio que es la emulación de sistemas de procesamiento biológicos y su implementación eficiente en hardware reconfigurable para su utilización en aplicaciones reales.

Las necesidades que se pretende satisfacer son: tiempo real, bajo coste computacional y desarrollo de una plataforma de procesamiento a medida.

Los objetivos concretos de este trabajo de investigación son los siguientes:

- Evaluación de las propiedades y esquemas de procesamiento en el sistema visual de los insectos.
- Desarrollo de un algoritmo de procesamiento eficiente de evaluación de movimiento inspirado en los sistemas de visión de los insectos.
- Evaluación de distintos esquemas de procesamiento utilizando los recursos de paralelismo de los dispositivos FPGAs.
- Evaluación de sus prestaciones como sistema embebido de procesamiento de imágenes en tiempo real.
- Aplicación en sistemas de ayuda a la conducción de automóviles. Sistema de monitorización automático de adelantamientos. Caracterización de prestaciones en este campo de aplicación.
- Adaptación del sistema para otras aplicaciones como dispositivos de ayuda a baja visión.

Puesto que el objetivo es emular el comportamiento del sistema visual en los insectos, en la Sección 1.3 de este capítulo, se presenta una revisión de las principales características de dicho sistema visual. El Capítulo 2 se dedica a la descripción del algoritmo de estimación de movimiento bio-inspirado objeto de este trabajo.



Un requisito del sistema es su funcionamiento en tiempo real, por lo que se utiliza tecnología basada en dispositivos lógicos programables (FPGAs). Como se ha comentado con anterioridad, estos dispositivos se caracterizan por su alto poder computacional si se hace uso intensivo de sus recursos de paralelismo. En el Capítulo 3 se describe la arquitectura de camino de datos microsegmentada que se ha diseñado, además se evalúan las prestaciones de los circuitos diseñados en cuanto a consumo de recursos y velocidad de procesamiento.

El Capítulo 4 se dedica a la evaluación de los resultados obtenidos por el algoritmo cuando se aplica en secuencias reales. La evaluación con secuencias reales es muy compleja, ya que no se tiene información acerca de la velocidad y dirección del movimiento de cada píxel de la imagen con la que poder comparar los resultados obtenidos por el algoritmo, por ello se ha diseñado un procedimiento de evaluación específico con secuencias reales.

Una vez comprobada la bondad de los resultados obtenidos con el algoritmo de procesamiento vamos a ilustrar la potencialidad de los resultados de movimiento extraídos, usando la arquitectura de estimación de movimiento diseñada a la detección de adelantamientos de vehículos durante la conducción (Capítulo 5). Para resolver completamente la aplicación será necesario introducir una capa de post-procesamiento de la información de movimiento para realizar el seguimiento de los vehículos, y otra de pre-procesamiento para paliar los efectos producidos por la perspectiva sobre la velocidad de los píxeles.

Finalmente, en el Apéndice se define otra aplicación, un sistema de ayuda para pacientes con Baja Visión, que está siendo evaluado para su uso.

## **1.3. El sistema visual de los insectos**

Los detalles de la óptica, anatomía y fisiología del procesamiento visual temprano en la mosca son bien conocidos [NIL89, STR89, LAU89]. El órgano de visión más especializado en los insectos es el ojo compuesto. Se encuentra a cada lado de la cabeza en los adultos así como en las formas inmaduras conocidas como "ninfas". No es una forma evolutiva reciente sino que en los registros fósiles de trilobites y artrópodos, de hace más de 500 millones de años, se comprobó que ya poseían ojos compuestos.

El ojo compuesto se forma de la repetición de unas unidades hexagonales llamadas "omatidias" o "facetras", que cubren aproximadamente el 86% del campo visual, unos 310° [HUB98]. El número total de omatidias varía mucho entre los insectos, algunas obreras de hormigas pueden tener de seis a siete omatidias, la mosca doméstica tiene 4.000, los escarabajos acuáticos de la familia Dytiscidae tienen 9.000, las mariposas tienen de 2.000 a 27.000 y las libélulas de 10.000 a 30.000.

De esta forma el ojo compuesto divide el conjunto de datos visuales en muchos canales de entrada paralelos, y las neuronas individuales en cada columna son responsables de representar y procesar la información de un solo "píxel" de la imagen visual. Cada una de las omatidias del ojo capta una diminuta porción de la imagen que se integra posteriormente

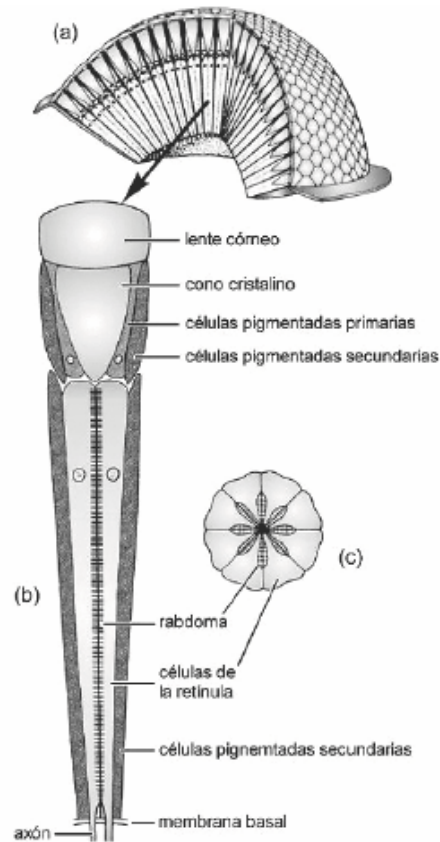
como si se tratara de un mosaico (en lugar de la idea falsa de que los insectos perciben cientos de imágenes idénticas, cada una recibida por una de las facetas).

Una omatidia se compone de una lente córnea usualmente convexa, debajo se halla un cono cristalino que enfoca la luz hacia la parte distal del rabdoma, el cual se encuentra rodeado por varias células retinulares que contienen un pigmento visual empaquetado en unas estructuras llamadas microvili. Cuando la luz llega al rabdoma, el pigmento visual cambia su configuración molecular ocasionando un cambio en el potencial eléctrico de la membrana celular. La señal generada es transmitida vía sinapsis hacia centros de mayor integración del cerebro. Esta organización en columnas se mantiene a lo largo de las tres estructuras que componen el tracto visual del sistema visual de los insectos. En la Figura 1.1 se puede observar la anatomía de una omatidia, así como su disposición dentro del ojo compuesto de los insectos. La luz que captura cada omatidia se proyecta sobre 8 fotorreceptores.

Se ha encontrado que numerosos grupos de insectos tienen varios tipos de células receptoras con diferentes niveles de sensibilidad al espectro de luz dentro de la omatidia. Esto implica que cada una de ellas muestra una respuesta apropiada a diferentes longitudes de onda. Muchos insectos tienen una capacidad visual en donde intervienen hasta cinco tipos de receptores de color y a esta condición se le llama "visión pentacromática". En cambio el ojo humano solo capta la imagen con tres tipos de receptores cromáticos, lo cual demuestra que los insectos en realidad tienen una mayor capacidad de "apreciar" los colores, además de su conocida capacidad para detectar luz en el espectro del ultravioleta.

El ojo de la mosca actúa como un filtro paso-bajo. La resolución espacial del ojo compuesto de una mosca es muy pobre en comparación con la del ojo humano. El número de omatidias determina el poder de resolución del ojo compuesto, lo cual se traduce en la habilidad para separar dos líneas de tal manera que se vean distintas una de otra. De manera que los patrones con una resolución espacial de  $\lambda < 2d\phi$ , donde  $d\phi$  representa la apertura angular entre dos omatidias vecinas (para la *Drosophila*  $d\phi = 7.7^\circ$ ) no se pueden resolver apropiadamente. Se obtienen mejores resoluciones cuando  $d\phi$  son menores, lo cual implica un mayor número de omatidias para el mismo campo visual. Sin embargo, esto afecta la característica de transferencia del contraste, o sensibilidad a la luz, porque habrá menos fotorreceptores disponibles para cada omatidia (menos de 8), y por tanto, decrecerá el número de niveles de intensidad que se pueden distinguir de forma fidedigna. La sensibilidad a la luz también está limitada por el pequeño diámetro de cada omatidia. En un día luminoso el diámetro de la pupila humana es 100 veces mayor que el diámetro de las lentes de la mosca. El brillo de la imagen retiniana (por unidad de ángulo visual) es 10.000 veces mayor en el ojo humano que en el de las moscas. En el caso de insectos crepusculares y nocturnos la sensibilidad a la luz es más importante que la resolución en la imagen. En estos insectos se ha modificado ligeramente el diseño óptico, y la luz de varias omatidias se superpone en la retina,

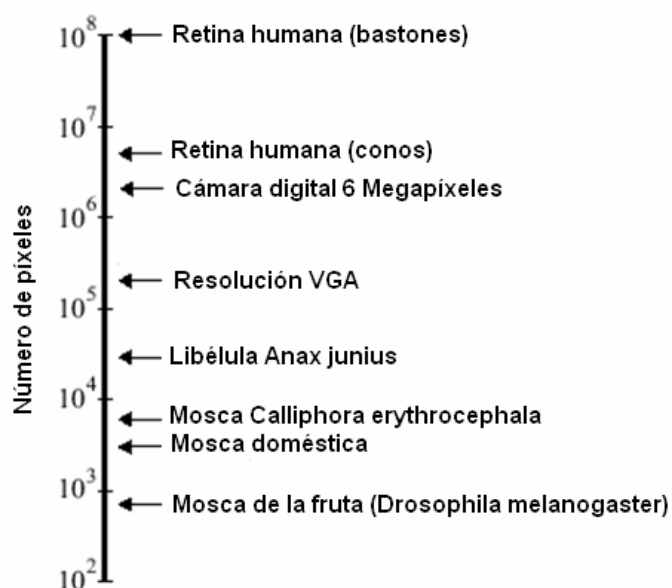
de forma que la sensibilidad es ampliada enormemente. Por ello, los insectos diurnos tienen una capacidad de resolución espacial mayor que los nocturnos.



**Figura 1.1.** Anatomía de una omatidia del ojo compuesto de los insectos: **(a)** Ojo compuesto; **(b)** Omatidia; **(c)** Corte transversal de la omatidia (figura extraída de la página web del Departamento de biodiversidad y biología experimental de la Universidad de Buenos Aires: <http://www.dbbe.fcen.uba.ar/> ).

Debido al tamaño reducido de las facetas la luz que pasa por las lentes es difractada por lo que la imagen que reciben los insectos es borrosa.

En muchos insectos la capacidad de enfoque es casi nula debido a que carecen de visión tridimensional o estereoscópica. A pesar de éstos argumentos se ha encontrado que algunos depredadores como la mantis religiosa o las libélulas sí tienen una visión binocular, que les permite apreciar la distancia a que se encuentra la presa y, por añadidura, el momento oportuno y el alcance del ataque.



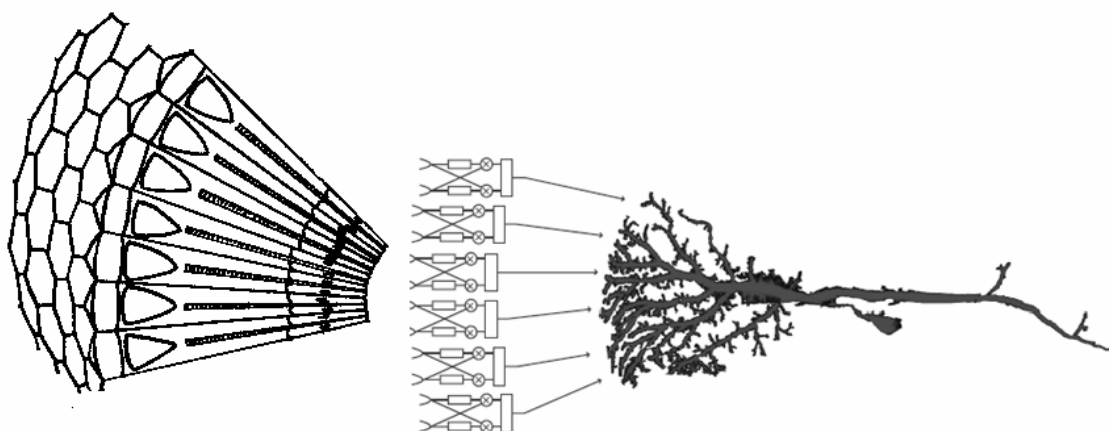
**Figura 1.2.** Comparativa en el número de píxeles en sistemas visuales biológicos y artificiales. Los sistemas visuales actuales tienen resoluciones (en términos de número de píxeles) próximas a la retina humana, sin embargo los insectos voladores tienen varios órdenes de magnitud menos. Todos los datos se han calculado para un único ojo [WAN95, LAN97].

La señal visual procedente de los ojos es procesada en 3 capas neuronales: lámina, médula y complejo lobular (que comprende el lóbulo y la lámina lobular).

La médula es la primera región del cerebro en la que podemos encontrar células sensibles al movimiento [BUC84, BAU92, DEV80, GIL91, DOU95, DOU96, DOU98]. Sin embargo, debido a su pequeño tamaño, estas columnas neuronales se han resistido sistemáticamente a las investigaciones electrofisiológicas. Desde la médula, se realizan proyecciones al complejo lobular. En la mosca, el complejo lobular se puede subdividir en el lóbulo y la lámina lobular [STR70]. Como la médula, el lóbulo contiene una gran cantidad de neuronas sensibles al movimiento y sensibles la dirección, pero de nuevo son tan pequeñas que son difíciles de estudiar y caracterizar electrofisiológicamente.

Nuestra principal fuente de información sobre selectividad del movimiento en el sistema visual de la mosca viene de las células de la lámina lobular, que se encuentra en la parte posterior del cerebro y toma los impulsos de la médula y del lóbulo [STR70]. La lámina contiene un conjunto de neuronas gigantes sensibles a la dirección, que integran la información a lo largo de amplias partes del campo visual. Debido a su tamaño y fácil accesibilidad, estas células tangenciales han sido objeto de estudios tanto anatómicos como electrofisiológicos [STR70]. Han sido identificadas en torno a 50 células tangenciales, todas con campos receptivos muy amplios (algunos de ellos cubren todo el campo visual binocular). Algunas de estas células responden a movimientos amplios ("wide-field cells"), mientras que otras son selectivas al movimiento de objetos pequeños [HAU89]. Células sensibles a la dirección del movimiento similares también se han encontrado en la médula de otros insectos, incluyendo las

polillas [COL70, COL72, MIL93, WIC99], mariposas [IBB91, MAD91], abejas [DEV82, IBB92] y saltamontes [KIE74, RIN90].



**Figura 1.3.** Flujo de información desde la retina hasta la lámina lobular. La respuesta de las células de largo alcance de la lámina lobular de la mosca (representada a la derecha de la imagen) son consistentes con una entrada proporcionada por un conjunto de detectores de movimiento elementales que cubren las células del campo receptivo.

Trabajos anatómicos recientes indican que hay diferencias en el número y la estructura de las células de largo alcance que detectan movimientos amplios (“wide-field cells”) entre las distintas especies de moscas [BUS97]. Los estudios sobre detección de movimiento en la mosca se han concentrado prioritariamente en 3 tipos de células de la lámina lobular: H1, HS y VS.

- a. Las células H1 [HAU76] poseen un árbol dendrítico que cubre todo el campo visual del mismo lado del cuerpo, y se conectan a través del cerebro con la lámina lobular contralateral. Las células H1 son células de *spikes* (o impulsos) que se excitan mayoritariamente por el movimiento amplio en la horizontal desde detrás hacia delante (movimiento regresivo) en el lado del cuerpo que cubren. Las células se inhiben cuando el movimiento es en la dirección contraria.
- b. El Sistema Horizontal (HS) consiste en tres tipos de células llamadas Norte, Ecuatorial y Sur (HSN, HSE y HSS) que se arborizan en las zonas dorsal, media y ventral del ojo respectivamente [HAU82, HAU82b]. Las células HS dan una respuesta gradual en lugar de mediante *spikes* (o impulsos nerviosos), lo cual facilita la interpretación de la respuesta de la célula ante los cambios que se producen. Dicha respuesta va desde la despolarización (cuando se detecta un movimiento amplio, horizontal y progresivo desde delante hacia atrás) hasta la hiperpolarización (cuando se detecta un moviendo en la dirección contraria). Las células HS se proyectan sobre los focos ópticos [HAU93], que son áreas de integración multisensorial que también reciben el impulso de otras áreas del cerebro. Como otras células de la lámina lobular, las células HS son muy accesibles y de gran tamaño, lo cual permite realizar medidas intracelulares estables y prolongadas. Ciertas líneas de investigación indican que estos detectores de flujo óptico están involucrados en la

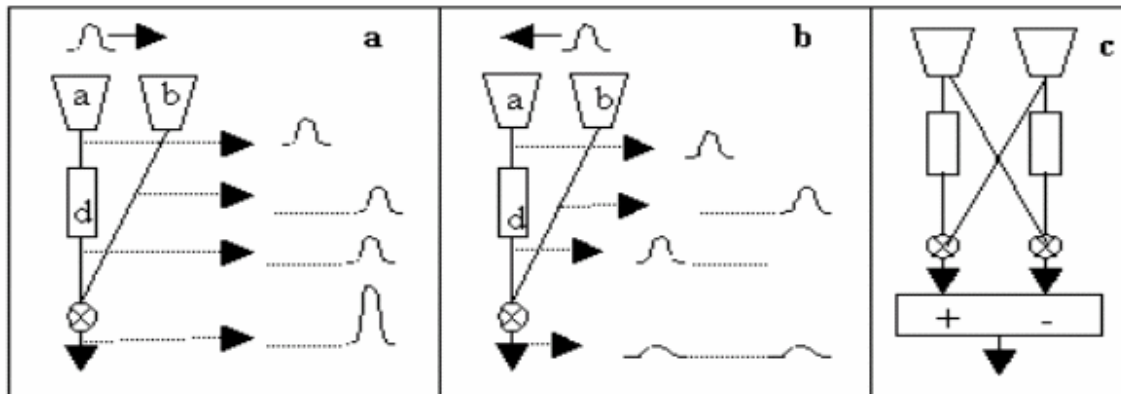
estabilización durante el vuelo, en particular son responsables de arbitrar la respuesta del sistema optomotor de la mosca de modificación de la trayectoria del vuelo. Cuando estas células se quitan mediante disección genética, ablación o micro-cirugía [HEI78, GEI81, HAU83, HAU90] la desviación en el vuelo producida por el sistema optomotor se reduce enormemente. Las células HS también están implicadas en el control inducido visualmente del movimiento de la cabeza [MIL87, STR87].

- c. El Sistema Vertical (VS) lo componen 11 células, llamadas VS1 a VS11 [HEN82b]. Dichas células poseen árboles dendríticos verticales y muy estrechos que se sitúan unos al lado de los otros hasta cubrir todo el campo visual. Al igual que las células HS y VS, se proyectan sobre los focos ópticos. Las células VS tienen una respuesta gradual, que alcanza su máximo de excitación cuando detecta un movimiento vertical, amplio y hacia abajo en el mismo lado del cuerpo [HEN82].

Un gran número de investigaciones indican que la respuesta de las células a la selectividad de la dirección es consistente con un modelo basado en la correlación de detectores elementales de movimiento, DEM (Elementary Motion Detectors) [EGE93]. Este procesamiento de la información mediante circuitos neuronales depende de forma crítica de la implementación de operaciones no lineales. La multiplicación se considera el bloque constructivo elemental que subyace bajo la detección del movimiento por el sistema visual de los insectos [REI87; BOR89]. El primer esquema de correlación para la detección del movimiento fue propuesto por Hassenstein y Reichardt [HAS56] mediante el estudio de la respuesta del sistema optomotor en escarabajos. Podemos entender el algoritmo de detección de movimiento considerando la versión simple ilustrada en la Figura 1.4a. Dos detectores muestrean la luminosidad local o contraste en dos puntos cercanos del campo visual (dos omatidias distintas). Una de las señales es retrasada respecto de la otra y ambas señales se multiplican entre sí. Así, el detector mide la correlación entre las dos señales de entrada recogidas de localizaciones próximas del campo visual muestreadas en tiempos ligeramente diferentes. Si consideramos una característica moviéndose delante de los dos receptores: la característica causará la misma activación en cada canal de entrada, pero con un retraso correspondiente al tiempo que tarda en viajar entre los dos receptores. Para el movimiento en una dirección (la dirección preferente, hacia la derecha en la Figura 1.4a), el retraso introducido por el detector compensará el retraso producido en las señales de entrada. En ese caso, las dos señales alcanzan la etapa de multiplicación a la vez y llegan altamente correlacionadas, así la señal de salida del detector será grande. Para el movimiento en la otra dirección (contraria a la dirección preferente del detector), el retraso introducido incrementa el desplazamiento temporal entre las señales de entrada, así que las señales que se multiplican están pobremente correlacionadas y la salida del detector es pequeña (Figura 1.4b). Un detector como el anterior es sensible al movimiento y selectivo en cuanto a la dirección del mismo.

Se puede mejorar el detector anterior restando a la salida del mismo, la salida de un correlador simétrico al primero (Figura 1.4c). Este detector completo nos dará una salida

positiva para el movimiento en la dirección preferente y negativa para el movimiento contrario [BOR93]. Esta operación de oposición nos asegura que el detector no dará respuesta a estímulos de parpadeo durante los que no hay movimiento.



**Figura 1.4.** Modelo de respuesta de las células a la selectividad de la dirección basado en detectores de movimiento elementales: **(a)** Detección de un estímulo en la dirección preferente del detector; **(b)** Detección de un estímulo en la dirección contraria a la preferente; **(c)** Detector elemental completo.

No todos los DEMs del sistema nervioso son idénticos. Los datos del comportamiento y electrofisiológicos indican que existen varias poblaciones de DEMs con diferentes propiedades espaciales. Cuando la luminosidad es alta, la respuesta de las células parece estar dominada por dos tipos de DEMs: uno toma sus entradas de ommatidias adyacentes, y el otro de vecindades próximas. Cuando los niveles de luminosidad son más bajos los correladores que muestrean ommatidias más alejadas comienzan a ser más importantes [BUC76, PIC79, SCH89].

Aunque las respuestas de las células de largo alcance son consistentes con una entrada basada en el esquema de correlación descrito, el sustrato neuronal de los DEMs sigue siendo poco claro. En particular, no sabemos cómo se llevan a cabo las operaciones de retardo y correlación. La naturaleza y localización de la etapa de resta de los correladores simétricos también permanece bajo controversia: puede tener lugar en las dendritas de las células de campo amplio o con anterioridad durante la transmisión de los datos del movimiento [EGE90, BOR90, BRO96, EGE93, DOU95, DOU96]. El modelo de correlación o una de sus variantes parece ser una estrategia predominante en la detección del movimiento y ha sido encontrada en numerosos sistemas visuales biológicos [BOR93]. En particular, mecanismos basados en correlación, es decir, mecanismos no lineales con interacciones de retardo y comparación entre datos visuales próximos, parecen contribuir en la selección de la dirección del movimiento a varios niveles en el sistema visual de los mamíferos (por ejemplo a nivel retiniano: [BAR65, WYA75, GRZ90, AMT93, GRZ93], a nivel sub-cortical: [STE69, IBB94, WOL94], a nivel cortical: [GOO75, MOV78]), aunque en todos los casos, los detalles de la implementación neuronal y la naturaleza de las interacciones lineales entre las señales de entrada son poco claras.

Un correlador teórico como el de la Figura 1.4c está sintonizado espacio-temporalmente (en el dominio de la frecuencia espacial y en el dominio de la frecuencia temporal). En el dominio temporal la frecuencia óptima (la que produce una salida del DEM de tamaño máximo) viene dada por la duración del elemento de retardo (cuanto mayor sea el retardo menor será la frecuencia temporal óptima). En el caso de la frecuencia espacial, ésta viene dada por la separación entre los brazos del correlador (a mayor distancia entre los receptores, la frecuencia espacial óptima será menor). Es importante indicar, que los correladores no dan medidas de velocidad de la imagen sin ambigüedad. Por ejemplo, la misma velocidad de la imagen se puede especificar utilizando distintas combinaciones de frecuencias espaciales y temporales, cada una de las cuales producirán como respuesta diferentes salidas del DEM. Los correladores también pueden confundir el movimiento en la imagen con el cambio en el contraste de la misma, resultando la detección de velocidades aparentes asociadas al cambio de patrones de intensidad en una imagen. Esta propiedad implica que no es trivial el uso de correladores.

La incapacidad de unos DEMs basados en correlación para indicar la velocidad se ha interpretado como un punto débil del modelo, dado que los datos indican que algunos insectos pueden determinar la velocidad de la imagen sin ambigüedad [SRI91, SRI93]. Algunos de estos escollos se han podido evitar introduciendo pequeñas modificaciones en el modelo de original tales como la introducción de saturaciones no lineales antes de la etapa de correlación [EGE93] o usando múltiples correladores con diferentes óptimos espaciotemporales [SRI99, FIE87]. Zanker [ZAN99] ha sugerido que los DEMs podrían modificarse para codificar la velocidad mediante el desequilibrio de los dos correladores simétricos en la etapa de sustracción.

Se han propuesto muchas otras variaciones sobre el modelo original de correlación, algunas por conveniencia computacional y otras por plausibilidad biológica. El retardo se puede implementar como un retardo fijo [CLI96] o como un filtro paso-bajo [DER86]. En otras versiones, el correlador tiene un filtro paso-bajo en uno de los correladores y otro filtro paso-alto en el correlador simétrico del primero [ZAA78] o incluso filtros paso-alto en ambas ramas del correlador, cada uno con diferente constante de tiempo [ZAA83]. Otra variación es el "Detector Elaborado de Reichardt" [VAN84, VAN85], que incorpora un prefiltrado espacial en ambos canales de entrada.

La operación de multiplicación es otra de las fuentes de variación entre los modelos. Poggio y Reichardt [POG76] establecieron que la operación de combinación de las señales debía ser no lineal. Por ello, la umbralización o la rectificación pueden reemplazar a la multiplicación [Buchner, 1984]. Barlow y Levick [BAR65] propusieron un modelo con una interacción inhibitoria entre los dos canales.

Otros estudios recientes sugieren que las respuestas dinámicas de los fotorreceptores difieren en función del estilo de vida del insecto. Las moscas más acrobáticas y rápidas tienen



fotorreceptores que responden a movimientos más rápidos, permitiéndoles codificar las altas frecuencias contenidas en las rápidas imágenes. Insectos más lentos tienen fotorreceptores con respuestas más lentas [LAU93].

En los siguientes capítulos nosotros nos centramos en el modelo básico de la Figura 1.4. Implementamos poblaciones de DEMs sintonizados a distintas frecuencias espacio-temporales y exploramos esquemas de integración eficiente de los patrones recibidos en estas poblaciones de detectores.

---

## CAPÍTULO 2:

# Algoritmo de detección de movimiento

---

En este capítulo se presenta un algoritmo de procesamiento de bajo nivel para la extracción de movimiento. Dicho algoritmo está basado en la superposición de distintas capas neuronales. La primera capa extrae características que proporcionan información sobre la escena, los bordes. Por tanto, tenemos un sistema de detección de movimiento disperso. La segunda capa utiliza los correladores de Reichardt para determinar cuáles de esas características están en movimiento. Una última capa neuronal, basada en integración de poblaciones, filtra la imagen de ruido. En este capítulo también se describen distintos aspectos que modifican la eficiencia de cada una de estas capas.

## 2.1. Introducción

La extracción de movimiento está basada en los correladores de Reichardt [HAS56], que, como vimos en el capítulo anterior, son la base algorítmica del procesamiento que realiza el sistema visual de la mosca.

El algoritmo de detección de movimiento tiene dos etapas fundamentales: (1) filtrado espacial paso-alto y (2) correlación espacio-temporal de señales. Estas dos etapas son propias de los correladores de Reichardt [HAS56].

El filtrado paso-alto define un sistema disperso, basado en características en movimiento.

Para dotar la salida de una mayor eficiencia en las etapas de post-procesado, propias de distintas aplicaciones, vamos a realizar un filtrado basado en principios biológicos de integración de poblaciones, esa integración proporciona coherencia espacio-temporal y la detección de sólidos rígidos en movimiento.

No hay que perder de vista que esta etapa de diseño previa tiene como objetivo final la implementación en un sistema embebido de tipo FPGA, que podrá afrontar aplicaciones que requieran tiempo real, por ello, ante distintas alternativas de diseño valoraremos positivamente aquellas cuya implementación en dispositivos FPGA se considera más viable.

## 2.2. Extracción de características

Las imágenes en escala de gris contienen una enorme cantidad de información, y en numerosos esquemas de procesamiento la reducción de los datos se incluye como una etapa inicial [KRU00, AGR05, DRU00]

Con objeto de simplificar la escena de la que vamos a extraer el movimiento, el primer paso de nuestro esquema de procesamiento consiste en la extracción de aquellas características que nos describan la escena, y que sean fáciles de identificar en cada una de las sucesivas imágenes de una secuencia.

Un borde es una característica local que indica un cambio brusco en el nivel de gris (alto contraste local) que puede ser una frontera entre un objeto y su entorno, e indica los límites entre objetos superpuestos. Por tanto, son características importantes que nos muestran la estructura de una escena. Lo que significa que si podemos detectar los bordes de una escena con exactitud, los objetos que hay en ella se podrían segmentar. De hecho el

sistema visual humano funciona basándose fundamentalmente en bordes (extracción de la estructura de la escena) [MAR82].

La detección de bordes es una de las áreas de visión por computador a bajo nivel más activa. A pesar del esfuerzo de los investigadores en esta dirección, la detección de bordes en imágenes reales es un problema aún abierto, y al ser la primera etapa sobre la que se sustentan niveles superiores de procesamiento, los resultados finales dependen de lo exacta y eficiente que sea.

Existen muchos procedimientos para obtener los bordes de un objeto. Fundamentalmente difieren en las propiedades geométricas y fotométricas del borde, así como en las propiedades matemáticas y algorítmicas del método [GON92, SHA97].

Ya que un borde se define como un cambio local en el nivel de gris de la imagen, todo operador que sea sensible a estos cambios podrá utilizarse como detector de bordes. Por ejemplo, los operadores derivativos aplicados a una imagen detectan cambios locales en los niveles de gris de la misma, su respuesta es alta cerca del borde y pequeña en áreas de intensidad homogénea. Tales cambios se corresponden con las altas frecuencias de la imagen. Como se indicó en el Capítulo 1, un modelo matemático sencillo (propuesto por Hassenstein y Reichardt (1956) [HAS56]), que es capaz de explicar los datos experimentales del sistema de detección de movimiento de la mosca, propone un filtrado paso-alto antes de una etapa de correlación. Este proceso de filtrado elimina de la salida las componentes continuas, y mantiene los cambios para su procesamiento. Es decir, este modelo propone una extracción de bordes previa a la detección del movimiento, lo que permite extraer movimiento posteriormente, correlacionando estos bordes en imágenes sucesivas de una secuencia.

Puesto que las imágenes son bidimensionales es necesario considerar derivadas direccionales de la imagen (de diversos órdenes en función del detector). En algunos esquemas de procesamiento se usa la orientación del gradiente para localizar los bordes. Según lo anterior, el gradiente (operador de primer orden) de la función imagen,  $I$ , viene dado por la Ecuación (2.1); la magnitud del gradiente viene dada por la Ecuación (2.2), y será un número real, que normalmente se convierte en entero mediante redondeo; y, por último, la orientación del borde se determina a partir de la Ecuación (2.3):

$$\nabla I(x, y) = \left[ \frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right] \quad (2.1)$$

$$|\nabla I| = \sqrt{\left( \frac{\partial I}{\partial x} \right)^2 + \left( \frac{\partial I}{\partial y} \right)^2} \quad (2.2)$$

$$\theta = \tan^{-1} \left( \frac{\partial I / \partial y}{\partial I / \partial x} \right) \quad (2.3)$$

Cualquiera que sea el operador que se use, el resultado del procesamiento contiene información sobre la intensidad del borde en el píxel (su magnitud) y sobre la orientación del mismo.

Como la imagen es discreta, se aproximan las derivadas por la diferencia (en niveles de gris) entre los píxeles de una región de la imagen. Partiendo de esta idea, se pueden usar matrices o máscaras que, al convolucionarlas con un área de la imagen, se comportan como la aproximación al operador derivativo en el píxel correspondiente al centro de la máscara.

Mediante este procedimiento de aproximación discreta a operadores de primer orden se definen numerosas máscaras para el cálculo de bordes: Roberts [ROB65], Prewitt [PRE70], Kirsch [KIR71], Sobel [SOB70, SOB78], entre otros; y la misma idea nos permite aproximar operadores de segundo orden, como por ejemplo el laplaciano [GON92, SHA97].

### **2.2.1. Detección de bordes mediante Sobel: convolución con una máscara cuadrada de 3x3**

Un gran número de investigadores han considerado la cuestión de medir el rendimiento de los detectores de bordes. La principal dificultad radica en el hecho de que realmente no sabemos cuáles son las características que queremos detectar (es decir, los bordes relevantes para cada tarea), y además pueden estar corrompidas por el ruido. A pesar de todo ello, se tienen en cuenta algunos criterios para evaluar el rendimiento de los detectores [ABD78, KIT81, PRA78, KAN94, VEN92].

En general, los operadores de gradiente tienen medidas similares, y sus respuestas se van deteriorando de forma similar de acuerdo con la cantidad de ruido en la imagen. Por tanto, no tiene en general mucha importancia el operador de gradiente seleccionado, y los problemas deben resolverse en niveles superiores de visión.

Para la fase inicial de nuestro modelo de procesamiento hemos escogido, de entre los detectores de gradiente, el detector de bordes de Sobel [SOB70, SOB78].

Según los resultados mostrados en [HAV03] los detectores de Roberts, Prewitt y Sobel son apropiados para indicar la parametrización del movimiento, en comparación con otros métodos habituales como LOG (Gausiana del Laplaciano) o Canny, porque son métodos relativamente rápidos y dan resultados precisos con una tasa de error relativamente baja. El operador de gradiente de Sobel es el que mejores resultados presenta cuando se usa una

máscara pequeña, comparado con Roberts y Prewitt [GON92], de entre otros, como hemos podido comprobar experimentalmente durante la etapa de desarrollo. Además permite obtener los bordes horizontales o verticales de forma independiente, y como veremos, los bordes verticales serán los que usemos para nuestro detector de movimiento.

De cara a la implementación en hardware específico, otra de las ventajas asociadas al uso del operador de Sobel es que se puede integrar fácilmente dentro de una estructura de procesamiento en segmentación de cauce, como veremos en el Capítulo 3. En este sentido, se ha optado por una implementación de un operador derivativo para la extracción de características en detrimento de otras metodologías que obtienen mayor calidad en la detección de bordes aunque su implementación puede ser más compleja.

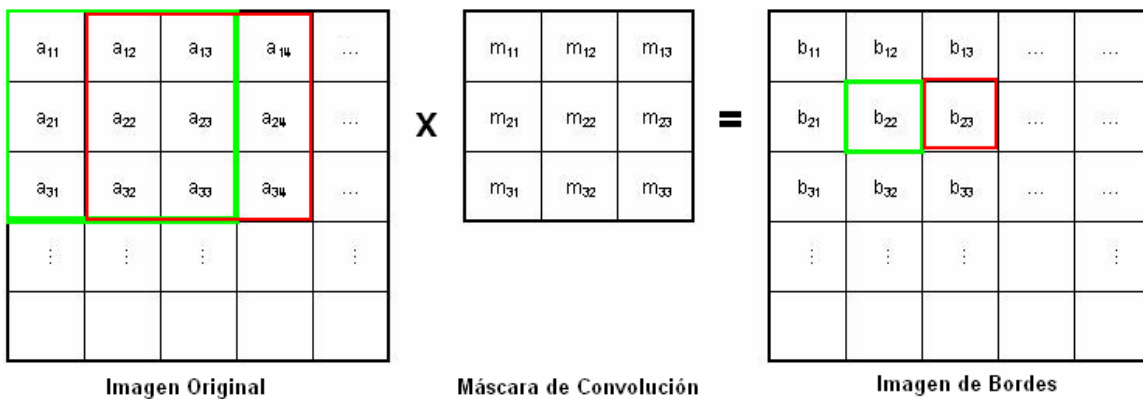
El operador de Sobel se compone de las máscaras 3x3 dadas por las Ecuaciones 2.4 y 2.5.

$$B_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad (2.4)$$

$$B_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad (2.5)$$

Mediante un proceso de convolución de la imagen original con la máscara de la Ecuación (2.4) se obtienen los bordes horizontales, mientras que los bordes verticales se obtienen al convolucionar la imagen con la máscara de la Ecuación (2.5).

El procedimiento de convolución seguido se muestra en la Figura 2.1 mediante un ejemplo. Como se puede ver, con cada convolución se obtiene el valor de la intensidad del píxel (de la imagen de bordes) correspondiente a la posición central del área convolucionada por la máscara. La imagen completa de los bordes se obtiene cuando desplazamos la máscara por la imagen original.



$$b_{22} = (a_{11} \cdot m_{11}) + (a_{12} \cdot m_{12}) + (a_{13} \cdot m_{13}) + (a_{21} \cdot m_{21}) + (a_{22} \cdot m_{22}) + (a_{23} \cdot m_{23}) + (a_{31} \cdot m_{31}) + (a_{32} \cdot m_{32}) + (a_{33} \cdot m_{33})$$

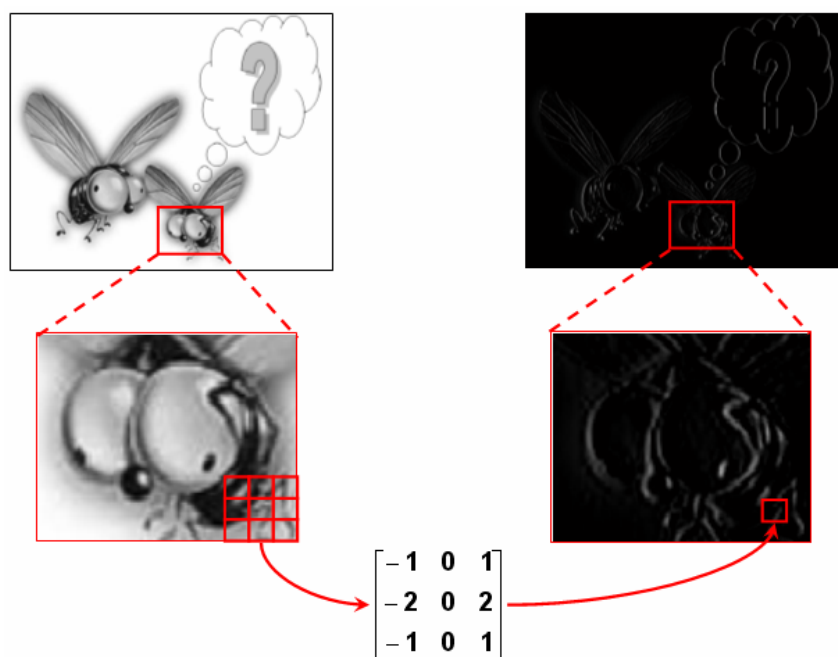
$$b_{23} = (a_{12} \cdot m_{11}) + (a_{13} \cdot m_{12}) + (a_{14} \cdot m_{13}) + (a_{22} \cdot m_{21}) + (a_{23} \cdot m_{22}) + (a_{24} \cdot m_{23}) + (a_{32} \cdot m_{31}) + (a_{33} \cdot m_{32}) + (a_{34} \cdot m_{33})$$

**Figura 2.1.** Procedimiento que se sigue en la convolución para la obtención de los bordes de la imagen.

En la Figura 2.2 se muestran un ejemplo de la aplicación de convolución para la obtención de los bordes verticales.

Los esquemas basados en detectores de gradiente tienen un gran número de problemas asociados, aunque siguen siendo los más usados por la comunidad de visión por computador. Uno de los principales problemas tiene que ver con la robustez del operador ante el ruido, que puede ocasionar la no detección o pérdida de bordes existentes, o la detección de bordes erróneos (falsos positivos). En la Figura 2.3 se muestra un ejemplo de la detección de bordes en una imagen con ruido de 'sal y pimienta'.

Este problema podría resolverse, en cierta medida, mediante el uso de máscaras mayores. Estas máscaras mayores tienen en cuenta los cambios en la intensidad en un área local mayor, utilizan más información contextual, por tanto, los bordes obtenidos serán mejores y más fiables, es decir, menos sensibles al ruido de tipo 'sal y pimienta'.

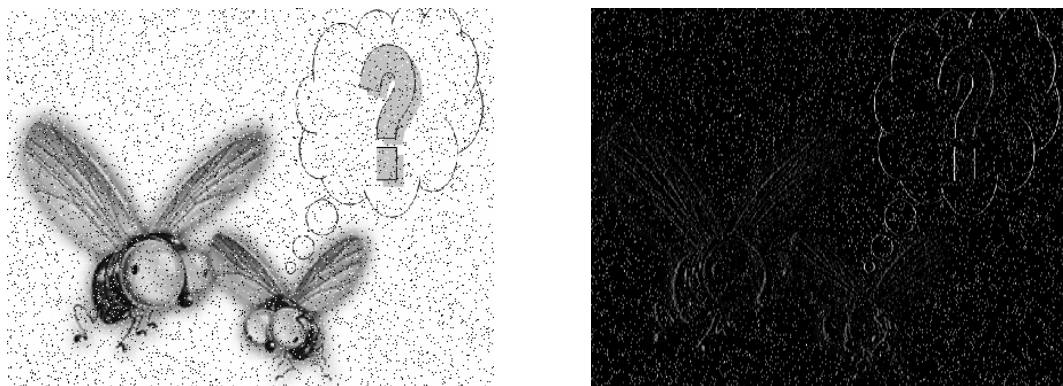


**Figura 2.2.** Resultado de la aplicación de la máscara de Sobel para la obtención de los bordes verticales de la imagen.

En la etapa de desarrollo del modelo hemos probado máscaras de distintos tamaños, pero éstas no suponían mejoras significativas para la etapa de detección del movimiento (véase el Capítulo 4), y sin embargo, el uso de las mismas implicaba un consumo mayor de hardware cuando se implementan como circuitos específicos. Por tanto, se ha optado por el uso de las máscaras de Sobel 3x3 para nuestro sistema.

Los bordes verticales nos ofrecen la información eficiente necesaria para la detección de movimiento horizontal, y por tanto, serán éstos los únicos que utilizemos en nuestro

algoritmo. Un problema asociado a esta elección será la incapacidad del sistema para la detección de movimientos rotatorios o en la dirección puramente vertical, lo cual limitará las aplicaciones para las que pueda estar indicado. En todo caso, la extensión del modelo a otras orientaciones es sencilla, basta con incorporar etapas de extracción de bordes orientados y correladores específicos para cada orientación.



**Figura 2.3.** Resultado de la obtención de los bordes verticales mediante la máscara de Sobel de 3x3 en el caso de una imagen contaminada con ruido de tipo 'sal y pimienta'.

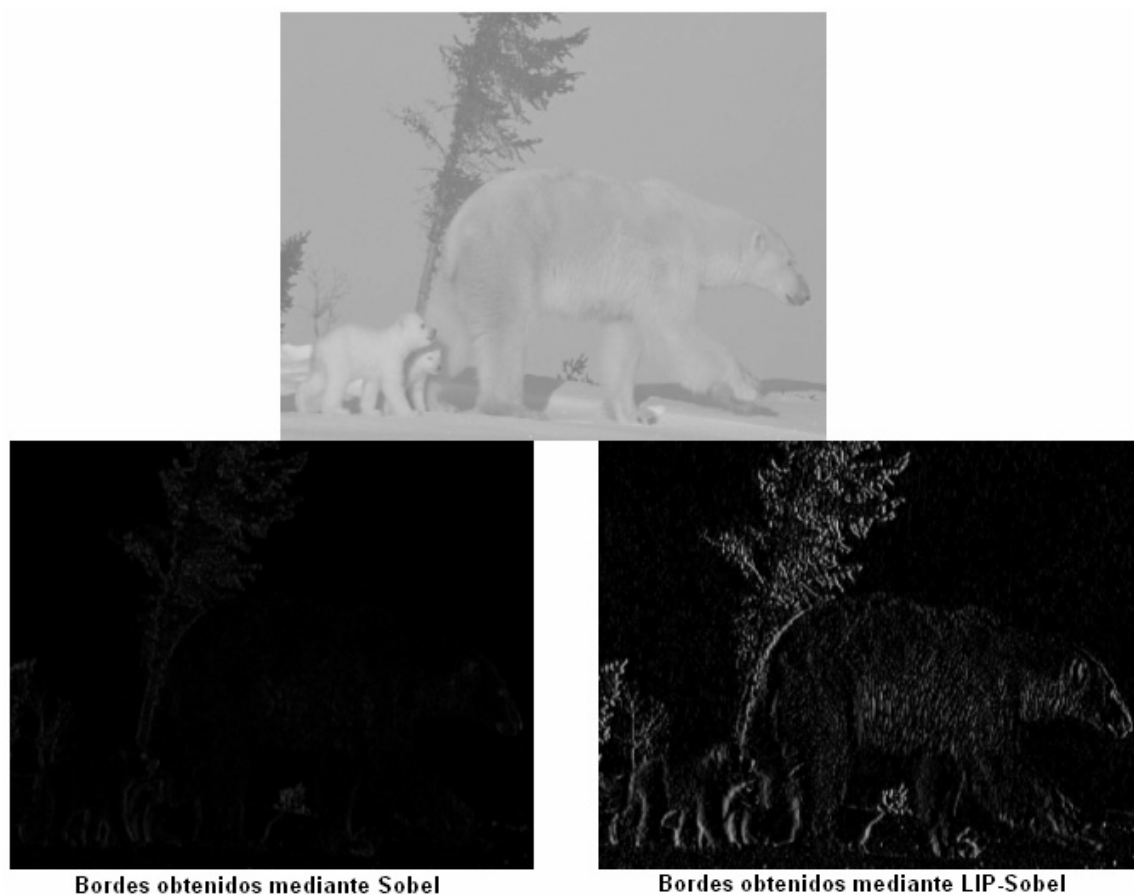
### 2.2.2. Detección de bordes mediante el detector de Sobel en el espacio logarítmico

La iluminación de la escena va a ser un factor determinante en la extracción de los bordes en imágenes; cuanto menos contraste tengamos menos robustos serán los bordes, ya que los cambios locales en la intensidad de los píxeles serán pequeños. Este fenómeno afecta, por ejemplo, a escenas recogidas al amanecer, al anochecer o en espacios irregularmente iluminados.

En condiciones normales de luminosidad, el rango dinámico al que se expone el sensor de la cámara es muy alto, y su respuesta limitada, de manera que partes de la imagen estarán sobresaturadas mientras que otras estarán demasiado oscuras. En ciertas aplicaciones, por ejemplo cuando se requiere una respuesta del sensor similar a la del ojo humano, se tiende al uso de cámaras de alto rango dinámico (HDR- High Dynamic Range). Actualmente hay dos aproximaciones principales para extender el rango dinámico del sensor de imagen. Unos sensores realizan un desplazamiento de la curva de luminosidad de la escena, mientras que otros efectúan una compresión logarítmica de la luminosidad de la imagen [IND03, LAR98, DEB97, KRA05]. El resultado es una imagen en la que no hay zonas totalmente oscuras ni zonas saturadas. Por otro lado, se presenta el problema de que el contraste en la imagen es bajo, limitando el rendimiento de los detectores de bordes. Por todo ello, el interés de los investigadores en este campo se ha dirigido hacia la detección de bordes bajo condiciones de pequeños cambios locales en la iluminación de la escena [ROB77, JOH90].



A mediados de los años 80 se desarrolló la estructura matemática del modelo de procesamiento logarítmico de imagen (LIP- Logarithmic Image Processing). Este modelo describe un espacio en el que se definen operaciones matemáticas (suma, resta, multiplicación, división, diferenciación, integración, convolución, entre otros) para el procesamiento de imágenes cuya intensidad esté restringida en un rango [JOU87, JOU88, PIN87, PIN92]. La obtención de los bordes de una imagen mediante este modelo está ampliamente justificada en los casos que se han expuesto en la introducción de ésta sección.



**Figura 2.4.** Comparativa de los bordes obtenidos mediante la máscara de Sobel descrita en la sección 2.2 y mediante el procesamiento de LIP-Sobel.

En el modelo LIP la intensidad de una imagen viene completamente representada por su función de niveles de gris asociada ( $\tilde{f}$ ). Esta función se define en un dominio espacial no vacío del espacio euclideo  $\mathbf{R}^2$ , con valores en el intervalo de números reales  $[0,M)$ , donde M es positivo (en nuestro caso M es igual a 255 porque trabajamos con imágenes de 8-bits en escala de gris). Esta función de niveles de gris no es más que la función clásica pero evaluada en una escala de intensidad invertida, como se muestra en la Ecuación (2.7).

La operación de convolucionar en el espacio lineal un área de la imagen original, A (Ecuación (2.6)), con la máscara de Sobel dada en la Ecuación (2.5), es equivalente, en el espacio logarítmico, a aplicar el operador LIP-Sobel, L, dado por la Ecuación (2.7) [DEN98].

$$A = \begin{pmatrix} f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{pmatrix} \quad (2.6)$$

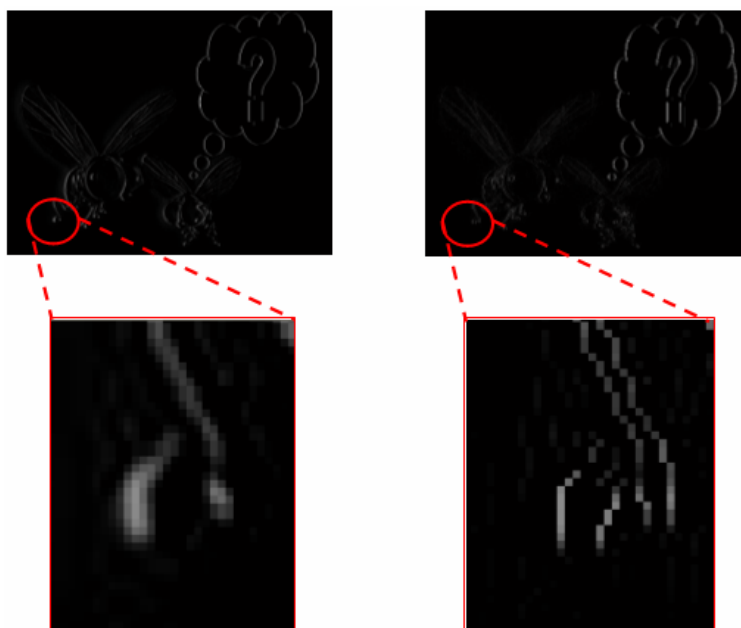
$$L = M - M \begin{pmatrix} \tilde{f}_3 \tilde{f}_6^2 \tilde{f}_9 \\ \tilde{f}_1 \tilde{f}_4^2 \tilde{f}_7 \end{pmatrix} \text{ donde } \tilde{f}_i = M - f_i \quad (2.7)$$

La detección de bordes mediante LIP-Sobel es una técnica robusta bajo condiciones de cambios locales muy pequeños en la iluminación. Como se puede observar en la Figura 2.4, ante una situación de bajo contraste en la imagen, el detector de LIP-Sobel (cuyo resultado se muestra en la parte inferior derecha de la Figura 2.4) se presenta más robusto que el operador Sobel lineal (cuyo resultado se muestra en la parte inferior izquierda de la Figura 2.4), lo que se traduce en un aumento considerable del número de los bordes detectados, así como una mayor intensidad de los mismos.

### 2.2.3. Principio de ‘non maximum suppression’

Si entendemos que un borde es la frontera entre un objeto y el resto del mundo, debería ser una línea fina de un píxel de anchura, sin embargo, normalmente donde tenemos un borde vertical encontramos una línea gruesa en la que hay un píxel de máxima intensidad (el borde propiamente dicho) generalmente situado en el centro de la línea, y a ambos lados del mismo (a derecha e izquierda) otros píxeles con un gradado de los niveles de gris. Por ello, los bordes obtenidos con el detector de Sobel son post-procesados para hacerlos más finos. Para ello usaremos el principio de ‘non maximum suppression’ o de supresión de valores no máximos [SHA97], es decir, vamos a quedarnos con el píxel cuya intensidad sea máxima en la dirección perpendicular al gradiente.

Puesto que la máscara que hemos aplicado nos ha proporcionado bordes verticales, la dirección perpendicular al gradiente es la dirección horizontal. El proceso para hacer más finos los bordes consiste en buscar el máximo de intensidad local en la horizontal del borde. El píxel que lo posea mantendrá su valor, mientras que los píxeles en torno suyo pierden el valor de la intensidad que tuviesen, y toman el valor cero (véase la Figura 2.5).



**Figura 2.5.** Vamos a quedarnos con el píxel cuya intensidad sea máxima (el central), y en la dirección perpendicular al gradiente (en la dirección horizontal) todos los píxeles de un área local de igual tamaño a la máscara de convolución cuyo valor de intensidad sea menos que el máximo tomarán un nuevo valor de intensidad igual a cero (perdiendo el valor que tenían).

#### 2.2.4. Umbralización de los bordes

Uno de los problemas fundamentales en la detección de los bordes es que, en todos los píxeles existe una respuesta a los operadores de gradiente, que será de mayor o menor intensidad. Pero solamente las respuestas de cierta magnitud constituyen ‘auténticos’ bordes. Por tanto, será necesario establecer un umbral sobre los valores obtenidos en la imagen de bordes, para quedarnos sólo con los que consideremos más significativos. La elección del umbral dependerá de las condiciones de iluminación de la escena y del rango dinámico de la cámara utilizada, entre otros factores. En definitiva, dependerá de la aplicación concreta, y no es algo trivial de resolver para un algoritmo de detección de uso genérico.

Las Figuras 2.6 y 2.7 muestran cómo se degrada la imagen de bordes a medida que aumentamos el umbral. Para observar este fenómeno incluso en aquellos píxeles cuya magnitud es muy baja, vamos a asignar un valor de magnitud máxima a aquellos píxeles cuyo valor sea superior al umbral, mientras que a los que no lo superan les asignaremos el valor de cero (binarización de la imagen). Se observa en las Figuras 2.6 y 2.7 que a medida que aumentamos el umbral perdemos un número considerable de píxeles de escasa magnitud, es decir, bordes débiles, detectados como consecuencia del ruido, de bajo contraste local o de otros factores. Cuando el umbral sobrepasa cierto límite perdemos también los píxeles correspondientes al objeto de interés de la imagen, es decir, el oso en el caso de la Figura 2.6.

La Figura 2.6 muestra el comportamiento de la imagen después de la umbralización (y binarización) de los bordes Sobel lineales, mientras que en la Figura 2.7 podemos observar el

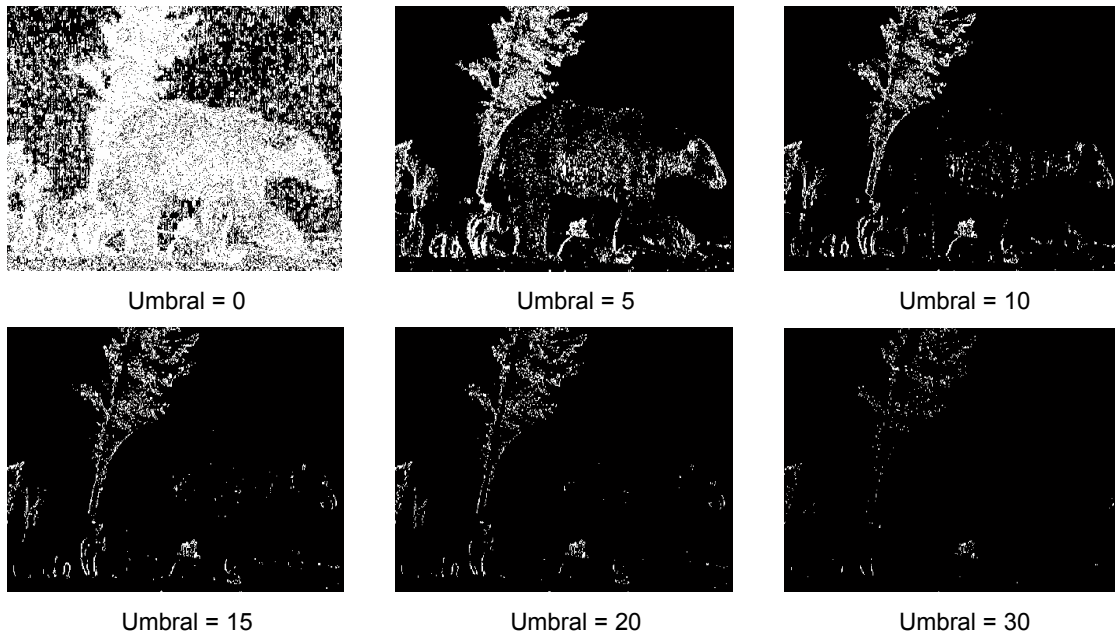
resultado de la umbralización en el caso de los bordes obtenidos mediante el operador de LIP-Sobel. Como se puede observar, otra de las ventajas de la aplicación del operador logarítmico sobre el operador de Sobel lineal, es que el rango de valores que puede alcanzar el umbral es mayor, como consecuencia de que la magnitud de los bordes obtenidos por este medio también es mayor.

En el caso de las Figuras 2.6 y 2.7, tenemos una imagen estática que hemos umbralizado mediante un umbral fijo. Cuando tenemos una secuencia se presenta una dificultad adicional, la luminosidad puede variar entre dos imágenes consecutivas o al cambiar el fondo la magnitud de los bordes de los objetos también cambia. Este fenómeno hace de la umbralización fija un inconveniente, ya que el número de bordes encontrados en dos imágenes consecutivas de la misma secuencia será distinto, y esto puede inducir a errores en etapas posteriores de procesamiento.

