Cálculo de Disparidad y Segmentación de Objetos en Secuencias de Video

Tesis de Maestría
en Ingeniería Eléctrica

Federico Lecumberry Ruvertoni

Directores de Tesis:
Nicolás Pérez de la Blanca, ETSII, Universidad de Granada, España
Álvaro Pardo, IIE, Facultad de Ingeniería, Universidad de la República

Director Académico:
María Simón, IIE, Facultad de Ingeniería, Universidad de la República

Tribunal:
Gregory Randall, IIE, Facultad de Ingeniería, Universidad de la República
Andrés Almansa, INCO, Facultad de Ingeniería, Universidad de la República
Álvaro Gómez, IIE, Facultad de Ingeniería, Universidad de la República

Revisor Externo:
Guillermo Sapiro, ECE, Universidad de Minnesota, EE.UU.

Instituto de Ingeniería Eléctrica
Facultad de Ingeniería
Universidad de la República

Montevideo, Uruguay
3 de agosto de 2005

Esta tesis fue preparada en LaTeXutilizando fuentes de la familia *Computer Modern*.
Se utilizaron los siguiente paquetes: *afterpage*, *amsfonts*, *amsmath*, *amssymb*, *babel*, *calc*, *color*, *fancyhdr*, *graphicx*, *ifthen*, *inputenc*, *makeidx*, *pifont*, *psfrag*, *pstricks*, *rotating*, *subfigure*, *textcomp*, *totpages*, *url* y *xspace*

Tiene un total de 204 páginas, 56 figuras y 5 tablas.

Versión del 24 de octubre de 2005 incluyendo las correcciones del tribunal.

`http://iie.fing.edu.uy/~fefo/tesis/`

Acta de Defensa / Tesis de Maestría

(Transcripción del Acta de Defensa de la Tesis de Maestría)

- Fecha: 3 de agosto de 2005
- Lugar: Montevideo, Facultad de Ingeniería - Universidad de la República
- Plan de Estudio: Maestría en Ingeniería (Ingeniería Eléctrica)
- Aspirante: Federico Lecumberry
- Documento de Identidad: 2.769.975-7
- Director/es de Tesis: Prof. Nicolás Pérez de la Blanca Capilla (Universidad de Granada, España); Dr. Ing. Álvaro Pardo
- Tribunal:
  - Ing. Prof. María Simon
  - Dr. Ing. Andrés Almansa
  - Prof. Adj. Álvaro Gómez
  - Dr. Ing. Gregory Randall

Los miembros del Tribunal hacen constar que en el día de la fecha el Sr. Ing. Federico Lecumberry ha sido APROBADO en la defensa de su Tesis de Maestría titulada: "Cálculo de disparidad y segmentación de objetos en secuencias de video".

La resolución del Tribunal se fundamenta en los puntos detallados a continuación: En el día de la fecha Federico Lecumberry ha defendido públicamente su tesis de Maestría titulada "Cálculo de disparidad y segmentación de objetos en secuencias de video".

En la misma se realiza un estudio profundo y documentado sobre dos aspectos importantes en tratamiento de imágenes por computador: el cálculo de disparidad en imágenes estereoscópicas y la segmentación de secuencias de video. Este tribunal hace suyas las opiniones vertidas por el revisor externo, Prof. Guillermo Sapiro, en el documento que se adjunta, donde se valora de manera muy positiva el trabajo presentado. El trabajo está excelentemente redactado, constituyéndose por momentos en un posible material de estudio, tal como lo señala el revisor externo en su informe. La defensa pública fue realizada con gran claridad, generando el ambiente propicio, por la solvencia demostrada, para propiciar una discusión de gran nivel, donde preguntaron tanto los miembros del tribunal examinador como el público presente. Por todas estas razones valoramos la Maestría de Federico Lecumberry como un excelente trabajo.

Para que conste, (siguen firmas) María Simón, Andrés Almansa, Álvaro Gómez y Gregory Randall

A modo de Prólogo

(Transcripción de la Nota del Revisor Externo Dr. Guillermo Sapiro)

Julio 25, 2005

Evaluación de Tesis de Maestría de Federico Lecumberry.

Es con una gran placer que evalúo esta tesis. Presentaré a continuación mi opinión, dando razones para aprobar esta tesis. La tesis comienza (Capítulo 2 de primera parte) con una muy buena descripción del problema en referencia al calculo de disparidad en imágenes estéreo. Descripciones como la básica presentación de disparidad son tan claras que pueden ser usadas como material de curso. Los pasos necesarios para computar disparidad son presentados también con mucha claridad. Las posibles dificultades a atacar son claramente presentadas. Esta descripción muestra un claro entendimiento del problema, lo que se manifiesta a lo largo de toda la tesis.

El Capítulo 3 continua con una descripción fundamental de la literatura en el área de calculo de disparidad. Las diferentes metodologías clásicas (block matching, level-set, etc.) y los algoritmos computacionales utilizado son claramente presentados. Esta sección cubre técnicas de optimización que son fundamentales en procesamiento de imágenes, como programación dinámica y corte de grafos. Conocimiento y domino de esta técnica es fundamental. Cabe destacar que el autor se vera interesado en resultados recientes en esta dirección formulados por D. Kirasanov and S. J. Gortler (trabajo relacionado a este por S.J. Gortler es comentado por el autor). La descripción en esta sección concluye con una enumeración de problemas extremos en el calculo de disparidad.

En el Capítulo 4 se describen en detalle los algoritmos que el autor implementó y comparó. Los algoritmos son descritos con una visión critica, presentando las ventajas y problemas que ellos presentan. Posibles soluciones a algunos de los problemas (por ejemplo búsqueda de coherencia) son propuestos e implementados.

El Capítulo 5 concluye y da nuevamente una visión crítica de las técnicas testeadas. Puntos importantes, como la influencia de oclusiones, son estudiados. Esta sección demuestra que no solo el autor realizo una gran cantidad de trabajo, pero que también adquirió un completo domino del tema de disparidad y estéreo en particular y de técnicas cono cortes de grafos en general. Esta parte de la tesis es sin lugar a dudas muy buena y muy completa.

Las segunda parte de la tesis se refiere a la segmentación de objetos. Nuevamente el autor comienza con una descripción del problema y de los métodos estudiados. El tema de segmentación es mucho mas amplio y variado que el de estero, y su cobertura absoluta es virtualmente imposible en una tesis o un libro. El autor por lo tanto, coherentemente, se concentra en presentar las bases necesarias para los estudios que el realizo. Acá describe la teoría de Bayes, nuevamente estudiando un marco cuya importancia va mucho mas del problema encarado en esta parte de la tesis. Esto le da un valor global al trabajo y al aprendizaje. Fundamental es también el estudio planteado en la Sección 7.2, la combinación de clasificadores, uno de los temas mas importantes del momento en el estudio de imágenes. Aquí le interesará al autor mirar resultados recientes relacionados a "producto de expertos"(trabajo considerado luego en el Capítulo 10), además de los fundamentales que el ya describe. Los modelos de mezclas Gaussianas y de "kernels"que el describe nuevamente muestran que el autor esta al corriente de las areas mas fundamentales del procesamiento de imágenes. Esta tesis se puede sin lugar a duda usar como base para un curso introductorio de métodos matemáticos y computacionales básicos y fundamentales para procesamiento de imágenes!

El Capítulo 8 describe conceptos básicos de color, un tema poco estudiado en procesamiento de images en el medio académico, y muy fundamental.

El Capítulo 9 se concentra en presentar literatura relacionada a la segmentación de objetos en video. Esta sección posee un nivel de descripción similar al muy alto demostrado anteriormente. En esta sección muestra el uso de muchos de los conceptos que introduce anteriormente, por ejemplo, modelado por mezclas de Gaussianas. En la descripción de literatura, una clara clasificación de los diferentes métodos es presentada, con ejemplos particulares para cada uno.

El Capítulo 10 presenta un nuevo algoritmo para segmentación de objetos en video, basado en teorías de detección de patrones. Previos capítulos presentaron descripciones de algoritmos existentes (con posibles modificaciones sugeridas por el autor). Este capítulo es novedoso. La técnica esta claramente descrita en la Figura 10.1, y contiene pasos muy interesantes y novedosos. Estos pasos combinan muchas de las técnicas anteriormente descritas (tratamiento de espacios de color, mezcla de expertos, kernel methods, etc).

La tesis culmina con propuestas de futuro trabajo y apéndices con detalles matemáticos.

Para concluir: La tesis es muy completa, presenta problemas fundamentales en procesamiento de images así como técnicas muy importantes. Es claro que el autor ha dominado el area, y ha aprendido técnicas con aplicaciones mucho mas profundas que las ya muy importantes acá atacadas. La tesis se complementa con propuestas novedosas para segmentación de objetos en video. Es indudable que el autor ha trabajado muchísimo en esta Tesis, ha aprendido muchísimo, y ha propuesto nuevas direcciones. Es por esto que recomiendo que esta Tesis sea aprobada.

Sinceramente, Guillermo Sapiro.

A la memoria de mi padre,
Jorge M. Lecumberry Garmendia.

Agradecimientos

A mis directores: María Simón, Álvaro Pardo y Nicolás Pérez de la Blanca, por disponer generosamente de su tiempo y conocimiento para guiarme por el difícil camino de la investigación académica.

Al tribunal: Gregory Randall, Andrés Almansa, Álvaro Gómez, y a Guillermo Sapiro también, por sus valiosos comentarios que aportaron mucho a esta tesis.

A la «Dirección Nacional de Ciencia y Tecnología» (DINACYT) en su Programa de Desarrollo Tecnológico (PDT), y a las empresas «Movicom Bellsouth» y «Tecnocom», por el apoyo económico imprescindible a lo largo de este estudio.

A los integrantes del «Grupo Multimedia» (GMM) y del «Grupo de Tratamiento de Imágenes» (GTI) del IIE, por brindar el entorno de conocimiento y diálogo crítico, y apoyo continuo (y discreto).

Al personal de las Bibliotecas de Facultad de Ingeniería por los esfuerzos para conseguir las publicaciones que permitieron culminar esta tesis.

A la familia (la personal y la del IIE) y los amigos por la comprensión en los tiempos dedicados a esta tesis.

A Julia.

Tabla de contenidos

Resumen

En la primera parte de esta tesis se realiza un estudio de los algoritmos de cálculo de disparidad. Se realiza una revisión bibliográfica del área y se analizan diferentes alternativas. Se presenta la comparación de dos algoritmos de cálculo de disparidad muy citados en la bibliografía. Uno de ellos se basa en la técnica de corte de grafos siendo uno de los algoritmos con mejores resultados reportados en la bibliografía. El otro se basa en el método de Programación Dinámica y es un algoritmo muy citado en el área. Se muestran ejemplos donde el algoritmo basado en corte de grafos presenta algunas limitaciones, mostrando no ser la mejor solución en todos los casos.

En la segunda parte se presenta un algoritmo de segmentación de objetos en secuencias de video. Este algoritmo implementa un esquema de clasificación simple basado en múltiples características (color, posición, movimiento) junto con una etapa de difusión de probabilidades. Se presenta una variante al esquema clásico de difusión de probabilidades incorporando información espacial de la imagen. También se presenta una segmentación donde se incorpora información de profundidad de la escena mediante el cálculo de la disparidad, mostrando que se mejoran los resultados obtenidos con el algoritmo planteado inicialmente.

Ambas partes de esta tesis han sido presentadas y aceptados como trabajos separados en el *III Workshop de Computación Gráfica, Imágenes y Visualización* (WCGIV) del *XI Congreso Argentino de Ciencias de la Computación* (CACIC 2005).

Palabras claves disparidad, estéreo, epipolar, objetos, video, segmentación.

Prefacio

La presente tesis es el producto del trabajo en mis estudios de Maestría en Ingeniería Eléctrica del programa de Posgrados de la Facultad de Ingeniería. La misma se divide en dos partes, en principio independientes, que corresponden a cada mitad del título de la misma. La primera parte, entonces, trata sobre el cálculo de disparidad en imágenes estéreo, y la segunda sobre la segmentación de objetos en secuencias de video.

La parte sobre el cálculo de disparidad en imágenes estéreo es el resultado del trabajo tutoreado por el Dr. Nicolás Pérez de la Blanca de la Universidad de Granada, España, que se realizó en dos estadías en el Departamento de Ciencias de la Computación e Inteligencia Artificial (DECSAI) de la Escuela Técnica Superior de Ingeniería Informática (ETSII) de dicha Universidad. La primera tuvo lugar de febrero a marzo de 2003 y fue financiada por el Dr. Pérez de la Blanca. En agosto de 2004 se me otorgó una beca del Programa de Desarrollo Tecnológico (PDT S/C/BE/33/04) para realizar una segunda estadía en la Universidad de Granada, la que transcurrió desde noviembre de 2004 a enero de 2005. Durante la misma di por finalizado el estudio del cálculo de disparidad comenzado en la estadía anterior y continuado en el Instituto de Ingeniería Eléctrica (IIE), y redacté la primera parte de esta tesis.

En junio de 2004 comenzó a ejecutarse en el IIE el Proyecto PDT «Análisis de Video» (S/C/OP/17/07) dirigido por el Dr. Álvaro Pardo, en el cual participo en tareas de investigación como docente del Grupo Multimedia del IIE. En esta nueva circunstancia de estudio de un tema se replantearon los objetivos de mi tesis, recortando los alcances de la primera parte, e incluyendo el estudio de la segmentación de objetos en secuencias de video de este proyecto como segunda parte, tutoreado por el Dr. Pardo. Este proyecto tiene fecha de culminación abril de 2006.

En `http://iie.fing.edu.uy/~fefo/tesis/` se encuentra esta tesis en formato digital junto con los ejemplos presentados.

Federico Lecumberry

Parte I

Cálculo de disparidad

1 Introducción

El fenómeno de la visión estéreo ya era conocido en los tiempos de Leonardo da Vinci [1], quien en su «*Trattato della Pittura*» de 1651 presenta varias observaciones sobre los efectos que suceden cuando se observa un objeto con cada ojo por separado y los efectos que aparecen cuando se observa con ambos ojos (visión binocular). En 1838 Sir Charles Wheatstone [2] comenta que Leonardo realizó sus estudios con una esfera y que si lo hubiera hecho con un cubo, además, habría observado que el objeto presenta una forma diferente a cada ojo. Wheatstone presenta varias observaciones y experimentos sobre la visión binocular.

Dada la posición de los ojos en los humanos y la forma de moverlos las imágenes que se reciben en cada ojo son prácticamente iguales, con una diferencia en la posición relativa de los objetos. Estas diferencias relativas en la posición en cada imagen (la disparidad), tiene una relación directa con la distancia (profundidad) a la que se encuentran los objetos entre sí, y al observador. El cerebro es capaz de interpretar esa diferencia y reconstruir la estructura de la escena que ve el observador. Según Marr y Poggio [3] existen tres etapas en el proceso de recuperación de la estructura de una escena. Estas son, primero, seleccionar un punto característico de un objeto en una de las imágenes (vistas por cada ojo), segundo, encontrar el mismo punto característico en la otra imagen, y tercero, medir la diferencia relativa (disparidad) entre la posición de estos dos puntos.

En las últimas tres (casi cuatro) décadas el tema de la visión estéreo ha sido abordado por la comunidad de Computer Vision. Se llama visión estéreo a la capacidad de recuperar la estructura tridimensional de una escena a partir de, por lo menos, dos vistas o imágenes diferentes de la misma. La estructura que se recupera es la posición de los objetos presentes en la escena, fundamentalmente recuperando la profundidad (distancia al observador) de los objetos. Una formulación alternativa de este problema puede ser la de localizar para *cada punto* de cada una de las imágenes su correspondiente en la otra imagen; entendiendo por puntos correspondiente aquellos que son

Figura 1.1: **Imágenes estéreo.** *Imágenes izquierda y derecha de la misma escena con un desplazamiento horizontal (hacia la derecha) de la cámara. [4]*

proyecciones del mismo punto del espacio en cada una de las imágenes (ver figura 2.1, página 7). De esta forma se recupera una imagen *densa* de profundidades para cada uno de los puntos de la escena proyectados en ambas imágenes

La figura 1.1 muestra un par de imágenes de una escena tomadas con un pequeño desplazamiento horizontal como serían vistas por el ojo izquierdo y derecho, respectivamente.

Lograr que una computadora «pueda ver» es un desafío en la comunidad de Computer Vision desde sus principios y que aún no ha sido resuelto completamente, más allá de lo que se entienda por «pueda ver» en el caso de una computadora. Existen varias dificultades que se plantean cuando se intenta abordar este problema utilizando una computadora como herramienta de procesamiento; por ejemplo, la adquisición de las imágenes y la preparación para su tratamiento, el ruido en la adquisición, diferencias de intensidad/color del mismo punto del espacio en las dos imágenes, las oclusiones y complejidad de la escena, etc. Sin embargo muchos avances se han realizado logrando resultados importantes en diversas aplicaciones.

Una de las aplicaciones de una imagen *densa* de profundidades de la escena es la descomposición de una imagen en capas de igual profundidad para su posterior procesamiento y generación de nuevas vistas (*Image Based Rendering*). Se utiliza en la reconstrucción tridimensional de un objeto a partir de varias vistas o una secuencia de video. También en la navegación de robots, creación de realidad virtual, codificación de imágenes estéreo seguimiento y vigilancia (conteo de personas), etc.

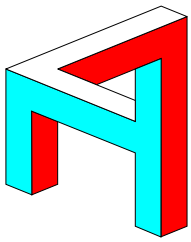
En la naturaleza. En los animales la capacidad de recuperar la estructura tridimensional está dada por la existencia de dos sensores, normalmente los ojos, aunque también pueden ser los oídos como el caso de los búhos y murciélagos. A partir de las «imágenes» obtenidas por cada uno de los sensores, entre los cuales debe existir una distancia espacial, es posible recuperar la geometría tridimensional. Esto no es igual en todos los animales, a pesar de tener dos ojos. Animales con una visión lateral, debida a tener los ojos a los lados del cuerpo, no tienen la misma capacidad para recuperar la estructura tridimensional, que los animales con ambos ojos al frente. Normalmente los primeros, son animales que deben protegerse de los predadores, que son los segundos. Para un predador es fundamental poder medir exactamente la distancia a una presa para poder hacer el ataque justo; mientras que para los otros es necesario mantener una vigilancia periférica para poder detectar movimientos que puedan significar peligro. Por esto es importante que los predadores tengan un buen sistema binocular estéreo, preparado para tareas específicas con el fin de la supervivencia. Los animales con visión periférica igualmente logran recuperar la estructura tridimensional que los rodea, pero sin la precisión que logran los animales con visión frontal. Por otro lado, los animales con visión frontal obtienen una visión periférica a partir de la visión frontal con movimientos de la cabeza, con un *cuello* más desarrollado.

En los humanos, la percepción visual de la estructura tridimensional es realizada por el Sistema Visual Humano (SVH) con los ojos y el cerebro. El aprendizaje (o adaptación) permite poder realizar tareas sencillas a pesar de tener anuladas o debilitadas las capacidades de visión estéreo. Por ejemplo, agarrar una taza con un solo ojo abierto. La precisión obtenida no es la misma pero igual se pueden realizar tareas básicas. Igualmente, es posible engañar la percepción que se obtiene de algunas imágenes, cuando el cerebro intenta recuperar una estructura tridimensional a partir de un dibujo plano, ver figura 1.2. [5]

1.1. Estructura de la Parte I

En el presente trabajo se realizó una revisión de la bibliografía del área; se implementó y estudió en profundidad un algoritmo clásico, y se hicieron experimentos de comparación con un segundo algoritmo que, al momento de escribir esta tesis, obtiene los mejores resultados según la bibliografía consultada.

En el capítulo 2 se hace una presentación geométrica de la relación entre la disparidad y la profundidad de los objetos en la escena, y del uso de la misma para la reconstrucción euclídea de la estructura de una escena



(a)

(b) *http://www.lordoftherings.net/legend/gallery/images/rivendell/rivendell9.jpg (Visitada el 2004-11-12)*

Figura 1.2: **Ilusión óptica.** (a) *Se intenta reproducir una estructura tridimensional a partir de una dibujo plano, con las incongruencias visibles.* (b) *Efecto visual de la película The Lord of the Rings: The Fellowship of the Ring [6]: La profundidad a la que se encuentran los personajes es diferentes generando la relación de tamaños visible, a pesar de no ser así en la realidad; al tener una visión monocular de la escena no se puede detectar la diferencia de profundidades (y la disparidad).*

tridimensional. En la sección 2.1 se presenta la estructura clásica y las consideraciones de los algoritmos estéreo aplicados al problema del cálculo de disparidad. En el capítulo 3 se hace una revisión de la bibliografía consultada. En el capítulo 4 se presentan los algoritmos estudiados. En el capítulo 5 se presentan los experimentos realizados, la discusión de los resultados y conclusiones del trabajo. En los apéndices B y C se presentan breves introducciones a los dos principales métodos usados por los algoritmos estudiados, Programación Dinámica y Corte de Grafos. En el apéndice A se presenta una introducción a la geometría epipolar.

2 La disparidad

Una forma de estimar la profundidad de cada uno de los puntos en la escena es mediante el cálculo de la disparidad entre las imágenes de la misma. Asumiremos que la escena es estática, es decir, los objetos visibles en la escena no cambien su posición en la misma ni sufren deformaciones.

Para definir la disparidad asumamos una configuración de dos cámara de características similares, como la que se muestra en la figura 2.1. Estas dos cámaras forman un par estéreo, y asumiremos que cada una de ellas cumplen un modelo *pinhole*. Los ejes ópticos de las cámaras son paralelos, $\overrightarrow{O_R o_R} \parallel \overrightarrow{O_L o_L}$. Ambas cámaras tienen la misma distancia focal, f, con centros O_L y O_R separados una distancia B, llamada línea base (*baseline*), de forma que las imágenes que se forman, I_L e I_R, estén en planos paralelos. De esta manera la línea base es paralela a la coordenada x de las imágenes. Con el modelo *pinhole* considerado, un punto en el espacio tridimensional P, con coordenadas $(X, Y, Z)^\top$, se proyecta en cada una de las imágenes bidimensionales en los puntos p_L y p_R, con coordenadas $(x_L, y_L)^\top$ y $(x_R, y_R)^\top$, respectivamente.

El plano que contiene a los puntos P, O_L y O_R, interseca a las imágenes en dos rectas e_L y e_R. Dada la configuración del par estéreo, éstas son *rectas epipolares* entre sí, o sea, un punto, p_L, en la recta e_L de la imagen I_L tiene

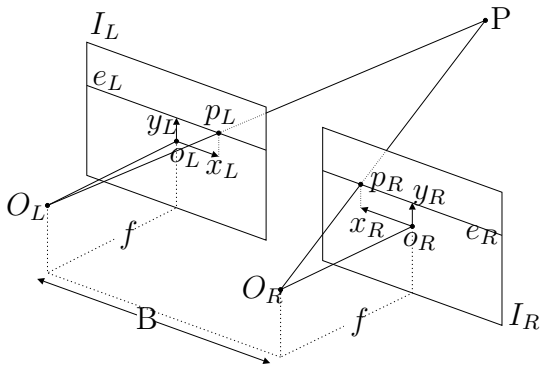


Figura 2.1: *Configuración de las cámaras del par estéreo.*

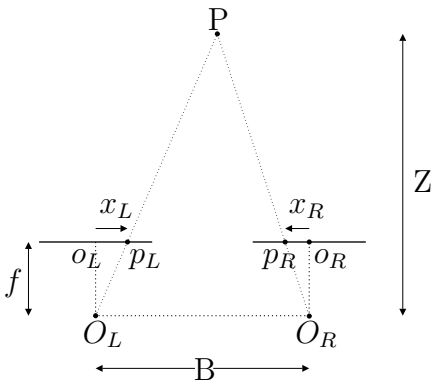


Figura 2.2: *Relación geométrica entre los parámetros del par estéreo para obtener la profundidad Z a partir de la disparidad d.*

su correspondiente en algún punto de la recta e_R. Esto reduce la búsqueda del correspondiente de p_L de toda la imagen I_R a la recta e_R.

En la figura 2.2 podemos ver cómo se relacionan los parámetros definidos en el par estéreo, que permiten obtener la relación entre la disparidad d y la profundidad Z del punto P.

La disparidad es la diferencia en las coordenadas horizontales de los puntos p_L y p_R, o sea, $d = x_L - x_R$. Dependiendo el sistema de referencia utilizado en las imágenes, la definición puede cambiar de forma que el signo sea siempre positivo. Las coordenadas de p_L y p_R quedan relacionadas mediante:

$$\begin{cases} x_L = x_R + d \\ y_L = y_R \end{cases} \tag{2.1}$$

Considerando los triángulos $P\overset{\triangle}{O_L}O_R$, $p_R\overset{\triangle}{o_R}O_R$ y $p_L\overset{\triangle}{o_L}O_L$, utilizando semejanza entre triángulos se llega a:

$$d = \frac{f}{Z}\, B \tag{2.2}$$

Entonces, se tiene la relación entre d y Z:

$$d \propto \frac{1}{Z} \tag{2.3}$$

Basados en la ecuación (2.3) podemos recuperar, a menos de una constante de escala, la profundidad de cada píxel en cada una de las imágenes a partir de la disparidad calculada.

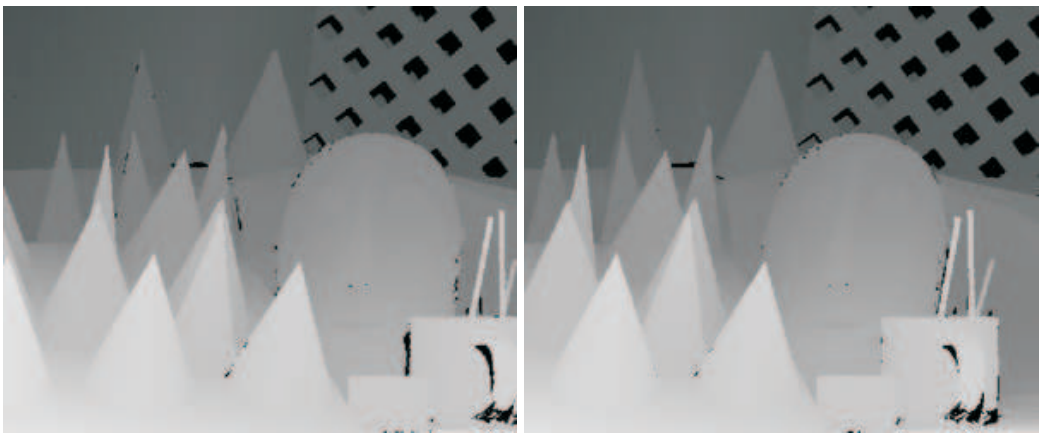


Figura 2.3: **Mapa de disparidad reales** *para las imágenes de la figura 1.1, el nivel de gris es proporcional a la disparidad e inversamente proporcional a la profundidad. En los puntos negros la disparidad es desconocida. [4]*

La relación de proporcionalidad inversa planteada en la ecuación (2.3) es fácilmente verificable observando alternadamente las imágenes izquierda y derecha, y notando que los objetos más cercanos a la cámara –menor Z– tienen mayor desplazamiento relativo –mayor d– en las imágenes (ver figura 1.1).

Los algoritmos de cálculo de disparidad obtienen una imagen con el valor de disparidad calculado en cada punto de las imágenes de entrada. Estas imágenes se conocen como mapas de disparidad; en la figura 2.3 se ven los mapas de disparidad *reales* de las imágenes de la figura 1.1. Los mapas de disparidad *reales* son obtenidos por métodos en los cuales otro tipo de información se agrega a las imágenes que permiten obtener la profundidad correcta; un ejemplo de estos procedimientos puede verse en el trabajo de Scharstein y Szeliski [7] basado en iluminar la escena con *luz estructurada*.

Algunos puntos de la escena son visibles en una sola de las imágenes de entrada, o sea se proyectan en una sola de las imágenes. En estos puntos se producen oclusiones por la disposición de los objetos en la escena y la disparidad no puede ser calculada por este método.

La proyección con el modelo pinhole considerado, que lleva de las coordenadas del punto en el espacio $P = (X, Y, Z)^\top$ al punto del plano $p = (x, y)^\top$, no es una transformación lineal. Para obtener una transfor-

mación lineal cerrada[1] se hace una extensión de la geometría euclídea a la geometría proyectiva, introduciendo las coordenadas homogéneas, añadiendo una nueva coordenada 1. La representación en coordenadas homogéneas para P y p quedan $P_h = (X, Y, Z, 1)^\top$ y $p_h = (x, y, 1)^\top$ respectivamente. Con esta nueva representación las propiedades del punto $(k\,x, k\,y, k)^\top$ para cualquier $k \neq 0$ son iguales, y todas se considera representantes del punto $p = (x, y)^\top$.

Con el agregado de esta nueva coordenada es posible definir un mapeo lineal entre las coordenadas homogéneas de un punto del espacio P_h y las coordenadas homogéneas del punto de la imagen p_h su proyección mediante el modelo de la cámara. Esta transformación lineal puede representarse mediante un producto matricial con la *matriz de la cámara* $\mathbf{P_C}$ [8]:

$$p_h = \mathbf{P_C}\ P_h \tag{2.4}$$

La forma que toma $\mathbf{P_C}$ varía dependiendo el tipo transformación que se considere (afín, proyectiva, etc.). La forma más sencilla que toma en el caso del modelo de cámara considerado es

$$\mathbf{P_C} = \begin{pmatrix} \alpha_u & 0 & u_0 & 0 \\ 0 & \alpha_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Donde $\alpha_u = f\,k_u$, $\alpha_v = f\,k_v$ y (u_0, v_0) son los parámetros intrínsecos de la cámara[2]. Escribiendo la ecuación (2.4), con $k_u = k_v = 1$ y $(u_0, v_0) = (0, 0)$ se obtiene la relación entre las coordenadas horizontales y verticales en la imagen a partir de las coordenadas del punto en el espacio $(X, Y, Z)^\top$ y la distancia focal de la cámara (en píxeles):

$$\begin{pmatrix} k\,x \\ k\,y \\ k \end{pmatrix} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Junto con la ecuación (2.2) dan las ecuaciones para recuperar la estruc-

[1] Al ser una transformación cerrada, la concatenación de transformaciones es otra transformación.

[2] k_u y k_v son los factores de escala horizontales y verticales del sensor de la cámara, mientras (u_0, v_0) son las coordenadas del centro de la proyección en el sistema de coordenadas de la imagen

tura de la escena, a menos de una constante,

$$\begin{cases} x = \dfrac{f}{Z}\, X \\ y = \dfrac{f}{Z}\, Y \\ d = \dfrac{f}{Z}\, B \end{cases} \tag{2.5}$$

Estas ecuaciones pueden verificarse geométricamente en la figura 2.1.

Si se conocen los parámetros intrínsecos de la cámara y la configuración geométrica del par estéreo (f y B en el caso más simple) es posible recuperar la profundidad de cada punto de la imagen, y por lo tanto la estructura euclídea de la escena.

2.1. Algoritmos para el cálculo de disparidad

El objetivo de un algoritmo estéreo de cálculo de disparidad es obtener la información de profundidad en *todos* los píxeles de las imágenes del par estéreo; esto es una imagen *densa* de disparidades. A partir de esta información se puede obtener la estructura de la escena, recuperando la posición tridimensional de los puntos proyectados a las imágenes.

Entonces, estos algoritmos deben determinar la correspondencia de todos los puntos entre ambas imágenes. Para hallar la correspondencia se basa en características «visuales» (intensidad/color, geometría[3], movimiento, u otras).

Los pasos que debería resolver un algoritmo estéreo de cálculo de disparidad son, a grandes rasgos los siguientes:

1. calibrar las cámaras,
2. rectificar las imágenes,
3. hallar la correspondencia entre *todos* los puntos de las imágenes, y
4. reconstruir la estructura de la escena

Este último paso no es necesario para calcular la disparidad, pues ya fue hecho en el paso anterior, pero es donde se hace uso de la misma, para recuperar la posición tridimensional de los puntos visibles.

[3]Se entiende por geometría de un punto como una propiedad de la geometría local en el entorno del punto, por ejemplo, pertenecer a un borde, ser una esquina de un objeto, etc.

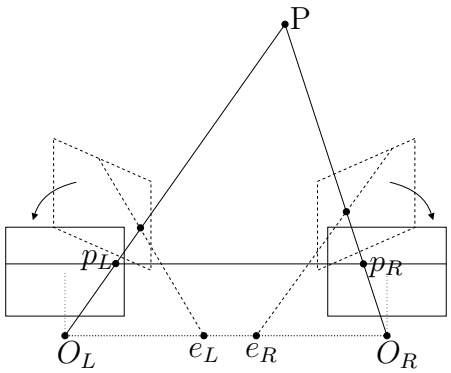


Figura 2.4: **Rectificación.** *Luego de la rectificación las imágenes de la escena quedan paralelas a la recta que une los centros de las cámaras; y los puntos correspondientes se encuentran en la misma fila de cada imagen.*

El proceso de calibración, consiste en determinar los parámetros intrínsecos (matriz $\mathbf{P_C}$) y extrínsecos (posición relativa) de las cámaras del par estéreo. En este proceso se determinan la correspondencia entre algunos puntos relevantes por sus características (puntos esquinas, color, etc.), y se ajustan los parámetros de las cámaras para cumplir con estas correspondencias. En este proceso se determina la posición tridimensional de los puntos seleccionados, pero no se logra una correspondencia para todos los puntos (densa) de las imágenes de entrada.

La rectificación de las imágenes consiste en proyectar las imágenes del par estéreo, de forma que los planos de las imágenes en cada cámara sean paralelos entre sí, y paralelos a la dirección en la cual existe el desplazamiento entre las imágenes. Luego de la rectificación las imágenes de la escena quedan en posición «fronto–paralela» como se muestra en la figura 2.4. Con esta configuración se simplifica la búsqueda de los puntos correspondientes pues se asegura que el correspondiente de un punto con coordenada vertical y_L en la imagen izquierda, se encuentra en la fila de coordenada $y_R = y_L$ de la imagen derecha, que se denomina *scanline*. La solución al problema de la rectificación de las imágenes de un par estéreo ha sido ampliamente tratado y existen soluciones en la literatura [9]. Lo importante es que siempre es posible hacer la rectificación de las imágenes llevándolas a la configuración de la figura 2.1.

La calibración y la rectificación no son abordados en esta tesis. Para referencias sobre algoritmos para estos métodos, así como para una introducción (y profundización en la geometría de múltiples vistas –geometría epipolar–) se recomiendan los libros de Hartley y Zisserman [8] y de Faugeras y otros [9].

Los algoritmos de cálculo de disparidad estudiados asumen que las imágenes con que se trabaja están rectificadas. Para garantizar esta hipótesis, las imágenes de la escena que se utilizan para la prueba de estos algoritmos, se adquieren realizando un desplazamiento horizontal de la cámara. Debido que las imágenes se adquieren en instantes diferentes es necesario que las escenas sean estáticas.

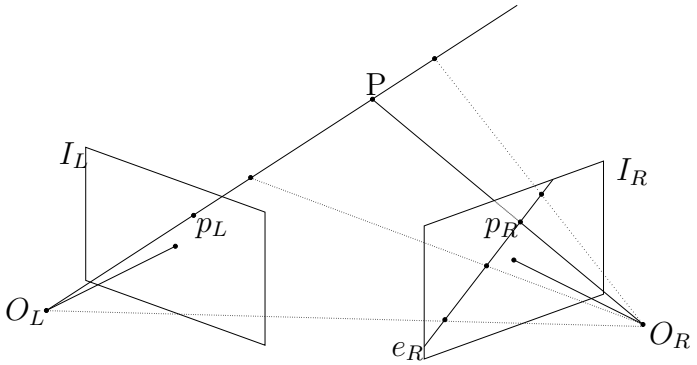
El problema de hallar la correspondencia consiste en encontrar qué punto de la imagen izquierda se corresponde con qué punto de la imagen derecha, es decir, encontrar los puntos en cada imagen que son proyección del mismo punto del espacio, para todos los puntos en cada una de las imágenes. De esta forma se recupera un mapa *denso* de disparidades. Este es sin duda el proceso que presenta mayor complejidad, y para el cual los algoritmo asumen ciertas hipótesis sobre la escena y la estructura que restringen las opciones a considerar en la búsqueda de los correspondientes.