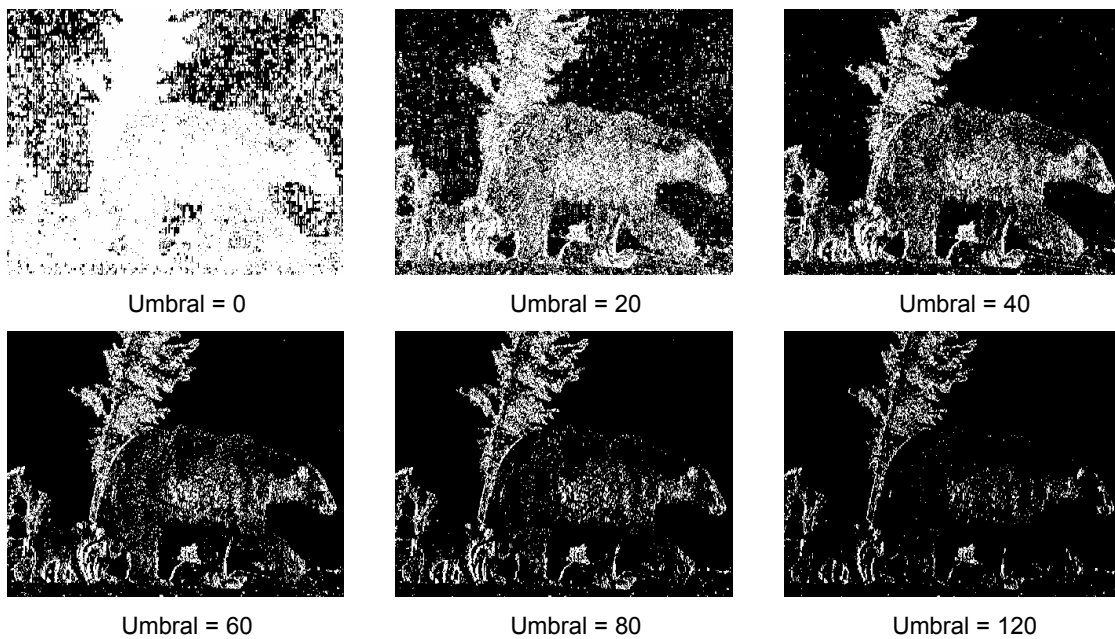
La Figura 2.8 muestra el número de bordes que superan un umbral dado a lo largo de varias imágenes de una secuencia (la secuencia utilizada es la misma que será estudiada con más detalle en el capítulo de evaluación, y que se ilustra en la Figura 2.12a). Como puede verse en la Figura 2.8, la variabilidad en el número de bordes entre imágenes consecutivas puede ser de más de cien píxeles.

Una posible solución a este problema consiste en la introducción de un umbral global variable, tal que el número de bordes a lo largo de las distintas imágenes de la secuencia se mantenga constante. Como se muestra en la Figura 2.9, partiendo de un umbral de 30, se ha tratado de estabilizar el número de bordes obtenidos. Como se puede comprobar en la Figura 2.8, cuando el umbral global fijo era de 30, las variaciones de los píxeles de bordes obtenidos en dos imágenes consecutivas, por ejemplo entre la 1 y la 2, llegaba a superar la cifra de 100, y entre las imágenes 3 y 4 superaba el valor de 50. Con la introducción del umbral variable entre las imágenes 1 y 2 hay una diferencia en el número de bordes que no llega a 10, mientras que entre las imágenes 3 y 4 la diferencia está en torno a 40 bordes. Pero la utilización de un umbral variable [OTS79] puede requerir procesos iterativos difíciles de implementar en una estructura con segmentación de cauce, aunque, dependiendo de la arquitectura concreta, se podría integrar en paralelo al cauce con ciertas limitaciones de funcionamiento.

Como se ha podido comprobar en las Figuras 2.6 y 2.7, la umbralización global afecta fundamentalmente a bordes externos al objeto de interés en la escena. Otra posibilidad, más interesante para la finalidad del algoritmo, podría consistir en utilizar algún criterio, por ejemplo, la información relativa al movimiento, para aumentar el número de bordes en una zona de interés o de atención, mediante el uso de un umbral local variable. De esta forma se podría dar relevancia a la información de una zona con respecto al resto de la imagen. La combinación de los umbrales local y global podría facilitar la detección de movimiento.



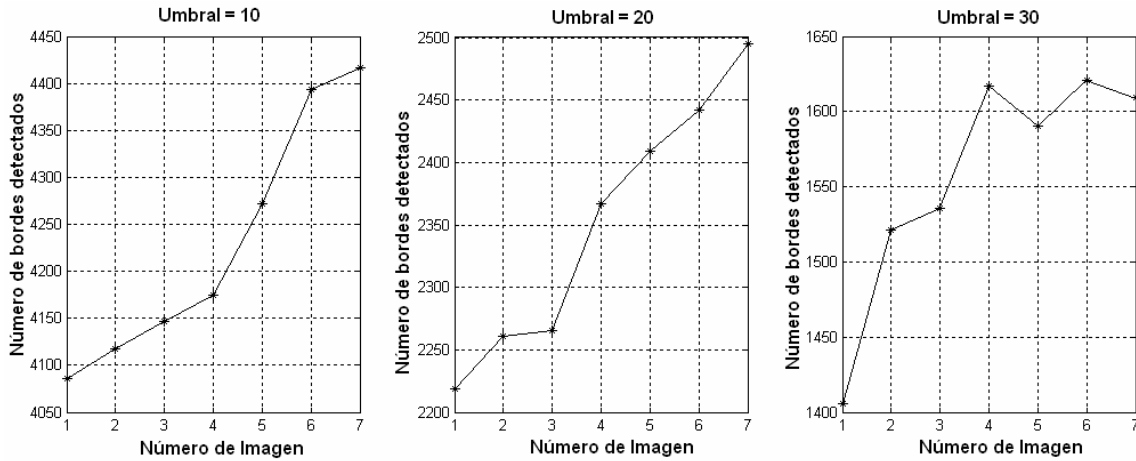
**Figura 2.6.** Umbralización y binarización (sólo para mostrar la cantidad de bordes) de la imagen de bordes obtenida mediante el operador de Sobel lineal.



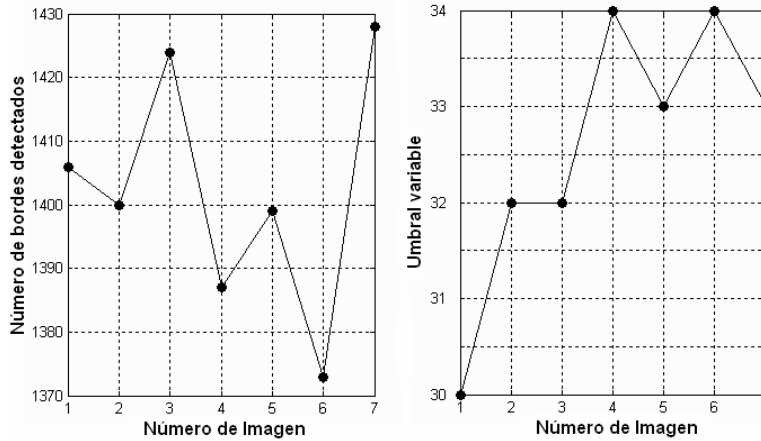
**Figura 2.7.** Umbralización y binarización (sólo para mostrar la cantidad de bordes) de la imagen de bordes obtenida mediante el operador de LIP-Sobel.

La Figura 2.9 se ha realizado adaptando el umbral de forma sencilla. Si el número de bordes aumenta incrementamos el umbral para la siguiente imagen, y viceversa. Es decir, se ha aplicado un sencillo mecanismo de control adaptativo proporcional.

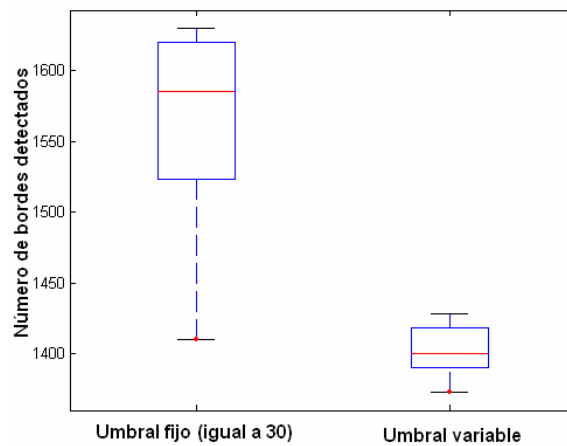
La Figura 2.10 muestra la comparación en la variabilidad en el número bordes detectados con un umbral variable y con un umbral fijo e igual a 30.



**Figura 2.8.** Variabilidad del número de bordes que superan un umbral fijo para el caso de una secuencia (ilustrada en la Figura 2.12a). Cada una de las gráficas muestra la variabilidad para el caso de un umbral fijo diferente: umbral=10 (gráfica de la izquierda); umbral=20 (gráfica central) y umbral=30 (gráfica de la derecha).



**Figura 2.9.** Partiendo de un umbral igual a 30, mediante la introducción de un umbral variable controlamos que el número de bordes a lo largo de la secuencia permanezca constante. La gráfica de la izquierda muestra la cantidad de bordes encontrados en cada imagen de la secuencia, mientras que la gráfica de la derecha muestra el umbral que se ha aplicado en cada imagen para estabilizar el número final de bordes. Para esto se puede seguir un proceso iterativo que ajuste el umbral en varios pasos, con el objetivo de obtener un número de bordes estable.



**Figura 2.10.** Variabilidad del número de bordes detectados con umbral igual a 30 y con umbral variable.

## 2.3. Detección de movimiento: Correladores de Reichardt

Una vez extraídas aquellas características que proporcionan la información de la escena (los bordes verticales en nuestro caso) procedemos a determinar cuáles de ellas se mueven. El movimiento de los bordes puede ser debido, o bien a que pertenecen a un objeto que se mueve por la escena, o a que la cámara que obtiene la secuencia está navegando por el escenario que visualiza, por lo que, incluso los objetos estáticos, tendrán movimiento, es lo que se conoce como 'movimiento propio' (o ego-movimiento).

Para la extracción del movimiento vamos a aplicar correladores de Reichardt [HAS56]. Como se explicó en el Capítulo 1, este esquema concuerda con los datos experimentales extraídos de las neuronas del sistema visual de la mosca, aunque también se ajusta muy bien al comportamiento del sistema visual de otros muchos animales, sobre todo insectos. El algoritmo está basado en la correlación multiplicativa de las señales de entrada a un Detector Elemental de Movimiento (DEM). En el caso de la mosca las dos señales que correlaciona se corresponden con la información detectada en dos de sus *omatidias* (véase el Capítulo 1). Cada *omatidia* se excita sólo por la luz recibida de un pequeño ángulo sólido. Por tanto, dos *omatidias* consecutivas no se excitarán a la vez, sino con un retardo que corresponde al tiempo que tarda la característica móvil (el borde) en entrar en el ángulo sólido del que la *omatidia* extrae información.

En nuestro caso, recibimos la información visual a través de un conjunto de fotorreceptores electrónicos que se excitan unas 30 veces por segundo (que es la frecuencia de captura de una cámara convencional), pero todos a la vez. Lo que vamos a hacer es correlacionar la información correspondiente a dos imágenes consecutivas de la secuencia.

En el caso de la mosca, existen una serie de correladores (DEM) cada uno caracterizado por una constante de tiempo. El correlador que obtiene la máxima respuesta es el que detecta el borde moviéndose, y la velocidad del borde detectado viene dada por el retardo característico de dicho DEM.

En nuestro modelo de detección de movimiento, para emular el esquema de procesamiento de la mosca, vamos a correlacionar cada píxel de la imagen recibida en  $t=0$  con un conjunto de píxeles en la imagen  $t=1$ , como se ilustra en la Figura 2.11. Para nosotros la constante de tiempo característica, viene proporcionada, para una velocidad de captura dada (fps), por la distancia (en píxeles) que existe entre las posiciones de los dos píxeles que se correlacionan, la del píxel en la imagen  $t=0$  y la del píxel en la imagen  $t=1$ .

La Figura 2.11 muestra 5 filas y 12 columnas de dos imágenes consecutivas de bordes. En la Figura 2.11.a se observan los bordes verticales obtenidos de la imagen tomada en  $t=0$ , mientras que en la Figura 2.11.b tenemos los bordes verticales de la imagen tomada en  $t=1$ . Para detectar el movimiento correlacionamos cada píxel de la imagen (a) con los píxeles de un entorno en la imagen (b).

Cuando queremos detectar el movimiento del píxel en la posición correspondiente a la fila 2 y la columna 7, posición que se corresponde con la existencia de un borde, correlacionamos el valor de este píxel con los valores de los píxeles que se encuentran en la fila 2, y entre las columnas 3 y 11 (posiciones enmarcadas en rojo), es decir, con los píxeles en un entorno simétrico (a derecha e izquierda) de la posición del píxel de la Figura 2.11.a, en dicho entorno presuponemos que podría estar la nueva posición del borde en movimiento.

Como realizamos una correlación multiplicativa, la respuesta de cada una de las correlaciones será el resultado de multiplicar el valor de los píxeles de ambas imágenes (uno a uno). Este valor será cero en todos los casos (el valor de los píxeles donde no hay borde es cero) excepto en la correlación con los píxeles en la posiciones correspondientes a las columnas 5 y 8, posiciones en la que hay otros bordes. El píxel en la columna 5 es un borde auténtico, sin embargo, el píxel en la columna 8 es el resultado de una detección errónea debida, por ejemplo, a una variación en la iluminación a lo largo de la secuencia, o a ruido en el detector (la cámara). En algunos casos, la respuesta del detector de bordes a ese ruido es tan intensa que no puede ser filtrada en la etapa de umbralización. Sin embargo, durante el proceso de detección de la velocidad, la respuesta del correlador debe ser máxima cuando se correlacione el píxel de la columna 7 en la imagen  $t=0$  con el píxel de la columna 5 en la imagen  $t=1$ , mientras que la otra correlación (con el píxel de la columna 8) tendrá una respuesta de menor magnitud (si no se ha binarizado la imagen). Podremos decir, finalmente, que el movimiento del borde es hacia la izquierda de la imagen y con una velocidad de 2 píxeles por imagen. Cuando la correlación máxima se corresponde con el falso borde estamos ante la aparición de ruido en la detección de movimiento.

El proceso descrito se repite para cada píxel de la imagen  $t=0$ , y correlacionamos cada imagen con la que le sigue en la secuencia, hasta el final de la misma.

Como se desprende del ejemplo ilustrado en la Figura 2.11, nuestro algoritmo detecta el movimiento que se realiza en dos direcciones (hacia la derecha y hacia la izquierda de la imagen), y para la visualización del resultado lo codificamos en colores: con el color rojo señalaremos los bordes que se mueven hacia la derecha y el con el verde aquellos bordes que se mueven hacia la izquierda.

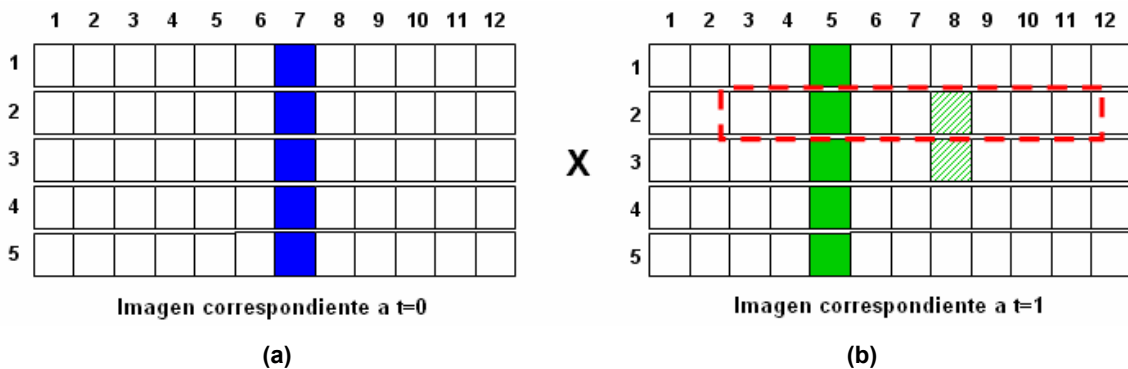


Figura 2.11. Funcionamiento del modelo de correladores de Reichardt desarrollado para el algoritmo.



En la Figura 2.12 se muestra el resultado de la detección del movimiento en varias imágenes de una secuencia. La secuencia está extraída de la película “Tiempos Modernos” de Charles Chaplin (1936) y muestra, desde una cámara estática, cómo se trasladan por la escena dos personajes. El personaje de Chaplin aparece por la derecha y se mueve hacia la izquierda de la imagen con velocidad (aproximadamente) constante, mientras empuja unas bandejas con comida por encima de un mostrador. Se puede considerar que el movimiento de las bandejas y de Chaplin es solidario, es decir, constituyen un único sólido trasladándose por la escena. El personaje de la camarera está en la izquierda de la imagen y se mueve hacia la derecha, muy despacio al principio, y a mayor velocidad al final de la secuencia. Durante el movimiento de traslación de ambos personajes, también se producen otros pequeños movimientos, por ejemplo, el personaje de Chaplin gira la cabeza a derecha e izquierda; aunque ese movimiento no afecta al movimiento de traslación de la silueta (bordes externos) del personaje, que es el movimiento principal, sí que afecta al movimiento de los rasgos de la cara (ojos, bigote, cejas, boca, etc.) que podrían ser detectados con movimiento propio. Por otro lado, el personaje de la camarera realiza, hacia el final de la secuencia, un movimiento compuesto de traslación (desplazándose hacia la derecha de la imagen) y de rotación (girándose hacia la izquierda).

Se puede observar en la Figura 2.12 la existencia de fenómenos ruidosos, esto es, píxeles codificados con el color rojo (con movimiento hacia la derecha) y que deberían estar codificados en verde (moverse hacia la izquierda), y viceversa. Por ejemplo, los píxeles en rojo que pertenecen a los alimentos de las bandejas que empuja Chaplin (imágenes 1 a 4 de la Figura 2.12), pueden tratarse como ruido, ya que, deberían ser verdes (moverse hacia la izquierda). La aparición de este ruido en la detección del movimiento se debe al fenómeno descrito anteriormente (Figura 2.11), es decir, a que el borde que ha maximizado la correlación no era el que correspondía al movimiento real, sino a otra característica de la imagen.

Como hemos dicho anteriormente, la secuencia está rodada con una cámara fija, sin embargo, resulta curioso ver cómo bordes que teóricamente no se mueven en la escena, por ejemplo los que pertenecen a los letreros que aparecen pegados a la pared, en realidad se detectan en movimiento. En la época en la que se realizó la película, 1936, el motor de la cámara en funcionamiento no era eficientemente amortiguado, de forma que la vibración producida por dicho motor se transmitía a la imagen. El resultado es la detección de pequeños movimientos a derecha e izquierda en objetos que son estáticos en la escena. En todo caso este tipo de ruido es muy frecuente en secuencias tomadas con cámaras fijas (con alguna vibración) o montadas sobre vehículos u otras plataformas en movimiento.

En el procedimiento descrito mediante la Figura 2.11, se hace una correlación píxel a píxel, sin embargo, una posible mejora de los resultados se podría dar si se realiza la correlación de estructuras mayores, es decir, de grupos de píxeles en lugar de un único píxel, y

se podría esperar que el ruido debido a correlaciones con píxeles de falsos bordes afecte en menor grado (véase los resultados en el capítulo de evaluación).



**Figura 2.12.** Extracción del movimiento mediante el algoritmo de correlación de Reichardt: **(a)** Imagen original procesada; **(b)** Características en movimiento.

De la descripción anterior se desprende que, para detectar la velocidad de un borde mediante los correladores de Reichardt, es necesario que la velocidad del borde esté dentro de las velocidades características que el conjunto de correladores posee. En caso contrario, estos

no responderán de forma significativa al estímulo, y el sistema no será capaz de detectar la velocidad del borde. Un ejemplo biológico de este hecho lo tenemos en el *Escarabajo tigre australiano* (cincindela hudsoni) [GIL97], que es el insecto más rápido del mundo (10 Km/h) teniendo en cuenta su tamaño. Sin embargo, sus detectores de movimiento responden a velocidades lentas, lo que lo convierte en ciego mientras corre. Por eso, necesita pararse de vez en cuando durante su carrera, para ajustar su trayectoria hacia la presa.

## 2.4. Filtrado de la imagen

Hemos visto en apartados anteriores que existen diversas causas que pueden introducir variabilidad en la magnitud de los bordes de los objetos a lo largo de la secuencia. Algunos de esos factores son las diferencias de luminosidad existente entre dos imágenes consecutivas, el ruido electrónico en el detector, y, sobre todo, el hecho de que la detección de los bordes de los objetos sea altamente dependiente del entorno de éstos, con lo que, al cambiar el fondo sobre el que se mueve el objeto durante su traslación, la magnitud de los bordes puede cambiar de forma sustancial.

Esta variabilidad es el origen de los fenómenos que, en la etapa de correlación, falsean los resultados de detección de movimiento, introduciendo lo que hemos denominado 'ruido' en la detección del movimiento.

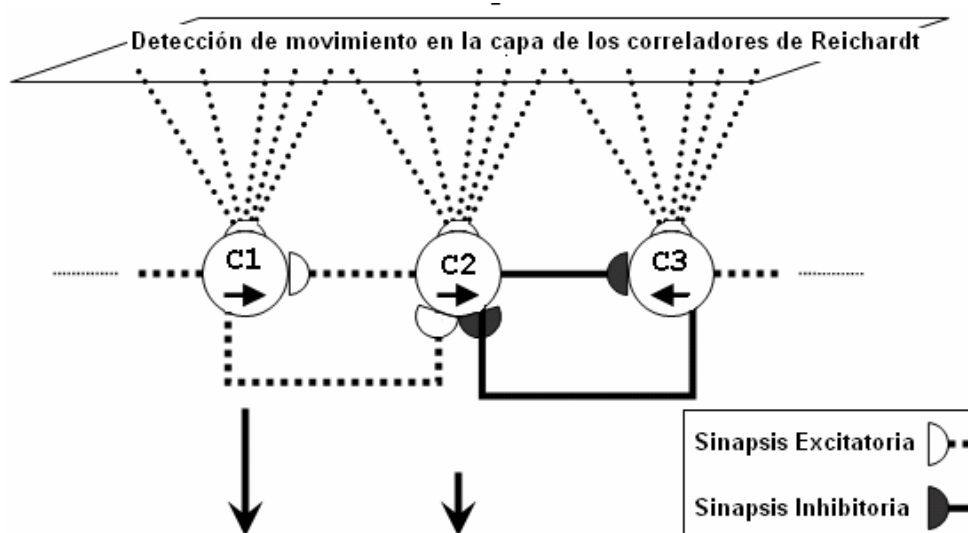
Con el fin de limpiar la imagen de ese ruido se proponen dos criterios de filtrado: un criterio de coherencia espacial y otro de coherencia temporal de los movimientos. Ambos son consistentes con fenómenos y principios de funcionamiento de estructuras neuronales biológicas.

Algunos estudios sugieren que la integración de la información local permite discriminar los objetos en entornos ruidosos [BAR78, FIE93, SAA97, GIL85]. El mecanismo de esta integración en los sistemas biológicos es casi desconocido, aunque algunos estudios neurofisiológicos en el cortex visual de los primates indican que la información local es integrada en patrones globales muy pronto en el sistema visual [GIL85, GRO93]. Siguiendo esta idea, se propone una estructura neuronal que recoge la salida de la etapa de los correladores de Reichardt [MOT04a]. Esta capa neuronal está compuesta por *células colectoras*. En cada sección de la capa existen distintas células, cada una de ellas sensible a una velocidad distinta o conjunto de velocidades próximas entre sí, que recoge las conexiones excitatorias de un área local de la capa de correladores de Reichardt. De esta manera, tendremos una serie de células colectoras que integran la información del movimiento a cierta velocidad en su área de influencia (campo receptivo).

Esta capa colectora está configurada como una estructura competitiva, en la que la célula colectora que recibe la mayor excitación inhibe al resto de las células colectoras en su área de influencia o campo receptivo, y domina, es decir, es la única que dispara a la salida (estructura competitiva de tipo 'winner-takes-all', WTA). Esta estructura funciona como un filtro que resalta la presencia de sólidos rígidos y omite los patrones debidos al ruido mediante la integración de la información local (coherencia espacial del movimiento).

También cabe considerar la constante de tiempo que determina la activación o desactivación de las neuronas de la capa colectora (la capa de filtrado). Si este tiempo es suficientemente largo, son necesarios patrones de movimiento duraderos para activar una neurona que domine frente a patrones espureos (coherencia temporal del movimiento).

Para facilitar la detección de sólidos de tamaño mayor se producen interacciones de carácter excitatorio o inhibitorio entre las células colectoras ganadoras de áreas de influencia contiguas.



**Figura 2.13.** La figura muestra las conexiones sinápticas de tres neuronas colectoras que integran la actividad de los correladores de Reichardt en un área local de la imagen. Dos neuronas detectan movimiento hacia la derecha (→) y la tercera detecta movimiento hacia la izquierda (←).

El comportamiento neuronal de la etapa de filtrado se describe en la Figura 2.13, que muestra las conexiones sinápticas de tres neuronas colectoras ganadoras, cada una de las cuales integra la máxima actividad de los correladores de Reichardt en su campo receptivo (área de influencia de la imagen), e inhibe a las otras neuronas colectoras con las que comparte el campo receptivo. Dos de las neuronas colectoras ganadoras (C1 y C2) han detectado el movimiento de un sólido rígido hacia la derecha (→) y la tercera (C3) ha detectado movimiento hacia la izquierda (←). Se producen interacciones de carácter excitatorio entre las neuronas C1 y C2, puesto que ambas han detectado un sólido rígido moviéndose a la misma velocidad; mientras que entre las neuronas C2 y C3 se produce una interacción sináptica

inhibitoria debido a que ambas neuronas han detectado movimientos diferentes, es decir, poco coherentes con lo que debería ser un sólido rígido de gran tamaño.

La inhibición lateral es uno de los mecanismos más ampliamente usados en la naturaleza; se han usado para explicar la sensibilidad al movimiento en las ranas [LET59], la adaptación a la baja luminosidad de las células ganglionares de la retina del gato [ENR66], entre otras funciones.

En determinadas situaciones, por ejemplo una escena con dos objetos en movimiento en direcciones contrarias y cierta oclusión o superposición entre ambos objetos, un sistema basado en una estructura WTA pura puede resultar muy restrictivo en el filtrado, es decir, la salida en algunos casos puede ser nula para alguno de los objetos, mostrándolo sólo parcialmente en aquellas áreas de influencia en las que no existe oclusión o filtrándolo completamente. Como alternativa para estos casos se propone una capa colectora en la que la estructura “winner-takes-all” se transforma en una estructura “some winners-take-all”, es decir, no sólo se dispara la neurona colectora ganadora de cada campo receptivo, sino que se establece un umbral de excitación a partir del cual se disparan aquellas neuronas colectoras cuya excitación lo sobrepasen (véase el capítulo dedicado a la evaluación del algoritmo).

Desde un punto de vista puramente algorítmico el comportamiento del sistema se describe en la siguiente sección.

### **2.4.1. Coherencia espacial: Movimiento del sólido rígido**

Si estamos detectando el movimiento (coherente) de los bordes pertenecientes a objetos sólidos, todos los píxeles pertenecientes a estos deben moverse en la misma dirección y con la misma velocidad (criterio de movimiento de traslación de un sólido rígido) [MOT04b, PUG05, MOT04c]. En definitiva, el criterio de coherencia espacial busca que, en la misma región del espacio, todos los píxeles tengan el mismo tipo de movimiento, que será coherente espacialmente si pertenecen al mismo sólido que se traslada de un lado a otro de la escena. Los píxeles aislados que no compartan el movimiento del sólido rígido se consideran ruido, y se filtran.

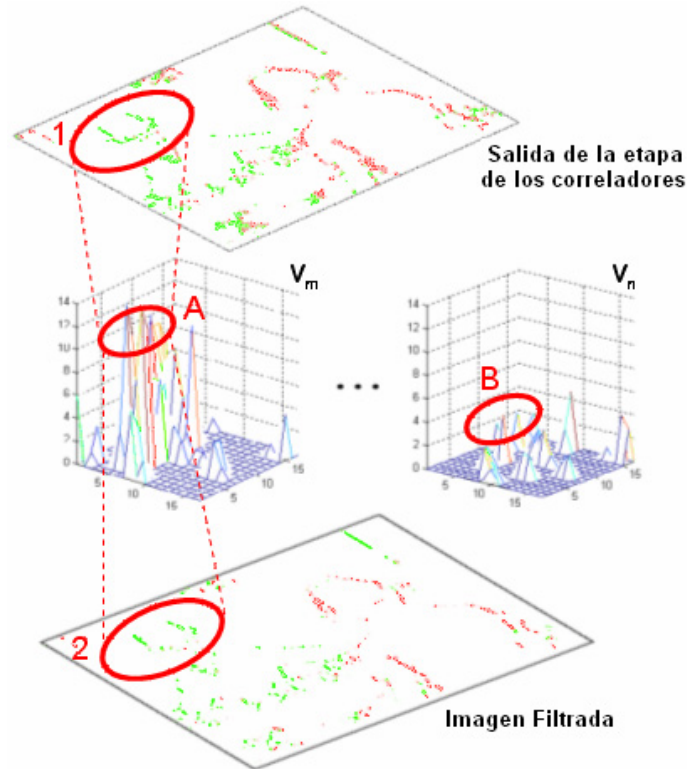
Para aplicar el criterio de coherencia espacial o de movimiento del sólido rígido se divide la imagen en secciones o áreas de influencia (Figura 2.14). En cada sección se integran los píxeles que están activos para cada velocidad (o conjunto de velocidades próximas entre sí) y dirección dadas (en la parte central de la Figura 2.14 observamos la actividad para las velocidades  $V_n$  y  $V_m$  en las distintas regiones de la imagen). La actividad debida a la velocidad  $V_m$  en las secciones delimitadas por la elipse A (parte central de la Figura 2.14) es mayor que la debida a la velocidad  $V_n$  en las mismas secciones (delimitadas por la elipse B). Podemos

interpretar que los píxeles cuya velocidad es  $V_m$  constituyen un sólido rígido, mientras que los que se mueven a  $V_n$  no, por tanto, la actividad dentro de la elipse B es inhibida por la actividad en la elipse A, y los píxeles que se movían a velocidad  $V_n$  en la imagen de salida de la etapa de correlación, son filtrados. De esta manera, los píxeles dentro de la elipse 1 (Figura 2.14), son filtrados mediante el criterio de sólido rígido (coherencia espacial), quedando activos después de la etapa de filtrado los píxeles dentro de la elipse 2 (Figura 2.14), es decir, aquellos cuyo movimiento coincide con el de la sección de máxima actividad.

El tamaño de las secciones (o áreas de influencia local) en las que dividimos la imagen va a tener efecto sobre el tamaño de los sólidos rígidos cuyo movimiento filtramos. Como se comentó antes, el personaje de Chaplin realiza un movimiento principal de traslación a través de la escena, y otro movimiento menor de giro de la cabeza (hacia la derecha y la izquierda) que, en la detección, afecta solamente a los bordes que pertenecen a las facciones de la cara (boca, bigote, cejas,...). Como el objeto sólido que queremos considerar es grande (los personajes), las secciones en las que hemos dividido la imagen son de un tamaño tal que sólo las zonas extensas del personaje (por ejemplo, la cabeza) son consideradas como un sólido rígido, y sin embargo, los rasgos de la cara, que están compuestas por muy pocos bordes en comparación, tienen poca influencia en la sección en la que se integran. Debido a esto son filtrados cuando se mueven en la dirección contraria al resto.

Si considerásemos regiones de menor tamaño, los objetos pequeños, compuestos por un número menor de bordes, comienzan a tener influencia en su sección, y también serían considerados sólidos rígidos, y por tanto no serían filtrados como sucede en el caso anterior (véase los resultados mostrados en la Figura 2.15). Como puede entenderse de la explicación precedente, el tamaño de las secciones va a venir determinado por la aplicación de detección de movimiento y por el tamaño de los sólidos rígidos que estemos tratando de detectar.

Para tener en cuenta el movimiento de sólidos aún mayores, cuyos píxeles ocupen el área comprendida por varias áreas de influencia (secciones de filtrado), también se aplica un criterio de coherencia espacial entre regiones vecinas. Es decir, si existe una región cuyo máximo número de píxeles con movimiento coherente está rodeada de otras regiones con movimiento coherente distinto, lo más probable es que sea una región ruidosa, y por tanto, será filtrada. Esta etapa de coherencia será también muy dependiente de la aplicación (tamaño de los objetos a detectar, tamaño de las zonas de influencia usadas, etc.). Además normalmente, éste segundo efecto de filtrado es de menor influencia, permitiendo que un objeto cercano moviéndose en dirección contraria pueda imponer su patrón de movimiento.



**Figura 2.14.** Descripción del proceso de filtrado mediante el criterio de movimiento del sólido rígido. Se muestra la imagen de salida de la etapa de los correladores de Reichardt (parte superior), la integración que se lleva a cabo en las distintas secciones de la imagen para cada velocidad y dirección (parte central) y el resultado final de la etapa de filtrado (parte inferior).

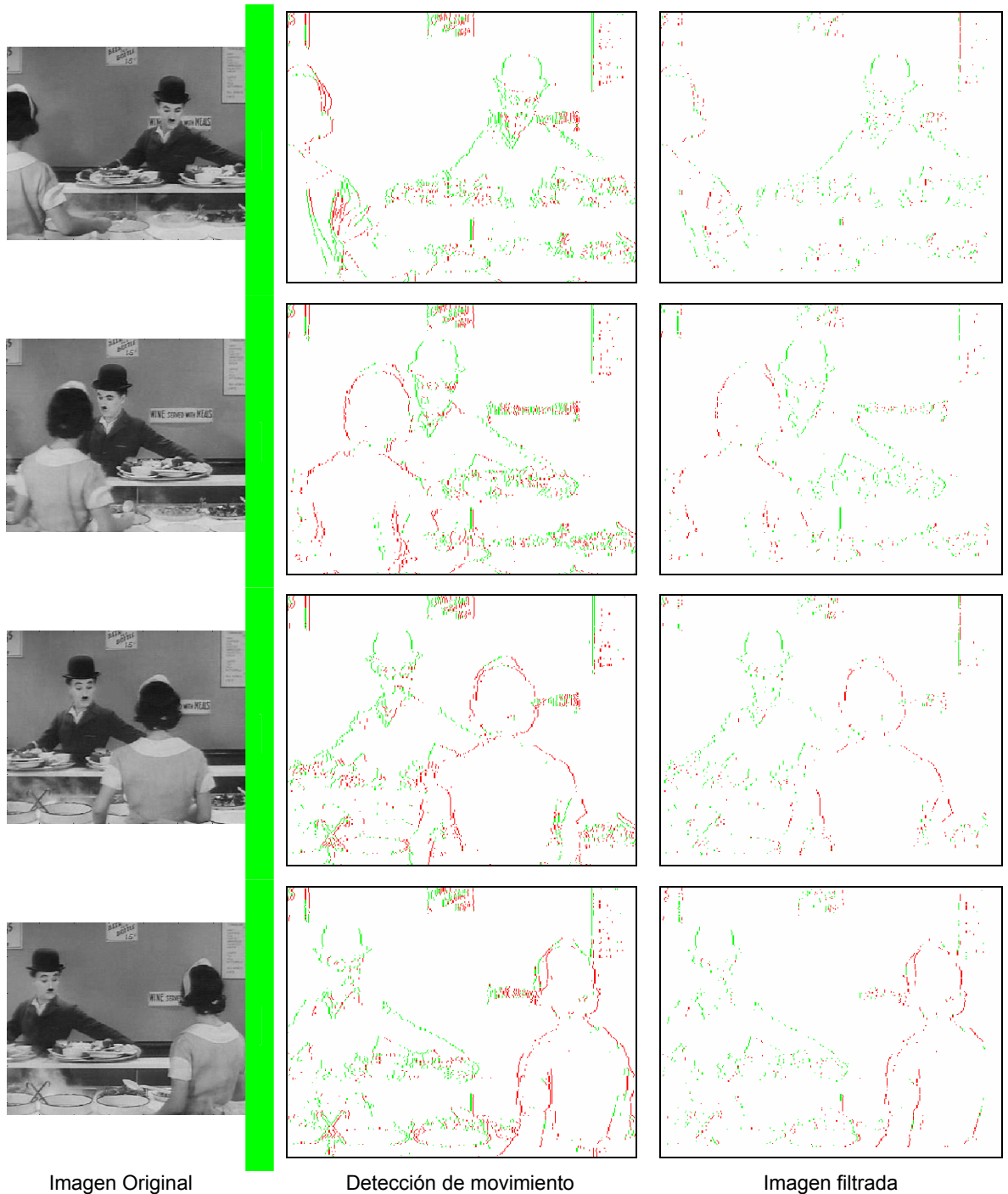
## 2.4.2. Coherencia temporal

El movimiento de un sólido rígido debe ser continuo, es decir, los píxeles que le pertenezcan deben de aparecer moviéndose en las distintas imágenes en las que el sólido aparezca. Cuando existen píxeles con movimiento en una imagen y no aparecen en la siguiente, no los podemos considerar pertenecientes a un sólido en movimiento, sino que, seguramente son debidos a la aparición de la correlación de un borde 'real' con un borde 'falso'. Es decir, los objetos no aparecen y desaparecen instantáneamente, ni se ponen en movimiento o se detienen de forma inmediata. Estos son los aspectos que se pretenden filtrar.

Aplicamos el criterio de coherencia temporal del movimiento para filtrar el ruido espúreo en la detección. Este criterio mejora la estabilidad en la respuesta a patrones de movimientos de traslación en entornos ruidosos.

En la Figura 2.15 se observa el resultado de la aplicación de estos criterios de filtrado. Como puede verse, una buena parte del ruido original ha sido eliminado. La optimización de los

parámetros del algoritmo dará como consecuencia una imagen mucho menos ruidosa (véase el capítulo 4).



**Figura 2.15.** Proceso de filtrado del movimiento extraído en la etapa de los detectores de Reichardt a lo largo de distintas imágenes de la secuencia: la columna (a) representa las imágenes de la secuencia que están siendo evaluadas; la columna (b) muestra aquellas características que se detectan moviéndose; en la columna (c) aparecen, aquellas características que responden a los criterios de filtrado. Se mantiene la codificación en colores expuesta anteriormente.



## 2.5. Conclusiones

- ☑ En este capítulo hemos presentado un algoritmo de detección de movimiento de bajo nivel.
- ☑ Dicho algoritmo está basado en la superposición de distintas capas neuronales, inspiradas en el principio de funcionamiento del sistema visual de la mosca y el de algunas estructuras neuronales biológicas.
- ☑ La extracción de características (bordes) se realiza mediante un proceso de convolución con la máscara de 3x3 píxeles del operador derivativo de Sobel porque son métodos relativamente rápidos que proporcionan resultados precisos con una tasa de error relativamente baja en comparación con otros métodos de detección de bordes habituales, nos permiten la integración sencilla dentro de una estructura de procesamiento en segmentación de cauce, nos permiten la obtención de la componente horizontal y vertical de los bordes de forma independiente con los mejores resultados con una máscara tan pequeña, y los problemas relacionados con sensibilidad a imágenes ruidosas no es diferente a otros operadores sencillos.
- ☑ Para hacer uso de cámaras de alto rango dinámico y bajo condiciones de iluminación con pequeños cambios locales se ha implementado la extracción de bordes en el espacio logarítmico mediante el modelo de LIP-Sobel.
- ☑ Se ha estudiado las ventajas de realizar una umbralización dinámica de los bordes obtenidos para mantener un número constante de características en la imagen frente al uso de umbrales fijos.
- ☑ Se ha utilizado el criterio de supresión de valores no máximos para la obtención de bordes de una anchura de un píxel.
- ☑ Se ha presentado la detección de la velocidad de las características previamente extraídas mediante el uso de conjuntos de correladores de Reichardt, cada uno capaz de sintonizar una velocidad diferente. La velocidad final vendrá dada por el correlador que maximice la correlación.
- ☑ El conjunto de correladores empleado es característico de la aplicación, es decir, de las velocidades que sea necesario detectar, y es, por tanto, un parámetro del algoritmo de detección de movimiento.
- ☑ La existencia de fenómenos ruidosos en la detección de movimiento hace necesaria la aplicación de otra capa neuronal que integra la información de movimiento y mediante criterios de coherencia espacio-temporal filtra dicho ruido. Esta capa neuronal está basada en estructuras competitivas de tipo winner-takes-all. Para ciertas aplicaciones, también se ha desarrollado una variante consistente en estructuras competitivas de tipo “some winner-take-all”, para la que es necesario establecer un umbral de disparo de las neuronas.

---

## CAPÍTULO 3:

# Implementación hardware del algoritmo de detección de movimiento

---

En este capítulo se describe la implementación en hardware específico del algoritmo de detección de movimiento descrito en el capítulo anterior. El resultado es un sistema en un chip (SoC) que funciona en tiempo real y puede emplearse en múltiples campos de aplicación. Se ha utilizado una estrategia de diseño modular por lo que resulta sencilla la integración de módulos que realicen tareas específicas. También hemos aprovechando el potencial de los recursos de computación paralela y de segmentación de cauce que son inherentes de las FPGAs, por tanto, la arquitectura adopta un esquema de segmentación de cauce de grano grueso en el que, cada una de las etapas de procesamiento constituyen módulos independientes entre sí que se ejecutan en paralelo, y por otro lado, cada uno de los módulos del procesamiento se han diseñado como una arquitectura de propósito específico siguiendo un esquema de segmentación de cauce fino o microsegmentación de cauce de datos.

## 3.1. Introducción

El desarrollo de los dispositivos electrónicos permite en la actualidad la integración de sistemas digitales completos en un único circuito integrado, son los denominados *System-on-Chip* o *SoC*. Estos sistemas incluyen con frecuencia uno o varios procesadores (sistemas multiprocesador), en función de la complejidad del diseño desarrollado, pero también comienzan a ser utilizados los dispositivos reconfigurables, siendo esta alternativa una de las más utilizadas dada la flexibilidad que ofrecen.

En este capítulo se describe y analiza la implementación en hardware específico del algoritmo de detección de movimiento descrito en el capítulo anterior, por lo que tenemos todo el sistema de procesamiento en un chip programable (PSoC) funcionando en tiempo real.

La entrada al sistema la constituye una cámara CCD que proporciona 25 imágenes por segundo de tamaño 720x576 y con 256 niveles de gris. También se han utilizado como entrada al sistema diversas secuencias obtenidas mediante una cámara de alto rango dinámico. Un PC suministra al circuito dichas secuencias, usándose para ello la entrada de S-Video que posee la plataforma de prototipado.

La mayor parte del procesamiento se realiza mediante la placa de prototipado RC-200 de Celoxica [CEL06a], que dispone de un dispositivo FPGA de la familia Virtex-II de Xilinx (XC2V1000) [XIL06] y dos bancos de memoria ZBT SRAM, cada dirección de la memoria puede alojar datos de longitud de palabra de 36 bits (Figura 3.1). El dispositivo de Xilinx Virtex-II tiene 1 millón de puertas equivalentes distribuidas en 5120 slices y 40 bloques de memoria embebida, con un total de 720 Kbits, también dispone de 40 multiplicadores embebidos.

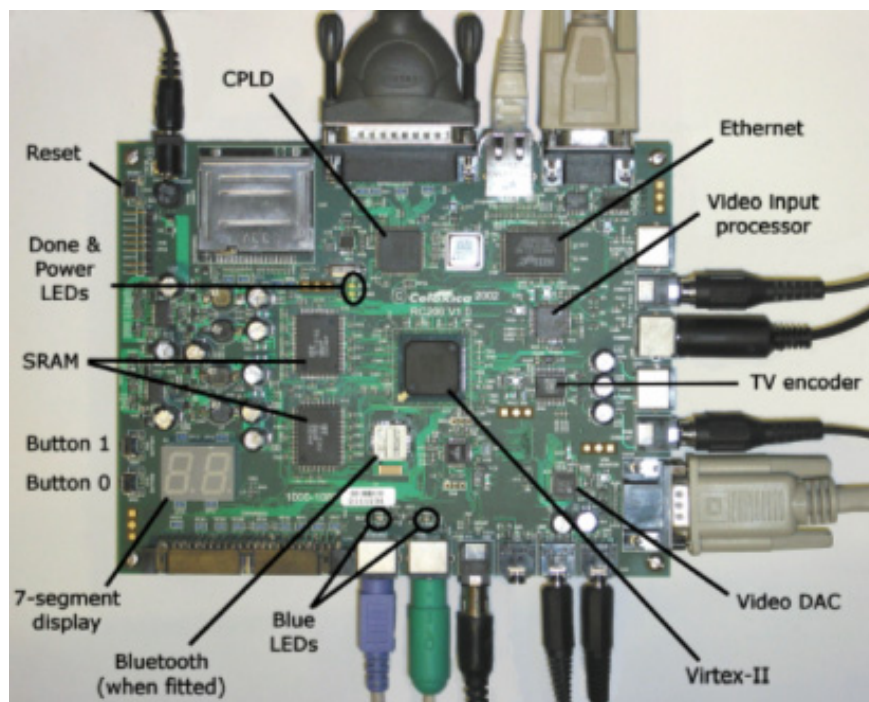
Para los diseños con mayor consumo de recursos también se ha utilizado otra placa de prototipado, la RC-203E de Celoxica, que dispone de un dispositivo FPGA de la familia Virtex-II de Xilinx (XC2V3000). Este dispositivo tiene 3 millones de puertas equivalentes distribuidas en 28672 slices y 96 bloques de memoria embebida y 96 multiplicadores embebidos.

Ambas plataformas de prototipado son idénticas en cuanto a interfaces e entrada y salida, las diferencias fundamentales están en la capacidad del dispositivo FPGA y la mayor capacidad de los bancos de memoria ZBT SRAM en la RC-203E.

Utilizamos la salida VGA de la placa como medio para mostrar los resultados del procesamiento en la FPGA. Por tanto, solamente van a ser procesados 640x480 píxeles de los 720x576 que nos suministra la cámara, aunque como se verá posteriormente, el dispositivo tiene capacidad para procesar imágenes de mayor tamaño y puede utilizar cámaras de mayor frecuencia de muestreo.

Hemos definido la arquitectura de procesamiento mediante un lenguaje de descripción de hardware de alto nivel: Handel-C de Celoxica [CEL06b]. Este lenguaje genera un archivo de tipo *Edif* que constituye la entrada para el entorno ISE de Xilinx, que se encargará de generar el archivo '\*.bit' de configuración de la FPGA. Handel-C está basado en ANSI-C, por lo que la descripción de los circuitos se realiza de forma muy algorítmica.

Hemos realizado la definición del sistema con un nivel de abstracción alto, implementando una estrategia de diseño modular, por lo que resulta sencilla la integración de módulos que realicen tareas específicas en el pre-procesamiento o el post-procesamiento de las secuencias, lo que permite crear diferentes versiones del mismo circuito, que lo dotan de bastante flexibilidad para su aplicación en diversos ámbitos.



**Figura 3.1.** Plataforma de prototipado RC200 de Celoxica empleada en la implementación del algoritmo.

El algoritmo de procesamiento se divide en módulos independientes usando una arquitectura de segmentación de cauce de grano grueso, y se ejecutan en paralelo. Cada uno de los módulos del procesamiento se ha diseñado como una arquitectura de propósito específico siguiendo un esquema microsegmentación de cauce de datos. Como resultado de dicha arquitectura el circuito es capaz de procesar un píxel en cada ciclo de reloj.

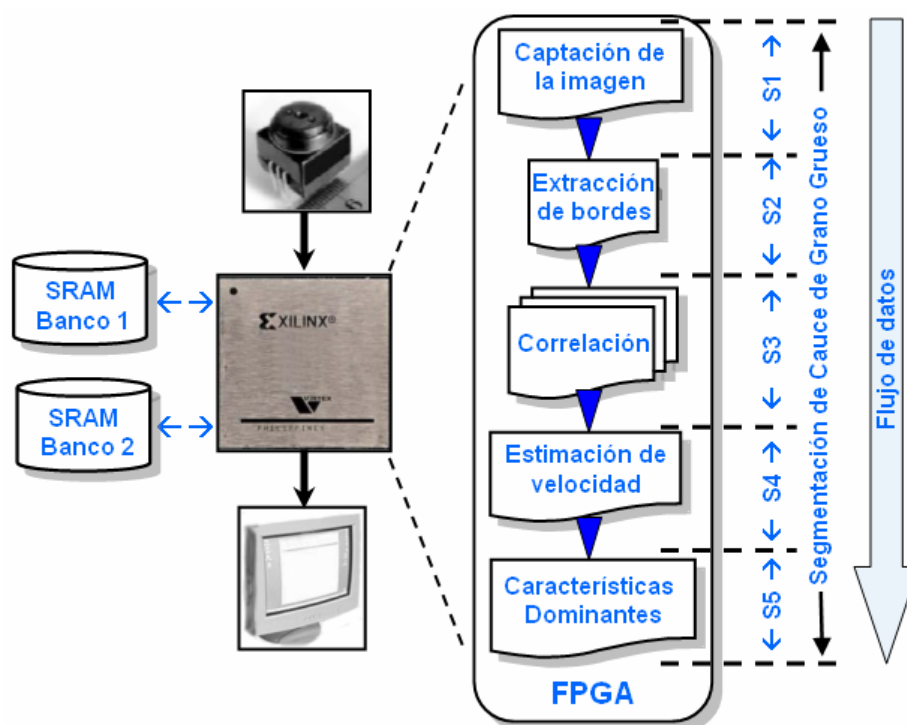
También hemos integrado módulos específicos (*cores*), para realizar divisiones optimizadas cuando ha sido necesario.

Se ha realizado la evaluación de los recursos empleados por el hardware en el procesamiento para las distintas aproximaciones.

## 3.2. Arquitectura de procesamiento con segmentación de grano grueso

Hemos implementado el algoritmo de detección del movimiento descrito en el Capítulo 2 aprovechando el potencial de los recursos de computación paralela y de segmentación de cauce que son inherentes de las FPGAs.

La arquitectura adopta un esquema de segmentación de cauce de grano grueso en el que, cada una de las diferentes etapas de procesamiento explicadas en el Capítulo 2, constituyen módulos independientes entre sí que se ejecutan en paralelo. Todas las etapas o módulos del procesamiento se comunican entre sí mediante el uso de estructuras específicas (canales de comunicación).



**Figura 3.2.** Estructura de la segmentación de cauce de grano grueso del sistema, en el que vemos la distribución de los diferentes módulos de procesamiento.

La arquitectura global se estructura en los siguientes módulos de procesamiento, como se muestra en la Figura 3.2, cada uno de los cuales realiza una tarea específica:

1. **Captación de la imagen (S1):** realiza las funciones de interfaz entre la cámara y el dispositivo de procesamiento (FPGA).
2. **Extracción de bordes (S2):** selecciona aquellas características de la imagen que determinan la estructura de la misma (bordes).

- 3. Selección de la dirección del píxel mediante correlación con diferentes constantes de tiempo (S3):** aplicamos los correladores de Reichardt a las imágenes proporcionadas por la cámara.
- 4. Etapa de cómputo del máximo (“Winner takes all”) o etapa de estimación de la velocidad (S4):** determinamos la velocidad a la que se mueve el píxel (la característica o borde) viendo qué correlador maximiza la correlación.
- 5. Selección de las características dominantes (S5):** tratamos de establecer criterios de coherencia espacial entre las características en movimiento.
- 6. Módulo de gestión de memoria:** gestionamos la lectura-escritura en los módulos SRAM de memoria externa.

Hay dos etapas críticas en el esquema anterior: la etapa de estimación de velocidad (S4) y la etapa de selección de las características dominantes (S5). En ambas etapas hay que determinar un máximo. Los algoritmos conocidos de ordenación o cálculo de máximos y mínimos son secuenciales e iterativos, para obtener un píxel procesado por cada ciclo de reloj no podemos permitirnos una estructura secuencial, por tanto hay que definir el cálculo de los máximos dentro de una arquitectura con segmentación de cauce. Son dos etapas para las que hay que tener en cuenta el coste de hardware frente a la eficiencia en términos sensibilidad del rango de velocidades que requiera la aplicación concreta.

### **3.3. Arquitectura de procesamiento con segmentación de grano fino**

Cada uno de los módulos de procesamiento, ha sido cuidadosamente diseñado como una arquitectura de propósito específico. El hecho de tener un flujo de datos regular permite diseñar cada módulo siguiendo un esquema de segmentación de cauce fino o microsegmentación de cauce de datos. Esta arquitectura supone numerosas ventajas, por ejemplo el circuito es capaz de procesar un píxel por cada ciclo de reloj, con una latencia que dependerá de la longitud del cauce. La latencia del sistema global y de cada una de las etapas de procesamiento es una característica de menor importancia ya que no influye en la potencia de cómputo (datos/estimaciones por segundo).

#### **3.3.1. Captación de la imagen (S1)**

La imagen de la cámara o la procedente del PC, según el caso, son recibidas por el circuito de captación que realiza las funciones de interfaz entre la cámara y la FPGA. Como

interfaz realiza una doble función: a) desentrelazar la imagen que nos proporciona el decodificador de video; b) estructurar los datos en un buffer de tipo FIFO, que se implementará en bloques de memoria interna (embebida).

**a) Desentrelazado de la imagen:**

La cámara utilizada, como muchas cámaras comerciales, tiene un sensor que recoge información sobre la imagen leyendo primero las líneas impares y luego las pares. Aunque existen cámaras progresivas que suministran todas las líneas de forma consecutiva.

Para poder realizar el procesamiento necesitamos transformar el modo entrelazado en un modo progresivo. Esta transformación se consigue escribiendo la imagen que actualmente recoge la cámara en uno de los bancos de la SRAM externa de una manera ordenada. Es decir, escribimos las filas impares en la memoria y reservamos entre ellas las direcciones que corresponderán a las filas pares que leeremos más tarde. Cuando recibimos las filas pares rellenamos los huecos de la memoria que hemos dejado para ellas.

**b) Creación de un buffer con los datos de la imagen actual:**

Una vez almacenada la imagen actual en la memoria, para la recuperación de la misma realizamos una lectura progresiva o continua, es decir, leemos las filas de la imagen en su orden natural. Esta etapa de lectura prepara los datos de entrada que se utilizarán en el siguiente módulo de extracción de bordes.

Cada ciclo de reloj recibimos un píxel de la SRAM que se almacena en un buffer de tipo FIFO. En total definimos dos buffers, y ambos tendrán la longitud de una línea de la imagen. Estos buffers se almacenan en los bloques de memoria embebida de la FPGA, en la que hemos definido una RAM de doble puerto, lo que significa que podremos escribir y leer de la misma en el mismo ciclo de reloj.

Hemos escogido definir la RAM en la memoria embebida porque es más eficiente y conlleva el uso de menos lógica asociada que otras alternativas como el uso de memoria distribuida de la FPGA.

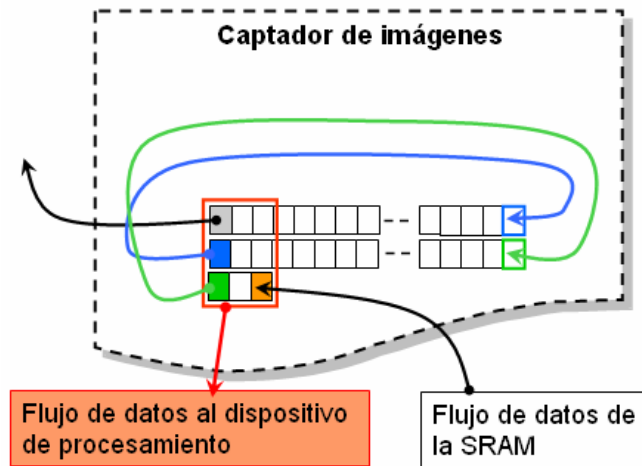
Una vez que hemos recibido píxeles suficientes de la SRAM y tenemos rellenos ambos buffers, recibimos tres nuevos píxeles de la fila siguiente, y puede comenzar el proceso transferencia de píxeles para ser usados en el siguiente módulo dedicado a la extracción de bordes. Ahora en cada ciclo de reloj se envían los tres primeros píxeles de cada uno de los buffers más los tres píxeles de la nueva fila a la etapa de extracción de bordes, y se recibe un nuevo píxel leído de la SRAM.

En paralelo con estas operaciones y dentro de la segmentación del cauce de datos de grano fino diseñado, se sigue un proceso cíclico de renovación de píxeles en los buffers. El

primer píxel del buffer correspondiente a la primera fila, es desechado, mientras que el resto de los píxeles de este buffer se desplazan una posición hacia el principio del mismo. De esta forma nos queda vacía la última posición del buffer, que será ocupada por el píxel situado en el primer lugar del buffer correspondiente a la segunda fila (Figura 3.3).

En el segundo buffer se procede de forma similar, es decir, todos los píxeles se desplazan una posición hacia el principio del mismo, y en la última posición de dicho buffer se sitúa el primer píxel de la tercera fila. El nuevo píxel recibido de la lectura de la SRAM, será el cuarto píxel de la tercera fila.