Restricciones. Existen restricciones (hipótesis) que se aplican generalmente en los algoritmos para hallar los puntos correspondientes. Estas restricciones están basadas en propiedades físicas y geométricas razonables de los objetos y superficies presentes en la escena, y su relación. Las restricciones comúnmente mencionadas en la literatura son:

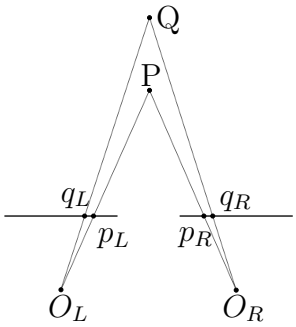
- *Restricción epipolar.* Viene dada por la geometría epipolar del par estéreo, e implica que el correspondiente de un punto en una imagen debe estar en la recta epipolar del punto en la otra imagen. Esta restricción reduce la búsqueda del correspondiente de toda la imagen a una recta en la misma [8, 9].

- *Restricción de orden.* Implica que si la proyección del objeto Q está a la izquierda de la proyección del objeto P en la imagen izquierda, entonces la proyección de Q estará a la izquierda de la proyección de P en la imagen derecha. Un caso que no cumple con esta restricción es por ejemplo dos árboles a distintas profundidades como se muestra en la figura 2.5(b).[4]

[4]Observar un lápiz en posición vertical, sujeto con la mano y detrás otro objeto «fino», por ejemplo una columna, tampoco cumple esta restricción; dependiendo en cual de los objetos se enfoca la vista (lápiz o columna) el otro objeto se representa como apareciendo dos veces en la escena o presentando una transparencia inexistente. Esto es un ejemplo que el SVH presenta ambigüedades al recuperar la estructura correcta cuando no se cumple esta restricción.



(a) *Recta epipolar del punto P en la imagen derecha, el correspondiente de p_L en I_R debe estar sobre e_P.*



(b) *Escena que no cumple la restricción de orden.*

Figura 2.5: **Restricciones.** (a) *Epipolar.* (b) *De orden.*

- *Restricción de unicidad.* Implica que cada punto de una imagen puede tener no más de un correspondiente en la otra imagen. Esta restricción contempla que pueda no existir ningún correspondiente, como puede ser en el caso que esté oculto en la otra imagen [3].

- *Restricción de semejanza.* Implica que las características de los puntos en una imagen (intensidad o color, etc.) no debe cambiar mucho entre ambas imágenes.

Oclusiones. Otro fenómeno a tener en cuenta en el planteo de la visión estéreo y que influye en el proceso de hallar los puntos correspondientes, son las *oclusiones*. Las oclusiones son las regiones que se ven en una imagen y que no se ven en la otra imagen por estar tapadas por un objeto (visibles por ejemplo en la figura 1.1). Las oclusiones siempre implican una discontinuidad en la profundidad de la escena y son fuente de errores en muchos algoritmos

estéreo. Sin embargo estas regiones ocultas pueden ser usadas para la recuperación de la estructura de la escena y dan información fundamental de ésta.

Los algoritmos estéreo han abordado de varias formas el problema de la oclusión, Brown y otros [10] clasifican los algoritmos según cómo hacen este abordaje en: (i) algoritmos que detectan las oclusiones, (ii) algoritmos que disminuyen la sensibilidad a las oclusiones, y (iii) los algoritmos que modelan las oclusiones. Los algoritmos que detectan las oclusiones generalmente lo hacen después de hallar los correspondientes, buscando los *outliers* en los mapas de profundidad estimados. Los *outliers* no siempre se deben a oclusiones por lo que esto puede llevar a la detección de oclusiones que no lo son. Otra forma de detectar las oclusiones se basa en que, generalmente, las discontinuidades en la profundidad se producen por la superposición de los objetos en la escena y se da en los bordes de los mismos, lo que lleva a incluir detectores de borde en los algoritmos estéreo.

Los algoritmos que disminuyen la sensibilidad se basan en métodos robustos para la medida de semejanza, que disminuyen los *outliers*. Por ejemplo, marcando los píxeles de la ventana de correlación que tiene un nivel de gris mayor que el píxel central, y comparando cómo se distribuyen dentro de la ventana respecto del pixel central (*Census transformation*):

156	151	149
148	150	151
147	153	149

$\rightarrow 11001010$

En este caso se recorren las filas de arriba a abajo de izquierda a derecha omitiendo el pixel central.

Los algoritmos que modelan las oclusiones, integran las oclusiones y las restricciones que implican (restricción de orden) en el proceso de búsqueda de los correspondientes. Bobick e Intille [11] lo hacen asignando un costo mayor a los puntos que están ocultos (ver capítulo 4). Belhumeur [12] define un modelo bayesiano para estimar la geometría de la escena; proponiendo tres modelos de la geometría de la escena. En cada nuevo modelo la complejidad de la estructura de la escena va en aumento.

La utilización de una tercera imagen (o más) permite reducir ambigüedades existentes con sólo dos imágenes, pudiendo reducir el error de los puntos correspondientes. Además permite atacar el problema de las oclusiones, si puntos que están ocultos en una de las imágenes se proyecta en otras dos. Dentro de estos trabajos citamos los de Kolmogorov y Zabih [13] y Roy y Cox [14]

3 Revisión bibliográfica

El cálculo de la disparidad ha sido abordado desde diferentes puntos de vista. Muchos son los algoritmos y las técnicas muy diferentes para lograr calcular un mapa de disparidad aproximado. En los últimos dos años ha habido publicaciones que resumen, clasifican y ponen al día el estado del arte en los algoritmos estéreo [10] y plantean una plataforma de comparación de los distintos métodos existentes [15].

Brown y otros [10] presentan una clasificación de los algoritmos de cálculo de disparidad según cómo utilizan la información de las imágenes para encontrar los puntos correspondientes en las *scanlines*. La clasificación más general es en aquellos que utilizan restricciones *locales* en una ventana alrededor del punto en cuestión, y aquellos que imponen restricciones *globales* en la *scanline* o la imagen. Los métodos locales son muy eficientes computacionalmente pero presentan dificultades en regiones localmente ambiguas de las imágenes, como son las oclusiones, o regiones de textura uniforme. Los métodos globales son más robustos frente a estos problemas, pero son computacionalmente más costosos.

Los algoritmos locales se subclasifican según el método que utilizan para la detección de los correspondientes, en:

Block Matching. Se basa en encontrar el punto correspondiente comparando una región (*template*) alrededor del punto con un conjunto de regiones iguales en la otra imagen. La región que presenta mayor semejanza es la elegida y se selecciona el punto correspondiente. Las medidas usadas para la semejanza son: la correlación normalizada (NCC),

$$\frac{\sum_{u,v}\left(I_L(u,v)-\overline{I_L}\right)\left(I_R(u+d,v)-\overline{I_R}\right)}{\sqrt{\sum_{u,v}\left(I_L(u,v)-\overline{I_L}\right)^2\left(I_R(u+d,v)-\overline{I_R}\right)^2}}$$

la suma de las diferencias al cuadrado (SSD),

$$\sum_{u,v}\left(I_L(u,v)-I_R(u+d,v)\right)^2$$

la suma de los valores absolutos (SAD),

$$\sum_{u,v} \left| I_L(u,v) - I_R(u+d,v) \right|$$

y un conteo de los niveles de gris mayores que el nivel de gris del centro de la región (*rank transformation*, *census transformation*).

Optical Flow. Se basa en plantear una ecuación diferencial que relaciona el desplazamiento, d, de un píxel entre las imágenes izquierda y derecha, con el movimiento, asumiendo que su intensidad no varía

$$\nabla_x I \, d + I_t = 0$$

donde $\nabla_x I$ es el gradiente horizontal de la imagen e I_t es la derivada «temporal». El desplazamiento vertical del píxel se asume que es nulo, $\nabla_y I = 0$, dada la configuración del par estéreo. El movimiento en este caso es entre la imagen izquierda y la derecha, no existe una variación temporal entre las imágenes de la escena.

Matcheo de características. Se basa en buscar los puntos correspondientes en regiones de la imagen donde existen características relevantes (vértices, bordes, etc). Estos puntos no son muchos, y los mapas de disparidad que se pueden calcular no son densos; se obtienen puntos ubicados en el espacio pero no un mapa de disparidad con la profundidad estimada para cada punto de la imagen.

La complejidad de los dos primeros métodos es similar. También presentan problemas en las discontinuidades en la profundidad, y en regiones de textura uniforme. El tercero tiene como principal desventaja no poder calcular un mapa de disparidad denso.

Los métodos globales, imponen restricciones globales en la minimización de alguna expresión de costo o energía que modele el fenómeno estéreo, como las mencionadas en el capítulo 2.1, reduciendo los errores en las regiones con problemas. Estos métodos utilizan comúnmente dos tipos de búsquedas (minimizaciones):

Programación Dinámica. Es un método que reduce la complejidad de cálculo en problemas de optimización descomponiendo el problema en sub–problemas menores. Las restricciones globales que se imponen con este método son, generalmente, la restricción epipolar y la restricción de

orden. Para esto se construye una representación de las posibles correspondencias para cada punto construyendo una imagen que se denomina *imagen del espacio del disparidad*[1] (DSI) donde se busca un camino que recorra este espacio y minimice un cierto costo. Una explicación más detallada de estos conceptos se da en la sección 4.1 y en el apéndice B. La mayor desventaja de este método es que se basa, generalmente, en la búsqueda entre *scanlines* correspondientes, y no agrega una coherencia entre las *scanlines* adyacentes, provocando un efecto rayado horizontal (ver figuras 4.6, 5.9 o 5.7). Los intentos por utilizar Programación Dinámica con restricciones en dos dimensiones no han dado resultados eficaces [14].

Corte de Grafos. El corte de grafos se basa en armar un grafo a partir de los datos de las imágenes y buscar un corte mínimo (ver el apéndice C por más detalles). Dependiendo como se arma el grafo, el resultado obtenido es la minimización de una cierta expresión de energía. Este procedimiento se puede considerar análogo al de hallar el mejor camino en una imagen bidimensional, con Programación Dinámica, pero extendido a tridimensional con coherencias en las dos dimensiones. El resultado es una superficie que minimiza un costo energético sobre una grafo plano. Estos métodos requieren un costo computacional mucho mayor que Programación Dinámica, pero en los últimos tres años se han desarrollado nuevas implementaciones que reducen sensiblemente el costo computacional. Los algoritmos más recientes y con mejores resultados se basan en esta técnica con variantes en la forma de armar el grafo y el algoritmo para la búsqueda del corte mínimo [16, 17, 18, 19, 13].

Existen otros métodos globales para atacar el problema de la búsqueda de correspondientes, como el trabajo de Faugeras y Keriven [20] basado en *Level Sets*, donde definen un principio variacional que deben cumplir las superficies y objetos en una escena, deduciendo un sistema de EDP a resolver. O el trabajo de Sun y otros [21] modelando el problema estéreo con una combinación de MRF (*Markov Random Fields*) y usando *Bayesian Belief Propagation* para su resolución.

Weng [22] presenta un abordaje diferente al considerado en los párrafos anteriores, basado en la fase de la Transformada de Fourier. La capacidad humana de observar un par de imágenes compuestas por un conjunto de puntos aleatoriamente distribuidos (*random dots stereograms*) y recuperar una estructura tridimensional, es conocida desde 1960, en el trabajo de B. Julesz

[1] Ver sección 4.1 para una definición completa de la imagen del espacio de disparidad.

citado por Marr y Poggio [3]. Cada una de las imágenes por separado no aporta ningún tipo de información pero al ser combinadas es posible observar una estructura tridimensional. Este tipo de imágenes no siguen los patrones normales de una escena estéreo, si se intenta encontrar una correspondencia entre las dos imágenes se generan muchos falsos correspondientes lo cual provoca resultados erróneos en la mayoría de los algoritmos. Los resultados que presenta Weng con imágenes del tipo *random dots stereograms* son muy buenos, la falsa detección de correspondientes es muy baja. El autor califica el matcheo como sencillo, rápido y uniforme, presentando un algoritmo que puede ser fácilmente paralelizable. Presenta, también, experimentos con una imagen natural donde hay una textura muy importante con buenos resultados.

Estos últimos métodos, solamente se citan a modo de ejemplo de otras herramientas para el abordaje del tema, pero no fueron profundizados en esta tesis.

Para finalizar esta sección queda citar el trabajo de Scharstein y Szelisski [15] donde realizan una evaluación y comparación de los métodos para el cálculo del mapa de disparidad a partir de dos imágenes. La comparación no solamente es desde el punto de vista del número de errores en la estimación que realizan, sino que «desarman» cada uno de los algoritmos y comparan las distintas aproximaciones que hace cada uno. En el sitio web de los autores [4] están disponibles los resultados, el código fuente para realizar los experimentos hechos en el trabajo publicado, y un repositorio de imágenes estéreo, con el mapa de disparidad real calculado. Proponen a los autores de otros algoritmos realizar las mismas pruebas que ellos realizaron y mandar sus resultados para mantener la comparación actualizada.

En el resto de este capítulo se hace una revisión de la bibliografía consultada en el tema intentando seguir un orden cronológico, agrupándola por el método de minimización utilizado (Programación Dinámica -sección 3.1- y Corte de Grafos -sección 3.2-). En los apéndices B y C se presentan breves introducciones a la Programación Dinámica y al Corte de Grafos, respectivamente.

3.1. Cálculo de disparidad y programación dinámica

Uno de los trabajos más antiguos consultado es el de Ohta y Kanade [23] de 1985. Presentan un algoritmo estéreo que utiliza Programación Dinámica para la búsqueda del correspondiente en la recta epipolar («intra-scanline») y entre las *scanlines* («inter-scanline») adyacentes de forma de aprovechar la dependencia vertical entre puntos vecinos de la imagen. Para que un par de puntos sean considerados como correspondientes deben verificarlo en ambos sentidos, horizontal y vertical (intra-scanline e inter-scanline). Lo original de este algoritmo es la incorporación de la búsqueda inter-scanline dentro de la búsqueda de los correspondientes, en lugar de hacerlo en procesos separados. Los resultados presentados son buenos en imágenes que contienen bordes conectados importantes, como las imágenes urbanas que presentan, detectando correspondencias entre los bordes de los objetos aprovechando las restricciones entre las *scanlines*. Las imágenes que muestran no son las estándar, por lo que la comparación con otros resultados no fue posible.

Fua [24] presenta un algoritmo que calcula un mapa de disparidad teniendo en cuenta las discontinuidades en la profundidad basándose en la correlación y una etapa de interpolación posterior. Calcula el mapa de disparidad repitiendo el procedimiento de búsqueda por correlación, dos veces en cada una de las imágenes, intercambiándolas entre sí; utilizando sólo los puntos correspondientes que se matchean en ambos procedimientos. Esto garantiza la no propagación de errores en la siguiente etapa. Luego de haber depurado cada uno de los mapas de disparidad, éstos son interpolados teniendo en cuenta la información de las imágenes originales para preservar las discontinuidades; también es utilizado para propagar la información de disparidad en zonas de pocas características (textura, bordes, etc.). Los resultados que presenta son cualitativamente correctos y presentan, al parecer, pocos puntos con una disparidad errónea; sin embargo los resultados no son muy precisos y las imágenes aparecen difusas, seguramente debido a la interpolación.

Geiger y otros [25] y Belhumeur [12] presentan dos modelos de la visión estéreo basados en la teoría bayesiana, usando Programación Dinámica para encontrar la solución óptima. Este modelo permite estimar la escena S a partir de las imágenes de entrada I_L e I_R. Por el Teorema de Bayes, la

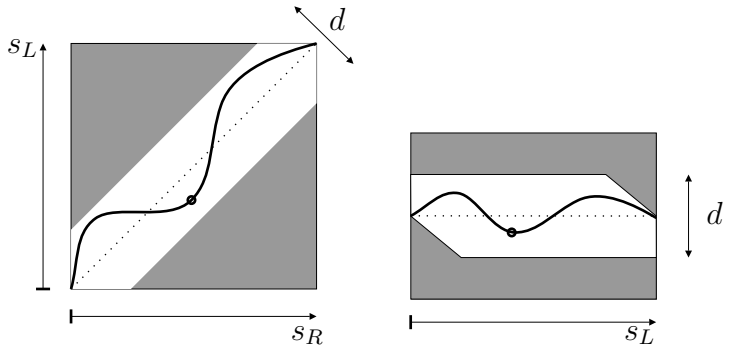


Figura 3.1: **Variantes en la construcción de la imagen del espacio de disparidad.** (a) *Imagen del espacio de disparidad utilizado en [25, 26].* (b) *Imagen del espacio de disparidad utilizado en [11]*

probabilidad a posteriori de la escena dadas las imágenes es:

$$P(S|I_L, I_R) = \frac{P(I_L, I_R|S)P(S)}{P(I_L, I_R)} \tag{3.1}$$

donde $P(I_L, I_R|S)$ es el *modelo de formación de la imagen*, $P(S)$ es el *modelo a priori* de la escena, y $P(I_L, I_R)$ está dado con las imágenes (constante).

Geiger y otros [25] plantean el modelo en una variante de la imagen del espacio de disparidad, en donde los ejes vienen dados por las coordenadas de las *scanlines* izquierda y derecha, igual que en el trabajo Tsai y Katsaggelos [26] (ver figura 3.1(a)), y en este espacio realiza la búsqueda de los correspondientes. La medida de semejanza entre los correspondientes se hace mediante una variante de la correlación normalizada con dos ventanas que contienen el punto en cuestión. Teniendo en cuenta la restricción de orden y de unicidad, y las medidas de semejanza, se genera un costo del modelo de la imagen para estimar los factores de la ecuación 3.1. Este costo es minimizado usando Programación Dinámica. Presentan un análisis detallado de las posibles oclusiones que el modelo puede manejar. Los resultados son, como lo dicen los autores, «aproximadamente correctos».

Belhumeur [12] presenta un análisis de la influencia de las oclusiones en el modelo, dejando explícitos todos los pasos en la generación del modelo de formación de las imágenes y del modelo a priori de la escena. Para el modelo a priori plantea tres modelos del mundo (*World I, II, III*) de creciente complejidad, introduciendo las oclusiones y superposición de objetos relativamente complejos. Uno de los aportes distintos de este algoritmo es el uso de

información vertical entre las *scanlines* (inter-scanline) adyacentes en un procedimiento que denomina *Iterated Stochastic Dynamic Programming*. Toma como punto inicial la solución clásica hallada con Programación Dinámica, selecciona (al azar) tres *scanlines* adyacentes, y encuentra la solución óptima para la *scanline* central. Esta selección se repite para todas las *scanlines* hasta que la solución no presenta cambios significativos. Este método introduce mucha regularización por lo que los mapas de disparidad obtenidos suelen difundir en demasía las discontinuidades de profundidad [27].

Las hipótesis y restricciones que se plantean hacen que estos algoritmos se pueden utilizar con un conjunto restringido de escenas.

Cox y otros [27] presentan un algoritmo para el cálculo de disparidad con una función de costo basada en la máxima verosimilitud. Esta aproximación no requiere conocer un modelo a priori de la escena, a diferencia del encare bayesiano. El planteamiento que realizan se basa en que la intensidad en los puntos correspondientes tiene una distribución normal centrada en la intensidad del punto real; la función de costo planteada es minimizada con Programación Dinámica. Agrega una serie de restricciones para minimizar el número de discontinuidades horizontales en la *scanline*, y verticales entre las *scanlines*, obteniendo un mapa de disparidad con mayor coherencia vertical. En este trabajo, además, generalizan el método para ser utilizado con más de dos imágenes de la escena.

Birchfield y Tomasi [28] introducen una nueva medida para el cálculo de la semejanza entre dos posibles puntos correspondientes, en lugar de las clásicas SAD, SSD o correlación. Demuestran que solamente modificando esta medida se obtiene mejores resultados, y puede incluirse dentro de otros algoritmos e incrementar su performance.

Los mismo autores presentan [29] un algoritmo para detectar las discontinuidades en la profundidad de una escena. A las regiones determinadas como ocultas no se les asigna ninguna disparidad. Luego propagan la información entre *scanlines*; usando una medida de confiabilidad del valor de la disparidad en cada punto determinado por los puntos vecinos en la vertical. Los puntos son clasificados en tres categorías de confianza, y según esa clasificación son propagadas los valores de la disparidad. Luego se repite el procedimiento en la dirección horizontal. Antes de este procedimiento se eliminan posibles errores aislados. Este método permite manejar regiones de textura uniforme, e incluye la medida de semejanza citada anteriormente [28].

Bobick e Intille [30] presentan un algoritmo para el cálculo de disparidad basado en Programación Dinámica en 1994, que inspiró a varios autores; el que está citado en esta tesis es [11] una ampliación del original y es comentado en profundidad en la sección 4.1. Introducen el concepto de punto de control (GCP), puntos donde se asegura la correspondencia entre las imágenes y fuerzan a incluirlos en la solución de la disparidad en cada *scanline*.

Tsai y Katsaggelos [26] presentan un algoritmo en la misma línea que Bobick e Intille [11], utilizando otra representación de la imagen del espacio de disparidad con las *scanlines* derecha e izquierda en los ejes (figura 3.1(a)). Este formato de imagen del espacio de disparidad fue planteada por Marr y Poggio [3]. La variante de esta aproximación está en proponer una técnica de «Dividir y Conquistar». Se buscan los puntos que con seguridad serán correspondientes entre sí debido a alguna característica sobresaliente. Estos puntos (los que Bobick e Intille denominan GCP) dividen la *scanline* en dos partes, a la que se les aplica el mismo procedimiento hasta que no se pueden hallar más puntos sobresalientes. Luego se aplica la Programación Dinámica a cada una de lo tramos obtenidos.

3.2. Cálculo de disparidad y corte de grafos

Una extensión del concepto de buscar la correspondencia entre *scanlines* correspondientes independientemente de otras *scanlines*, es la de buscar la correspondencia de todas las *scanlines* simultáneamente. Este nuevo concepto logra pasar de buscar un matcheo entre *scanlines* considerándolas independientes entre sí, a buscar un matcheo de una superficie de mínimo costo [14, 31]. Ajustar una superficie minimizando alguna energía, presenta una coherencia local en todas las direcciones, en particular la horizontal (intra-scanline) y la vertical (inter-scanline), por la forma en que se construye la solución, y no se fuerza esta coherencia mediante restricciones en la minimización.

La utilidad del corte de grafos viene dada por la capacidad para minimizar una cierta energía. Dependiendo de la expresión que se plantee los resultados pueden ser aplicables a varios problemas. Boykov y Kolmogorov [16, 17] presentan una comparación de varios algoritmos, incluyendo el original de Ford–Fulkerson y otro desarrollado por los autores, utilizando variantes de una expresión de energía en aplicaciones como restauración de imágenes, cálculo de disparidad, y segmentación interactiva de imágenes. Kolmogorov y Zabih [32] caracterizan las funciones de energía que pueden

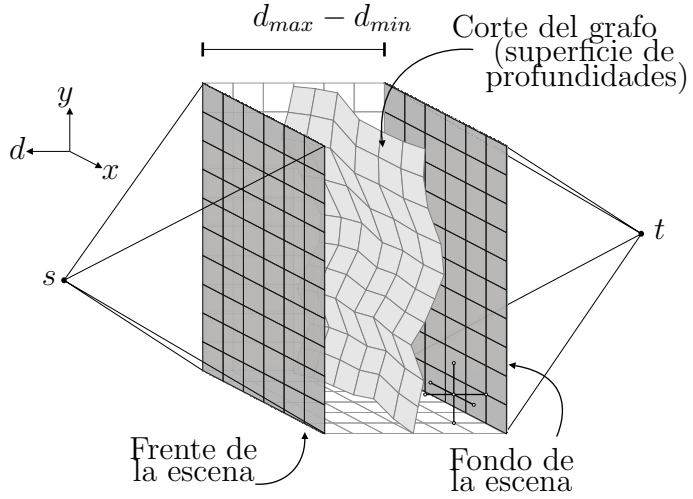


Figura 3.2: **Representación del problema de cálculo de disparidad mediante corte de grafos.** *El grafo se arma de forma que cada nodo* (x, y, d) *del mismo está conectado con cuatro puntos a igual disparidad* d *(dos en la horizontal y dos en la vertical), y con dos a disparidades* $d - 1$ *y* $d+1$*. La superficie representa el corte del grafo que minimiza alguna expresión de energía.*

ser minimizadas con este tipo de formulaciones, planteando condiciones necesarias y suficientes; y presentan una construcción general del grafo para la minimización.

A continuación se presenta un repaso de los principales algoritmos que hacen uso del corte de grafos que por su importancia y aplicación al cálculo de disparidad sobresalen. La diferencia entre los distintos algoritmos y las aplicaciones se da en la forma en que se construye el grafo y la expresión de la energía a minimizar.

El método de corte de grafos es anterior a los problemas de visión, la primera implementación en el área de estéreo fue la realizada por Roy y Cox en 1998. Roy y Cox [14] presentan un algoritmo para solucionar el problema de correspondencia con N cámaras utilizando corte de grafos. A partir de N imágenes de una escena se recupera el mapa de disparidad para una de las vistas, con las triangulaciones de las $N - 1$ imágenes restantes. El grafo construido tiene un nodo por cada píxel de la imagen en cada posible valor de disparidad, o sea $(d_{max} - d_{min}) \times m \times n$, más los dos nodos terminales, s y t, donde m y n son el número de filas y de columnas de las imágenes (ver figura 3.2). Con el agregado de los nodos terminales se tiene una estructura de grafo. La asignación de las capacidades a los enlaces es de

acuerdo a la siguiente convención: los enlaces entre nodos en la dirección de los ejes de las imágenes, $c_{oclu}(\cdot)$, los enlaces entre nodos en la dirección de la disparidad, $c_{disp}(\cdot) = k\ c_{oclu}(\cdot)$. El parámetro k controla la suavidad de la solución obtenida. La forma de resolver el corte del grafo es con uno de los algoritmos clásicos [33] conocido como *preflow–push lift–to–front*.

Boykov y otros [18] presentan un nuevo algoritmo para hallar el corte de grafos que encuentra un mínimo local en forma eficiente (más rápida) que los algoritmos tradicionales [16, 17]. Este algoritmo presenta dos variantes (α–*expansions* y $\alpha\beta$–*swaps*) y han sido utilizados en los últimos algoritmos de cálculo de disparidad que presentan los mejores resultados [16, 17, 19, 13, 15].

La minimización se plantea de la siguiente forma: encontrar una configuración de etiquetas $L = \{L_p/p \text{ en el grafo}\}$, que minimizan una energía de la forma,

$$E(L) = E_{data}(L) + E_{smooth}(L) \tag{3.2}$$

donde $E_{smooth}(L)$ mide la suavidad que presenta la solución, y $E_{data}(L)$ mide la diferencia entre L y los datos originales. Para el caso de cálculo de disparidad, las etiquetas que se desean asignar son los posibles valores de disparidad. Un caso particular de la energía (3.2) es la energía de Potts,

$$E_P(L) = \sum_{p \in \mathcal{P}} D_p(L_p) + \sum_{\{p,q\} \in \mathcal{N}} D_{p,q}\ T\left(L_p \neq L_q\right) \tag{3.3}$$

donde $D_p(L_p)$ es un costo por tener una disparidad L_p (la etiqueta para el punto p) y $D_{p,q}$ es un potencial de penalización cuando una pareja de puntos $\{p, q\}$ en un vecindario $\mathcal{N}$ tiene diferentes etiquetas (disparidades), $T(\cdot)$ vale 1 si el argumento es verdadero o 0 si es falso. Para la medida de la semejanza entre los puntos usan la propuesta por Birchfield y Tomasi [28].

Kolmogorov y Zabih [19] presentan un algoritmo donde plantean una formulación de la energía de Potts (3.3) en la cual contemplan las oclusiones, y lo resuelven con el algoritmo presentado por Boykov y otros [18]. La energía para una configuración L, que plantean, tiene la expresión

$$E(L) = E_{data}(L) + E_{smooth}(L) + E_{oclu}(L) \tag{3.4}$$

Este algoritmo es el que está dando los mejores resultados prácticos según los resultados que presentan los autores, y por otros trabajos de comparación de los distintos métodos existentes para la resolución del cálculo de disparidad [15]. Por este motivo, este algoritmo fue estudiado (sin realizar implementación del mismo) con mayor profundidad que los otros algoritmos

basados en corte de grafos, y se detalla en la sección 4.2, debido a que luego se utiliza para las pruebas realizadas.

Los mismos autores presentan [13] una variante de este algoritmo para la reconstrucción de una escena a partir de N imágenes, variando la expresión de la energía en la ecuación (3.4).

Buehler y otros [31] presentan una extensión de la formulación del problema estéreo a tres imágenes. Modelan las oclusiones agregando un costo de oclusión, y utilizan la tercera imagen de la escena para permitir que la superficie de mínimo costo que minimiza la energía sea más «flexible» en las regiones ocultas y obtener mejor precisión.

3.3. Aplicaciones y otras consideraciones

Una de las principales aplicaciones, hoy en día, para lo cual es necesario la estimación de un mapa denso de disparidades para una escena es la descomposición de una imagen en capas de igual profundidad para su posterior procesamiento y generación de nuevas vistas (*Image Based Rendering*) y en la reconstrucción tridimensional de un objeto a partir de varias vistas o una secuencia de video.

Otras aplicaciones se han encontrado en la literatura. Demirjian y Darrell [34, 35] han utilizado las imágenes del mapa de disparidad para estimar movimientos rígidos de los objetos en la escena tridimensional. Además dan [35] las propiedades geométricas del espacio de disparidad como un espacio proyectivo con la configuración de cámaras presentada en el capítulo 2, y presentan un análisis del ruido en este espacio.

Otra aplicación de un mapa de disparidad denso se encuentra en la segmentación de video [36, 37, 38]. Utilizando la información de profundidad se logra segmentar la escena en objetos que se encuentran a distintas profundidades en la misma.

Una de los restricciones mayores es que la resolución de la disparidad puede no ser suficiente como para lograr separar objetos diferentes, si se encuentran a una profundidad relativamente parecida. Esto lleva a que esta aplicación se restrinja a escenas con una configuración determinada, con los objetos de interés relativamente separados en planos paralelos a la cámara para poder distinguirlos sin mucho error.

La mezcla de la disparidad con otras características (color, posición, movimiento, textura, etc.) podría robustecer esta segmentación (ver sección 11.2).

Los mapas de disparidad también son utilizados en vigilancia y conteo de personas en espacio cerrados y abiertos [39]; en creación de realidad virtual [40], etc.

Para finalizar esta sección mencionamos tres variantes a considerar en distintas aplicaciones del cálculo de disparidad, que requieren añadir otros métodos a los ya comentados. Estas variantes son: algoritmos para cálculo en tiempo real, mapas de disparidad con resolución de subpíxel, y soluciones al problema de grandes disparidades.

Tiempo real

Los algoritmos para el cálculo de mapas de disparidad densos en tiempo real generalmente están asociados a algún tipo de hardware específico (DSP, FPGA) o microprocesadores que permita realizar el procesamiento de datos en forma eficiente [41, 42]. Los métodos que implementan, son algoritmos de los clasificados por imponer restricciones locales basados generalmente en encontrar correspondientes utilizando alguna medida de semejanza (SSD, SAD) junto con un proceso de *block matching*, y agregando consistencia entre los correspondientes obtenidos en las dos imágenes.

Subpíxel

La resolución en la disparidad que se obtiene con la mayoría de los algoritmos es con resolución a nivel de píxel. Esta resolución puede ser útil en algún tipo de aplicaciones como seguimiento o detección de objetos de tamaño importante dentro de la escena (por ejemplo para movimientos de un robot). Pero en aplicaciones como realidad virtual (construcción de escenarios virtuales), *Image Based Rendering*, o reconstrucción de superficies con mucho detalle, no es suficiente. La solución para estos casos es hacer un refinamiento de la disparidad a nivel sub-píxel.

En la bibliografía consultada hay algunos trabajos que atacan el tema. Las formas de hacerlo son variadas, por ejemplo, el método de «gradient descent» y el ajuste de una curva en puntos conocidos, o el uso de más imágenes para refinar las correspondencias y obtener una precisión mayor a la del píxel. [43, 44, 45, 46]

Grandes disparidades

Cuando el rango de disparidades es muy grande los algoritmos comentados pueden tener problemas en hallar la correspondencia correcta al aumentar el número de posible candidatos. Generalmente este problema se presenta al tener una mayor separación entre las cámaras por lo que las características de los puntos (color, iluminación) pueden haber cambiado y la búsqueda de correspondiente con los métodos comentados fallaría.

Las soluciones para estos casos podrían obtenerse con soluciones jerárquicas basadas en multiresolución, pero el submuestreo de las imágenes podría ocasionar la pérdida de detalles, que no podrían ser capturados al aumentar la resolución, provocando errores en el mapa de disparidad final.

Van Gool, Strecha y Fransens [47] proponen un enfoque probabilístico. Modelan la distribución del color de los puntos como I.I.D., y la probabilidad de oclusión mediante una mezcla de densidades. Utilizan una variante del algoritmo de Expectation Maximization (EM), para adaptar las incógnitas de estas distribuciones. Para la búsqueda de correspondientes se basan en el método planteado por Tuytelaars y Van Gool [48] de búsqueda y recuperación de características invariantes a movimiento afines, basados en la información en los niveles de intensidad de las imágenes.

Kanade y otros [49] presentan un sistema estéreo, desarrollado en *hardware* específico para procesamiento de señales (cinco cámaras, *array* de DSP (C40), PLD's, ROM, RAM, etc.) que logra obtener mapas densos de disparidad en tiempo real (30 cuadros por segundo) con imágenes de 256×240 píxeles, y capaz de manejar rangos de disparidad de hasta 60 píxeles. Las aplicaciones en la que lo utilizan son creación de realidad virtual fusionando varias escenas

4 Algoritmos estudiados

Como se comentó en la sección anterior, para una mejor comprensión de las herramientas utilizadas en el cálculo de disparidad se implementó el algoritmo de Bobick e Intille, basado en Programación Dinámica, y se analizaron los efectos de las distintas característica que lo componen.

También se estudió el algoritmo propuesto por Kolmogorov y Zabih, basado en corte de grafo. De este algoritmo no se realizó implementación pues los autores tienen una implementación que se puede obtener en Internet [50].

En las próximas secciones se detallan estos algoritmos, haciendo mayor hincapié en el primero, debido a que se le dedicó mayor estudio.

4.1. Algoritmo de Bobick e Intille

El algoritmo presentado por Bobick e Intille [11] fue implementado y probado durante este trabajo. Utiliza muchos de los métodos tradicionales de abordaje del problema e incorpora nuevas características que lo hacen un algoritmo muy citado en la literatura.

Presenta un método para el cálculo del mapa de disparidad que modela las oclusiones y las integra en el proceso de cálculo de un costo mínimo mediante el uso de Programación Dinámica. Introducen el concepto de *Ground Control Points* (GCP) y utilizan una *imagen del espacio de disparidad* (DSI) que permite «ver la disparidad» y los efectos de las oclusiones.

Como se comentó en el capítulo 2, las imágenes estéreo que se utilizan están rectificadas, y las filas de las imágenes izquierda y derecha son correspondientes una a una entre sí. Estas filas son denominadas (*scanlines*), y notaremos la correspondencia entre ellas como $s_L \leftrightarrow s_R$ donde s_L es la *scanline* de la imagen izquierda y s_R la *scanline* de la imagen derecha.

El algoritmo se ejecuta para cada una de las *scanlines* por separado y no se introduce información de las *scanlines* adyacentes. Esto presenta

desventajas pues no se fuerza una coherencia entre las *scanlines* adyacentes en el proceso de cálculo de una *scanline* en particular.