Ambos procesos (desentrelazado de la imagen y gestión de los buffers de datos) se realizan en paralelo. El circuito de captación se ha diseñado mediante una arquitectura con un cauce de grano fino que tiene 3 etapas para el desentrelazado de la imagen y 4 etapas para la gestión de los buffers, por tanto, la microsegmentación de cauce del circuito de captación tiene 4 etapas en total.



**Figura 3.3.** Proceso de renovación cíclica de píxeles en los buffers generados por el circuito de captación de imágenes. El píxel que ocupaba la posición sombreada en azul se desplaza a la posición enmarcada en azul. Y el píxel que ocupaba la posición sombreada en verde se desplaza a la posición enmarcada en verde. El píxel que ocupa la posición sombreada en gris se elimina, y entra un nuevo píxel (procedente de la memoria SRAM externa) a ocupar la posición sombreada en naranja. Todos los píxeles que ocupan posiciones sombreadas en blanco se desplazan una posición a la izquierda. Todos estos desplazamientos se producen en paralelo y en el mismo ciclo de reloj.

Puesto que tenemos que llenar dos buffers (uno después del otro), cada uno de capacidad 640 píxeles, es decir, la longitud a procesar de una línea de la imagen, más tres píxeles de la nueva línea, tenemos una latencia asociada a este módulo igual a  $640 \cdot 2 + 3$ , es decir, de 1283 ciclos de reloj.



### 3.3.2. Extracción de bordes (S2)

Puesto que el esquema de procesamiento implementado es modular, es posible introducir un módulo de extracción de bordes basado en el algoritmo Sobel en el espacio lineal o la extracción de bordes usando el algoritmo de Sobel en el espacio logarítmico (LIP-Sobel) según las necesidades de la aplicación. Por tanto, mostraremos la arquitectura de procesamiento empleada en ambos casos a continuación.

Como se comentó con anterioridad extraemos bordes espaciales usando un proceso de convolución con una máscara de dimensiones 3x3. A la salida de este módulo, el píxel clasificado como borde es almacenado en el banco 1 de la memoria SRAM externa, y al mismo tiempo se envía al módulo de detección de velocidad (véanse la Sección 3.4 y la Figura 3.6).

En la entrada del modulo actual, cuando el circuito de captación de imágenes está listo, envía un conjunto de 3x3 píxeles al módulo de extracción de bordes, que pasará a procesarlos de la siguiente forma:

#### a) Algoritmo de Sobel lineal:

Cuando aplicamos el algoritmo de Sobel para la extracción de bordes estamos realizando la convolución de la máscara 3x3 de Sobel con el conjunto de 3x3 píxeles de la imagen actual que envía el circuito de captación de imágenes a este módulo. Como vimos en el Capítulo 2, esto supone realizar operaciones sencillas de sumas, restas y multiplicaciones por 2, que se implementan con operadores de desplazamiento de bits.

Además se aplica el criterio de supresión de valores no máximos ('non-maximum suppression') y la umbralización. En ambos casos sólo hay que realizar comparaciones entre píxeles.

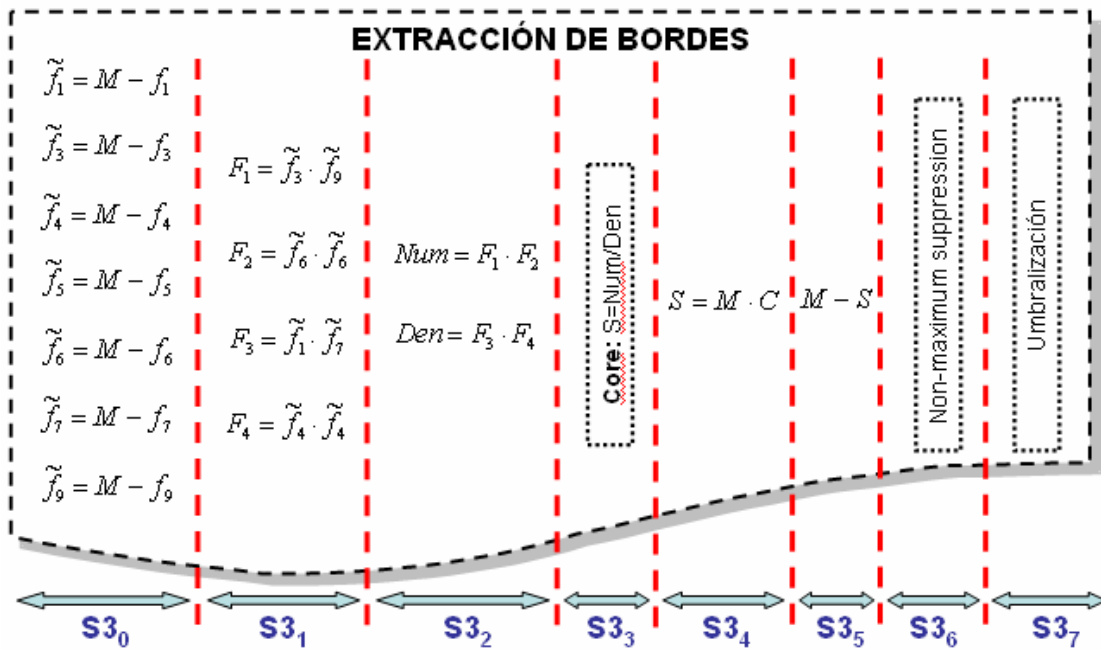
Todo este procesamiento se realiza mediante un cauce de datos segmentado en grano fino en 8 etapas.

#### b) Algoritmo de LIP-Sobel:

Como se vio en el Capítulo 2, la extracción de bordes en el espacio logarítmico es algo más compleja. Para obtener el píxel de bordes hay que aplicar la Ecuación 2.7, lo cual supone realizar 4 multiplicaciones de 8x8 bits y 2 multiplicaciones de 16x16 bits; y además hay que realizar una división donde, tanto el numerador como el denominador, tienen 32 bits.

La división es el paso crítico en la extracción de bordes en el espacio logarítmico, no sólo en cuanto a consumo de hardware, sino que el diseño total se ralentiza según las estimaciones del entorno ISE de Xilinx (véase en la Sección 3.5 la Tabla 1).

La alternativa está en el uso de un *core* parametrizable. El *core* escogido puede dividir dos variables de hasta 36 bits, y la profundidad de bits de numerador y denominador se pueden ajustar de forma independiente. El *core* está diseñado con una segmentación de cauce de datos fino, aunque también esto se puede particularizar para reducir el consumo de hardware a expensas de la velocidad, produciéndose una división cada 2, 4 ó 8 ciclos de reloj. Cuando el *core* funciona en el modo completamente optimizado, siendo esta la opción de configuración elegida en nuestro diseño, realiza una operación de división en un único ciclo de reloj después de una latencia inicial, que en nuestro caso es de 38 ciclos de reloj.



**Figura 3.4.** Segmentación de cauce para realizar las operaciones de la ecuación 2.7 que se usa en la etapa de extracción de bordes mediante LIP-Sobel. Las operaciones situadas en la misma columna se realizan en paralelo en el mismo ciclo de reloj.

De la misma forma que optimizamos el diseño del módulo de LIP-Sobel mediante el uso de un *core* para realizar la división, también podemos hacer uso de los multiplicadores embebidos para implementar las multiplicaciones. Se pueden ver los resultados en cuanto a consumo de recursos en la Tabla 3.1 de la Sección 3.5.

En la Figura 3.4 se muestran las distintas etapas del cauce de datos ( $S3_0$  a  $S3_7$ ) para la extracción de bordes mediante Sobel en el espacio logarítmico. En la etapa  $S3_0$  se realiza en paralelo y en el mismo ciclo de reloj el cálculo de las variables  $\tilde{f}_1$  a  $\tilde{f}_9$ ; lo mismo se puede decir de la etapa  $S3_1$  en la que en paralelo y en el mismo ciclo de reloj calculamos  $F_1$  a  $F_4$ .

Todo este procesamiento, incluidos el afinamiento de los bordes mediante el criterio de supresión de valores no máximos y la umbralización, se realiza con una segmentación de

cauce de grano fino de 8 etapas, también en este caso de extracción de bordes basado en LIP-Sobel.

### **3.3.3. Selección de la dirección del píxel mediante correlación con diferentes constantes de tiempo (S3)**

Como se explicó en la Sección 3 del Capítulo 2, es necesario realizar la correlación con diferentes constantes de tiempo para estimar la velocidad del píxel de bordes dentro de un rango de velocidades. Por tanto, correlacionamos cada píxel de bordes de la imagen actual con una población de píxeles de la imagen anterior. También sería posible correlacionar el píxel de la imagen actual con píxeles de varias imágenes anteriores, pero eso significaría tener que almacenar dichas imágenes en la memoria SRAM externa, y el almacenamiento de más de una imagen de bordes anterior a la actual es muy costoso, dadas las dimensiones de los bloques de SRAM y las necesidades de almacenamiento que tenemos para el procesamiento total.

La estructura con segmentación de cauce de la etapa S3 (Figura 3.2) recibe, desde el módulo de extracción de bordes, un píxel de la imagen que se está procesando (imagen actual).

En paralelo, se están recibiendo los píxeles correspondientes a los bordes de una imagen anterior, que fueron almacenados en la SRAM externa, y que están situados en la misma fila y alrededor de la posición que ocupa el píxel actual. Almacenaremos tantos píxeles de bordes de la imagen anterior como constantes de velocidad deseemos implementar. Cuantas más constantes de velocidad mayor será el rango de velocidades para el que nuestro sistema es capaz de detectar movimiento, y esto será un parámetro que dependerá de las necesidades de la aplicación.

Los bordes de la imagen anterior se almacenan en una cola de datos con estructura de FIFO. Aunque el almacenamiento en tablas de consulta requiere de más lógica asociada y es menos eficiente que el uso de una RAM definida en bloques de memoria interna (block ram), como ya comentamos en la Sección 3.3.1, se hace necesaria la definición de un vector de datos usando recursos de lógica distribuida puesto que tenemos que acceder a todas las posiciones del vector en el mismo ciclo de reloj para realizar todas las correlaciones necesarias en paralelo y en el mismo ciclo de reloj (Figura 3.5). Esta es una limitación insuperable si se utilizan bloques de memoria interna RAM similar a la utilizada en la creación de los buffers de la Sección 3.3.1.

Debido a las necesidades de correlacionar píxeles de bordes procedentes de dos imágenes consecutivas, el sistema tiene como mínimo una imagen de latencia. Sin embargo

este problema es propio de cualquier algoritmo que requiera de un procesamiento temporal, y no perjudica al procesamiento en tiempo real.

### 3.3.4. Etapa de cómputo del máximo (“Winner takes all”) o etapa de estimación de la velocidad (S4)

La velocidad del píxel actual será la que maximice el valor de la correlación, y de esta tarea se encarga el módulo de procesamiento que describimos en esta sección. Para ello comparamos todas las correlaciones correspondientes a cada píxel de la imagen y buscamos el máximo de ellos.

Obtener el máximo se puede hacer de manera más eficiente después de que la lista de datos ha sido ordenada. Cuando se analiza el tiempo de ejecución de un método de ordenación, hay que determinar cuántas comparaciones e intercambios se realizan para el caso más favorable, para el caso medio y para el caso más desfavorable. Los distintos métodos de ordenación se pueden clasificar en: intercambio, selección e inserción. En cada familia se distinguen dos versiones: un método simple y directo, fácil de comprender pero de escasa eficiencia respecto al tiempo de ejecución, y un método rápido, más sofisticado en su ejecución por la complejidad de las operaciones a realizar, pero mucho más eficiente en cuanto a tiempo de ejecución. En general, para vectores con pocos elementos, los métodos directos son más eficientes (menor tiempo de ejecución) mientras que para grandes cantidades de datos se deben emplear los llamados métodos rápidos u optimizados. Estos algoritmos de ordenación son secuenciales y requieren de varias pasadas para completar la ordenación de la lista de elementos [GUE98].

El método de la burbuja es un método de intercambio directo. Se basa en comparar los elementos del vector e intercambiarlos si su posición actual o inicial es inversa a la deseada. Aunque no es muy eficiente para ordenar listas grandes, es fácil de entender y muy adecuado para ordenar una lista de unos 100 elementos o menos. La ordenación por el método de la burbuja consiste en una serie de pasadas ("burbujeo"), en cada una de las cuales se hace un recorrido completo a través del vector durante el que se comparan los contenidos de las casillas adyacentes, y se cambian si no están en orden. El proceso termina cuando se realiza una pasada en la que ya no se hacen cambios porque todo está en orden.

Si el vector tiene  $N$  elementos, para estar seguro de que el vector está ordenado, hay que hacer  $N-1$  pasadas, por lo que habría que hacer  $(N-1) \cdot (N-1)$  comparaciones, para cada  $i$  desde 1 hasta  $N-1$ . El número de comparaciones es, por tanto,  $N(N-1)/2$ , lo que nos deja un orden de complejidad algorítmica  $O(N^2)$ .

Si bien el método de la burbuja era considerado como el peor método de ordenación simple o menos eficiente, el método *quicksort* basa su estrategia en la idea intuitiva de que es

más fácil ordenar una gran estructura de datos subdividiéndolas en otras más pequeñas introduciendo un orden relativo entre ellas (“divide y vencerás”). En ella, se divide el vector en dos sub-vectores, para luego resolver cada uno por separado, y unirlos. El ahorro de tiempo es grande: el tiempo necesario para ordenar un vector de elementos mediante el método de la burbuja es cuadrático, como hemos visto. Si dividimos el vector en dos y ordenamos cada uno de ellos, el tiempo necesario para resolverlo es ahora la mitad, pero sigue siendo cuadrático. Si los subproblemas son todavía demasiado grandes, podemos dividirlos a ellos también, utilizando un algoritmo recursivo que vaya dividiendo más el sub-problema hasta que su solución sea trivial. Si aplicamos este método al *quicksort*, el tiempo disminuye hasta ser logarítmico, con lo que el tiempo ahorrado es mayor cuanto más aumenta N.

El tiempo de ejecución de un algoritmo de divide y vencerás,  $T(N)$ , viene dado por la suma de dos elementos:

➤ El tiempo que tarda en resolver los A subproblemas en los que se divide el original,  $A \cdot T(N/B)$ , donde  $N/B$  es el tamaño de cada sub-problema.

➤ El tiempo necesario para combinar las soluciones de los sub-problemas para encontrar la solución del original; normalmente es  $O(N^k)$

Por tanto, el tiempo total es:  $T(N) = A \cdot T(N/B) + O(N^k)$ . La solución de esta ecuación, si A es mayor o igual que 1 y B es mayor que 1, es:

$$\text{Si } A > B^k, T(N) = O(N^{\log_B A})$$

$$\text{Si } A = B^k, T(N) = O(N^k \cdot \log N)$$

$$\text{Si } A < B^k, T(N) = O(N^k)$$

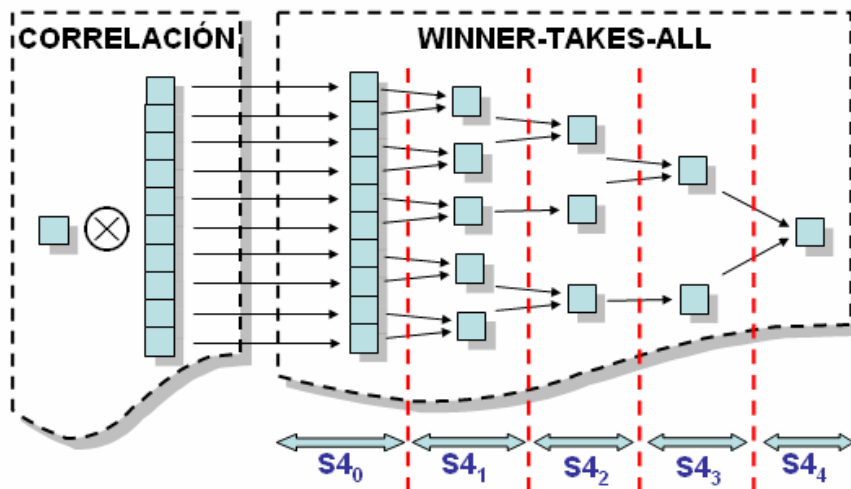
Uno de los aspectos que hay que tener en cuenta en los algoritmos de divide y vencerás es dónde colocar el umbral, esto es, cuándo se considera que un sub-problema es suficientemente pequeño como para no tener que dividirlo para resolverlo. Normalmente esto es lo que hace que un algoritmo de divide y vencerás sea efectivo o no. Por ejemplo, en el algoritmo de ordenación *quicksort*, cuando se tiene un vector de longitud 3, es mejor ordenarlo utilizando otro algoritmo de ordenación (con 6 comparaciones se puede ordenar), ya que el *quicksort* debe dividirlo en dos sub-vectores y ordenar cada uno de ellos, para lo que utiliza más de 6 comparaciones.

El esquema de procesamiento adoptado en nuestra aproximación se inspira en el algoritmo de la burbuja y el *quicksort*. En nuestro caso, para obtener un píxel procesado por cada ciclo de reloj no podemos permitirnos una estructura secuencial, y mucho menos realizar varias pasadas por cada lista. Para calcular el máximo lo más rápidamente posible (calcular la velocidad de un píxel cada ciclo de reloj, lo que supone calcular el máximo de N correlaciones cada ciclo de reloj) hay que integrar la arquitectura con un cauce de datos finamente segmentado. Hemos usado un esquema microsegmentado “winner-takes-all” (S4), como el que se muestra en la Figura 3.5.

Cuando hemos recibido las correlaciones del módulo anterior en la etapa  $S4_0$ , hacemos una comparación por pares de píxeles, es decir, cada píxel (representado mediante un cuadrado en la Figura 3.5) se compara con su vecino, y el que resulte ser máximo local (en esta primera etapa local) pasará a una segunda etapa ( $S4_1$ ); en esta nueva etapa se producirá la comparación de dos ganadores locales de la etapa anterior, y así sucesivamente ( $S4_2$  y  $S4_3$ ) hasta que en la última etapa ( $S4_4$ ) obtenemos finalmente el máximo.

Todas las comparaciones por pares de una misma etapa del camino de datos se producen en paralelo y en el mismo ciclo de reloj. Esta arquitectura “winner-takes-all” siguiendo un esquema de segmentación de cauce de grano fino (como se ilustra en la Figura 3.5) permite calcular el máximo de la velocidad de un píxel en un ciclo de reloj, pero con una latencia que dependerá del número de correlaciones que tengamos (en el ejemplo ilustrado en la Figura 3.5 la latencia es de 5).

La ventaja de esta estrategia de diseño es que es fácilmente escalable para distintas aplicaciones. Es decir, el rango de velocidades que necesitamos detectar es dependiente de la aplicación, y podemos aumentar o disminuir dicho rango de velocidades a los que el dispositivo es sensible de forma sencilla, teniendo en cuenta que esta variación influye en el área del dispositivo empleada. (Véanse los resultados de la Tabla 3.2 en la Sección 3.5).



**Figura 3.5.** Esquema de la estructura microsegmentada en las etapas de correlación y estimación de velocidad. Todos los procesos que se encuentran en la misma columna se realizan en paralelo y en el mismo ciclo de reloj.

Es interesante señalar el uso que en estas etapas se hace de los recursos de computación paralela propios de las FPGAs. En otros módulos de procesamiento se realizan en paralelo diferentes operaciones para obtener resultados parciales que en otra etapa posterior del camino de datos se van a utilizar (véase por ejemplo la Figura 3.4 que muestra las operaciones realizadas para la extracción de bordes mediante LIP-Sobel). Sin embargo aquí, los datos sufren una dispersión de los caminos de datos (con el proceso de correlación) para

luego converger todos los subcaminos de datos a un resultado único final que es uno de los datos originales, y todo ello tiene lugar en paralelo.

### **3.3.5. Selección de las características dominantes (S5)**

Como se explicó en la Sección 2.4.1 del Capítulo 2, hacemos una división de la imagen en áreas de influencia, cada una de las cuales tiene unas dimensiones de  $M$  filas por  $N$  columnas. Por lo que tendremos una matriz cuadrada de áreas, que contará de  $640/N$  columnas y  $480/M$  filas, donde  $640 \times 480$  es el área total de la imagen original que estamos procesando.

En cada área de influencia existe un conjunto de contadores. Cada conjunto cuenta con un número de contadores igual al rango de velocidades al que es sensible el sistema.

La información relevante que almacenamos en la etapa anterior para cada píxel es el valor de su velocidad. Por tanto, cada contador de una región se incrementa cuando el píxel procedente de una lectura de la imagen se mueve a la velocidad contabilizada por dicho contador.

Como los píxeles que llegan a este módulo proceden de una lectura progresiva de la imagen, tendremos activos  $640/N$  conjuntos de contadores, un conjunto por cada área de influencia ubicada en la misma fila. Cuanto mayor sea  $N$ , tendremos un número menor de áreas de influencia, y por tanto, de conjuntos de contadores, pero estas áreas de influencia serán de mayor tamaño, con lo que el sistema será menos sensible al movimiento coherente de objetos pequeños, como ya se vio en el Capítulo 2.

Cuando llegue el píxel correspondiente a la primera fila y primera columna de la imagen se incrementa en una unidad el contador que integra a los píxeles que se mueven a su velocidad en el conjunto de contadores perteneciente a la región primera, y así sucesivamente hasta que se llegue al píxel correspondiente a la fila 1 y columna  $(640/N)+1$ , en este caso se incrementará en una unidad el contador correspondiente a su velocidad, dentro del conjunto de contadores pertenecientes a la segunda región; y este proceso continúa, y a medida que realizamos la captura de la fila se va activando y actualizando el conjunto de contadores correspondientes a la región a la cual pertenece el píxel de entrada.

Cuando ha entrado el píxel de la última fila y última columna perteneciente a una región, y se ha incrementado su contador asociado, puede comenzar el cálculo de la característica dominante en esa región, en paralelo con la etapa de incremento de los contadores de restantes regiones.

Para determinar la característica dominante en cada región de la imagen tenemos que ver el contador que integra el mayor número de píxeles moviéndose a la misma velocidad y con la misma dirección. Una vez que se ha obtenido el máximo de los contadores de una región,

estos se actualizan a cero para ser reutilizados en el cómputo de características dominantes de la siguiente fila de regiones.

Para el cálculo de este máximo, podríamos usar una arquitectura igual a la empleada en el módulo de estimación de velocidad descrito con anterioridad. Sin embargo, como veremos en la Sección 3.5 el consumo de recursos en una etapa con ésta arquitectura es muy grande, y podría ser inviable el uso de dos arquitecturas iguales en una plataforma de tamaño medio si el rango de velocidades del sistema es amplio.

Para resolver el problema realizamos, para ésta etapa, el cálculo del máximo de forma secuencial. Usamos una estructura basada en el algoritmo de la burbuja, que en un número de ciclos de reloj igual al número de contadores obtiene el máximo. Este módulo secuencial se integra en paralelo con el resto del proceso de actualización de contadores.

Para obtener el máximo de una familia de contadores de forma secuencial, éste se debe realizar antes de que se necesiten actualizar a cero los contadores de esa región. Una familia de contadores necesita actualizarse a cero justo antes de volver a utilizarse, y esto sucede cuando se realiza la lectura de velocidad de los  $640-640/N-1$  píxeles que existen a continuación en la misma fila, es decir,  $640-640/N-1$  ciclos de reloj después de acabar de contabilizar en una región.

### **3.3.6. Módulo de gestión de memoria**

Los distintos módulos de procesamiento necesitan almacenar, o recuperar datos previamente almacenados, en los bancos de memoria SRAM externa. Este almacenamiento tiene dos funciones fundamentales, que el dato guardado sea postprocesado más tarde por otro bloque del circuito que lo tendrá que recuperar o la sincronización entre subsistemas bajo distintos dominios de reloj. En cualquier caso, estamos obligados a diseñar un módulo que gestione los accesos de forma eficiente sin que existan conflictos entre los módulos de procesamiento, y que permita escribir y leer de la SRAM tantas veces como sea necesario.

Cada proceso de escritura o lectura consume 2 ciclos de reloj por dirección de memoria a la que se accede (durante el primero se establece la dirección de memoria a la que se accede, y durante el segundo ciclo se lee o escribe el dato en dicha posición de memoria), por lo que es de gran interés manejar de forma eficiente los procesos de almacenamiento y recuperación de datos de la memoria externa. Esto es especialmente crítico en caminos de datos altamente segmentados.

Se ha introducido también una segmentación de cauce de grano fino en la gestión de la memoria de manera que, con una latencia de 2 ciclos somos capaces de leer y/o escribir en un único ciclo de reloj un dato en una posición de la memoria.



Cada dirección de la memoria puede alojar datos de longitud de palabra de 36 bits, lo cual hace posible almacenar 4 píxeles, cada uno con 9 bits, en cada dirección, es decir, por cada bloque de memoria de los que disponemos caben algo menos de 5 imágenes de 720x576 píxeles.

Por tanto, se hace necesario usar circuitos específicos de empaquetamiento y desempaquetamiento de datos. Dadas las características de la entrada de imágenes en el circuito, para que el almacenamiento de los mismos sea eficiente estamos limitados a empaquetar datos que sean consecutivos. Por eso trabajamos con una lectura progresiva de las sucesivas imágenes almacenadas, y toda la arquitectura se ha diseñado conforme a esta característica.

Por otro lado, dado que la placa de prototipado solamente dispone de dos bancos de memoria SRAM externos, necesitamos definir un doble puerto de lectura/escritura de manera que, en el mismo ciclo de reloj, un módulo de procesamiento pueda escribir en el mismo bloque de SRAM externa que otro módulo de procesamiento lee de otra zona distinta de la SRAM.

Hemos diseñado una unidad de gestión de memoria para cada uno de los chips de SRAM externa, cada módulo de gestión está basado en una máquina de estados, que, secuencialmente, lee y escribe los paquetes de datos (4 píxeles por paquete) en los puertos de lectura y escritura que gestiona en cada bloque de memoria. Con la arquitectura descrita sólo es necesario acceder a la memoria externa una vez cada 4 ciclos de reloj, para escribir o leer un paquete de 4 datos.

El gestor del bloque de memoria que hemos llamado SRAM-0 en la Figura 3.6, tramita 4 accesos a dicho bloque, dos de escritura y dos de lectura, y en cada uno se lee o escribe secuencialmente un bloque de cuatro píxeles. Con esta proporción de accesos a la memoria externa tenemos un nuevo píxel para procesar en el módulo de recepción por cada ciclo de reloj, y así tenemos saturado el número de accesos permitidos a ese bloque de memoria, que viene determinado por la capacidad de almacenamiento de píxeles en el paquete. Si aumentamos esta capacidad aumentará también el número de accesos de lectura/escritura permitidos.

Cuando el almacenamiento en el bloque de memoria externo persigue además la sincronización entre dominios de reloj tendremos la limitación que imponen las frecuencias de los relojes de los distintos dominios. Esto es lo que sucede en el bloque SRAM-0, donde además de almacenar píxeles preprocesados que serán recuperados posteriormente para ser postprocesados, tenemos como misión la sincronización del circuito de procesamiento que trabaja a 31.5 MHz ( $f_{circuito}$ ) con la cámara que captura las imágenes que procesamos, que suministra un píxel cada ciclo de reloj, con este funcionando a 6.5 MHz ( $f_{cámara}$ ). Según esto el número de accesos posibles a la memoria (N), sin sufrir pérdidas de píxeles de la cámara o de paquetes viene determinado por la Ecuación 3.1:

$$f_{cámara} \cdot N \leq f_{circuito} \quad (3.1)$$

Según esto, el proceso de sincronización nos limita a cuatro el número de accesos a la memoria externa que podemos hacer con las frecuencias de trabajo que estamos usando.

Este método de gestión de la memoria externa minimiza los cuellos de botella que encontramos en los accesos a memoria, que, por otro lado, son necesarios para el procesamiento.

## **3.4. Flujo de los datos y comunicación entre las etapas de procesamiento**

Como hemos visto con anterioridad, cada módulo de procesamiento de los descritos en las Secciones 3.3.1 a 3.3.6, se ha diseñado para trabajar en paralelo mediante una segmentación de cauce de grano grueso, cada uno de los módulos procesa el dato que recibe desde otra de las etapas de procesamiento. Este diseño modular permite añadir o quitar etapas de procesamiento de la imagen capturada por la cámara.

La comunicación entre los distintos módulos del camino de datos se realiza a través de canales. Los canales definidos en Handel-C por Celoxica, se usan para comunicación entre procesos separados para sincronizarlos y/o para pasar datos entre ellos. La sincronización se ve forzada por la forma de trabajo de los canales, ya que la escritura y lectura de un canal sólo tiene lugar en el mismo ciclo de reloj. Esto significa que el proceso de escritura en un canal debe esperar hasta que el dato puede ser leído por el otro proceso, y el proceso de lectura de un canal debe esperar a que el dato sea escrito por el otro proceso.

En nuestro sistema, puesto que todos los procesos funcionan bajo el mismo reloj los canales sólo realizan la actividad de paso de datos.

En la Figura 3.6 puede verse, codificado en colores, qué módulo comunica con otro. La naturaleza de los datos que recibe cada módulo depende del tipo de procesamiento que se vaya a realizar en ese módulo y de quién le comunique el dato a procesar.

El circuito de captación de imágenes se comunica con el módulo de empaquetamiento, éste envía a la SRAM externa un paquete de cuatro píxeles, cada uno de los cuales tiene un valor entre 0 y 255, este valor es la intensidad luminosa del píxel en la imagen original capturada por la cámara.

Cuando se realiza la lectura de la imagen original almacenada en el bloque 1 de la SRAM, el conjunto de píxeles se desempaquetan y se envían a los buffer de tipo FIFO que ha

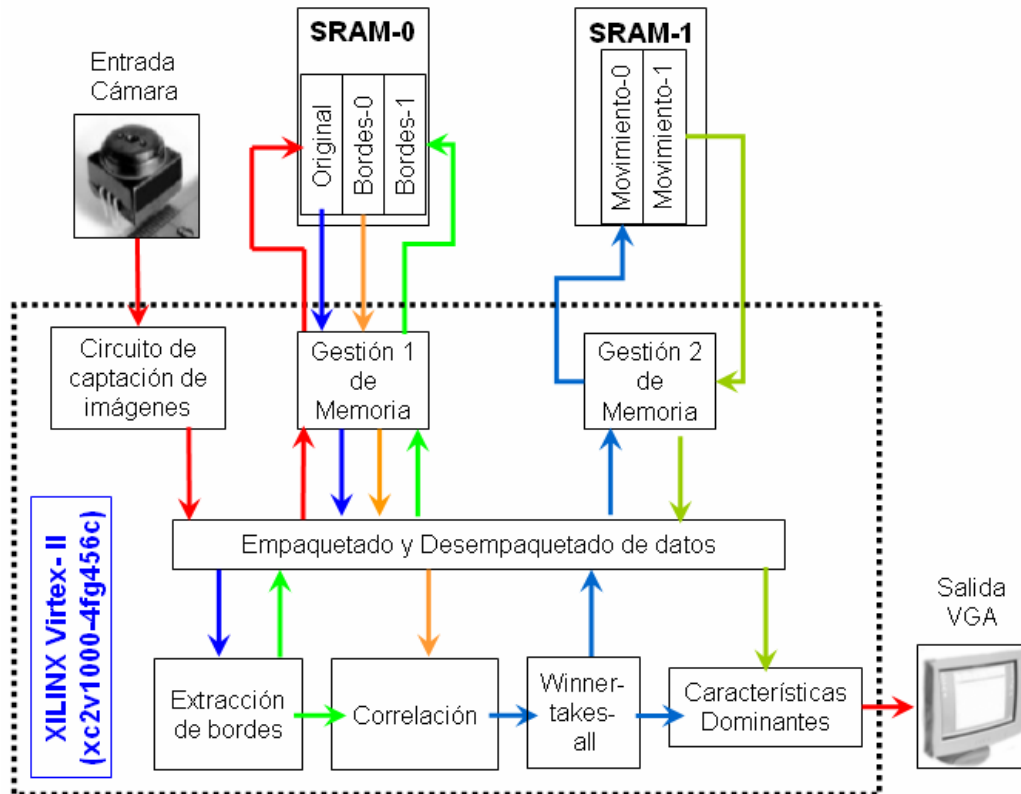
generado el circuito de captación. Cuando es posible disponer de una ventana de 3x3 píxeles de la imagen original en dichos buffers, se hace un envío mediante canal al módulo de extracción de bordes. Éste procesa la ventana de 3x3 y genera un dato cuyo valor está entre 0 y 255, y cuya naturaleza es la intensidad del borde detectado.

El borde detectado es enviado, por un lado al módulo de correlación, y por otro, al módulo de empaquetado de datos para almacenarlo en la SRAM externa, en la zona de la memoria denominada Borde-0. En paralelo, se recupera la información de bordes de una imagen anterior almacenada en la zona de la SRAM denominada Borde-1, se desempaqueta y se envía al módulo de detección de movimiento, para proceder a la correlación.

La actividad de las zonas de la SRAM denominadas Borde-1 y Borde-2 se alterna, esto es, se almacena la información de bordes de la imagen actual en la zona Borde-0, mientras que se lee la información de bordes de una imagen anterior en la zona Borde-1, y cuando se acaba el proceso de almacenamiento y lectura de dichas imágenes, la siguiente imagen entrante se almacena en Borde-1, mientras que se lee de Borde-0 la imagen almacenada, y así continúa el proceso de forma continua. Es decir, se utiliza una estrategia de doble buffer.

El módulo de correlación se comunica con el módulo de estimación de la velocidad o "winner-takes-all". El valor del píxel de salida del módulo de estimación de velocidad define la velocidad de movimiento de dicho píxel. Este dato es almacenado, por un lado en el bloque 2 de la SRAM externa, en la zona de memoria denominada Movimiento-0, previo empaquetamiento de un conjunto de cuatro datos; y por otro lado se envía al módulo de selección de características dominantes, donde se actualizan los contadores de la región de influencia a la que pertenece.

Mientras tanto, y en paralelo, el módulo de selección de características recibe desde el bloque 2 de la SRAM, de la zona de memoria denominada Movimiento-1, los valores de velocidad de los píxeles de una imagen anterior a la actualmente en procesamiento, estos valores son filtrados y enviados a la salida VGA. El proceso de filtrado se realiza comparando el valor de la velocidad del píxel recibido con los valores de la velocidad dominante en la zona a la que pertenece dicho píxel en la imagen anterior, dichos valores se mantienen almacenados en bloques de memoria embebida.



**Figura 3.6.** Flujo de Datos entre los distintos módulos de procesamiento dentro de la FPGA y con los bloques de memoria externa SRAM, la cámara y la salida VGA.

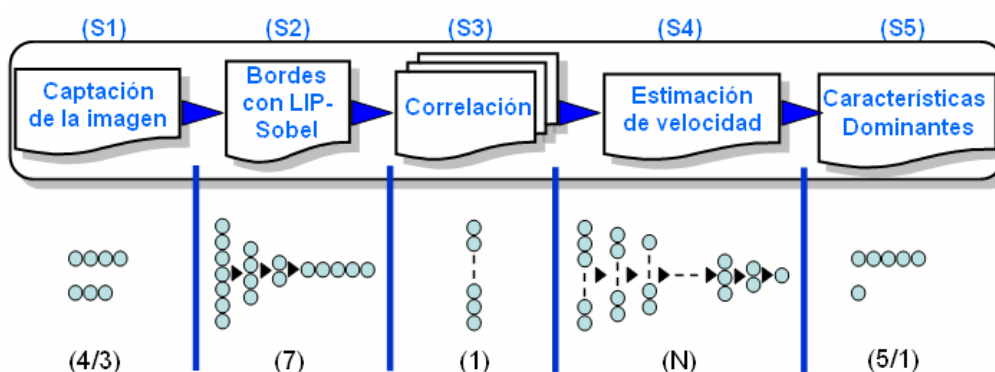
La gestión de las regiones del bloque segundo de SRAM denominadas Movimiento-0 y Movimiento-1, se realiza con doble buffer, es decir de forma similar a la gestión realizada con Bordes-0 y Bordes-1 en el primer bloque de SRAM.

El flujo de datos hacia/desde la memoria tiene dos objetivos, como se ha comentado en la sección anterior, en primer lugar es necesario almacenar datos preprocesados que serán requeridos posteriormente por otros módulos de procesamiento. Esto es lo que se persigue guardando datos en la SRAM 0. El gestor 1 de este bloque de memoria sincroniza los accesos en paralelo a la misma del circuito de captación de imágenes, el circuito de extracción de bordes y el de correlación. El almacenamiento y la recuperación de datos están precedidos siempre por el proceso de empaquetamiento y desempaquetamiento de los datos en bloques de cuatro.

Por otro lado, se hace necesario guardar datos como técnica de sincronización entre subsistemas que funcionan bajo diferentes relojes, esta es la tarea que realiza el gestor 2 de la SRAM 1 para comunicar la salida del circuito con la VGA. En este caso se transfieren los datos a la memoria externa y luego se recuperan dichos datos y se comunican a la VGA a través de un buffer de sincronización. También se almacenan las imágenes de entrada (imágenes originales) para sincronizar el circuito con la cámara, y de esta tarea de sincronización se encarga el gestor 1.

En la Figura 3.7 se muestran las distintas unidades superescalares (representadas por círculos) que forman el camino de datos microsegmentado de los módulos de procesamiento. Algunas de estas unidades se componen de operaciones internas que se realizan en paralelo para obtener una estimación final en un ciclo de reloj, otras realizan una única operación. Por tanto, el nivel de paralelismo en cada módulo se representa de una manera esquemática. Los círculos en la misma columna son unidades superescalares que se procesan en paralelo en el mismo ciclo de reloj.

Se observa que el cauce de datos tiene un número de etapas que depende del número de correladores que tenga el sistema, es decir, del rango de velocidades a las que es sensible.



**Figura 3.7.** En la parte inferior de la figura se representan los caminos de datos para los distintos módulos de procesamiento. Cada uno de los círculos representa una unidad superescalar del cauce de datos microsegmentado, que tiene lugar en un ciclo de reloj. Los círculos que están en la misma columna de un módulo representan etapas de procesamiento que se ejecutan en paralelo y en el mismo ciclo de reloj. Entre paréntesis se indica el número de etapas que requiere cada módulo de procesamiento, que coincide con el número de unidades superescalares que hay en la horizontal de cada etapa. En la etapa de correlación (S3) existirán tantas unidades superescalares ejecutándose en paralelo, en el mismo ciclo de reloj, como correladores existan en el sistema. Del número de correladores depende la cantidad de etapas necesarias para realizar el procesamiento en S4.

### 3.5. Recursos de Hardware consumidos

En la Tabla 3.1 se exponen los resultados sobre recursos utilizados en el módulo de extracción de bordes mediante LIP-Sobel. Se presentan las distintas posibilidades que se han explorado durante la implementación hardware: con y sin el uso de un *core* para realizar la división, y con y sin el uso de multiplicadores embebidos para realizar las multiplicaciones (la Virtex-II tiene 40 multiplicadores embebidos).

Se observa que cuando se utiliza un divisor combinacional el problema fundamental está en que el circuito funciona a 3.862 MHz, según las estimaciones del sistema ISE. Al hacer uso de un *core* optimizado conseguimos que el circuito funcione a 66 MHz, con lo que la

división no nos limita el rendimiento global del sistema. Además, el uso del *core* no sólo mejora el rendimiento del circuito en cuanto a velocidad, sino que el consumo de hardware es menor

El uso de los multiplicadores embebidos no mejora la frecuencia de trabajo del circuito, sino que está en el mismo orden de magnitud. Sin embargo, sí se optimiza el consumo de recursos, como se puede observar en la Tabla 3.1 existe un decremento en el tanto por ciento de ocupación del dispositivo en todos los casos en los que se usan multiplicadores embebidos con respecto a la no utilización de los mismos, sin embargo, la migración a otros dispositivos FPGA de familias más baratas que no posean multiplicadores embebidos no será directa.

**Tabla 3.1.** Comparativa en el consumo de recursos en la extracción de bordes mediante LIP-Sobel con y sin el uso de un *core* para realizar la división, y con y sin el uso de los multiplicadores embebidos.

	Número de Slices	% ocupación del dispositivo	Nº de multiplicadores embebidos	ISE Max. Frecuencia de reloj (MHz)
<b>Sin core y sin</b> multiplicadores	3379	65	0	3.714
<b>Sin core y con</b> multiplicadores	3029	59	6	3.799
<b>Con core y sin</b> multiplicadores	1958	38	0	64.045
<b>Con core y con</b> multiplicadores	1574	30	6	88.386

Como se comentó con anterioridad, podemos seleccionar el rango de velocidades que el sistema es capaz de detectar. Sin embargo, durante la etapa de selección de la velocidad el consumo de recursos se verá incrementado. Como se observa en la Tabla 3.2, cuando se duplican las velocidades detectadas, aproximadamente, también se duplican los recursos consumidos por ésta etapa.

**Tabla 3.2.** Aumento en el consumo de recursos con el incremento en el rango de velocidades en la etapa de selección de la velocidad.

Rango de velocidades	Número de Slices	% ocupación del dispositivo
16	338	6
32	699	13
48	1312	25

La Tabla 3.3 resume el rendimiento y el coste de hardware de las diferentes etapas de procesamiento implementadas y descritas anteriormente.

Es necesario destacar que el uso de una arquitectura que aprovecha al máximo los recursos de procesamiento paralelo de la FPGAs (segmentación de cauce de grano fino) hace que el dispositivo funcione en una FPGA de 1 millón de puertas lógicas equivalentes,

consumiéndose tan sólo el 79% de los *slíces* totales en el caso del Sistema 1 (Tabla 3.4). Esto hace posible el uso de otros dispositivos con un número menor de recursos, la introducción de nuevos módulos de procesamiento específicos para una aplicación concreta, o incluso el procesamiento de imágenes de mayor resolución con unidades de procesamiento en paralelo para las distintas partes de la imagen. Sin embargo, el Sistema 2, que implementa el circuito completo usando la extracción de bordes mediante LIP-Sobel, ocupa más *slíces* de los disponibles en el dispositivo XC2V1000, por tanto se ha usado un dispositivo con mayores recursos, el XC2V3000, y los datos de utilización del dispositivo en la Tabla 3.4 hacen referencia a ese chip.

La estructura del camino de datos de grano grueso mostrada en la Figura 3.2 proporciona un píxel de salida por cada ciclo de reloj, para todas las etapas de procesamiento, gracias al uso de una segmentación de cauce de grano fino.

Las limitaciones del sistema se deben al necesario acceso a los bancos de memoria externos. A pesar de esta limitación, el sistema está procesando imágenes de 640x480 píxeles con una tasa de procesamiento de 103 imágenes por segundo, o hasta 31641 Kilopíxeles por segundo adecuando el tamaño de las imágenes a procesar y la tasa de transferencia de las mismas al dispositivo, cuando el reloj del circuito trabaja a 31.5 MHz. En la Tabla 3.4 podemos comprobar que el diseño admite la utilización de un reloj de trabajo de mayor frecuencia que permitiría procesar hasta 135 imágenes por segundo de 640x480 píxeles o hasta 41472 Kilopíxeles por segundo de imágenes de tamaño y tasa de transferencia convenientes.

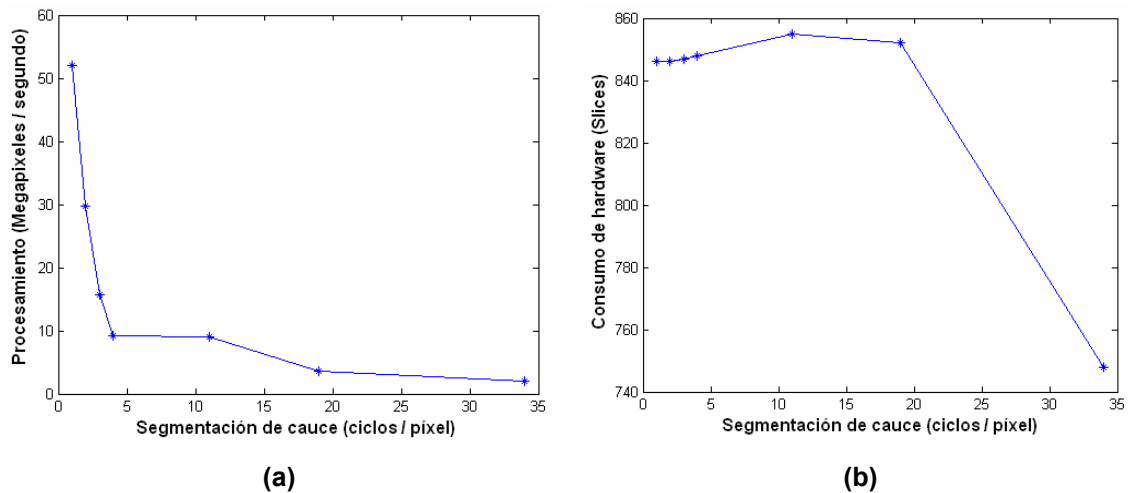
Esta capacidad de procesamiento supera ampliamente las necesidades del sistema descrito, donde la entrada la proporciona una cámara que adquiere 25 imágenes por segundo. Sin embargo, la capacidad de procesamiento del circuito posibilita su uso en otras aplicaciones que requieran de cámaras de mayor velocidad de adquisición (existen cámaras en el mercado cuya velocidad de adquisición sobrepasa las 1000 imágenes por segundo, con tamaños de adquisición de hasta 1280x1024 píxeles por imagen [MOT06]) o de mayores tamaños de imagen.

Como se ha comentado con anterioridad, la arquitectura con microsegmentación del cauce de datos aprovecha el procesamiento paralelo de los dispositivos FPGA, lo que hace viable el procesamiento de 1 píxel en cada ciclo de reloj, proporcionando una alta velocidad de procesamiento. Sin embargo, puede ser necesaria la utilización de más de un ciclo de reloj para que algún módulo de procesamiento (añadido al sistema para una aplicación específica) tenga la funcionalidad deseada. Por ello, es interesante estudiar la degradación del circuito, en cuanto a recursos consumidos y tasa de procesamiento, cuando se emplea más de un ciclo de reloj para procesar cada píxel. A modo de ejemplo vamos a ver la degradación del circuito en el caso del módulo S4 (estimación de la velocidad).

En la Figura 3.8a se observa la cantidad de Megapíxeles por segundo procesados cuando en la etapa de estimación de velocidad se utiliza un número de ciclos mayor de uno. El sistema se degrada mucho y muy rápidamente, siendo necesario reducir o bien el tamaño de

las imágenes, o bien el número de imágenes por segundo que son transferidas al dispositivo, o adoptar ambas soluciones, para que el circuito sea capaz de responder de forma apropiada y en tiempo real.

En la Figura 3.8b podemos observar cómo afecta el incremento en el número de ciclos por píxel en el consumo de recursos (número de Slices) por el circuito. Se observa un notable descenso en los slices utilizados cuando hay un gran incremento en el número de ciclos empleados por cada píxel procesado que se debe al gran número de variables que ya no es necesario definir dado que al realizarse el procesamiento de forma secuencial se reutilizan las mismas en cada ciclo. También existe un ligero aumento en el número de recursos consumido cuando son pocos los ciclos por píxel posiblemente debido a que es necesario realizar un ruteo adicional que no se compensa por el decremento (mínimo en esos casos) en el número de variables utilizadas en el procesamiento.



**Figura 3.8.** Degradación de las características del circuito cuando aumenta el número de ciclos empleados en procesar un píxel en la etapa de estimación de velocidad (S4).

El entorno ISE nos proporciona los resultados de todas las tablas. Las estimaciones de dicha tabla son aproximaciones ya que han sido extraídas de la compilación de sub-diseños realizadas con el entorno ISE. Cuando el sistema se compila como un todo muchos de los recursos se comparten, reduciéndose de esta forma el coste total de hardware, como puede observarse en la Tabla 3.4.

La estrategia de diseño, haciendo uso de un camino de datos finamente segmentado, proporciona 1 píxel procesado por cada ciclo de reloj. Esta estrategia permite que todo el dispositivo FPGA esté funcionando en paralelo, no existiendo circuitos ociosos en el sistema. Esto es importante ya que en los procesadores de propósito general la mayoría de los circuitos “están ociosos” durante la ejecución de instrucciones concretas.



**Tabla 3.3.** Coste del procesamiento de las diferentes etapas descritas. El reloj global del diseño trabaja a 31.5 MHz, aunque la tabla incluye un campo en el que se puede ver la máxima frecuencia de trabajo permitida en cada etapa (según las estimaciones del entorno ISE). Procesamos 640x480 píxeles; discriminamos en un rango de 33 velocidades; en la etapa de correlación usamos 33 multiplicadores embebidos; Kpps indica el número de Kilopíxeles por segundo que puede procesar el módulo y Ips indica las imágenes por segundo que somos capaces de procesar en ese módulo cuando la imagen tiene un tamaño de 640x480 píxeles.

Etapas del camino de datos	Nº de Slices (% recursos)	Nº de blockram (% recursos)	Multiplicadores Embebidos (% recursos)	Ips	Kpps
Captación de la imagen	753 (14)	2 (5)	0 (0)	247	75878
Extracción de bordes	21 (0.4)	0 (0)	0 (0)	207	63590
Correlación	373 (6)	0 (0)	33 (82)	170	52224
Selección de la velocidad	699 (13)	0 (0)	0 (0)	170	52224
Características dominantes	1675 (32)	16 (40)	0 (0)	151	46387
Gestión de la SRAM	581 (11)	0 (0)	0 (0)	175	53760

**Tabla 3.4.** Coste del procesamiento de los dos sistemas propuestos. Ambos sistemas procesan 640x480 píxeles por imagen, usan multiplicadores embebidos y son capaces de detectar un rango de 33 velocidades distintas. El Sistema 1 extrae bordes mediante el módulo de Sobel lineal, mientras que el Sistema 2 extrae bordes usando LIP-Sobel. El Sistema 1 se ha implementado en un chip Virtex-II XC2V1000, mientras que el Sistema 2 se ha implementado en el chip Virtex-II XC2V3000. Kpps indica el número de Kilopíxeles por segundo y Ips indica las imágenes por segundo que puede procesar el sistema cuando la imagen tiene un tamaño de 640x480 píxeles y el circuito trabaja a la frecuencia de funcionamiento máxima permitida según las estimaciones de la herramienta ISE.

	Nº de Slices (% recursos)	Nº de blockram (% recursos)	Multiplicadores embebidos (% recursos)	Ips	Kpps
Sistema 1	4084 (79)	18 (45)	33 (82)	135	41472
Sistema 2	6397 (22)	18(18)	39(40)	117	35942

## 3.6. Conclusiones

- En este capítulo se ha descrito la implementación en hardware específico (circuito FPGA) del algoritmo de detección de movimiento.
- Todo el sistema de procesamiento funciona en un solo circuito programable (PSoC) y en tiempo real.

- ☑ El sistema es modular, siendo posible sustituir módulos de procesamiento o introducir módulos adicionales que particularicen el procesamiento para alguna aplicación, y que se conectarían de forma natural a los módulos existentes. Un ejemplo de la modularidad del circuito la constituye la posibilidad de sustituir el módulo de extracción de bordes mediante el detector lineal de Sobel por el módulo de detección logarítmico de Sobel.
- ☑ El circuito se ha diseñado mediante una arquitectura de camino de datos de grano grueso compuesto por los distintos módulos de procesamiento.
- ☑ Todos los módulos son independientes entre si y se ejecutan en paralelo, La comunicación entre ellos se realiza mediante canales de comunicación específicos.
- ☑ Cada uno de los módulos de procesamiento se han diseñado como una arquitectura de propósito específico con un esquema microsegmentado del cauce de datos.
- ☑ El flujo regular de datos ha permitido el diseño de una arquitectura microsegmentada, lo que posibilita que el procesamiento de un píxel en cada ciclo de reloj.
- ☑ Esta estrategia de diseño permite que todo el dispositivo FPGA esté funcionando en paralelo, no existiendo circuitos ociosos en el sistema.
- ☑ Un circuito más lento, que requiera de más ciclos de reloj para procesar cada píxel, seguirá trabajando en tiempo real, pero el
- ☑ Las limitaciones del sistema se deben al necesario acceso a los bancos de memoria externos, que se usan para almacenar datos preprocesados y para sincronizar circuitos que funcionan bajo diferente reloj.
- ☑ El sistema es capaz de procesar imágenes de 640x480 píxeles con una tasa de procesamiento de 103 imágenes por segundo y 31641 Kilopíxeles por segundo, cuando el reloj del sistema trabaja a 31.5 MHz.
- ☑ Se ha realizado un estudio del consumo de recursos para distintas arquitecturas de los módulos de procesamiento y para el sistema completo, así mismo se ha estudiado la degradación del sistema cuando se utilizan arquitecturas alternativas que emplean más de un ciclo de reloj por cada píxel procesado.



---

## **CAPÍTULO 4:**

# **Evaluación del algoritmo de detección de movimiento**

---

En este capítulo se evalúa el algoritmo que se ha presentado en los capítulos anteriores. Como las imágenes con las que vamos a trabajar son reales no conocemos de antemano el movimiento que realiza cada píxel, por tanto, el primer paso será el diseño de un sistema de evaluación específico para trabajar con secuencias reales. Las secuencias que vamos a utilizar nos permitirán evaluar el algoritmo en situaciones con un grado de dificultad creciente. Durante la evaluación estudiaremos la influencia que tiene el valor de los parámetros de configuración del algoritmo sobre la salida final del procesamiento, y obtendremos los valores que permiten el máximo rendimiento del algoritmo en cada una de las situaciones bajo estudio.

## 4.1. Introducción

En este capítulo se evalúa la eficiencia del algoritmo descrito en secciones anteriores mediante el programa Matlab.

La definición del algoritmo en Matlab se ha hecho de forma que no hay ninguna diferencia entre los resultados numéricos de las versiones software y hardware. Los valores numéricos de la velocidad para cada píxel son idénticos en ambos casos, ya que todas las operaciones que se realizan son con enteros sin signo, y, al no trabajar en coma flotante, conservamos la profundidad en el número de bits de las distintas variables obtenidas a lo largo del procesamiento para que no influya en la precisión de los resultados finales. Por tanto, no existe degradación de los resultados en etapas sucesivas, y las únicas diferencias radican en la arquitectura utilizada en ambas aproximaciones.

Como en el modelo Software hemos usado el entorno Matlab estamos aplicando una arquitectura basada en matrices, mientras que la arquitectura del modelo hardware sigue otros patrones basados en los recursos de segmentación de cauce y paralelismo propios de las FPGAs, como se ha expuesto en el Capítulo 3.

Para realizar la evaluación se pueden utilizar secuencias sintéticas pero, teniendo en cuenta las aplicaciones finales del sistema, hemos preferido desarrollar un esquema de evaluación con secuencias reales que nos permitan evaluar distintas configuraciones del sistema en escenarios reales relacionados con aplicaciones industriales.

El procedimiento seguido consiste en segmentar manualmente las secuencias utilizadas para la evaluación, es decir, marcar y etiquetar manualmente los objetos que se mueven en cada imagen de la secuencia. Usaremos secuencias con distinto grado de dificultad.

Vamos a determinar la bondad de nuestro detector de movimiento mediante una serie de parámetros que definiremos a continuación: el *Nivel de Clasificación*, la *Sensibilidad Derecha* y la *Sensibilidad Izquierda*.

Como hemos visto anteriormente, nuestro algoritmo es capaz de detectar bordes moviéndose hacia la derecha y bordes moviéndose hacia la izquierda. Además, en la escena tenemos varios objetos de movimiento conocido (etiquetados manualmente). Por tanto, podemos clasificar el movimiento de las características detectadas en las siguientes categorías:

**Verdadero Derecha (VD)**. Son aquellos bordes que, por pertenecer al objeto con movimiento a la derecha, se etiquetan con movimiento hacia la derecha, y el algoritmo los detecta moviéndose hacia la derecha. Es **Derecha** porque el movimiento del objeto segmentado manualmente es hacia la derecha; y es **Verdadero** porque la dirección de

movimiento del borde detectado coincide con la del objeto al que pertenece (es decir, es una detección correcta).

**Falso Derecha (FD).** Son aquellos bordes que, por pertenecer al objeto con movimiento a la derecha, se etiquetan con movimiento a la derecha, y el algoritmo los detecta moviéndose hacia la izquierda. En este caso decimos que es **Falso** porque la dirección de movimiento del borde detectado no coincide con la del objeto al que pertenece (detección incorrecta).

**Verdadero Izquierda (VI).** Son aquellos bordes que, por pertenecer al objeto con movimiento a la izquierda, se etiquetan con movimiento a la izquierda, y el algoritmo los detecta moviéndose hacia la izquierda. Es **Izquierda** porque el movimiento del objeto segmentado manualmente es hacia la izquierda; y es **Verdadero** porque la dirección de movimiento del borde detectado coincide con la del objeto al que pertenece (detección correcta).

**Falso Izquierda (FI).** Son aquellos bordes que, por pertenecer al objeto con movimiento a la izquierda, se etiquetan con movimiento a la izquierda (**Izquierda**), y el algoritmo los detecta moviéndose hacia la derecha (**Falso**) (detección incorrecta).

Con las definiciones anteriores se pueden definir: el *Nivel de Clasificación*,  $NC$  (Ecuación 4.1); la *Sensibilidad Derecha*,  $SE_D$  (Ecuación 4.2), y la *Sensibilidad Izquierda*,  $SE_I$  (Ecuación 4.3):

$$NC = \frac{VD + VI}{VD + FD + VI + FI} \quad (4.1)$$

$$SE_D = \frac{VD}{VD + FD} \quad (4.2)$$

$$SE_I = \frac{VI}{VI + FI} \quad (4.3)$$

Nótese que el *Nivel de Clasificación* representa la relación entre los bordes de los objetos segmentados manualmente que están bien caracterizados por el algoritmo de detección en cuanto a dirección del movimiento, y el total de los bordes etiquetados (los bordes pertenecientes a los objetos segmentados de la escena). La *Sensibilidad Derecha* ( $SE_D$ ) representa la proporción de bordes correctamente detectados en el objeto con movimiento hacia la derecha, es decir, la relación entre los bordes que están bien caracterizados por el algoritmo, y el total de los bordes detectados en el objeto con movimiento hacia la derecha. La *Sensibilidad Izquierda* ( $SE_I$ ) tiene la misma interpretación que la  $SE_D$ , pero referida al objeto que se mueve hacia la izquierda.

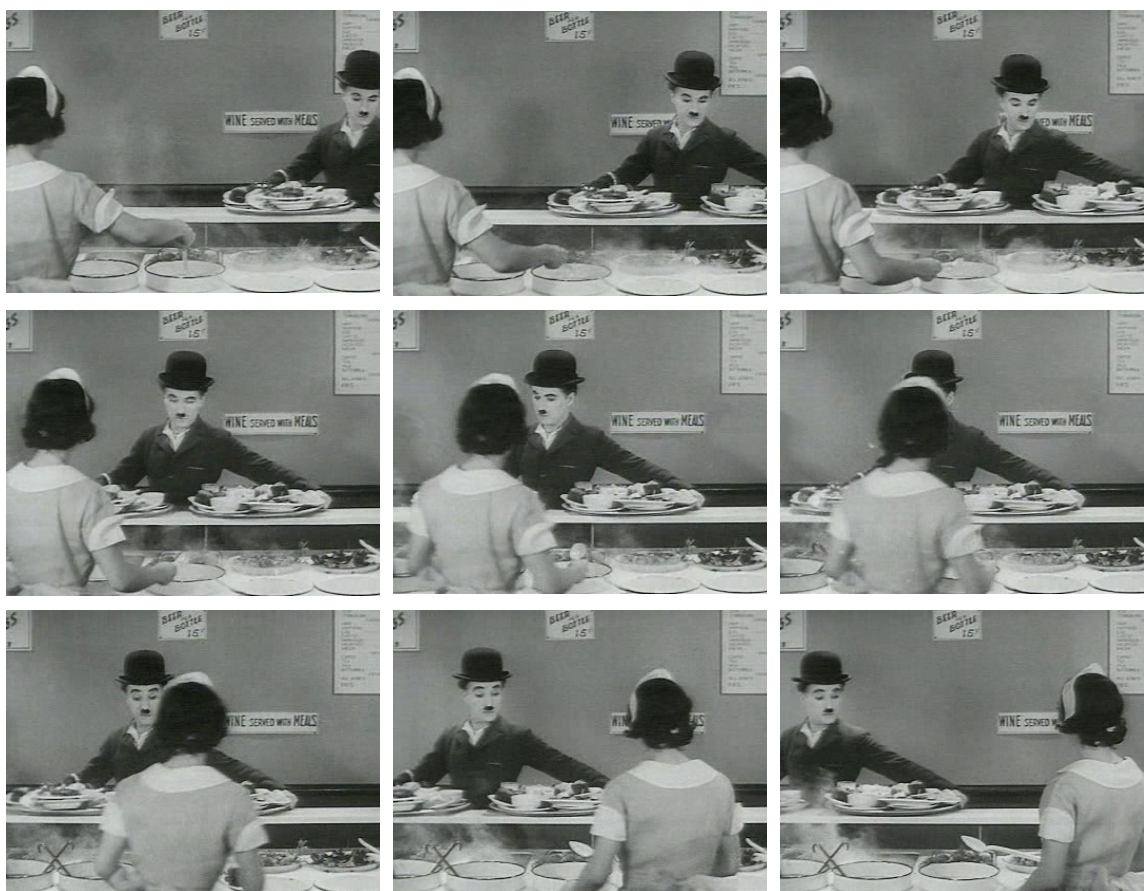
Definimos la sensibilidad en ambas direcciones porque en muchos escenarios en los que la cámara es móvil la sensibilidad a ambas direcciones no será simétrica (véase la sección 4.4 del presente capítulo o el capítulo 5).

## 4.2. Evaluación I

La primera secuencia, que ya se describió en el Capítulo 2, ha sido extraída de la película “Tiempos Modernos” de Charles Chaplin (1936). Esta película tiene una tasa de transferencia de 25 imágenes por segundo, donde cada imagen tiene 360 x 270 píxeles.

En la Figura 4.1 se muestran algunas imágenes de la misma. Podemos ver dos personajes moviéndose por una escena en la que todos los elementos son inmóviles. La cámara es estática, y presenta un plano de medio cuerpo de los personajes, por tanto, el tamaño de los mismos con respecto a la escena total es grande. Ambos personajes realizan un movimiento en la dirección perpendicular a la cámara. El personaje de la camarera se mueve hacia la derecha de la imagen, mientras que el personaje de Chaplin se desplaza hacia la izquierda de la imagen. El tamaño de ambos personajes se mantiene aproximadamente constante a lo largo de la secuencia.

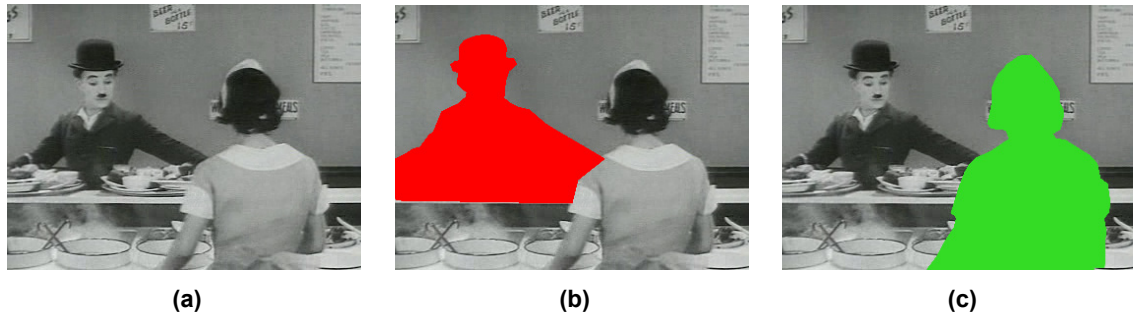
Ambos personajes se mueven a diferente velocidad, Chaplin mantiene la velocidad constante, mientras que la camarera tiene una velocidad muy variable y mucho mayor que la de Chaplin.



**Figura 4.1.** Selección de algunas imágenes correspondientes a la película “Tiempos Modernos” de Charles Chaplin, que vamos a utilizar para realizar la primera evaluación del algoritmo de detección de movimiento.

Como se ha expuesto en la sección anterior, vamos a segmentar manualmente los objetos que se mueven en la escena. Como se puede observar en las imágenes extraídas de la secuencia, estos objetos son: el personaje de Chaplin, que se mueve hacia la izquierda, y la camarera, cuyo movimiento es hacia la derecha.

En la Figura 4.2 se puede ver un ejemplo de la segmentación manual realizada en las distintas imágenes de la secuencia. Nótese que en la escena existe una zona de superposición de ambos objetos, y por tanto, esta zona es común a ambos.



**Figura 4.2.** Ejemplos de segmentación manual sobre los objetos que se desplazan por la escena. (a) Imagen real; (b) y (c) Imágenes segmentadas manualmente.

A continuación estudiaremos el efecto que, sobre el *Nivel de Clasificación*,  $NC$ ; la *Sensibilidad Derecha*,  $SE_D$ , y la *Sensibilidad Izquierda*,  $SE_I$ , tienen distintos valores de los parámetros de configuración, con ello podremos escoger dichos parámetros de forma que su efecto sobre la eficiencia del algoritmo sea óptimo.

Vamos a estudiar los siguientes parámetros del algoritmo:

- ❖ Umbralización de los bordes (Sobel)
- ❖ Tamaño de la máscara de convolución de Sobel
- ❖ Tamaño de la estructura de correlación
- ❖ Tamaño de las áreas de influencia en el filtrado por coherencia espacial
- ❖ Efecto de los parámetros sobre el ruido de fondo

La secuencia bajo estudio tiene más de 70 fotogramas, y aunque los resultados mostrados aquí se refieren solamente a 7 de estos fotogramas, son extensibles al resto de los de la secuencia.

#### a) Umbralización de los bordes:

Se ha visto con anterioridad (Sección 2.4) que la umbralización en la etapa de los bordes tiene un papel fundamental en la densidad de los mismos (Figura 2.8). Un aumento del umbral significa un descenso, a veces significativo, en el número de bordes detectados.



La umbralización afecta primero a aquellos puntos donde la respuesta a la máscara de Sobel ha sido débil, pero ¿afecta la umbralización, o sea la densidad de píxeles de bordes, a los valores de  $NC$ ,  $SE_D$  y  $SE_I$ ?

Hemos estudiado este fenómeno mediante la secuencia mostrada en la Figura 4.1. Como se muestra en la Figura 4.3, a medida que aumentamos el umbral, aumentan también los valores de  $SE_D$  y  $SE_I$ , mientras que el número de píxeles que se han procesado disminuye. Esto se debe a que, como vimos en la Sección 2.4, los bordes que comienzan a ser filtrados son fundamentalmente bordes débiles que no pertenecen al objeto principal, y que son, posiblemente, la fuente de las detecciones erróneas.

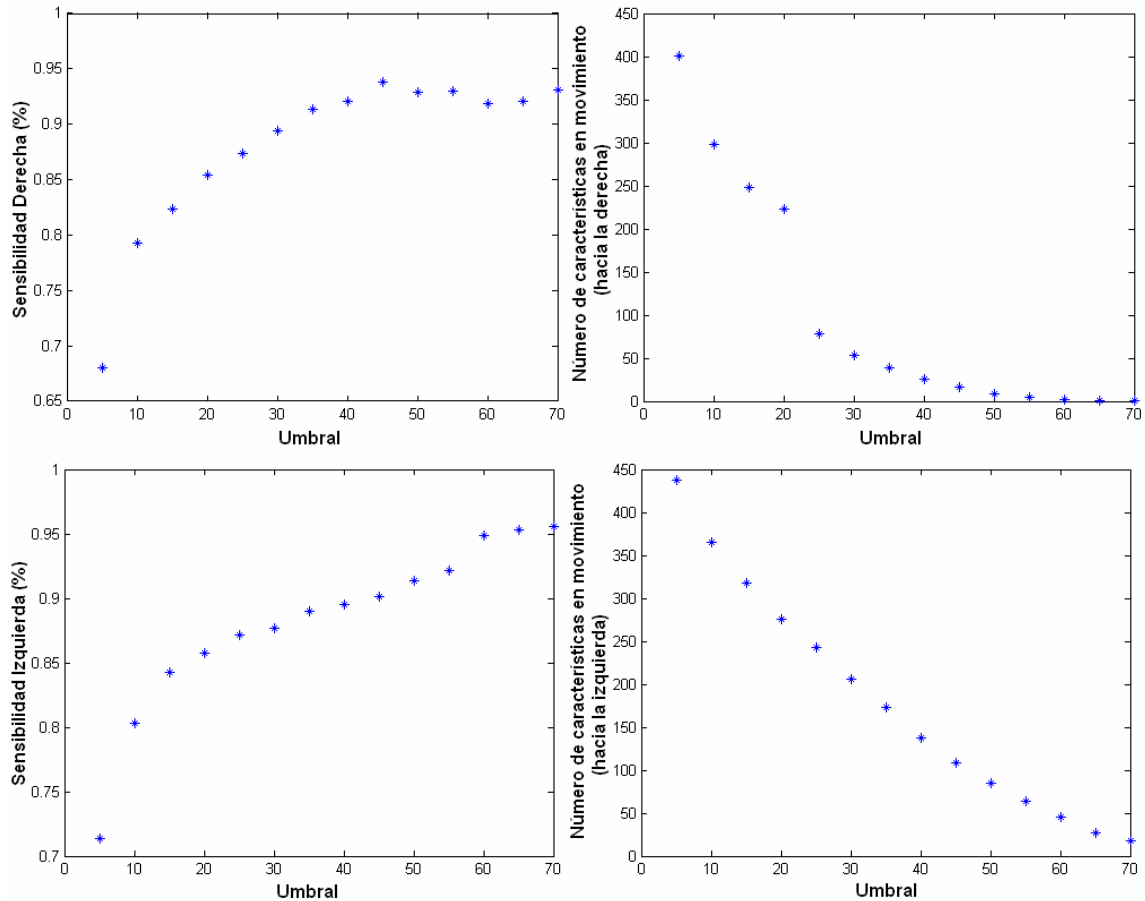
Sin embargo, no podemos aumentar el umbral indefinidamente para obtener niveles de clasificación cada vez mayores, porque llega un momento en que la umbralización también afecta al objeto principal. Una umbralización excesiva puede dar como resultado un Nivel de Clasificación del 100%, basado en un número de píxeles activos muy reducido, en ocasiones igual a 1 ó 2. Habrá que tener en cuenta este hecho, y buscar el punto de equilibrio entre umbralización, niveles de clasificación y número de píxeles activos. Recuérdese que esta es sólo la primera etapa de procesamiento, y los píxeles activos de esta etapa serán los que nos permitan detectar el movimiento de la secuencia en etapas de procesamiento posteriores.

Dadas las dimensiones de los objetos que estamos detectando en la secuencia mostrada en la Figura 4.1, la umbralización debería ser tal que el número de píxeles (bordes) en movimiento sea suficiente, por ejemplo que no sea inferior a 100. Por tanto, el umbral escogido podría estar situado entre 20 y 30, con lo que los niveles promedio de  $SE_D$  y  $SE_I$  estarán por encima del 85%, según se muestra en la Figura 4.3.

Hay que destacar que estos valores de sensibilidad se han obtenido con un valor arbitrario (no optimizado) del conjunto de los parámetros que vamos a estudiar a continuación, y que caracterizan el rendimiento del algoritmo de detección de movimiento, por lo que, aunque el valor de  $SE_D$  y  $SE_I$  no es el mayor de los que se observan en la figura, este debería mejorar en cuanto optimicemos los restantes parámetros.

Como ya se indicó en el Capítulo 2, la umbralización dependerá de la iluminación de la escena, y por tanto, éste es un factor dependiente de la aplicación.

Obsérvese que después de la extracción de bordes y la umbralización estamos reduciendo la cantidad de píxeles iniciales de una forma drástica, manteniendo activos en torno al 0.1% de los píxeles de la imagen inicial (97200 píxeles), que son los que usaremos para la estimación del movimiento.



**Figura 4.3.** Las gráficas de la izquierda representan el promedio en la secuencia de  $SE_D$  y  $SE_I$  frente al umbral que se aplica en la etapa de extracción de bordes. Las gráficas de la derecha representan el promedio en la secuencia del número de bordes que se han detectado formando parte de los objetos en movimiento.

**b) Tamaño de la máscara de convolución de Sobel:**

Como vimos en la Sección 2.1, los operadores de gradiente, como el operador de Sobel, son sensibles al ruido tipo ‘sal y pimienta’, y el uso de máscaras de tamaño mayor podría ayudar a mejorar la detección de bordes.

Para comprobar esto, vamos a comparar los resultados de los niveles de  $NC$ ,  $SE_D$  y  $SE_I$  cuando se usa una máscara de 3x3 y de 7x7. Usaremos la secuencia mostrada en la Figura 4.1.

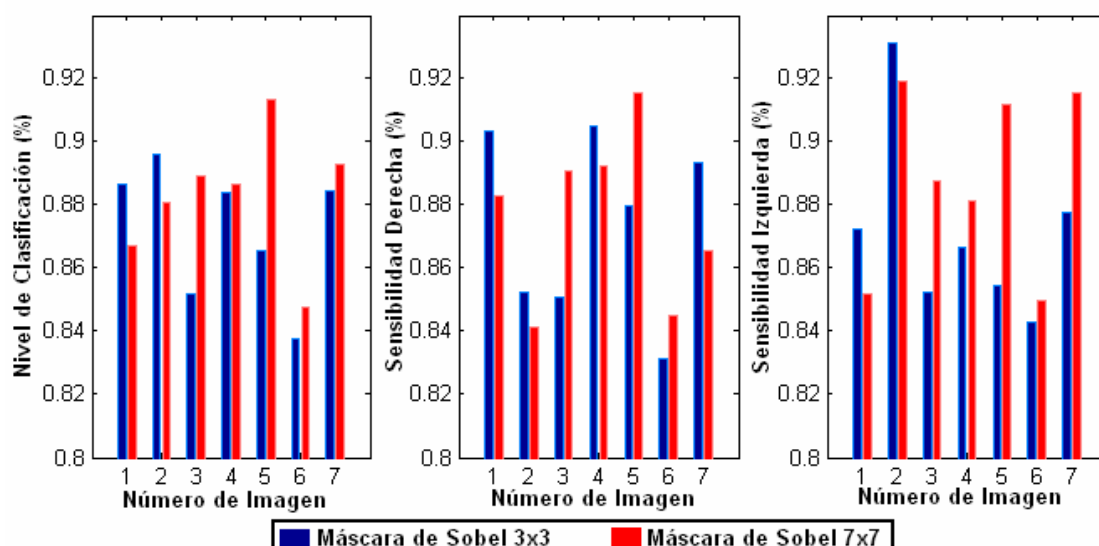
Como se comprueba en la Figura 4.4, los valores promedio en la secuencia son:

- con una máscara de Sobel 3x3:

$$NC = (87.2 \pm 1.9) \% ; SE_D = (87.4 \pm 2.8) \% ; SE_I = (87.1 \pm 3.0) \%$$

- con una máscara de Sobel 7x7:

$$NC = (88.2 \pm 2.1) \% ; SE_D = (87.6 \pm 3.2) \% ; SE_I = (88.8 \pm 3.1) \%$$



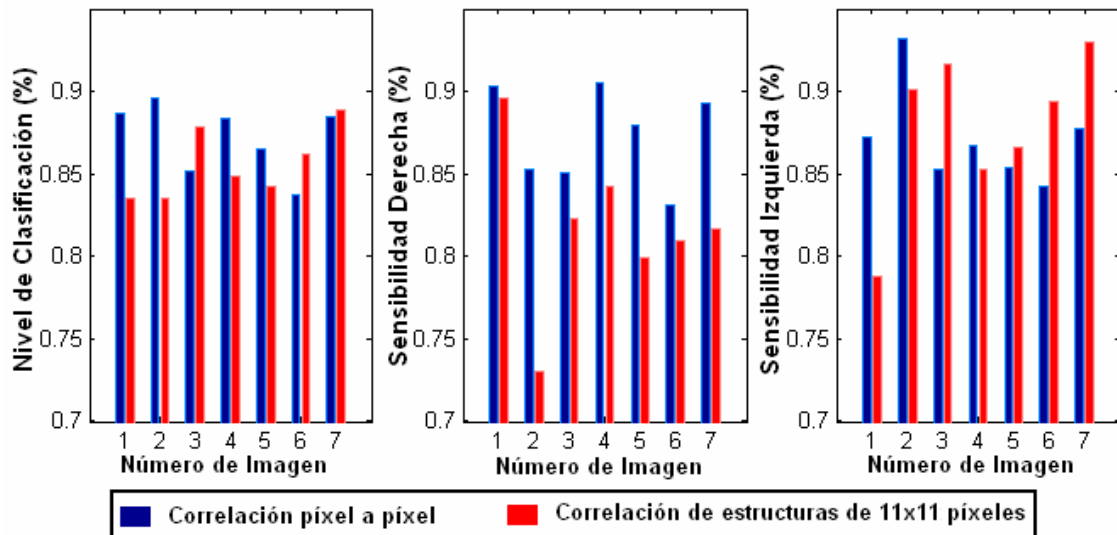
**Figura 4.4.** Se muestra la representación gráfica de NC,  $SE_D$  y  $SE_I$  detectados para varias imágenes de la secuencia de Chaplin, después de la etapa de filtrado, cuando usamos una máscara de Sobel de 3x3 píxeles (barras azules), y cuando usamos una máscara de Sobel de 7x7 píxeles (barras rojas).

Los resultados anteriores sugieren que, el movimiento hacia la izquierda, que era el detectado con más ruido (detecciones de movimiento incorrectas), como vimos en el Capítulo 2, debido a la existencia de una zona con mucha textura (los elementos apilados sobre las bandejas), se ve beneficiado por el aumento en el tamaño de la máscara en algo más del 1% en el nivel de  $SE_I$ . Sin embargo, el movimiento hacia la derecha no se beneficia del aumento de la máscara de Sobel. El aumento en  $SE_I$  se refleja también en un aumento del 1% en NC.

Para el tamaño de imagen con el que estamos trabajando, el beneficio obtenido duplicando el tamaño de la máscara no se traduce en un aumento significativo de los niveles de NC,  $SE_D$  y  $SE_I$ , esto se debe a que las máscaras de Sobel de mayor tamaño tienen un efecto beneficioso en la detección de bordes en imágenes con contenido de ruido de tipo “sal y pimienta”, y la secuencia bajo estudio carece de este tipo de ruido. En todo caso, puesto que luego correlacionaremos bordes para estimar movimiento, el ruido de “sal y pimienta” tendrá un efecto reducido en la salida final. Por tanto, en lo sucesivo vamos a usar máscaras de Sobel de tamaño 3x3.

### c) Tamaño de la estructura de correlación:

Como se comentó en el Capítulo 2, los resultados de clasificación quizás se podrían mejorar realizando la correlación de grupos de píxeles en lugar de un único píxel, de esta forma, el ruido debido a correlaciones con píxeles de falsos bordes afectará en menor grado.



**Figura 4.5.** Se muestra la representación gráfica de NC,  $SE_D$  y  $SE_I$  detectados para algunas imágenes de la secuencia de Chaplin después de la etapa de filtrado, cuando la correlación se realiza píxel a píxel (barras azules), y cuando usamos una estructura de correlación compuesta por un conjunto de 11x11 píxeles (barras rojas).

Para comprobar este efecto, se ha realizado, para la secuencia de la Figura 4.1, una comparativa entre los resultados de NC,  $SE_D$  y  $SE_I$ , después de la etapa de filtrado, cuando se realiza la correlación píxel a píxel (barras azules de la Figura 4.5), y cuando la correlación se hace entre estructuras de 11x11 píxeles (barras rojas de la Figura 4.5). Como puede observarse, los resultados mejoran en algunas imágenes de la secuencia, aunque no lo hacen en general, de hecho la correlación píxel a píxel es menor en el caso de la  $SE_D$ ; y un poco mayor en algunas imágenes en el caso de  $SE_I$ , aunque bastante irregular.

Se puede observar en la Figura 4.6, una comparativa entre los resultados del algoritmo cuando se usa la convolución píxel a píxel (Figura 4.6b) y cuando se utiliza la estructura de convolución de 11x11 píxeles (Figura 4.6c).

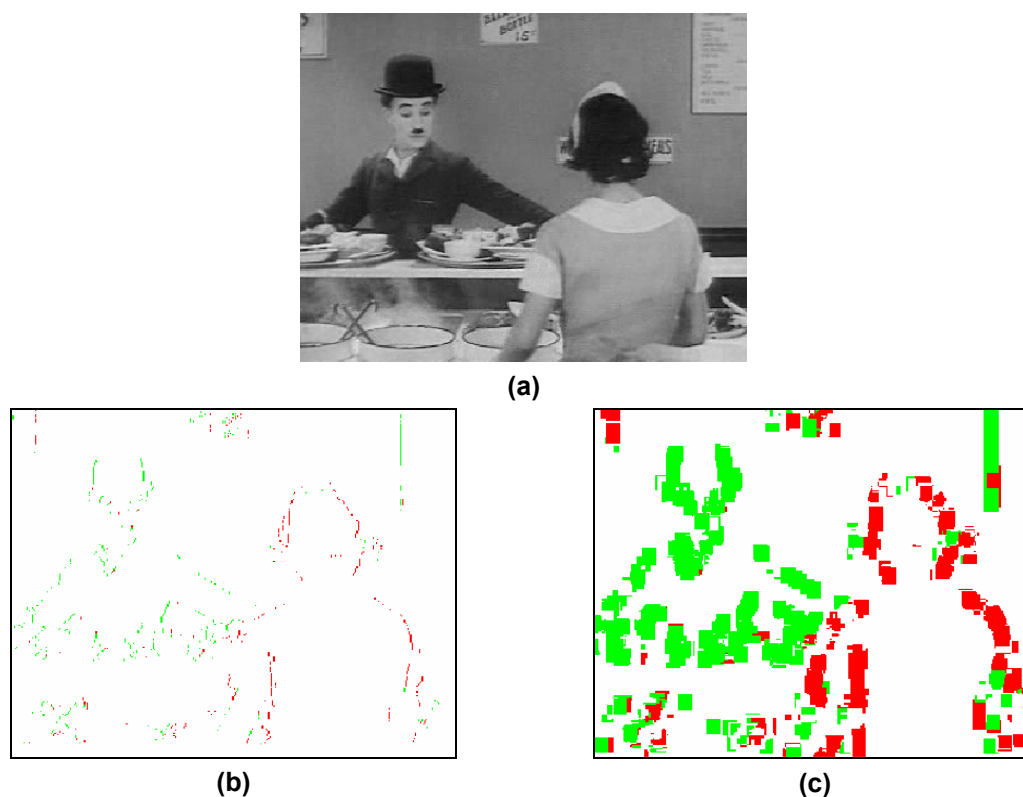
**d) Tamaño de las secciones o áreas de influencia local en el filtrado por coherencia espacial:**

Como se vio en el Capítulo 2, el tamaño de las áreas de influencia local en las que dividimos la imagen va a tener efecto sobre el tamaño de los sólidos rígidos cuyo movimiento somos capaces de filtrar.

En esta subsección vamos a ver el efecto que tiene el tamaño de estas secciones sobre los parámetros NC,  $SE_D$  y  $SE_I$ , para la secuencia mostrada en la Figura 4.1.

En la Figura 4.7a vemos cómo varía el promedio para la secuencia de los niveles de NC,  $SE_D$  y  $SE_I$  para distintos tamaños del área de influencia en la etapa de filtrado por coherencia espacial. Los tamaños de área utilizados son: 5x10, 10x15, 15x20, 20x25, 25x30,

30x35, 35x40, 40x45, 45x50, 50x55, 55x60, 60x65, 65x70, 70x75, 75x80 píxeles. Para elaborar esta figura hemos utilizado un umbral de 25, que como hemos visto en la Sección 4.2.1.a nos proporciona suficientes píxeles.



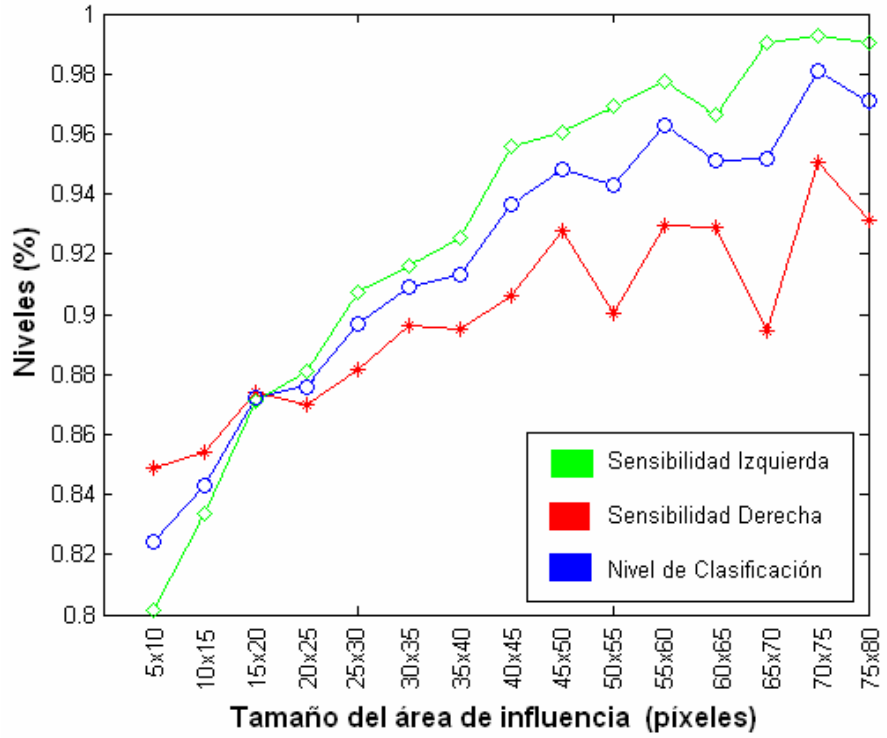
**Figura 4.6.** Comparativa de los resultados de detección de movimiento cuando se usa una estructura de convolución píxel a píxel **(b)** y cuando se usa una estructura de 11x11 píxeles **(c)**. En **(a)** se observa la imagen original de la secuencia que está siendo evaluada.

En la Figura 4.7b se representa la variabilidad en los niveles de clasificación cuando usamos un tamaño de área de influencia, y variamos el umbral. Aunque el estudio se ha realizado para todos los tamaños de área descritos en el párrafo anterior, en la figura se muestran solamente los resultados más representativos entre todos los obtenidos.

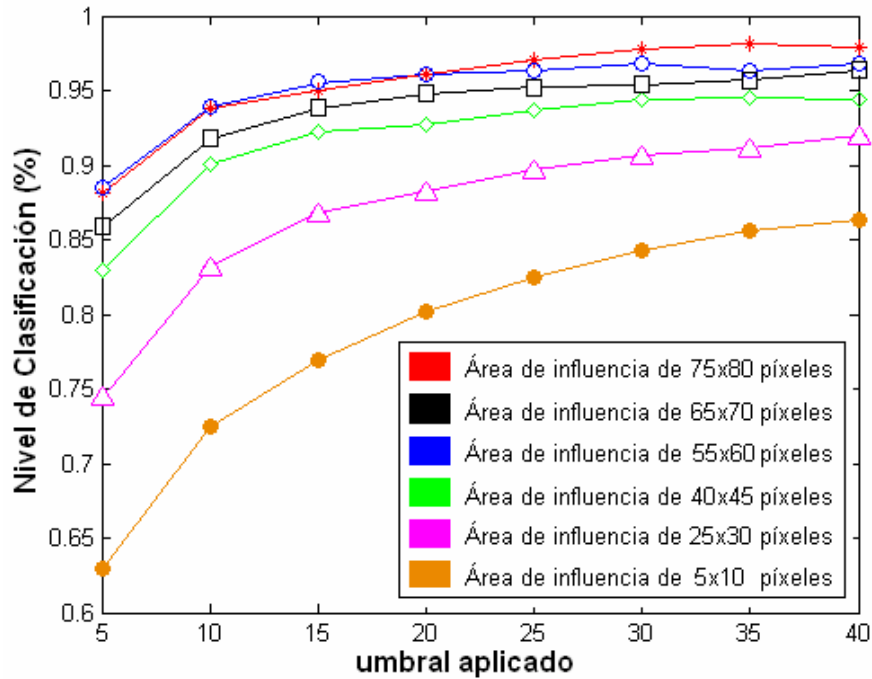
Se observa que, para tamaños de ventana pequeños el umbral utilizado produce mucha variabilidad de NC, sin embargo, para tamaños de ventana grandes el umbral utilizado induce cambios menos significativos, ya que la variabilidad de NC es menor.

Esto se debe a que, como los objetos que se mueven a través de la secuencia son de gran tamaño, el filtrado mediante integración de información (coherencia espacial) es capaz de eliminar el ruido que se produce cuando bajamos el nivel de umbral y se introducen bordes débiles, que producen una correlación defectuosa. Por otro lado, también somos capaces de filtrar mediante la integración de información el ruido que se producido por los detalles del objeto principal que poseen movimiento propio (como los rasgos de la cara de Chaplin).

En todo caso, no podemos aumentar mucho el área de influencia local porque perderíamos la capacidad de detectar el movimiento de objetos pequeños.



(a)



(b)

**Figura 4.7.** (a) Relación entre NC,  $SE_D$  y  $SE_I$  y el tamaño del área de influencia para un umbral igual a 25; (b) Relación entre el NC y el umbral para distintas áreas de influencia.

#### e) Efecto de los parámetros sobre el ruido del fondo

En esta secuencia, tanto la cámara como el fondo son estáticos. Por tanto, no deberíamos detectar movimiento aparte de los dos personajes principales que se mueven por

la escena. Sin embargo, como vimos en el Capítulo 2, posiblemente debido a la vibración de la cámara, se detecta el movimiento de algunos objetos que son estáticos, y que pertenecen a la escena.

Vamos a comprobar a continuación qué parámetros nos permiten minimizar ese ruido en el fondo de la escena y, por tanto, podrían hacer posible la segmentación de los objetos en movimiento de su entorno, teniendo en cuenta solamente consideraciones de movimiento.

Como hemos visto, el *tamaño de la máscara de convolución* o el *tamaño de la estructura de correlación* no juegan un papel importante en la reducción del ruido dentro de los objetos en movimiento, y tampoco reducen la detección de objetos en el fondo de la imagen.

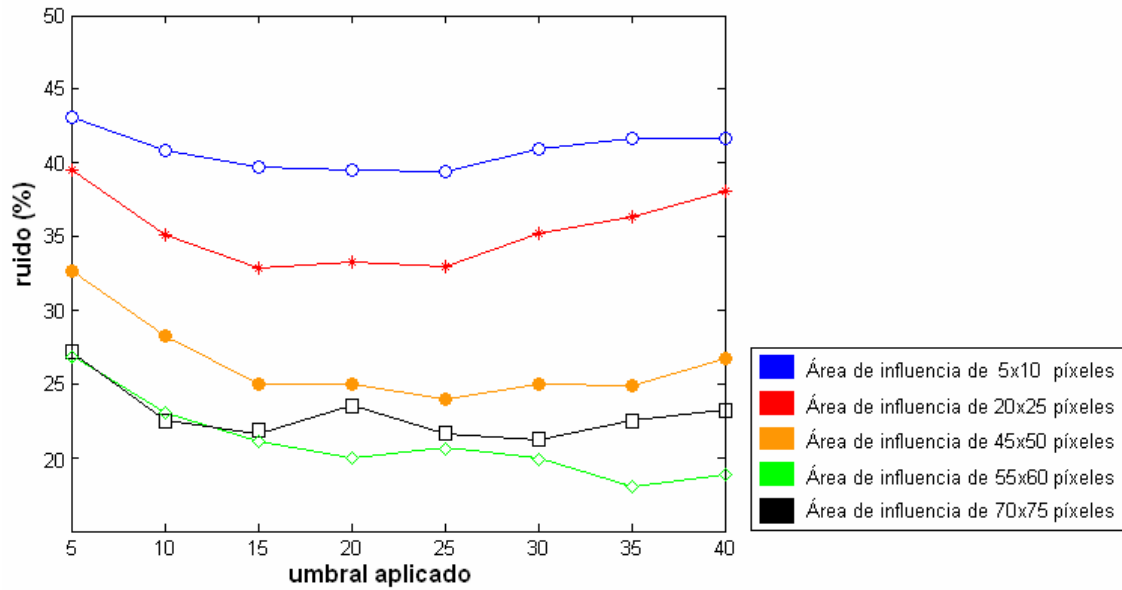
La *umbralización* va a afectar a todos los bordes de la imagen por igual, de manera que tampoco va a ser un factor determinante en la reducción de las detecciones del fondo.

Dado que los objetos fijos que pertenecen al fondo de la imagen son de tamaño pequeño en comparación con los objetos móviles es de esperar que la elección adecuada del *tamaño de las regiones de integración* pueda reducir las detecciones del fondo. Pero esto depende de la escena.

En todo caso, para comprobar esto vamos a realizar un estudio similar al realizado en la Figura 4.7b. En la Figura 4.8 se representa el porcentaje de ruido, es decir, la relación entre el número de detecciones en el fondo estático y el número de detecciones en los objetos en movimiento, multiplicado por cien, y muestra los resultados obtenidos para los tamaños de área más representativos de entre los estudiados.

Como se puede observar, usando un tamaño de área de influencia o integración de 5x10 píxeles, el más pequeño de los estudiados, las detecciones del fondo estático no son filtradas o lo son en muy poca medida, por tanto, obtenemos un porcentaje de ruido muy alto, es decir, entre el 40 y el 44% de los píxeles detectados en movimiento corresponden al fondo. También se observa que la influencia del aumento de la umbralización con áreas pequeñas es baja. Sin embargo, el ruido del fondo es filtrado en gran medida cuando se usa una máscara de 55x60 píxeles. Cuando las áreas de influencia son mayores también aumenta el efecto de la umbralización, es decir, cuando el umbral es bajo el porcentaje de ruido es mayor que cuando la umbralización es alta. Para un umbral igual a 25 el porcentaje de ruido ronda el 20%.

Por otro lado, podemos comprobar en la Figura 4.7b que la elección de un área de influencia de 55x60 píxeles maximiza también el parámetro NC, y por tanto  $SE_D$  y  $SE_I$ .



**Figura 4.8.** Relación entre el porcentaje de ruido de fondo y el umbral para distintos tamaños de área de influencia.

#### f) Eficiencia del algoritmo sobre la secuencia

Después del estudio paramétrico realizado, hemos utilizado los siguientes valores de los parámetros para optimizar el rendimiento del algoritmo: umbral igual a 25, máscara de Sobel de 3x3, estructura de correlación de 1 píxel (píxel a píxel), área de integración de 55x60 píxeles.

Podemos observar en la Figura 4.9 la eficiencia en cuanto a la capacidad de clasificación del movimiento de las distintas características detectadas.

En la fila inferior de la Figura 4.9 podemos ver que, la cantidad de puntos antes de la etapa de filtrado es aproximadamente el doble que después de la etapa de filtrado. Los puntos que quedan mejoran las medidas de NC,  $SE_D$  y  $SE_I$  en promedio entre el 16 y el 18%, como se puede ver:

- antes de la etapa de filtrado: NC = 80.7%;  $SE_D$  = 83.4%;  $SE_I$  = 78.2%;
- después de la etapa de filtrado son: NC = 96.3%;  $SE_D$  = 92.9%;  $SE_I$  = 97.8%.

Los resultados anteriores muestran que la eficiencia del detector de movimiento se incrementa considerablemente cuando se utiliza la capa bio-inspirada de integración de información, como ya se comprobó en el Capítulo 2.

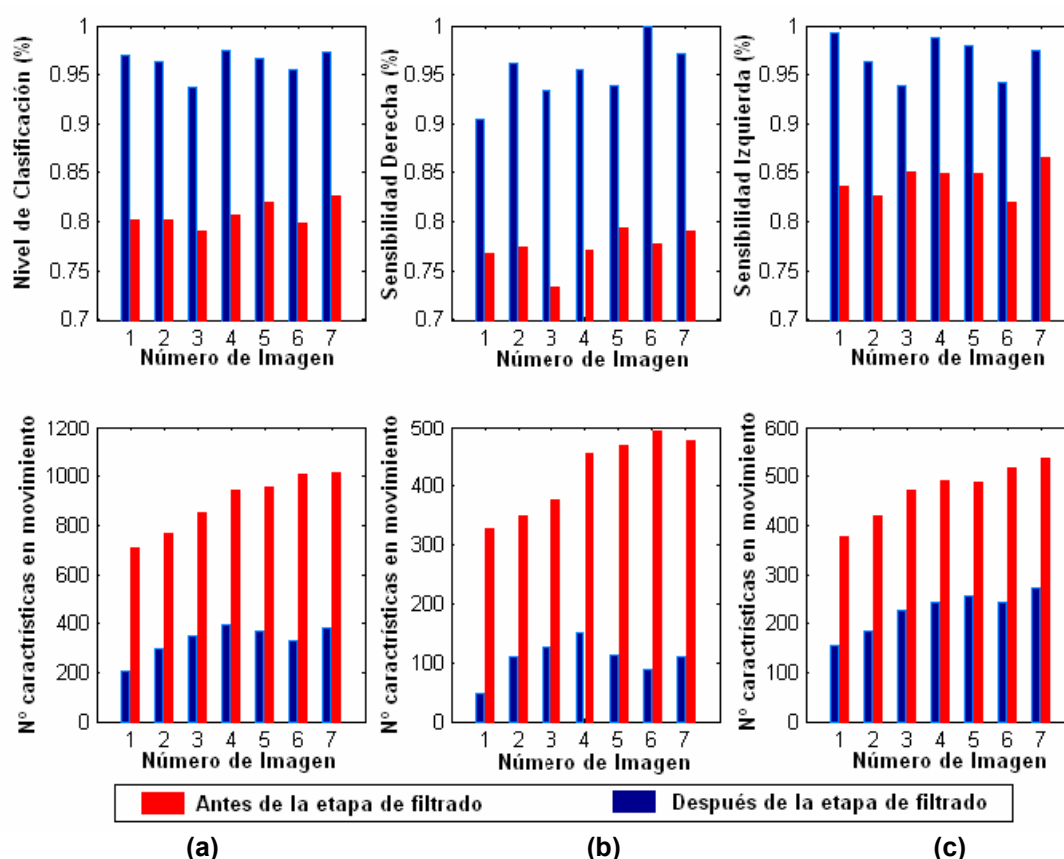
Recordemos, que la escena era especialmente ruidosa en la parte correspondiente al personaje de Chaplin debido, en parte a que la textura existente en la zona de las bandejas de comida provoca falsas detecciones de movimiento. Vemos que después del filtrado, buena parte de ese ruido se ha eliminado, en gran medida ya que hemos usado áreas de integración



de tamaño medio que permiten eliminar de la salida los píxeles cuyo movimiento no es coherente con el objeto principal.

Además, el tamaño de las secciones de integración que determinan la coherencia espacial permite que, el movimiento detectado en los rasgos de la cara debido al giro de la cabeza también sea filtrado, mejorando considerablemente los resultados.

En una escena simétrica, como la anterior (en la que se producen patrones de movimiento hacia la derecha y hacia la izquierda en la misma proporción), la  $SE_D$  y la  $SE_I$  van a estar dentro del mismo rango, como hemos podido observar en la Figura 4.9, y será interesante que ambos valores sean máximos de cara a que la exactitud del algoritmo sea la máxima posible. En este caso, el Nivel de Clasificación será un valor significativo de la bondad del algoritmo.



**Figura 4.9.** En la fila superior se muestra la representación gráfica del Nivel de Clasificación (izquierda), la Sensibilidad Derecha (centro) y la Sensibilidad Izquierda (derecha) del algoritmo para la detección de movimiento en cada una de las imágenes de la secuencia de Chaplin. En la fila inferior tenemos la representación de la cantidad de puntos con los que se obtienen los valores de las gráficas superiores en cada imagen de la secuencia. En la columna (a) se representan los puntos etiquetados, es decir, los puntos que pertenecen a ambos objetos segmentados manualmente. En el centro (columna (b)) se representan los puntos que pertenecen al objeto que se mueve hacia la derecha, y en la columna (c), los puntos que pertenecen al objeto que se mueve hacia la izquierda. En todos los casos, las barras de color rojo representan las medidas antes de la etapa de filtrado, y las barras azules representan las medidas después de la etapa de filtrado.

Sin embargo, en escenas poco simétricas ambos parámetros (Sensibilidad Derecha y Sensibilidad Izquierda) podrían ser muy distintos, y el algoritmo podrá considerarse bueno cuando uno de ellos sea máximo. En este caso, el Nivel de Clasificación puede no ser el medidor que mejor caracterice la bondad del algoritmo de detección, sino la  $SE_D$  o la  $SE_I$ , según el caso. Veremos un ejemplo de un caso no simétrico cuando usemos el algoritmo en la aplicación real de monitorización de adelantamientos (Capítulo 5).

Cuando se utilizan los parámetros obtenidos en el estudio anterior (una secuencia simétrica), el algoritmo presenta el rendimiento óptimo para la secuencia bajo estudio, lo que significa que tenemos un Nivel de Clasificación (NC) del 96.2%, es decir, de las características detectadas en movimiento que pertenecen a los objetos en movimiento, el 96.2% tienen un movimiento que coincide con el movimiento neto de dichos objetos, el restante 3.8% no coincide con el mismo. Este "error" en la detección de características en movimiento se debe a un error real, o a la existencia de movimientos menores dentro del objeto, por ejemplo el giro de la cabeza, o el movimiento giratorio de hombros, que no han sido considerados de forma independiente cuando se realizó la segmentación manual, pero que el algoritmo etiqueta de forma fiel a la realidad aunque de forma diferente a las etiquetas manuales.

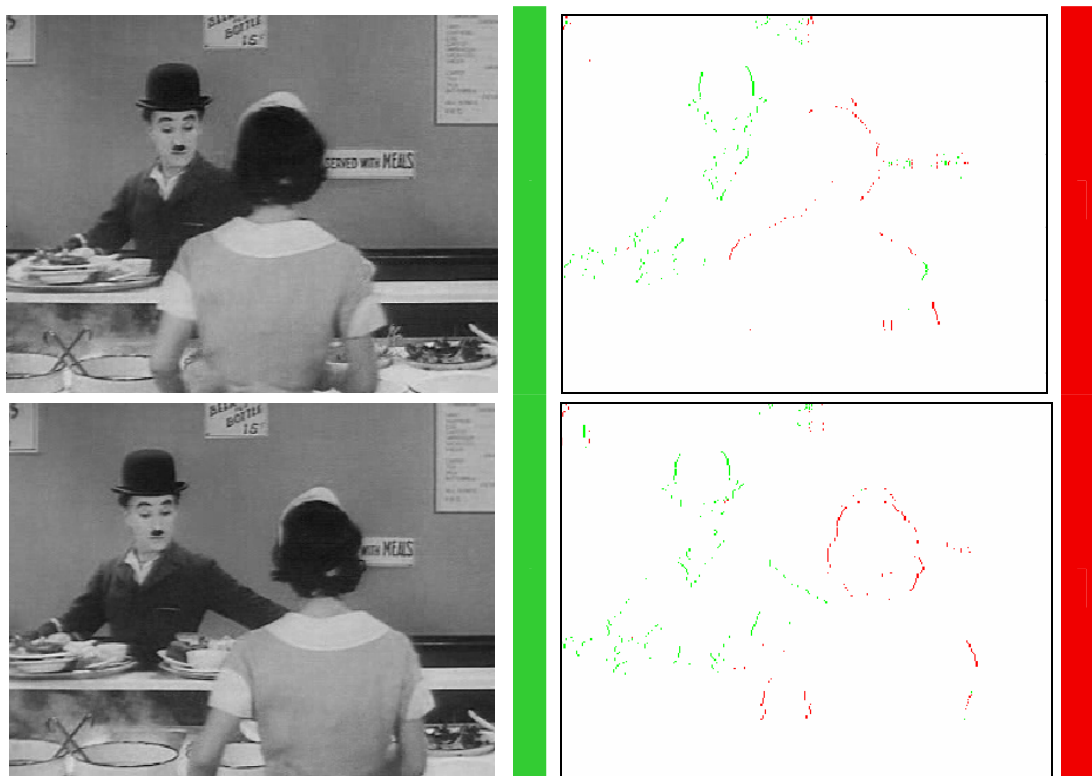
También reduce el ruido del fondo estático (debido a la vibración de la cámara), y lo deja en el 20%, es decir, del total de características detectadas en movimiento por el algoritmo, el 20% pertenecen al escenario y aunque en realidad son estáticas, posiblemente debido a vibraciones de la cámara, se produce un movimiento aparente que es detectado.

Este efecto debido a la vibración de la cámara se va a tener en numerosas situaciones reales, por lo que resulta de interés la capacidad del algoritmo de filtrar el movimiento *ficticio* originado por dicha vibración.

En la Figura 4.10 podemos ver la salida del algoritmo de detección de movimiento para algunos fotogramas de la secuencia.

Si bien durante el procesamiento (en las fases de correlación y filtrado) se hace un uso intensivo tanto del valor de la dirección del movimiento como del valor del módulo de la velocidad, durante los procesos de optimización de parámetros y de evaluación se utiliza exclusivamente la dirección del movimiento como criterio para determinar la eficiencia del algoritmo.

Una vez más, la justificación de este modo de actuación está en la dificultad para etiquetar el módulo de la velocidad para cada uno de los píxeles en movimiento de una secuencia real.



**Figura 4.10.** Salida del algoritmo de detección de movimiento usando los parámetros que optimizan la eficiencia.

No obstante, el proceso de etiquetar la dirección del movimiento tampoco está libre de problemas. Como se ha visto anteriormente, cuando se trabaja con sólidos deformables (o con movimientos que incluyen giros) a los que aplicamos criterios de sólido rígido se asignan etiquetas que pueden no ser correctas (en la secuencia anterior hemos etiquetado el movimiento de brazos y cabeza de la misma forma que el tronco, aunque tienen componentes de movimiento distintas).

Hasta aquí, el proceso seguido obtiene resultados que, a priori, son bastante exactos, como se comprueba en la Figura 4.10. Sin embargo, es necesario comprobar que los resultados que obtiene el algoritmo sobre el módulo de la velocidad, son también exactos. A continuación vamos a realizar un estudio de la eficiencia del algoritmo en cuanto a la detección del módulo de la velocidad de los objetos en movimiento.

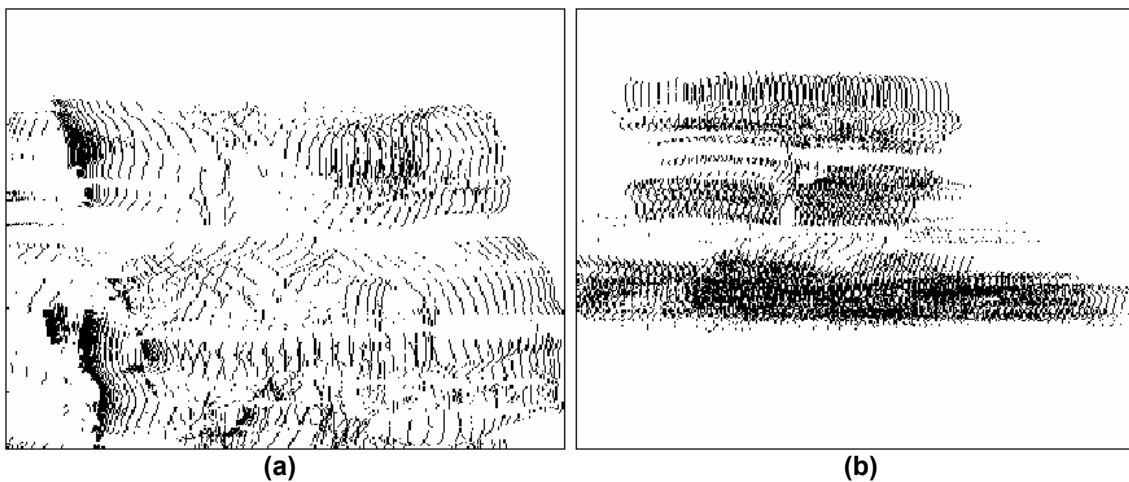
Por tratarse de secuencias reales, resulta difícil establecer de forma automática las velocidades efectivas de los píxeles pertenecientes a los objetos en movimiento. Por tanto, vamos a plantear una aproximación manual para el cálculo de dichas velocidades. Posteriormente compararemos la velocidad extraída manualmente con la velocidad que obtiene de forma automática el algoritmo. La velocidad experimental de uno de los personajes vendrá dada por la media de las velocidades de los píxeles que forman parte de él o ella.

El primer paso de esta aproximación manual consiste en la extracción de los bordes de todas las imágenes de la secuencia. Trabajaremos con parejas de imágenes consecutivas.

A continuación seleccionamos manualmente algunos bordes de cada pareja de imágenes. Como criterio de selección utilizaremos aquellos bordes que pertenecen a la misma zona del objeto en movimiento y que están presentes en las dos imágenes consecutivas que estamos comparando (por ejemplo, uno de los laterales del bombín que viste Chaplin, o el hombro derecho de la camarera).

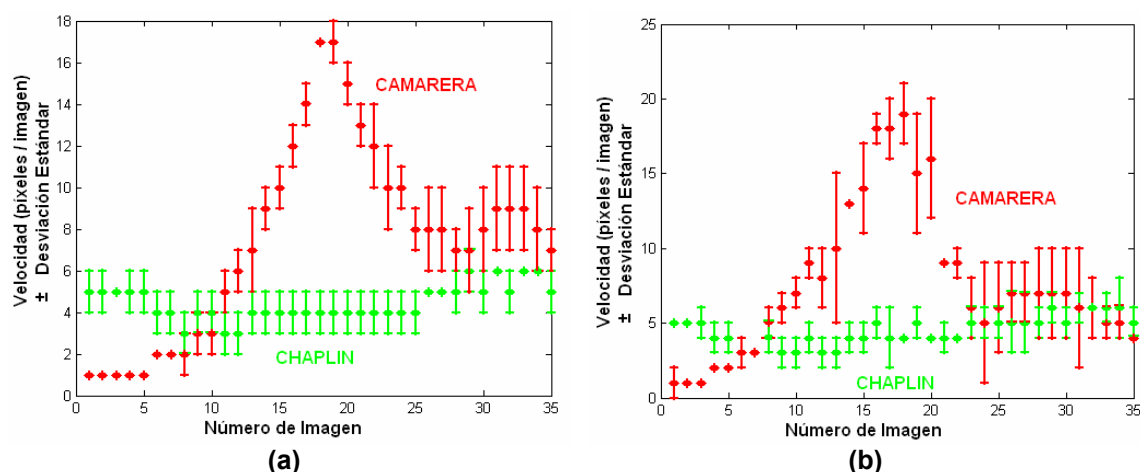
El siguiente paso consiste en determinar la posición de dichos bordes en cada una de las dos imágenes consecutivas y, posteriormente, medir la variación en dicha posición de una imagen a la siguiente. Este proceso se repite, al menos, en tres lugares distintos del objeto para el que se está determinando la velocidad.

Consideramos la estimación manual del módulo de la velocidad (en el par de imágenes) como la media de esas tres medidas de la variación en la posición de los bordes.



**Figura 4.11.** Superposición en una única imagen de los bordes principales pertenecientes a los objetos en movimiento de la secuencia. **(a)** Representación para la camarera; **(b)** Representación para Chaplin.

La Figura 4.11 muestra, superpuestos en una única imagen, los bordes de los objetos en movimiento (Chaplin y la camarera), en cada uno de los fotogramas de la secuencia. A la vista de dichas figuras podemos comprobar que la velocidad de la camarera es muy irregular a lo largo de la secuencia (Figura 4.11a): es prácticamente nula al principio de la misma; se produce una aceleración progresiva en la parte central de la secuencia; y posteriormente, decelera al final de la misma, manteniéndose aproximadamente constante en los últimos fotogramas. Mientras tanto, el personaje de Chaplin mantiene una velocidad que es aproximadamente constante a lo largo de toda la secuencia (Figura 4.11b).



**Figura 4.12. (a)** Valor promedio del módulo de la velocidad obtenido mediante el algoritmo de detección de movimiento y desviación estándar de las medidas de velocidad experimentales; **(b)** Valor promedio del módulo de la velocidad obtenido de forma manual y desviación estándar de las medidas de velocidad manuales. En todos los casos, el rojo representa el valor de la medida para el caso de la camarera y en verde se utiliza para representar el valor de la medida de Chaplin.

La Figura 4.12 recoge los resultados del módulo de la velocidad que hemos obtenido. Las gráficas **(a)** y **(b)** representan la velocidad, es decir, la variación (en píxeles) en la posición de los personajes en la imagen actual con respecto a la posición que ocupaba en la imagen anterior a ella en la secuencia.

El color rojo representa el módulo de la velocidad de la camarera, mientras que el color verde representa el módulo de la velocidad de Chaplin. La figura **(a)** muestra los valores del módulo de la velocidad medidos por el algoritmo y la figura **(b)** representa los valores obtenidos manualmente. Obsérvese que, tal y como hemos estimado antes, la camarera lleva un movimiento acelerado, mientras que Chaplin lleva un movimiento aproximadamente constante.

La velocidad obtenida manualmente y la velocidad detectada por el algoritmo son similares, con oscilaciones de  $\pm 1$  píxel. Hay que tener en cuenta que la velocidad experimental representada es el promedio del valor de todos los píxeles que pertenecen al objeto y que se detectan moviéndose con la dirección del mismo, es decir, es el promedio de aproximadamente 50 píxeles por imagen; mientras que la velocidad manual es el promedio de la velocidad de tan solo tres píxeles.

Es destacable que, pese al esfuerzo por realizar una medida manual lo más exacta posible en cada una de las tres medidas realizadas por cada personaje y por cada par de imágenes, se obtienen resultados con una desviación estándar muy variable, y en algunos casos bastante alta (Figura 4.12d). Esto es solamente una muestra de la dificultad que conlleva etiquetar la velocidad en secuencias reales. En el proceso de etiquetado manual que hemos realizado el problema fundamental ha sido la identificación manual del mismo borde en ambas imágenes para la asignación de etiquetas sobre el módulo de la velocidad.

## 4.3. Evaluación II

Esta segunda secuencia, que nos va a servir para evaluar el algoritmo, pertenece también a la película “Tiempos modernos” de Charles Chaplin, y está representada parcialmente por algunos fotogramas en la Figura 4.13. La secuencia completa consta de más de 60 fotogramas de 360 x 270 píxeles. En ella, la cámara es estática, y nos muestra al personaje de Chaplin en movimiento hacia la cámara sobre un fondo estático. En la escena Chaplin está patinando, por lo que realiza un movimiento en zig zag hacia la derecha y la izquierda de la imagen durante su desplazamiento. El tamaño del personaje se hace cada vez mayor a medida que transcurre la secuencia debido al efecto de perspectiva de acercamiento a la cámara.

Aunque la velocidad de Chaplin es constante, la perspectiva produce un efecto similar al de un movimiento acelerado.