Para cada pareja de *scanlines* correspondientes, $s_L = i$ y $s_R = i$, se crea la «imagen del espacio de disparidad» de la imagen izquierda (DSI_i^L), para los posibles correspondientes, donde el rango de disparidades buscado se acota en $d \in [d_{min},\, d_{max}]$. Esta «imagen» tiene la misma cantidad de columnas de ancho que las imágenes originales y $(d_{max} - d_{min})$ filas. En cada punto $(x,\, d)$ del DSI_i^L se coloca una medida de la semejanza entre el punto de la imagen izquierda $(x,\, i)$ y el punto de la imagen derecha $(x + d,\, i)$. Para obtener la medida de semejanza se utiliza la expresión

$$\text{DSI}_i^L(x, d) = \sum_{u=-c_x}^{w_x-c_x} \sum_{v=-c_y}^{w_y-c_y} \left(\overline{I_L}(x+u, i+v) - \overline{I_R}(x-d+u, i+v)\right)^2 \qquad (4.1)$$

donde w_x, w_y, c_x y c_y determinan una ventana de dimensiones $w_x \times w_y$ con centro en (c_x, c_y); $\overline{I_L}$ e $\overline{I_R}$ son los valores de la imagen a los cuales se les resta la media en la ventana, para eliminar posibles *bias* (aditivas) entre las imágenes. En las cercanías de las oclusiones las ventanas de correlación pueden producir errores para lo cual se implementa una variación utilizando nueve ventanas que contienen al punto en cuestión; las nueve ventanas se muestran en la figura 4.1(a). En cada punto se utiliza sólo la ventana que genera el mejor matcheo.

De esta forma recorriendo para cada punto de la *scanline* izquierda los posibles $(d_{max} - d_{min})$ candidatos de la *scanline* derecha y asignando un costo a su correspondencia se construye la «imagen del espacio» de disparidad para cada *scanline* en cada una de las imágenes. El procedimiento para la imagen derecha es similar tomando la disparidad con el signo opuesto.

En la figura 4.2 se muestra un DSI para una *scanline* particular. Se pueden observar regiones verticales y diagonales con un nivel de gris alto (claras) y regiones horizontales con un nivel de gris bajo (oscuras). Las regiones oscuras corresponden a puntos correspondientes entre sí $(x \leftrightarrow x + d)$.

Encontrar un recorrido de un extremo a otro del DSI implica determinar la disparidad de cada punto en la *scanline*, o sea, hallar parejas de puntos correspondientes para cada una de las *scanlines*, $(x, s_L) \leftrightarrow (x + d, s_R)$. El camino a través del DSI se calcula usando Programación Dinámica. Para poder hacerlo hay que definir ciertas restricciones en cómo se recorre pues no todos los caminos son válidos.

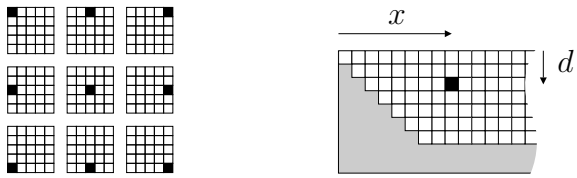


Figura 4.1: **Armar la imagen del espacio de disparidad en la imagen izquierda.** (a) *Las nueve ventanas que se utilizan para medir la semejanza entre los puntos. En negro se marca donde se coloca el punto de referencia* (c_x, c_y) (b) *En el punto marcado se coloca el resultado de la semejanza medida entre el punto de coordenada* x *en* s_L *y el de coordenada* $x + d$ *en* s_R*. Este* DSI *tiene* $d_{min} = 0$ *y* $d_{max} = 7$*.*

Basándose en la posibilidad de oclusiones algunas transiciones no son posibles. Una oclusión en la *scanline* izquierda hace que un solo punto se «corresponda» con un segmento en la *scanline* derecha; para un x fijo, un rango de puntos $(x + d_1,\ x + d_2)$ de la *scanline* derecha son posibles candidatos (ver figuras 4.3(a) y 4.3(b)). Esta oclusión se ve como un «salto» vertical en el DSI de (x, d_1) a (x, d_2), con $d_1 > d_2$; al corresponder con un objeto que se encuentra detrás, la profundidad será mayor por lo que la disparidad será menor (ver figura 4.3(c)). De forma similar una oclusión en la *scanline* derecha hace que para un punto de coordenada x_R tenga como posibles correspondientes un segmento de x_L tal que $(x_{L_i} - d_i) = x_R$. Esta oclusión se ve como un «salto» diagonal en el DSI.

Un comentario sobre la restricción de unicidad y lo expresado en el párrafo anterior: la restricción de unicidad implica que un punto de una imagen se corresponde con no más de un punto en la otra imagen, esto no se viola pues todos los puntos del segmento referido antes son *posibles* correspondientes,

Figura 4.2: **Imagen del espacio de disparidad para un caso real.** *Notar la línea horizontal quebrada de baja intensidad causadas por parejas de correspondientes. La imagen tiene una ecualización de histograma para mejor visualización.*

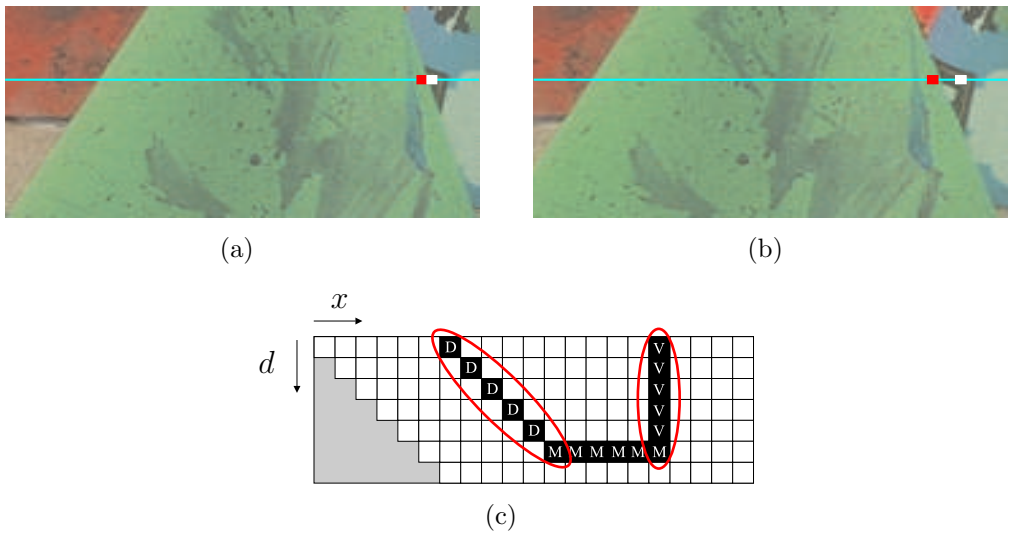


Figura 4.3: **Modelado de las oclusiones.** (a)–(b) *Imagen izquierda y derecha. Se marca una scanline particular, dos puntos adyacentes en la imagen izquierda y sus correspondientes en la imagen derecha.* (c) *Recorrido del* DSI *para la imagen izquierda. Las oclusiones provocan que el recorrido del* DSI *sólo pueda tomar tres direcciones: horizontal* (M), *vertical* (V – *disminuyendo el valor de disparidad*), *y diagonal* (D – *aumentando el valor de disparidad*).

uno lo será, o ninguno, pero no todos.

La otra dirección que puede tomar el camino en el DSI es la horizontal, manteniendo la disparidad constante para un segmento de puntos. Este caso corresponde a un plano frente a la cámara paralelo al plano de la imagen, donde la profundidad será la misma para todos los puntos del plano.

Con estas restricciones en las transiciones entre los puntos del DSI la forma de recorrerlo se reduce; estas posibles transiciones deben reflejarse en los estados entre los cuales decidir en cada etapa (punto) de la Programación Dinámica. Cada punto del camino óptimo puede estar en alguno de los tres posibles estados:

- M: matcheo horizontal,
- D: oclusión en la *scanline* derecha, camino diagonal,
- V: oclusión en la *scanline* izquierda, camino vertical.

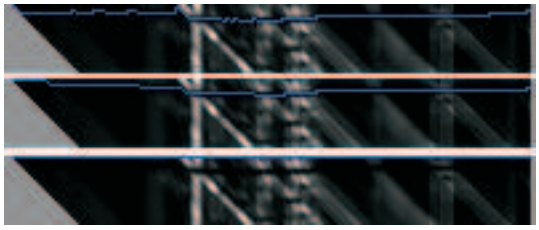


Figura 4.4: **Recorridos de la imagen del espacio de disparidad obtenidos con diferentes costos de oclusión.** *Arriba: costo bajo. Medio: costo medio. Abajo: costo alto. Notar en el último caso la zona oscura en medio del* DSI *por donde debería pasar el camino óptimo, y debido al alto costo de oclusión no es tenido en cuenta.*

No todas la transiciones son posibles pues no puede pasarse de una oclusión vertical a una diagonal, y viceversa, sin antes haber pasado por un punto de matcheo horizontal (ver figura 4.2).

Queda asignar un costo a cada píxel para poder aplicar la Programación Dinámica. El costo asignado es el valor calculado del DSI en ese punto, al cual se le suma un *costo de oclusión*, CO, en caso de ser V ó D. Por lo tanto el valor del DSI en caso de oclusión queda

$$\text{DSI}_i^L(x,d)_{oclu} = \text{DSI}_i^L(x,d) + \text{CO} \tag{4.2}$$

donde $\text{DSI}_i^L(x,d)$ es el calculado con la ecuación (4.1).

El valor del costo de oclusión es un parámetro fundamental del algoritmo. Un valor alto hará que no se elija nunca la opción de una oclusión devolviendo un mapa de disparidad constante. Por otro lado, un valor bajo, hará que el camino obtenido sea muy *ruidoso* y seguramente erróneo. En la figura 4.4 se muestran recorridos obtenidos con diferentes valores de costo de oclusión.

4.1.1. Ground Control Points (GCP)

Una de las características principales de este algoritmo es el uso de lo que los autores denominan *Ground Control Points* (GCP). Estos puntos se definen como puntos relevantes del DSI en los que puede asegurarse la correspondencia; definir el punto del DSI (x_G, d_G) como GCP implica afirmar que los puntos con coordenadas $x_L = x_G$ y $x_R = (x_G + d)$, se corresponden. Los GCP ayudan a encontrar el camino óptimo a través del DSI en presencia de oclusiones grandes.

Para hallar los GCP los autores proponen varios «filtros» para la selección definitiva. Primero, un GCP debe ser el mejor correspondiente considerando ambas *scanlines*, lo cual implica ser el «mejor correspondiente» tanto en la columna como en la diagonal. Segundo, el valor del DSI en el GCP debe ser menor que el costo de oclusión, así evitar correspondientes espúreos. Tercero, es necesario que el vecindario del GCP presente suficiente textura para determinar una buena correspondencia. Finalmente, se descartan los GCP aislados, es decir, que no tiene otros GCP en el vecindario.

Para forzar el camino óptimo a través de los GCP se prohíbe el pasaje (con un costo muy elevado) por puntos que puedan hacer que se eluda el GCP. Por ejemplo, ningún otro punto del DSI con coordenada $x = x_G$ puede estar habilitado. De esta forma se conduce al algoritmo para utilizar los puntos donde se conoce la existencia de una correspondencia. Al mismo tiempo esto reduce la carga computacional del algoritmo, pues descarta una cantidad de puntos que no serán utilizados en la evaluación, acelerando el proceso de búsqueda del camino óptimo.

En la figura 4.5 podemos ver el camino a través de un DSI calculado sin el uso de los GCP, los GCP encontrados en el DSI y el camino calculado teniendo en cuenta los GCP. Notar cómo se prohíbe el pasaje por una región del DSI de forma de conducir el camino hacia el GCP.

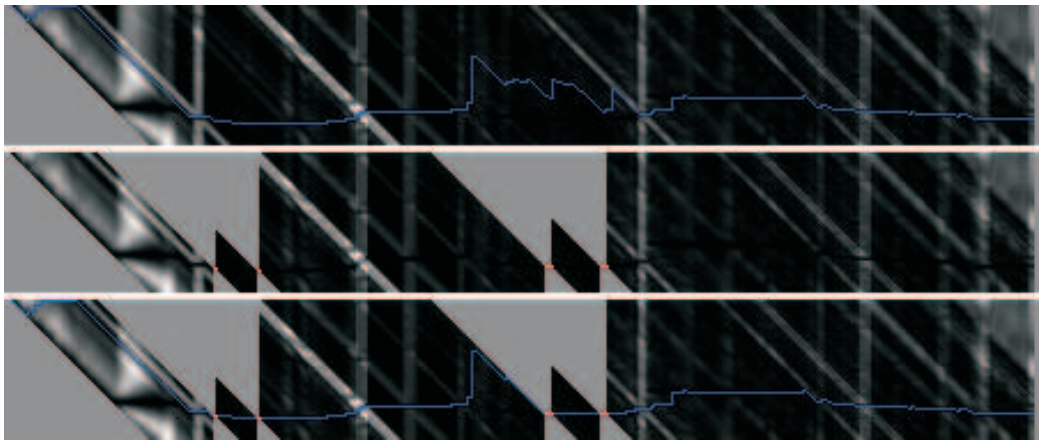


Figura 4.5: **Uso de los GCP.** *Arriba: el camino (en azul) a través de un* DSI *calculado sin el uso de los* GCP*. Medio: los* GCP *encontrados en el* DSI*, en rojo, y las regiones donde se prohíbe el pasaje del camino. Abajo: el camino (en azul) calculado teniendo en cuenta los* GCP*.*

4.1.2. Uso de bordes en el DSI

Los objetos en la imagen generalmente provocarán que sucedan oclusiones en las imágenes. Estas oclusiones se reflejan en el DSI como bordes verticales o diagonales según la oclusión está en la imagen izquierda o derecha, respectivamente. La reducción del costo en los *bordes del* DSI, incrementa la detección de las oclusiones sin agregar complejidad al algoritmo. Esta característica se agrega al algoritmo produciendo mejoras en las regiones donde existe oclusión y el algoritmo no puede definir ningún valor de disparidad, dándole un aspecto más definido a los bordes y sin tanta difusión.

4.1.3. Discusión

Los mapas de disparidad que se obtienen con este algoritmo son muy aproximados a la solución real; en el capítulo 5 se hace un estudio y comparación de los resultados. En la figura 4.6 se puede ver los resultados y la imagen estéreo correspondiente, su mapa de disparidad real, y varios mapas de disparidad variando el costo de oclusión.

Como se ve, la solución es sensible al valor del costo de oclusión utilizado. El uso de los GCP colabora en este sentido, al garantizar algunos matcheos correctos, y ayudan a obtener una mejora local de la solución.

La posibilidad de propagación de errores a través de la *scanline* también es un problema de este algoritmo si el costo de oclusión no es el adecuado pues los GCP no colaboran en hallar un camino óptimo global.

De las desventajas más notorias de este algoritmo, y de todos los algoritmos que se basan en búsquedas intra-scanline, sin incorporar información de *scanlines* adyacentes (inter-scanline), es el «rayado» horizontal que se ve en los mapas de disparidad calculados. Este rayado se origina al tener distintas soluciones para *scanlines* consecutivas, debido, principalmente a la propagación de un error. Normalmente este tipo de errores se producen en los bordes de los objetos de la escena, donde hay discontinuidades en la disparidad por las oclusiones.

La implementación que se realizó en MatLab© permite analizar los efectos de cada una de las componentes características de este algoritmo; pero no permite tener una estimación de los tiempos de cálculo. En las pruebas que se presentan en el capítulo 5 se utilizó un código que implementa este algoritmo en el marco de una comparación de varios métodos de cálculo de disparidad [15]. El algoritmo compilado a partir de este código realiza

el cálculo del mapa de disparidad en un promedio de 450 mseg. para imágenes de 384×288 píxeles.[1] Sí podemos afirmar que la parte más costosa de cálculo es la construcción de la imagen del espacio de disparidad, por todas las medidas de semejanza que se deben hacer. En la implementación en MatLab© se utilizó un algoritmo rápido de cálculo de correlación que reduce los tiempos considerablemente, pero aún sigue siendo la parte más costosa.

En la sección 4.1.4 se prueban algunos métodos para mejorar la solución reduciendo el efecto de rayado horizontal en los mapas de disparidad, con el objetivo de mostrar las mejoras que se pueden obtener con un post–procesamiento de los mismos.

4.1.4. Agregado de coherencia inter-scanline

Como se comentó la mayor desventaja de los algoritmos que utilizan Programación Dinámica es que se basan en la búsqueda en las *scanlines* (intra-scanline) correspondientes, y no agregan una coherencia entre las *scanlines* adyacentes (inter-scanline), provocando un efecto rayado. En varios trabajos [12, 29, 28, 27] se presentan métodos para solucionar esta desventaja luego que se obtuvo el resultado. Estos métodos fueron comentados en el capítulo 3.

Para complementar el estudio del algoritmo de Bobick e Intille, se implementaron algunos métodos para la propagación de los resultados entre las *scanlines*[2]. Se intenta mostrar que un post–procesamiento, generalmente con poca carga computacional, puede mejorar los resultados frente al problema del rayado horizontal de este tipo de algoritmo.

Se implementaron tres métodos, que a continuación se detallan:

1. Promediado vertical,
2. Filtro de mediana en la vertical, y
3. Estimación de confianza y propagación.

En la figura 4.7 se muestran los resultados obtenidos.

El primero, considera la información de m *scanlines* adyacentes y toma un promedio de las mismas. Se usaron dos variantes, una ponderando las

[1] En un PC, Pentium IV, 2 GHZ, 768 MB de RAM

[2] El algoritmo de Bobick e Intille no incluye ningún tipo de procesamiento de este tipo.

scanlines adyacentes (menor peso a las más alejadas), y otra donde se le da el mismo peso a todas. El resultado obtenido mejora levemente el mapa de disparidad. El promediado realizado es un filtrado pasabajos en la vertical, si pensamos cada columna como una señal unidimensional, por lo que reduce el «rayado» del mapa de disparidad original, pero al mismo tiempo difunde los bordes definidos. El mapa de disparidad final es muy difuso (borroso) y el número de errores (píxeles con disparidad mal calculada) aumenta. El rayado horizontal causa altas frecuencias en la dirección vertical; esta característica puede ser explotada para actualizar el tamaño de la ventana (m) de muestras que se utiliza para tomar el promedio.

El segundo método implementado realiza un filtrado de mediana entre las *scanlines*. Para cada punto se consideran puntos de las m *scanlines* adyacentes en su columna y se le asigna el valor de la mediana. Los resultados que se obtienen son buenos considerando el poco procesamiento que se realiza. Los bordes de los objetos se mantiene bien definidos y se elimina la mayoría del «rayado» original. Algunos errores aún se mantienen que este simple método no puede eliminar.

El tercer método, se basa en una simplificación del método propuesto por Birchfield y Tomasi [29]. Para cada punto se toma una medida de la confianza del valor obtenido; esto se hace contando el número de puntos adyacentes (en la columna) que tienen el mismo valor de disparidad. Se utiliza un umbral para determinar los puntos «poco confiables». En la figura 4.7(d), se muestra la disparidad en los puntos «confiables». Queda calcular la disparidad en los puntos no confiables. Los métodos para hacer eso pueden ser muchos, en este caso se propagó la información en la vertical (hacia arriba y hacia abajo) hasta obtener un mapa de disparidad completo.

Los resultados que se obtienen con este método son buenos, se reduce significativamente el «rayado» original, aunque la propagación de las disparidades no es la óptima; se nota que esta propagación crea un rayado vertical. Se aplicó este mismo procedimiento en la dirección horizontal, para el mapa de disparidad obtenido luego de la propagación en la vertical. Los resultados mejoran subjetivamente, obteniendo regiones mejor definidas en los bordes y con pocas rayas en la vertical.

Una solución, que no se probó, es agregar información de los objetos presentes en la escena; extrayendo los bordes de la imagen estéreo correspondiente, y no dejando que la propagación se realice a través de los bordes («propagación anisotrópica»). O mejor aún, realizar una segmentación de la

	RMS	$B(\%)$
$Original$	1,252	8,9
$Promedio$	1,124	9,6
$Mediana$	1,105	7,2
$B\&T$	1,243	6,0
$B\&T \times 2$	1,226	5,7

Tabla 4.1: Resultados obtenidos para B y RMS con los métodos de agregado de coherencia entre *scanlines*.

escena en objetos, e intentar asignar una disparidad a cada objeto; se pueden permitir diferencias en niveles de disparidad pequeñas, para permitir superficies con una inclinación (planos inclinados) respecto al plano de la imagen.

De los métodos realizados el segundo (filtro de mediana) obtiene buenos resultados (ver Tabla 4.1), preservando regiones importantes que el tercer método (confianza + propagación) elimina, por ejemplo los brazos de la lámpara; además tiene una carga computacional menor.

El agregado de alguno de estos métodos puede dar un mejor resultado sin sobrecargar en exceso el procesamiento del algoritmo.

4.2. Algoritmo de Kolmogorov y Zabih

El segundo algoritmo estudiado es el presentado por Kolmogorov y Zabih [19]. En este algoritmo se basan en las técnicas de Corte de Grafos para hallar la correspondencia entre los puntos de las imágenes, imponiendo la restricción de unicidad. La expresión de energía que minimizan se basa en la energía de Potts, ecuación (3.3), a la cual se agrega un término que considera las oclusiones en la escena. Así, el problema de los puntos ocultos se modela dentro de la minimización de la energía.

Para resolver de forma eficiente el corte del grafo presentan dos algoritmos similares al algoritmo presentado por Boykov y otros [18], lo cual permite obtener tiempos mucho menores que con otros métodos de corte de grafos.

La energía que se plantea tiene la expresión

$$E(L) = E_{data}(L) + E_{oclu}(L) + E_{smooth}(L) \tag{4.3}$$

donde L es una configuración de posibles etiquetas a colocar en cada píxel. Para el caso de cálculo de disparidad, las etiquetas son: los posibles valores de disparidad, $d \in [d_{min}, d_{max}]$, y la etiqueta de oculto.

El término $E_{data}(L)$ mide la semejanza entre puntos correspondientes, basado en la diferencia en los niveles de gris de los mismos,

$$E_{data}(L) = \sum (I(p) - I(q))^2$$

donde p y q son los puntos correspondientes. $E_{oclu}(L)$ añade un costo fijo, C_p, al asignar la etiqueta de oculto a un punto; el cual se determina cuando no se le puede asignar otro punto correspondiente (número de correspondientes $N_C = 0$)

$$E_{oclu}(L) = \sum C_p \; T(N_C = 0)$$

donde $T(\cdot)$ vale 1 si el argumento es verdadero o 0 si es falso.

Finalmente, $E_{smooth}(L)$ es un término de suavidad que intenta asignar una disparidad similar a los puntos cercanos; imponiendo una coherencia en todas las direcciones, no sólo en la de la *scanline*. La expresión que tiene este término es similar al término $E_{smooth}(L)$ de la ecuación (3.3),

$$E_{smooth}(L) = \sum_{\{p,q\} \in \mathcal{N}} D_{p,q} \; T\left(L_p \neq L_q\right)$$

donde $D_{p,q}$ es un potencial de penalización cuando una pareja de puntos $\{p, q\}$ en un vecindario $\mathcal{N}$ tiene diferentes disparidades.

La construcción del grafo no es trivial y está fuera del alcance de esta tesis, por detalles en su construcción, asignación de costos y algoritmo de minimización consultar la bibliografía citada.

En la implementación que proponen los autores, todas las componentes de la energía (4.3) son configuradas a partir de un único parámetro λ, fijando otros por resultados experimentales. En la publicación donde presentan los resultados sostienen [19] que la variación de los mismos es relativamente insensible a la elección del mismo, para las pruebas que realizaron. Por lo tanto proponen una «sintonización» de los términos de energía fija en función de λ. Al término $E_{smooth}(\cdot)$ se le asigna un valor proporcional a λ. El costo de oclusión, C_p, también es proporcional a λ.

En la figura 4.8 se ven los mapas de disparidad estimados variando el parámetro λ de forma «exagerada». Los puntos en rojo son etiquetados como ocultos. Podemos ver que la definición en los bordes es mucho más real que lo que logra el otro algoritmo testeado.

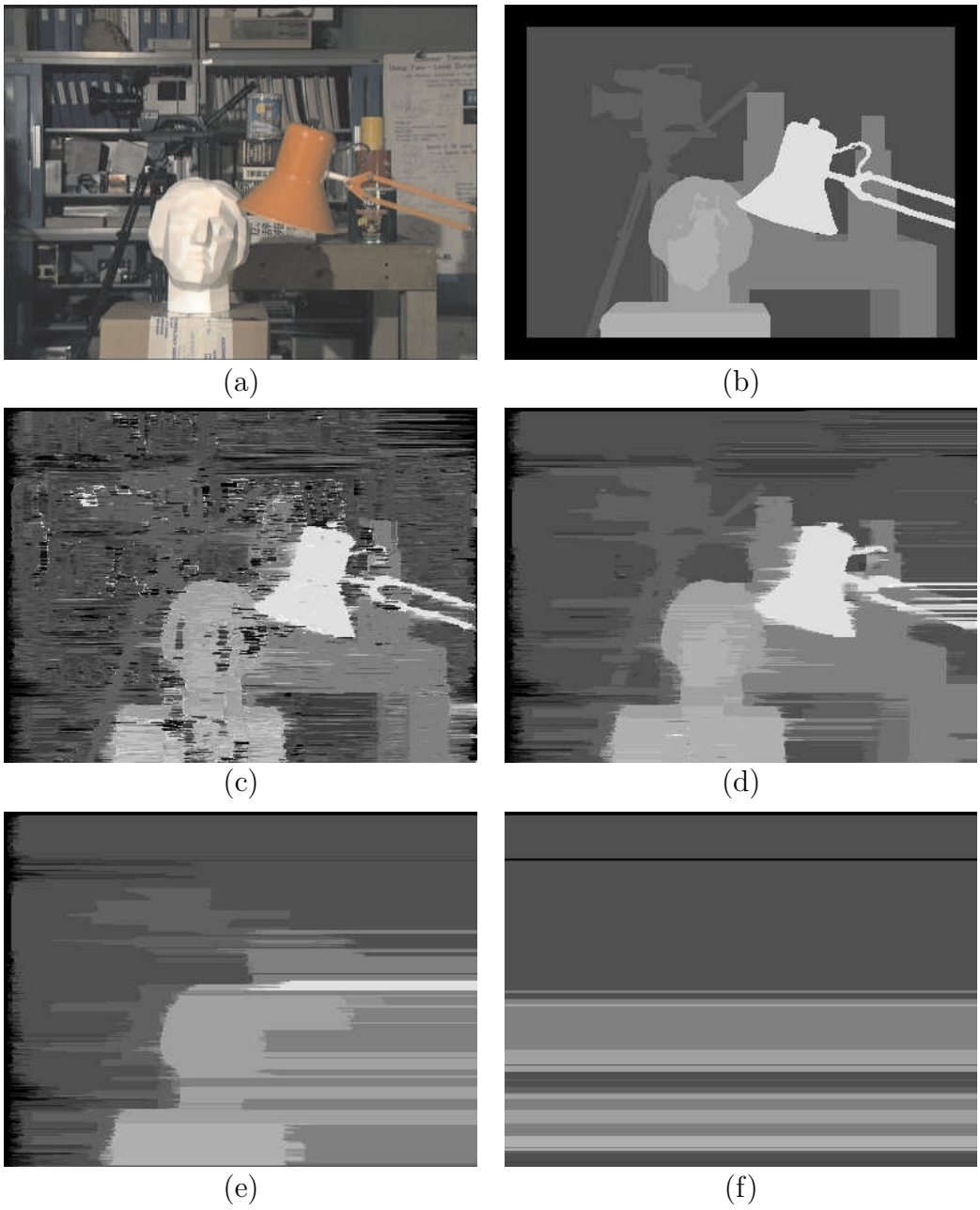
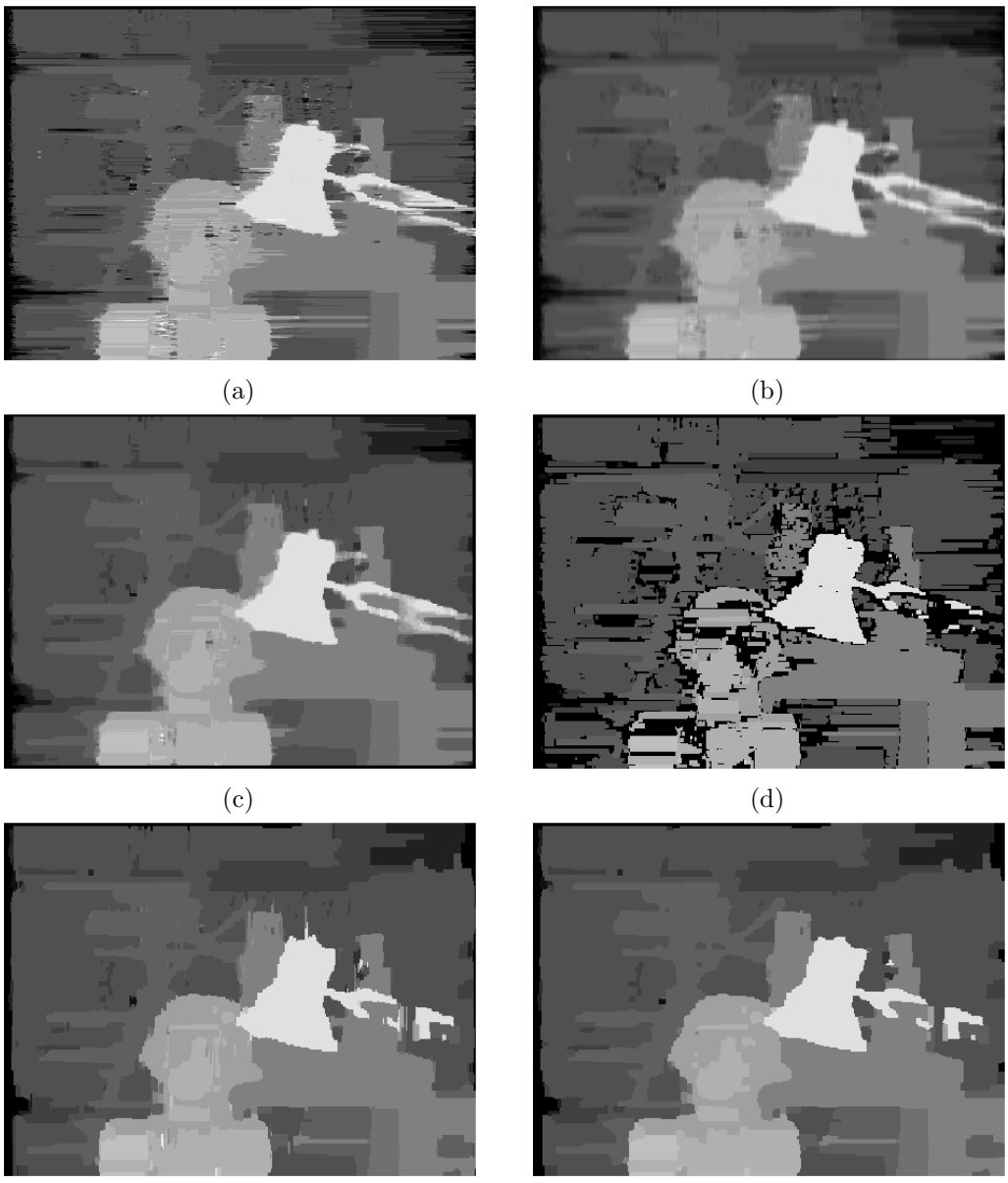


Figura 4.6: **Mapas de disparidad obtenidos con el algoritmo de Bobick e Intille.**(a) *Imagen izquierda del par estéreo.* (b) *Mapas de disparidad real (Groundtruth).* (c–f) *Mapas de disparidad obtenidos con diferentes costos de oclusión (crecientes).*



(a) (b) (c) (d) (e) (f)

Figura 4.7: **Refinamiento de los mapas de disparidad agregando coherencia intra-scanline.** (a) *Mapa de disparidad obtenido con el algoritmo de Bobick e Intille.* (b) *Refinamiento obtenido con promedio.* (c) *Refinamiento obtenido con filtro de mediana.* (d) *Mapa de disparidad en los puntos «confiables».* (e) *Refinamiento obtenido con difusión vertical en los puntos «no confiables».* (f) *Refinamiento obtenido con difusión vertical y horizontal en los puntos «no confiables».*

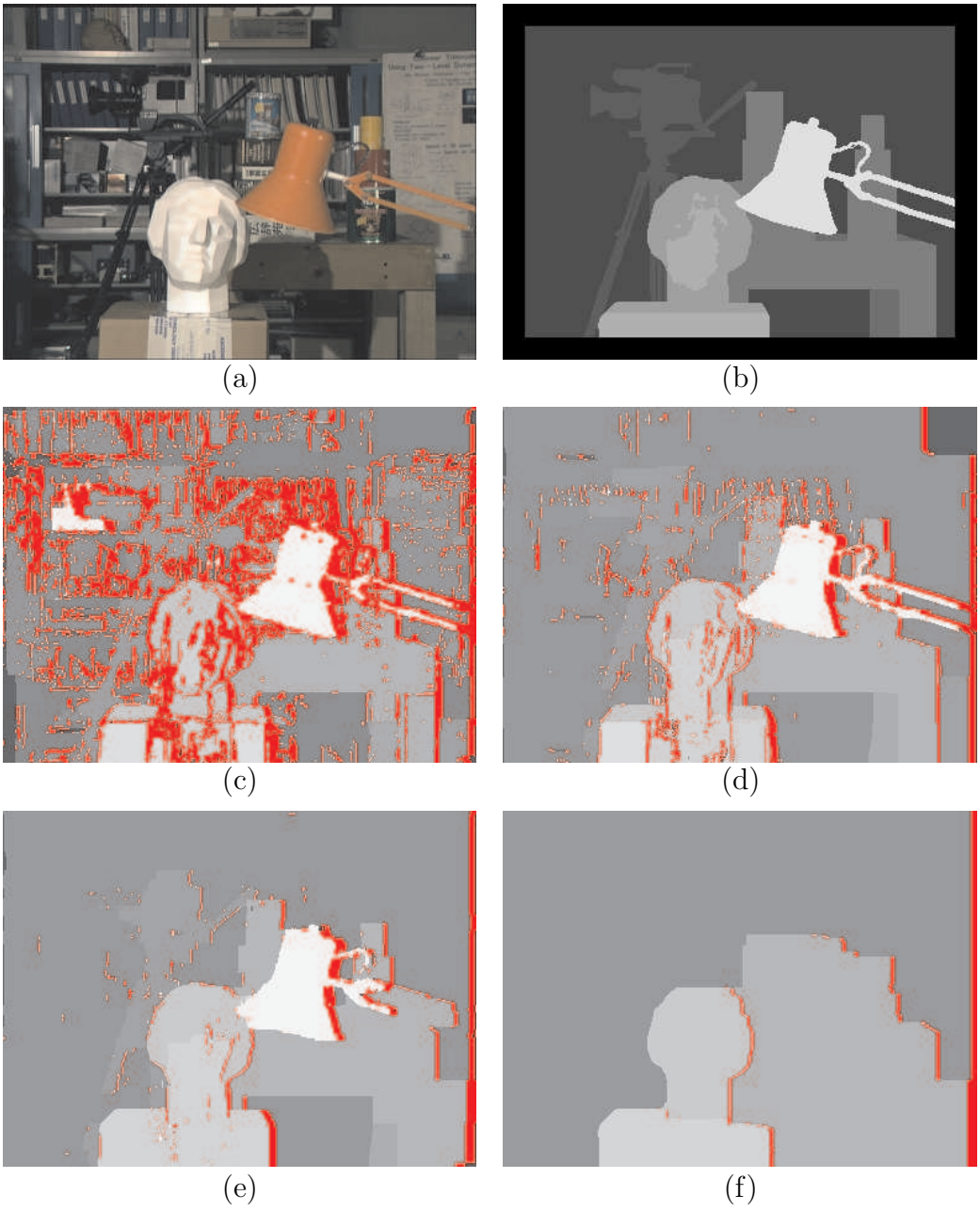


Figura 4.8: **Mapas de disparidad obtenidos con el algoritmo de Kolmogorov y Zabih.** (a) *Imagen izquierda del par estéreo.* (b) *Mapas de disparidad real (Groundtruth).* (c–f) *Mapas de disparidad obtenidos variando el parámetro de configuración. Los puntos en rojo son puntos etiquetados como ocultos por el algoritmo.*

5 Experimentos, discusión y conclusiones

Se realizó la comparación de dos métodos de cálculo de disparidad, el algoritmo de Bobick e Intille descrito en la sección 4.1 y el algoritmo de Kolmogorov y Zabih descrito en la sección 4.2. Para ambos métodos se pueden encontrar implementaciones eficientes, con el código libre para su uso y prueba, en Internet [50, 4].

Para la comparación de ambos algoritmos no se realiza la etapa de post–procesamiento inter-scanline descrita en la sección 4.1.4, con el objetivo de utilizar el algoritmo propuesto originalmente por los autores.

Uno de los experimentos que se hicieron con estos métodos fue añadir ruido gaussiano aditivo con diferentes varianzas a las imágenes estéreo, y probar el funcionamiento de cada uno de éstos. Llamaremos: DP, al de Bobick e Intille (por Dynamic Programming), y GC, al de Kolmogorov y Zabih (por Graph Cuts).

Se obtuvieron los resultados para cada uno de los algoritmos variando un parámetro en cada uno de ellos. Para DP se hizo variar el *costo de oclusión*, que como se dijo anteriormente es crítico al momento de hallar el camino óptimo en cada *scanline*. Para GC, los autores realizaron un ajuste de los parámetros de forma de obtener resultados razonables, dejando el algoritmo en función de un solo parámetro λ; este es el parámetro que se hizo variar.

Para los experimentos se usaron imágenes estéreo obtenidas de bases públicas, en niveles de grises (8 bits), para las cuales se conoce la disparidad real. El rango de disparidad de las imágenes seleccionadas es de 16 píxeles.

Las imágenes que se utilizaron son conocidas como `tsukuba`[1] y `corridor`[2]

[1] De la Universidad de Tsukuba. Autores: Y. Ohta y Y. Nakamura. También conocida como «head and lamp».

[2] De la Universidad de Bonn (`http://www-dbv.cs.uni-bonn.de/stereo_data/`).(Visitada el 2005-11-27)

(ver figura 5.1). La primera es una imagen de una escena real, donde se pueden apreciar varios objetos relativamente planos dada la distancia a la que se encuentran de la cámara. La disparidad real fue obtenida manualmente por los autores. La segunda es una escena artificial, generada por computadora, en la cual se ve un pasillo con varios objetos en el piso y las paredes.

En `tsukuba` existen varios planos de profundidades distintas que generalmente son bien capturados por los algoritmos. La mayor dificultad se plantea en las oclusiones que se dan entre los objetos; y en el rostro de la cabeza, donde es posible (dada la cercanía a la cámara) detectar las diferencias de disparidad entre los planos de la nariz y de los ojos.

En `corridor` la profundidad de la escena varía de forma continua y rápida en las paredes, piso y techo del corredor provocando que la mayoría de los algoritmos no puedan seguir esta variación, generando un mapa de disparidad cuantificado en forma gruesa, provocando errores importantes. Este fue el principal motivo de su elección, sumado que al ser una imagen generada por computadora el ruido de las imágenes originales es nulo y la disparidad es conocida con muy buena precisión. También hay zonas de textura nula (en el damero del piso) que presenta inconvenientes al momento de buscar correspondientes.

Las medidas que se tomaron fueron: la cantidad de puntos donde fue mal calculada la disparidad, B, y una medida de *la potencia del error* de disparidad, RMS. Estas medidas fueron planteadas por Scharstein y Szeliski [15],

$$B = \frac{1}{N} \sum_{x,y} |d_C(x,y) - d_R(x,y)| > \Delta_d \tag{5.1}$$

$$RMS = \sqrt{\frac{1}{N} \sum_{x,y} \Big(d_C(x,y) - d_R(x,y) \Big)^2} \tag{5.2}$$

donde $d_C(x,y)$ es la disparidad calculada, $d_R(x,y)$ es la disparidad real en el punto (x,y), Δ_d es una tolerancia al error en la disparidad y N es el número de puntos donde se calcula la disparidad. Se tomó $\Delta_d = 1$ para las resultados que se presentan.

Para las imágenes de `corridor` además se estudiaron las soluciones que presentan los algoritmos a lo largo del corredor en la precisión de la variación de la disparidad.

5.1. Discusión

Antes de presentar los resultados, conviene plantear algunas observaciones sobre los parámetros que se manejan. El costo de oclusión en el algoritmo DP tiene una interpretación bastante intuitiva y directamente relacionada con los datos que se están utilizando. En este caso el costo de oclusión se suma a la medida de semejanza que se coloca en el DSI, por lo que la influencia es fácilmente apreciable, y relacionable con los niveles de intensidad de las imágenes. En cambio el parámetro λ de GC no tiene una interpretación o relación directa con los niveles de gris de las imágenes, dada la forma de construcción del grafo, planteo de la energía a minimizar y «sintonización» de los parámetros.

La primera verificación que se realizó fue la dependencia del algoritmo DP con el tamaño de la ventana de correlación utilizada. Con un tamaño de ventana mayor las regiones que se recuperan son más difundidas, con menos definición en los bordes. El hecho de la utilización del mejor matcheo de nueve posibles posiciones hace que la solución sea coherente a pesar de poder introducir error al aumentar el tamaño de las ventanas. Los resultados no varían demasiado por lo que se fijó este parámetro en el resto de las pruebas; se utilizó una ventana de 7×7 píxeles.

Los mapas de disparidad generados con DP se muestran en las figuras 5.7 y 5.9. El comportamiento frente a la potencia de ruido añadido a las imágenes del algoritmo de DP es de acuerdo a lo esperado, con gran regularidad. Al aumentar la potencia del ruido introducido en las imágenes del par estéreo, la potencia del error aumenta, al igual que el número de puntos con disparidad mal calculada. Este comportamiento se ve claramente en la figuras 5.2 y 5.3, donde se grafican RMS y B en función del costo de oclusión, paramétrico en la potencia del ruido introducido.

Es interesante el comportamiento al variar el costo de oclusión. En la figura 5.2(a) se observa que la potencia del error disminuye al aumentar el costo de oclusión. La explicación es directa recordando la ecuación (4.2) y la figura 4.4; al aumentar el costo de oclusión la capacidad de cambio de la disparidad es menor, por lo que se adapta más a regiones suficientemente planas y paralelas al plano de la imagen; como las que se ven en `tsukuba`.

Por otro lado, en `corridor` no existen estos planos paralelos al plano de la imagen; por el contrario, los planos tienen una pendiente importante. En la figura 5.3 vemos que, tanto RMS como B, no tiene el comportamiento anterior para potencias de ruido bajas; en este caso, tiene un aumento pequeño

(una variación relativa menor del 15 %). Para potencias de ruido mayores el comportamiento es el mismo que en el caso anterior.

Estos resultados se pueden observar en los mapas de disparidad que se muestran. En las figuras 5.7 y 5.9 vemos como para `tsukuba` hay una aproximación a la solución correcta al aumentar el costo de oclusión. Para `corridor`, la aproximación también mejora con el aumento del costo de oclusión, sin embargo, en la primera fila (con potencia de ruido baja) la mejora no es tan apreciable.

Para el algoritmo GC, el comportamiento frente a la potencia del ruido agregado tiene un comportamiento similar, que se comprueba tanto en las gráficas (figuras 5.4 y 5.5) como en los mapas de disparidad (figuras 5.6 y 5.8). Es relevante en este algoritmo la dependencia con el parámetro λ; no se puede afirmar que la forma en que varían los resultados a las distintas pruebas tenga un comportamiento tan claro como en el caso de DP. Aquí los resultados son más irregulares (figuras 5.4 y 5.5).

Conviene analizar la influencia de λ en los distintos términos de la energía dada por la ecuación 4.3. A los términos $E_{smooth}(\cdot)$ y $E_{oclu}(\cdot)$ se les asigna un valor proporcional a λ. Esto implica que al aumentar λ se tenderá a que la solución de mínima energía presente mayor suavidad (*smoothness*), pues se penaliza la poca suavidad de la solución; y será más costoso etiquetar un punto como oculto. Esto hace que las regiones tiendan a hacerse más homogéneas (ver figura 4.8). Al aumentar λ el ruido deja de tener tanto peso en la minimización, y pesa más tener discontinuidades en el mapa de disparidad, por lo que la aproximación es mejor. De la mano de este comportamiento, el número de puntos que se etiquetan como ocultos disminuye, al haber menos planos que se solapen en las soluciones (ver figura 4.8).

Encontrar una solución mediante el método de corte de grafos es similar, como se mencionó, a encontrar una superficie que se adapte a los datos de entrada de forma de minimizar la energía propuesta. Por esto la solución de este algoritmo se puede interpretar como un conjunto de parches de superficies planas (mismo valor de etiqueta–disparidad) que guardan una coherencia local. Los cambios entre los valores de estos parches dependen de la penalización en términos de la energía que representen.

Desde el punto de vista del número de puntos mal clasificados (B) en los experimentos realizados, `tsukuba` tienen un comportamiento como el que se reporta en la literatura referida. Con DP y sin ruido agregado el porcentaje de puntos con disparidad mal calculada varía de un máximo de 32 % con un costo de oclusión bajo, llagando a un 10 % con costos de oclusión medios

y altos (ver figura 5.2(b)). Por otro lado, GC en el mismo experimento no supera el 5 % de puntos con disparidad mal calculada, excepto con un valor de λ muy bajo que obtiene un 8 %. Cuando se agrega ruido aumentando la potencia del mismo, este porcentaje aumenta drásticamente con DP, aún con costos de oclusión altos superando el 50 %. GC tiene un comportamiento diferente que DP pues con valores de λ medios y altos se obtiene porcentajes de error en la disparidad menores al 10 % (esto puede verse en la última columna de la figura 5.8).

Con `corridor` la situación es diferente, siendo DP quien presenta un comportamiento más regular ante la variación de los parámetros. El porcentaje de error en el caso de DP no supera el 50 % en la mayoría de los experiementos realizados, mientras que GC no logra disminuir del 70 % en ninguno de ellos. Si se observan los mapas de disparidad generados (figuras 5.6 y 5.7) se entienden las variaciones numéricas que presentan. En este par de imágenes los mayores problemas para los algoritmos se dan en las paredes, techo y piso del corredor donde el plano inclinado no es bien capturado por los algoritmos. Sin embargo, DP tiene mejor comportamiento que GC en este caso.

En las escenas donde existen planos paralelos al plano de la imagen las ventajas de GC al forzar una coherencia en *todas* las direcciones del mapa de disparidad calculado, se reflejan en una mejor definición en las regiones que se determinan. Mientras que DP al sólo imponer las restricciones epipolar y de orden en la *scanline*, obtiene el efecto de «rayado» ya comentado. Como se mostró en la sección 4.1.4 una etapa posterior de agregado de coherencia vertical a los mapas de disparidad obtenidos con DP puede ayudar a dar resultados mucho más aproximados al real.

En la figura 5.10 vemos un caso de interesante análisis. Se muestran los mapas de disparidad calculados a partir del mismo par de imágenes ruidosas, con ambos algoritmos con valores de costo de oclusión y λ medios, respectivamente. Lo destacable es en la zona del brazo de la lámpara. GC crea una región falsa uniendo ambos brazos y asignando una disparidad promedio a esta región. Por su parte, DP mantiene ambos brazos separados y asigna una disparidad *razonable* entre los mismos. Este fenómeno, se debe a que GC intenta colocar una superficie plana en cada valor de disparidad diferente, penalizando la variación en las disparidades. GC plantea una coherencia en *todas las direcciones* de la imagen, en este caso particular, esta coherencia no ayuda, pues hay una dirección, que no es ni horizontal ni vertical, en la cual se debe forzar esta coherencia, y no en las otras. Este problema, se da para un

valor de λ grande para esta región, pero podría ser válido para otra región del mapa de disparidad, por lo que el uso de un parámetro global podría causar problemas en diferentes partes del mapa de disparidad. Agregar una detección de bordes y direcciones, podría darle una mayor coherencia para resolver este tipo de problemas.

Tampoco DP logra un buen resultado en este caso, pero la solución es más próxima a la real. La clave está en que fuerza una coherencia *local* en cada *scanline*. Utilizar alguno de los métodos comentados para agregar coherencia en la vertical podría aproximar aún más la solución sin llegar a crear falsas regiones, como GC.

Las disparidades obtenidas para `corridor`, ponen de manifiesto uno de los principales problemas de los algoritmos de cálculo de disparidad, la poca resolución que se obtiene en la disparidad. De la disparidad real, vemos que presenta un suave cambio a lo largo del corredor, lo cual no es recuperado por ninguno de los métodos testeados. Tanto GC como DP obtienen cuantificaciones gruesas de la disparidad real, resultando en malos valores de RMS y B.

Para evaluar el grado de aproximación de la variación de la disparidad a lo largo del corredor, se tomó una región del techo del corredor que se muestra marcado en la figura 5.11. Este perfil de disparidades se grafica en la figura 5.12, junto con los perfiles de disparidades medios calculados por cada algoritmo, variando su parámetro. Se muestran los perfiles para las pruebas realizadas con ambos algoritmos, con y sin ruido agregado a las imágenes estéreo. Se verifica que ninguno de los algoritmos logra aproximar la pendiente correcta. Sin embargo DP se aproxima más a la pendiente real que GC, cuando no se tiene ruido agregado. Cuando se le suma ruido, continúa teniendo una mejor aproximación pero presenta mayor dispersion.

Otro comentario para realizar sobre esta escena es la dificultad existente en zonas grandes sin textura, por ejemplo, dentro de cada baldosa del damero del piso. En este punto observamos (ver figuras 5.6 y 5.7) que ambos algoritmos realizan un buen matcheo de los correspondientes y no presentan errores en la mayoría las pruebas realizadas. En los mapas de disparidad con una potencia de ruido agregado alta y costos bajos, donde se dan los errores más importantes, se aprecia una discontinuidad en la disparidad en los bordes del damero del piso. Esta discontinuidad no es errónea, al contrario, es la disparidad correcta. Dentro de las baldosas de intensidad uniforme sólo se tiene ruido que no se puede correlacionar, mientras que en los bordes entre baldosas existen características locales que permiten suponer una correspondencia.

5.2. Conclusiones

Ninguno de los algoritmos es óptimo para cualquier tipo de escenas y pueden presentar errores en algunas partes del mapa de disparidad. Como sucede con muchos algoritmos, el funcionamiento es óptimo con cierta estructura geométrica de la escena, pero cuando alguna de las restricciones en la estructura o hipótesis de la escena no se verifican, se generan errores.

GC plantea una energía que se adapta correctamente a una estructura de superficies planas paralelas al plano de la imagen, pero que con imágenes reales, con planos con pendiente y ruido, puede generar problemas. DP parece ser un poco más robusto a estos problemas pero necesita una etapa de agregado de coherencia inter-scanline para poder dar resultados más aproximados.

Las oclusiones presentes en las imágenes son modeladas y detectadas por ambos algoritmos. Dependiendo de la aproximación que se logre con el mapa de disparidad calculado, tampoco existe una forma de estimar la correcta disparidad en las zonas donde se producen las oclusiones pues se tiene una sola vista de la misma.

Desde el punto de vista de la complejidad, sin duda, GC presenta mayor complejidad que DP, tanto en la estructura de datos que debe manejar, y sobre la cual debe realizar la minimización, como en el tiempo y número de operaciones para resolver el problema.

El tiempo de ejecución de DP es varias decenas de veces menor que GC. En las pruebas que se realizaron, los tiempos promedio fueron de 450 mseg. contra 29.2 seg., promedio en unas 140 pruebas con cada algoritmo.

En cuánto a la complejidad para seleccionar los parámetros para una buena solución, la implementación que dan los autores de GC incluye un modo donde λ se calcula automáticamente para las imágenes de entrada, obteniendo buenos resultados. Pero si se desea hacer una configuración «personalizada» la forma en que influye la variación de los parámetros en los resultados es más sencillo DP.

Dependiendo del uso del mapa de disparidad, la resolución y precisión de la misma, podrán inclinar la decisión por uno u otro, si la escena se puede aproximar por planos paralelos a las cámaras y regiones grandes, la coherencia local que plantea GC le da mayores ventajas, además de dar un mapa de disparidad con poco ruido, y regiones bien definidas. En cambio

si existen regiones con planos inclinados, o direcciones importantes angostas (como los brazos de la lámpara en `tsukuba`), DP puede ser considerado un buena opción.

Es posible agregar una etapa de difusión de la información intra-scanline en DP que mejora los resultados obtenidos y resuelve el principal problema de este algoritmo. Además por los tiempos de ejecución, una etapa de post–procesamiento del resultado de DP, aún dejaría a este algoritmo con ventajas frente a GC.

A pesar que podamos encontrar fallas y posibles mejoras, los resultados globales con imágenes sin agregado de ruido y con valores de los parámetros razonablemente configurados, dan al algoritmo GC como el que presenta los mejores resultados, no solo frente a DP, sino frente al resto de los algoritmos que mejores resultados han tenido en cálculo de disparidad [15].

En este trabajo se realizó un relevamiento de la bibliografía en el área del cálculo de disparidad y sus aplicaciones. Se estudió e implementó uno de los algoritmos con buenos resultados, el cual es citado reiteradamente en la bibliografía. Se estudió otro algoritmo reciente, que utiliza una técnica de aplicación relativamente novedosa en el área, y que es considerado uno de los que genera mejores resultados.

Se realizaron experimentos enfrentando a los algoritmos a pruebas de las cuales no se han encontrado referentes en la bibliografía, utilizando implementaciones de los algoritmos realizadas por otros autores.

Por último se mostró que una etapa de post-procesamiento puede mejorar los resultados de los algoritmos basados en métodos como el descrito por Bobick e Intille.

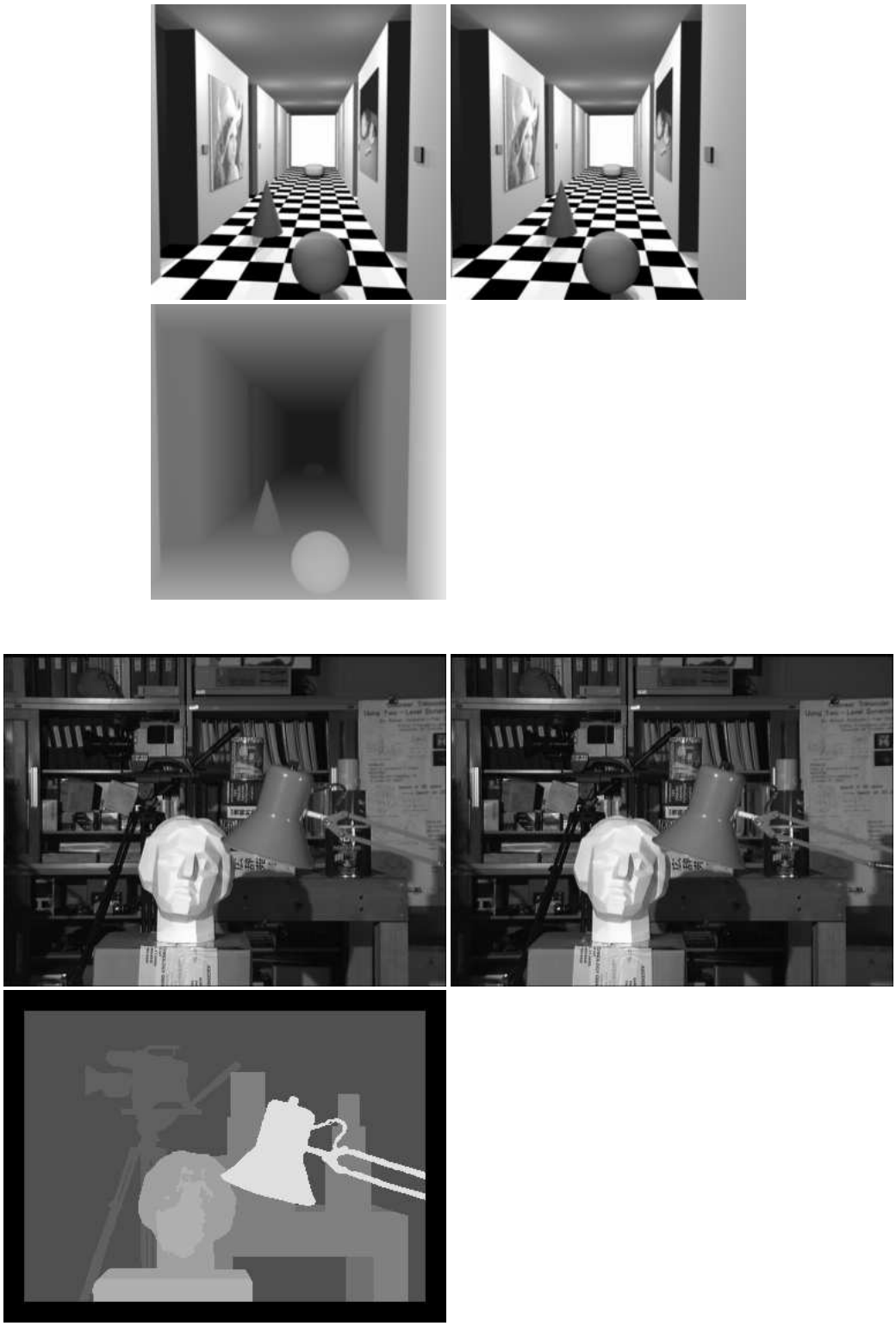
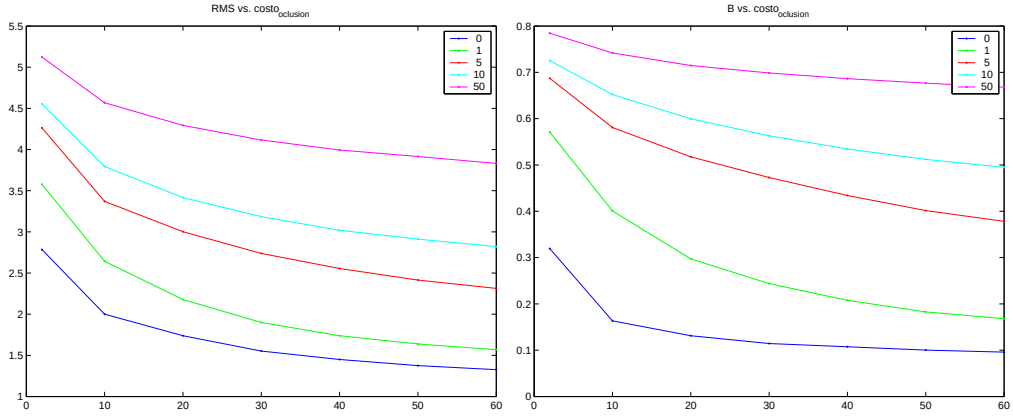
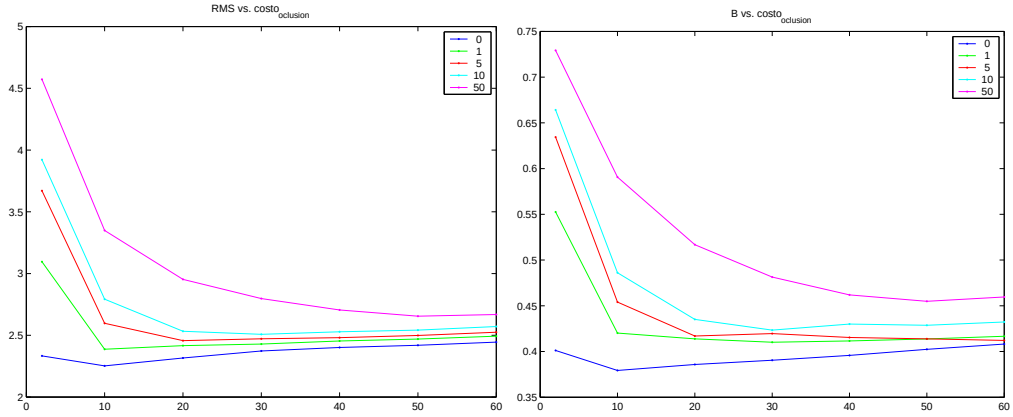


Figura 5.1: **Imágenes estéreo utilizadas en los experimentos.** *Imágenes izquierda y derecha de escenas utilizadas en las pruebas, y mapa de disparidad real para la imagen izquierda de cada escena. Arriba:* `corridor`. *Abajo:* `tsukuba`.



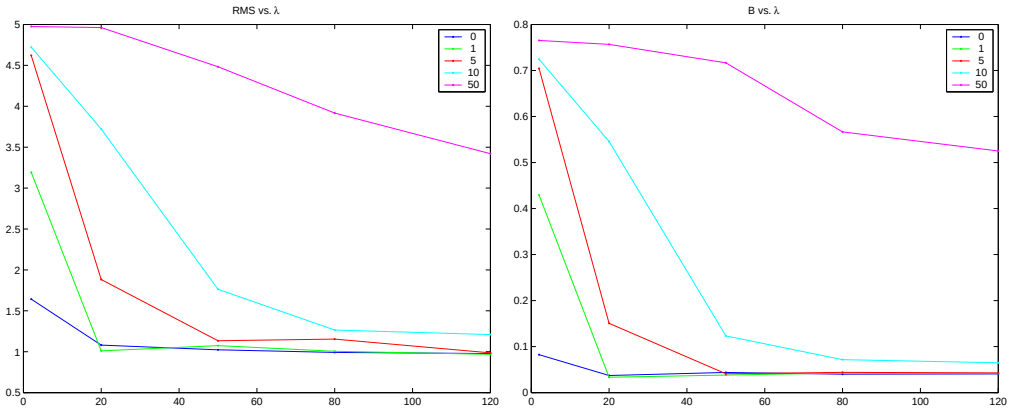
(a) (b)

Figura 5.2: **Resultados obtenidos para la escena `tsukuba` con DP.** *Se varía el costo de oclusión, paramétrico en la potencia del ruido agregado a las imágenes.* (a) *RMS* (b) *B*



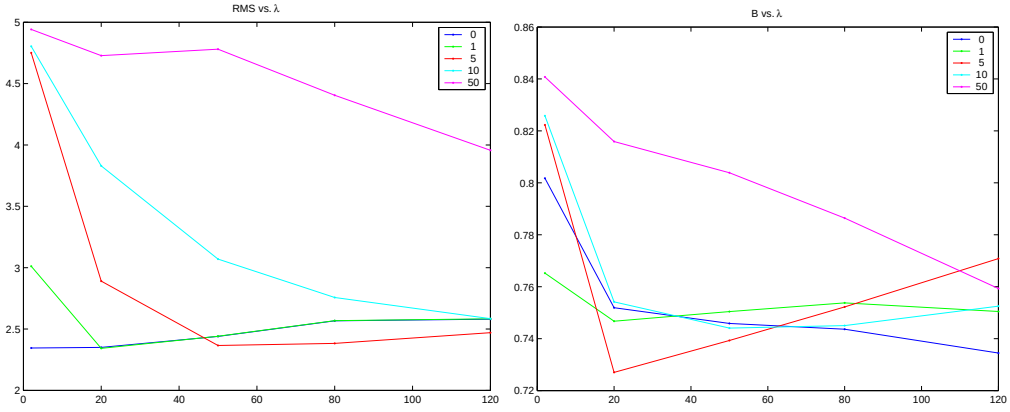
(a) (b)

Figura 5.3: **Resultados obtenidos para la escena `corridor` con DP.** *Se varía el costo de oclusión, paramétrico en la potencia del ruido agregado a las imágenes.* (a) *RMS* (b) *B*



(a) (b)

Figura 5.4: **Resultados obtenidos para la escena `tsukuba` con GC.** *Se varía λ paramétrico en la potencia del ruido agregado a las imágenes.* (a) *RMS* (b) *B*



(a) (b)

Figura 5.5: **Resultados obtenidos para la escena `corridor` con GC.** *Se varía λ paramétrico en la potencia del ruido agregado a las imágenes.* (a) *RMS* (b) *B*

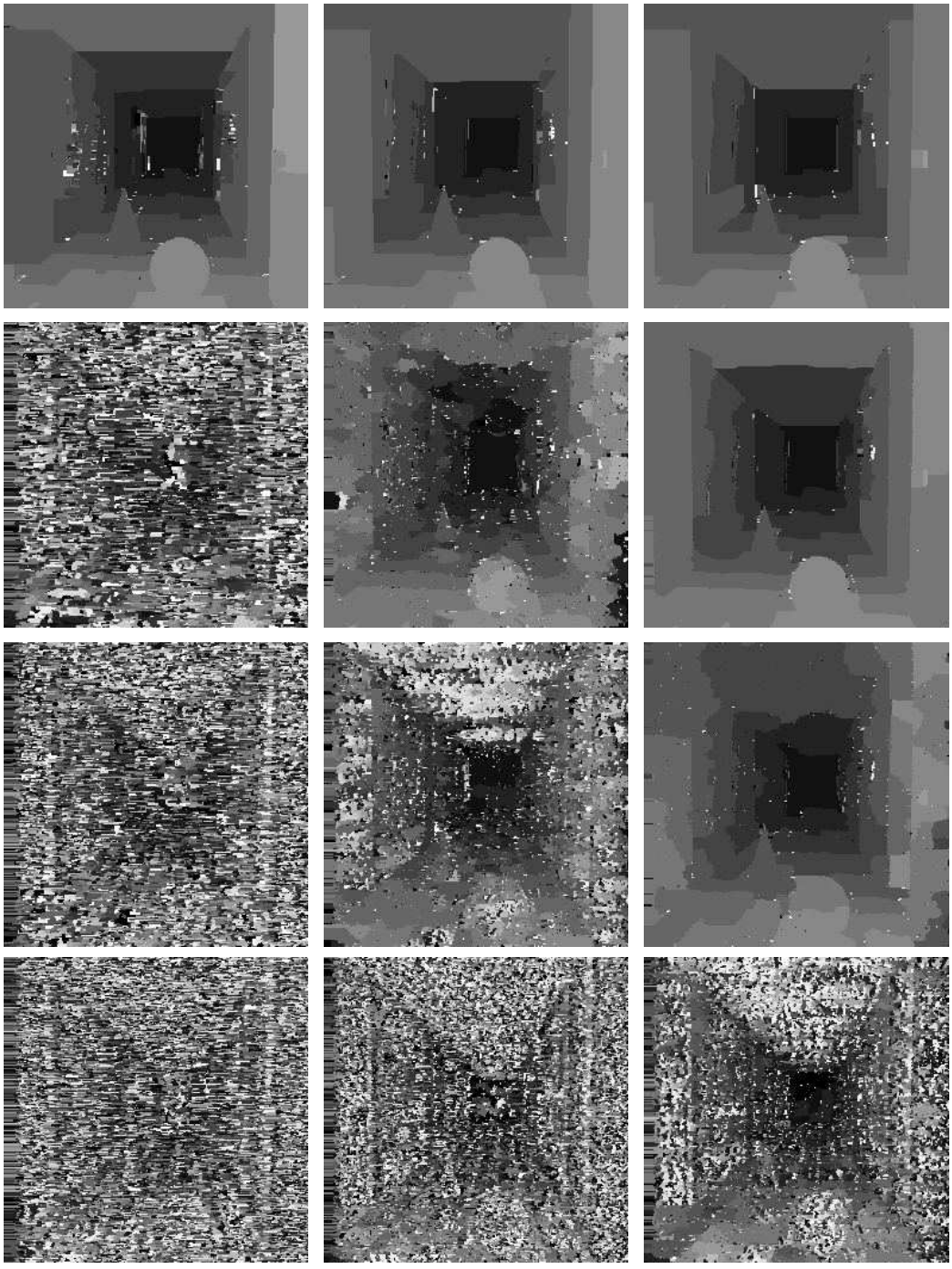


Figura 5.6: **Mapas de disparidad obtenidos para la escena `corridor` con GC.** *Cada columna corresponde a la salida con el mismo λ aumentando la potencia del ruido. Cada fila corresponde a la salida con la misma potencia de ruido aumentando λ.*

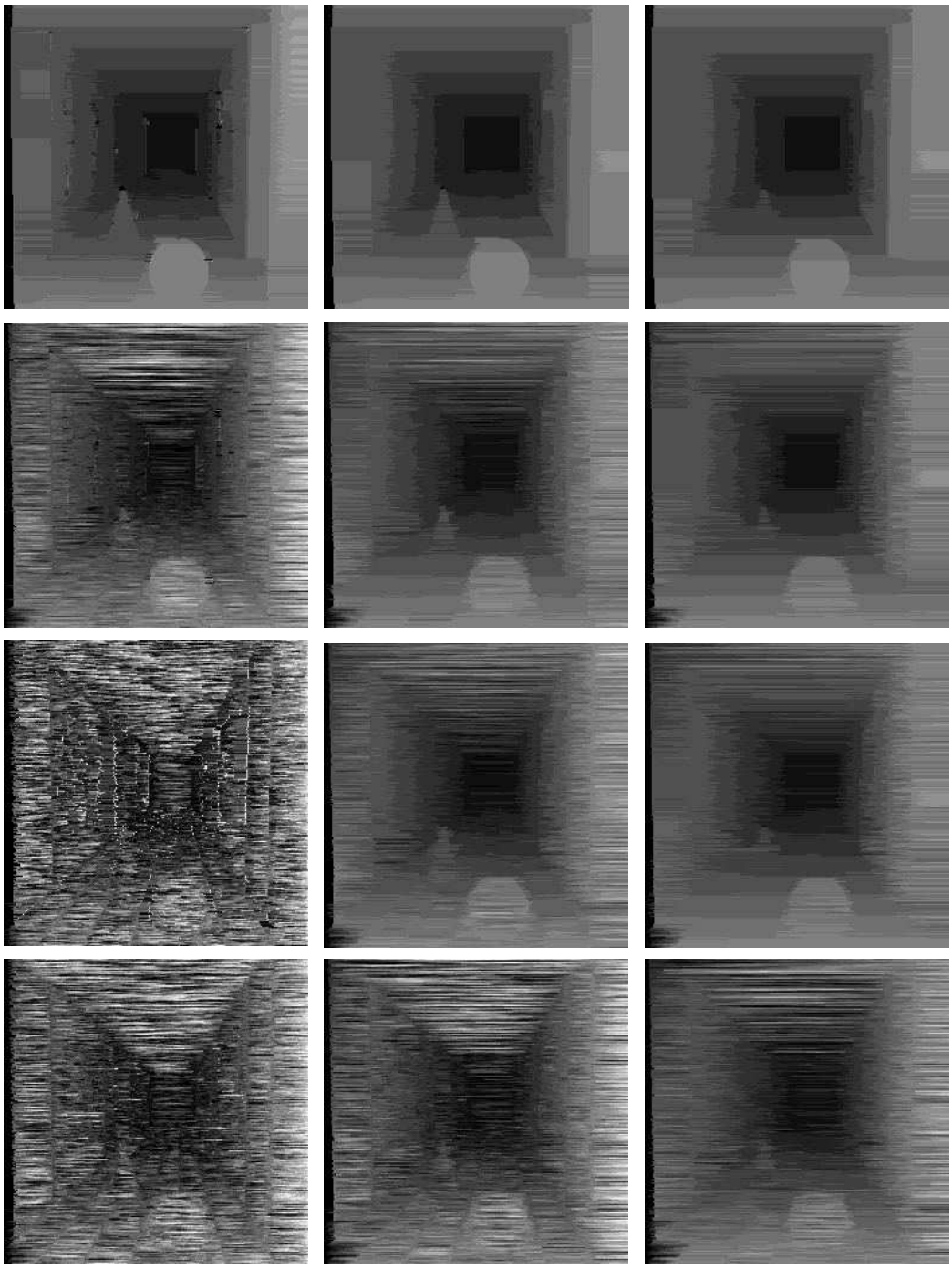


Figura 5.7: **Mapas de disparidad obtenidos para la escena `corridor` con DP.** *Cada columna corresponde a la salida con el mismo costo de oclusión aumentando la potencia de ruido. Cada fila corresponde a la salida con la misma potencia de ruido aumentando el costo de oclusión.*