Debido al movimiento de acercamiento hacia la cámara esta secuencia es más compleja que la estudiada anteriormente, ya que el tamaño del personaje a través de la secuencia aumenta, con lo que el número de características en movimiento que lo definen es variable a lo largo de la secuencia. Este hecho repercutirá sobre todo en que el proceso de definición de las áreas de coherencia espacial será más complejo.

Vamos a realizar un estudio paramétrico similar al realizado en la sección anterior para la otra secuencia de Chaplin. En este caso usaremos máscaras de Sobel de 3x3 píxeles, y no realizaremos el estudio para máscaras de Sobel de mayor tamaño ya que, como vimos anteriormente, el uso de dichas máscaras solamente produce un beneficio cuando existe un ruido de tipo “sal y pimienta” en las imágenes, que en la secuencia bajo estudio no se da.

Puesto que el tamaño del objeto en movimiento es variable, para realizar el estudio vamos a dividir la secuencia en 4 tramos de imágenes. En único criterio que se ha tomado para realizar la división en tramos es que en cada tramo el personaje tenga un tamaño similar, por tanto, cada tramo puede tener un número de fotogramas distinto, y no siempre tenemos una dirección de movimiento única en el mismo tramo.

Estudiaremos los parámetros óptimos para cada tramo de la secuencia, y finalmente asumiremos aquellos parámetros que nos den el mejor rendimiento para toda la secuencia, es decir, para todos los tramos.



**Figura 4.13.** Selección de algunas imágenes correspondientes a la película “Tiempos Modernos” de Charles Chaplin, que vamos a utilizar para realizar la segunda evaluación del algoritmo de detección de movimiento.

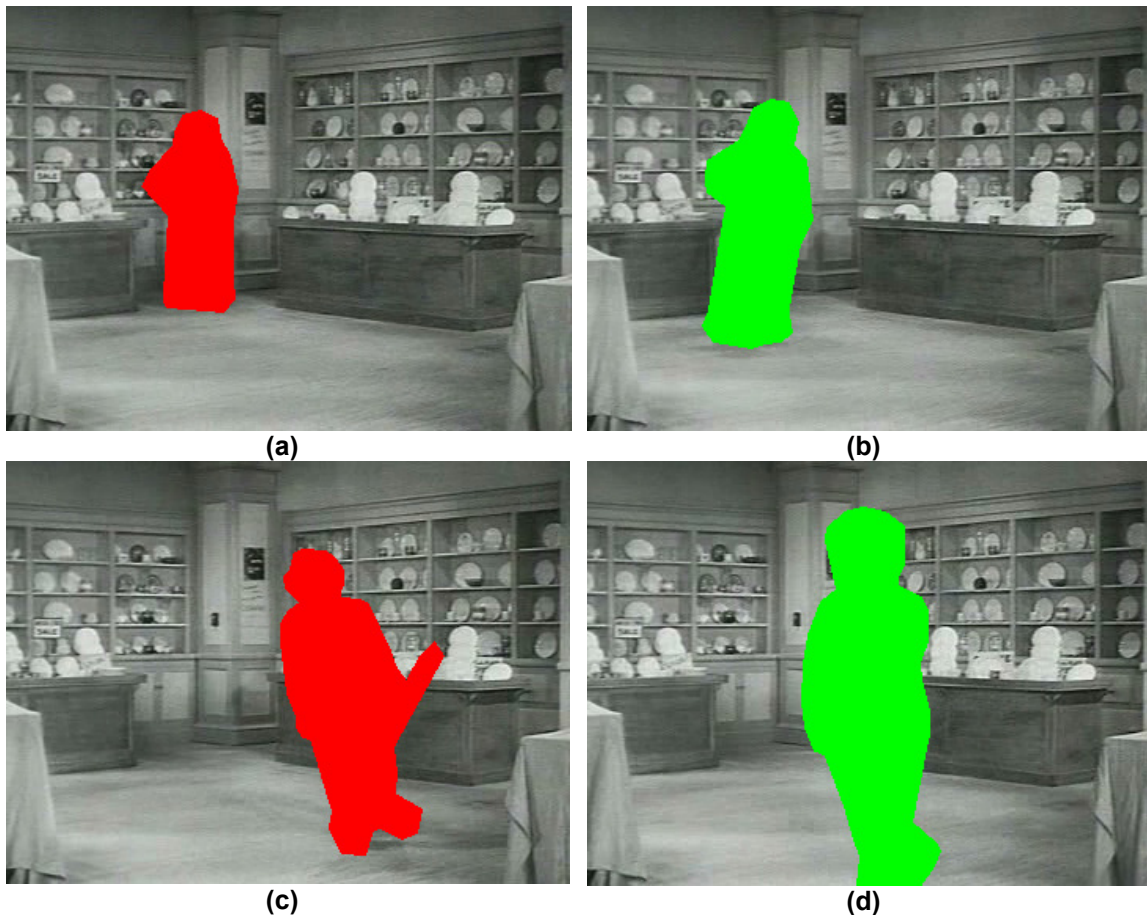
#### a) Umbralización de los bordes:

Como en el estudio realizado en la sección anterior, hemos obtenido el valor de  $NC$ ,  $SE_D$  y  $SE_I$  cuando el umbral es igual a: 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70 y 75.

En las Figuras 4.15 y 4.16 podemos ver algunos de los resultados para distintos umbrales y la totalidad de los 4 tramos en que hemos dividido la secuencia.

Nótese que para esta secuencia el parámetro  $NC$  coincide con  $SE_D$  cuando Chaplin se desplaza a la derecha, y con  $SE_I$  cuando se desplaza a la izquierda, dado que solamente tenemos un objeto en movimiento por la escena.

Obsérvese que, en los fotogramas de transición de un movimiento a otro, es decir, aquellos fotogramas en los que Chaplin cambia de dirección de movimiento, y pasa de desplazarse hacia la derecha a desplazarse hacia la izquierda y viceversa, parte del cuerpo comienza a realizar el cambio de dirección, mientras que la otra parte mantiene el movimiento, con lo que los índices  $SE_D$  y  $SE_I$ , y por tanto  $NC$ , tienen unos niveles muy bajos (véase la Figura 4.15).



**Figura 4.14.** (a) Imagen representativa del tramo 1; (b) Imagen representativa del tramo 2; (c) Imagen representativa del tramo 3; (d) Imagen representativa del tramo 4. Se observa en todos los casos la segmentación del personaje en movimiento. Cuando la segmentación se realiza con color rojo se indica que el movimiento es hacia la derecha, y si la segmentación se realiza con color verde el movimiento es hacia la izquierda.

Según los resultados de las Figuras 4.15 y 4.16, el umbral óptimo podría estar entre 20 y 40, ya que, en el caso de umbrales menores de 20 se alcanzan los niveles más bajos de NC para todos los tamaños; y para umbrales mayores de 40 tenemos los mayores valores de NC el número de características activas (bordes) es muy reducido.

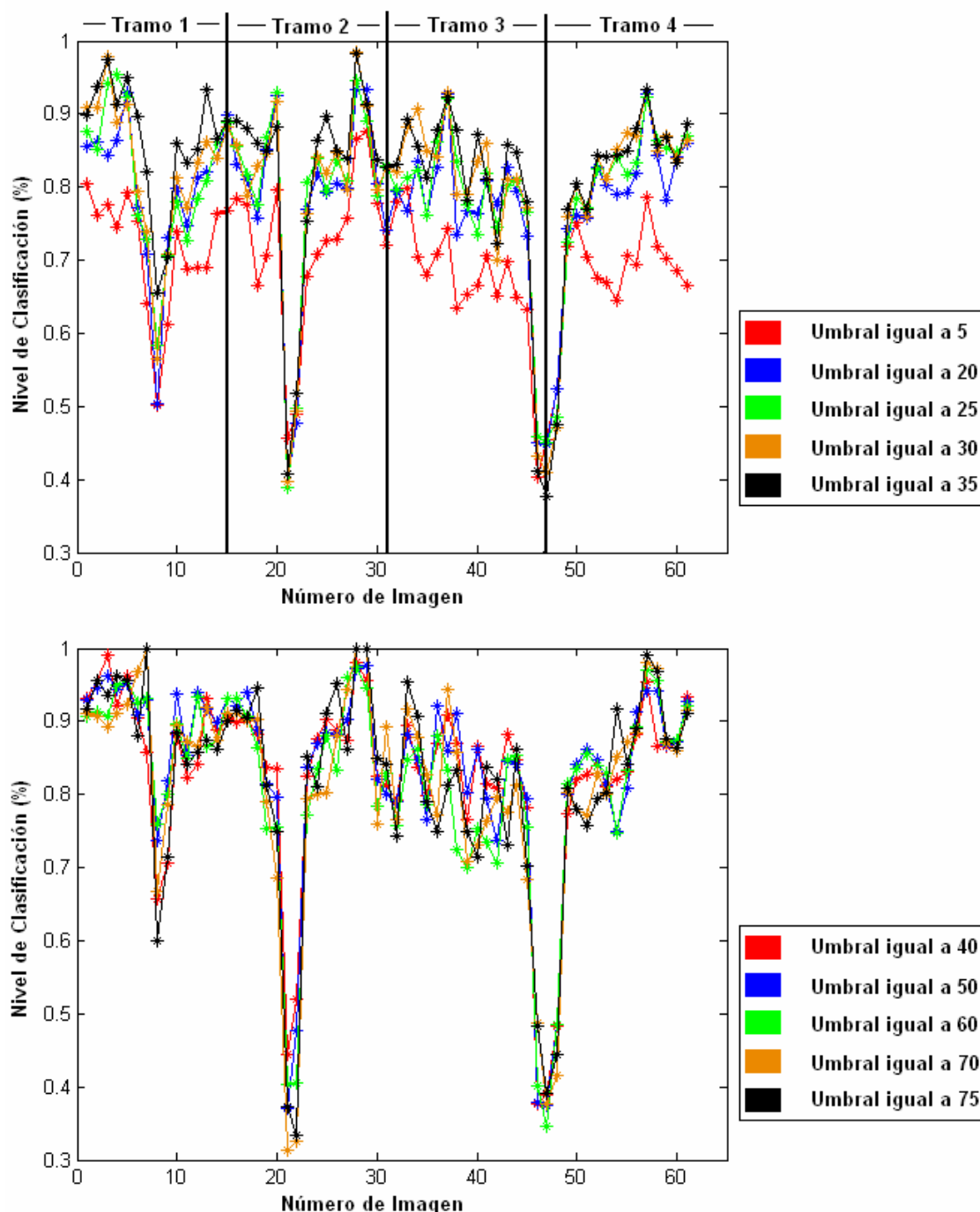
Se observa en las figuras anteriores que cuando el umbral está comprendido entre 30 y 40 los valores de NC son similares entre sí, y del mismo rango que con umbralizaciones superiores.

Cuando el umbral está entre 20 y 40 tenemos también un número de bordes activos por encima de 100, por lo que podremos caracterizar el movimiento para cualquier tamaño del personaje en movimiento.

Como en el caso de la secuencia estudiada con anterioridad, cuando realizamos la umbralización, posteriormente a la extracción de características, mantenemos activos en la imagen un número de píxeles que constituyen aproximadamente el 0.15% del número de píxeles de la imagen original. Estos píxeles que permanecen activos son los que nos van a permitir realizar la estimación del movimiento en las siguientes etapas de procesamiento.



Como en el estudio realizado en la secuencia anterior, aquí también es esperable una mejora de los parámetros de clasificación en cuanto se utilice el valor optimizado de la totalidad de los parámetros que intervienen en el rendimiento del algoritmo.



**Figura 4.15.** Representación del Nivel de Clasificación obtenida para cada fotograma de la secuencia, y para distintas umbralizaciones; también se indica la división en 4 tramos que se ha realizado en función del tamaño del personaje en movimiento.

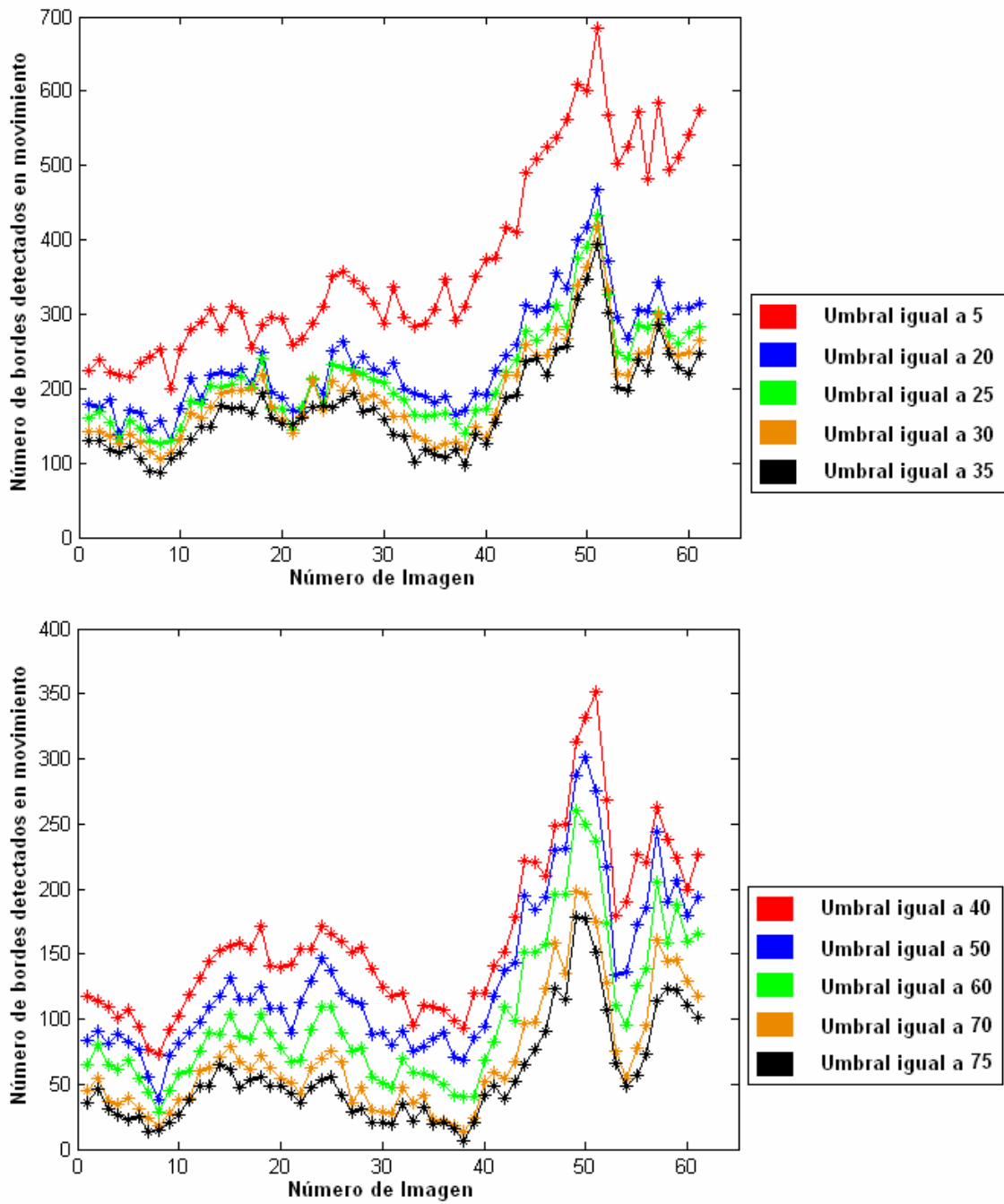
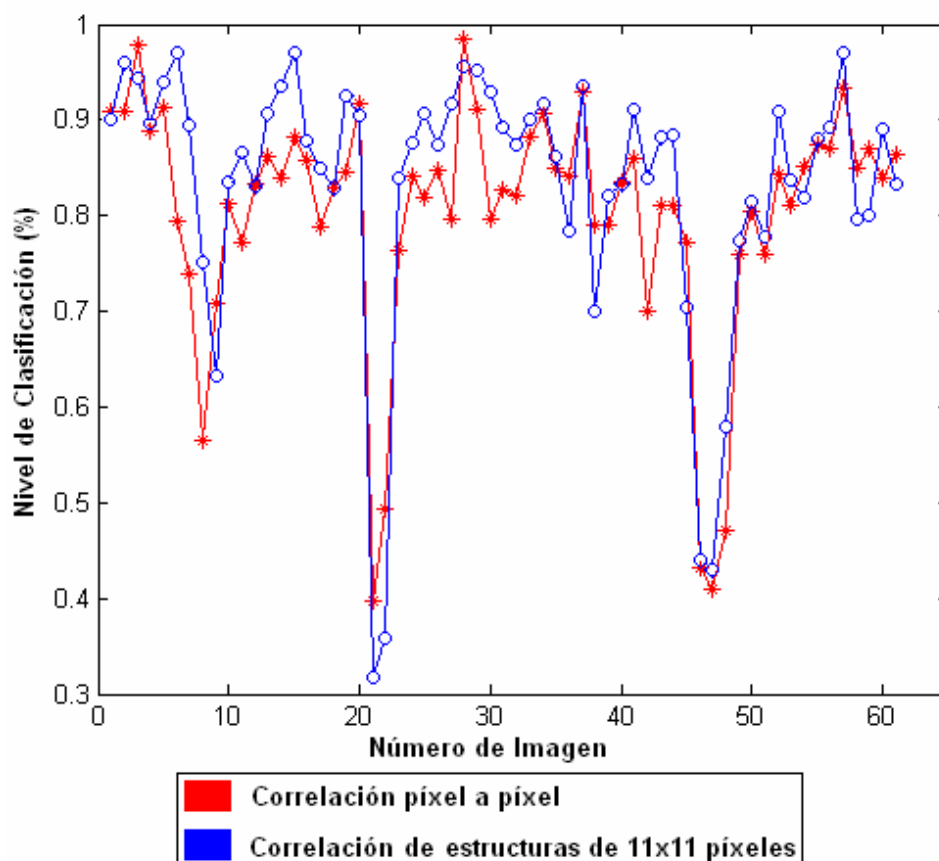


Figura 4.16. Número de bordes en movimiento a lo largo de la secuencia y para los mismos umbrales que en la Figura 4.15.

**b) Tamaño de la estructura de correlación:**

En la Figura 4.17 podemos ver el efecto que tiene sobre el valor de NC el uso de estructuras de correlación mayores.



**Figura 4.17.** Efecto sobre los valores del Nivel de Clasificación ( $SE_D$  y  $SE_I$ ) de la correlación mediante estructuras de  $1 \times 1$  o de  $11 \times 11$  píxeles para toda la secuencia en estudio.

Como se observa en la figura, cuando el tamaño del objeto en movimiento es pequeño (los primeros 5 fotogramas de la secuencia son los de menor tamaño) no existe apenas mejora en el uso de estructuras de correlación mayores.

Cuando el tamaño del objeto en movimiento es grande (lo que ocurre en los últimos 14 fotogramas de la secuencia) existen irregularidades en el aumento de NC, es decir, unas veces aumenta con una estructura de correlación mayor, y otras veces disminuye. Este efecto fue el que encontramos al realizar el estudio en la sección anterior, donde los objetos en movimiento tenían un tamaño mayor o igual al del objeto que se mueve en la secuencia actualmente bajo estudio.

Cuando el tamaño del objeto en movimiento es mediano (en las restantes imágenes de la secuencia) existe una mejora muy variable en NC, que en los mejores casos supera el 15%, sin embargo, en promedio no es mayor del 3%.

Por todo lo anteriormente expuesto, utilizar una estructura de correlación mayor de 1 píxel requerirá de un consumo de hardware, en el proceso de implementación visto en el capítulo anterior, que no revierte en un incremento sustancial en la efectividad del algoritmo, por tanto, usaremos estructuras de correlación de un único píxel.

**c) Tamaño de las secciones o áreas de influencia local en el filtrado por coherencia espacial:**

Cuando obtenemos la variabilidad en los Niveles de Clasificación (como en la subsección b) respecto a distintos tamaños de área de influencia, y utilizando distintos umbrales, se observa que, como ocurría en el estudio realizado en la Sección 4.2d, para tamaños de ventana pequeños el uso de umbrales distintos produce mucha variabilidad en NC, sin embargo, para tamaños de ventana grandes el umbral utilizado induce cambios menos significativos de NC (véase la Tabla 4.1).

Para realizar la Tabla 4.1 se ha calculado NC medio (para toda la secuencia), que se muestra en las filas **A**; el NC cuando el tamaño del objeto en movimiento es pequeño (**B**), mediano (**C**) y grande (**D**). Los valores mostrados en **B**, **C** y **D** se han obtenido sobre unos pocos fotogramas de cada caso, evitando la situación de cambio de dirección de movimiento del personaje de Chaplin. Por ello, el promedio para toda la secuencia (**A**) es ligeramente inferior al promedio de los otros tres. Como se observa en la Tabla 4.1, para un tamaño de área de influencia y un umbral dados, los valores de NC que se obtienen son distintos para los distintos fotogramas en función del tamaño del objeto en movimiento.

Se han sombreado en amarillo las combinaciones de área de influencia y umbral que proporcionan mayores porcentajes de NC.

**d) Efecto de los parámetros sobre el ruido del fondo**

Como sucedía en la secuencia estudiada en la Sección 4.2, tanto la cámara como el fondo son estáticos pero, posiblemente debido a la vibración que produce la cámara, se detectan en movimiento algunos objetos estáticos pertenecientes al entorno. Estas detecciones de objetos móviles se califican como ruido de fondo.

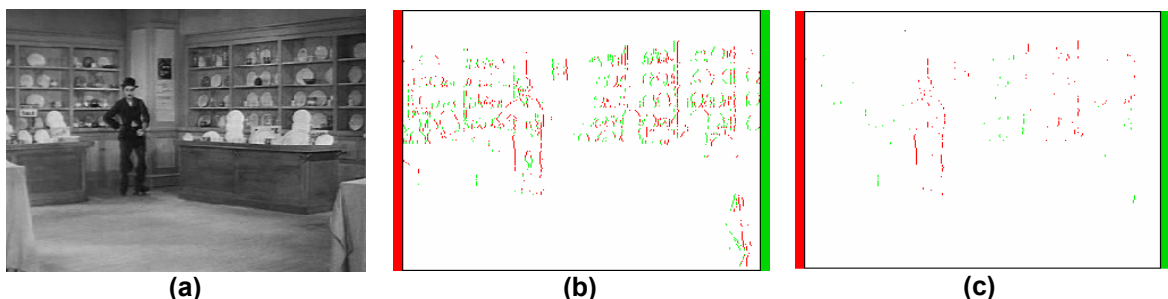
Como se hizo en la Sección 4.2e, vamos a estudiar a continuación cómo se reduce el porcentaje de ruido de fondo (es decir, la relación entre el número de las características que se detectan en movimiento y que pertenecen al entorno, y el número de las características que se detectan en movimiento y que pertenecen a los objetos que tienen un movimiento real, multiplicado por cien), mediante la elección adecuada de los parámetros del algoritmo. Esta reducción nos permitirá una segmentación mejor de los objetos que realmente se mueven por la escena.

Como se muestra en la Tabla 4.2, el porcentaje de ruido de fondo para esta secuencia es mayor que el que obteníamos en la secuencia estudiada en la Sección 4.2, esto se debe a, en primer lugar, la existencia de un fondo mucho más rico en estructura en esta secuencia (hay un número de objetos estáticos mucho mayor en la escena), y en segundo lugar, el tamaño del objeto en movimiento es menor que en el caso anterior y ocupa menos espacio de la imagen, por tanto, la proporción de píxeles que pertenecen a él también es menor (véase la Figura 4.18).

En la tabla también se muestran sombreados los valores del porcentaje de ruido de fondo correspondientes a los valores que proporcionan un mayor NC. Se puede comprobar que cuando se usa el umbral igual a 35 o 40, con un área de influencia de 70x75 píxeles, se obtiene el máximo de Nivel de Clasificación y el mínimo de Ruido de fondo.

**Tabla 4.1.** Porcentaje de Nivel de Clasificación (NC) para los distintos valores del tamaño de las áreas de influencia y el umbral, para toda la secuencia bajo estudio. En cada fila se han introducido los datos para cada umbral de: **A**) NC medio (%) para la secuencia; **B**) NC medio (%) de las primeras imágenes de la secuencia, donde el personaje de Chaplin tiene el tamaño mínimo; **C**) NC medio (%) de varias imágenes centrales de la secuencia, en las que el personaje de Chaplin tiene un tamaño mediano; **D**) NC medio (%) de las últimas imágenes de la secuencia, donde el personaje de Chaplin tiene el mayor tamaño. Se han sombreado los valores para el porcentaje de NC mejores.

Umbral		Tamaño del área de influencia														
		5x10	10x15	15x20	20x25	25x30	30x35	35x40	40x45	45x50	50x55	55x60	60x65	65x70	70x75	75x80
15	A	65.4	69.0	71.8	76.4	79.3	82.9	82.5	83.9	86.2	87.8	87.1	83.5	86.2	90.8	90.0
	B	64.0	71.2	75.9	85.5	90.4	92.0	91.8	90.4	91.2	98.4	92.0	94.4	99.7	99.1	97.6
	C	65.8	69.6	73.2	78.0	81.7	85.1	84.4	87.2	87.8	89.1	89.7	84.2	89.3	93.7	92.6
	D	72.3	75.2	77.3	79.4	81.1	85.5	86.1	89.4	91.4	91.4	89.9	91.3	87.6	88.8	92.1
20	A	66.9	70.8	72.9	77.9	80.4	83.8	82.8	84.3	87.5	89.2	86.7	83.7	87.3	90.6	90.1
	B	66.1	73.8	78.2	87.1	91.3	93.5	93.9	92.0	93.6	99.2	94.3	95.5	99.2	99.3	98.2
	C	66.9	71.2	74.1	79.1	82.5	86.7	84.8	88.4	89.2	91.6	90.7	86.4	90.3	93.8	92.9
	D	74.8	77.0	79.2	81.7	82.1	85.2	86.0	87.7	92.6	92.0	89.5	90.4	88.9	90.9	92.3
25	A	68.2	71.9	74.4	79.1	81.5	85.4	84.7	85.6	87.6	89.6	87.9	85.9	87.8	90.0	89.9
	B	69.5	77.3	81.0	90.9	91.8	93.7	96.7	95.4	95.4	98.1	96.7	97.9	99.8	99.3	99.2
	C	68.0	72.1	75.4	80.0	83.0	88.8	85.6	88.7	89.5	91.6	91.5	89.1	90.7	94.3	92.7
	D	76.4	78.2	80.9	83.7	83.6	87.0	89.2	90.0	92.2	92.7	90.5	91.2	91.2	88.7	93.7
30	A	69.4	72.7	75.9	80.2	82.3	86.3	86.5	87.4	88.2	90.2	88.8	87.3	88.3	90.5	90.0
	B	72.0	78.6	86.2	91.9	91.7	97.5	96.7	97.0	95.6	98.7	98.3	99.4	100.0	99.2	97.1
	C	69.2	73.0	76.5	81.4	84.0	88.9	88.7	89.7	89.9	92.5	91.1	90.9	90.5	93.9	92.8
	D	77.1	79.3	81.7	84.7	85.3	89.1	89.9	92.2	93.3	94.9	92.6	91.1	92.3	91.7	93.6
35	A	70.4	73.8	77.5	82.0	83.9	87.4	87.3	88.9	88.9	91.1	91.3	89.2	88.3	91.9	91.1
	B	73.2	79.3	87.3	93.4	93.8	98.7	95.6	97.9	95.0	99.5	99.8	100.0	100.0	100.0	97.3
	C	70.3	74.3	78.7	83.3	86.0	89.8	90.4	90.3	91.0	93.5	94.2	93.0	90.0	94.5	94.0
	D	77.5	79.8	81.7	85.0	86.9	89.8	90.4	95.2	94.2	96.6	95.8	93.5	92.0	95.8	93.8
40	A	71.4	74.6	78.1	82.8	83.5	86.7	87.4	88.6	89.4	91.2	91.1	89.7	89.3	92.4	91.7
	B	76.1	81.0	88.8	95.3	91.1	97.4	95.7	98.0	96.3	98.0	99.8	100.0	100.0	100.0	97.1
	C	71.3	75.0	79.0	84.0	85.4	89.0	90.3	90.7	91.2	93.5	93.8	93.1	91.6	94.4	94.5
	D	78.3	80.8	82.7	86.0	87.7	89.8	90.7	93.5	94.6	97.2	96.0	94.4	92.7	96.8	94.7



**Figura 4.18.** Muestra de los resultados del algoritmo de detección de movimiento sobre la escena bajo estudio: **(a)** Imagen original; **(b)** Detección de movimiento antes de la etapa de filtrado; **(c)** Detección de movimiento después de la etapa de filtrado.

**Tabla 4.2.** Porcentaje de ruido de fondo debido a la vibración de la cámara, que se percibe como movimiento en objetos estáticos del entorno. Se han sombreado los valores del porcentaje de ruido para aquellos valores de los umbrales y áreas de influencia que en la Tabla 4.1 proporcionaban los mayores porcentajes de NC.

		Tamaño del área de influencia														
Umbral		5x10	10x15	15x20	20x25	25x30	30x35	35x40	40x45	45x50	50x55	55x60	60x65	65x70	70x75	75x80
15		84.2	82.9	82.1	81.4	80.9	80.2	77.3	77.6	78.3	78.1	74.3	77.3	77.5	73.7	73.6
20		82.4	81.1	80.2	79.2	78.4	77.8	73.7	74.1	75.3	74.5	70.7	72.2	72.9	68.9	69.0
25		80.8	79.3	78.6	77.6	76.0	76.2	70.9	72.3	73.1	71.9	67.6	69.1	70.9	65.0	65.5
30		79.3	78.0	77.5	76.2	74.3	75.1	69.2	69.8	71.3	69.5	65.4	67.8	69.1	62.8	63.3
35		77.2	76.0	75.9	74.3	72.0	73.0	67.3	66.8	69.1	67.2	62.4	65.6	68.2	60.5	61.6
40		77.2	76.0	75.9	74.3	72.0	73.0	67.3	66.8	69.1	67.2	62.4	65.6	68.2	60.5	61.6

**e) Eficiencia del algoritmo sobre la secuencia**

Según el estudio previo, cuando se utiliza alguna de las combinaciones de parámetros con mayor porcentaje de Nivel de Clasificación (las casillas sombreadas en las Tablas 4.1 y 4.2) obtenemos el mejor rendimiento del algoritmo para esta secuencia. Por ejemplo, usando los siguientes parámetros:

- Umbral igual a 40
- Máscara de Sobel de 3x3
- Estructura de correlación de 1x1 píxel
- Área de influencia de 70x75 píxeles

obtenemos un NC medio del 92.4% (para tamaños pequeños del objeto en movimiento tenemos un NC del 100%; para tamaños medianos del objeto en movimiento tenemos un NC del 94.4%; y para tamaños grandes del objeto en movimiento tenemos un NC del 96.8%) y un ruido de fondo del 60.5%, lo cual significa que de la totalidad de características que se han detectado en movimiento, el 60.5% pertenecen al fondo y a objetos estáticos, por tanto, son consideradas ruido; el restante 39.5% de características detectadas en movimiento pertenecen realmente a un objeto que se mueve por la escena. De éstas, el 92.4% están bien detectadas, es decir, el movimiento detectado se corresponde con el movimiento neto realizado por el objeto; y el 7.6% de las características, que no coinciden con el movimiento neto del objeto, pueden ser detecciones erróneas o pueden deberse a otros movimientos diferentes al movimiento neto.

Dado que el objeto en movimiento no es un sólido rígido perfecto, sino que se trata de una persona, hay partes (brazos, piernas, cabeza) que tienen movimiento propio (por ejemplo, el movimiento pendular de los brazos y piernas, o el giro de la cabeza), que puede detectarse de forma independiente, y que no tiene por qué coincidir con el movimiento principal del objeto, dando lugar a dicho porcentaje de detecciones “erróneas”.

En la Figura 4.19 se muestra el resultado del algoritmo de detección de movimiento sobre algunos fotogramas de la secuencia bajo estudio.

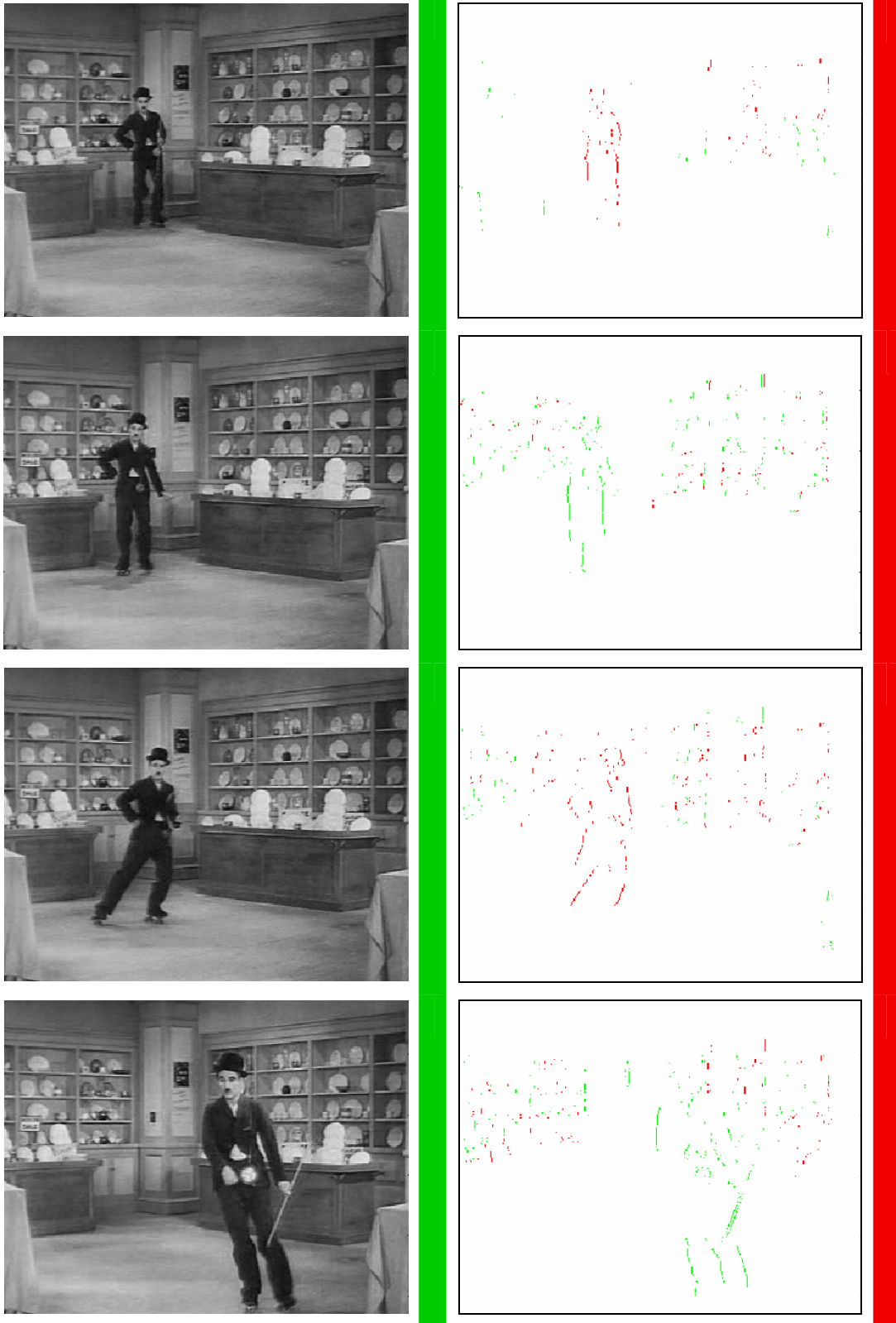
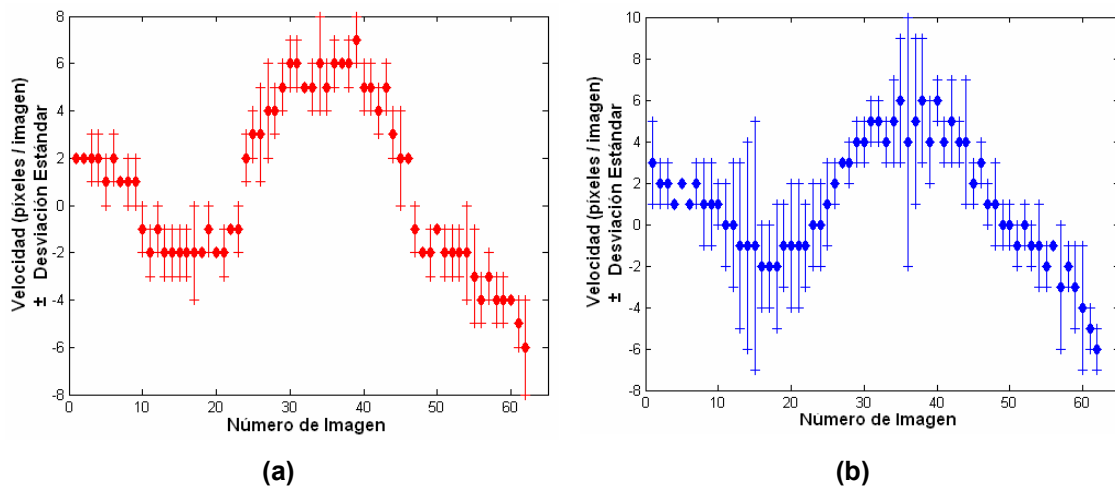


Figura 4.19. Resultados de la segmentación sobre la secuencia de evaluación.

En la Figura 4.20 se presentan los resultados de la eficiencia del algoritmo sobre la detección del módulo de la velocidad. Para la construcción de las graficas correspondiente a las medidas manuales (Figura 4.20b) hemos seguido un procedimiento igual al de la sección anterior.

En este caso existe un único personaje moviéndose por la escena, por eso ambas gráficas contienen una línea solamente. Obsérvese que un cambio en la dirección de movimiento del personaje se visualiza en la gráfica como un valor positivo (cuando el movimiento neto es hacia la derecha) o negativo (cuando el movimiento neto es hacia la izquierda).

Como en el caso estudiado en la sección anterior, existe una buena correlación entre los valores del módulo de la velocidad obtenidos de forma experimental y manual, por lo que el algoritmo es muy eficiente no sólo en la detección de la dirección de moviendo sino también en la detección del valor de la velocidad.



**Figura 4.20.** (a) Valor promedio del módulo de la velocidad obtenido mediante el algoritmo de detección de movimiento y desviación estándar de las medidas de velocidad experimentales; (b) Valor promedio del módulo de la velocidad obtenido de forma manual y desviación estándar de las medidas de velocidad manuales (es decir, a partir de las imágenes marcadas).

## 4.4. Evaluación III

Esta tercera secuencia pertenece a la película “El increíble hombre menguante” de Jack Arnold (1957), y está representada parcialmente por algunos fotogramas en la Figura 4.21. La secuencia, que consta de más de 250 fotogramas de tamaño 384 x 288, muestra un plano medio sobre un personaje que se mueve por la escena. La cámara, que se mantiene estática en su posición en la escena, gira sobre su propio eje para seguir al personaje en movimiento por la escena.



El fondo de la escena es estático, de manera que el único objeto con movimiento real es el personaje. Sin embargo, debido al movimiento giratorio que realiza la cámara, el escenario aparenta realizar un movimiento en sentido contrario al del personaje. Por tanto, esta secuencia presenta una dificultad añadida sobre las anteriores secuencias estudiadas en el proceso de evaluación, esto es, todos los elementos de la secuencia están en movimiento, bien debido a que tienen movimiento propio, bien debido a un movimiento aparente provocado por la rotación de la cámara.

Dado que el movimiento del personaje tiene lugar en perpendicular a la posición de la cámara, su tamaño se mantiene aproximadamente constante, no existe ningún efecto de perspectiva sobre el objeto.



**Figura 4.21.** Selección de algunos fotogramas correspondientes a la película “El increíble hombre menguante” de Jack Arnold, que vamos a utilizar para realizar la tercera evaluación del algoritmo de detección de movimiento.

En los ejemplos que hemos utilizado en las secciones anteriores para realizar la evaluación, la cámara era un elemento estático por lo que la velocidad del objeto en movimiento era una medida absoluta, es decir, se medía la variación, en imágenes consecutivas, en la posición del objeto con respecto a la escena, que era estática.

El ejemplo que analizaremos a continuación es más complejo en el sentido de que lo que obtendremos será la velocidad relativa entre el movimiento real del objeto y el movimiento aparente del escenario, debido al giro de la cámara. Bajo estas circunstancias, podría darse el

caso de que la cámara gire a tal velocidad que “enmascare” el movimiento real del objeto, es decir, que compense el cambio de posición del objeto en movimiento, de manera que este aparente estar siempre en la mismas coordenadas de posición con respecto a la escena, y por tanto, parecería que el objeto está estático; o también puede ocurrir que el movimiento de la cámara sea más rápido que el movimiento del objeto, con lo que el resultado será que el objeto parece moverse en sentido contrario que realmente se mueve.

Esta situación es común en aquellas aplicaciones en las que se procesan imágenes procedentes de cámaras en movimiento, y es, por tanto, uno de los casos más complejos con los que vamos a trabajar.

Como en los estudios realizados en las secciones anteriores, el primer paso consiste en la segmentación manual de los objetos que se mueven en la secuencia. En nuestro caso, vamos a segmentar solamente el objeto con movimiento propio. Trataremos el resto de objetos estáticos, cuyo movimiento aparente procede del de la cámara, como si fuesen un único objeto que ocupa la región complementaria de la segmentada manualmente.



**Figura 4.22.** Segmentación manual del personaje en movimiento en la secuencia bajo estudio.

Siguiendo los procedimientos de las secciones anteriores, vamos a obtener los parámetros que optimizan el rendimiento del algoritmo. En este caso vamos a centrarnos en el estudio de la umbralización y el tamaño del área de influencia, dado que nuestra imagen no contiene ruido de tipo “sal y pimienta” no será necesario usar una máscara de Sobel de mayor tamaño que 3x3; y tampoco usaremos áreas de correlación de tamaño superior a 1 píxel puesto que, como hemos visto en el estudio realizado sobre las secuencias anteriores, para los objetos en movimiento de tamaño similar al que nos encontramos en esta secuencia, la correlación entre estructuras de píxeles mayores no reporta un beneficio significativo.

Nótese que en el contexto actual, el objetivo de la parametrización puede ser función de la aplicación. Es decir, podríamos quedarnos con los parámetros óptimos para segmentar los objetos con movimiento propio, es decir, considerar el valor de los parámetros que obtienen los mejores índices de clasificación para los objetos que se mueven independientemente del movimiento de la cámara; podríamos optimizar aquellos parámetros que nos permiten estimar mejor el movimiento propio de la cámara; o, podríamos buscar aquellos parámetros que nos permitan optimizar los dos casos anteriores de forma conjunta. En el caso de la secuencia que

vamos a analizar, estaríamos hablando de optimizar los parámetros para obtener los mayores índices de  $SE_D$ ,  $SE_I$  o NC.

#### **a) Umbralización de los bordes:**

En esta sección vamos a estudiar la influencia del umbral utilizado en la etapa de extracción de bordes sobre  $SE_D$  y  $SE_I$  de forma similar a como se hizo con la primera secuencia.

Observamos que cuando el umbral está entre 25 y 40 tenemos un número de bordes superior a 150 para realizar la estimación del movimiento, y obtenemos valores a priori, es decir, sin optimizar los restantes parámetros, que son bastante altos, y que mejorarán a medida que se adecue el valor de estos al óptimo.

Como en los casos precedentes, también aquí mantenemos activos, después de la umbralización, un número de píxeles entre el 0.1 y el 0.2% del total de la imagen original, que servirán para realizar la estimación del movimiento en las siguientes etapas de procesamiento.

En la Figura 4.23 la tendencia de la Sensibilidad Derecha es creciente, podríamos considerar utilizar un umbral tal que obtengamos un valor de  $SE_D$  del 100%, pero obsérvese que el número de características que se mantiene activas disminuye con el aumento del umbral. Por tanto, tendríamos que estimar el movimiento en la secuencia a partir de un número muy bajo de características activas, y esta detección sería muy sensible a fenómenos ruidosos, como se vio en la Sección 2.3.

#### **b) Tamaño de las secciones o áreas de influencia local en el filtrado por coherencia espacial:**

A continuación vamos a obtener el valor del parámetro área de influencia que genera los máximos niveles de  $SE_D$  y  $SE_I$ . Seguiremos un procedimiento paralelo al de los estudios realizados anteriormente, es decir, estudiaremos la variación de  $SE_D$  y  $SE_I$  para valores de área de influencia de: 5x10, 10x15, 15x20, 20x25, 25x30, 30x35, 35x40, 40x45, 45x50, 50x55, 55x60, 60x65, 65x70, 70x75, 75x80, 80x85, 85x90, 90x95, 95x100, 100x105, 105x110, 110x115, 115x120, 120x125, 125x130, 130x135, 135x140 píxeles. Para ello, fijaremos el valor umbral en 25, 30, 35 y 40.

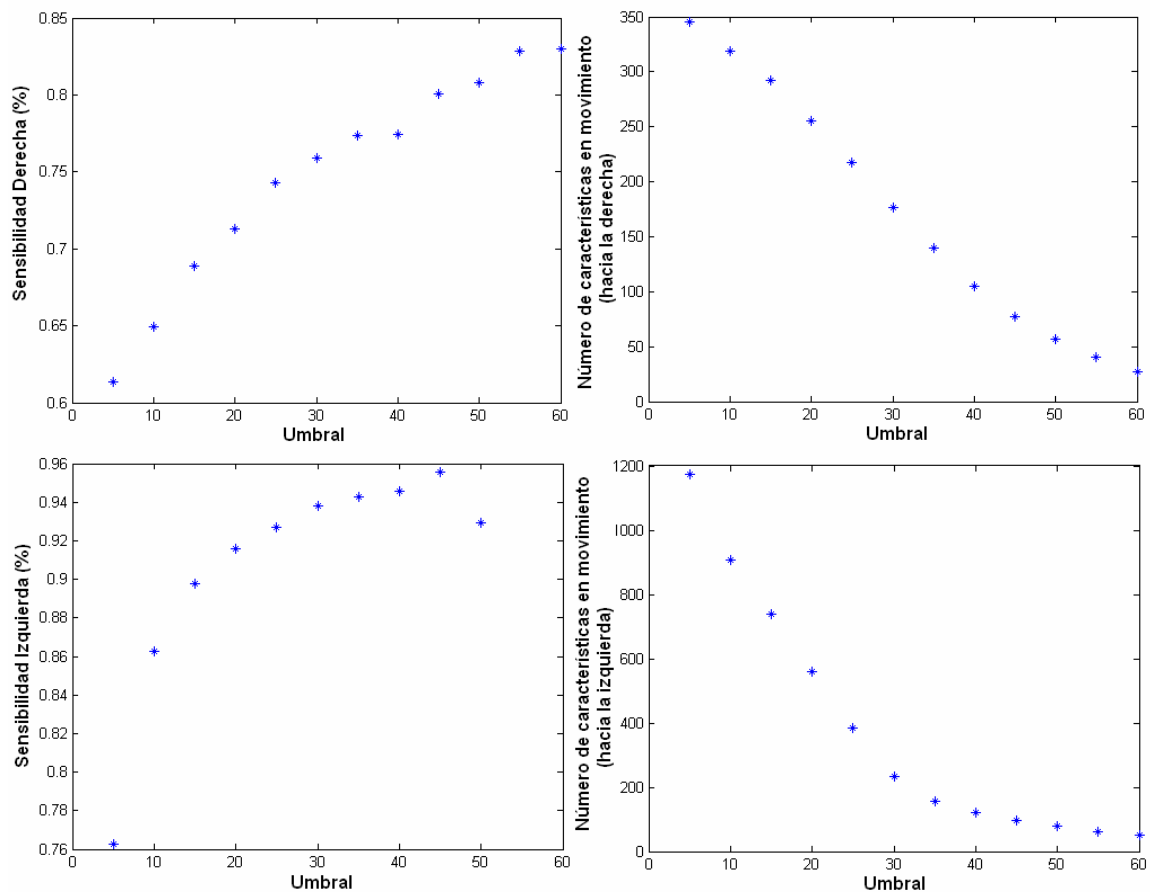
La Figura 4.24 resume los resultados de dicho estudio. De ella podemos concluir que, para los valores del umbral seleccionados en la Sección 4.4a:

- si la aplicación persigue la segmentación del objeto que tiene movimiento propio, el valor óptimo de los parámetros vendrá dado por aquellos que obtienen el máximo rendimiento de la Sensibilidad Derecha, es decir, obtendremos una  $SE_D$ : del 88% con un umbral de 25 y un área de influencia de 60x65 píxeles; del 91% con un umbral de 30 y un área de influencia de

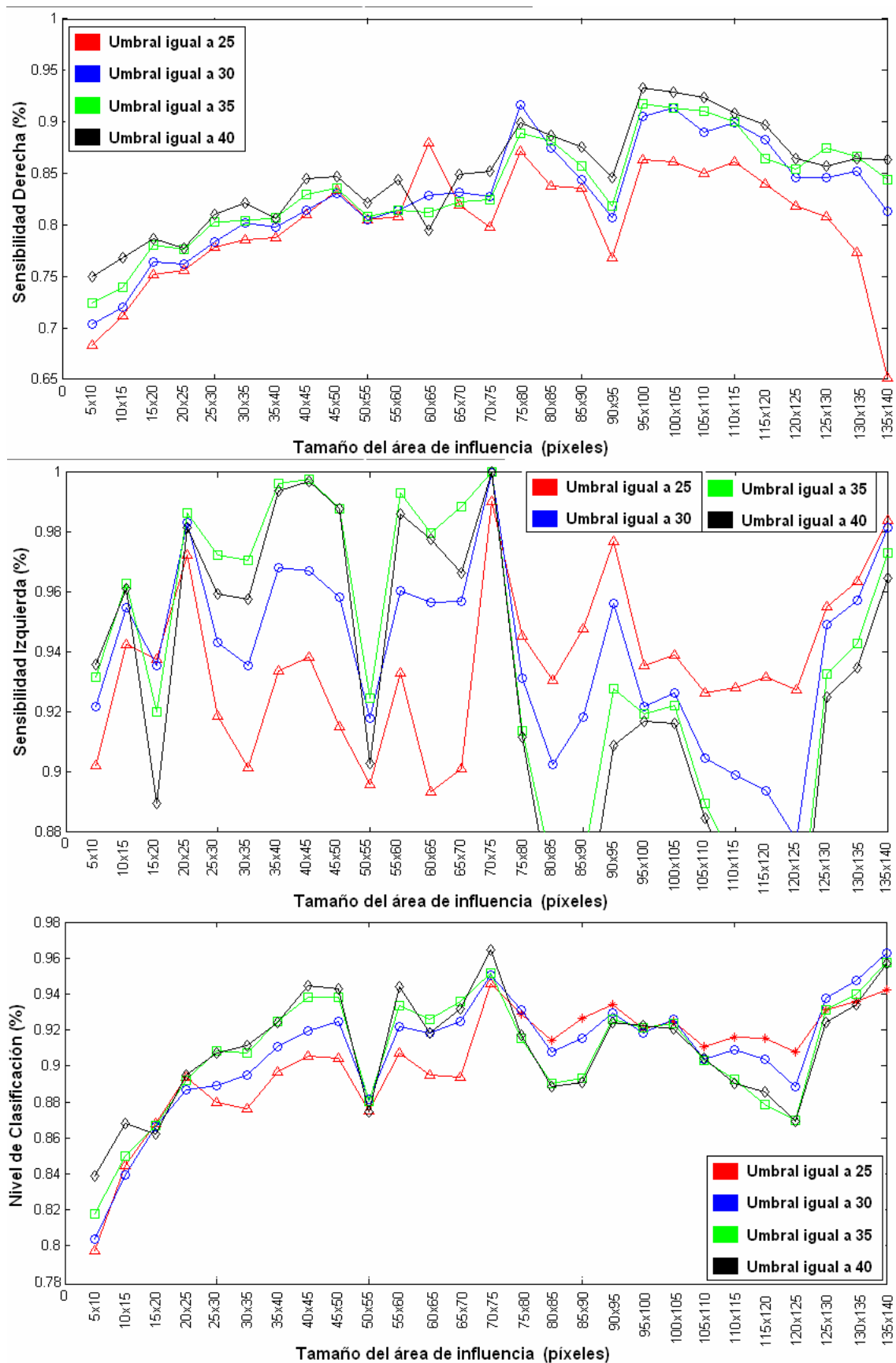
75x80 píxeles; del 93% con un umbral de 35 y un área de influencia de 95x100 píxeles; y del 94% con un umbral de 40 y un área de influencia de 95x100 píxeles.

- si la aplicación persigue el estudio del movimiento propio de la cámara deberemos obtener el máximo rendimiento de  $SE_I$ . Tendremos un valor de  $SE_I$  por encima del 98% para los umbrales de 35 y 40 para varios tamaños del área de influencia.

- si queremos optimizar la detección de movimiento para los dos casos anteriores de forma conjunta, tendremos que obtener el mayor valor de los posibles para el NC. Tendremos un NC por encima del 93% para distintos valores del área de influencia cuando utilizamos los umbrales de 35 y 40. Puesto que los objetos del entorno tienen menor tamaño que el personaje en movimiento los mejores resultados de  $SE_I$  se obtienen para tamaños de área de influencia menores de 75x80 píxeles, mientras que en el caso del objeto principal los mejores resultados de  $SE_D$  se obtienen para tamaños de ventana mayores, por eso, el NC es óptimo para un valor del área de influencia de 75x80 píxeles.



**Figura 4.23.** Las gráficas de la izquierda representan el promedio en la secuencia de  $SE_D$  y  $SE_I$  frente al umbral que se aplica en la etapa de extracción de bordes. Las gráficas de la derecha representan el número de bordes detectados que forman parte de los objetos en movimiento.



**Figura 4.24.** Valor promedio del porcentaje de la Sensibilidad Derecha, Sensibilidad Izquierda y Nivel de Clasificación para distintos tamaños de las áreas de influencia y para los valores del umbral que se seleccionaron en la sección anterior.

### **c) Efecto de los parámetros sobre el ruido del fondo**

En esta escena no es necesario el estudio del ruido de fondo en el sentido en el que se estudió en las secciones precedentes, ya que, debido al movimiento de la cámara, todo en la escena se mueve, como ya se comentó anteriormente.

### **d) Eficiencia del algoritmo sobre la secuencia**

Como se ha visto en la sección anterior, podemos obtener un valor de la Sensibilidad Derecha entre el 88 y el 91%, dependiendo del valor concreto que le demos a los parámetros umbral y área de influencia. Lo que significa que del total de características que pertenecen al objeto en movimiento (en nuestro caso es la mujer que se mueve por la escena) el 91% de ellas caracterizan correctamente la dirección de movimiento del personaje, es decir, se mueven hacia la derecha como este, y el 9% se mueven en dirección contraria.

En el caso de la Sensibilidad Izquierda, obtenemos una clasificación por encima del 98%. Es decir, el 98% de las características que pertenecen al escenario son detectadas moviéndose hacia la izquierda, por tanto, se mueven correctamente debido al movimiento de giro que realiza la cámara.

En la Figura 4.25 se muestra la salida del algoritmo de detección de movimiento para algunos fotogramas de la secuencia.

En la Figura 4.26 se muestra un ejemplo, extraído de la secuencia procesada, en el que el movimiento de la cámara enmascara el movimiento del personaje. Debido a que el giro de la cámara es más rápido que el desplazamiento por la escena del personaje, la salida del algoritmo de detección de movimiento muestra al personaje como estático o moviéndose en sentido contrario a su movimiento real.

En la Figura 4.27 se presentan los resultados de la eficiencia del algoritmo en cuanto a la detección del módulo de la velocidad. Para la construcción de las graficas de esta figura hemos seguido un procedimiento igual al de las secciones anteriores.

Si comparamos las gráficas (a) y (b), se puede decir que la detección en cuanto al módulo de la velocidad es bastante eficiente. Las pequeñas variaciones se pueden deber, como en los casos anteriores, a que contribuyen muchos más píxeles al cálculo de los valores experimentales que al de los valores manuales.