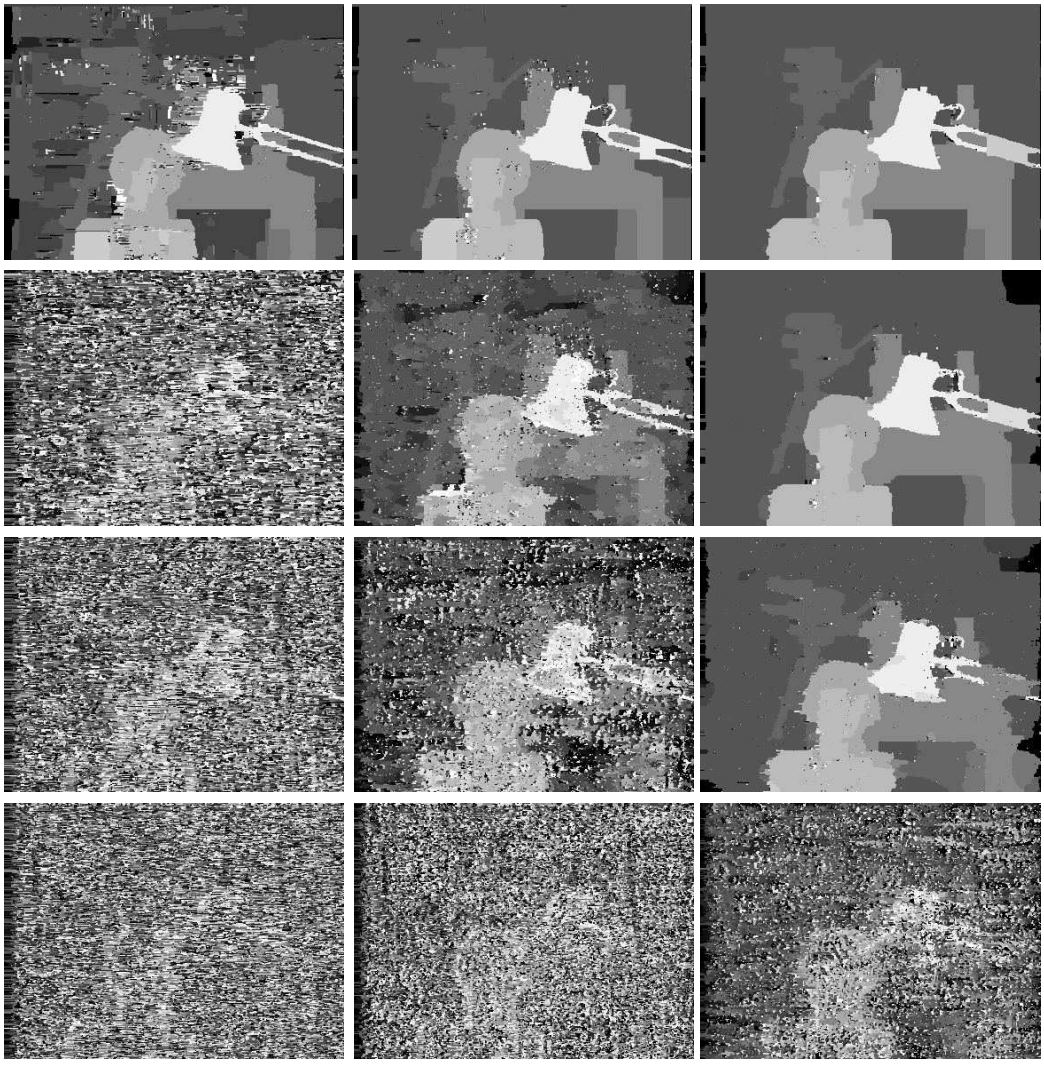


Figura 5.8: **Mapas de disparidad obtenidos para la escena `tsukuba` con GC.** *Cada columna corresponde a la salida con el mismo λ aumentando la potencia de ruido. Cada fila corresponde a la salida con la misma potencia de ruido aumentando λ.*

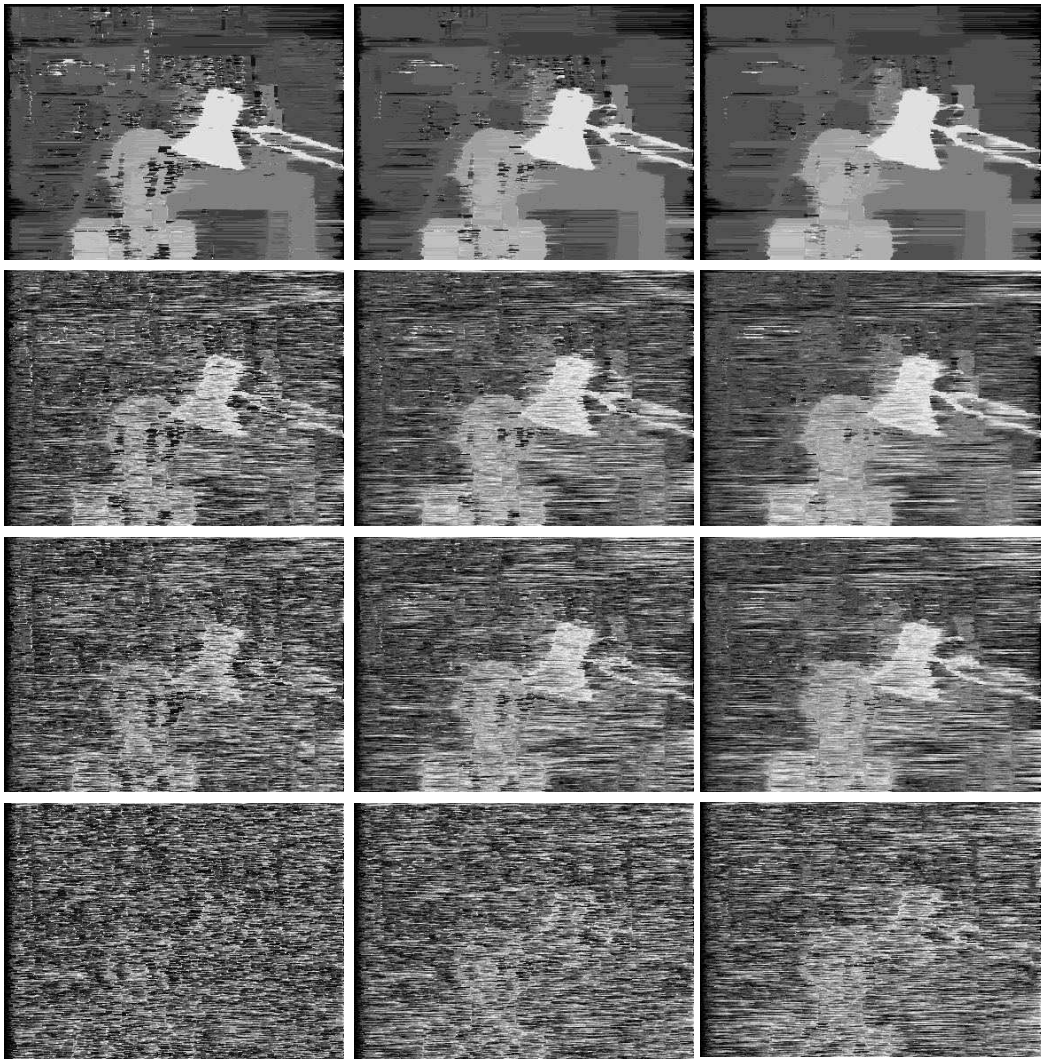
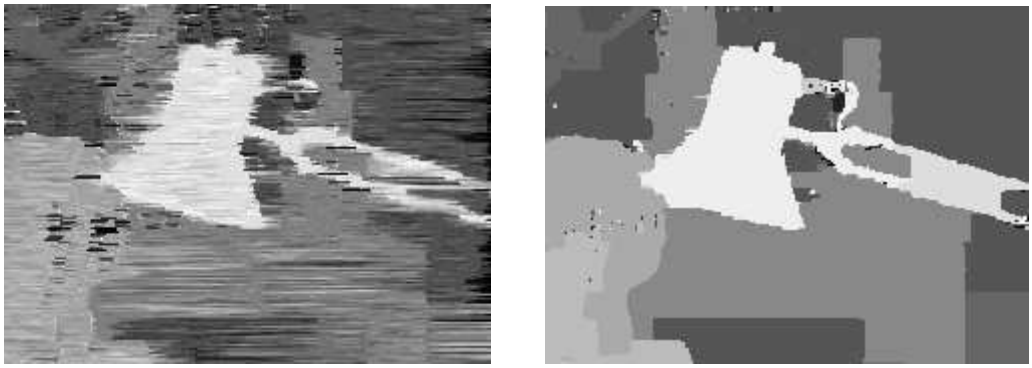


Figura 5.9: **Mapas de disparidad obtenidos para la escena `tsukuba` con DP.** *Cada columna corresponde a la salida con el mismo costo de oclusión aumentando la potencia de ruido. Cada fila corresponde a la salida con la misma potencia de ruido aumentando el costo de oclusión.*



(a) (b)

Figura 5.10: **Detalle de los mapas de disparidad calculados.** *Mapas de disparidad calculados a partir del mismo par de imágenes ruidosas, con* DP *y* GC *con valores de costo de oclusión y* λ *medios.* (a) DP. (b) GC.

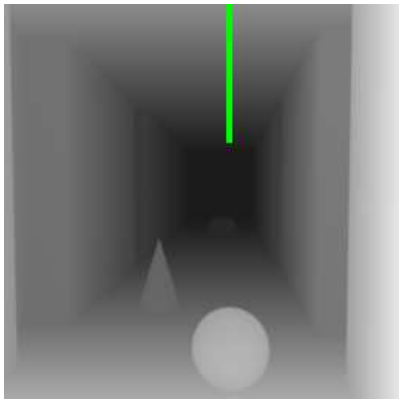
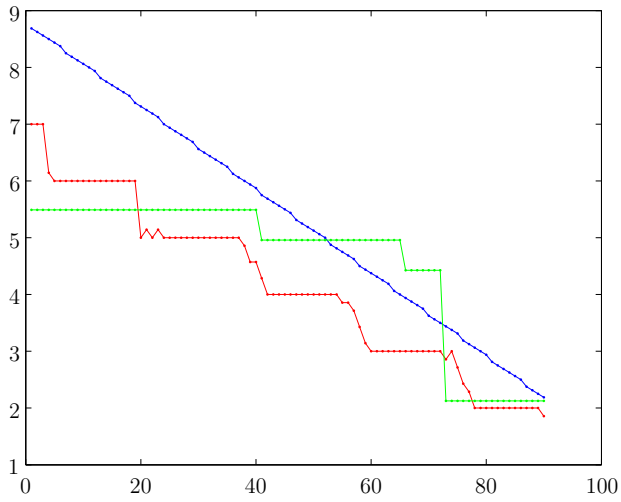
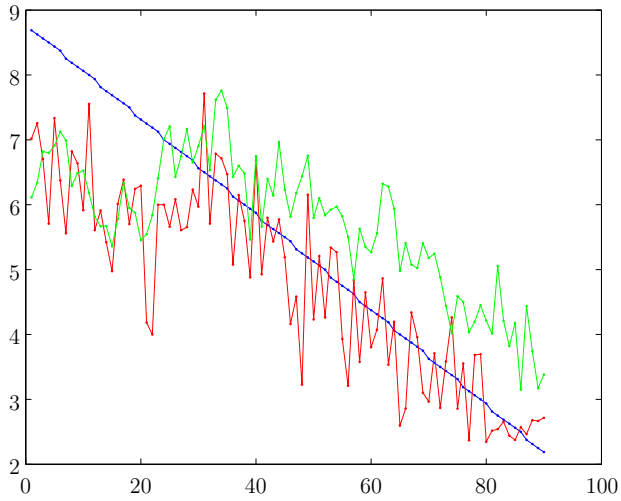


Figura 5.11: *En verde se muestra los puntos de la imagen que definen el perfil de disparidad que se muestra en la figura 5.12.*



(a)



(b)

Figura 5.12: *En azul el perfil de disparidad extraído del techo del corredor que se muestra en la figura 5.11. En verde el mismo perfil medio calculado por* GC, *tomando un promedio de las soluciones con varios* λ. *En rojo, idem para* DP, *tomando un promedio de las soluciones con varios Costo de oclusión.* (a) *Sin ruido.* (b) *Con ruido.*

Parte II

Segmentación de objetos

6 Introducción

Al observar una imagen o un video es inmediato para un humano intentar hacer una descomposición del mismo en distintos objetos. Cada uno de los objetos tendrá una interpretación semántica dentro de la escena, o resaltará por su estructura, textura, color, movimiento u otra característica relevante.

Esta segmentación que resulta sencilla de realizar en los humanos es uno de los mayores desafíos del procesamiento de imágenes y video por computadora o la «inteligencia artificial». La segmentación de secuencias de video consiste en la extracción y seguimiento de los objetos independientes del fondo de la escena a través de los cuadros o *frames* que componen la secuencia.

La interpretación de cual es el objeto de interés depende de la aplicación, y es una tarea de significado semántico que es muy difícil que sea realizada automáticamente por un sistema de segmentación de video de propósito general sin un entrenamiento previo. Es muy común que sea necesaria la interacción con el usuario para realizar una segmentación inicial del objeto de interés a segmentar, o para «enseñarle» al sistema las características del mismo. Esta segmentación inicial no tiene por qué ser exacta permitiendo, por ejemplo, seleccionar los objetos marcando algunos puntos del interior del objeto, suficientes para que permitan diferenciarlo del fondo. El agregado de hipótesis sobre la estructura semántica de la escena permite realizar una segmentación automática de los objetos de interés; por ejemplo, en secuencias de cámaras de vigilancia, donde el fondo es estático y los objetos de interés son las personas que se mueven y las acciones que realizan. Estos métodos a partir de un fondo estático y conocido, se enmarcan dentro de los métodos de *background substraction*.

Algunos autores [51, 52] consideran los objetos de interés como un conjunto de regiones con una cierta topología. Estas regiones son determinadas con algún criterio de homogeneidad, por ejemplo, color o textura. Esta «descomposición» de los objetos en sus partes es tenida en cuenta en uno de los últimos estándares de video, MPEG-4 [53], permitiendo acceder y manipular a los distintos objetos definidos en la secuencia.

Aplicaciones Las aplicaciones de la segmentación de video son diversas, entre otras, citamos: codificación de video, descripción del video para su catalogación y búsqueda, aplicaciones multimedia, post-producción de video, vigilancia, videoconferencia, etc. [54].

La codificación de video es una necesidad a la hora de utilizar video digital dado el inmenso volumen de datos que se genera. Dependiendo de la aplicación final (almacenamiento, procesamiento, transmisión, etc.) el nivel de compresión necesario puede variar en varios órdenes. También varía la carga computacional para la codificación (muchas veces fuera de línea) y la decodificación (necesariamente en tiempo real en muchas aplicaciones). La posibilidad de segmentar los objetos de interés de la escena permite explotar la redundancia temporal presente en el video. Los estándares de codificación de video MPEG-1, MPEG-2 (DVD[1]) [55, 56, 57] y H.261 [56] utilizan la estimación y compensación de movimiento basados en píxeles o bloques (*block matching*) que distan mucho de representar objetos de interés semántico para el sistema visual humano. El estándar MPEG-4 [53] (codificadores de segunda generación), agrega la posibilidad de definir regiones de interés (*objets*, *sprites*), realizando de forma separada una codificación del contorno y de la textura de los objetos, permitiendo asignar distinto nivel de compresión a la textura de los distintos objetos presentes. Por tal motivo, la segmentación de video es fundamental en el proceso de codificación de estos nuevos estándares.

La segmentación automática es sumamente útil en la post–producción de video, la generación de efectos especiales y el agregado de efectos visuales. La posibilidad de seleccionar un objeto y tener la segmentación del mismo a través de los cuadros de la secuencia, reduce enormemente el trabajo manual. Un sistema semiautomático genera una segmentación a partir de una segmentación inicial, la cual puede ser corregida por el usuario en caso de errores.

La descripción del video mediante los objetos presentes, las acciones que realizan, su aspecto, etc., permite facilitar el agregado de anotaciones para la catalogación del video en bases de datos. Esta estructura de datos permite recuperar videos a través de una descripción de las características deseadas, o la búsqueda de material multimedia por contenido. El estándar MPEG-7 [58], formaliza un «lenguaje» común que permite la descripción de la información visual en material multimedia.

En el caso de vigilancia el uso de la segmentación permitiría, entre otras cosas, la detección de «actividades anormales» generando alertas que serían atendidas por el personal de vigilancia, evitando fatigas y desatenciones de los mismos. Entre otras aplicaciones relacionadas con la segmentación de

[1]Digital Video Disc

video y la vigilancia podemos encontrar la detección de humo y fuego [59], vigilancia de autopistas (detección de embotellamientos), etc.

Métodos existentes Las formas de atacar el problema, al igual que sus aplicaciones son variadas. Existen muchas clasificaciones de los métodos de segmentación de objetos en la literatura [60, 61, 62, 56], pero ninguna de ellas concuerda completamente en las categorías ni en la forma de clasificación de los mismos. Megret y DeMenthon [61] presentan una clasificación de las técnicas de agrupamiento espacio–temporal[2], diferenciando tres categorías según el énfasis del uso de la información espacial o temporal. Zhang y Lu [60] realizan una clasificación y revisión del área muy exhaustiva. Plantean una clasificación en dos grandes categorías, métodos basados en movimiento y métodos basados en información espacio–temporal. Cada una de las categorías es sub-clasificada contemplando una amplia gama de métodos, que otras clasificaciones no incluyen.

Muchos métodos de segmentación de secuencias de video son extensiones de métodos de segmentación de imágenes fijas, a los que se le agrega una etapa donde se toma en cuenta la coherencia temporal entre los cuadros de la secuencia de video. Esta coherencia temporal viene dada por las restricciones en los movimientos de los objetos en la secuencia.

En el capítulo 9 se presenta una revisión bibliográfica basada en estas dos clasificaciones y la descripción de algunos trabajos específicos.

Método estudiado El abordaje que se le da al problema de la segmentación de objetos en secuencias de video, en este trabajo, tiene un enfoque basado en tratar la segmentación de objetos como un problema de clasificación. Esta «transformación» se da cuando se cambia la pregunta:

«¿Dónde está el objeto?»

por la pregunta:

«Para todos los píxeles del cuadro:
»¿este píxel pertenece al objeto o al fondo?»

Para poder responder esta pregunta se realiza un modelado de las características del objeto, mediante la estimación de la densidad de probabilidad (PDF) de las características de los objetos (color, posición, textura, movimiento,

[2]Con información «espacial» nos referimos a la información extraída de cada cuadro. Mientras que con información «temporal» nos referimos a la que se obtiene entre los cuadros sucesivos de la secuencia.

etc.) en que se desea clasificar los píxeles del cuadro. Es decir, dadas las características medidas en cada píxel, se estima la probabilidad que pertenezca a cada uno de los objetos o al fondo. Luego se aplica el principio de *Maximum A Posteriori* (MAP) para clasificar cada uno de los píxeles de cada cuadro, asignando el píxel al objeto al que tiene mayor probabilidad de pertenecer dadas sus características.

Esta aproximación tiene sus inconvenientes si objetos diferentes poseen características similares (color, textura, etc.). El uso de múltiples características, en principio independientes entre ellas, agrega robustez al modelado de las distintas clases, permitiendo la diferenciación entre ellas. Esta técnica ha sido muy utilizada para el problema de la segmentación de objetos [62, 63, 64, 65].

Se agrega una etapa de difusión de las probabilidades estimadas antes de la clasificación para mejorar la segmentación.

6.1. Estructura de la Parte II

En el capítulo 7 se presenta el marco teórico del problema de clasificación. En el capítulo 8 se presenta una introducción al color, su formación y su representación. En el capítulo 9 se presenta una revisión bibliográfica del tema de segmentación de objetos. En el capítulo 10 se presenta el algoritmo propuesto y en el capítulo 11 las pruebas realizadas. En el capítulo 12 se presentan las conclusiones de esta parte.

7 Marco teórico

En este capítulo se presentan los conceptos y resultados teóricos básicos de la teoría Bayesiana [66] para los problemas de clasificación («Reconocimiento de Patrones»), «Combinación de Clasificadores» y modelado de características (*features*). Los resultados que se presentan son resultados generales, independientes de la fuente de información con que se esté trabajando. En el capítulo 10 se presenta el algoritmo propuesto utilizando estas técnicas en el problema de segmentación de objetos en secuencias de video.

Se presentan los conceptos básicos sobre la formación de color, y los diferentes modelos de color existentes en la literatura revisada para la representación del mismo en el procesamiento de video e imágenes.

7.1. Regla de decisión de Bayes

Se desea segmentar una secuencia de video en dos clases disjuntas de un conjunto $\Omega = \{\omega_i, i = 1, 2\} = \{O, B\}$, donde O corresponde al objeto de interés y B al fondo (*background*).

Se considera un conjunto de N clasificadores. Cada clasificador tiene asociado un vector de características medidas para cada píxel; llamemos f_i al vector de características asociado al clasificador i-ésimo con $i = 1, \ldots, N$. Estas características pueden ser, por ejemplo, el color, la posición dentro del cuadro, la textura del entorno del píxel, el flujo óptico, la disparidad, etc. La medida de estas características en el píxel m-ésimo forman un *patrón*, $\mathcal{X}^m = \{f_1^m, \ldots, f_N^m\}$.

Cada clasificador, basado en la medida de su característica asociada, es capaz de tomar una decisión respecto a cual clase de Ω asignar el patrón $\mathcal{X}^m$. Para poder tomar esta decisión cada clasificador se basa en la densidad de probabilidad condicional, $p(f_i|\omega_k)$, de la característica f_i para cada una de las clases ω_k, y en la probabilidad de ocurrencia de cada una de las clases, $P(\omega_k)$, llamada también *probabilidad a priori*. La densidad de probabilidad condicional, $p(f_i|\omega_k)$, también recibe el nombre de *verosimilitud* de ω_k respecto de f_i.

La regla de decisión de Bayes brinda una forma de clasificación del patrón $\mathcal{X}^m$, que minimiza la probabilidad de error, utilizando las *probabilidades condicionales a posteriori* de cada clase dado el patrón $P(\omega_k|f_1^m,\ldots,f_N^m)$: se asigna $\mathcal{X}^m$ a la clase ω_j que tiene mayor probabilidad a posteriori, esto es,

$$\begin{gathered}\text{Asignar}\quad \mathcal{X}^m \to \omega_j \quad \text{si} \\ P(\omega_j|f_1^m,\ldots,f_N^m) = \max_k P(\omega_k|f_1^m,\ldots,f_N^m)\end{gathered} \tag{7.1}$$

En el caso particular de dos clases $\Omega = \{O, B\}$ la comparación puede escribirse como,

$$P(O|f_1^m,\ldots,f_N^m) \underset{\mathcal{X}^m\to B}{\overset{\mathcal{X}^m\to O}{\gtrless}} P(B|f_1^m,\ldots,f_N^m) \tag{7.2}$$

que se interpreta como: en caso de que se dé «>» se asigna $\mathcal{X}^m \to O$ y en caso de que se dé «<» se asigna $\mathcal{X}^m \to B$.

Para poder aplicar la ecuación (7.1) es necesario conocer la probabilidad a posteriori, esto se logra utilizando el «Teorema de Bayes»

$$P(\omega_k|f_1^m,\ldots,f_N^m) = \frac{p(f_1^m,\ldots,f_N^m|\omega_k)\,P(\omega_k)}{p(f_1^m,\ldots,f_N^m)} \tag{7.3}$$

La *verosimilitud* de ω_k respecto del patrón medido es una medida de cuán «parecida» es esta clase a la clase verdadera. Esta medida es pesada por la probabilidad a priori de cada clase. El denominador de la ecuación (7.3) es la densidad de probabilidad conjunta de las características, y puede calcularse mediante

$$p(f_1^m,\ldots,f_N^m) = \sum_{\omega_k} p(f_1^m,\ldots,f_N^m|\omega_k)\,P(\omega_k) =$$

$$= p(f_1^m,\ldots,f_N^m|O)\,P(O) \;+\; p(f_1^m,\ldots,f_N^m|B)\,P(B)$$

funcionando como un factor de escala en la ecuación (7.3); por lo tanto no es necesario calcularlo explícitamente para la comparación.

Entonces, sólo es necesario el cálculo del numerador de la ecuación (7.3) para el uso de la regla de decisión de Bayes (7.1), que queda

$$\begin{gathered}\text{Asignar}\quad \mathcal{X}^m \to \omega_j \quad \text{si} \\ p(f_1^m,\ldots,f_N^m|\omega_j)\,P(\omega_j) = \max_k \{p(f_1^m,\ldots,f_N^m|\omega_k)\,P(\omega_k)\}\end{gathered} \tag{7.4}$$

En el caso de dos clases

$$p(f_1^m, \ldots, f_N^m|O)\, P(O) \underset{\mathcal{X}^m \to B}{\overset{\mathcal{X}^m \to O}{\gtrless}} p(f_1^m, \ldots, f_N^m|B)\, P(B) \tag{7.5}$$

Para poder aplicar la ecuación (7.4) utilizando las medidas de todas las características, éstas deben ser utilizadas simultáneamente, es decir, se debe conocer la densidad de probabilidad conjunta, $p(f_1, \ldots, f_N|\omega_k)$, y no alcanza con conocer la densidad de probabilidad de cada vector de características, $p(f_i|\omega_k)\ \forall i$, como lo hace cada clasificador individualmente.

Estimar la densidad de probabilidad a priori de las características dada la clase es independiente del número de características que se estén utilizando, en teoría. En la práctica, al aumentar la dimensión del vector de características, el problema es computacionalmente mucho más costoso, además de aumentar los errores en las aproximaciones y la necesidad de mucho mayor número de muestras para tener estimaciones confiables (fenómeno conocido como la «maldición de la dimensionalidad»).

Esto nos lleva a la necesidad de realizar algunas hipótesis para poder simplificar el modelo propuesto, y realizar una combinación de la información dada por las características en lo que se conoce como *combinación de clasificadores* o *mezcla de expertos*.

7.2. Combinación de clasificadores

La combinación de clasificadores permite organizar el proceso de clasificación dividiendo el problema, utilizando clasificadores más simples, creando implementaciones de soluciones en cascada o jerárquicas [67]. Entre los tipos de clasificadores que se citan en la bibliografía mencionamos: Redes Neuronales Artificiales [66], *Support Vector Machines* [68], *AdaBoost* [69], Bayesianos [66], etc.

En los criterios con que los clasificadores analizan las características medidas existen dos enfoques diferentes [70]. En el primero cada uno de los clasificadores hace uso de una características diferente, y basado sólo en ésta genera su «opinión»; por ejemplo un clasificador utiliza el color y otro el tamaño. En el segundo, diferentes clasificadores utilizan las mismas características medidas, y basados en diferentes criterios generan sus «opiniones»; por ejemplo dos clasificadores de *K Nearest Neighbors*, con diferente valor de K.

La combinación de clasificadores puede hacerse de tres formas diferentes [70]. En el primer caso cada clasificador genera una única etiqueta correspondiente a la clase en la cual ha clasificado el patrón de turno. Estas etiquetas luego se combinan para dar la clasificación final. La forma en que se combinan normalmente es mediante un sistema de votación por mayoría simple. También puede darse un cierto peso a cada uno de los clasificadores teniendo en cuenta la «confianza» que se tiene en cada uno, y realizar una votación ponderada.

Un segundo caso consiste en una variación del anterior, en que cada clasificador genera una lista ordenada de clases a las cuales asignar el patrón. Un sistema de votación puede ser implementado teniendo en cuenta el orden en que fueron asignadas las clases.

El tercer caso se diferencia de los dos anteriores pues cada clasificador no devuelve una clase (o varias); la salida en este caso es la probabilidad condicional de pertenecer a cada clase (probabilidad a posteriori). La forma en que se combinan estas probabilidades puede ser mediante un promedio u otra combinación de las mismas [70, 71, 72, 73].

Dentro de estos últimos los métodos más utilizados son la regla del producto y la regla de la suma. Existen otros esquemas de combinación a partir de estos, como la regla del máximo (*maxmax*), regla del mínimo (*maxmin*), regla de la media (*maxmed*) [70].

7.2.1. Regla del producto

La regla del producto está basada en el enfoque tradicional para descomponer la densidad de probabilidad $p(f_1^m, \ldots, f_N^m|\omega_k)$, considerando que las distintas características son estadísticamente independientes entre sí,

$$p(f_1^m, \ldots, f_N^m|\omega_k) = \prod_{i=1}^{N} p(f_i^m|\omega_k)$$

Sustituyendo la expresión anterior en (7.4), la condición para la decisión queda

$$P(\omega_j) \prod_{i=1}^{N} p(f_i^m|\omega_j) = \max_k \left\{ P(\omega_k) \prod_{i=1}^{N} p(f_i^m|\omega_k) \right\} \tag{7.6}$$

en función de las distribuciones a priori. Aplicando el Teorema de Bayes a $p(f_i^m|\omega_j)$, podemos escribir (7.6) en función de las probabilidades a posteriori,

quedando la regla de decisión del producto,

$$\begin{gathered}\text{Asignar} \quad \mathcal{X}^m \to \omega_j \quad \text{si} \\ P(\omega_j)^{1-N} \prod_{i=1}^{N} P(\omega_j|f_i^m) = \max_k \left\{ P(\omega_k)^{1-N} \prod_{i=1}^{N} P(\omega_k|f_i^m) \right\}\end{gathered} \tag{7.7}$$

Esta regla permite, bajo hipótesis razonables y verificables experimentalmente, simplificar el proceso de clasificación, haciendo uso de clasificadores más sencillos y el uso de vectores de características de dimensiones «manejables». Sin embargo, si las medidas son demasiado ruidosas, puede provocar errores graves. Igualmente la hipótesis de independencia es muy fuerte y puede ser otra fuente de error en el uso de esta regla [72].

7.2.2. Regla de la suma

Otra regla comúnmente usada, es la regla de la suma, que puede deducirse a partir de la regla del producto, considerando una aproximación de la probabilidad a posteriori, [70]

$$P(\omega_k|f_i^m) \simeq P(\omega_k)(1+\delta_{ki}) \tag{7.8}$$

donde $\delta_{ki} \ll 1$. Sustituyendo esta expresión en la ecuación (7.6),

$$\begin{aligned} P(\omega_k)^{1-N} \prod_{i=1}^{N} P(\omega_k|f_i^m) &= P(\omega_k) \prod_{i=1}^{N} (1+\delta_{ki}) \\ &\simeq P(\omega_k) \; + \; P(\omega_k) \sum_{i=1}^{N} \delta_{ki} \end{aligned} \tag{7.9}$$

(en el último paso se expandió la productoria, descartando los términos de orden mayor o igual a dos).

Por otro lado con las ecuaciones (7.8) y (7.9) se llega a

$$\begin{aligned} \sum_{i=1}^{N} P(\omega_k|f_i^m) &= N\,P(\omega_k) \; + \; P(\omega_k) \sum_{i=1}^{N} \delta_{ki} \\ &= (N-1)\,P(\omega_k) \; + \; P(\omega_k)^{1-N} \prod_{i=1}^{N} P(\omega_k|f_i^m) \end{aligned}$$

Entonces

$$P(\omega_k)^{1-N} \prod_{i=1}^{N} P(\omega_k|f_i^m) = (1-N)\,P(\omega_k) \; + \; \sum_{i=1}^{N} P(\omega_k|f_i^m) \tag{7.10}$$

Sustituyendo (7.10) en la ecuación (7.7) resulta la regla de decisión de la suma

$$\begin{gathered}\text{Asignar} \quad \mathcal{X}^m \to \omega_j \quad \text{si} \\ (1-N)\,P(\omega_j) \;+\; \sum_{i=1}^{N} P(\omega_j|f_i^m) \\ = \max_k \left\{ (1-N)\,P(\omega_k) \;+\; \sum_{i=1}^{N} P(\omega_k|f_i^m) \right\}\end{gathered} \tag{7.11}$$

Tax y otros [72] estudiaron el problema de la clasificación utilizando la combinación de clasificadores con las reglas del producto y de la suma. Sostienen que la regla de la suma obtiene mejores resultados en los casos en que las probabilidades a posteriori contienen errores en su estimación; la regla del producto mejora los resultados de la regla de la suma cuando la estimación es buena y la independencia estadística entre las características es real. Asimismo sostienen que el promediado que se realiza al aplicar esta regla reduce los errores en la estimación.

Cuando las estimaciones de las probabilidades a posteriori tienen pocos errores en su estimación, ambas reglas obtienen resultados similares.

Kittler y otros [70] plantean las siguientes cotas para la regla de la suma y del producto, en caso de que las clases sean equiprobables,

$$\prod_{i=1}^{N} P(\omega_k|f_i^m) \leq \min_{i=1}^{N} P(\omega_k|f_i^m) \leq \frac{1}{N}\sum_{i=1}^{N} P(\omega_k|f_i^m) \leq \max_{i=1}^{N} P(\omega_k|f_i^m) \tag{7.12}$$

Vemos que la regla de la suma, ponderada con pesos iguales $\left(\frac{1}{N}\right)$, es menos estricta que la regla del producto; lo cual, junto con la hipótesis que el promediado reduce los errores, permite justificar que obtenga mejores resultados cuando las probabilidades son estimadas con error.

7.3. Modelado de funciones de densidad de probabilidad

La regla de decisión de Bayes y sus variantes, planteadas en la sección 7.1 implican el conocimiento de las probabilidades a priori, $P(\omega_k)$ y las densidades de probabilidad condicionadas, $p(f_i|\omega_k)$. Estas probabilidades difícilmente sean conocidas de antemano dada la estructura del problema; lo cual implica que deberán ser estimadas. Para realizar la estimación, generalmente, se recurre a una muestra de datos representativos de los cuales se

conocen sus características; este conjunto de muestras se conoce como *conjunto de entrenamiento* o *muestras de diseño*; que llamaremos $L = \{x_1, \ldots, x_L\}$, (x_i de dimensión d).

De los elementos que son necesarios estimar, las probabilidades a priori de cada clase, generalmente no plantean mayores dificultades. Pero para la estimación de las funciones de densidad de probabilidad la situación es diferente. Los métodos para la estimación pueden dividirse en dos grandes categorías: paramétricos y no paramétricos. Los primeros consideran un modelo de función (comúnmente gaussiano) del cual estiman los parámetros que mejor ajustan al conjunto de entrenamiento. Los métodos no paramétricos, no hacen ninguna hipótesis sobre la estructura de la función. Estos métodos sirven no sólo para realizar la estimación de las funciones de densidad de probabilidad, sino que también son utilizados para estimar las probabilidad a posteriori directamente, o diseñar clasificadores, por ejemplo, *K Nearest Neighbors*.

7.3.1. Modelos con mezcla de gaussianas

Uno de los principales métodos paramétricos utilizados en la literatura para el modelado de funciones de densidad de probabilidad es la «Mezcla de Gaussianas» (GMM - *Gaussian Mixture Model*).

El método aproxima la densidad de probabilidad por la suma de un número finito de n_G gaussianas de parámetros $\theta_i = \{\mu_i, \Sigma_i\}$,

$$\hat{p}(x|\Theta) = \sum_{i=1}^{n_G} \pi_i \mathcal{N}_{\theta_i}(x) \tag{7.13}$$

donde π_i es la probabilidad a priori de la i-ésima gaussiana, Θ es el vector de incógnitas a determinar

$$\Theta = \{\pi_1, \ldots, \pi_{n_G}, \mu_1, \ldots, \mu_{n_G}, \Sigma_1, \ldots, \Sigma_{n_G}\}$$

y $\mathcal{N}_{\theta_i}(x)$ es un núcleo gaussiano d-dimensional de media μ_i y matriz de covarianza Σ_i

$$\mathcal{N}_{\mu_i,\Sigma_i}(x) = \frac{1}{(2\pi)^{\frac{d}{2}} \, |\Sigma_i|^{\frac{1}{2}}} \, e^{-\frac{1}{2}(x-\mu_i)^\top \Sigma_i^{-1}(x-\mu_i)} \tag{7.14}$$

Para resolver este problema una de las posibles soluciones es aplicar el algoritmo de *Expectation and Maximization* (EM) [66], a partir de una inicialización de los parámetros de Θ. Para una profundización en este algoritmo se puede consultar [66, 74]. El número n_G de gaussianas que se utiliza en la mezcla se considera fijo en el algoritmo EM.

Figueiredo y Jain [74] proponen un algoritmo no supervisado para el aprendizaje de los parámetros de una mezcla de modelos, en particular de gaussianas, determinando el número óptimo de componentes que aproximan la densidad de probabilidad junto con los parámetros de las mismas. Este algoritmo se basa en el principio de (*Minimum Message Length* - MML) para encontrar el modelo que mejor representa los datos de entrenamiento; estableciendo un compromiso entre la complejidad del modelo y la representación de los datos por éste.

Es un algoritmo computacionalmente costoso, debido que evalúa y compara las descripciones del modelo con todos los posibles valores para el número de gaussianas en la mezcla; pero a cambio obtiene la mejor forma para la descripción de los datos mediante un modelo GMM.

7.3.2. Modelos basados en núcleos

La distribución del conjunto de entrenamiento en el espacio de las muestras es un indicador de la densidad de probabilidad que se quiere estimar. Será más probable que un patrón a clasificar se encuentre en una región donde hay muchas muestras del conjunto de entrenamiento.

En este concepto se basa la estimación de densidades mediante núcleos, también conocido como *Ventanas de Parzen* [66].

El método consiste en sumar los aportes de los núcleos, K, centrados en cada uno de los puntos del conjunto de entrenamiento,

$$\hat{p}(x) = \sum_{i=1}^{L} K(x - x_i)$$

El núcleo $K(\cdot)$ es una función en el espacio de muestras de volumen unidad. Comúnmente el tipo de núcleo que se utiliza es un núcleo gaussiano $\mathcal{N}_{0,\Sigma_K}(x)$ de media $\mu_i = 0$ y matriz de covarianza Σ_K.

Cuando el conjunto de entrenamiento es representativo los resultados obtenidos con este método son muy buenos. Una de las ventajas de estos métodos es que pueden implementarse con una convolución, y generalmente tiene menor carga computacional que los métodos paramétricos.

Para el modelado de algunas características, por ejemplo la posición o la forma de los objetos, GMM no es un método eficaz. GMM intenta modelar la densidad de probabilidad mediante la superposición de núcleos elipsoides que difícilmente puedan adaptarse eficientemente a la forma de cualquier objeto. En estos casos, los métodos basados en núcleos dan mejores resultados [52, 63].

7.3.3. Modelos basados en histogramas

Otro método no paramétrico utilizado se basa en histogramas como estimadores de la densidad de probabilidad. El principal inconveniente de este método es la dificultad de trabajar con altas dimensiones. Sin embargo es un método con una carga computacional baja comparada con otros métodos.

Everingham y Thomas [75] suponen la independencia entre las componentes y estiman la densidad conjunta como el producto de las densidades estimadas mediante histogramas. Para esto utilizan el color y la textura como características en cada píxel. Agregan la estimación de la posición con un modelo basado en núcleos. Con este esquema realizan la comparación con el método utilizando mezcla de gaussianas para el modelado, obteniendo mejores resultados con el esquema propuesto.

8 El color

En este capítulo se realiza una introducción al tema de color, su proceso de formación, y los modelos y espacios usados para su representación. No pretende ser una descripción exhaustiva, por lo que además se dejan otras referencias consultadas, que amplían y profundizan el tema.

8.1. El fenómeno psicofisiológico del color

El color de un objeto es una propiedad que percibimos debido a la luz reflejada o emitida por el mismo. Cuando un objeto es iluminado por una fuente de luz, una parte de la energía es absorbida por el objeto (provocando su calentamiento), otra parte atraviesa el objeto (transparencias), y otra parte es reflejada. La longitud de onda de la luz reflejada dará el color con que veremos al objeto.[1] El rango de longitudes de onda del espectro electromágnetico que son captadas por el ojo humano es conocido con el nombre de «luz visible», y está formado por una banda desde los 380 nm (azul) hasta los 760 nm (rojo).

El ojo

La luz proveniente del objeto llega a cada ojo, a la *córnea*. La cantidad de luz que ingresa al ojo es regulada por el *iris* aumentando o disminuyendo el diámetro de su parte central (la *pupila*). Luego es concentrada por el *cristalino*, que funciona como una lente, en la *retina*[2], donde se encuentran dos tipos de células foto-receptoras: los *conos* y los *bastones*. Estas células transforman la luz en señales nerviosas que se envían al cerebro a través del *nervio óptico*[3] [78].

Los conos tienen respuesta a luz de alta intensidad. Existen tres tipos

[1] Los modelos de color modernos (como ACE [76]) sostienen que «el color» que se observa en un objeto depende, no solamente de la longitud de onda reflejada, sino también del entorno del objeto considerado.

[2] La retina es considerada parte del cerebro [77].

[3] El nervio óptico se conecta con la retina en el *punto ciego*, denominado de esta forma pues en esta área de la retina no hay ningún tipo de foto-receptores.

de conos diferentes (L-conos, M-conos y S-conos respectivamente), cada uno «sintonizado» a una longitud de onda diferente (larga, media y corta). Son los responsables de la formación del color y los detalles finos de las imágenes. El número de conos es de alrededor de 6 millones y se concentran alrededor de la *fóvea*.

Los bastones detectan intensidades de luz débiles y son los responsables de la visión con bajos niveles de iluminación. Son varias veces más en número que los conos, alrededor de 100 millones, y se distribuyen prácticamente en toda la retina, excepto donde hay mayor concentración de conos. Los bastones no son sensibles al color, es por esto que, en la noche, con baja iluminación natural, los colores de los objetos aparecen «muy apagados» debido a la poca sensibilidad de los conos en esa circunstancia.

La percepción del color

En la percepción de un color se combinan tres factores diferentes [79]: el *tono*, la *saturación* y la *luminancia*. El tono, o matiz, está asociado a la longitud de onda dominante, la clase de color; es lo que se especifica cuando se describe el color de un objeto: «es azul». La saturación está relacionada con «la pureza» del color, un color es «más puro» (más saturado) cuanto menor cantidad de luz blanca[4] tenga. La luminancia está relacionada con la noción de intensidad, o «cantidad de luz» que tiene el color.

El tono y la saturación de un color conjuntamente constituyen la *cromacidad*. La *luz acromática* es aquella que no tiene componentes de color y sólo se caracteriza por su intensidad, o *niveles de gris*; por ejemplo en los televisores monocromáticos («blanco y negro»).

8.2. Representación del color

Debido a la existencia de tres tipos de receptores de color (L-conos, M-conos y S-conos) la sensación del color que percibimos se forma con la combinación de la información proveniente de cada uno de ellos, los cuales están «sintonizados» a las frecuencias cercanas al rojo, verde y azul, respectivamente. De aquí, la importancia de estos tres colores como veremos enseguida.

La cantidad de información proveniente del «canal» rojo al cerebro se

[4]La luz blanca es luz con igual proporción en todas las longitudes de onda del espectro electromagnético visible.

puede calcular como [80]

$$R = \int E(\lambda)\, S_R(\lambda)\, d\lambda \tag{8.1}$$

donde $E(\lambda)$ es la densidad espectral de potencia de la luz presente, y $S_R(\lambda)$ es la función de respuesta frecuencial («la transferencia») del canal rojo. De forma similar se definen las señales G y B para el verde y el azul, con las funciones $S_G(\lambda)$ y $S_B(\lambda)$, respectivamente.

Entonces para la formación y representación de un color es necesario especificar tres componentes o coordenadas. Los espacios de representación del color varían dependiendo de la aplicación. Así para los monitores y cámaras de video, el espacio más utilizado es el *RGB*, o variantes de éste, como el *YIQ* para la televisión color. Estos espacios se caracterizan por formar los colores mediante una mezcla aditiva de *fuentes de luz* de diferente longitud de onda. Por otro lado en los sistemas de impresión, el espacio de color normalmente utilizado es el *CMYK*. En este espacio, los colores se generan mediante una mezcla sustractiva de *pigmentos*; éstos reflejan una cierta longitud de onda de la luz que los ilumina, con la cual se asocia el color del mismo.[5]

En 1931 la CIE (*Commission Internationale de l'Eclairage*) estandarizó la longitud de onda para los tres colores primarios en 700 nm (rojo), 546.1 nm (verde) y 435.8 nm (azul). Con este estándar y el *diagrama de cromaticidad* es posible la especificación de los colores. El diagrama de cromaticidad de la CIE es una forma de representar el espacio de colores en un plano. La especificación y variación de los colores es apropiada para los sistemas de reproducción, no para el Sistema Visual Humano. Las coordenadas de este diagrama, (x, y, z), se forman en un nuevo sistema de coordenadas normalizado, *CIE XYZ* a partir de las coordenadas en el espacio *RGB*,

$$x = \frac{X}{X+Y+Z} \qquad y = \frac{Y}{X+Y+Z} \qquad z = \frac{Z}{X+Y+Z}$$

donde $(X, Y, Z)^\top$ se calcula como [80]

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0{,}490 & 0{,}310 & 0{,}200 \\ 0{,}177 & 0{,}812 & 0{,}011 \\ 0 & 0{,}010 & 0{,}990 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{8.2}$$

[5] Normalmente se utiliza una estandarización de la mezcla de pigmentos conocida como Pantone Matching System© [81].

donde las coordenadas del espacio *RGB* también están normalizadas a $[0, 1]$.

Así, especificando solamente dos coordenadas (x, y), queda determinada la tercera, y pueden observarse todos los colores visibles del espectro con igual luminancia. La tercera coordenada, z, hace variar la luminancia de los colores. En los bordes de este diagrama se encuentran los «colores puros» o no saturados, es decir que no tienen componente de luz blanca. La recta que une dos colores cualquiera del diagrama, contiene los colores que pueden representarse mediante la mezcla de los mismos.

De esta forma, cualquier sistema de representación del color, define tres colores base, que forman un triángulo dentro de este diagrama. El conjunto de colores que puede representarse con este sistema queda restringido a los colores dentro de este triángulo (Ley de Grasman)[6]. Por esto se ve que no es posible formar todos los colores visibles del diagrama de cromaticidad tomando como base los colores definidos anteriormente, es necesario variar la longitud de onda de los colores primarios para lograrlo [79].

Colores primarios y secundarios (luz y pigmento)

El espacio *RGB* es una de las representaciones comúnmente utilizadas en monitores, cámaras de video y dispositivos de formación de imágenes color. Estos tres colores son llamados *colores primarios de luz*. Este espacio de colores es un espacio *aditivo*, es decir, otros colores se forman sumando los distintos aportes, en las proporciones adecuadas, de estos tres colores. Por ejemplo, el color blanco, corresponde a la suma de los tres colores en su totalidad; el color amarillo se forma sumando el máximo aporte del rojo y del verde y sin aporte del azul. Al ser un espacio aditivo, el blanco puede formarse sumándole toda la componente de azul al amarillo, o lo que es lo mismo, quitando las componentes de azul a la luz blanca se obtienen el amarillo.

Esta última propiedad define al amarillo como el *color complementario* o *complemento* del azul. De la misma forma se obtiene el cian, complementario del rojo, y el magenta, complementario del verde.

Los tres colores complementarios (cian, magenta y amarillo) definen los *colores secundarios de luz*. Forman un espacio *substractivo*, es decir los diferentes colores se obtienen substrayendo las proporciones adecuadas de cada uno de ellos al color blanco. El color negro corresponde a la «suma» de las tres componentes en su totalidad, o sea, restarle todas las componentes de color al blanco. Este espacio es conocido con el nombre de *CMY*, iniciales de los colores cian, magenta y amarillo en inglés (*cyan*, *magenta* y *yellow*).

Los colores secundarios de luz, son los *colores primarios de pigmento*

[6]Este rango de colores se conoce como *gamut* [82].

y los primarios de luz son los *colores secundarios de pigmento*. Esta nueva representación es utilizada en los sistemas de impresión offset, donde los colores se forman debido a la luz reflejada en el pigmento impreso en el papel, o sea, la luz que no es absorbida por el pigmento (substraída al color de la luz que lo ilumina —generalmente blanca—). En los sistemas de imprenta el negro no se obtiene mezclando en iguales proporciones los tres pigmento primarios, sino que se agrega un cuarto pigmento de color negro; es por esto que este nuevo sistema es conocido como *CMYK*, donde la *K* es debida al negro (*blacK*).

La diferencia entre la mezcla de luz y de pigmentos es clara con el siguiente ejemplo. Si se mezclan dos fuentes de luz de colores azul y amarillo se obtiene el blanco, como se comentó anteriormente. Mientras que si se mezclan los pigmentos de colores azul y amarillo se obtiene un color similar al verde.

8.3. Modelos de color

Como se comentó en la sección anterior los modelos de color son variados y dependen de la aplicación. A continuación enumeramos y damos una breve descripción de los más comunes en las aplicaciones relativas al tratamiento de imágenes y procesamiento de video.

RGB

En el modelo *RGB* cada color es representado por las coordenadas correspondientes a las componentes en cada uno de los colores primarios de luz. El modelo se representa mediante un sistema de coordenadas cartesianas, donde cada eje corresponde con cada color en el rango $[0, 1]$. En la figura 8.1 se observa esta representación con el color correspondiente en cada vértice.

Este modelo es muy utilizado en el hardware de reproducción o adquisición de imágenes color, como monitores, escáners, cámaras, etc. Cada imagen consiste en tres planos, cada uno conteniendo la coordenada del color respectivo de cada píxel. Sin embargo no es muy utilizado en la segmentación de objetos en secuencias de video, prefiriendo otros modelos como el *CIE L*a*b** o el *HSV*. Pero es generalmente el modelo del cual se obtienen las imágenes iniciales.

CIE XYZ

El modelo *RGB* no siempre es capaz de reproducir cualquier color con todas las coordenadas positivas, a veces es necesario utilizar la componente

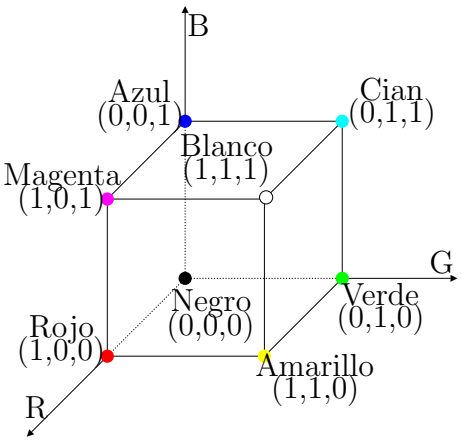


Figura 8.1: **Modelo *RGB***

con peso negativo [80]. Para solucionar este problema, y tener coordenadas siempre positivas, la CIE definió en 1931 el modelo *CIE XYZ*. Este espacio se logra con una transformación lineal a partir del *RGB* con las ecuaciones (8.2).

Este modelo es el modelo básico para todos los estándares de la CIE, con el cual se define el diagrama de cromaticidad, y los nuevos modelos como *CIE L*a*b**, *CIE Luv*, *CIE YUV*, etc.

CIE L*a*b*

En 1976 la CIE crea un modelo refinado del *CIE XYZ*, conocido como *CIE L*a*b**, con el objetivo de linealizar la percepción de las diferencias de colores, o sea, lograr variaciones relativas similares a las percibidas por el SVH. Con este objetivo se intenta reproducir la respuesta logarítmica del ojo. Es un espacio no lineal, donde los colores están referenciados al punto blanco del sistema. En la figura 8.2 se puede ver una representación de este espacio de color. L^* es la luminancia del color en el rango $[0, 100]$, mientras que a^* y b^* definen las proporciones de verde-rojo y amarillo-azul, respectivamente. Para obtener las coordenadas en este modelo se parte de las coordenadas del modelo *CIE XYZ*, al cual se le aplica una normalización para llevar las coordenadas unitarias, (X_n, Y_n, X_n) correspondientes al punto blanco elegido

para el sistema.[7] Así

$$\begin{aligned} L^* &= \begin{cases} 116\, y'^{\frac{1}{3}} - 16 & \text{si } y' > 0{,}008856 \\ 903{,}3\, y' & \text{si } y' \leq 0{,}008856 \end{cases} \\ a^* &= 500\left(f(x') - f(y')\right) \\ b^* &= 200\left(f(z') - f(y')\right) \end{aligned} \tag{8.3}$$

donde

$$f(t) = \begin{cases} t^{\frac{1}{3}} & \text{si } t > 0{,}008856 \\ 7{,}787\, t + \frac{16}{116} & \text{si } t \leq 0{,}008856 \end{cases}$$

y

$$x' = \sqrt[3]{\frac{X}{X_n}}, \quad y' = \sqrt[3]{\frac{Y}{Y_n}}, \text{ y} \quad z' = \sqrt[3]{\frac{Z}{Z_n}}$$

Una de las características principales de este modelo es que la representación del color es independiente del dispositivo en que se despliega la información.

Otra forma de representar este espacio que se utiliza en este trabajo es normalizando las coordenadas de forma que varíen en el rango $[0, 1]$. Lo llamaremos *LabN*.

YIQ

El modelo *YIQ* es una transformación lineal de *RGB* que se utiliza para la transmisión de imágenes en los sistemas comerciales de televisión. Su uso viene dado, fundamentalmente, por la necesidad de mantener la compatibilidad del sistema de transmisión en colores con el antiguo sistema de

[7] Normalmente se utiliza el D65, correspondiente a la radiación que emite un «cuerpo negro» a una temperatura de $6500°K$

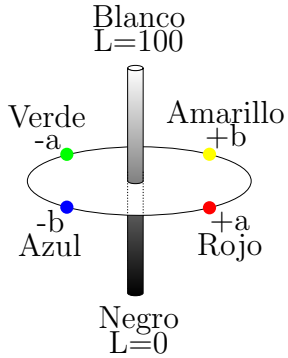


Figura 8.2: **Modelo *CIE L*a*b* simplificado.***

transmisión monocromático. Las tres componentes de este sistema se obtienen como [79]

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0{,}299 & 0{,}587 & 0{,}114 \\ 0{,}596 & -0{,}275 & -0{,}321 \\ 0{,}212 & -0{,}523 & 0{,}311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{8.4}$$

La señal Y corresponde a la luminancia de la imagen, y es la información que precisa un sistema de televisión monocromático para desplegar los cuadros[8]; además logra el mejor aprovechamiento de la sensibilidad del SVH a los cambios de saturación [79]. Cabe resaltar que esta componente de luminancia tiene casi un 60 % de componente en el verde, 30 % de rojo y 10 % de azul; lo cual verifica que el pico de sensibilidad de los bastones (correspondiente a la luz monocromática) se dé alrededor de los 510nm, longitud de onda correspondiente a la «parte verde» del espectro electromagnético [82].

Las componentes I y Q, contienen la información cromática de la señal. Además estas señales están desacopladas de la Y, lo cual permite, por ejemplo, procesar de forma separada las componentes de croma de la de luminancia, sin afectar la otra; lo cual es imposible en el modelo *RGB*.

YUV

El modelo *YUV* es similar al modelo *YIQ*, variando la descomposición de las cromas,

$$\begin{aligned} Y &= 0{,}299\,R + 0{,}587\,G + 0{,}114\,B \\ U &= 0{,}493\,(B - Y) \\ V &= 0{,}877\,(R - Y) \end{aligned} \tag{8.5}$$

La principal diferencia entre estos dos modelos es el consumo de ancho de banda para su transmisión, siendo menor en el caso de *YIQ*, por lo cual fue elegido como modelo para la transmisión de señales de televisión. Sin embargo, *YUV*, es uno de los formatos más utilizados para la representación de secuencias de video para almacenamiento (sin compresión); principalmente debido a que son modelos independientes del dispositivo de reproducción (igual que *YIQ* y *YCbCr*).

YCbCr

Este es el modelo de color utilizado para la especificación del color en el estándar JPEG. Es similar a los dos modelos anteriores de separación en

[8] Aparte de la información de sincronismo de líneas y cuadros.

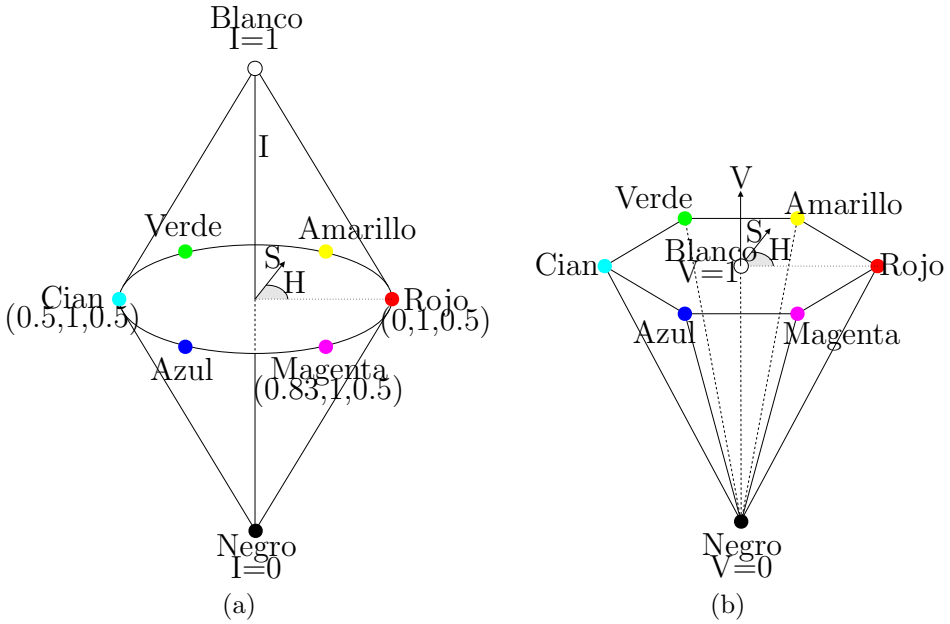


Figura 8.3: **Modelos *HSI* y *HSV*** (a) *HSI* (b) *HSV*

luminancia y cromas.

$$\begin{aligned} Y &= 0{,}299\,R + 0{,}587\,G + 0{,}114\,B \\ Cb &= B - Y \\ Cr &= R - Y \end{aligned} \tag{8.6}$$

HSI

En la sección 8.1 se comentó que el tono, la saturación y la intensidad son las tres características que determinan el color de un objeto, y que la intensidad es independiente de la información cromática del color. El modelo *HSI* se basa en esta descomposición y explota estas propiedades. Las componentes de este modelo son, precisamente, H, el tono (*hue*), S, la saturación del color, e I, la intensidad.

Esta descomposición basada en el funcionamiento del SVH hace de este modelo uno de los más utilizados en el desarrollo de algoritmos de tratamiento de imágenes y procesamiento de video.

La representación de este modelo la podemos ver en la figura 8.3(a). El tono se mide como el ángulo respecto al eje rojo, la saturación se mide como la distancia al eje principal de este espacio, y la intensidad se mide a lo largo del eje principal. La forma de medición es similar a la representación del diagrama de cromaticidad, con la información cromática generada en un plano (dado por la base triangular con la medida de H y S). Y la luminancia

en la tercera coordenada perpendicular a este plano

La transformación del espacio *RGB* a *HSI* no es una transformación lineal; se presentan las ecuaciones para la transformación de *RGB* a *HSI*. Para la deducción de las mismas y la transformación inversa ver Gonzalez y Woods [79]. Las coordenadas normalizadas se calculan como

$$\begin{aligned} H &= \frac{1}{2\pi} \arccos \left\{ \frac{\frac{1}{2}\big((R-G)+(R-B)\big)}{\sqrt{(R-G)^2+(R-B)(G-B)}} \right\} \\ S &= 1 - \frac{3}{R+G+B} \min(R,G,B) \\ I &= \frac{1}{3}(R+G+B) \end{aligned} \tag{8.7}$$

Además,

$$\begin{cases} H = 2\pi - H & \text{si } B > G \\ H \text{ no definido} & \text{si } S = 0 \\ H, S \text{ no definido} & \text{si } I = 0 \text{ ó } I = 1 \end{cases} \tag{8.8}$$

HSV

El modelo *HSV* es muy parecido al modelo *HSI*, basado en las componentes cromáticas de tono H y saturación S, y cambiando la componente de luminancia, por otra de «valor» (V). Este modelo representa los colores con el espacio que se muestra en la figura 8.3(b),

Este modelos ha sido utilizado en trabajos relacionados con la segmentación de caras basados en el color de la piel, por la forma de separación de las cromas y la luminancia [83].

Vezhnevets [83] cita varias propiedades de este modelo, por ejemplo su invarianza frente a cambios de iluminación en superficies mate o cambios de orientación de las fuentes de luz.

RGBN

Este modelo se calcula a partir del modelo *RGB* de forma de obtener una componente proporcional a la «luminosidad» de la imagen [52]

$$s = \frac{R+G+B}{3}$$

y otras dos, r y g, con la información de croma

$$r = \frac{3R}{R+G+B} \quad \text{y} \quad g = \frac{3G}{R+G+B}$$

Es utilizado por Elgammal y otros [52] para la detección de objetos en escenas con fondo estático, evitando la detección de la sombra de los mismos.

9 Revisión bibliográfica

La literatura sobre el tema de segmentación de objetos en secuencias de video es amplia, el problema se ha abordado desde muchos y diferentes criterios, cada uno haciendo énfasis en una característica o aplicación de la segmentación.

Muchos métodos son extensiones de los métodos de segmentación de imágenes fijas, segmentando cada uno de los cuadros y luego imponiendo restricciones temporales. Otro tipo de algoritmos, se basan principalmente en la información dada por el movimiento. Existen diferentes métodos para detectar movimientos o cambios entre los cuadros. Por ejemplo, para detectar movimientos, basados en flujo óptico (*optical flow*) [84], *block matching* [56] y modelos paramétricos de movimiento [56, 85]. Por otro lado, para detectar cambios los más utilizados son la diferencia entre cuadros [60], y modelado del fondo (*background subtraction*) [52].

Existen varias clasificaciones de los métodos de segmentación de video en la literatura [60, 61, 62, 56], cada una agrupando los métodos con diferentes criterios; y las categorías en que se realiza la clasificación no son completamente iguales en ninguna de las clasificaciones consultadas.

En este capítulo se presenta una revisión de la bibliografía basada en las clasificaciones de Zhang y Lu [60] y de Megret y DeMenthon [61]; se dividen los métodos en dos grandes grupos: basados en información espacio–temporal (sección 9.1) y basados en la información dada por el movimiento (sección 9.2). Dentro de cada una de estas clases se hace una sub-clasificación para mejor exposición. En la sección 9.3 se ven métodos que combinan el uso de información espacio–temporal y de movimiento. En la sección 9.4 se presenta el caso particular de secuencias con fondo estático. Y en la sección 9.5 se presentan métodos con aplicación a la vigilancia.

9.1. Métodos espacio–temporales

Megret y DeMenthon [61] sub-clasifican esta categoría en aquellos que hacen énfasis en las características espaciales y su modelado, aquellos que hacen énfasis en las características temporales (agrupamiento de trayectorias), y los que las usan de forma conjunta. Los primeros segmentan cada cuadro y luego agregan coherencia temporal. Los segundos hacen un seguimiento de características particulares, por ejemplo, color, textura, posición, etc., y luego las agrupan según movimientos similares.

9.1.1. Características y modelos

El primer paso de los métodos basados en información espacial es la selección de las características a utilizar (color, textura, posición, flujo óptico, etc.). Una vez seleccionadas el objetivo es poder utilizar estas características para asignar cada píxel en alguna de las clases. El número de clases necesarias para tener una correcta segmentación es uno de los puntos críticos en la inicialización de estos métodos.

Dentro de los métodos basados en modelado de características se mencionan, entre otros: *Region–Merging* [86], *K–means* [56], Modelado con Mezcla de Gaussianas (GMM) [56, 87, 88], y Corte de Grafos [89].

Uno de los principales inconvenientes de estos métodos es la elección del número de clases de forma no supervisada, al igual que la actualización del modelo. Esta actualización incluye la actualización de los parámetros de las diferentes clases, al igual que el número de las mismas, por la desaparición o aparición de una nueva clase.

Modelado de clases

El color es una de las características más utilizadas para describir los objetos. Es robusto a oclusiones parciales, escalado, deformaciones e incluso a pequeñas rotaciones y cambios de profundidad. Las formas de modelar las características son varias, paramétricas y no paramétricas. Dentro de las primeras destacamos la Mezcla de Gaussianas (GMM). De las segundas destacamos el modelado utilizando histogramas [75].

La Mezcla de Gaussianas es un método muy utilizado para modelar la función densidad de probabilidad de una característica cualquiera. Una de las mayores desventajas es determinar el número de gaussianas que formarán la mezcla para la estimación. Figueiredo y Jain [74] proponen un algoritmo

no supervisado para el aprendizaje de un modelo a partir de mezclas. La selección del número de gaussianas es automática, y no requiere una inicialización muy robusta al contrario que el algoritmo EM; utilizan *Deterministic Annealing* para evitar la dependencia de la segmentación inicial de *K–means*.

Los histogramas permiten obtener una estimación de la densidad de probabilidad de forma no paramétrica. La principal desventaja reside en la necesidad de un conjunto de puntos relativamente grande para tener una estimación correcta. Otra desventaja es que no es un método aplicable con características de dimensionalidad muy alta, a menos que se considere la independencia entre las distintas dimensiones.

Hasler y Süsstrunk [90] presentan un método de segmentación, mediante el modelado de los *outliers* utilizando histogramas basados en el color. De esta forma obtienen un modelo con el cual poder diferenciar los objetos sobre el fondo.

Chalom y Bove [64] presentan uno de los primeros trabajos de segmentación de secuencias utilizando un esquema de segmentación basado en la combinación de múltiples atributos (características) de la imagen, y con un modelado estadístico de los objetos.

McKenna y otros utilizan [91] un modelo de la densidad de probabilidad del color basado en mezcla de gaussianas para el seguimiento de objetos. El modelo se inicializa con el algoritmo EM, y se actualiza dinámicamente con los cambios de los objetos. La actualización asume que el número de gaussianas de la mezcla no cambia. El método no obtiene una segmentación fina de los objetos pero permite el seguimiento de los mismos ante variaciones de iluminación, y variaciones en los parámetros de la cámara (auto–iris). Para modelar el color utilizan el espacio de color *hue-saturation*.

Everingham y Thomas [75] modelan la densidad de probabilidad conjunta de la mezcla de tres características espaciales de cada cuadro: color, posición y textura, utilizando histogramas multidimensionales. Para poder aplicar el método con la alta dimensionalidad de la característica utilizada, consideran que las distintas componentes de la característica son independientes entre sí, permitiendo descomponer el histograma multidimensional en una combinación de histogramas unidimensionales. Para modelar la posición o forma utilizan el método de estimación por núcleos [66] con un núcleo gaussiano bidimensional.

Thirde y otros [65] proponen un sistema para segmentación de secuencias en escenas basado en la detección de los objetos presentes. Los objetos son inicialmente segmentados por el usuario, y el sistema realiza de forma automática el seguimiento y segmentación. Para esto se basan en el color y la forma. El modelo de la forma se obtiene mediante estimación por núcleos, realizando la actualización del mismo con filtros α–β, una idea similar a la ecuación (9.2) aplicada al modelo de la forma. Así, el modelo de la forma se actualiza teniendo en cuenta la forma en los cuadros pasados y la estimación en el cuadro actual. Para el color utilizan un modelo basado en mezcla de gaussianas; no realizan adaptación de este modelo, ésta debe ser realizada por el usuario reinicializando el algoritmo.

9.1.2. Métodos basados en clusters o agrupamientos

Los métodos basados en *clusters* (agrupamientos) tienen como objetivo crear regiones agrupando píxeles con características similares. Dependiendo de la aplicación estas regiones pueden ser parte de objetos con significado semántico.

Esta técnica ha sido muy utilizada en segmentación de imágenes fijas [86, 92], considerando únicamente características espaciales de las imágenes. Para secuencias de video puede agregarse al conjunto de características alguna que incorpore información temporal o de movimiento (detector de cambios o vectores de movimiento).

Estos métodos se pueden subdividir en dos categorías, los que se basan en agrupar regiones considerando características similares tomando alguna medida de distancia entre las características, o basados en el ajuste a un modelo de la distribución de las características de los píxeles [61].

Agrupamientos basados en movimiento

Estos métodos estiman el movimiento tomando como base alguna región (bloques o píxeles). Luego se agrupan las regiones que presentan movimientos similares.

Wang y Adelson [93] utilizan una representación de la escena en capas para describir los diferentes objetos que integran el video, para ello calculan un modelo de movimiento afín entre regiones para posteriormente agruparlos por semejanza. Pueden tomarse otros modelos de movimiento que contemplen otro tipo de deformaciones, con la consiguiente complejidad en el número de parámetros y de la estimación. Para evitar problemas en la medición de la similitud de movimientos en el espacio de los parámetros del movimiento, se comparan los errores luego de la compensación de movimiento o la distancia

entre los vectores de movimiento asociados.

Irani y otros [94] y Black [95, 96] utilizan la idea de clasificación en *inliers* y *outliers* del modelo estimado. Normalmente el fondo es considerado como el modelo de referencia y los objetos son segmentados como *outliers* de ese modelo.

El agrupamiento teniendo en cuenta la información temporal es uno de los pasos cruciales en los métodos espacio–temporales. Esta coherencia puede incluirse de varias formas. Por ejemplo, la segmentación espacial inicial en un cuadro es creada a partir de la proyección de la segmentación en cuadros anteriores; y luego refinada con los modelos estimados. Brady y Connor [97] integran las restricciones de coherencia temporal dentro del paso E del algoritmo de EM, para estimar las posiciones de los objetos conocidos en el siguiente cuadro.

Agrupamiento de trayectorias

Los métodos comentados anteriormente utilizan la información temporal entre dos o pocos cuadros consecutivos. Los métodos dentro de esta categoría consideran el movimiento a través de un período de tiempo más largo (varios cuadros de la secuencia). En este caso los movimientos detectados son menos ambiguos debido a que tienen una consistencia mantenida a través de un tiempo considerable.

La estimación de las trayectorias se realiza como un paso previo de la segmentación espacial, utilizando matcheo de puntos relevantes o seguimiento de patrones de textura [61], pero sin utilizar las características espaciales como los métodos anteriores. Por esta razón estos métodos son aplicables en secuencias donde el seguimiento pueda hacerse con poco ruido en las trayectorias, por ejemplo en secuencias con movimiento lentos y con objetos con características relevantes.

Tsai y otros [98] presentan un método en dos etapas. En la primera etapa realizan una segmentación del video en volúmenes, considerando el video como un conjunto de datos tridimensionales con un esquema basado en *Watershed*. Cada volumen es un conjunto de regiones en los cuadros de la secuencia de video. En la segunda etapa utilizan *Markov Random Fields* para modelar las relaciones temporales de los diferentes volúmenes. La segmentación final se realiza agrupando los volúmenes según su relación temporal y espacial, en N objetos que desean ser segmentados (definido por el usuario). De esta manera los objetos se segmentan automáticamente por su coherencia espacial y temporal a través de la secuencia.

9.1.3. Detección de bordes

Los bordes de los objetos son características muy relevantes en la percepción y análisis de imágenes. En la segmentación de objetos en video un modelo de los objetos y del fondo puede aprenderse mediante los bordes detectados. La representación de este modelo es generalmente invariante a cambios de iluminación [61]. Algunas implementaciones en segmentación utilizando bordes pueden verse en los trabajos de Yang y Levine [99], Tsaig [100], Ballerini [101] y Tsaig y Averbuch [102].

La detección de bordes es utilizada generalmente junto con otro tipo de técnicas como *Watershed Segmentation* [102] y *Active Contour Models* [101].

9.2. Métodos basados en movimiento

Los métodos basados en movimiento generalmente están compuestos de tres elementos [60]. El primero es la región de soporte en la cual se estimará el movimiento (píxel, bloque, región, bordes, esquinas, etc.). El segundo es un modelo del movimiento a detectar (flujo óptico bidimensional, modelos paramétricos de movimiento tridimensional, etc.). Y el tercer elemento es un criterio de segmentación (*Maximum A Posteriori*, *Expectation and Maximization*).