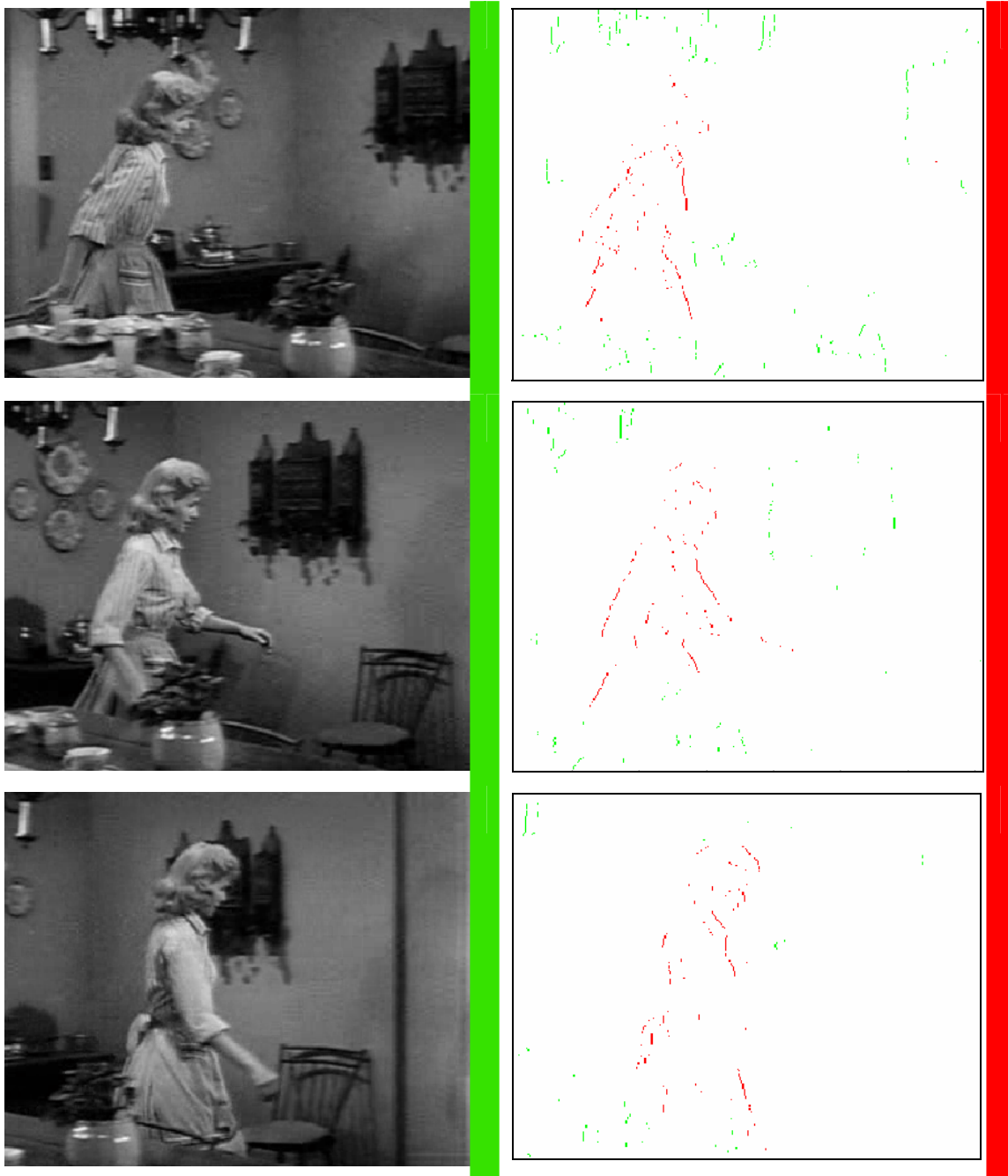
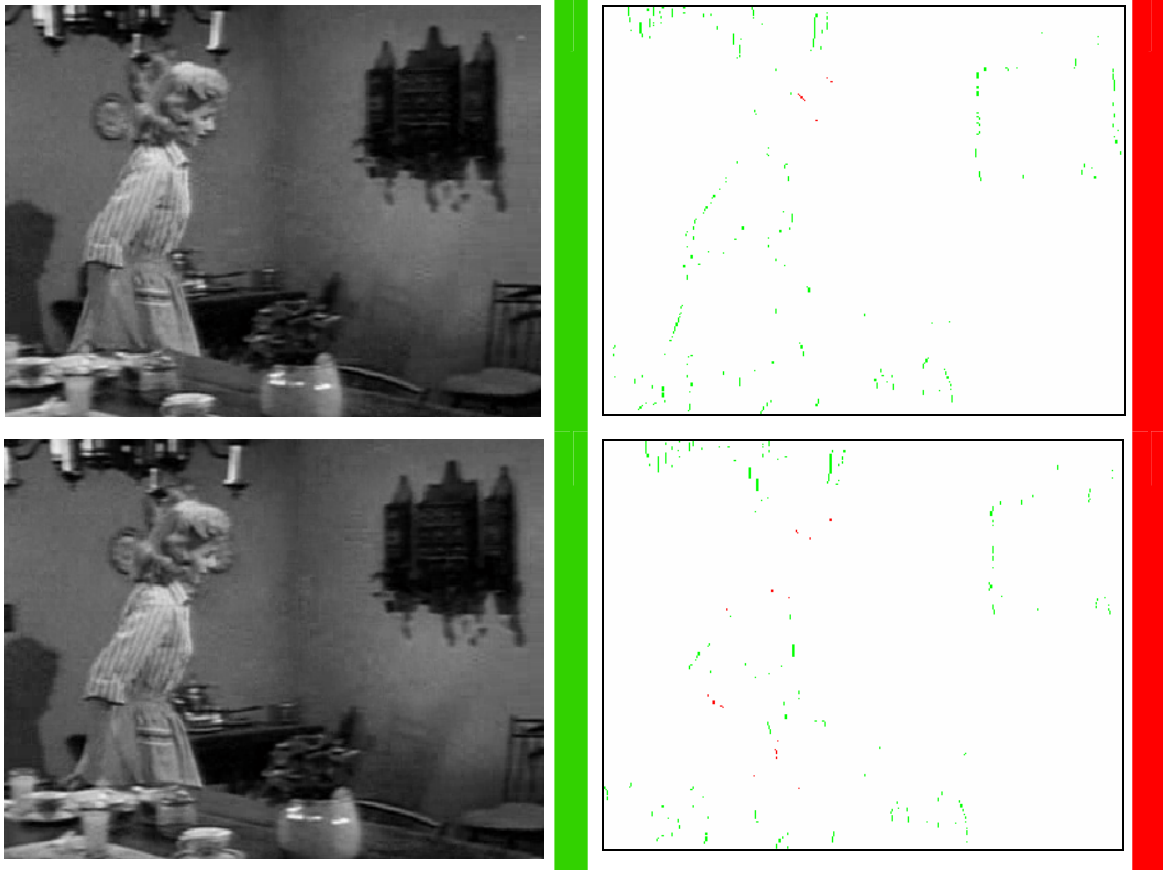


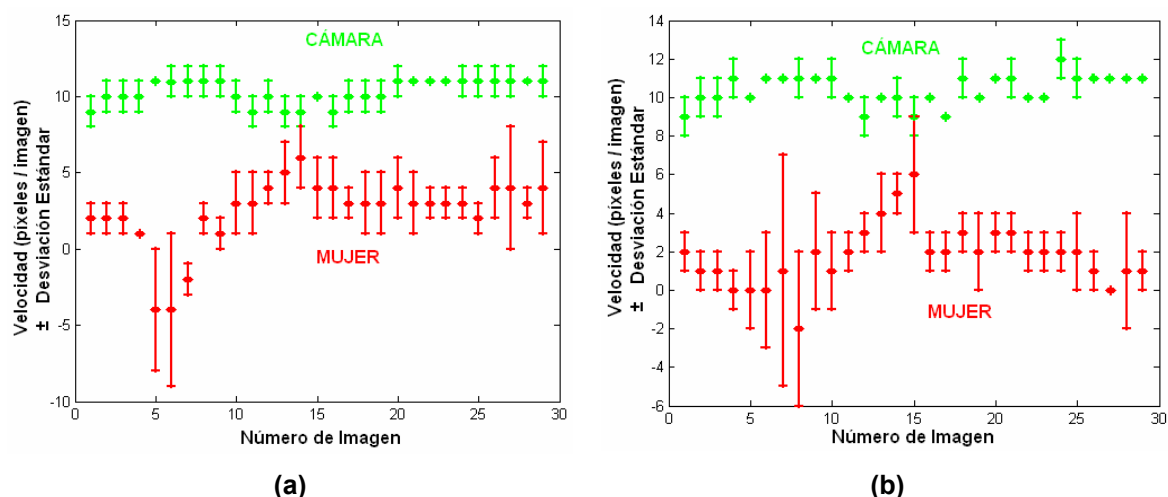
Figura 4.25. Imagen original y salida de la detección de movimiento de la secuencia bajo estudio.



**Figura 4.26.** Detecciones anómalas debidas al movimiento de la cámara. En la imagen superior el movimiento de la cámara provoca que el personaje aparente estar casi estático. En la imagen inferior el movimiento de la cámara provoca la sensación de que el personaje se mueve en sentido contrario al de su movimiento real.

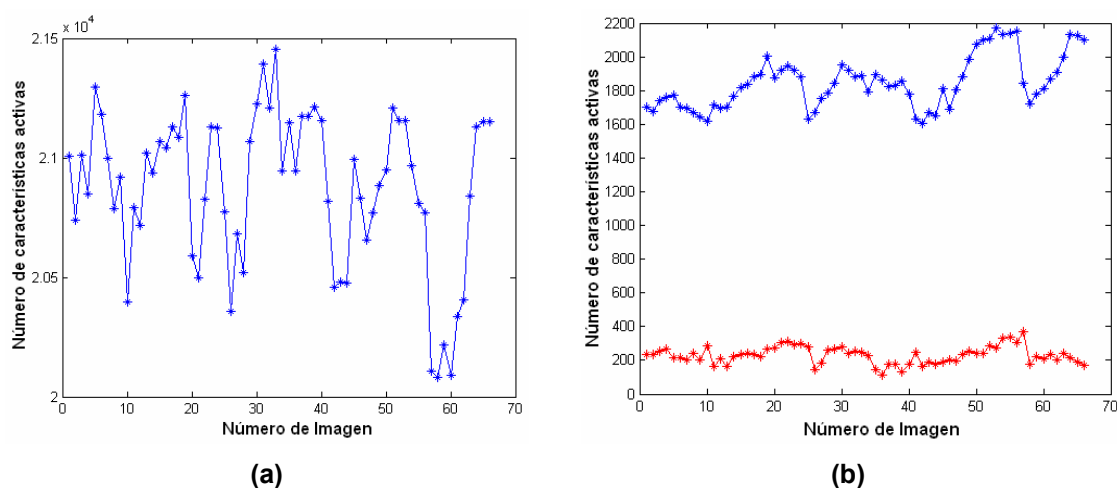
Obsérvese que, al principio de la secuencia, la velocidad de la mujer disminuye, mientras que la velocidad de la cámara aumenta ligeramente. La consecuencia inmediata de estos sucesos simultáneos en el tiempo es que el movimiento de la cámara provoca la sensación de que la mujer se desplaza en sentido contrario al que realmente lleva (véase la Figura 4.26), de ahí la existencia de valores negativos en las gráficas (a) y (b) de la Figura 2.27.





**Figura 4.27.** (a) Valor promedio del módulo de la velocidad obtenido mediante el algoritmo de detección de movimiento y desviación estándar de las medidas de velocidad experimentales; (b) Valor promedio del módulo de la velocidad obtenido de forma manual y desviación estándar de las medidas de velocidad manuales. El rojo representa el valor de la medida para el caso de la mujer, y el verde se utiliza para representar el valor de la medida para el movimiento aparente del fondo producido por el movimiento de la cámara.

En la Figura 4.28 se muestra el número de píxeles activos en cada etapa de procesamiento a lo largo de la secuencia. Se puede observar como se obtienen una alta eficiencia de la detección de movimiento con un número muy bajo de píxeles, en comparación con las dimensiones de la imagen de entrada, para la que cada imagen tiene 360 x 270 píxeles.



**Figura 4.28.** Degradación en el número de píxeles activos en cada etapa del procesamiento para la secuencia actual. (a) Número de características activas después de la extracción de bordes; (b) en azul se muestran las características activas que tenemos después de aplicar un umbral a los bordes, estas características son las que pasarán a la etapa de detección de movimiento; en rojo se muestran las características activas después de la etapa de filtrado mediante el criterio del sólido rígido.

## 4.5. Conclusiones

- ☑ Se ha realizado una evaluación del algoritmo presentado en el Capítulo 2 con secuencias reales, cada una de las cuales tiene un grado de dificultad mayor que la anterior. Las distintas secuencias estudiadas han sido tomadas tanto con cámaras estáticas como móviles.
- ☑ La elección de los parámetros del algoritmo viene condicionada por la aplicación que se quiera realizar, obteniéndose resultados optimizados en cada caso. Se ha realizado un estudio intensivo de distintas combinaciones de los parámetros de configuración relevantes para cada uno de los casos bajo estudio:

	Evaluación I		Evaluación II		Evaluación III	
	Se evaluó	¿Influye?	Se evaluó	¿Influye?	Se evaluó	¿Influye?
Umbralización de los bordes	SI	SI	SI	SI	SI	SI
Tamaño de la máscara de convolución de Sobel	SI	NO, sólo en imágenes ruidosas	NO	--	NO	--
Tamaño de la estructura de correlación	SI	NO	SI	SI, sólo para algunos tamaños del objeto	NO	--
Tamaño de las áreas de influencia	SI	SI	SI	SI	SI	SI

- ☑ El algoritmo demuestra una alta eficiencia en el filtrado de las vibraciones de la cámara que se transmiten a la imagen. Esta es una situación muy común, y por tanto, esta característica del algoritmo es muy interesante en aplicaciones reales.
- ☑ La eficiencia del algoritmo sobre la detección de la dirección del movimiento es muy alta, medida en términos de los parámetros Sensibilidad Derecha, Sensibilidad Izquierda y Nivel de Clasificación.
- ☑ La eficiencia del algoritmo sobre la detección del módulo de la velocidad es también muy alta considerando el valor medio de velocidad medido por el algoritmo y el valor medio obtenido de forma manual.
- ☑ La alta eficiencia del algoritmo de detección de movimiento se alcanza con un número muy reducido de píxeles, en comparación con las dimensiones de la imagen inicial. Esto implica reducción en los recursos de procesamiento necesarios.



---

## **CAPÍTULO 5:**

# **Aplicación a la monitorización de adelantamientos: Vigilancia del ángulo muerto**

---

En este capítulo se aplica el sistema de detección de movimiento en tiempo real a la vigilancia del ángulo muerto de los vehículos para la asistencia a los adelantamientos y cambio de carril en general. Los resultados de velocidad serán post-procesados para realizar la monitorización de los vehículos que nos siguen en carriles próximos. Se estudiarán los resultados bajo diferentes situaciones de iluminación y meteorológicas, y para la detección de distintos tipos de vehículos.

Además se incluirá un módulo de pre-procesamiento para la corrección de la deformación debida a la perspectiva.

## 5.1. Introducción

Cada año se producen miles de muertes y millones de heridos por accidentes de tráfico. Estas cifras han propiciado que se hable de las tres “C” como causas principales de muertes en el *primer mundo*: Corazón, Coches y Cáncer.

La tasa de muertes está relacionada con la tipología del accidente. En España, según datos de 2004 de la Dirección General de Tráfico [DGT04] extraídos de su página web, los accidentes que se produjeron más frecuentemente y que causaron mayor número de víctimas fueron las colisiones fronto-laterales (23.139 accidentes con 642 víctimas mortales), las colisiones por alcance (13.860 accidentes con 180 víctimas mortales), las colisiones laterales (7.905 accidentes con 125 víctimas mortales), las colisiones frontales (4.909 accidentes con 529 víctimas mortales) y las colisiones múltiples o en caravana (3.195 accidentes con 73 víctimas mortales).

Paralelamente al incremento en la siniestralidad, se promueve el desarrollo de nuevas tecnologías que tratan de atenuar las alarmantes cifras. En este sentido, hace unos años la mayor parte de las muertes se debían a choques frontales, sin embargo, el uso sistemático del cinturón de seguridad y la introducción en los vehículos de los airbag frontales ha reducido considerablemente el número de víctimas en este tipo de accidentes. Actualmente los accidentes por colisión lateral se han convertido en causantes de un número elevado de víctimas. En la Unión Europea 60.000 menores resultan heridos de diversa consideración cada año, y unos 750 mueren anualmente víctimas del tráfico. Se estima que casi siete de cada diez niños fallecidos lo son por impactos laterales [SAN01].

Hasta hace pocos años la industria automovilística apostaba por la inversión en sistemas de protección de los ocupantes (airbag, cinturones de seguridad, etc.), es decir, los llamados sistemas de seguridad pasiva o paliativa. Sin embargo, estos sólo pueden reducir los daños que se producen tras el accidente. Los errores humanos o las distracciones son la causa de un gran número de accidentes de tráfico (en aproximadamente el 93% de los casos), de acuerdo con los datos publicados por la Comisión Europea en el informe sobre transporte de 2001 [COM01]. Por ello, se está realizando un considerable esfuerzo, por parte de los investigadores de numerosas áreas (visión, control inteligente, fusión multimodal, etc.), en la aplicación de la tecnología más puntera a los automóviles, principalmente en el campo de asistencia a la conducción. Por eso, la industria automovilística está cambiando de estrategia, y está cada vez más interesada en introducir sistemas aplicados a la asistencia a la conducción que sirvan para prevenir el accidente en sí (sistemas de seguridad activa). Actualmente son comunes en una amplia gama de vehículos sistemas como la dirección asistida, sistemas antibloqueo (ABS), sistema electrónico de estabilidad (ESP), neumáticos más fiables, etc.

Una de las apuestas de la industria automovilística la constituyen los sistemas de alerta basados en la detección por radar [SCH03, EWA00], que son ampliamente utilizados en el tráfico aéreo y marítimo, y que, en áreas congestionadas, permiten detectar la trayectoria de otro transporte con tiempo suficiente para permitir el cambio de rumbo y evitar la colisión.

Debido a la densidad del tráfico terrestre, la aplicación de tales dispositivos de alerta en coches y camiones supone un reto mucho mayor que las aplicaciones en aviones y barcos. Detectar cambios en la posición, relativamente pequeños en relación con el entorno, que representen posibles amenazas, y hacer esto con la suficiente exactitud para discriminar cambios en la velocidad relativa o la dirección del movimiento es complejo, incluso para la tecnología actual. Sin embargo, debería ser factible, al menos, proporcionar una alerta al conductor.

La otra gran apuesta, no menos interesante, consiste en la introducción de dispositivos basados en visión artificial que ayudarán en la conducción [HAN98, GOR98, FRA00, FLE03, BRO04]. Se estima que en 10 años los vehículos dispondrán de dichos dispositivos con plena funcionalidad, aunque ya se están empezando a introducir, de forma tímida y con funcionalidad limitada, en distintos modelos de coches de gama alta o media-alta.

Aunque ambas tecnologías tienden a competir por el mercado, en algunos trabajos se están utilizando sistemas mixtos, que usan conjuntamente las aproximaciones de radar y visión artificial [HEI96, FAN02] aprovechando las ventajas que cada una de ellas tiene, y minimizando los inconvenientes.

La construcción de vías rápidas (autovías y autopistas) ha contribuido al aumento de la seguridad y la reducción de cierto tipo de accidentes, sin embargo, otras situaciones peligrosas no se han visto beneficiadas. Una de las operaciones más peligrosas durante la conducción es el cambio de carril con el objetivo de adelantar a un vehículo precedente.

En autovías y autopistas hay dos a más carriles paralelos para cada sentido de circulación, a veces con una gran densidad de vehículos circulando por ellos. El conductor centra su atención al frente, en el camino que tiene que recorrer. Cuando realiza el cambio de carril, a veces no utiliza el espejo retrovisor o este no resulta de utilidad debido a la existencia del ángulo muerto, produciéndose situaciones de riesgo considerable. El espejo retrovisor resulta de poca ayuda cuando, durante la maniobra de cambio de carril, el vehículo que adelanta (que denominaremos *vehículo objetivo*) se posiciona en el ángulo muerto del vehículo adelantado (que denominaremos *vehículo vigilante*), o cuando el conductor del vehículo adelantado no utiliza dicho espejo retrovisor antes de realizar el cambio de carril para asegurarse de que puede realizar la maniobra sin peligro.

Este mismo problema lo encontramos en carreteras con un único carril para cada uno de los sentidos. Cuando se produce una maniobra de adelantamiento. En este caso, además de la colisión lateral que se podría producir con el cambio de carril, también se puede producir una colisión frontal, con lo que se incrementa la peligrosidad de la maniobra.

Otras situaciones, comunes en la circulación por ciudad, también se convierten en peligrosas para vehículos más frágiles, como bicicletas o motos, debido al ángulo muerto del espejo retrovisor. La incorporación a otra vía o la apertura de las puertas de los vehículos aparcados producen situaciones muy peligrosas (véase las Figuras 5.1a y 5.1b). Según las estadísticas, en Bélgica muere un ciclista diariamente debido al ángulo muerto de los vehículos [BEL03].

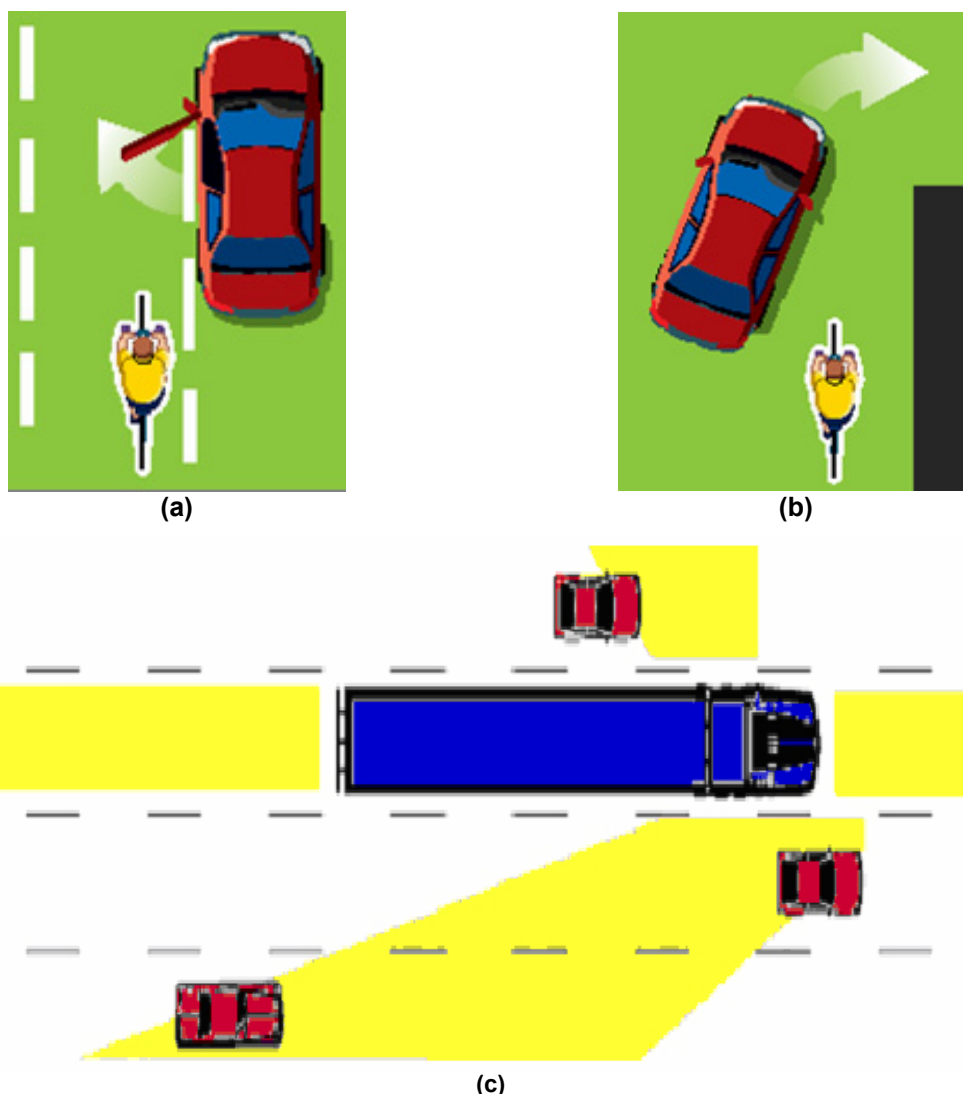
Si la existencia del ángulo muerto resulta bastante peligrosa para los vehículos ligeros, la situación se agrava en el caso de los vehículos pesados (camiones) que tiene un número aún mayor de ángulos muertos de visión (véase la Figura 5.1c).

Cada año, un gran número de personas son víctimas de accidentes por causa del giro a la derecha de los camiones. La principal causa de estos accidentes es la mala visibilidad que tiene el conductor en el lado derecho de su vehículo. Aunque tienen la obligación de instalar tres espejos retrovisores en dicho lado derecho, todavía existe un considerable ángulo muerto sin cubrir. No hay estadísticas serias a nivel europeo de este tipo de accidentes, sin embargo, sí que existen ejemplos de la situación en Holanda. En el año 2000 se produjeron 19 muertos y 47 heridos graves debido al giro a la derecha de camiones, y estas estadísticas se mantienen constantes año a año [EUR06]. Extrapolando estos datos al resto de la Unión Europea podemos derivar que en torno a 500 accidentes de los 40.000 al año son producidos por el giro a la derecha de camiones.

Este problema podría resolverse o al menos reducirse considerablemente equipando los vehículos pesados con espejos retrovisores especiales, cuyo coste se estima en torno a los 225€) o un sistema que combine cámara y monitor (de coste aproximado de 900€). Sin embargo, los sistemas basados en espejos no consiguen evitar el ángulo muerto, solamente lo desplazan, y tanto los espejos como los sistemas basados en monitores requieren de la atención del conductor, que a veces no se aplica por imprudencia o por distracción. En el caso de los monitores en cabina, recientemente se ha incorporado una nueva normativa al Reglamento General de Circulación (publicado en el número 306 del Boletín Oficial del Estado), según la cual “Queda prohibido el uso por el conductor con el vehículo en movimiento, de dispositivos que puedan distraer su atención mientras conduce, tales como pantallas con acceso a Internet, monitores de televisión y reproductores de vídeo o DVD. Excepto monitores necesarios para maniobras y dispositivo GPS” (Art. 18.1), lo cual hace que estos dispositivos dependan de la homologación por parte de la Dirección General de Tráfico.

Estas iniciativas de la industria automovilística se ven avaladas por la nueva normativa comunitaria relativa a la seguridad en el transporte. Recientemente, en el 2003, el Parlamento Europeo adoptó la Directiva 2003/97/EC sobre los espejos retrovisores y sistemas suplementarios de visión indirecta para los vehículos a motor, cuyo objetivo es mejorar la seguridad por carretera mejorando el rendimiento de los espejos retrovisores y acelerando la introducción de nuevas tecnologías que incrementen el campo de visión indirecta de los conductores de coches, autobuses y camiones. Esta Directiva fue reformada posteriormente con la 2005/27/EC que extiende la instalación de espejos retrovisores de gran ángulo a más clases de vehículos.

En el escenario descrito, si un vehículo se acerca adelantando, observaremos que se mueve hacia delante, mientras que el resto de la escena se mueve hacia atrás debido al movimiento propio del vehículo vigilante. Por tanto para esta aplicación, un sistema basado en la detección del movimiento puede ser una herramienta que nos proporcione indicios suficientes para segmentar los objetos que se mueven hacia el vehículo vigilante.



**Figura 5.1. a) y b)** Distintas situaciones peligrosas que se dan durante la conducción debido a la existencia de ángulos muertos o zonas ciegas en distintos vehículos. **c)** La figura muestra sombreado en amarillo las zonas ciegas de los camiones.

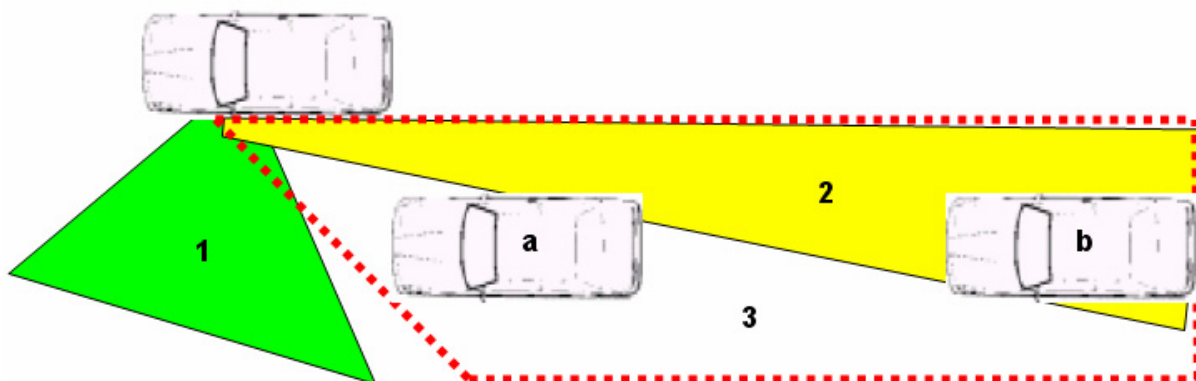
En este capítulo usamos el procedimiento de detección de movimiento descrito en los capítulos anteriores como sistema de asistencia a la conducción, para monitorizar los adelantamientos, vigilando el ángulo muerto del vehículo vigilante.

El sistema que se propone, además de la extracción de movimiento descrita, realizará un post-procesamiento de los datos de velocidad, de esta forma detectará al *vehículo objetivo* (es decir, el vehículo más próximo al vehículo vigilante); discriminará si se está aproximando o no; y lanzará una alerta local sólo en caso de peligro, es decir, cuando el vehículo objetivo se sitúe en las proximidades del ángulo muerto o en una posición preestablecida. Para ello usaremos la imagen capturada por una cámara situada sobre el espejo retrovisor izquierdo.

La Figura 5.2 ilustra el planteamiento del problema. El área (1) corresponde a la visión directa del conductor del vehículo vigilante; para que todas las posiciones de este área sean alcanzables el conductor debe girar la cabeza cierto ángulo hacia la izquierda, perdiendo de vista su camino. El área



(2) es la que puede alcanzar a ver el conductor a través del espejo retrovisor. El área (3) es la que cubre la cámara que vamos a utilizar. Un vehículo en la posición (b) podría ser visto tanto por la cámara como por el conductor a través del espejo retrovisor. Por otro lado, cuando el vehículo objetivo se posiciona en el ángulo muerto del vehículo vigilante (posición a), sólo la cámara puede detectarlo. Esta será la zona monitorizada por el dispositivo. Sin embargo, para lanzar la alarma local será necesario segmentar y seguir al vehículo objetivo desde una posición anterior.



**Figura 5.2.** Planteamiento del problema al que aplicaremos en sistema de detección de movimiento.

Para la evaluación del algoritmo en esta aplicación hemos usado secuencias de adelantamiento reales tomadas con una cámara situada sobre el espejo retrovisor izquierdo. Estas imágenes han sido grabadas por la empresa automovilística Hella mediante un coche instrumentado, para ello han usado una cámara de alto rango dinámico (High Dynamic Range, HDR). Las secuencias se componen de al menos 50 imágenes con una resolución de 640x480 píxeles por imagen, y 256 niveles de gris, y con una tasa de transmisión de 25 imágenes por segundo.

La base de datos de adelantamientos incluye secuencias bajo diferentes condiciones climatológicas (lluvia, niebla, soleado, nublado) y de iluminación (medio día, amanecer, atardecer, anochecer). Las secuencias también incluyen el adelantamiento a diferentes velocidades relativas entre el vehículo objetivo y el vehículo vigilante, que se han medido con un sistema basado en luz láser (LIDAR) [MER06].

La primera parte de éste capítulo consistirá en la evaluación del algoritmo para su uso en este tipo de aplicación, es decir, evaluaremos la fiabilidad y exactitud de los datos de movimiento extraídos bajo distintas condiciones.

A continuación se utilizarán los datos de movimiento para hacer un seguimiento del vehículo objetivo y para generar una señal de alerta; se estudiará la capacidad de alerta del sistema total.

Finalmente se estudiarán algunas innovaciones que disminuyan los problemas que ocasiona el efecto de la perspectiva, mejorando la detección.

Un sistema basado en FPGAs como el descrito en los capítulos anteriores tiene una ventaja fundamental al utilizarse para una aplicación como la que se aborda en este capítulo, esta es el concepto de tiempo real (gracias a su alta capacidad de procesamiento en paralelo), un factor muy

importante en cualquier sistema de alerta, y que dota al dispositivo de funcionalidad cuando la alerta se activa sin retrasos.

## **5.2. Evaluación del algoritmo de detección de movimiento en tareas de monitorización de adelantamientos bajo diferentes condiciones de trabajo**

En todas las secuencias de la base de datos se observa cómo se aproxima uno o varios vehículos (vehículo objetivo) durante una maniobra de adelantamiento, y por el carril contiguo al que ocupa el vehículo desde el que se realiza la observación (vehículo vigilante). La cámara en este caso es móvil, ya que está situada sobre el retrovisor del conductor de un coche en marcha. Por tanto, todo en la escena está en movimiento. Sin embargo, el movimiento en la imagen está muy estructurado, todo el paisaje se mueve hacia atrás (o hacia la izquierda de la imagen), desde la perspectiva del vehículo de observación; excepto el vehículo objetivo que se mueve hacia delante (o hacia la derecha de la imagen), y a mayor velocidad que el vehículo vigilante, y serán la fuente de potencial peligro.

Puesto que solamente una pequeña parte de la imagen se mueve hacia la derecha (el vehículo que realiza la maniobra de adelantamiento), la escena es claramente no simétrica en cuanto al número de características que se mueven en cada dirección, como ocurría en la Evaluación III del Capítulo 4.

La aplicación que proponemos en el presente capítulo es, sin duda, una de las más complejas con las que podemos trabajar. En primer lugar, debido a que todo en la escena está en movimiento real o provocado por el movimiento de la cámara. En segundo lugar, debido a la posición de la cámara en el vehículo vigilante y al área que tiene que vigilar, existe un efecto de perspectiva que afecta a varios niveles. Por una parte, el vehículo objetivo no mantiene un tamaño constante a lo largo de la detección, sino que aumenta de tamaño a medida que se aproxima al vehículo vigilante, por ello además del movimiento de aproximación existirá un movimiento de expansión. Por otro lado, la velocidad del vehículo objetivo tampoco será constante, sino que el efecto de la perspectiva hace que el movimiento se detecte como acelerado. Más aún, las partes delantera y trasera de un mismo vehículo se detectarán moviéndose a distinta velocidad debido al efecto de la perspectiva, lo cual complica aún más el empleo de un algoritmo no denso, basado en características de la imagen y en la detección de sólidos rígidos como el presentado aquí, porque en esta aplicación dos partes de un mismo sólido rígido se mueven a velocidades ligeramente distintas.

Teniendo en cuenta todos los problemas anteriores, en este capítulo se evaluará el algoritmo de detección de movimiento para determinar su viabilidad como sistema de monitorización para la asistencia a los adelantamientos. El algoritmo se va a someter a una serie de pruebas bajo diferentes situaciones propias de la aplicación:

1. Estudio del rango de velocidades relativas para las que el dispositivo es capaz de detectar movimiento.
2. Estudio bajo diferentes condiciones luminosas (mediodía, atardecer, anochecer,...).
3. Estudio bajo diferentes condiciones climatológicas (día soleado, lluvioso, con niebla,...).
4. Estudio respecto a la detección de distinto tipo de vehículos objetivo (motos, coches, camionetas, camiones, etc.).

Lo primero que debemos hacer es establecer los parámetros del algoritmo que son óptimos para la detección del movimiento de la escena para todas las secuencias.

Hemos seguido un proceso similar al realizado en el capítulo de evaluación del algoritmo. Sobre todo se han estudiado el umbral que se aplica a los bordes (para que el funcionamiento sea óptimo bajo cualquiera de las condiciones de iluminación que nos vamos a encontrar) y el tamaño de las áreas de influencia (que deberá ser tal que podamos segmentar cualquier tipo de vehículo usando sólo el criterio de sólido rígido). Como vimos en el Capítulo 4 éstos son los parámetros más influyentes en una correcta detección de movimiento.

### **5.2.1. Estudio del rango de velocidades detectadas**

Para evaluar las prestaciones del dispositivo, para esta aplicación, debemos considerar la exactitud de la detección en función del rango de velocidades relativas que nos vamos a encontrar entre el vehículo objetivo y el vehículo vigilante. Para ello, vamos a utilizar un conjunto de cuatro secuencias en las que el adelantamiento se produce a diferentes velocidades relativas. En ellas el vehículo objetivo se aproxima al vehículo vigilante a velocidades dentro de los rangos 1-3 m/s, 5-10 m/s, 15-20 m/s y 25-30 m/s.

La única diferencia entre las cuatro secuencias evaluadas en esta sección es la velocidad relativa a la que tiene lugar el adelantamiento. Se mantienen constantes otros factores que podrían influir en la detección, y que se estudiarán posteriormente, como son la iluminación (todas las secuencias se han tomado en las mismas horas del día), las condiciones meteorológicas (que son las mismas para las cuatro secuencias), el color del vehículo objetivo (en todas las secuencias tenemos el mismo vehículo objetivo), el paisaje, etc.

En el escenario de adelantamiento el vehículo objetivo realiza un movimiento de aproximación que contrasta con el paisaje y las marcas viales de la carretera, que aparentan moverse en dirección contraria debido al movimiento propio de la cámara (véase la Figura 5.3).



**Figura 5.3.** Selección de algunas imágenes correspondientes a una de las secuencias que vamos a utilizar para realizar el estudio del rango de velocidades detectadas por el algoritmo.

Para realizar la evaluación hemos seguido un proceso de segmentación manual igual al seguido en el capítulo de evaluación del algoritmo. Hemos marcado el vehículo objetivo en cada imagen de la secuencia mediante un rectángulo (véase una muestra en la Figura 5.3). Este proceso se ha realizado de forma manual, de manera que los límites del rectángulo son conocidos en cada imagen de la secuencia, y por tanto, es fácil determinar el centro del mismo.

Procedemos de la misma forma que en el capítulo de evaluación, es decir en una primera etapa etiquetamos el movimiento de las características que pertenecen al interior del rectángulo con movimiento hacia la derecha, mientras que todas las características del exterior del rectángulo se etiquetan con movimiento hacia la izquierda.

En una segunda etapa, calculamos la Sensibilidad Derecha ( $SE_D$ ) del algoritmo comparando las etiquetas con los resultados experimentales de detección de movimiento del mismo.

Podemos calcular, así mismo, el *Rendimiento de la Segmentación* ( $R_S$ ) del algoritmo definido como:

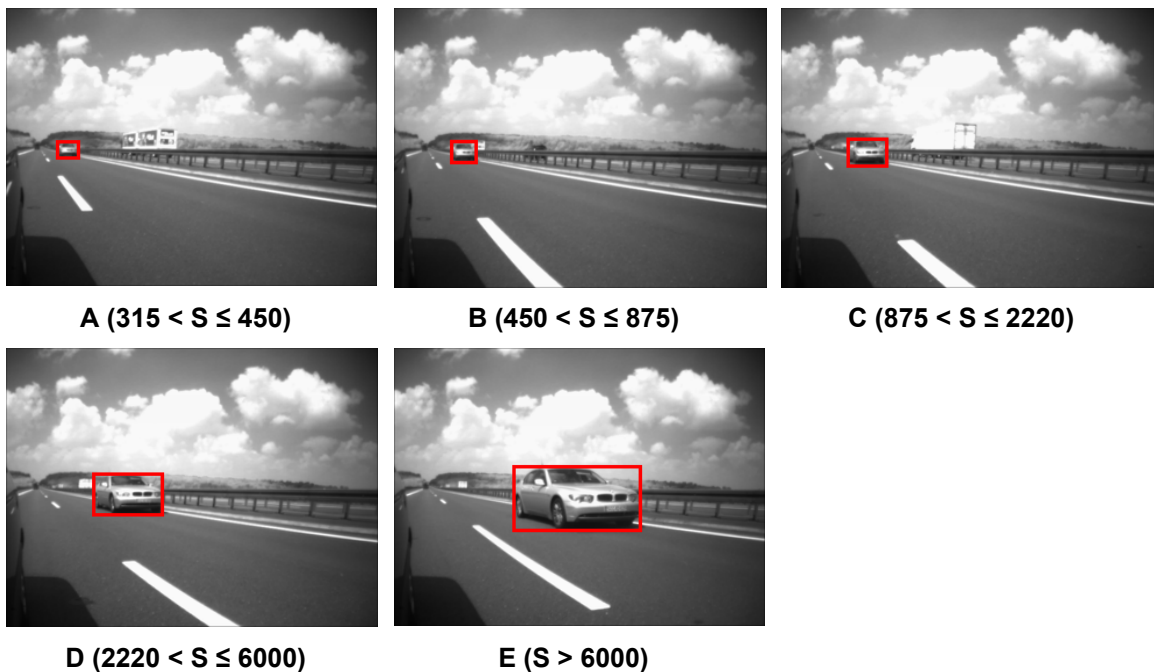
$$R_S = \frac{VD}{VD + FI} \quad (1)$$

Donde, tal y como se definieron en el Capítulo 4,  $VD$  (Verdadero Derecha) representa a las características que pertenecen al vehículo objetivo, por lo que se etiquetan moviéndose hacia la derecha, y el algoritmo las detecta moviéndose hacia la derecha; mientras que  $FI$  (Falso Izquierda) son los píxeles del entorno que se detectan moviéndose hacia la derecha, pero por pertenecer al fondo deberían detectarse con movimiento hacia la izquierda.

El Rendimiento de la Segmentación ( $R_S$ ) nos indica la contaminación por ruido con movimiento hacia la derecha que tiene la detección de movimiento. Valores próximos a cero indican

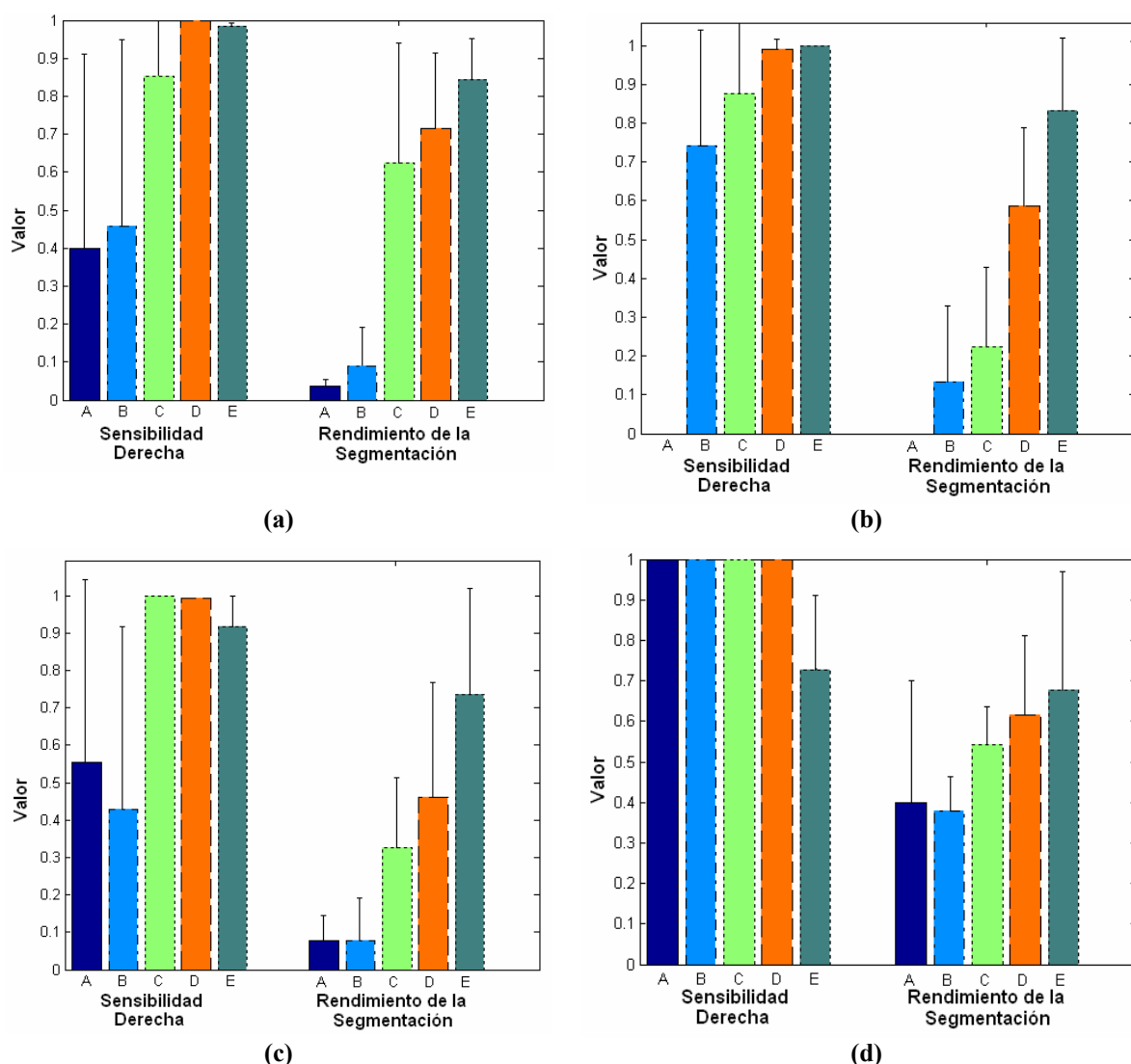
una gran proporción de ruido respecto de la cantidad de píxeles pertenecientes al vehículo objetivo correctamente determinados por el algoritmo, mientras que valores próximos a uno indican una detección con poco ruido.

Tanto la Sensibilidad Derecha ( $SE_D$ ) como el Rendimiento de la Segmentación ( $R_S$ ) van a depender no sólo de la velocidad relativa entre los dos vehículos, sino del tamaño del vehículo objetivo. Por ello, vamos a distinguir diferentes casos dependiendo de dicho tamaño. Para estimar los diferentes tamaños vamos a considerar el área del rectángulo dibujado manualmente ( $S$ ) que contiene al vehículo objetivo, y expresaremos dicha área mediante el número de píxeles contenidos en ella. Así tendremos vehículos objetivo de tamaños: **A** ( $315 < S \leq 450$ ), **B** ( $450 < S \leq 875$ ), **C** ( $875 < S \leq 2220$ ), **D** ( $2220 < S \leq 6000$ ) y **E** ( $S > 6000$ ). En la Figura 5.4 se pueden ver algunos ejemplos de los diferentes tamaños considerados.



**Figura 5.4.** Distintos tamaños de vehículo objetivo considerados. La figura muestra el vehículo objetivo cuando está en el límite inferior de cada uno de los rangos de tamaño.

En la Figura 5.5 se pueden ver los resultados de  $SE_D$  y de  $R_S$  para los diferentes rangos de velocidad.



**Figura 5.5.** Media y Desviación Estándar de la Sensibilidad Derecha ( $SE_D$ ) y Rendimiento de la Segmentación ( $R_S$ ) para los tamaños A, B, C, D y E cuando el vehículo objetivo se aproxima a: **(a)** 1-3 m/s, **(b)** 5-10 m/s, **(c)** 15-20 m/s y **(d)** 25-30 m/s.

En los resultados anteriores podemos observar que tanto la Sensibilidad Derecha como el Rendimiento de la Segmentación aumentan a medida que el vehículo objetivo se aproxima al vehículo vigilante. Esto se debe, en primer lugar, a que cuando el vehículo objetivo está muy lejos hay muy pocos píxeles que le pertenecen en la imagen con lo que es difícil definir el sólido rígido, por otro lado el ruido existente en el entorno es muy alto en comparación con los píxeles reales del vehículo objetivo por lo que tenemos valores de  $R_S$  cercanos a cero. Además, cuando el vehículo objetivo está muy lejos la velocidad detectada puede ser subpíxel (es decir, inferior a un píxel por imagen), y dado que la capacidad de detección del algoritmo es de un píxel como mínimo, puede existir una detección de la velocidad errónea o falta de detecciones.

A distancias medias y cortas, cuando el vehículo objetivo es suficientemente grande (tamaños C, D y E), el vehículo objetivo se segmenta bastante bien, es decir, se detectan correctamente suficientes características en movimiento que le pertenecen, además existe una cantidad ruido cada vez menor.

Obsérvese que en las gráficas de la Figura 5.5 se está realizando el promedio de las medidas en las imágenes en las que el tamaño del vehículo objetivo está dentro del rango, y en algunos casos existe una gran variabilidad para esas medidas dentro de dichas imágenes.

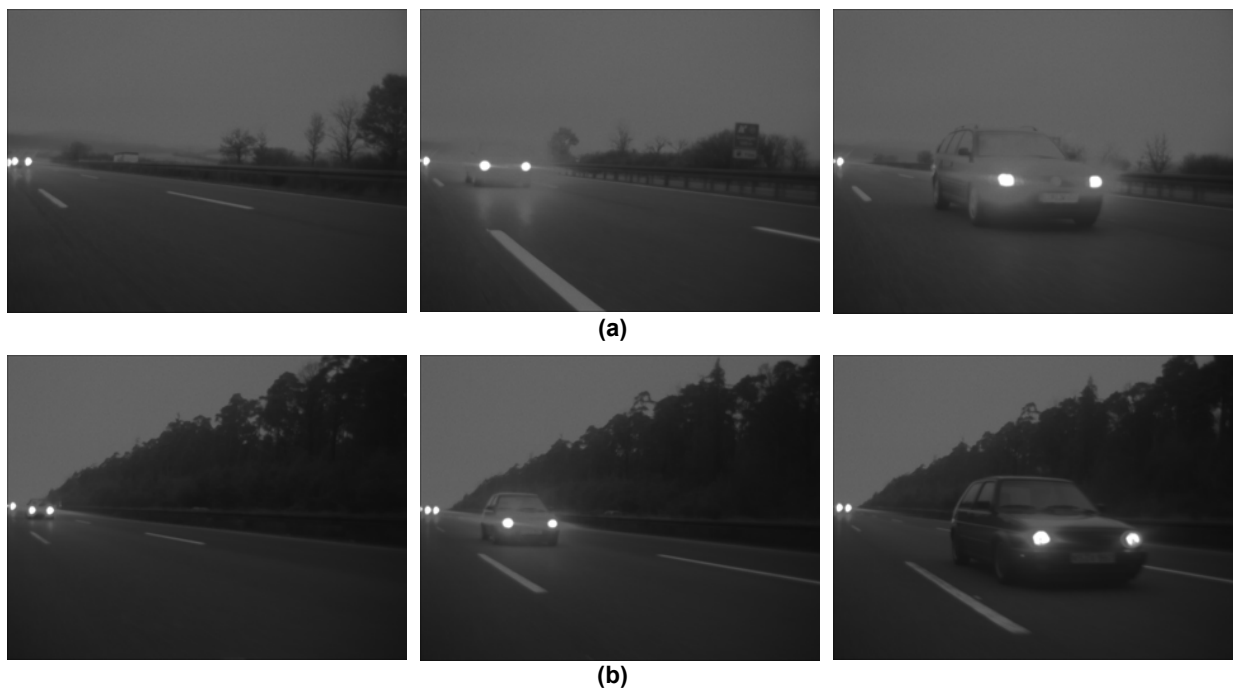
La secuencia (b) de la Figura 5.5 no presenta ninguna imagen en la que el vehículo objetivo tenga un tamaño de rango A, por eso no hay ninguna medida en este caso.

La variabilidad en el rango de velocidades que se produce a lo largo de la secuencia (debido al efecto de la perspectiva), y que es capaz de afrontar el algoritmo de detección de movimiento presentado, con desplazamientos entre imágenes consecutivas entre 1 y más de 60 píxeles (dependiendo de las velocidades relativas entre los vehículos), es difícil de procesar por otros modelos de visión bio-inspirados existentes en la bibliografía, que dan buenas respuestas a movimientos lentos o rápidos, pero que difícilmente abarcan todo el rango de velocidades debido al tamaño de los filtros espacio-temporales que implementan, estos algoritmos alternativos sólo pueden afrontar este rango de velocidades mediante el uso de estrategias multiescala.

## 5.2.2. Estudio bajo diferentes condiciones luminosas

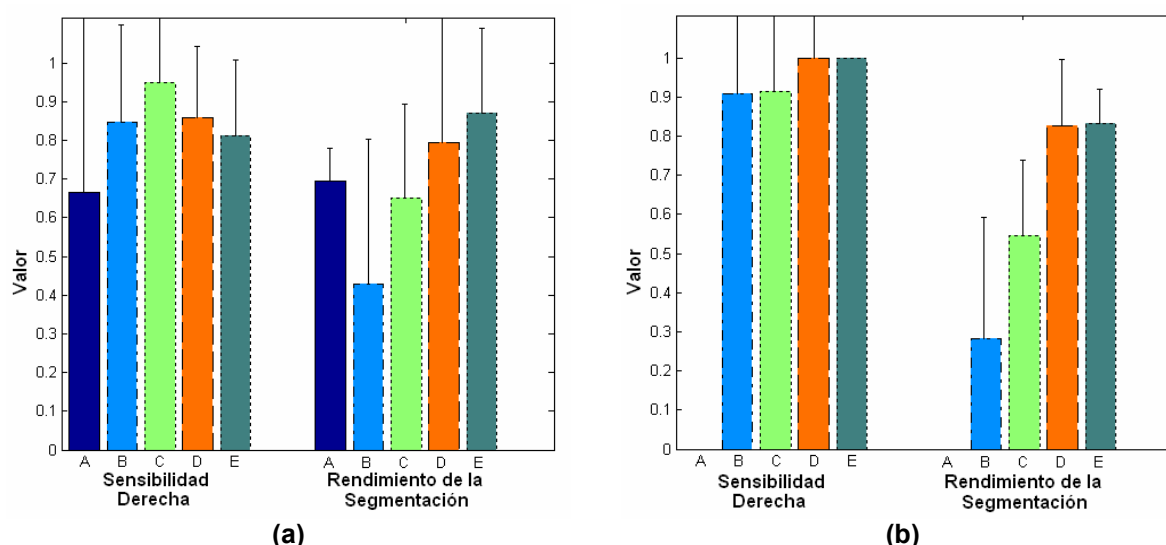
El sistema propuesto debe ser útil a cualquier hora del día, por eso es importante estudiar su funcionalidad bajo diferentes condiciones de iluminación.

La Figura 5.6 muestra las dos secuencias bajo estudio, en las horas del anochecer.



**Figura 5.6.** Secuencias tomadas al anochecer.

La Figura 5.7 muestra los resultados de la Media y la Desviación Estándar de la Sensibilidad Derecha y Rendimiento de la Segmentación para los tamaños A, B, C, D y E en las dos secuencias al anochecer.



**Figura 5.7.** Media y Desviación Estándar de la Sensibilidad Derecha y Rendimiento de la Segmentación para los distintos tamaños del vehículo objetivo en secuencias tomadas al anochecer: **(a)** resultados para la secuencia (a) de la Figura 5.6; **(b)** resultados para la secuencia (b) de la Figura 5.6.

En las Figuras 5.7 a y b se observa que a pesar de que al anochecer hay un contraste muy bajo en la imagen y, por lo tanto no hay bordes fuertes (con un nivel de intensidad alto que sobrepase el umbral de filtrado), tanto la Sensibilidad Derecha como el Rendimiento de la Segmentación son muy altos para todos los tamaños del vehículo objetivo. Esto se debe a que con ese nivel de luz es obligatorio circular con las luces encendidas, lo cual genera un área de alto contraste con unos bordes muy bien definidos y fáciles de seguir. Mientras que en el resto de la imagen, al no existir apenas bordes por el bajo contraste tampoco hay ruido.

La secuencia (b) de la Figura 5.6 no presenta ninguna imagen en la que el vehículo objetivo tenga un tamaño de rango A, por eso no hay ninguna medida en este caso.

### 5.2.3. Estudio bajo diferentes condiciones climatológicas

Como en el caso anterior, el sistema debe funcionar correctamente bajo cualquier condición meteorológica que pueda alterar la visibilidad. Por tanto, a continuación se realiza un estudio en diferentes situaciones: un día muy nublado, día lluvioso, día con niebla suave y día con niebla densa y lluvia.

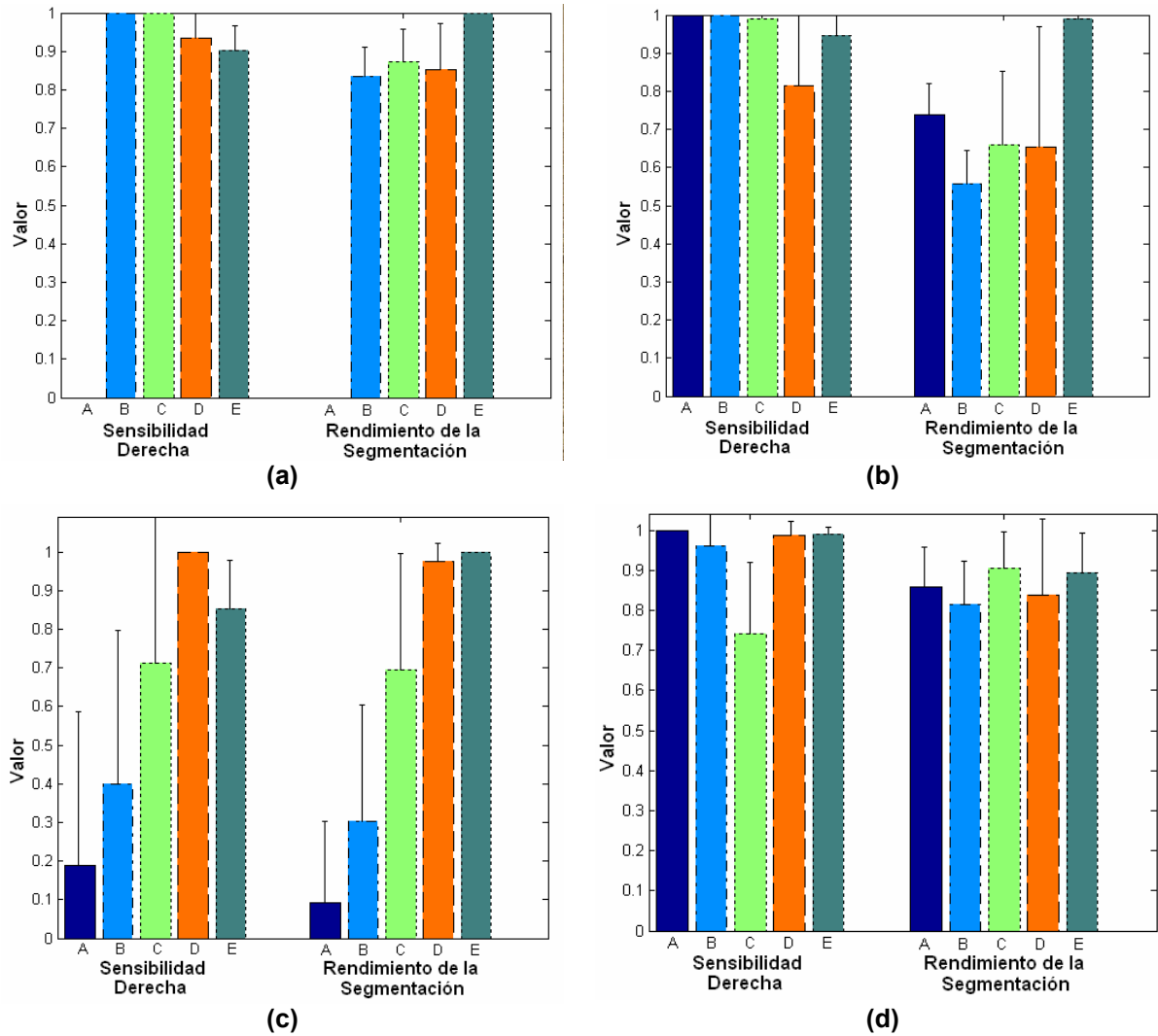
En la Figura 5.8 se puede ver algunas imágenes de las secuencias que vamos a estudiar.