Tradicionalmente estos métodos, que sólo utilizan información de movimiento para realizar la segmentación, son útiles en secuencias donde las escenas presentan objetos con movimientos rígidos, o donde los objetos están compuestos de partes que realizan movimientos rígidos.

Zhang y Lu [60] sub–clasifican estos métodos en dos grandes categorías según utilizan movimiento bidimensional ó tridimensional. De los primeros se presentan a continuación los métodos basados en flujo óptico y en detección de cambios. Luego se presenta una revisión de los métodos basados en movimientos tridimensional.

9.2.1. Basados en flujo óptico y en detección de cambios

El flujo óptico es un conjunto de vectores que representan el movimiento de cada píxel entre dos cuadros. Métodos para estimar el flujo óptico entre dos imágenes existen muchos en la literatura, por referencias consultar [56, 84].

Los objetos normalmente se mueven de forma independiente entre ellos y con el fondo. La idea de estos métodos es detectar cambios en el campo de velocidades formado por los vectores de movimiento de cada píxel.

Uno de los principales problemas con estos métodos, al igual que los detectores de bordes, es la sobre-segmentación que se obtiene utilizando sólo esta información. Es posible utilizar alguna técnica de regularización para obtener un campo de velocidades más uniforme, pero tiende a empeorar la precisión de la segmentación. Finalmente, computacionalmente tiene una carga elevada [60].

La detección de cambios entre los cuadros es una primera aproximación a la segmentación de video dado que puede obtenerse una primera aproximación con poca carga computacional.

La principal desventaja de estos métodos es que detectan cualquier tipo de cambios entre los cuadros, así sean dados por movimientos de los objetos o sus sombras, movimientos en el fondo, o ruido. Es por esto que estas técnicas tienen su mayor aplicación en escenas donde el fondo es estático y los cambios se deben principalmente al movimiento de los objetos. En la sección 9.4 se describen más extensamente estos métodos.

9.2.2. Basados en movimientos 3D

Los métodos basados en movimientos bidimensional detectan proyecciones, al plano de cámara, de los movimientos reales que ocurren en la escena tridimensional. Estos métodos no tienen en cuenta ni logran imponer restricciones naturales en los movimientos debido a la estructura que tiene la escena. En los métodos basados en movimientos tridimensionales esta estructura es utilizada.

La carga computacional de estos modelos es mucho mayor que los métodos bidimensionales. Y puede ser necesaria la utilización de más de una cámara o condiciones especiales en los movimientos que realiza la misma.

Zhang y Lu [60] sub-clasifican esta categoría en aquellos que recuperan la estructura a partir de los movimientos (*structure from motion*), y los que plantean un modelo paramétrico de movimiento a estimar.

Dentro de la primera categoría podemos citar aquellos que utilizan la profundidad a partir de la disparidad como característica para la segmentación. Por ejemplo, Gordon y otros [103] estudian el uso de la disparidad para modelar el fondo de la escena y su segmentación. Esta característica es insuficiente para obtener una segmentación fina, objetos que se encuentren muy cerca del fondo pueden no ser distinguibles del mismo; la combinación con el color logra resultados aceptables.

Harville y otros [104] proponen otro algoritmo para la segmentación de secuencias estéreo basado en la combinación de color y disparidad, utilizando cámaras estáticas. Otros trabajos en el mismo sentido pueden consultarse en

los trabajos de Challapali y otros [36], Eveland y otros [37] e Izquierdo [38].

Los métodos basados en un modelo paramétrico consideran que los posibles movimientos que pueden realizar los objetos son rotaciones y traslaciones. Dependiendo del tipo de proyección con que se modela la cámara (afín o proyectiva) el número de parámetros varía. Para las superficies de los objetos se toma un modelo simplificado, considerándolas planos o superficies parabólicas [60]. Wang y Adelson [93] plantean un modelo afín de movimiento; en este caso el número de parámetros a estimar es seis.

9.3. Métodos basados en información espacio-temporal y de movimiento

En esta categoría se encuentran métodos que combinan ideas de métodos anteriores e incluyen al movimiento como una características que describe a los objetos [56].

Piroddi y Vlachos [62] proponen un método en dos etapas utilizando movimiento, textura e intensidad. En la primera etapa, objetos perceptualmente relevantes son detectados basados en alguna de las características e información global del cuadro, para evitar la «sobre–segmentación». En la segunda etapa integran las múltiples características e información local en un método de *region–merging* para refinar la segmentación de la primera etapa.

Khan y Shah [63] proponen un método que combina color, movimiento y posición utilizando una mezcla de gaussianas, y estimación por núcleos, para modelar las densidades de probabilidad. Cada característica tiene asociada un nivel de confianza que se ajusta dinámicamente a cada píxel, por ejemplo, dando menor relevancia al movimiento en regiones donde suele haber errores, como en los bordes de los objetos.

Castagno y otros [51] presentan un algoritmo que combina múltiples características (color, posición, textura y movimiento) con un nivel de confianza. Para la estimación de movimiento utilizan el algoritmo de Lucas y Kanade [105]. Para inicializar los objetos el algoritmo realiza una segmentación automática en regiones (regiones homogéneas según algún criterio); el usuario agrupa un conjunto de estas regiones en el objeto con relevancia semántica que será segmentado y seguido en la secuencia.

9.4. Segmentación de objetos en secuencias con fondo estático

En muchas aplicaciones de vigilancia la cámara está fija, por lo que la escena que se detecta es la misma a menos que existan objetos que se muevan, que provocan cambios en el fondo conocido. Estos objetos son los que tienen interés para estas aplicaciones. Estos enfoques tienen la hipótesis extra de que lo que interesa es lo que se mueve en la escena, pues el fondo no cambia, lo que cambia es la estructura semántica de la escena, por lo tanto es posible plantear un sistema de detección que no requiera de la interacción con el usuario para definir los objetos de interés. Los métodos relacionados con esta configuración particular de la escena se conocen como *background substraction*.

Con esta configuración la detección de los objetos se puede alcanzar tomando la diferencia de un cuadro con el cuadro anterior, las regiones donde hubo un cambio serán las detectadas. Esta comparación tan sencilla sólo será posible para escenas donde los objetos aparecen súbitamente de un cuadro a otro, como los «efectos especiales» de series de televisión de fines de los años 60. En las escenas «reales» hay varias fuentes de cambios del fondo que deben ser considerados en el método de segmentación. Existen cambios en la iluminación de la escena; que pueden ser graduales debido al movimiento del sol, o bruscos al encender o apagar una luz artificial en la escena. Existen cambios debido a pequeños movimientos de la cámara, o movimientos de alguna región del fondo (ramas de un árbol). O pueden existir cambios en el fondo que deben ser tenidos en cuenta, por ejemplo un auto que se estaciona dentro de la escena, el movimiento de alguna parte del mobiliario de la escena, o una persona que queda en una posición estacionaria (dormida) y pasa a ser parte del fondo. Estos diferentes tipos de cambios deben ser considerados en la aplicación para no generar falsas detecciones o pasar por alto algún acontecimiento relevante [52, 106, 107].

Las técnicas más comunes son basadas en la intensidad o color de los píxeles. La más simple es «aprender» el fondo en el cuadro t, $B(t)$, como un promedio de la escena en los últimos N cuadros,

$$B(t) = \frac{1}{N}\sum_{i=1}^{N} I(t-i) = \frac{N-1}{N}\,B(N-1) + \frac{1}{N}\,I(N) \tag{9.1}$$

donde $I(t)$ es el cuadro t–ésimo de la secuencia. Los cambios se detectan umbralizando la diferencia del cuadro actual con el fondo conocido en ese instante. Este método permite detectar zonas donde hay cambios pero no da una segmentación fina del objeto.

Una variante para la estimación del fondo B se basa en cambiar el peso del nuevo cuadro dentro del promedio (9.1)

$$B(t) = (1 - \alpha)\, B(N-1) \; + \; \alpha\, I(N) \tag{9.2}$$

Este método, conocido como *exponential forgetting*, es similar a realizar un seguimiento del fondo con un filtro de Kalman [108, 109]. El valor de α permite controlar la velocidad de actualización (taza de aprendizaje) del fondo a los cambios, o visto de otra forma, controla qué tipos de movimiento afectan el cambio del fondo, movimientos rápidos son descartados bajando el valor de α.

Friedman y Russell [108] y Stauffer y Grimson [110, 111] plantean un modelo de la intensidad/color del fondo, o de cada píxel con una Mezcla de Gaussianas, si la probabilidad de pertenecer al modelo es muy pequeña se considera que el píxel pertenece a otro objeto que no es el fondo. Matsuyama y otros [112] utilizan un método basado en dividir la imagen en bloques, representando cada bloque por su media y su varianza, los bloques que presentan una variación relativa mayor que un umbral son clasificados como pertenecientes al frente. Normalmente los objetos de interés no se forman con bloques lo cual presenta el mayor inconveniente de este método. La decisión del tamaño del bloque afecta directamente el resultado del algoritmo; por un lado con un tamaño muy pequeño el método se aproxima al planteado en [110] considerando la variación de cada píxel. Por otro lado, bloques muy grandes darían una estimación muy gruesa de región que cambia. Sin embargo este procedimiento puede ser un paso inicial para obtener una primera segmentación gruesa de los objetos.

Toyama y otros [113] predicen el valor de intensidad del píxel basados en un historial de sus valores, utilizando predicción lineal con un filtro de Wiener; actualizando los coeficientes en cada cuadro. También se pueden utilizar filtros de Kalman para realizar una predicción lineal [52]. Este modelo genera buenos resultados cuando el rango de variación de la intensidad es pequeño. Para contemplar variaciones mayores se utilizan métodos basados en modelos Markovianos (HMM - *Hidden Markov Models*). Rittscher y otros [114] realizan un modelado que permite discriminar entre el objeto, el fondo y la sombra, aplicado a seguimiento de autos en carretera, con HMM.

Estos métodos pueden tener problemas con las sombras de los objetos (a menos que las modelen expresamente), pues cambian las características del píxel, pero no pertenecen al objeto. Para tratar este problema Elgammal y otros [52] utilizan la información cromática del color que presenta mayor robustez a las sombras.

Para finalizar esta sección mencionamos el método de Mosaico de Imágenes (*Image Mosaicing*) que no presenta un fondo estático, pero obtiene una representación del fondo, eliminando los objetos que se mueven en la escena. Normalmente este tipo de escenas se dan en cámaras que cubren un área amplia con «paneos» y acercamientos de la misma. Irani y Anandan [115] proponen un modelo considerando la escena como una superficie plana bidimensional; los movimientos de la cámara se modelan por un único movimiento global paramétrico entre los cuadros sucesivos, permitiendo proyectar cada cuadro en el siguiente. Los objetos que se mueven dentro de la escena generan un movimiento que no corresponde con el movimiento global detectado. Extienden este método considerando no solamente superficies planas dentro de la escena, proponiendo un método general para tratar diferentes tipos de escena. Esta técnica tiene muchas aplicaciones, entre ellas, codificación de video [116], indexado de video [117], alineación de secuencias [118], etc.

9.5. Aplicaciones en vigilancia

En los sistemas de vigilancia automática se usan cámaras u otro tipo de sensores para monitorear actividades con el objetivo de detectar las acciones que ocurren. La detección automática de estas acciones, y su comprensión, podría facilitar enormemente las tareas de vigilancia de los operadores humanos.

Para los sistemas de vigilancia automática es necesario realizar un seguimiento de los objetos a través de toda la escena incluso cuando experimentan oclusiones o realizan alguna interacción con otros objetos presentes en la escena. Siendo aún más importante el seguimiento automático cuando el tipo acciones que realizan los objetos no son «comunes».

Elgammal y otros [52] realizan una segmentación de las personas (los objetos) mediante la detección de regiones (*blobs*) con una cierta forma y relación entre ellas; por ejemplo, una persona de pie es modelada por un conjunto de regiones con una estructura vertical. Las características de cada una de estas regiones las consideran homogéneas, además de suponer que son independientes entre las diferentes regiones. El seguimiento de estas regiones, considerando las restricciones espaciales de las personas, permite el

seguimiento de diferentes individuos dentro de un grupo; asimismo permiten la ocurrencia de oclusiones entre ellos. El modelado de las densidades de probabilidad es realizado mediante núcleos de Parzen.

Wang y otros [119] realizan un modelo dinámico de la «atención» en la escena para su monitoreo utilizando «contextos» e información pasada para detectar y seguir objetos.

Stringa y otros [120] presentan una aplicación de un sistema de vigilancia para alertar a los operadores cuando un objeto es abandonado en la sala de espera de una estación de trenes.

Otra de las consideraciones particulares que tienen las aplicaciones de vigilancia es la necesidad de trabajar en tiempo real. La bibliografía consultada contiene muchos métodos y algoritmos, pero no todos tienen la capacidad de procesamiento en tiempo real. Dentro de los que pertenecen a esta última categoría se citan [119, 36, 42]

10 Algoritmo propuesto

En este capítulo se presenta el algoritmo desarrollado para la segmentación de objetos en secuencias de video. Este algoritmo plantea la segmentación como un problema clásico de clasificación de patrones, al cual se le incorporan algunas variantes e innovaciones para adaptarse al hecho de trabajar con una secuencia de video. Se utilizan diferentes características del video combinando distintos clasificadores (*mezcla de expertos*) para describir el objeto y se incorpora una etapa de difusión de probabilidades antes de hacer la clasificación final.

El trabajo que se presenta a continuación es parte del comienzo de un proyecto de investigación en «Análisis de Video» que incluye la segmentación semiautomática de objetos en secuencias de video. En este sentido se presentan los primeros resultados en el abordaje de este problema, presentando el algoritmo propuesto, ejemplos, y líneas de trabajo a seguir a futuro.

Por segmentación semiautomática se entiende que la segmentación es asistida por el usuario, quien define el objeto de interés. De esta forma se introduce información semántica de alto nivel en el algoritmo, delimitando las regiones del cuadro inicial que componen el objeto. Además, el usuario tendrá la capacidad de ajustar la segmentación en cada cuadro, en caso de ser necesario, durante la ejecución del algoritmo debido a errores en la misma. También será requerida la actuación del usuario en caso de aparición de otros objetos que puedan afectar la segmentación. Por ejemplo, en la segmentación de un rostro en caso de ser ocluido por una mano, o regiones del fondo que el algoritmo no sea capaz de discriminar del objeto.

10.1. Estructura del algoritmo

Para secuencias genéricas[1], el algoritmo tiene una estructura general basada en los pasos que se muestran en la figura 10.1. La entrada es la

[1]Esto implica que no tienen una característica especial, como ser secuencias con fondo estático.

Dada la segmentación del objeto en el cuadro t, $S(t)$, se realizan los siguientes pasos:

1. Aproximar la posición del objeto en el cuadro $t+1$, $\tilde{S}(t+1)$, utilizando la distribución espacial de $S(t)$ y una estimación de movimiento entre los cuadros.
2. Estimar las probabilidades a posteriori para cada una de las características seleccionadas.
3. Estimar la probabilidad a posteriori de objeto y fondo con una Combinación de Clasificadores (Mezcla de Expertos).
4. Difundir la probabilidad a posteriori con una modificación de la Difusión Vectorial de Probabilidades (ecuación (10.6)).
5. Obtener la nueva segmentación del objeto, $S(t+1)$, a partir de las probabilidades a posteriori difundidas.
6. Actualizar los modelos de las características utilizadas, en caso de ser necesario.

Figura 10.1: **Pasos del algoritmo** *propuesto para secuencias genéricas.*

segmentación del objeto deseado en el cuadro inicial de la secuencia, $S(0)$, junto con los parámetros básicos de configuración del algoritmo. En la figura 10.2 se muestra un diagrama de bloques del algoritmo propuesto.

Se probaron distintas variantes dentro de los diferentes puntos con el objetivo de comprender la influencia de cada uno en el resultado final. Estas variaciones, junto con los resultados de cada una de éstas se analizan en las secciones siguientes.

El planteo de un problema de segmentación de imágenes o secuencias de video como un problema de clasificación de patrones, implica la selección de las características que describan al objeto y permitan diferenciarlo del fondo. Las características que se utilizan en el algoritmo que se presenta son el color y la posición. Estas características aportan información espacial de los objetos, como descriptores del objeto en cada cuadro de la secuencia. La

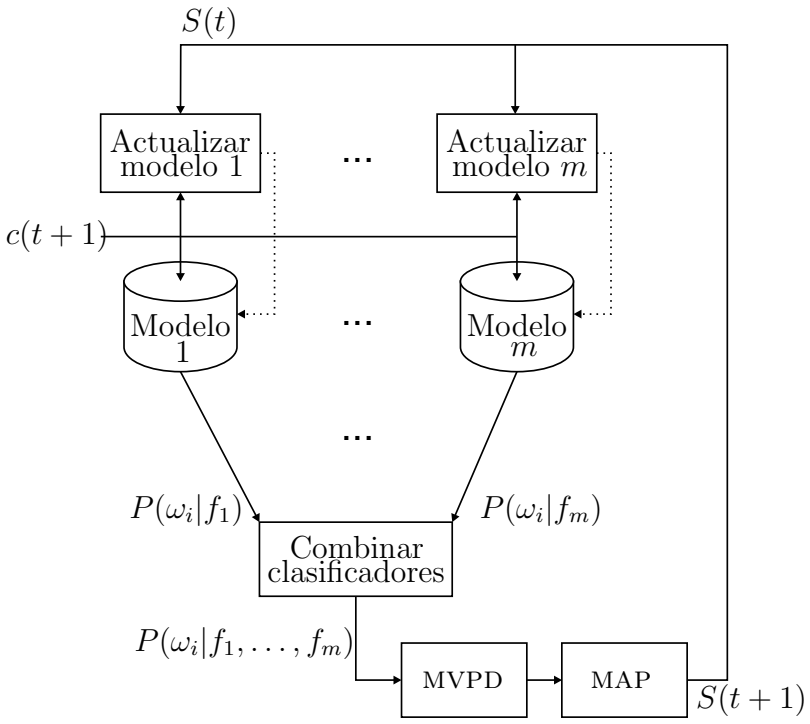


Figura 10.2: **Diagrama de bloques del algoritmo** *propuesto para secuencias genéricas.*

información temporal existente entre los cuadros de la secuencia de video es utilizada a través del movimiento. Éste no es utilizado como una característica para la clasificación mediante la combinación de clasificadores, sino que se utiliza para la propagación de la segmentación entre los cuadros de la secuencias.

Así, el conjunto de características a medir en cada píxel será representado por un patrón $\mathcal{X} = \{f_{\mathrm{C}}, f_{\mathrm{S}}\}$, donde f_{C} es el color y f_{S} es la posición del píxel dentro del cuadro. En el diagrama de bloques de la figura 10.2 se muestra una generalización del método considerando m descriptores diferentes de las regiones.

Aparte de este esquema de características se realizó una prueba utilizando la disparidad de la escena como característica para la segmentación, utilizando los algoritmos presentados en la primera parte de esta tesis. En la sección 11.2 se explica cómo se utilizó y se muestran los resultados obtenidos. Este caso corresponde al uso de tres descriptores en el esquema de la figura 10.2.

En la descripción de cada una de las partes del algoritmo se presentan resultados intermedios correspondientes al proceso de segmentación del décimo cuadro de la secuencia `foreman`. Como se explica en la siguiente sección se estima el modelo de color del objeto y del fondo con la segmentación manual en el primer cuadro.

10.2. Segmentación inicial y estimación de los modelos iniciales de las características

Para la segmentación de video es necesario seleccionar el objeto de interés. Es posible obtener una segmentación inicial automática, pero en general es difícil que tenga significado semántico relevante para la secuencia.

La segmentación inicial se realiza manualmente por el usuario, en forma de una máscara que define los píxeles pertenecientes al objeto de interés. De esta forma se obtienen los datos de entrenamiento de las características a utilizar para el objeto y para el fondo. En la figura 10.3 se muestra la máscara seleccionada y el objeto definido por la misma.

Con los datos de entrenamiento se realiza la estimación de los modelos iniciales de las características, dependiendo de cada una de ellas.

(a) (b)

Figura 10.3: **Segmentación inicial.** (a) *Máscara inicial ingresada por el usuario.* (b) *Segmentación inicial con la máscara dada (se deja ver el fondo de la segmentación con una luminancia menor para visualización)*

10.2.1. Estimación del modelo del color

Para estimar la distribución de probabilidad (el modelo) del color tanto en el objeto como en el fondo se utiliza una mezcla de gaussianas, GMM. Este modelo tiene como parámetro el número de gaussianas que se utilizan en la mezcla. Este parámetro puede ser seleccionado por el usuario o aproximarse mediante el algoritmo de Figueiredo y Jain [74].

Conociendo el número de gaussianas n_G que se utilizarán es necesario hallar los parámetros iniciales (media y matriz de covarianza) para cada una de ellas. La forma de realizar esto es mediante el uso de algún algoritmo de «clusterización»; se optó por *Fuzzy C–Means* [66], un algoritmo iterativo de agrupamiento basado en semejanza de características. De esta forma se obtienen las medias de las gaussianas del modelo, μ_i con $i = 1, \ldots, n_G$. Al mismo tiempo se obtiene n_G subconjuntos disjuntos de los datos de entrenamiento, T_i asociado cada uno a una de las medias, μ_i. Cada una de la n_G matrices de covarianza C_i es estimada como una matriz diagonal[2] a partir de las varianzas de los datos de entrenamiento clasificados en cada una de las gaussianas,

$$C_i = \begin{pmatrix} \sigma_{i1}^2 & 0 & 0 \\ 0 & \sigma_{i2}^2 & 0 \\ 0 & 0 & \sigma_{i3}^2 \end{pmatrix} \qquad \sigma_{ik}^2 = \frac{1}{N_i} \sum_{j=1}^{N_i} (t_{ik}^j - \mu_{ik})^2$$

donde el subíndice $k = \{1, 2, 3\}$ indica la componente de color, N_i es el número de elementos de T_i y t_{ik}^j es el j-ésimo elemento de T_i.

Para estudiar la variación de la clusterización generada por *Fuzzy C–Means* se corrió este algoritmo con el mismo conjunto de entrenamiento y se analizaron los resultados obtenidos. En la l-ésima iteración se reordenan aleatoriamente los elementos de la secuencia de entrenamiento, se corre el algoritmo y se generan n_G centros de regiones («centroides»),

$$\mu^l = \{\mu_1^l, \ldots, \mu_{n_G}^l\}$$

Se calculó la media de los centroides generados $\overline{\mu} = \{\overline{\mu_1}, \ldots, \overline{\mu_{n_G}}\}$ y se calculó la varianza δ de los centroides generados respecto al centroide promedio

$$\delta_i^l = \frac{\|\mu_i^l - \overline{\mu_i}\|_2}{\|\overline{\mu_i}\|_2}$$

con $i = 1, \ldots, n_G$. La máxima variación relativa que se obtuvo fue menor al 0.1 %. Estos resultados confirman la robustez en la generación de los centroides y la poca variación con la condición inicial del método. Lo cual permite utilizarlo considerando que los resultados generados no dependen de la división inicial de los datos de entrenamiento en las n_G gaussianas.

[2]Esto implica que las diferentes componentes del color se consideran independientes.

Los centroides que se generan con este algoritmo son considerados como las medias de las gaussianas que formarán el modelo. Con estas gaussianas determinadas se ejecuta el algoritmo de *Expectation and Maximization* para refinar el modelo. Este será el modelo de la distribución del color del objeto y del fondo que se utilizará para la clasificación.

10.2.2. Espacios de color

Para utilizar el color es necesario determinar el espacio de color en el cual se trabajará. En la bibliografía consultada comúnmente se utiliza el espacio *CIE L*a*b**, debido a su aproximación a la respuesta perceptiva del SVH; también es utilizado el modelo *HSV*.

Para evaluar estos y otros modelos de color a utilizar se realizaron dos experimentos a fin de determinar alguna característica particular que permitiera decidir por alguno de los modelos.

Se tomaron dos consideraciones respecto a los modelos de color que interesa evaluar. Primero, el número de gaussianas necesarias para obtener una estimación ajustada del modelo de color, con este experimento se busca el espacio de color que «compacta» la variación del color con menor número de gaussianas. Segundo, estimar la correlación entre las tres componentes del color. De forma de utilizar un espacio de color cuya descripción mediante gaussianas tenga una matriz de correlación lo más parecida a una matriz diagonal.

El primer experimento evalúa el número de gaussianas necesarias para obtener una estimación ajustada del modelo. Se tomaron los datos de entrenamiento de cuadros de varias secuencias y se determinó el número de gaussianas con el cual se obtiene la «mejor» descripción del espacio de entrada; para esto se utilizó el algoritmo de Figueiredo y Jain.

Los resultados para este experimento se resumen en la tabla 10.1.

RGB	RGBN	HSV	Lab	LabN	YIQ
8	5	4	8	7	8

Tabla 10.1: *Cantidad de Gaussianas necesarias para describir una distribución de colores en diferentes espacios de color.*

En el segundo experimento, se estima la matriz de coeficientes de correlación entre las componentes de los diferentes modelos, para evaluar la

correlación entre las mismas. Esta matriz se calcula como

$$\mathcal{R}(i,j) = \frac{C(i,j)}{\sqrt{C(i,i)C(j,j)}} \qquad i,j = 1,2,3$$

donde C es la matriz de covarianza de los datos. Los valores de esta matriz reflejan la correlación que existe entre la componente i y la j. Los elementos de la diagonal son uno, reflejando la correlación máxima de una componente con sí misma.

La medida que se tomó es el error cuadrático medio entre la matriz de coeficientes de correlación y la matriz identidad

$$\|\mathcal{R} - I_{3\times3}\|^2 = \sum_{i,j} \big(\mathcal{R}(i,j) - I_{3\times3}(i,j)\big)^2$$

donde $\mathcal{R}$ es la matriz de coeficientes de correlación estimada, e $I_{3\times3}$ es la matriz identidad de tamaño 3×3.

Los resultados para este experimento se resumen en la tabla 10.2, relativos al valor obtenido para el modelo *RGB*.

RGB	RGBN	HSV	Lab	LabN	YIQ
100 %	11.4 %	18.7 %	8.2 %	8.2 %	13.6 %

Tabla 10.2: *Medida del error cuadrático medio entre la matriz de correlación y la identidad, relativo al valor para el modelo RGB.*

Estos experimentos no dejan uno de los modelos preferible frente a los otros, pero descartan el uso de algunos, fundamentalmente por el segundo experimento. Desde el punto de vista del número de gaussianas utilizadas el modelo a utilizar sería *HSV*, pero la correlación entre las componentes de color no garantiza que se tengan matrices diagonales. Finalmente se optó por utilizar el modelo *LabN*, es de los más referidos en la literatura consultada y brinda una buena «decorrelación» entre las componentes.

10.2.3. Estimación de la posición

Para estimar el modelo de la posición del objeto en el cuadro $t+1$ se realiza una aproximación con «Núcleos de Parzen» con un núcleo gaussiano convolucionando la segmentación $\tilde{S}(t+1)$. Esta segmentación se obtiene en el primer paso del algoritmo propagando la información de la segmentación en el cuadro t, $S(t)$, y se explica en la sección 10.3. De esta forma se obtiene una

distribución de probabilidad que es máxima en el interior del objeto, decae rápidamente cerca de los bordes del mismo y se anula en puntos alejados del mismo. La distribución de probabilidad del fondo se obtiene como el complemento de la distribución de probabilidad del objeto $P(X \in B|f_p) = 1 - P(X \in O|f_p)$.

Esta distribución permite que el objeto varíe su forma al dejar que los píxeles del borde cambien la clase en la cual clasifican. Este cambio se realizará teniendo en cuenta todas las características del píxel (color, vecindario) y no sólo su posición.

El parámetro de esta estimación es la varianza del núcleo que convoluciona la segmentación inicial. La determinación del mismo depende del tamaño del objeto y de cuánto se le permita variar su forma. Inclusive, es posible que la varianza del mismo varíe teniendo en cuenta la curvatura del objeto, y la rigidez del mismo.

10.3. Propagación de la segmentación

Dada la segmentación del objeto, $S(t)$, en un cuadro t es necesario propagar la información al cuadro siguiente. Los modelos de las características serán tenidos en cuenta para la estimación de las probabilidades a posteriori y eventualmente serán actualizados con la nueva segmentación. El primer paso en esta propagación de la información de la segmentación es obtener una estimación de la posición del objeto en el siguiente cuadro, $\tilde{S}(t+1)$. En este paso se utiliza la información temporal presente en la secuencia de video. Durante el desarrollo del algoritmo se probaron varias formas de «proyectar» la segmentación al cuadro siguiente.

Finalmente se optó por una versión que realiza el seguimiento de la deformación del objeto basado en el cálculo del flujo óptico para obtener los parámetros de un movimiento afín del objeto [93].

Este método obtiene buenos resultados en objetos rígidos o cuya deformación es lenta y pueda ser modelada como una deformación afín. La segmentación de un objeto con componentes que se deforman independientemente, por ejemplo una persona caminando, no es posible con este esquema.

El flujo óptico en cada punto (x, y) del objeto $S(t)$, $(v_x(x, y), v_y(x, y))$, se obtiene como la solución de un sistema de mínimos cuadrados lineal:

$$\min_{v_x, v_y} \sum_{(x,y)\in S(t)} \big(I(x - v_x(x, y), y - v_y(x, y); t + \Delta t) - I(x, y; t)\big)^2 \qquad (10.1)$$

donde el Δt corresponde a cuadros consecutivos.

Haciendo una aproximación de primer orden de (10.1) y usando la aproximación afín

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} a_1 + a_2 x + a_3 y \\ b_1 + b_2 x + b_3 y \end{bmatrix}$$

con las incógnitas $(a_1, a_2, a_3, b_1, b_2, b_3)$ se obtiene un sistema de ecuaciones lineales, de la forma

$$\sum M_1(x,y)\, A(x,y) = -\sum M_2(x,y)$$

donde

$$M_1 = \begin{bmatrix} I_x^2 & I_x^2 x & I_x^2 y & I_x I_y & I_x I_y x & I_x I_y y \\ I_x^2 x & I_x^2 x^2 & I_x^2 xy & I_x I_y x & I_x I_y x^2 & I_x I_y xy \\ I_x^2 y & I_x^2 xy & I_x^2 y^2 & I_x I_y y & I_x I_y xy & I_x I_y y^2 \\ I_x I_y & I_x I_y x & I_x I_y y & I_y^2 & I_y^2 x & I_y^2 y \\ I_x I_y x & I_x I_y x^2 & I_x I_y xy & I_y^2 x & I_y^2 x^2 & I_y^2 xy \\ I_x I_y y & I_x I_y xy & I_x I_y y^2 & I_y^2 y & I_y^2 xy & I_y^2 y^2 \end{bmatrix}$$

$$A = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} \quad \text{y} \quad M_2 = \begin{bmatrix} I_x I_t \\ I_x I_t x \\ I_x I_t y \\ I_y I_t \\ I_y I_t x \\ I_y I_t y \end{bmatrix}$$

En las matrices anteriores el subíndice x, y y t indican las derivadas respecto a las coordenadas espaciales (horizontal y vertical) y temporal, respectivamente. Para que esta aproximación sea más robusta sólo se consideran, en la sumatoria, los puntos de $S(t)$ en los cuales se calcula el gradiente con cierta «confianza». En la implementación realizada sólo se consideran los puntos (x, y) donde el gradiente es relevante, $||\nabla I(x,y)|| > th = 16$.

Luego de la propagación se hacen algunas operaciones morfológicas sobre la segmentación obtenida para obtener una segmentación sin agujeros. En la figura 10.4 se muestra la segmentación obtenida por este método, $\tilde{S}(t+1)$, junto con la segmentación en el cuadro anterior, $S(t)$. Se observa que la segmentación obtenida contiene al objeto de interés y otros píxeles del fondo. Estos último no deben ser tenidos en cuenta en la segmentación final, lo cual se logrará mediante el refinamiento de esta segmentación con los modelos de las características. En los experimentos que se presentan en el capítulo 11 se discuten los posibles errores que puede traer este método, y se proponen algunas soluciones.

(a) (b)

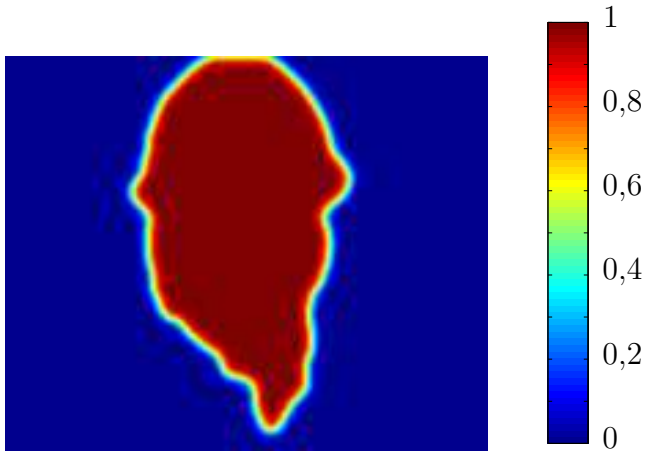
Figura 10.4: (a) *Segmentación del cuadro* $t = 9$, $S(t)$. (b) *Segmentación aproximada por movimiento afín en el cuadro* $t + 1$, $\tilde{S}(t + 1)$.

10.4. Estimación de las probabilidades a posteriori y combinación de clasificadores

La clasificación se realiza a partir de la probabilidad de cada píxel del cuadro de pertenecer al objeto o al fondo. Esta probabilidad se obtiene mediante una combinación de clasificadores, «mezclando las opiniones» de cada uno de ellos. En el caso de este algoritmo las «opiniones» se representan con las probabilidades a posteriori. Los clasificadores corresponden a cada una de las características seleccionadas para la descripción de los objetos.

La probabilidad a posteriori de pertenecer a una región (objeto o fondo) con cada característica se calcula a partir del modelo de la característica en esa región. Para obtener una probabilidad correcta el modelo debe ajustarse a los cambios que pueda tener la región, para lo cual puede ser necesaria la actualización de los modelos de las características. Para el caso de las características seleccionadas, la posición debe ser actualizada en cada cuadro, lo cual se realiza con una estimación basada en un núcleo gaussiano a partir de la segmentación $\hat{S}(t+1)$ hallada con la propagación basada en flujo óptico, como se comentó en la sección 10.2.3. La necesidad de actualización del modelo de color es más compleja, y se discute en la sección 10.7.

Con las probabilidades a posteriori de cada una de las características se realiza la combinación de las mismas para obtener la probabilidad de cada



(a) (b)

Figura 10.5: (a) *Probabilidad del objeto dada la posición en el cuadro, la probabilidad del fondo es el complemento.* (b) *Escala de colores utilizada para visualizar las probabilidades (esta escala se usa en todas las figuras que visualizan alguna probabilidad).*

píxel, de pertenecer al objeto o al fondo. La forma en que se combinan estas probabilidades es mediante la regla de la suma. Así para cada uno de los píxeles X del cuadro se calcula la probabilidad de pertenecer al objeto O, como

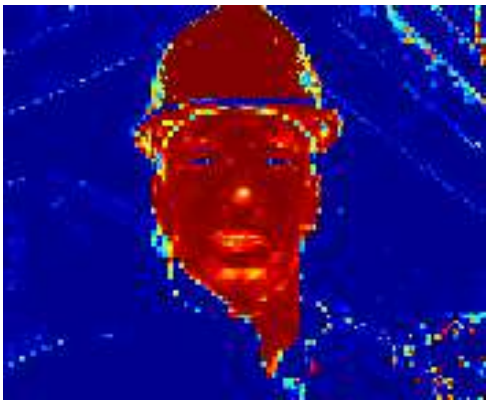
$$P(X \in O|f_c, f_p) = \sum_{i=\{c,p\}} \alpha_i \, P(X \in O|f_i) =$$

$$= \alpha_c \, P(X \in O|f_c) + \alpha_p \, P(X \in O|f_p)$$

donde se agregan las factores α_i, los cuales ponderan la confianza en cada una de las características, estas ponderaciones se calculan de forma tal que la estimación obtenida sea una probabilidad (positiva y que suma uno en cada píxel). Para hallar la probabilidad que el píxel X pertenezca al fondo B, se procede de forma similar.