**Figura 5.8.** Secuencias tomadas al bajo diferentes condiciones meteorológicas: **(a)** Niebla suave; **(b)** Niebla densa y lluvia; **(c)** Día muy nublado y **(d)** Lluvia.

La Figura 5.9 muestra los resultados de  $SE_D$  y  $R_S$  para las secuencias de la Figura 5.8.



**Figura 5.9.** Media y la Desviación Estándar de la Sensibilidad Derecha y Rendimiento de la Segmentación para los distintos tamaños del vehículo objetivo en las secuencias mostradas en la Figura 5.8: **(a)** Niebla suave; **(b)** Niebla densa y lluvia; **(c)** Día muy nublado y **(d)** Lluvia.

Como en la sección anterior, las condiciones meteorológicas adversas producen poco contraste en la imagen, el efecto es un filtro natural que limita el ruido de la imagen procedente del paisaje, y por tanto  $R_S$  es próximo a 1. Por otro lado, los faros encendidos (secuencias **a**, **b** y **d**) permiten un alto contraste local que genera altos niveles de  $SE_D$ .

En el caso de la secuencia **c**, los niveles de  $SE_D$  y  $R_S$  son similares a los de las secuencias en un día soleado utilizadas en la Sección 5.2.1 para el estudio del rango de velocidades.

La umbralización de los bordes permite obtener buenos resultados en cuanto a la Sensibilidad Derecha y el Rendimiento de la Segmentación en diversas situaciones con luminosidad muy variable.

## 5.2.4. Estudio respecto a la detección de distintos tipos de vehículos objetivo

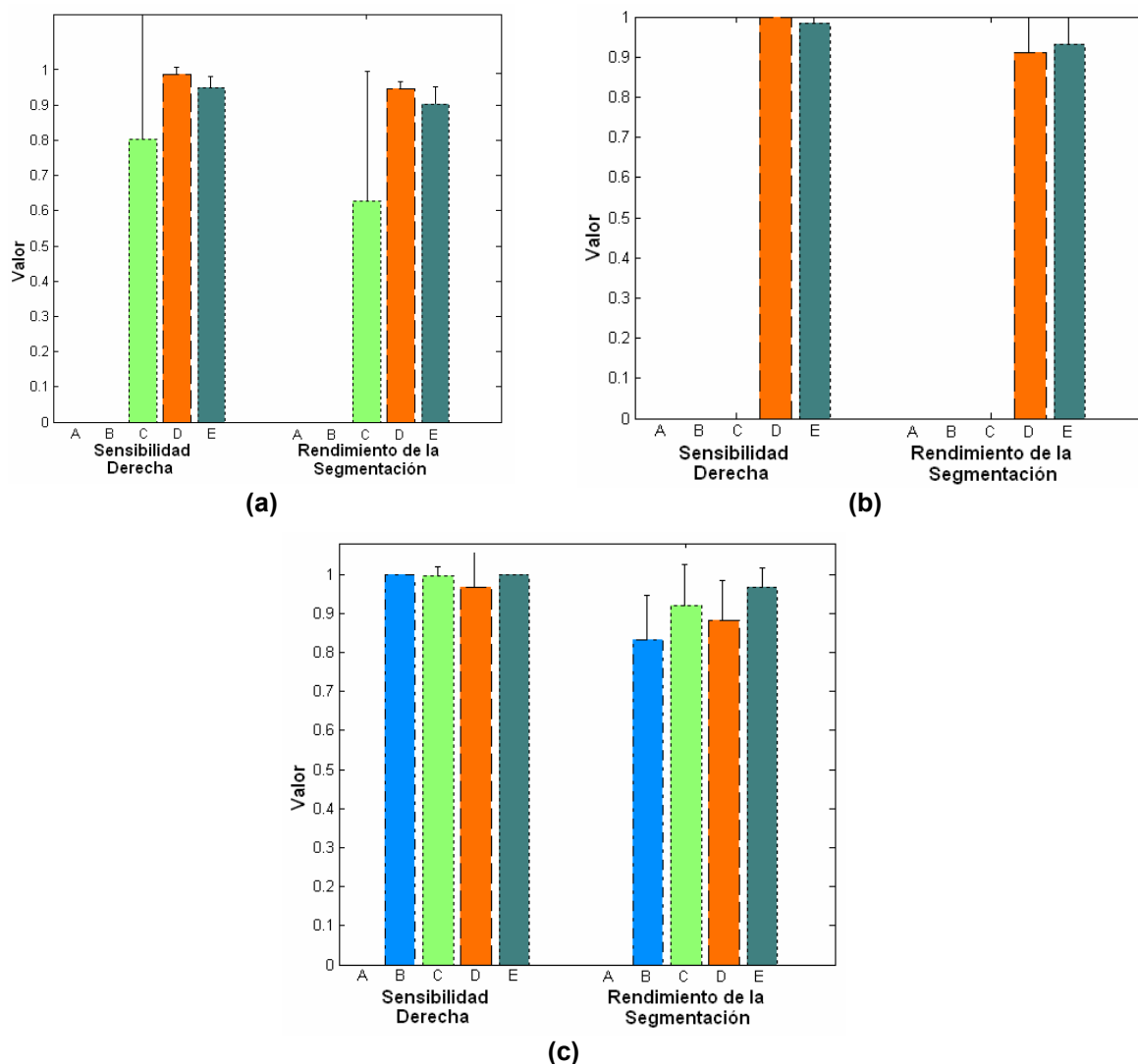
En esta sección vamos a comprobar que los tamaños del área de influencia escogidos son apropiados para todos los tipos de vehículo que podemos encontrar habitualmente por las carreteras.

Los casos que se podrían considerar altamente peligrosos son el de vehículos largos como camiones o vehículos que arrastran un remolque y las motos, que son los casos que se ilustran a continuación. En el caso de los primeros, el peligro sería sobre todo para el vehículo vigilante, mientras que en el caso de la moto la situación es peligrosa para la propia moto.



**Figura 5.10.** Secuencias con distinto tipo de vehículo objetivo: (a) Furgoneta con caravana; (b) Camión; (c) Moto.

La Figura 5.11 muestra los resultados de  $SE_D$  y  $R_S$  para las secuencias mostradas en la Figura 5.10.



**Figura 5.11.** Media y la Desviación Estándar de la Sensibilidad Derecha y Rendimiento de la Segmentación para los distintos tamaños del vehículo objetivo en las secuencias mostradas en la Figura 5.10: **(a)** Furgoneta con remolque; **(b)** Camión; **(c)** Moto. Por las características de los vehículos objetivos y la distancia en la que se producen algunos de los adelantamientos, en las figuras no existen datos para algunos de los tamaños del vehículo objetivo.

Como se puede observar en la Figura 5.11, aunque no hay datos para tamaños pequeños del vehículo objetivo (debido a las dimensiones de los vehículos y al momento en el que comienza la secuencia a grabarse), en los casos en los que si tenemos resultados estos son del mismo orden de magnitud que los de los vehículos objetivo de tipo coche. Por lo que podemos concluir que el tamaño de las áreas de influencia escogidas permite la segmentación de cualquier tipo de vehículo objetivo.

## **5.3. Sistema de alarma basado en el seguimiento del vehículo objetivo**

### **5.3.1. Algoritmo de seguimiento**

Se puede implementar un sistema de alerta basándonos en el seguimiento del vehículo objetivo segmentado mediante los resultados de detección de movimiento.

Los resultados mostrados en las Figuras 5.5, 5.7, 5.9 y 5.11 sugieren que podríamos usar un sistema de seguimiento sencillo basado en el cálculo de un centro de masas de todas las características con movimiento hacia la derecha en la imagen de salida del algoritmo (centro de masas global). En el caso ideal en el que sólo se detectan con movimiento hacia la derecha píxeles que pertenecen al vehículo el centro de masas calculado se situará en el centro del vehículo objetivo. Sin embargo, los resultados del seguimiento pueden verse alterados por patrones ruidosos, que como hemos visto antes existen en mayor o menor grado, y que pueden desplazar el centro de masas calculado del centro del vehículo objetivo.

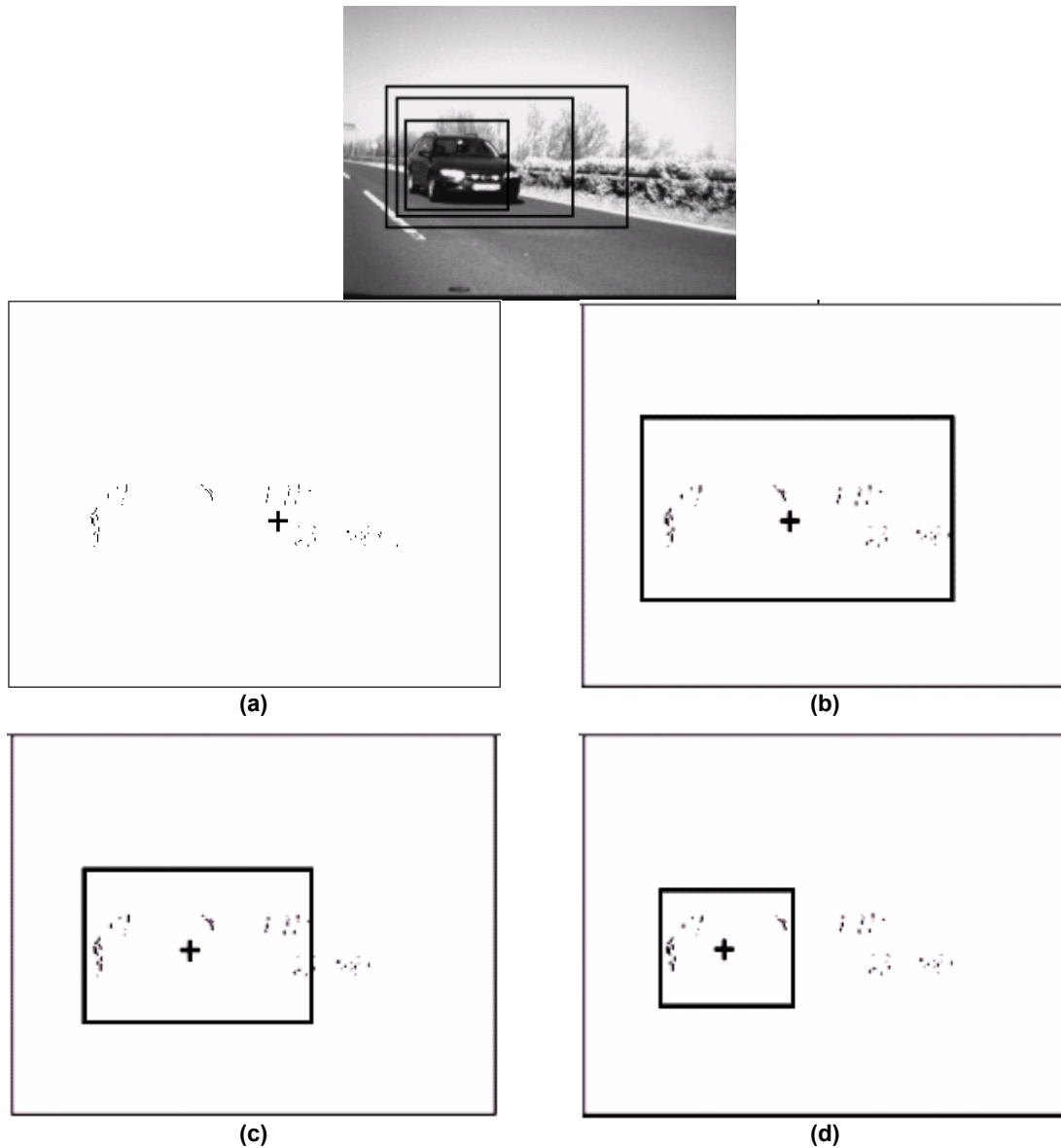
Por ello, proponemos un sistema de seguimiento sencillo basado en el cálculo de densidades de píxeles en movimiento. Esta idea se basa a su vez en la idea anterior de centro de masas global, pero en cierto modo, trata de resolver los problemas de desplazamiento que provoca el ruido existente respecto al movimiento computado en la imagen.

El sistema propuesto sigue los siguientes pasos:

1. Cálculo del centro de masas global, es decir, considerando todas las características con movimiento hacia la derecha que existan en la imagen.
2. Se re-calcula el centro de masas restringiendo el cálculo a un área alrededor del centro de masas global calculado en el paso anterior.
3. Volvemos a calcular un centro de masas pero restringiendo nuevamente el área con respecto a la utilizada en el paso 2. Este paso se repite tantas veces como se considere conveniente para un cálculo óptimo. El criterio para estimar el número de repeticiones puede ser, por ejemplo, la estabilidad en la posición del centro de masas.

Mediante este método se pretende eliminar del cálculo del nuevo centro de masas a aquellos píxeles que están lejos de la posición calculada en la etapa anterior, y por tanto contribuyen a desplazar el centro de masas de la posición ideal, es decir, el centro del vehículo objetivo. Esto tiene una importancia crítica ya que sólo un seguimiento progresivo y regular del vehículo objetivo puede conducir a una implementación sencilla del sistema de alarma.

La Figura 5.12 muestra un ejemplo del proceso seguido para el seguimiento a través de la secuencia. Tres ciclos parece ser suficientes para la mayoría de los casos estudiados.



**Figura 5.12.** Proceso iterativo realizado para la obtención de la posición del vehículo objetivo en cada imagen de la secuencia. En la parte superior de la figura se muestra la imagen real extraída de la secuencia; el resto de las imágenes muestran los píxeles que han sido detectados con movimiento hacia la derecha en la imagen superior. En todos los casos una cruz señala el punto de la imagen donde está situado el vehículo objetivo según el algoritmo de seguimiento. **(a)** Posición del vehículo objetivo mediante el cálculo del centro de masas global, es decir teniendo en cuenta todas las características de la imagen con movimiento hacia la derecha. **(b)** Posición del vehículo objetivo mediante el cálculo del nuevo centro de masas restringiendo el área de cálculo a la que se indica mediante el recuadro, es decir alrededor de la posición calculada mediante el cálculo del centro de masas global. **(c)** y **(d)** Las áreas de cálculo se reducen nuevamente (se indican mediante rectángulos) para la obtención de las nuevas posiciones. Sólo las características pertenecientes al coche se utilizarán en la estimación final del centro de masas (posición del vehículo objetivo).

### 5.3.2. Implementación hardware

El módulo para el cálculo del centro de masas está compuesto por dos sumadores, un contador y un circuito *core* optimizado para realizar la división. Cuando entra un píxel con movimiento hacia la derecha en el módulo de cálculo del centro de masas, se estima si pertenece al área en la que se está computando el centro de masas. En caso positivo, se incrementa el contador y se suman su fila y su columna en los sumadores correspondientes.

Una vez que se han considerado todos los píxeles del área se realiza la división mediante el *core*, que realiza el cálculo en un solo ciclo de reloj.

El proceso tiene lugar en un bucle con tantas iteraciones como sub-áreas de la imagen estemos considerando para refinar la posición del centro de masas. En realidad, es posible obtener una posición con una exactitud bastante aceptable usando tan solo tres iteraciones, es decir, considerando tres sub-áreas para el cálculo del centro de masas basado en densidades.

El módulo para el cálculo del centro de masa realiza el procesamiento en paralelo con el resto de módulos, y su arquitectura tiene un camino de datos microsegmentado, por lo que se integra muy bien dentro de la arquitectura de detección de movimiento planteada.

Una vez obtenidos los valores finales de la posición estimada para el vehículo objetivo, el resultado se muestra por pantalla superpuesto, o bien a la imagen original, o bien a la imagen de salida del algoritmo. El sistema de seguimiento se integra perfectamente en la arquitectura.

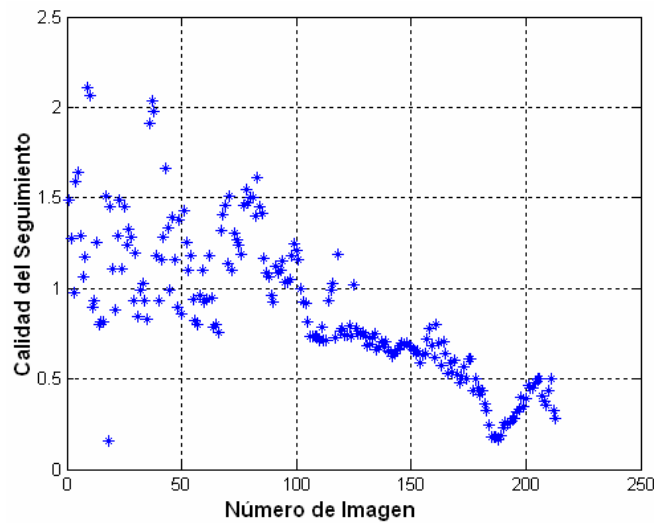
**Tabla 5.1.** Recursos de hardware consumidos por el sistema de seguimiento cuando se implementa en el chip Virtex-II XC2V3000.

	Nº de Slices (% recursos)	Nº de blockram (% recursos)
<b>Sistema de seguimiento</b>	1,780 (6)	0 (0)

### 5.3.3. Resultados

Para evaluar la exactitud del seguimiento vamos a obtener la *Calidad del Seguimiento* ( $C_s$ ), que definiremos como la distancia entre el centro de masas, obtenido de la forma expuesta anteriormente, y el centro del rectángulo que hemos dibujado alrededor del vehículo objetivo. Esta distancia estará normalizada por el radio del máximo círculo inscrito en el rectángulo. Si el centro de masas cae dentro de dicho círculo  $C_s$  estará por debajo de uno, y estaremos realizando un buen seguimiento del vehículo objetivo. En otro caso, obtendremos valores por encima de uno, con lo que el ruido será predominante sobre las detecciones de velocidad correctas, con lo que el centro de masas estará desplazado y no realizaremos un seguimiento correcto del vehículo objetivo. Estos errores en el seguimiento ocurren cuando el vehículo objetivo es muy pequeño (está muy lejos). Sin embargo, a distancias medias, cuando el vehículo es suficientemente grande, el seguimiento se realiza correctamente. De hecho, las detecciones correctas ocurren cuando el vehículo objetivo empieza a convertirse en un peligro potencial.

En la Figura 5.13 se muestran los resultados del seguimiento realizado sobre un vehículo objetivo que realiza una maniobra de adelantamiento a 110 Km/h, mientras que el vehículo vigilante se desplaza a 90 Km/h. La Figura 5.13 (a) muestra la evolución de  $C_S$  cuando se utiliza únicamente el cálculo del centro de masas global como algoritmo de seguimiento, mientras que la Figura 5.13 (b) muestra la evolución de  $C_S$ , pero usando el algoritmo de seguimiento basado en el cálculo de densidades que se ha descrito antes. Como puede observarse en los resultados de la Figura 5.13 el algoritmo de seguimiento basado en densidades, aunque es muy sencillo, nos permite realizar un seguimiento del vehículo objetivo que podría utilizarse como para disparar una alerta local en el vehículo vigilante en caso de peligro.



**Figura 5.13.** Resultados de  $C_S$  para la secuencia mostrada en la Figura 5.14 cuando se utiliza el cálculo del centro de masas basado en densidades. El vehículo objetivo se sigue perfectamente a partir de la imagen 125.

En la Figura 5.14 se muestran los resultados del seguimiento sobre la secuencia real. La posición del vehículo objetivo calculada con los datos de la detección de movimiento se señala mediante una cruz sobre la imagen.

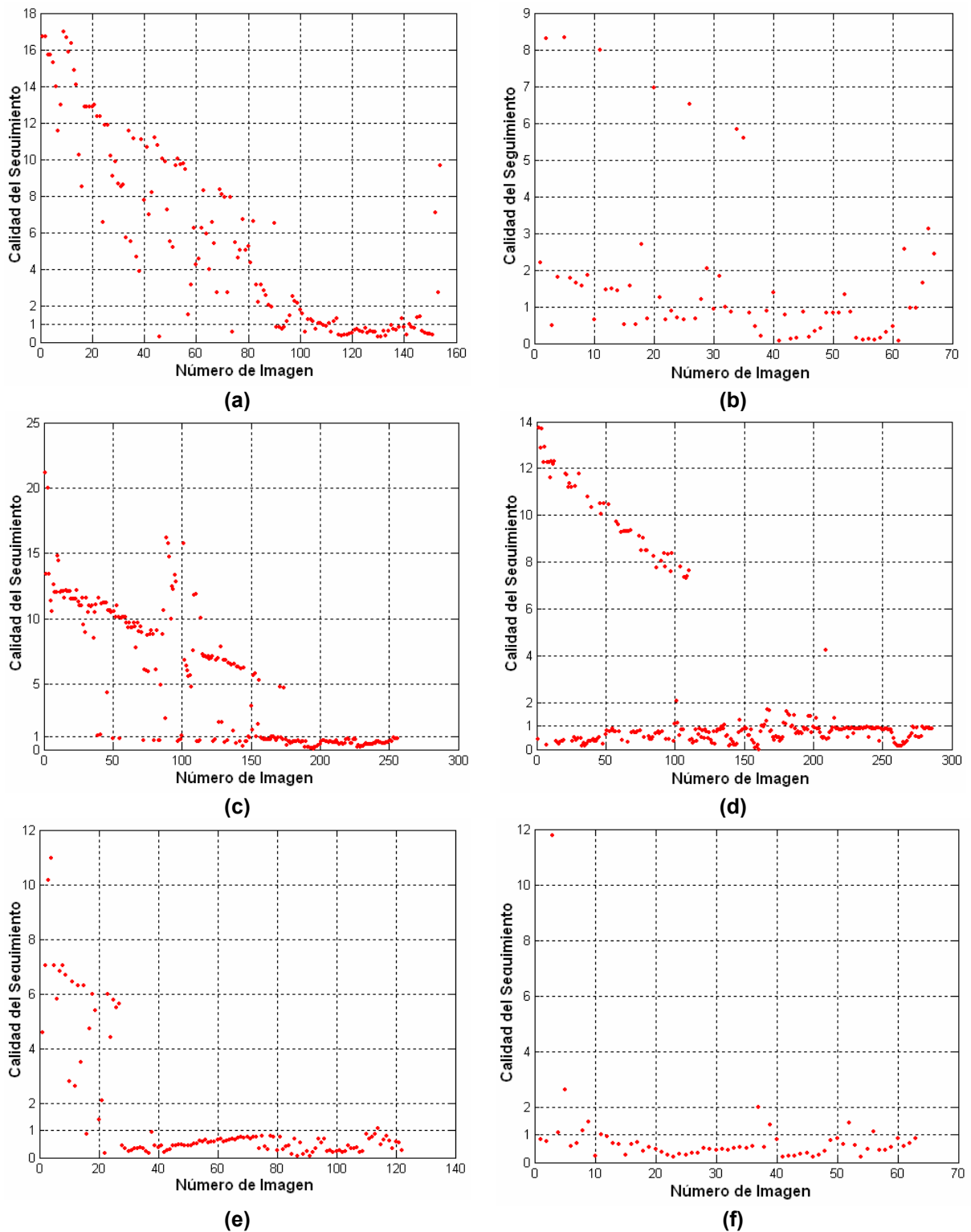


Figura 5.14. Resultados del seguimiento sobre la secuencia real.

En la Figura 5.15 se muestra la evolución de  $C_S$  para algunas de las secuencias estudiadas anteriormente. Se observa en la gráfica de  $C_S$  que existe una convergencia a cero, en otras palabras,



a partir de un determinado momento durante la maniobra de adelantamiento el algoritmo es capaz de detectar y seguir al vehículo objetivo sin errores, de forma que sería posible diseñar un sistema de alerta de presencia de un vehículo en el ángulo muerto del vehículo vigilante.



**Figura 5.15.** Resultados de la Calidad del Seguimiento para: (a) Velocidad entre 5-10 m/s; (b) Anochecer; (c) Día nublado; (d) Mucha niebla y lluvia; (e) Remolque; (f) Moto.

El objetivo fundamental del algoritmo de seguimiento descrito en esta sección es tan sólo mostrar un ejemplo de una potencial aplicación que tienen los resultados de movimiento extraídos por

el algoritmo de detección para aplicaciones de la vida real en las que la detección de movimiento en tiempo real es una capacidad importante. Los resultados mostrados son fácilmente mejorables si los datos de movimiento son post-procesados por alguno de los algoritmos de seguimiento más complejos que existen en la bibliografía, algunos de ellos con cualidades predictivas como los filtros de Kalman [KAL60, KAL61].

## 5.4. Mejoras en la detección de movimiento: reducción del efecto de la perspectiva

En cualquier sistema de visión es necesario proyectar la realidad tridimensional en un sensor óptico artificial o en la retina (ambos sistemas bidimensionales) antes de la extracción y análisis de los datos. Debido a esta proyección se produce una distorsión de la escena por la perspectiva que afecta a la velocidad de los objetos en movimiento, y se aprecia especialmente cuando la trayectoria del movimiento no es perpendicular a la cámara.

Como se comentó anteriormente los efectos de la perspectiva sobre la detección de movimiento son:

- el vehículo objetivo aumenta de tamaño a medida que se aproxima al vehículo vigilante (existirá un movimiento de expansión adicional);
- el vehículo objetivo parece acelerado por efecto de la perspectiva;
- las partes delantera y trasera de un mismo vehículo se detectarán moviéndose a distinta velocidad.

Para finalizar este capítulo se propondrán dos aproximaciones distintas que suponen la reducción del efecto que produce la perspectiva.

### 5.4.1. Métodos de reducción de la deformación: Canales de Velocidad frente a Remapeo Espacial de la imagen

Como hemos visto antes, uno de los principales problemas que introduce la deformación de la perspectiva es que dos partes diferentes del mismo vehículo objetivo (la parte frontal y trasera) aparentan moverse a velocidades distintas, con lo que se hace difícil aplicar el criterio de sólido rígido para filtrar la salida de la detección de movimiento del algoritmo.

Para reducir este efecto introducimos el concepto de *Canal de Velocidad (CV)*. Un Canal de Velocidad acumula las características cuyas velocidades se detectan dentro de un rango de velocidades próximas ( $V \pm \Delta V$  donde  $\Delta V$  representa pequeñas variaciones del módulo de la velocidad). De esta forma, las diversas velocidades con las que se mueven las partes de un sólido rígido quedarán acumuladas por un CV.

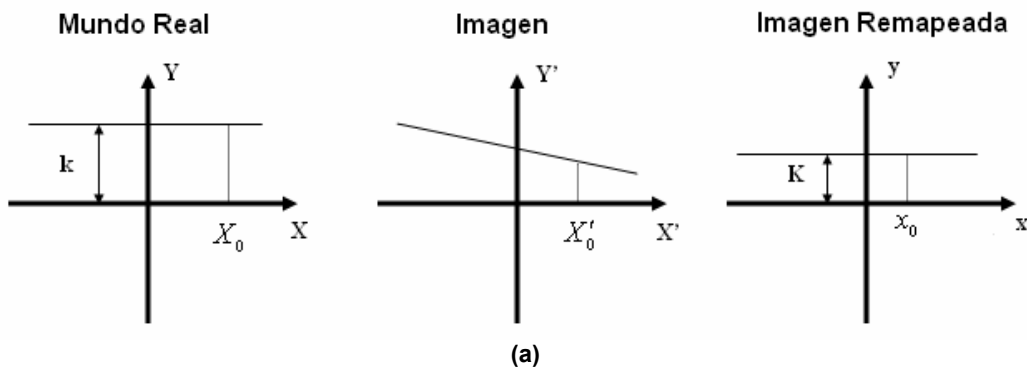
Definiremos tantos Canales de Velocidad como rangos distintos de velocidades consideremos que necesitan agruparse en la imagen. Y de nuevo, adoptamos una estructura competitiva entre los Canales de Velocidad (*winner-takes-all*), de manera que estos canales de velocidad funcionan como filtros paso-banda que sólo dejan pasar al mapa de saliencia aquellos píxeles que pertenecen al CV que acumula un máximo local de velocidades. El máximo se corresponderá con la presencia del sólido rígido.

Por tanto, los CV introducen coherencia espacial sobre la velocidad de las características en movimiento. Se usan Canales de Velocidad en lugar de los acumuladores de velocidades independientes (o neuronas colectoras) que usaba el algoritmo de detección de movimiento sin corrección de la perspectiva.

Esta primera solución a los problemas provocados por la perspectiva resuelve el relacionado con apreciar qué píxeles pertenecen a un sólido rígido, sin embargo no resuelven el problema de la aceleración ni del aumento del tamaño del vehículo objetivo con su aproximación.

El segundo método que introduciremos para paliar los problemas ocasionados por la proyección en 2D es el *Remapeo Espacial de la Imagen (REI)*. El Remapeo Espacial de la Imagen es una transformación afín de las coordenadas que trata de invertir el proceso de proyección de una escena en 3D a una superficie en 2D [MAL91, TAN04]. Es posible compensar el efecto de la perspectiva transformando la imagen de forma que las líneas paralelas y las distancias semejantes en la escena real sean remapeadas en líneas paralelas y distancias semejantes en la imagen procesada. Esencialmente, en nuestra aplicación eso significa expandir el lado izquierdo de la imagen (correspondiente a la parte de la escena más alejada de la cámara) y contraer el lado derecho de la imagen (correspondiente a la parte de la escena más cercana a la cámara). El método de interpolación utilizado es el desarrollo de Taylor truncado, conocido como *Local Jet* [TAN04].

La Figura 5.16 muestra un ejemplo de una imagen remapeada y las expresiones que se usan para realizar dicha transformación.



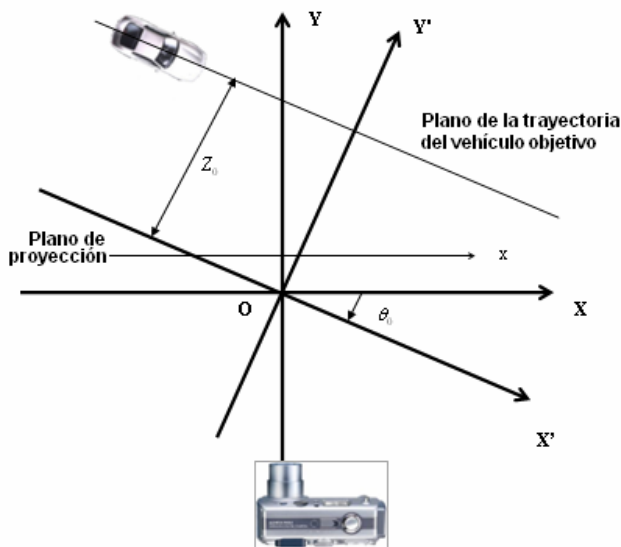


(b)

**Figura 5.16.** (a) Transformación de las coordenadas; (b) Resultado del remapeo sobre diferentes imágenes de una de las secuencias de adelantamiento.

La Figura 5.17 muestra la transformación que debe realizarse sobre las coordenadas de los distintos píxeles de la imagen para producir el remapeo de la imagen.

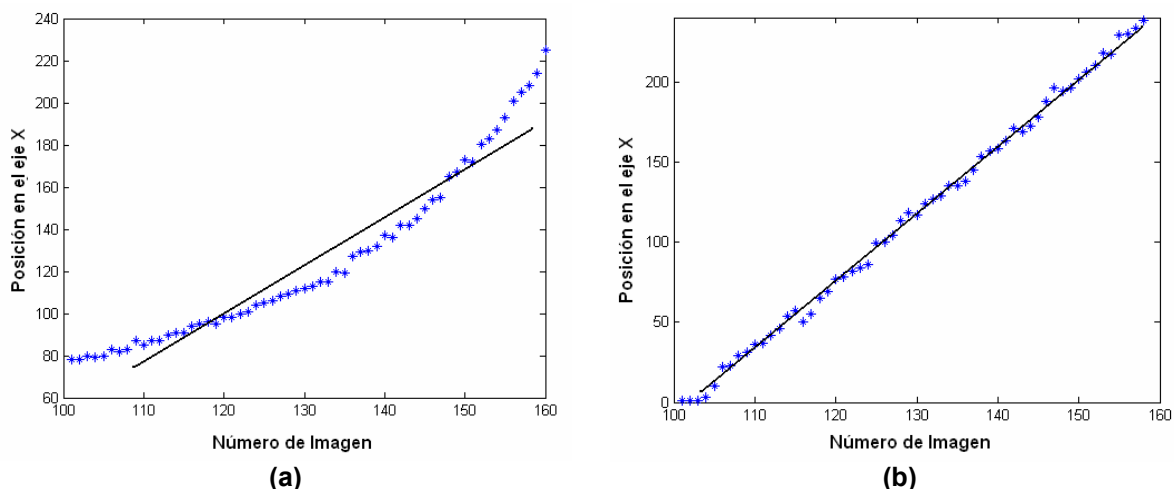
En la secuencia remapeada la velocidad media del vehículo objetivo es más constante a lo largo de la secuencia y cada punto que pertenece al sólido rígido se mueve aproximadamente a la misma velocidad. La Figura 5.18 muestra el efecto que tiene sobre la velocidad del centro de masas del vehículo objetivo el remapeo de la imagen previo a la detección del movimiento. Se observa en la Figura 5.18a que aunque la velocidad de la maniobra de adelantamiento es constante la curva está deformada (la velocidad constante se representaría por una línea) debido a la perspectiva. Por otro lado, la Figura 5.18b muestra la misma representación, pero esta vez sobre la imagen remapeada. En este caso la velocidad del vehículo objetivo es constante, es decir, se aproxima bastante bien por una línea con una pendiente que dependerá de la velocidad que se codifique.



$$x = c_0 \frac{X' - Z_0 \tan \theta_0}{X' \tan \theta_0 + Z_0}$$

$$y = \frac{c_0}{\cos \theta_0} \times \frac{Y'}{X' \tan \theta_0 + Z_0}$$

**Figura 5.17.** Transformación sobre las coordenadas de los píxeles de la imagen. Las coordenadas del píxel en la imagen de la cámara ( $X'$ ,  $Y'$ ) se transforman en ( $x$ ,  $y$ ) en la imagen remapeada.

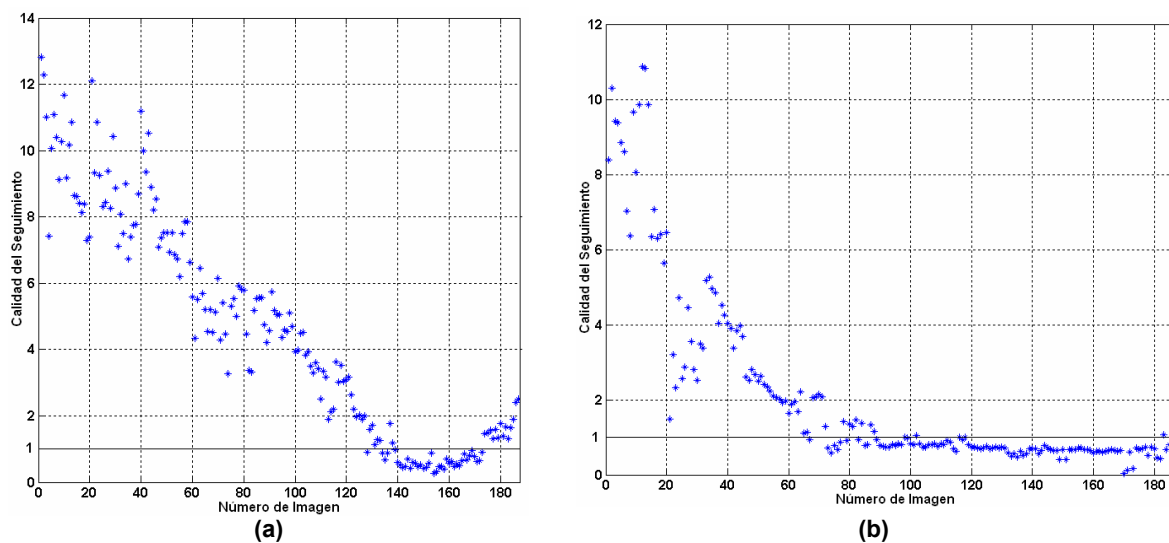


**Figura 5.18.** Transformación de la velocidad por la perspectiva para la secuencia mostrada en la Figura 5.16: **(a)** Posición en el eje X del centro de masas del vehículo objetivo a lo largo de una secuencia antes de realizar el remapeo; **(b)** Posición en el eje X del centro de masas del vehículo objetivo a lo largo de una secuencia después de realizar el remapeo en la imagen.

La principal diferencia entre las dos estrategias de corrección de los efectos de la perspectiva es que con CV necesitamos acumular características con velocidades diferentes para sincronizar las partes frontal y trasera del vehículo objetivo, mientras que con REI esto no es necesario ya que el efecto es compensado mediante el remapeo de la imagen y se resuelve el problema de la aceleración aparente de los vehículos.





A continuación vamos a comparar ambos métodos entre si. Para ello vamos a usar el parámetro Calidad del Seguimiento ( $C_s$ ) que definimos anteriormente.

La Tabla 5.2 muestra los resultados de la comparativa sobre cuatro secuencias distintas.



**Figura 5.19.** Calidad del Seguimiento para la secuencia mostrada en la Figura 5.16 usando: **(a)** Canales de velocidad; **(b)** Remapeo Espacial de la imagen.

**Tabla 5.2.** Resultados comparados de la aplicación de ambos métodos en cuatro secuencias distintas

Imagen de la secuencia	Método	1ª. Imagen a partir de la que se hace un buen seguimiento	Tamaño del vehículo objetivo (píxeles)	Número de características detectadas	Mejor Resultado
	CV	139	10660	83	
	REI	89	3216	15	√
	CV	1	899	8	√
	REI	1	899	10	√
	CV	1	1813	22	√
	REI	1	1813	10	√
	CV	36	32469	30	
	REI	23	14385	17	√

### 5.4.2. Implementación en hardware específico

Como pudimos comprobar en el Capítulo 3, la arquitectura diseñada para el circuito de detección de movimiento tiene un camino de datos microsegmentado que procesa un píxel por cada ciclo de reloj. Por tanto, cualquier nuevo módulo que queramos integrar en el sistema deberá diseñarse cuidadosamente de manera que no degrade el rendimiento del sistema total, es decir, deberá procesar un píxel por cada ciclo de reloj.

Introducir los canales de velocidad en el cauce de datos microsegmentado es inmediato ampliando el rango de datos que actualiza los contadores que controlan la cantidad de características que se mueven en cada área de influencia en el módulo de selección de características dominantes. De esta forma no sólo se consumen menos recursos, es necesario un número menor de contadores en cada área de influencia, sino que además se necesitan menos ciclos de reloj para obtener el máximo que determina las características dominantes en el área de influencia.

En el caso del remapeo de la imagen es necesario introducir un nuevo módulo que trabaje en paralelo con los demás módulos. La Figura 5.20 muestra cómo se integra en el esquema de procesamiento global,

Como se ha mostrado en la Figura 5.17, las coordenadas del espacio deformado por la cámara ( $X'$ ,  $Y'$ ), es decir las coordenadas de los píxeles de la imagen, se transforman mediante

productos, divisiones, sumas, diferencias y funciones coseno y tangente, en las coordenadas del espacio remapeado (x, y).

El módulo de Remapeo Espacial de la Imagen utiliza algunas unidades superescalares para realizar su función. Para calcular el coseno y la tangente se han utilizado tablas de consulta. Sin embargo, para la división se utiliza un circuito optimizado (*core*) que permite realizar una operación en cada ciclo de reloj. Todo el sistema ha sido probado a 31.5 MHz.

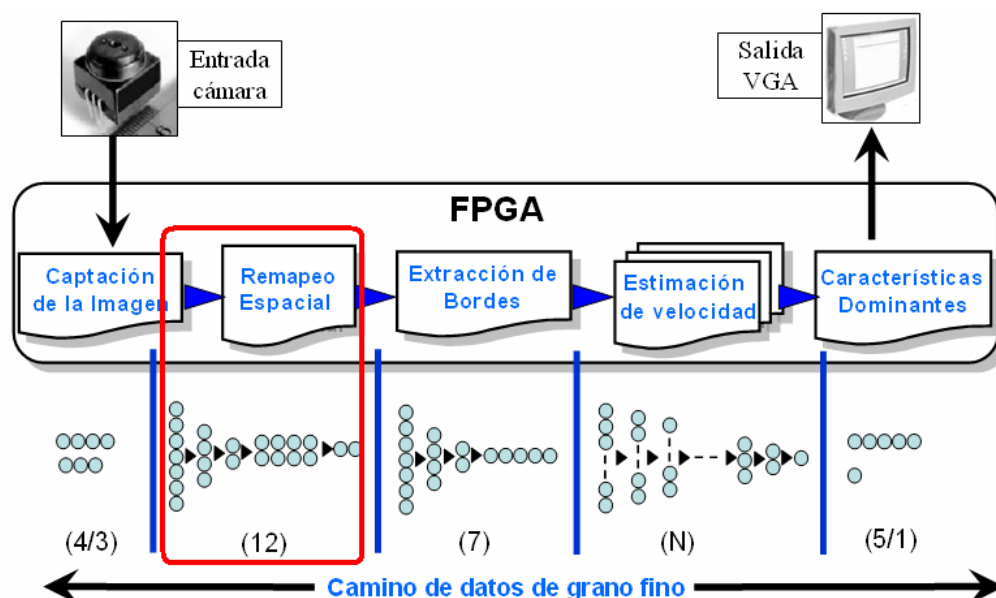


Figura 5.20. Camino de datos para el sistema completo que incluye la etapa de remapeo espacial de la imagen.

Tabla 5.3. Recursos de hardware consumidos por el módulo de Remapeo Espacial de la imagen cuando el diseño se implementa en un chip Virtex-II XC2V1000.

	Nº de Slices (% recursos)	Nº de blockram (% recursos)
Remapeo Espacial	838 (16)	0 (0)

## 5.5. Conclusiones

- ☑ En este capítulo se ha descrito una aplicación que requiere de tiempo real y que hace uso de las primitivas de movimiento extraídas mediante el algoritmo de detección.
- ☑ La aplicación realiza un seguimiento de los vehículos que nos adelantan desde la perspectiva del espejo retrovisor.
- ☑ Se han estudiado los resultados para diferentes situaciones meteorológicas y de iluminación.
- ☑ Se ha estudiado el seguimiento realizado a diferentes tipos de vehículos.

- ☑ Se ha propuesto dos soluciones diferentes para paliar la deformación debida a la perspectiva en la imagen y se han comparado entre si, siendo el Remapeo Espacial de la imagen el que mejores resultados obtiene.
- ☑ Los resultados obtenidos son satisfactorios. Por lo tanto, un sistema de alarma para la detección de colisiones laterales basado en detección de primitivas de movimiento es claramente viable.





---

## **CAPÍTULO 6:**

# **Conclusiones**

---

Este capítulo es un resumen del trabajo presentado en esta memoria tesis. En él se discuten los principales resultados obtenidos, la contribución de este trabajo, así como la innovación que representa. También se enumeran las contribuciones científicas que ha generado este trabajo.

## 6.1. Discusión

El trabajo descrito se ha desarrollado en el marco de los proyectos europeos de investigación ECOVISION y DRIVSCO, y el proyecto nacional DEPROVI. El objetivo del proyecto ECOVISION era diseñar una nueva generación de sistemas de visión artificiales bio-inspirados, que sea flexible y auto-adaptable, y que opere en entornos reales y en tiempo real, dedicando hardware digital específico al procesamiento de aquellas etapas que requieren de mayor tiempo de computación. Por su parte, el proyecto DRIVSCO tratará de diseñar, probar e implementar una metodología de combinación de mecanismos de aprendizaje con el control convencional, obteniendo un sistema que actúe de forma proactiva usando diferentes mecanismos predictivos, para ello uno de los objetivos es la creación de sensores visuales adaptativos en hardware. El proyecto DEPROVI se relaciona con el diseño de sistemas empotrados para procesamiento de visión en tiempo real, y los aplicará en medicina, vehículos y robots.

Los objetivos de este trabajo implican:

- Creación de un algoritmo bio-inspirado de bajo nivel para la detección de movimiento;
- Sistema de bajo coste computacional con un alto rendimiento en la detección;
- Implementación del sistema en hardware de propósito específico para obtener el procesamiento en tiempo real;
- Aplicación del sistema a situaciones reales donde el tiempo real sea un factor clave, y se realizará el estudio de viabilidad en dichas situaciones.

Para ello, el trabajo se ha estructurado en las siguientes etapas:

1. Revisión bibliográfica sobre modelos de visión de insectos.
2. Elaboración de un algoritmo de detección de movimiento.
3. Estudio de viabilidad de implementación hardware en dispositivos de tipo FPGA.
4. Diseño de una arquitectura de procesamiento eficiente.
  - a. Estudio de diversas alternativas.
  - b. Evaluación de la complejidad de computación y costes de implementación.
5. Evaluación de prestaciones. Velocidad y precisión de resultados.
6. Aplicación en sistemas de ayuda a la conducción. Sistema de monitorización automático de adelantamientos:
  - a. Implementación del sistema de procesamiento embebido.
  - b. Evaluación de la precisión utilizando secuencias de adelantamiento reales.
7. Aplicación en sistemas de ayuda a pacientes con baja visión.

Se ha utilizado como modelo biológico el esquema de detección de movimiento de los insectos, para lo que se ha realizado un estudio bibliográfico de los conocimientos que la neurofisiología tiene del comportamiento del sistema visual de los mismos (véase el Capítulo 1).

El algoritmo de detección de movimiento se ha estructurado en distintos módulos, cada uno de los cuales tiene una funcionalidad determinada siguiendo una estructura bio-inspirada de procesamiento en columnas (véase el Capítulo 2). Los módulos implementados son:

- Extracción de características espaciales de carácter local (bordes) que definen la escena. Este módulo convierte al sistema en un detector de movimiento no denso, por lo que el coste computacional será mucho menor que otros sistemas de detección de movimiento densos. Se han estudiado dos aproximaciones para la extracción de la estructura de la escena.
- Detección de movimiento, basado en conjuntos de correladores de Reichardt, que es el modelo matemático que describe el comportamiento neurofisiológico de las neuronas de la mosca, y donde el conjunto de correladores empleado es función de la aplicación.
- Filtrado espacio-temporal que se basa en estructuras competitivas de tipo winner-takes-all, y que integra información del movimiento mediante criterios de coherencia espacio-temporal, permitiendo una buena segmentación de los objetos móviles sólo con información de velocidad.

Puesto que uno de los objetivos es la aplicación en situaciones reales del algoritmo de detección de movimiento, este ha sido evaluado también con secuencias reales (con un grado de dificultad creciente, grabadas con cámaras estáticas y en movimiento), con lo que se ha podido comprobar que los resultados obtenidos tienen un alto rendimiento y eficiencia, aún cuando estos resultados se alcanzan con un número muy bajo de píxeles en comparación con el tamaño inicial de la imagen. El algoritmo demuestra también ser robusto frente a las vibraciones de la cámara.

Dada la dificultad de evaluar cualquier algoritmo con secuencias reales, ya que éstas no se encuentran etiquetadas (es decir, no se conoce previamente el movimiento de cada uno de los píxeles que componen la secuencia), se ha confeccionado una metodología de evaluación con secuencias reales que ha sido descrita en el Capítulo 4. Para ello se han definido diferentes parámetros que dan la medida de las características de detección en la imagen. Ha sido necesario también etiquetar manualmente las distintas secuencias utilizadas en la evaluación.

Para la implementación en tiempo real se han utilizado dispositivos FPGA en los que para aprovechar sus recursos se ha implementado una segmentación de cauce en una arquitectura específica, y altamente optimizada (véase el Capítulo 3). Se han estudiado

distintas aproximaciones de la arquitectura. También se han evaluado los recursos consumidos para cada una de las arquitecturas propuestas.

La arquitectura modular implementada utiliza un flujo de datos regular y, aún cuando el reloj del sistema es mucho más lento que los relojes de los procesadores actuales, el uso intensivo de los recursos paralelos de la FPGA, con un camino de datos finamente segmentado en cada uno de los módulos de procesamiento, y el uso de canales específicos de comunicación entre módulos, permite el procesamiento de un píxel por cada ciclo de reloj, lo que supone el procesamiento de hasta 41472 Kpps, convirtiendo el sistema en muy eficiente y de altas prestaciones.

El cuello de botella de cualquiera de las arquitecturas lo constituye el necesario acceso a la memoria externa, por lo que ha sido necesaria la definición de un módulo específico que permita los máximos accesos a la misma con las mínimas restricciones.

Se han definido dos aplicaciones reales donde la necesidad de un procesamiento en tiempo real justifica el empleo de un sistema de detección de movimiento como el mostrado en este trabajo.

La primera aplicación (véase el Capítulo 5) consiste en un sistema basado en visión para la monitorización de adelantamientos, el objetivo es evitar posibles colisiones laterales. El sistema debe detectar especialmente la presencia de otro vehículo en el ángulo muerto existente en el espejo retrovisor. Esta aplicación se engloba en el marco del proyecto europeo ECOVISION. De hecho, esta tesis ha tenido su origen en el marco de dicho proyecto europeo. La empresa automovilística Hella, que participa como miembro del consorcio ECOVISION, ha extraído secuencias reales de adelantamiento mediante una cámara situada sobre el espejo retrovisor del conductor de un coche instrumentado, que han sido utilizadas para la evaluación de esta primera aplicación.

Ha sido necesario introducir dos nuevos módulos de procesamiento para resolver la aplicación, el primero para corregir la deformación de la perspectiva existente que afecta a la velocidad; y el segundo para realizar un seguimiento de los vehículos segmentados mediante criterios de velocidad.

Los resultados obtenidos permitirían que un sistema de alarma avisase al conductor de la presencia de un vehículo antes de que éste se posicione en el ángulo muerto del retrovisor, avisando en situaciones de colisión lateral.

Se ha realizado una evaluación de la eficiencia del sistema para la aplicación, encontrándose que funciona para un amplio rango de velocidades relativas entre los vehículos, para distintas condiciones luminosas y meteorológicas, y que la detección es independiente del tipo de vehículo que realiza la maniobra de adelantamiento.

La segunda aplicación (véase el Apéndice) nace de la colaboración con la Universidad de Murcia, donde se realizaba un proyecto para el desarrollo de ayudas optoelectrónicas para la rehabilitación visual de la restricción severa del campo visual periférico, la implementación de

prototipos y la evaluación con pacientes. Los resultados preliminares de evaluación del sistema estático son prometedores. Es de esperar que el sistema dinámico basado en la superposición de la información del movimiento de la escena sobre el área de visión residual del paciente también tenga buenos resultados, pero no ha sido evaluado con pacientes.

Los aspectos innovadores más relevantes de este trabajo son los siguientes:

- Desarrollo de circuitos de propósito específico para su integración en distintas plataformas.
- Capacidad de análisis de imágenes a bajo nivel y en tiempo real. Los procesamientos que hemos descrito están lejos de ser realizados en tiempo real en un procesador de propósito específico.
- Adaptación sencilla de los distintos circuitos a aplicaciones concretas.
- Reconfiguración y adaptación de plataformas de computación. Estos circuitos se pueden integrar juntos en un mismo dispositivo. Además se han evaluado las capacidades de estos circuitos. El hecho de que se base en un dispositivo reconfigurable hace sencilla la adaptación a distintos campos de aplicación introduciendo módulos adicionales de procesamiento cuando sea necesario.

## 6.2. Resultados publicados

Los resultados de trabajo se han publicado en los siguientes artículos:

### 6.2.1. Publicaciones internacionales con índice de impacto (SCI)

1. J. Díaz, E. Ros, S. Mota and R. Rodriguez-Gomez, "FPGA based architecture for motion sequence extraction", *International Journal of Electronics*, 2006 (en prensa).
2. J. Díaz, E. Ros, F. Pelayo, E. M. Ortigosa and S. Mota, "FPGA based real-time optical-flow system," *IEEE Transactions on Circuits and Systems for Video Technology*, 16 (2), pp. 274-279, 2006
3. J. Díaz, E. Ros, S. Mota, F. Pelayo, and E. M. Ortigosa, "Sub-pixel motion computing architecture," *IEE Proc. Vision, Image & Signal Processing*, 15(6), pp. 869-880, 2006.
4. E. M. Ortigosa, A. Cañas E. Ros, P. M. Ortigosa, S. Mota, J. Díaz, "Hardware description of multi-layer perceptrons with 3 different abstraction levels," *Microprocessors and Microsystems*, 30(7), pp. 435-444, 2006.
5. S. Mota, E. Ros, J. Díaz, E. M. Ortigosa, and A. Prieto, "Motion-driven segmentation by competitive neural processing ", *Neural Processing Letters*, 22(2), pp. 125-147, 2005.
6. S. Mota, E. Ros, E. M. Ortigosa, and F. Pelayo, "Bio-inspired motion detection for blind spot overtaking monitor," *International Journal of Robotic and Automation*, 19(4), pp. 190-196, 2004.
7. F. Vargas-Martin, M. D. Peláez-Coca, E. Ros, Díaz J. and S. Mota, "Optoelectronic Visual Aid Based on Reconfigurable Logic for Severe Peripheral Vision Loss Rehabilitation", *Ophthalmic Research*, vol. 36, no. 1, pp. 60, 2004

### 6.2.2. Publicaciones en Lecture Notes in Computer Science (también con Índice de Impacto SCI)

8. S. Mota, E. Ros, J. Díaz, R. Rodríguez and R. Carrillo, "A space variant mapping architecture for reliable car segmentation", *Lecture Notes in Computer Science*, Springer-Verlag, 2007, (en prensa).
9. E. Ortigosa, A. Cañas, R. Rodriguez, J. Diaz, S. Mota, "Towards an optimal implementation of MLP in FPGA," *Lecture Notes in Computer Science*, Springer-Verlag, 3985, pp. 46-51, 2006.
10. J. Díaz, E. Ros, S. Mota and R. Rodriguez-Gomez, "Highly paralellized architecture for image motion estimation, " *Lecture Notes in Computer Science*, Springer-Verlag, 3985, pp. 75-86, 2006.
11. E. Ros, J. Díaz, S. Mota, F. Vargas-Martín and M.D. Peláez-Coca, "Real time image processing on a portable aid device for low vision patients," *Lecture Notes in Computer Science*, Springer-Verlag, 3985, pp. 158-163, 2006.
12. S. Mota, E. Ros, J. Diaz, and F. de Toro, "General purpose real-time image segmentation system, "*Lecture Notes in Computer Science*, Springer-Verlag, 3985, pp. 164-169, 2006.
13. J. Díaz, E. Ros, S. Mota and R. Agis, "Real-time embedded system for rear-view mirror overtaking car monitoring," *Lecture Notes in Computer Science*, Springer-Verlag, 4017, pp. 385-394, 2006.
14. S. Mota, E. Ros, J. Díaz, R. Agis and F. de Toro, "Bio-inspired motion-based object segmentation," *Lecture Notes in Computer Science*, Springer-Verlag, 4141, pp. 196-205, 2006.
15. J. Díaz, E. Ros, S. Mota, R. Carrillo, R. Agís, "Real time optical flow processing system," *Lecture Notes in Computer Science*, 3203, pp.617-626, 2004.
16. S. Mota, E. Ros, J. Díaz, E. M. Ortigosa, R. Agís and R. Carrillo R, "Real-time visual motion detection of overtaking cars for driving assistance using FPGAs," *Lecture Notes in Computer Science* vol. 3203, pp. 1158-1161, 2004.
17. J. Díaz, S. Mota, E. Ros and Guillermo Botella, "Neural Competitive Structures for Segmentation Based on Motion Features," *Lecture Notes in Computer Science*, vol. 2686, pp. 710-717, 2003.

### 6.2.3. Conferencias internacionales

18. F. Vargas-Martín, M. D. Peláez-Coca, E. Ros, S. Mota and J. Díaz, "A generic real-time video processing unit for low vision," presentado en Vision 2005, London, England, Abril 2005
19. J. Díaz, E. Ros, S. Mota, E.M. Ortigosa, "Real-time optical flow computation using FPGAs", presentado en the Early Cognitive Vision Workshop, Isle of Skye, Scotland, UK, Mayo 2004. Disponible Online: <http://www.cn.stir.ac.uk/ecovision-ws/schedule.php>
20. S. Mota, E. Ros, J. Díaz, S. Tan, J. Dale and A. Johnston, "Detection and tracking of overtaking cars for driving assistance," presentado en the Early Cognitive Vision Workshop, Isle of Skye, Scotland, UK, Mayo 2004. Disponible Online: <http://www.cn.stir.ac.uk/ecovision-ws/schedule.php>
21. E. Ros, J. Díaz, S. Mota, "Neural multilayer structure for motion pattern segmentation," presentado en Brain Inspired Cognitive Systems (BICS-2004), Stirling, Scotland (UK), Agosto 2004

22. F. Vargas-Martin, M. D. Peláez-Coca, E. Ros, J. Díaz and S. Mota, "Video processing based on reconfigurable logic for low vision aids," presentado en the II EOS Topical Meeting on Physiological Optics, Granada, España, Septiembre 2004
23. F. Vargas-Martin, M. D. Peláez-Coca, E. Ros, J. Díaz. and S. Mota, "Augmented view visual aid based on reconfigurable logic for peripheral vision loss", presentado en Conference and Workshop on Assistive Technologies for Vision and Hearing Impairment, CVHI'2004, Granada, España, Junio 2004.
24. J. Díaz, E. Ros, S. Mota, G. Botella, A. Cañas, and S. Sabatini, "Optical flow for cars overtaking monitor: the rear mirror blind spot problem," In *Proc. of 10th. International Conference on Vision in Vehicles (VIV'2003)*, Granada, España, Septiembre 2003
25. S. Mota, E. Ros, J. Díaz, G. Botella, F. Vargas, and A. Prieto, "Motion driven segmentation scheme for car overtaking sequences," In *Proc. of 10th. International Conference on Vision in Vehicles (VIV'2003)*, Granada, España, Septiembre 2003.

#### 6.2.4. Conferencias nacionales

26. J. Díaz, E. Ros, S. Mota, E. M. Ortigosa, R. Carrillo, and R. Agis, "Cómputo de flujo óptico en tiempo real mediante FPGAs," presentado en las *Jornadas sobre computación reconfigurable y aplicaciones*, Barcelona, España, Septiembre 2004, pp. 481-488.
27. E. Ros-Vidal, J. Díaz, S. Mota, and F. Vargas-Martín, "Procesamiento de imágenes en tiempo real para ayuda a pacientes con baja vision", presented at the *XXI Congreso Anual de la Sociedad Española de Ingeniería Biomédica, (CASEIB 2003)*, Mérida, España, Noviembre 2003, pp. 153-156.
28. S. Mota, E. Ros, B. del Pino, G. Botella, F. Pelayo, A. Prieto, "Sistema de monitorización de adelantamientos en tiempo real", presentado en las *II Jornadas sobre computación reconfigurable y aplicaciones*, Almuñecar, Granada, España, Septiembre 2002, pp. 93-96.

Una de las características principales del sistema descrito en este trabajo es que, con un número de píxeles muy reducido en comparación con el número de píxeles en la imagen procedente de la cámara, es capaz de realizar una segmentación basada sólo en primitivas de movimiento. La extracción de bordes en la etapa inicial hace que el sistema sea poco denso en las siguientes etapas de procesamiento. Por tanto, como trabajo futuro se plantea diseñar una arquitectura que aproveche esta característica, realizando un procesamiento dirigido por eventos (*event-driven*). En esta arquitectura, los circuitos de los diferentes módulos de procesamiento estarán en espera y sólo funcionarán tras la llegada de un nuevo evento.

En el trabajo futuro también se abordará la evaluación de las primitivas de movimiento en el marco de la aplicación de Visión de Túnel.

Otro trabajo futuro consiste en desarrollar módulos de post-procesamiento específicos para aplicar el sistema de detección de movimiento descrito a aplicaciones relacionadas con la seguridad.



## 6.3. Conclusiones

Finalmente se resumen las principales contribuciones de esta tesis:

- Hemos desarrollado un algoritmo para la detección de movimiento, original e inspirado en el comportamiento del sistema de detección de movimiento de los insectos, en particular en el de la mosca. El algoritmo sigue una estructura en columnas propia del sistema visual de la mosca y está basado en la extracción previa de características espaciales, por lo que es un algoritmo poco denso, y con un coste computacional bajo.
- Dada la dificultad para trabajar con secuencias reales, hemos desarrollado un procedimiento de evaluación del algoritmo. Se han definido parámetros de calidad, como la Sensibilidad Derecha o el Rendimiento de la Segmentación. Se ha contrastado un alto rendimiento y unas altas prestaciones del algoritmo. Esto hace posible que con secuencias reales, sea factible la segmentación de objetos basándonos sólo en criterios de movimiento. Esta alta eficiencia del algoritmo se alcanza con un número muy reducido de píxeles, en comparación con las dimensiones de la imagen inicial.
- Hemos implementado el sistema en tiempo real. Una arquitectura de altas prestaciones, con un camino de datos microsegmentado que hace un uso intensivo de los recursos de procesamiento paralelos propios de las FPGAs. Esto conforma un sistema muy eficiente capaz de procesar más de 40 Mpps. La arquitectura es modular, lo que la hace muy adecuada para su uso en distintos campos de aplicación.
- Se ha descrito una aplicación para la monitorización de los adelantamientos donde el tiempo real es un requisito imprescindible. Para ello, ha sido necesario introducir dos nuevos módulos de procesamiento; uno dedicado a la corrección de la perspectiva que afecta a la percepción de la velocidad (esto se ha abordado desde dos aproximaciones distintas), y el otro para realizar un seguimiento del vehículo que realiza la maniobra de adelantamiento, y que podría usarse para disparar una señal de alerta o aviso en caso de situaciones peligrosas de posible colisión lateral. El sistema ha sido evaluado con secuencias reales de adelantamientos, resultando su eficiencia independiente de las condiciones luminosas, climatológicas, de la tipología del vehículo que adelanta y de su velocidad relativa.
- Se ha descrito otra aplicación donde el tiempo real también es un factor imprescindible, un sistema de ayuda en ambientes dinámicos para pacientes con visión de túnel. Aunque esta aplicación ha sido bien definida, no se ha evaluado aún con pacientes, por lo que su eficiencia está aún por comprobar.

---

**APENDICE:**

**Sistema de realidad aumentada para la  
ayuda a pacientes con Visión de Túnel**

---

En este apéndice se introduce la aplicación del sistema de detección de movimiento para la ayuda a pacientes con Visión de Túnel en el contexto de los sistemas paliativos basados en realidad aumentada. Se describen algunos de los problemas visuales relacionados con la Visión de Túnel y algunos de los dispositivos paliativos existentes en el mercado.

## A1.1. Introducción

Existen varias enfermedades (Retinosis Pigmentaria, El Síndrome de Usher y el Glaucoma) que causan restricciones del campo visual periférico, lo que se conoce como Visión de Túnel. Cuando la pérdida de la visión periférica es severa, con campos visuales residuales de menos de 20 grados de amplitud (legalmente una persona con menos de 20 grados de campo visual es calificada como ciega), su capacidad para caminar o desenvolverse en un medio urbano está muy limitada, ya que no tienen información de contexto. El efecto es similar al que sentimos cuando miramos a través de tubo de cierta longitud. Las relaciones sociales también se ven afectadas ya que les resulta difícil situar a su interlocutor durante una conversación con distintos miembros de un grupo o localizar amigos y familiares en la calle. La reducción del campo visual aparece de forma gradual y normalmente el paciente no advierte el problema hasta que está en una etapa severa.

A pesar de la reducción del campo visual, sin embargo, estos pacientes pueden conservar una visión central (la fovea) de alta resolución [PAG88, DIC98]. Esto hace que estos pacientes tengan gran agudeza visual, lo que les permite leer o ver la televisión.

Estas enfermedades también suelen producir ceguera nocturna, muy a menudo antes de que se produzca la reducción del campo visual. Los individuos que padecen de ceguera nocturna no sólo ven de forma deficiente por la noche, sino que requieren de más tiempo para adaptarse a los cambios de iluminación tanto de día como de noche.

Las ayudas que existen actualmente para paliar la visión de túnel van desde el uso de lentes divergentes portátiles a dispositivos telescópicos invertidos que reducen la imagen del campo visual total superponiéndolo en la región foveal. Una consecuencia inmediata de estos dispositivos es que se compromete la alta resolución de la visión central de los pacientes [DIC98, HOE85]. Son muchos los pacientes de visión de túnel que rechazan este tipo de dispositivos, especialmente en situaciones dinámicas. La insatisfacción tiene diferentes causas, entre ellas la pérdida de resolución, la distorsión de la imagen que producen las lentes y la restricción del campo dinámico que se obtiene con el movimiento de barrido libre que efectúa de forma natural el ojo.

Las ayudas visuales para paliar la ceguera nocturna consisten en visores nocturnos y son más comunes debido a su desarrollo y uso en aplicaciones militares, caza nocturna y supervivencia. Muchos de estos dispositivos son amplificadores de la luz ambiente [BER74]. También se han evaluado dispositivos de video con iluminación infrarroja combinados con visores portátiles (Head Mounted Display (HMD)). Otras estrategias consisten en el incremento en la iluminación del entorno y los objetos [MOR83]. Se han desarrollado lámparas con un campo amplio cuya ventaja fundamental con respecto a los dispositivos basados en visión nocturna es que preservan la visión estereoscópica del paciente y permiten una percepción más real del entorno.

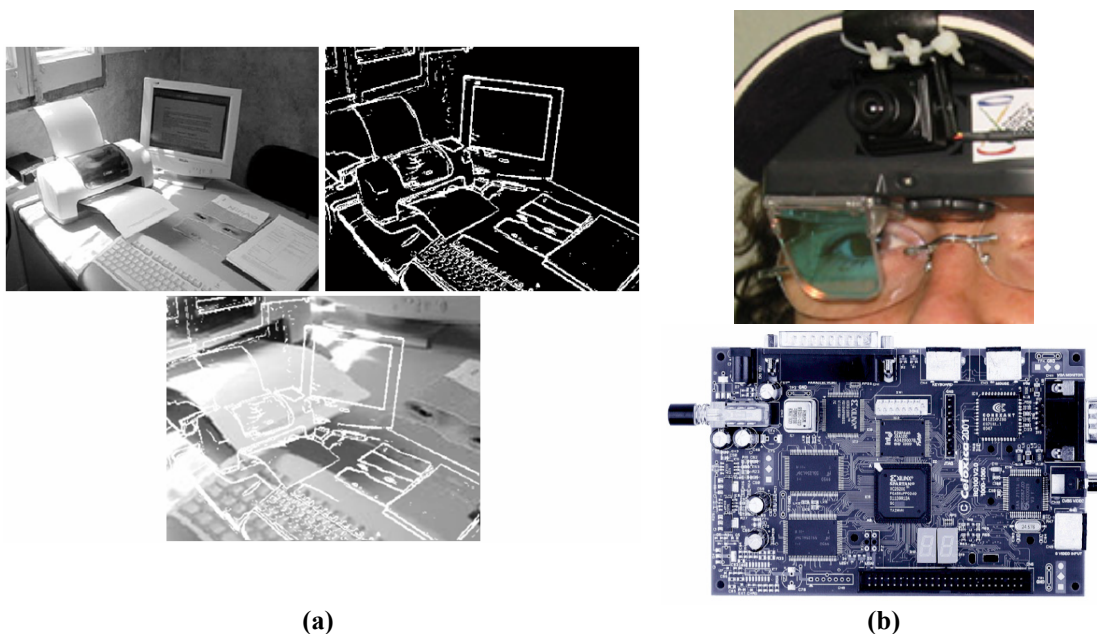
Vargas-Martín y Peli [VAR02] han sugerido que un sistema portátil para pacientes con baja visión que les permita cierta movilidad debe reunir los siguientes requisitos:

- proporcionar información de los objetos existentes en el campo visual periférico;
- ser compatible con las capacidades visuales remanentes del paciente;
- ser compatible con los movimientos de barrido libre que efectúa el ojo;
- deben ser capaces de funcionar con luz del día y en la oscuridad;
- deben permitir el uso de gafas graduadas;
- deben ser portátiles, poco pesados, duraderos para la actividad diaria y con un aspecto físico aceptables;
- con un coste de producción limitado.

## **A1.2. La estrategia de realidad aumentada para dispositivos de ayuda a la baja visión**

Peli [PEL99, VAR02] propuso el uso de la realidad aumentada como nueva estrategia para el diseño de dispositivos de ayuda para la visión de túnel. Consiste en proporcionar suficiente información del campo visual periférico, para mejorar la movilidad de los pacientes, sin comprometer la resolución de la visión central remanente del paciente. La idea es presentar un resumen de la información ambiental (los bordes de la escena) superpuesta en la zona de visión foveal de alta resolución (Figura A.1 a). Un prototipo de dicho sistema fue implementado para demostrar la viabilidad del concepto de realidad aumentada.

Ros et al. [ROS03, ROS06] proponen una nueva implementación del sistema de ayuda de realidad aumentada basada en plataformas de bajo coste con dispositivos reconfigurables (FPGA). Este nuevo diseño permite personalizar los dispositivos para cada paciente de una forma muy sencilla. La FPGA permite el procesamiento de la imagen en tiempo real y que el paciente tenga un control directo de un conjunto de parámetros del procesamiento mediante el uso de un interfaz de entrada específico (Figura A.1 a y b).



**Figura A.1.** (a) Las imágenes superiores son la escena real y la imagen de bordes que contiene el resumen de la información ambiental; la imagen inferior es la superposición de la imagen de bordes de la escena sobre la región de visión residual (la fovea) del paciente, que puede escoger entre centrarse en la información real de la imagen o en la información contextual (los bordes); (b) Dispositivo completo de ayuda a la baja visión compuesto de: una cámara (adquisición de la imagen), FPGA (dispositivo de procesamiento de imágenes en tiempo real) y un visor portátil (HMD -Head Mounted Display-).

Son varios los factores que motivan la utilización de hardware reconfigurable para este tipo de dispositivos:

- La población a la que van dirigidos es pequeña y muy diversa. Es decir, cada paciente tiene unas necesidades distintas debido al grado distinto del desarrollo de la enfermedad. Por lo tanto, en vez de desarrollar una plataforma distinta para cada tipo de paciente, se puede dotar de una plataforma única y personalizable. Es decir, distintos pacientes con visión túnel pueden tener más o menos campo visual que el dispositivo puede explotar de forma eficiente.
- Evolución de la enfermedad con el tiempo. El dispositivo se debe poder adaptar a la evolución de cada paciente.
- Portabilidad. El dispositivo de procesamiento de imágenes está embebido en un sistema de ayuda a la visión portátil, para que pueda ser utilizado para caminar, etc.
- Bajo coste. Estos dispositivos sólo son paliativos y personales. Es decir cada paciente tiene unas necesidades propias que lo hacen difícilmente compatible. Por ello, el coste de la plataforma completa de baja visión debe ser reducido.
- El uso de cámaras que funcionen en el rango de los infrarrojos posibilita su uso bajo distintas condiciones de iluminación (diurnas y nocturnas).
- Es posible realizar distintos procesamientos en el mismo equipo que integren ayuda para varias deficiencias visuales.

El dispositivo de ayuda a pacientes con visión de túnel cumple bastante bien con los requisitos enumerados por Vargas-Martín y Peli [VAR02]. Así mismo, los resultados preliminares de la evaluación de dicho dispositivo [VAR05] muestran que:

- el campo visual efectivo en sujetos con visión de túnel aumenta de forma significativa con el uso del dispositivo, más cuanto menor sea el campo visual residual;
- no restringe la agudeza visual residual;
- no restringe la sensibilidad al contraste;
- aumenta notablemente la movilidad de los pacientes dado que les resulta mucho más fácil localizar objetos en el entorno y evitar obstáculos;
- en tareas de localización, lectura y manipulación los pacientes encuentran muy útil la ayuda.

### **A1.3. Dispositivos de realidad aumentada para ayuda a la Visión de Túnel con el movimiento como descriptor de la escena**

Como se ha visto, los dispositivos paliativos de realidad aumentada basados en FPGAs proporcionan grandes ventajas con respecto a otras aproximaciones en el mercado. Como se ha comentado, es posible realizar distintos tipos de procesamiento que proporcionen ayudas para diversas situaciones. Sin embargo, la información contextual que se ofrece en los dispositivos basados en FPGAs mostrados es estática [ROS03, ROS06] y la evaluación que se ha realizado ha tenido en cuenta solamente situaciones y tareas en entornos con objetos estáticos [VAR05].

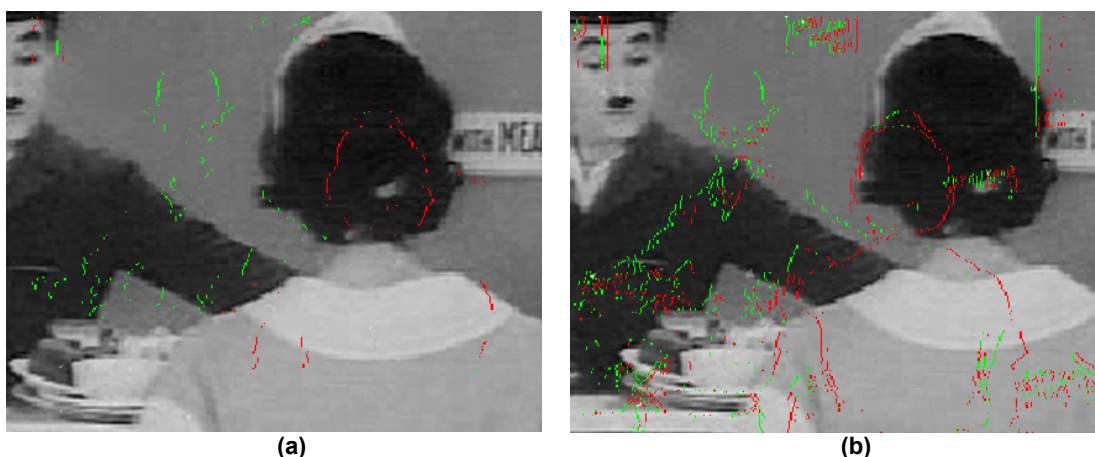
Si usamos un sistema de detección de movimiento y superponemos sobre el campo de visión residual la información de movimiento extraída de la escena, el paciente podría disponer de datos dinámicos que le serían de utilidad en situaciones en las que se producen movimientos rápidos de objetos, por ejemplo, unos grandes almacenes, un cruce de una calle, etc. (Figura A.2).

El sistema de detección de movimiento descrito en este trabajo sería muy aplicable para este tipo de dispositivos de ayuda porque:

- está basado en mapas de saliencia poco densos, por lo que al superponer la información de movimiento sobre la visión residual esta no se satura.
- es un sistema reconfigurable que admite la superposición de bordes y/o la superposición de información relativa al movimiento sobre la imagen real.

- no requiere de módulos de pre-procesamiento ni de post-procesamiento que interpreten la información de salida (en el caso de la aplicación descrita en el Capítulo 5, era necesario un módulo de interpretación de la información para realizar el seguimiento; y otro módulo de corrección de la perspectiva). En caso de la ayuda descrita en este apéndice es el cerebro del propio paciente es el que se encargará de esta tarea.

Obsérvese que la información de movimiento que obtenemos con el algoritmo completo se reduce a muy pocos puntos (Figura A.2 a). Incluso para un individuo sin anomalías visuales sería deseable la aportación de información en un número de píxeles mayor. Recordemos que el algoritmo completo requería de una etapa de filtrado basada en la detección de sólidos rígidos en la imagen. Si no incluimos este filtrado (Figura A.2 b) el resultado incluye mayor cantidad de información, parte de la cual es errónea, pero dado que el módulo de post-procesamiento lo constituye el propio cerebro del paciente, esa información errónea no debería confundirlo, sino que dota de mayor textura al sistema aportando mayor cantidad de información. Sin embargo, esto deberá ser evaluado con pacientes reales.



**Figura A.2.** (a) Superposición de la información dinámica sobre la visión residual considerando el sistema de detección de movimiento completo; (b) Superposición de la información dinámica sobre la visión residual considerando el sistema de detección de movimiento sin el módulo de detección de sólidos rígidos.

La evaluación de este dispositivo en pacientes con visión de túnel reales está por realizarse. Sin embargo los buenos resultados obtenidos en situaciones estáticas con el sistema equivalente hacen suponer que estos también serían muy positivos en el caso dinámico. En el trabajo futuro se abordará la evaluación de esta modalidad visual (movimiento) en el marco de esta aplicación. Esta evaluación nos va a permitir determinar en qué circunstancias la información de movimiento resulta útil para el paciente, así como las transformaciones adicionales a realizar según el contexto de funcionamiento, por ejemplo cuando convenga aislar el movimiento externo del propio movimiento del paciente que porte el dispositivo.

Existen algunos pacientes (casos muy raros) con un déficit en la detección de movimiento causado por una lesión en la región parietal-occipital. Dicha lesión provoca la incapacidad para percibir el movimiento por encima de 6 °/segundo. Esto es, son capaces de ver un coche, pero no de ver su movimiento. Sin embargo, otras capacidades visuales no se ven afectadas (agudeza visual, estereopsis, discriminación del color) [HES89]. En estos casos, una codificación en color del movimiento de la escena superpuesto sobre la visión normal es de esperar que sí suponga una ayuda.





---

---

## REFERENCIAS

---

---

- [ABD78] Abdou I.E. (1978) Quantitative methods of edge detection. Technical report n° 830, Image Processing Institute, University of Southern California.
- [ADE85] Adelson, E. H. and Bergen, J. R. (1985). Spatiotemporal energy models for the perception of motion. *Journal of the Optical Society of America A*, 2,284-299.
- [AGR05] Motilal Agrawal, Kurt Konolige and Luca Locchi. "Real-time detection of independent motion using stereo", *IEEE workshop on Motion*, 2005, Breckenridge, Colorado.
- [AMT93] Amthor, F. R. & Grzywacz, N. M. (1993). "Inhibition in on-off directionally selective ganglion cells of the rabbit retina", *Journal of Neurophysiology*, 69, 2174-2187
- [BAR65] Barlow, H. B. & Levick, W. R. (1965). "The mechanism of directionally selective units in the rabbit's retina", *Journal of Physiology*, 178, 477-504
- [BAR78] Barlow H. B. (1978) The efficiency of detecting changes of intensity in random dot patterns. *Vision Research* 18 (6): 637-650.
- [BAU92] Bausenwein, B. & Fischbach, K. F. (1992). "Activity labelling patterns in the medulla of *Drosophila melanogaster* caused by motion stimuli", *Cell and Tissue Research*, 270, 25-35
- [BEL03] Belgian Road Safety Institute (2003), Cycling accidents in urban areas: [http://www.fiets.irisnet.be/download/cycling\\_accidents.pdf](http://www.fiets.irisnet.be/download/cycling_accidents.pdf) (último acceso en enero de 2007).
- [BER74] Berson, E. L., L. Mehauffey, 3rd, et al. (1974). "A night vision pockscope for patients with retinitis pigmentosa. Design considerations." *Arch Ophthalmol* 91(6): 495-500.
- [BOR89] Borst A, Egelhaaf M (1989) Principles of visual motion detection. *Trends Neurosci* 12:297–306.
- [BOR90] Borst, A. & Egelhaaf, M. (1990). "Direction selectivity of blowfly motion-sensitive neurons is computed in a two-stage process", *Proceedings of the National Academy of Sciences, USA*, 87, 9363-9367
- [BOR93] Borst, A. & Egelhaaf, M. (1993). "Detecting visual motion: theory and models", In Miles, F. A. & Wallman, J. (Eds), *Visual Motion and its Role in the Stabilisation of Gaze*, London: Elsevier
- [BRO96] Brotz, T. M. & Borst, A. (1996). "Cholinergic and gabaergic receptors on fly tangential cells and their role in visual-motion detection", *Journal of Neurophysiology*, 76, 1786-1799
- [BRO04] Broggi, A., Cerri, P., Antonello, P.C., Multi-resolution vehicle detection using artificial vision. In: *Intelligent Vehicles Symposium*, pp. 310- 314, 2004.
- [BRU05] A. Bruhn, J. Weickert, C. Feddern, T. Kohlberger, and C. Schnorr, "Variational optical flow computation in real time," *IEEE Transactions on Image Processing*, vol. 14 no.5, pp. 608-615, May 2005.
- [BUC76] Buchner, E. (1976). "Elementary movement detectors in an insect visual system", *Biological Cybernetics*, 24, 85-101
- [BUC84] Buchner, E., Buchner, S. & Bülthoff, I. (1984). "Deoxyglucose mapping of nervous activity induced in *Drosophila* brain by visual movement", *Journal of Comparative Physiology A*, 155, 471-483
- [BUC84b] Buchner, E. (1984b). "Behavioural analysis of spatial vision in insects", In Ali, M. (Ed). *Photoreception and Vision in Invertebrates*, New York: Plenum
- [BUS97] Buschbeck, E. K. & Strausfeld, N. J. (1997). "The relevance of neural architecture to visual performance: Phylogenetic conservation and variation in Dipteran visual systems", *Journal of Comparative Neurology*, 383, 282-304

- [CEL06a] Plataforma de prototipado RC-200 de Celoxica:  
<http://www.celoxica.com/products/boards/rc200.asp> (último acceso en enero de 2007)
- [CEL06b] Celoxica, Handel-C Language Reference Manual Online:  
[www.celoxica.com/techlib/files/CEL-W0410251JJ4-60.pdf](http://www.celoxica.com/techlib/files/CEL-W0410251JJ4-60.pdf) (último acceso en enero de 2007).
- [CLI96] Clifford, C. W. G. & Langley, K. (1996). "A model of temporal adaptation in fly motion vision", *Vision Research*, 36, 2595-2608
- [COL70] Collett, T. (1970). "Centripetal and centrifugal visual cells in medulla of the insect optic lobe", *Journal of Neurophysiology*, 33, 239-256
- [COL72] Collett, T. (1972). "Visual neurones in the anterior optic tract of the privet hawk moth", *Journal of Comparative Physiology*, 78, 396-433
- [COM01] Comisión Europea, WHITE PAPER European transport policy for 2010: time to decide, 2001, [http://europa.eu.int/eur-lex/en/com/wpr/2001/com2001\\_0370en.html](http://europa.eu.int/eur-lex/en/com/wpr/2001/com2001_0370en.html) (último acceso en enero de 2007)
- [DAR06] A. Darabiha, W. J. MacLean and Jonathan Rose, "Reconfigurable Hardware Implementation of a Phase-Correlation Stereo Algorithm," *Machine Vision and Applications Journal*, March, 2006.
- [DEB97] Debevec P.E. and Malik J. (1997) Recovering High Dynamic Range Radiance Maps from Photographs. *Proceedings of ACM SIGGRAPH 97*, ACM, 369-378.
- [DEL93] Delbrück, T. (1993). Silicon retina with correlation-based, velocity-tuned pixels. *IEEE Trans. Neural Networks* 4: pp. 529-541.
- [DEN98] Deng, G and Pinoli J.C. (1998) Differentiation-based edge detection using the logarithmic image processing model. *J. of Mathematical Imaging and Vision* 8:161-180.
- [DER86] de Ruyter van Steveninck, R. R., Zaagman, W. H. & Mastebroek, H. A. K. (1986). "Adaptation of transient responses of a movement-sensitive neuron in the visual system of the blowfly *Calliphora erythrocephala*", *Biological Cybernetics*, 54, 223-236
- [DEV80] DeVoe, R. D. (1980). "Movement sensitivities in the fly's medulla", *Journal of Comparative Physiology A*, 138, 93-119
- [DEV82] DeVoe, R. D., Kaiser, W., Ohm, J. & Stone, L. S. (1982). "Horizontal movement detectors of honeybees: directionally-selective visual neurons in the lobula and brain", *Journal of Comparative Physiology A*, 147, 155-170
- [DGT04] Dirección General de Tráfico, Anuario de accidentes de circulación con víctimas, 2004.  
<http://www.dgt.es/estadisticas/accidentes.htm> (último acceso en enero de 2007)
- [DIA06] Díaz, J., Ros, E., Pelayo, F., Ortigosa, E. M. and Mota, S. FPGA based real-time optical-flow system. *IEEE Transactions on Circuits for Video Technology*. Volume 16, Number 2, pp. 274-279, February 2006.
- [DIC98] Dickinson, C. (1998) "Low vision: principles and practice" Oxford; Boston: Butterworth-Heinemann.
- [DOU95] Douglass, J. K. & Strausfeld, N. J. (1995). "Visual-motion detection circuits in flies: Peripheral motion computation by identified small-field retinotopic neurons", *Journal of Neuroscience*, 15, 5596-5611
- [DOU96] Douglass, J. K. & Strausfeld, N. J. (1996). "Visual motion-detection circuits in flies: Parallel direction-sensitive and non-direction-sensitive pathways between the medulla and lobula plate", *Journal of Neuroscience*, 16, 4551-4562
- [DOU98] Douglass, J. K. & Strausfeld, N. J. (1998). "Functionally and anatomically segregated visual pathways in the lobula complex of a Calliphorid fly", *Journal of Comparative Neurology*, 396,

84-104

- [DRA76] Drasdo, N. (1976). "Visual field expanders." *Am J Optom Physiol Opt* 53(9 Pt 1): 464-7.
- [DRU00] T. Drummond, R. Cipolla, *Real-time tracking of multiple articulated structures in multiple views*, in: D. Vernon (Ed.), *Proc 6th European Conference on Computer Vision*, Dublin, Eire, June/July 2000.
- [EGE90] Egelhaaf, M., Borst, A. & Pilz, B. (1990). "The role of GABA in detecting visual motion", *Brain Research*, 509, 156-160
- [EGE93] Egelhaaf, M. & Borst, A. (1993). "Movement detection in arthropods", In Miles, F. A. & Wallman, J. (Eds), *Visual Motion and its Role in the Stabilisation of Gaze*, London: Elsevier
- [ENR66] Enroth-Cugell C. and Robson J. C.(1966) The contrast sensitivity of retinal ganglion cells of the cat. *J. Physiol.*, 187:517-552.
- [ETI96] Etienne-Cummings, R. and Van der Spiegel, J. (1996). *Neuromorphic vision sensors. Sensors and Actuators A* 56: pp.19-29.
- [EUR06] European road safety observatory (2006):  
[http://www.erso.eu/knowledge/content/50\\_vehicle/heavy\\_goods\\_vehicles.htm](http://www.erso.eu/knowledge/content/50_vehicle/heavy_goods_vehicles.htm) (último acceso en enero de 2007)
- [EWA00] Ewald, A., Willhoeft, V., 2000. Laser Scanners for Obstacle Detection in Automotive Applications. In *Proceedings of the IEEE Intelligent Vehicle Symposium*, 682-687.
- [FAN02] Fang, Y., Masaki, I., Horn, B., 2002. Depth-Based Target Segmentation for Intelligent Vehicles: Fusion of Radar and Binocular Stereo. *IEEE Transactions on Intelligent Transportation Systems* 3 (3).
- [FIE87] Field, D. J., Relations between the statistic of natural images and the response properties of cortical cells, *J. Opt. Soc. Am. A4*, 1987, 2379-2394.
- [FIE93] Field D. J., Hayes A. and Hess R. F. (1993) Contour integration by the human visual system: evidence for local "association field". *Vision Research* 33 (2): 173-193.
- [FIC06] Ficosa Digital blind spot detector. Web link:  
[http://www.ficosa.com/eng/home\\_noticiaseventos.htm](http://www.ficosa.com/eng/home_noticiaseventos.htm) (último acceso en enero de 2007)
- [FLE03] Fletcher, L., Petersson, L., Zelinsky, A., Driver assistance systems based on vision in and out of vehicles. In: *Intelligent Vehicles Symposium*, pp. 322- 327, 2003.
- [FOR04] S. Forstmann, S. Thüring, Y. Kanou, J. Ohya, and A. Schmitt, "Real Time Stereo By Using Dynamic Programming", presented at the Conference on Computer Vision and Pattern Recognition Workshop, Washington, D.C., June 27 - July 02, 2004, Vol. 3, pp.29.
- [FRA92] Franceschini, N., Pichon, J.M., and Blanes, C. (1992). From insect vision to robot vision. *Phil. Trans. R. Soc. B* 337: pp. 283-294.
- [FRA00] Franke U. et al. (2000) From door to door- Principles and Application on Computer Vision for driver assistant systems. IN: *Proceeding of Intelligent Vehicle Technologies: Theory and Applications*, Arnold.
- [GEI81] Geiger, G. & Nässel, D. R. (1981). "Visual orientation behaviour of flies after selective laser beam ablation of interneurons", *Nature*, 293, 398-399
- [GIL85] Gilbert C. D. and Wiesel T. N. (1985) Intrinsic connectivity and receptive field properties in visual cortex, *Vision Research* 25(3):365-374.
- [GIL91] Gilbert, C., Penisten, D. K. & DeVoe, R. D. (1991). "Discrimination of visual-motion from flicker by identified neurons in the medulla of the fleshfly *Sarcophaga bullata*", *Journal of Comparative Physiology A*, 168, 653-673

- [GIL97] Gilbert C. (1997) Visual control of cursorial prey pursuit by tiger beetles (cincindelidae). *J. Comparative Physiology A*181: 217-230.
- [GON92] Gonzalez R. & Woods R. (1992) *Digital Image Processing*. Ed. Addison-Wesley.
- [GON05] M. Gong; R. Yang, "Image-gradient-guided real-time stereo on graphics hardware", in *Proc. Fifth International Conference on 3-D Digital Imaging and Modeling*, Ottawa 13-16 June 2005, pp. 548-555.
- [GOO75] Goodwin, A. W. & Henry, G. H. (1975). "Direction selectivity of complex cells in a comparison with simple cells", *Journal of Neurophysiology*, 38, 1524-1540
- [GOR98] S. Görzig & U. Franke, ANTS-Intelligent Vision in urban Traffic, *Proc. IEEE Conference on Intelligent Transportation Systems*, 1998.
- [GRO93] Grosf D. H., Shapley R. M. and Hawken M. J. (1993) Macaque V1 neurons can signal 'illusory' contours. *Nature*, 365:550-552.
- [GRZ90] Grzywacz, N. M., Amthor, F. R. & Mistler, L. A. (1990). "Applicability of quadratic and threshold models to motion discrimination in the rabbit retina", *Biological Cybernetics*, 64, 41-49
- [GRZ93] Grzywacz, N. M. & Amthor, F. R. (1993). "Facilitation in on-off directionally selective ganglion cells of the rabbit retina", *Journal of Neurophysiology*, 69, 2188-2199
- [GUE98] Técnicas de Diseño de Algoritmos. Rosa Guerequeta y Antonio Vallecillo. Servicio de Publicaciones de la Universidad de Málaga. 1998 (ISBN: 84-7496-666-3)
- [GUY96] Guy, G. and Medioni, G. Inferring global perceptual contours from local features. *International Journal of Computer Vision*, v.20 n.1-2, p.113-133, Oct. 1996
- [HAN98] Handmann U. et al. (1998) Computer Vision for Driver Assistance Systems. IN: *Proceeding of SPIE*, vol. 3364, Session Enhance and Synthetic Vision, pp. 136-147.
- [HAR98] Harrison, R.R. and Koch, C. (1998). An analog VLSI model of the y elementary motion detector, In *Advances in Neural Information Processing Systems 10*, Jordan, M.I., Kearns, M.J., and Solla, S.A. (Eds) MIT Press: Cambridge, Mass. pp. 880-886.
- [HAR00] Harrison, R.R. and Koch, C. A silicon implementation of the fly's optomotor control system. *Neural Computation*, 12: 2291-2304, 2000.
- [HAS56] Hassenstein, B. & Reichardt, W. (1956). "Systemtheoretische analyse der Zeit-, Reihenfolgen- und Vorzeichenbewertung bei der bewegungsperzeption des rüsselkäfers *Chlorophanus*", *Zeitschrift für Naturforschung*, 11b, 513-524
- [HAU76] Hausen, K. (1976). "Functional characterisation and anatomical identification of motion sensitive neurons in the lobula plate of the blowfly *Calliphora erythrocephala*", *Zeitschrift für Naturforschung*, 31, 628-633
- [HAU82] Hausen, K. (1982). "Motion sensitive interneurons in the optomotor system of the fly.1: The horizontal cells, structure and signals", *Biological Cybernetics*, 45, 143-156
- [HAU82b] Hausen, K. (1982b). "Motion sensitive interneurons in the optomotor system of the fly.2: The horizontal cells, receptive field organisation and response characteristics", *Biological Cybernetics*, 46, 67-79
- [HAU83] Hausen, K. & Wehrhahn, C. (1983). "Microsurgical lesion of horizontal cells changes optomotor yaw responses in the blowfly *Calliphora erythrocephala*", *Proceedings of the Royal Society London B*, 208, 57-71
- [HAU89] Hausen, K. & Egelhaaf, M. (1989). "Neural mechanisms of visual course control in insects", In *Stavenga, D. G. & Hardie, R. C. (Eds), Facets of Vision*, Berlin: Springer-Verlag

- [HAU90] Hausen, K. & Wehrhahn, C. (1990). "Neural circuits mediating visual control of flight in flies. 2: Separation of two control systems by microsurgical brain lesions", *Journal of Neuroscience*, 10, 351-360
- [HAU93] Hausen, K. (1993). "The decoding of retinal image flow in insects", In Miles, F. A. & Wallman, J. (Eds), *Visual Motion and its Role in the Stabilisation of Gaze*, London: Elsevier
- [HAV03] Havlík, Jan: *Edge Detection for Image Processing Based Motion Parametrization*. Proceedings of POSTER 2003, FEL ČVUT Praha, 2003.
- [HEI78] Heisenberg, M., Wonneberger, R. & Wolf, R. (1978). "Optomotor-blind H31 – a *Drosophila* mutant of the lobula giant neurons", *Journal of Comparative Physiology*, 124, 287-296
- [HEI96] Heisele, B., Neef, N., Ritter, W., Schneider, R., Wanielik, G., 1996. Object Detection in Traffic Scenes by a Colour Video and Radar Data Fusion Approach. First Australian Data Fusion Symposium, 48–52.
- [HEN82] Hengstenberg, R. (1982). "Common visual response properties of giant vertical cells in the lobula plate of the blowfly *Calliphora*", *Journal of Comparative Physiology A*, 149, 179-193
- [HEN82b] Hengstenberg, R., Hausen, K. & Hengstenberg, B. (1982b). "The number and structure of giant vertical cells (VS) in the lobula plate of the blowfly *Calliphora erythrocephala*", *Journal of Comparative Physiology A*, 149, 163-177
- [HES89] Hess, R.H., Baker, C.L.Jr. and Zihl, J. The "Motion-blind" patient: low-level spatial and temporal filters. *J. Neurosci.* 9: 1628-1640 (1989).
- [HIL87] Hildreth, E. C. & Koch, C. (1987). "The analysis of visual motion: from computational theory to neuronal mechanisms", *Annual Review of Neuroscience*, 10, 477-533
- [HOE85] Hoefft, W.W., Feinbloom, W., et al. (1985) "Amorphic lenses: a mobility aid for patients with retinitis pigmentosa" *Am J Optom Physiol Opt*, 62(2):142-8.
- [HUB98] Huber, S.A. and Bühlhoff, H. H. (1998). Simulation and Robot Implementation of Visual Orientation Behaviors of Flies. In: Pfeifer et al. (Eds.) *From animals to animats*, proceedings of the fifth conference on the simulation of adaptive behaviour, pp 77 – 85. Cambridge, MA: MITpress.
- [IBB91] Ibbotson, M. R., Maddess, T. & Dubois, R. (1991). "A system of insect neurons sensitive to horizontal and vertical image motion connects the medulla and midbrain", *Journal of Comparative Physiology A*, 169, 355-367
- [IBB92] Ibbotson, M. R. (1992). "Direction-selective neurons with tonic and phasic response profiles contribute to the optokinetic system of *Apis mellifera*", *Naturwissenschaften*, 79, 467-470
- [IBB94] Ibbotson, M. R., Mark, R. F. & Maddess, T. L. (1994). "Spatiotemporal response properties of direction-selective neurons in the nucleus of the optic tract and dorsal terminal nucleus of the wallaby, *Macropus eugenii*", *Journal of Neurophysiology*, 27, 2927-2943
- [IND03] Industrial Light & Magic (2003) OpenEXR, High Dynamic Range Image File Format. Lucas Digital Ltd.
- [ITT03] Itti, L. (2003), The Beobot Platform for Embedded Real-Time Neuromorphic Vision, In: *Advances in Neural Information Processing Systems*, Vol. 15, Hardware Demo Track, (T. G. Dietterich, S. Becker, Z. Ghahramani Ed.), Cambridge, MA:MIT Press.
- [JOH90] Johnson R.P. (1990) Contrast-based edge detection. *Pattern Recogn.*, 23:311–318.
- [JOU87] Jourlin M. and Pinoli J.C. (1987) Logarithmic image processing. *Acta Stereol.*, 6:651–656.
- [JOU88] Jourlin M. and J.C. Pinoli J.C. (1988) A model for logarithmic image processing. *J. Microsc.*, 149:21–35.

- [KAL60] Kalman, R. E. "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME - Journal of Basic Engineering* Vol. 82: pp. 35-45 (1960).
- [KAL61] Kalman, R. E., Bucy R. S., "New Results in Linear Filtering and Prediction Theory", *Transactions of the ASME - Journal of Basic Engineering* Vol. 83: pp. 95-107 (1961).
- [KAN94] Kanungo T., Jaisimba M.Y., Palmer J. and Haralick R.M. (1994) Modelling edges and subpixel accuracy using the local energy approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(4): 405-410.
- [KIE74] Kien, J. (1974). "Sensory integration in the locust optomotor system. II: Direction selective neurons in the circumoesophageal connectives and the optic lobe", *Vision Research*, 14, 1255-1268
- [KIR71] Kirsch, R.A., Computer Determination of the Constituent Structure of Biological Images, *Comp. Biomed. Res.*(4), No. 3, June 1971, pp. 315-328. Kirsch operator edge detector.
- [KIT81] Kitchen L. and Rosenfeld A. (1981) Edge evaluation using local edge coherence. *IEEE Transaction systems, man and cybernetics*, SMC-11(9): 597-605.
- [KRA05] Grzegorz Krawczyk, Michael Goesele and Hans-Peter Seidel. Photometric Calibration of High Dynamic Range Cameras. MPI Technical Report MPI-I-2005-4-005, 2005.
- [KRU00] Computation in Neural Systems (2002). 13(4):553-576. Multi-modal Estimation of Collinearity and Parallelism in Natural Image Sequences. Norbert Krüger\_ and Florentin Wörgötter
- [KRU02] Krüger, N. and Wörgötter, F. Multi--Modal Statistics of Edges in Natural Image Sequences. Proceedings of the Workshop 'Dynamic Perception', Bochum, Germany 2002.
- [LAN97] Land, M. F. (1997). Visual acuity in insects. *Annual Review of Entomology*, 42:147, 177.
- [LAN97b] Land, M. F. & Collett, T. S. (1997b). "A survey of active vision in invertebrates", In Srinivasan, M. V. & Venkatesh, S. (Eds), *From Living Eyes to Seeing Machines*, Oxford: Oxford University Press
- [LAR98] Larson G.W. (1998) LogLuv Encoding for Full-Gamut, High-Dynamic Range Images. *Journal of Graphics Tools*. A K Peters Ltd., 3(1):815.830.
- [LAU89] Laughlin, S. B. (1989). "Coding efficiency and design in visual processing", In Stavenga, D. G. & Hardie, R. C. (Eds), *Facets of Vision*, Berlin: Springer-Verlag
- [LAU93] Laughlin, S.B., Weckström, M., 1993. Fast and slow photoreceptors - a comparative study of the functional diversity of coding and conductances in the Diptera. *J Comp Physiol A* 172, 593-609
- [LET59] Lettvin J. Y., Maturana H. R., McCulloch W. S. and Pitts W. H. (1959) What the frog's eye tells the frog's brain, in *Proc. IRE*, 47:1940-1951.
- [LEW98] Lewis, M.A. (1998). Visual navigation in a robot using zig-zag behavior. In *Advances in Neural Information Processing Systems 10*, Jordan, M.I., Kearns, M.J., and Solla, S.A. (Eds) MIT Press: Cambridge, Mass. pp. 822-828.
- [MAD91] Maddess, T., Dubois, R. A. & Ibbotson, M. R. (1991). "Response properties and adaptation of neurons sensitive to image motion in the butterfly *Papilio aegeus*", *Journal of Experimental Biology*, 161, 171-199
- [MAL91] Mallot, H. A., Bulthoff, H. H., Little, J. J., Bohrer, S. (1991) Inverse perspective mapping simplifies optical flow computation and obstacle detection. *Biol. Cybern.*, 64: 177-185.
- [MAR81] Marr, D. & Ullman, S. (1981). "Directional selectivity and its use in early visual processing", *Proceedings of the Royal Society London B*, 211, 151-180
- [MAR82] Marr, D., *Vision* (New York: W. H. Freeman and Company, 1982)



- [MAR05]** J.L. Martín, A. Zuloaga, C. Cuadrado, J. Lázaro, and U. Bidarte, "Hardware implementation of optical flow constraint equation using FPGAs," *Computer Vision and Image Understanding*, vol.98, no. 3, pp. 462-490, June 2005.
- [MER06]** Sistema de medición de distancias con laser (LIDAR): <http://www.merrick.com/servicelines/gis/espanol/lidar.aspx> (último acceso en enero de 2007)
- [MIL87]** Milde, J. J., Seyan, H. S. & Strausfeld, N. J. (1987). "The neck motor system of the fly *Calliphora erythrocephala*. 2: Sensory organisation", *Journal of Comparative Physiology A*, 160, 225-238
- [MIL93]** Milde, J. J. (1993). "Tangential medulla neurons in the moth *Manduca sexta*. Structure and responses to optomotor stimuli", *Journal of Comparative Physiology A*, 173, 783-799
- [MOI97]** Moini, A., Bouzerdoum, A., Eshraghian, K., Yakovleff, A., Nguyen, X.T., Blanksby, A., Beare, R., Abbott, D., and Bonger, R.E. (1997). An insect vision-based motion detection chip. *IEEE J. Solid State Circuits* 32: pp. 279-284.
- [MOR83]** Morrissette, D. L., M. F. Marmor, et al. (1983). "An evaluation of night vision mobility aids." *Ophthalmology* 90(10): 1226-30.
- [MOR90]** Moravek, H. "El Hombre Mecánico (El futuro de la Robótica y la Inteligencia Humana)". Ediciones Temas de Hoy. Madrid, 1990
- [MOT04a]** Mota, S.; Ros, E.; Díaz, J.; Ortigosa, E. M.; Prieto, A.; Motion-driven segmentation by competitive neural processing; *Neural Processing Letters* ; 22(2); pp.125-147
- [MOT04b]** Mota, S.; Ros, E.; Díaz, J.; Ortigosa, E. M.; Agis, R. and Carrillo, R., Real-Time Visual Motion Detection of Overtaking Cars for Driving Assistance Using FPGAs *Lecture Notes in Computer Science*, Volume 3203/2004 pp. 1158-1161
- [MOT04c]** Bio-Inspired Motion Detection for a Blind Spot Overtaking Monitor, S. Mota, E. Ros, E.M. Ortigosa, and F.J. Pelayo *International Journal of Robotics and Automation*, 19(4), 190-196, 2004
- [MOT06]** Motion Capture Technologies. <http://www.mctcameras.com/hsv.htm>
- [MOV78]** Movshon, J. A., Thompson, I. D. & Tolhurst, D. J. (1978). "Receptive field organization of complex cells in the cat's striate cortex", *Journal of Physiology*, 283, 79-99
- [NII04]** Niitsuma, H. and Maruyama, T. "Real-Time Detection of Moving Objects," *Lecture Notes in Computer Science*, FPL 2004, vol. 3203, pp. 1153-1157, September 2004.
- [NIL89]** Nilsson, D.-E. (1989). "Optics and evolution of the compound eye", In Stavenga, D. G. & Hardie, R. C. (Eds), *Facets of Vision*, Berlin: Springer-Verlag
- [ORT06]** Ortigosa, E.M., Cañas, A., Ros, E., Ortigosa, P.M., Mota, S. and Díaz, J. Hardware description of Multi-layer perceptrons with different abstraction levels. *Microprocessors and Microsystems*, 30(7):435-444, 2006.
- [OTS79]** N. Otsu, *A threshold selection method from gray-level histograms*, *IEEE Trans. Systems Man Cybernet.* 9 (1) (1979) 62--66.
- [PAG88]** Pagon, R.A. (1988) "Retinitis Pigmentosa", *Survey of Ophthalmology*, 33, (3), 137-177.
- [PEL99]** Peli, E. (1999) "Augmented Vision for Central Scotoma and Peripheral Field Loss. Vision Rehabilitation: Assessment, Intervention and Outcomes". Lisse, The Netherlands, Swets & Zeitlinger: 70-74.
- [PIC79]** Pick, B. & Buchner, E. (1979). "Visual movement detection under light- and darkadaptation in the fly, *Musca domestica*", *Journal of Comparative Physiology A*, 134, 45-54
- [PIN87]** Pinoli J.C. (1987) Contribution à la modélisation, au traitement et à l'analyse d'image. D.Sc.

thesis, Department of Mathematics, University of Saint-Etienne, France.

- [PIN92]** Pinoli J.C. (1992) Modélisation and traitement des images logarithmiques: Théorie and applications fondamentales. Report No. 6, Department of Mathematics, University of Saint-Etienne, France.
- [POG76]** Poggio, T. & Reichardt, W. (1976). "Visual control of orientation behaviour in the fly. 2: Towards the underlying neural interactions", Quarterly Reviews of Biophysics, 9, 377-438
- [POT94]** Potters, M. & Bialek, W. (1994). "Statistical-mechanics and visual signal-processing", Journal de Physique I, 4, 1755-1775
- [PRA78]** Pratt W.K. (1978) Digital image processing. Wiley-Interscience Publications.
- [PRE70]** J. M. S. Prewitt, Object enhancement and extraction, in: Picture Processing and Psychopictures, B. S. Lipkin and A. Rosenfeld, eds., Academic Press, New York (1970).
- [PUG05]** Extraction of rich and reliable scene representations making use of perceptual grouping and motion. N Pugeault, F Woergoetter and N Krueger. 29 European Conference on Visual Perception St-Petersburg, Russia 2005
- [REI87]** Reichardt W (1987) Evaluation of optical motion information by movement detectors. J Comp Physiol [A] 161:313–315.
- [RIN90]** Rind, F. C. (1990). "Identification of directionally selective motion-detecting neurones in the locust lobula and their synaptic connections with an identified descending neurone", Journal of Experimental Biology, 149, 21-43
- [ROB65]** L.G. Roberts. Machine perception on three-dimensional solids. In J.T. Tippet et al, editor, Optical and Electro-Optical Information Processing, 159-197. MIT Press, 1965
- [ROB77]** Robinson G.S. (1977) Edge detection by compass gradient masks. Comput. Graph. Image Proc., 6:492–501.
- [ROS03]** Ros-Vidal, E., Vargas-Martín, F. et al.(2003) "Procesamiento de imágenes en tiempo real para ayuda a pacientes con baja visión " CASEIB 2003
- [ROS06]** Ros, E., Díaz, J., Mota, S., Vargas-Martín, F., Pelaez-Coca, M.D. (2006): Real-time image processing on a portable aid device for low vision patients. Lecture Notes on Computer Science 3985, 158-163.
- [SAA97]** Saarinen J., Levi D. M and Shen B. (1997) Integration of local pattern elements into a global shape in human vision, in Proceeding of the National Academic of Sciences USA, 94:8267-8271.
- [SAN01]** J. Sánchez, Sillitas infantiles: seguridad mejorable. Tráfico y seguridad vial, 149, pp. 31-33, 2001. Dirección General de Tráfico.
- [SAR96]** Sarpeshkar, R., Kramer, J., Indiveri, G., and Koch, C. (1996). Analog VLSI architectures for motion processing: from fundamental limits to system applications, In Proc. of the IEEE 84: pp. 969-987.
- [SCH89]** Schuling, F. H., Mastebroek, H. A. K., Bult, R. & Lenting, B. P. M. (1989). "Properties of elementary movement detectors in the fly *Calliphora erythrocephala*", Journal of Comparative Physiology A, 165, 179-192
- [SCH03]** Schneider, R., Wenger, J., 2003. High resolution radar for automobile applications. Advances in Radio Science 1, 105–111
- [SHA97]** Shah M. (1997) Fundamentals of computer vision. Ed. University of Central Florida
- [SHI94]** Shi, J. and Tomasi, C. Good Features to Track. IEEE Conference on Computer Vision and Pattern Recognition, pages 593-600, 1994

- [SOB70] Sobel, I.E. (1970) 'Camera Models and Machine Perception', PhD thesis, Stanford Univ.
- [SOB78] I. Sobel, Neighborhood coding of binary images for fast contour following and general array binary processing, *Comput. Graphics Image Process.* 8 (1978) 127-135.
- [SRI91] Srinivasan, M. V., Lehrer, M., Kirchner, W. H. & Zhang, S. W. (1991). "Range perception through apparent image speed in freely flying honeybees", *Visual Neuroscience*, 6, 519-535
- [SRI93] Srinivasan, M. V. (1993). "How insects infer range from visual motion", In Miles, F. A. & Wallman, J. (Eds), *Visual Motion and its Role in the Stabilisation of Gaze*, London: Elsevier
- [SRI99] Srinivasan, M. V., Poteser, M. & Kral, K. (1999). "Motion detection in insect orientation and navigation", *Vision Research*, 39, 2749-2766
- [STE69] Sterling, P. & Wickelgren, B. G. (1969). "Visual receptive fields in the superior colliculus of the cat", *Journal of Neurophysiology*, 32, 1-15
- [STR70] Strausfeld, N. J. (1970). "Golgi studies on insects. Part II: The optic lobes of Diptera", *Philosophical Transactions of the Royal Society London*, 258, 135-223
- [STR76] Strausfeld, N. J. (1976). *Atlas of an Insect Brain*, Berlin: Springer-Verlag
- [STR87] Strausfeld, N. J., Seyan, H. S. & Milde, J. J. (1987). "The neck motor system of the fly *Calliphora erythrocephala*. 1: Muscles and motor neurons", *Journal of Comparative Physiology A*, 160, 205-224
- [STR89] Strausfeld, N. J. (1989). "Beneath the compound eye: neuroanatomical analysis and physiological correlates in the study of insect vision", In Stavenga, D. G. & Hardie, R. C. (Eds), *Facets of Vision*, Berlin: Springer-Verlag
- [TAN04] Tan, S., Dale, J., Johnston, A. (2004) Effects of Inverse Perspective Mapping on Optic Flow. ECOVISION Workshop 2004 (Isle of Skye, Scotland, UK).
- [TOM91] Tomasi, C. and Kanade, T.. Detection and Tracking of Point Features. Carnegie Mellon University Technical Report CMU-CS-91-132, April 1991. pp.438-445.
- [TUR50] Turing, A. M. (1950)"Computing machinery and intelligence" *Mind* 50: 433-460.
- [VAN84] van Santen, J. P. H. & Sperling, G. (1984). "Temporal covariance model of human motion perception", *Journal of the Optical Society of America A*, 1, 451-473
- [VAN85] van Santen, J. P. H. & Sperling, G. (1985). "Elaborated Reichardt detectors", *Journal of the Optical Society of America A*, 2, 300-321
- [VAR02] Vargas-Martín, F. and E. Peli (2002). "Augmented-view for restricted visual field: multiple device implementations." *Optom Vis Sci* 79(11): 715-23.
- [VAR05] Vargas-Martín, F., Pelaez-Coca, M.D., Ros, E., Díaz, J., Mota, S. (2005): A generic real-time video processing unit for low vision. *Internacional Congreso Series* 1282, 1075-1079.
- [VEN92] Venkatesh S. and Kitchen L.J. (1992) Edge evaluation using necessary components. *CVGIP: Graphical Models and Image Processing*, 54(1):23-30.
- [VOL06] Volvo BLIS system. Web link: <http://www.mynrma.com.au/blis.asp> (último acceso en enero de 2007).
- [WAN95] Wandell, B. A. (1995). *Foundations of Vision*. Sinauer Associates, Sunderland, MA.
- [WIC99] Wicklein, M. & Varjú, D. (1999). "Visual system of the European hummingbird hawkmoth *Macroglossum stellatarum* (Sphingidae, Lepidoptera): Motion-sensitive interneurons of the lobula plate", *Journal of Comparative Neurology*, 408, 272-282
- [WOL94] Wolf-Oberhollenzer, F. & Kirschfeld, K. (1994). "Motion sensitivity in the nucleus of the basal

optic root of the pigeon", Journal of Neurophysiology, 71, 1559-1573

- [WYA75]** Wyatt, H. J. & Daw, N. W. (1975). "Directionally sensitive ganglion cells in the rabbit retina: specificity for stimulus direction, size, and speed", Journal of Neurophysiology, 38, 613-626
- [XIL06]** [http://www.xilinx.com/xlnx/xweb/xil\\_publications\\_display.jsp?iLanguageID=1&category=-18774&sGlobalNavPick=PRODUCTS&sSecondaryNavPick=Design+Tools](http://www.xilinx.com/xlnx/xweb/xil_publications_display.jsp?iLanguageID=1&category=-18774&sGlobalNavPick=PRODUCTS&sSecondaryNavPick=Design+Tools) (último acceso en Enero de 2007)
- [ZAA78]** Zaagman, W. H., Mastebroek, H. A. K. & Kuiper, J. W. (1978). "On the correlation model: performance of a movement detecting neural element in the fly visual system", Biological Cybernetics, 31, 163-168
- [ZAA83]** Zaagman, W. H., Mastebroek, H. A. K. & de Ruyter van Steveninck, R. R. (1983). "Adaptive strategies in fly vision: on their image processing qualities", IEEE Transactions on Systems, Man and Cybernetics, SMC-13, 900-906
- [ZAN99]** Zanker, J. M., Srinivasan, M. V. & Egelhaaf, M. (1999). "Speed tuning in elementary motion detectors of the correlation type", Biological Cybernetics, 80, 109-116