En la figura 10.6(b) se muestra la segmentación que se obtiene utilizando únicamente el color como característica (clasificación MAP). Se puede ver que hay regiones del objeto (en los ojos o en la visera del casco) que no clasifican según este modelo como objeto. Esto se puede ver también en la figura 10.6(a) donde estas regiones presentan una probabilidad baja como objeto.

En la figura 10.7 se muestra la probabilidad a posteriori del objeto dada por la combinación de los clasificadores seleccionados. Se observa que es poco



(a)

(b)

Figura 10.6: (a) *Probabilidad de pertenecer al objeto según el modelo de color del objeto.* (b) *Clasificación basada en regla de* MAP *según el modelos de color.*

homogénea, presenta píxeles vecinos con probabilidades dispares en regiones centrales de la cara, cercanos a los pómulos o frente. En las regiones comentadas en el párrafo anterior la probabilidad de pertenecer al objeto aumenta, pues se le sumó el aporte de la posición como característica (considerando pesos iguales $\alpha_c = \alpha_p$).

A lo largo del borde entre el objeto y el fondo, en la dirección normal al mismo, existe una variación muy rápida de la probabilidad; lo cual es correcto, pero se observa que el borde es muy irregular a lo largo del mismo.

Con estas probabilidades calculadas se hace la segmentación final, pero antes se le agrega una etapa de difusión de las probabilidades para obtener una segmentación con bordes menos irregulares.

10.5. Difusión de las probabilidades

Un patrón $X \in \Omega$ que puede ser clasificado en κ categorías diferentes tiene asociada, dadas sus características, una probabilidad P_i de pertenecer a cada una de las categorías ($1 \leq i \leq \kappa$). Trivialmente $\|P\|_1 = \sum_{i=1}^{\kappa} P_i = 1$. El método de *Vector Probability Diffusion* de Pardo y Sapiro [121] considera un vector de probabilidades $P(x) = (P_1, \ldots, P_\kappa) \in \mathcal{P} = \{P \in I\!R^\kappa / \|P\|_1 =$

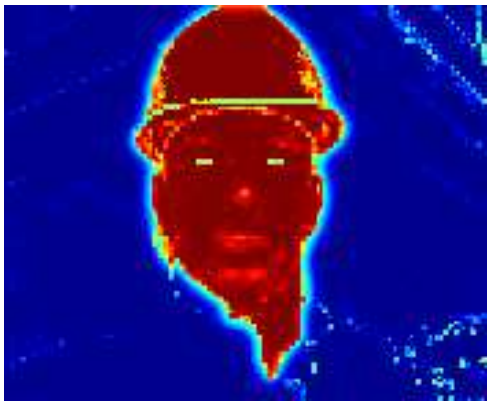


Figura 10.7: **Combinación de clasificadores.** *Probabilidad de pertenecer al objeto dada la combinación de las características seleccionadas (color y posición).*

$1, P_i \geq 0\}$ y minimiza

$$\int_{\Omega} \|\nabla P\|_2 \, \mathrm{d}\Omega = \int_{\Omega} \sqrt{\sum_{i=1}^{\kappa} |\nabla P_i|^2} \, \mathrm{d}\Omega \tag{10.2}$$

restringido a que P pertenezca al simplex $\mathcal{P}$. La difusión que resuelve este problema es

$$\frac{\partial P_i}{\partial t} = \nabla . \left(\frac{\nabla P_i}{\|\nabla P\|_2} \right) \quad i = 1, \ldots, \kappa \tag{10.3}$$

La evolución planteada en esta ecuación, es semejante a la Ecuación del Calor, que rige la forma en que el calor, $h(t)$, se propaga en un cuerpo,

$$\frac{\partial h}{\partial t} = \nabla . \left(k \nabla h \right) \tag{10.4}$$

La dirección en que se difunde esta evolución es ∇h con un coeficiente de conductividad k. En las regiones donde el coeficiente de conductividad es bajo la evolución será lenta, mientras que se difundirá más rápidamente donde el coeficiente de conductividad es alto.

La evolución de la ecuación (10.3) es semejante a la de la ecuación (10.4), donde se identifica $k = \|\nabla P\|_2^{-1}$. Esto implica que la evolución de la probabilidad en cada componente de P se realiza en la dirección ∇P_i, siendo más lenta donde $\|\nabla P\|_2$ es grande. En los bordes del espacio de probabilidad, esta difusión se detendrá, por lo tanto es una difusión anisotrópica en el espacio de probabilidad.

En el caso de segmentación de secuencias de video en objeto/fondo, κ es igual a dos, siendo respectivamente la probabilidad del píxel m-ésimo, X, de pertenecer al objeto y probabilidad de pertenecer al fondo,

$$P(X) = (P(X \in O|\mathcal{X}^m), P(X \in B|\mathcal{X}^m))$$

donde $\mathcal{X}^m$ es el vector de características medidas en X.

Un objeto se diferencia del fondo por varias características que se traducen en la presencia de un borde entre el objeto y el fondo. Sería deseable que la difusión propuesta en la ecuación (10.3) tenga en cuenta *también* los bordes de la imagen, de forma de evitar la difusión de la probabilidad a través de los bordes del objeto, es decir, anisotrópica en el espacio de la imagen.

La modificación que se realiza en este trabajo agrega esta nueva característica, permite la difusión en la dirección del borde del objeto, mientras que penaliza la difusión en la dirección normal, u.

Supongamos que tenemos un campo de direcciones u, normal a los bordes de la imagen y normalizados en su longitud. Para detener la difusión en los bordes del objeto se modifica la dirección de difusión restando la componente paralela a los bordes, o sea, paralela a u. La nueva dirección de difusión será

$$\nabla P_i - \langle u, \nabla P_i \rangle u \tag{10.5}$$

Incorporando esta variante a la ecuación (10.3) la ecuación de la *Modified Vector Probability Diffusion* (MVPD) queda (la deducción se presenta en el Anexo D):

$$\frac{\partial P_i}{\partial t} = \nabla . \left(\frac{\nabla P_i - \langle u, \nabla P_i \rangle u}{\sqrt{\sum_{i=1}^{m} \left\| \nabla P_i - \langle \overrightarrow{u}, \overrightarrow{\nabla P_i} \rangle \overrightarrow{u} \right\|^2}} \right) \quad i = 1, \ldots, m \tag{10.6}$$

Los bordes de la imagen se seleccionan tomando, por ejemplo,

$$u = \left(\frac{L_x}{\sqrt{\beta^2 + L_x^2 + L_y^2}}, \frac{L_y}{\sqrt{\beta^2 + L_x^2 + L_y^2}} \right)$$

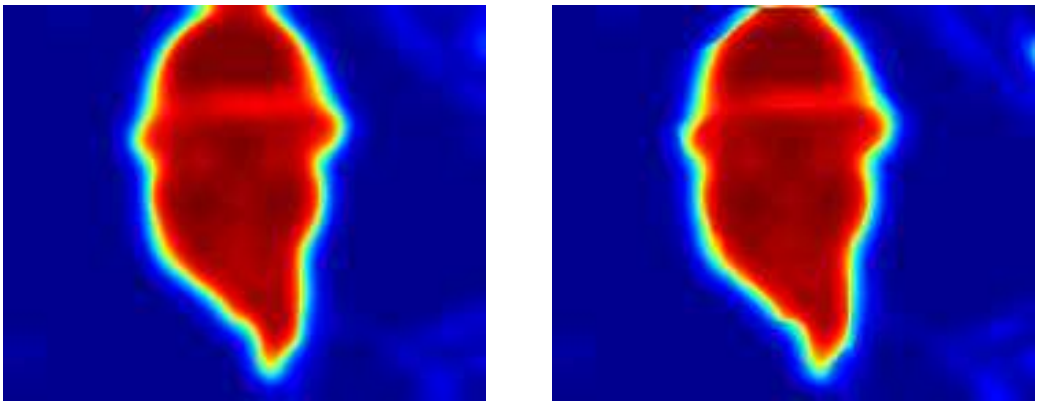
donde L es la componente de luminancia de la imagen y β es un parámetro para evitar singularidades en regiones con poca textura, y para determinar la relevancia de los bordes, considerando aquellos en que $||\nabla L(x, y)|| \gg \beta$.

Para detener la iteración de la ecuación (10.6) se considera un criterio de reducción del valor del funcional dado por la ecuación (10.2) a una fracción de su valor inicial. El factor dado por la ecuación (10.2) es una estimación de la «suavidad» de la solución. También se incluye un término de parada considerando el número de iteraciones.

10.5.1. Efecto de la MVPD

La difusión de probabilidades se realiza a las probabilidades a posteriori generadas por la mezcla de expertos. Al difundirlas la probabilidad tenderá a ser homogénea en cada una de las regiones deteniendo la difusión tanto en los «bordes» del espacio de probabilidad como en los bordes de la imagen.

En la figura 10.9 se muestran los efectos de la difusión de probabilidades, para la probabilidad del objeto (la probabilidad del fondo es el complemento). La difusión propaga la información del vecindario mitigando los

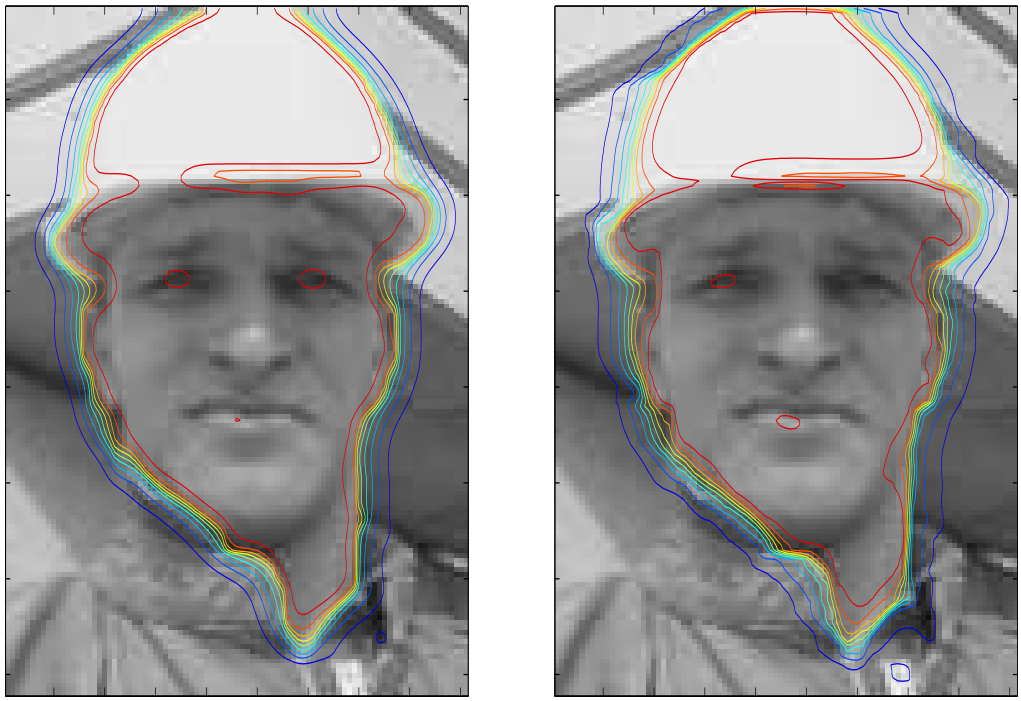


(a) (b)

Figura 10.8: **Efecto de la difusión de probabilidades.** *Resultado de la difusión de las probabilidad del objeto, en la figura 10.7* (a) *con la* VPD *clásica, en la* (b) *con la* MVPD *propuesta.*

efectos comentados sobre la probabilidad generada por la combinación de los clasificadores.

En la figura 10.8(a) se muestra la salida luego de la difusión de probabilidades clásica, y en la figura 10.8(b) se muestra la salida luego de la difusión modificada planteada en este trabajo. Se siguen obteniendo las ventajas de la difusión de probabilidades, mientras que se mejora levemente la respuesta del algoritmo en algunos bordes de la imagen. Por ejemplo, en la parte superior izquierda del casco, en la visera del mismo, o en la pera. Estos fenómenos se ven más claros en las figuras 10.9(a) y 10.9(b) donde se muestran las curvas de nivel de la probabilidad del objeto; se ve que se ajustan mejor a los bordes de la imagen. Se observa que las mayores diferencias se encuentran en las regiones de probabilidad alta (0.9-0.8) o baja (0.1-0.2), pero donde las probabilidades son medias, donde se dan las decisiones más comprometidas para la segmentación, la diferencia no es apreciable. Por esto como se verá en los



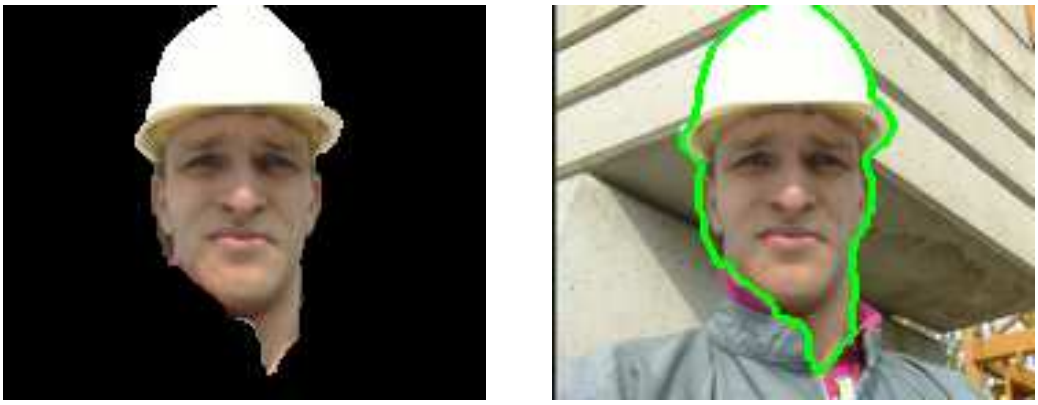
(a) (b)

Figura 10.9: **Efecto de la difusión de probabilidades.** (a) *Líneas de nivel para la probabilidad con* VPD (b) *Líneas de nivel para la probabilidad con* MVPD

experimentos las diferencias entre ambas difusiones no son apreciables dado que las probabilidades definen bien los bordes entre el objeto y el fondo, y la modificación introducida no produce variaciones importantes en los píxeles del borde.

10.6. Segmentación

Con las probabilidades difundidas según la ecuación (10.6) se realiza la segmentación final del cuadro aplicando la regla de MAP. La segmentación final para el cuadro se muestra en la figura 10.10. Dadas las operaciones que se realizan en el algoritmo los objetos que se segmentan son regiones conexas del cuadro.



(a) (b)

Figura 10.10: *Segmentación final.*

10.7. Actualización de los modelos

La principal componente del algoritmo propuesto es la clasificación de cada píxel según los modelos estimados de objeto y fondo. Para que esta clasificación no introduzca errores que pueden propagarse a través de los cuadros de la secuencia es necesario que los modelos de las características sean buenos descriptores de los datos. En este sentido no alcanza con que las características seleccionadas para la descripción permitan la «separación» de los objetos, sino también que su modelo esté ajustado, y sea capaz de contemplar cambios que puedan tener los objetos durante la secuencia. Para que esto último se cumpla es necesario actualizar los modelos en caso que éstos varíen.

La actualización de los modelos es uno de los componentes delicados del sistema. Una actualización muy fina de los parámetros de los modelos puede llevar a un sobre-ajuste (*overfitting*) de los mismos, que resulte en un modelo demasiado específico, y no contemple nuevas variaciones de los datos. Una mala actualización de los modelos, puede llevar a perder las características del objeto y fallar la segmentación, incluyendo puntos del fondo o recortando el objeto.

Dependiendo de la característica y su modelado, la actualización en todos los cuadros de la secuencia puede ser muy costosa, el caso característico es el modelo de color mediante la mezcla de gaussianas. Es posible realizar una actualización del modelo mediante la ejecución del algoritmo de EM; para esto habría que determinar bajo qué condiciones es necesario realizar esta actualización. Incluso es posible ejecutar el algoritmo de Figueiredo y

Jain para volver a calcular el número óptimo de gaussianas y sus parámetros, lo cual puede implicar una carga de cálculo que no sería la requerida para el caso.

La actualización del modelo de la posición es necesaria en cada cuadro, y así se hace. Para el modelo de color es más difícil determinar la necesidad de actualización, en los experimentos que se muestran en este trabajo no se actualiza el modelo de color. En los ejemplos que se muestran en el siguiente capítulo se puede ver que en varias de las secuencias el modelo inicial es suficientemente robusto como para obtener una segmentación aceptable en secuencias de varios cientos de cuadros.

En la literatura consultada existen varios ejemplos de métodos de actualización del modelo de mezcla de gaussianas. Como referencia se cita el trabajo de McKenna y otros [91], donde utilizan un modelo de la densidad de probabilidad del color basado en mezcla de gaussianas para el seguimiento de objetos. El modelo se inicializa con el algoritmo EM, y se actualiza dinámicamente, considerando constante el número de gaussianas que integran la mezcla. Bajo esta hipótesis realizan la adaptación de los parámetros de las gaussianas; de esta manera suponen que, aunque sea necesario actualizar las gaussianas de la mezcla, la cantidad de ellas no influyen en la descripción. Para realizar la actualización, primero realizan una estimación de los parámetros en el cuadro t, $\Theta(t) = (\pi(t), \mu(t), \Sigma(t))$ donde consideran únicamente los píxeles del cuadro t. Luego obtienen el modelo final mediante una suma ponderanda con el modelo en t, $\Theta(t)$ y el modelo estimado con los últimos $L-1$ cuadros $\Theta_L(t)$. Variando el parámetro L controlan la capacidad de adaptación del modelo.

11 Experimentos y discusión

Durante el desarrollo del algoritmo se probó con secuencias de video comunes en la literatura de codificación de video; las pruebas consideraron las secuencias `foreman`, `flower garden`, `container` y `mobile & calendar`. En estas secuencias los objetos segmentados cumplen con las hipótesis que exige el algoritmo, en el estado actual de desarrollo. Son secuencias en colores, con un muestreo temporal alto, con objetos que presentan pocas deformaciones entre cuadros consecutivos, pero pueden tener una variación importante a lo largo de la secuencia.

11.1. Experimentos con el algoritmo propuesto

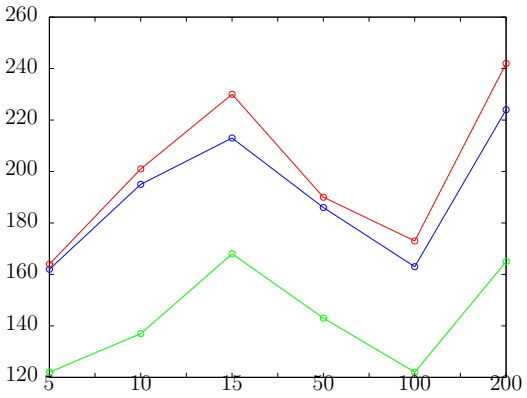
Las segmentaciones que se muestran en las figuras tienen un borde verde alrededor del objeto segmentado, es decir, los píxeles en verde pertenecen al fondo del cuadro.

`foreman`

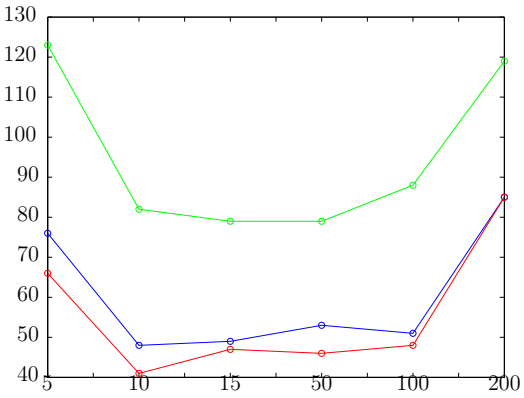
La secuencia `foreman` presenta un actor que se mueve y habla directamente a la cámara. El objeto a segmentar es la cabeza, incluido el casco que utiliza. El personaje habla, gira su cabeza, se acerca y aleja de la cámara. Esto provoca deformaciones varias en el objeto. En la figura 11.4 se muestra la segmentación lograda en esta secuencia.

Se comparan los resultados del algoritmo propuesto usando la difusión MVPD, con el algoritmo usando la difusión VPD y el algoritmo sin la etapa de difusión, que llamaremos (WOP - *without probabilistic relaxation*). Para medir el ajuste de los resultados obtenidos se segmentaron manualmente los cuadros 5, 10, 15, 50, 100 y 200 de la secuencia, y se comparan con las segmentaciones obtenidas por las tres variantes comentadas.

En la figura 11.1 se muestran los resultados del conteo de falsos positivos y falsos negativos para los cuadros segmentados manualmente. En la figura 11.3 se muestran las segmentaciones para los cuadros seleccionados con las tres variantes (WOP, VPD y MVPD). Se puede ver cómo los resulta-



(a)



(b)

Cuadro	5	10	15	50	100	200
Area M	6161	6177	6162	6281	7811	7287
Perim M	314	321	323	321	340	350

(c)

Figura 11.1: **Falsos positivos y falsos negativos.** *Falsos positivos* (a) *y falsos negativos* (b) *calculados contra la segmentación manual de los cuadros. En rojo el algoritmo con* MVPD*, en azul con* VPD *y en verde sin difusión –*WOP*–.* (c) *Áreas y perímetros de la segmentación manual (en píxeles).*

dos con difusión reducen el número de falsos negativos (figura 11.1(b)), esto es, disminuye el número de puntos del objeto que son mal clasificados como fondo. Asimismo se produce un aumento de los falsos positivos (figura 11.1(a)). En la figura 11.2 se muestra la localización de los píxeles clasificados como falsos positivos (en negro) y falsos negativos (en blanco) para uno de los cuadros segmentados manualmente. Primero, se ve que la segmentación manual omite el pelo cercano al cuello, siendo esta la mayor fuente de falsos positivos[1]. Por esto se entiende que el número de falsos positivos aumente, siendo en realidad menor que el que se muestra en la figura 11.1(a). Segundo, la segmentación manual es rugosa en los bordes mientras que las segmentaciones con difusión presentan bordes más regulares. La difusión implica un compromiso entre el número de falsos negativos y falsos positivos; los bordes definidos con difusión son más regulares que sin difusión.

Los falsos negativos se concentran mayormente en el casco, en píxeles donde el color no difiere mucho del fondo próximo, y donde el borde no es

[1]La presencia de pelo en esa región de la imagen se detectó debido a la segmentación automática por el método, antes no era conocida.

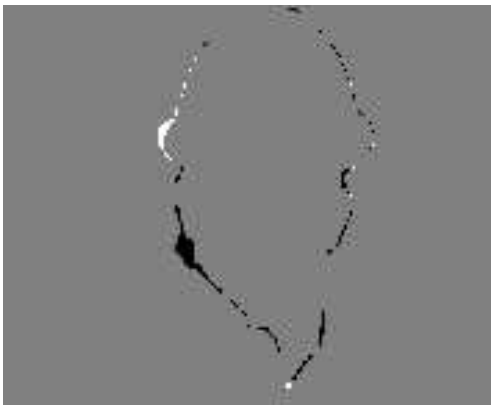


Figura 11.2: **Píxeles marcados como falsos positivos o falsos negativos.** *En negro se muestran los falsos positivos y en blanco los falsos negativos.*

claramente definido. Incluso manualmente la segmentación es difícil, y entran en juego decisiones subjetivas («se sabe» que se está segmentando un casco y «debe tener esa forma»). Sin embargo la segmentación obtenida a través de los casi trescientos cuadros de la secuencia es aceptable y el casco se segmenta correctamente.

De la comparación entre el algoritmo con la difusión MVPD y con VPD se concluye que los resultados son más ajustados que sin difusión –WOP–. MVPD difunde respetando los bordes de la imagen, que, dada la distribución de los colores entre el objeto y el fondo, son muy similares a los bordes en el espacio de probabilidad. De ahí que la modificación presentada no presente grandes mejoras en la segmentación.

En la evolución de la segmentación en los primeros cuadros se observa que la máscara inicial (figura 10.3(a)) no cubre completamente el cuello. Luego de unos pocos cuadros la segmentación se extiende a todo el cuello visible en los cuadros. Dado que no hay restricciones en la forma con que puede evolucionar la segmentación este comportamiento es razonable que suceda así, pues el modelo de color asignará una probabilidad alta, y el modelo de la posición hará lo mismo. La segmentación se detiene en los bordes del cuello con la camisa, lo cual es correcto.

`flower garden`

Los resultados para la secuencia `flower garden` se muestran en la figura 11.5. El objeto que se segmenta es el árbol que aparece en primer plano y

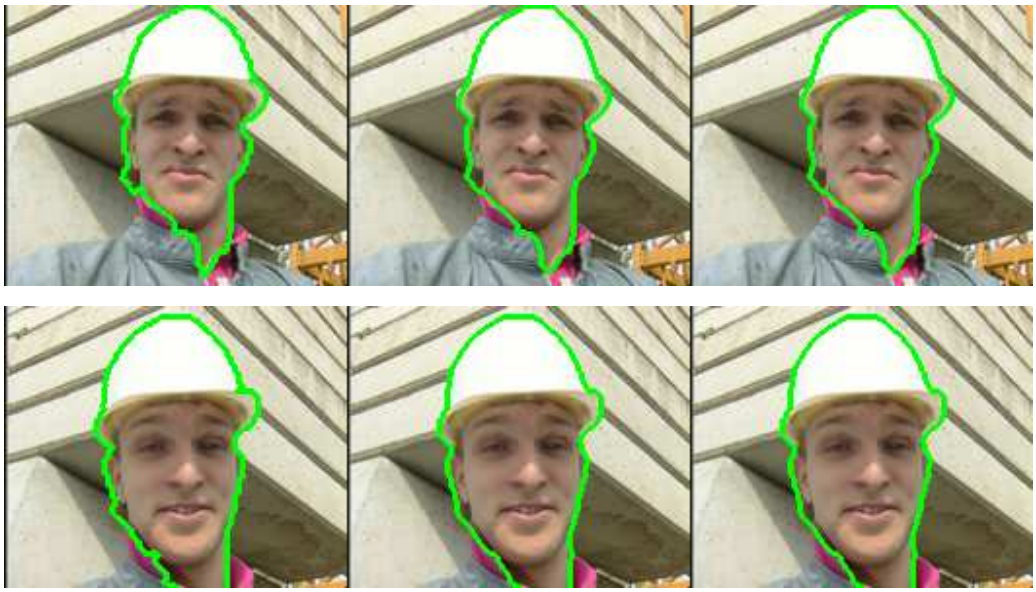


Figura 11.3: **Segmentación de la secuencia** `foreman` *Se muestra la comparación de las segmentaciones obtenidas sin difusión* –WOP–*, con difusión* VDP *y con difusión* MVPD *en los cuadros 50 y 100.*

que se desplaza de derecha a izquierda, y al alejarse disminuye su tamaño. A pesar de ser una secuencia donde el objeto a segmentar está presente en pocos cuadros, presenta condiciones particulares para el análisis del algoritmo.

La segmentación en esta secuencia es difícil para el estado actual del algoritmo. Se observa que regiones del fondo vecinas al objeto, presentan un color semejante (cerco color madera en las construcciones), lo cual hace que el color le asigne una probabilidad alta de pertenecer al objeto. Al estar en el vecindario del objeto, la posición también le asignará una probabilidad mayor a estos píxeles vecinos que a píxeles de color semejante pero más alejados del objeto. La combinación de estos hechos hace que la segmentación del objeto se «adhiera» a regiones del fondo de características semejantes.

Otro fenómeno de esta secuencia, con este objeto, es que al desplazarse se produce una rotación del mismo en la proyección a la cámara y aparecen nuevas regiones del objeto que no eran visibles inicialmente. El mismo efecto de «adherencia» se produce en la parte superior del árbol hacia donde se extiende la segmentación. En alguna de estas nuevas regiones el modelo de color no necesariamente cumple con el modelo utilizado, lo cual implica la

necesidad de la actualización del modelo del mismo, que en este caso no se realiza.

La segmentación generada es razonable con el algoritmo propuesto, pero no es correcta como resultado. La misma considera las nuevas regiones del objeto que aparecen en los cuadros, y concuerdan con el conocimiento *a priori* de qué «es un árbol». Esta información no es considerada en el algoritmo, la «adherencia» al fondo y la extensión a la parte superior del árbol, se producen por el mismo fenómeno en la combinación de las probabilidades, en este caso correcto y en el otro no.

Para contemplar estos fenómenos es necesario agregarle otras «restricciones» al algoritmo, lo cual se discute en las conclusiones (capítulo 12), y una solución alternativa se muestra en la sección 11.2.

`carphone`

La secuencia `carphone` es similar a la secuencia `foreman` en el sentido que es una «cabeza parlante» (no sólo por el «actor» que la representa). El objeto a segmentar, es la cabeza, que se mueve por la escena, varía su posición, gira levemente (lo cual deforma la proyección del objeto), se acerca a la cámara y luego se aleja (variando su tamaño). En la figura 11.6 se muestra la segmentación lograda.

Es remarcable en esta secuencia que el objeto a segmentar sufre una oclusión parcial por un nuevo objeto (la mano), que luego se retira. Se puede observar en el cuadro 178 que el algoritmo considera este nuevo objeto dentro de la segmentación generada, esto es debido al color similar (piel) de la mano; el efecto de «adherencia» antes comentado. Luego, al retirarse la mano, el algoritmo vuelve a segmentar sólo la cabeza. Se muestra el cuadro 187 (9 cuadros después), donde además ocurre el punto de mayor acercamiento del actor a la cámara, aumentando el tamaño del objeto al máximo. Luego al alejarse el objeto se achica y el algoritmo lo sigue de forma aceptable.

En la figura 11.8(a) se muestran los resultados de comparar la segmentación obtenida por el algoritmo con una segmentación manual de los mismos cuadros. En la tabla 11.1 se muestra el conteo de los Falsos Positivos (FP) y Falsos Negativos (FN). También se muestran los perímetros y áreas del objeto calculadas con la segmentación manual (M) y con la segmentación generada por el algoritmo (A). De los valores numéricos que se muestran se observa que la segmentación generada con el algoritmo es muy

Cuadro	2	89	115	178	187	300
FP	59	90	77	764	193	133
FN	47	67	152	50	32	45
Perím. M	222	225	228	326	288	247
Perím. A	235	237	241	281	290	240
Area M	3903	4240	4194	6357	6094	4352
Area A	3891	4217	4269	5643	5933	4264

Tabla 11.1: *Comparación con una segmentación manual de los cuadros 2, 89, 115, 178, 187 y 300 de* `carphone`*. Se muestran los Falsos Positivos (FP), Falsos Negativos (FN), perímetros y áreas con la segmentación manual (M) y la del algoritmo (A).*

próxima a la segmentación manual, excepto en el cuadro 178, donde se produce el fenómeno de «adherencia» a la mano y el algoritmo genera una salida incorrecta.

`container`

En la figura 11.7 se muestran los resultados de la segmentación generada para la secuencia `container`. El objeto que se segmenta, el barco, tiene un desplazamiento muy lento hacia la derecha. Una de las características particulares de este ejemplo es que el objeto tienen bordes rectilíneos en la parte superior por los contenedores que transporta el barco, con ángulos rectos en varios puntos del mismo. Además hay partes del barco de unos pocos píxeles de ancho, como antenas en la parte superior, que no logran ser segmentados por el algoritmo.

La segmentación es aceptable y relativamente estable. Los errores en la segmentación se dan fundamentalmente en los ángulos rectos, y en que el desplazamiento del barco hace que los contenedores «se acerquen». Delante del objeto, hay un mástil con una bandera, la cual no se desplaza junto con el objeto. Dada la segmentación inicial del objeto la bandera forma parte del objeto a segmentar. Esto ocasiona una deformación de la segmentación del objeto (ver cuadros 49 y 107). Cuando la bandera está completamente delante del barco la segmentación es correcta; sobre el final de la secuencia hay una deformación de la segmentación debido a que la estimación de la posición por la convolución con el núcleo gaussiano, une la región entre los contenedores y no se logra distinguir el espacio entre ellos.

En la figura 11.8(b) se muestran los resultados de comparar la segmentación obtenida por el algoritmo con una segmentación manual. En la

Cuadro	2	14	49	107	185	300
FP	207	170	176	124	175	246
FN	142	192	174	210	159	143
Perím. M	368	354	342	333	313	270
Perím. A	405	396	398	387	356	342
Area M	4051	3932	3859	3720	3491	3344
Area A	3986	3954	3857	3806	3475	3241

Tabla 11.2: *Comparación con una segmentación manual de los cuadros 2, 14, 49, 107, 185 y 300 de* `container`*. Se muestran los Falsos Positivos (FP), Falsos Negativos (FN), perímetros y áreas con la segmentación manual (M) y la del algoritmo (A).*

tabla 11.2 se muestra el conteo de los Falsos Positivos (FP) y Falsos Negativos (FN). También se muestran los perímetros y áreas del objeto calculadas con la segmentación manual (M) y con la segmentación generada por el algoritmo (A). En este caso el conteo de falsos positivos y falsos negativos es mucho mayor que en el caso de `carphone` debido a la forma del objeto y los detalles que se segmentan manualmente pero que el algoritmo no contempla. Vale aclarar que estos detalles son, en general de unos pocos píxeles, como ser antenas y otros «artefactos» cercanos a la parte superior del barco.

`mobile & calendar`

En la figura 11.9 se muestran los resultados en la secuencia `mobile & calendar` donde se notan otras limitaciones del algoritmo. El modelo de color del objeto le asigna una probabilidad baja al rojo de la chimenea, mientras que el modelo de color del fondo le asigna una probabilidad alta. En la combinación de las probabilidades a posteriori no se logra mejorar esta situación, obteniendo una segmentación mala, luego de pocos cuadros.

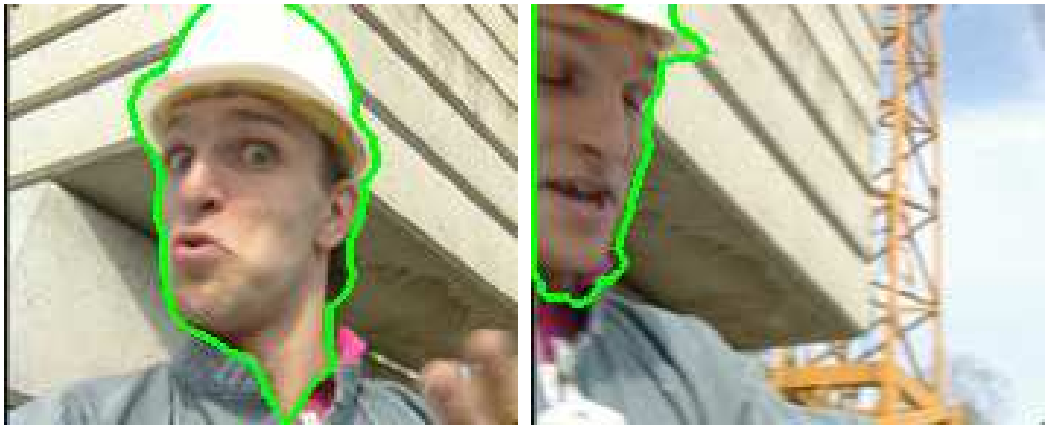
Al combinar las probabilidades a posteriori dando mayor confianza al modelo de la posición, se mejora la segmentación. En la figura 11.10 se muestran las probabilidades a posteriori generadas por el algoritmo en dos configuraciones diferentes. En la figura 11.10(a) se configuró el algoritmo con igual confianza en ambas características ($\alpha_p = \alpha_c$); mientras que en la figura 11.10(b) se le asigna mayor confianza a la posición frente al color ($\alpha_p = 3\,\alpha_c$); en la misma figura se muestran las probabilidades luego de la difusión, con las líneas de nivel.

La segmentación que se logra en esta segunda configuración mejora los errores que se cometían en la primera configuración. La asignación de los valores de confianza en esta etapa de desarrollo del algoritmo son realizadas

por el usuario.

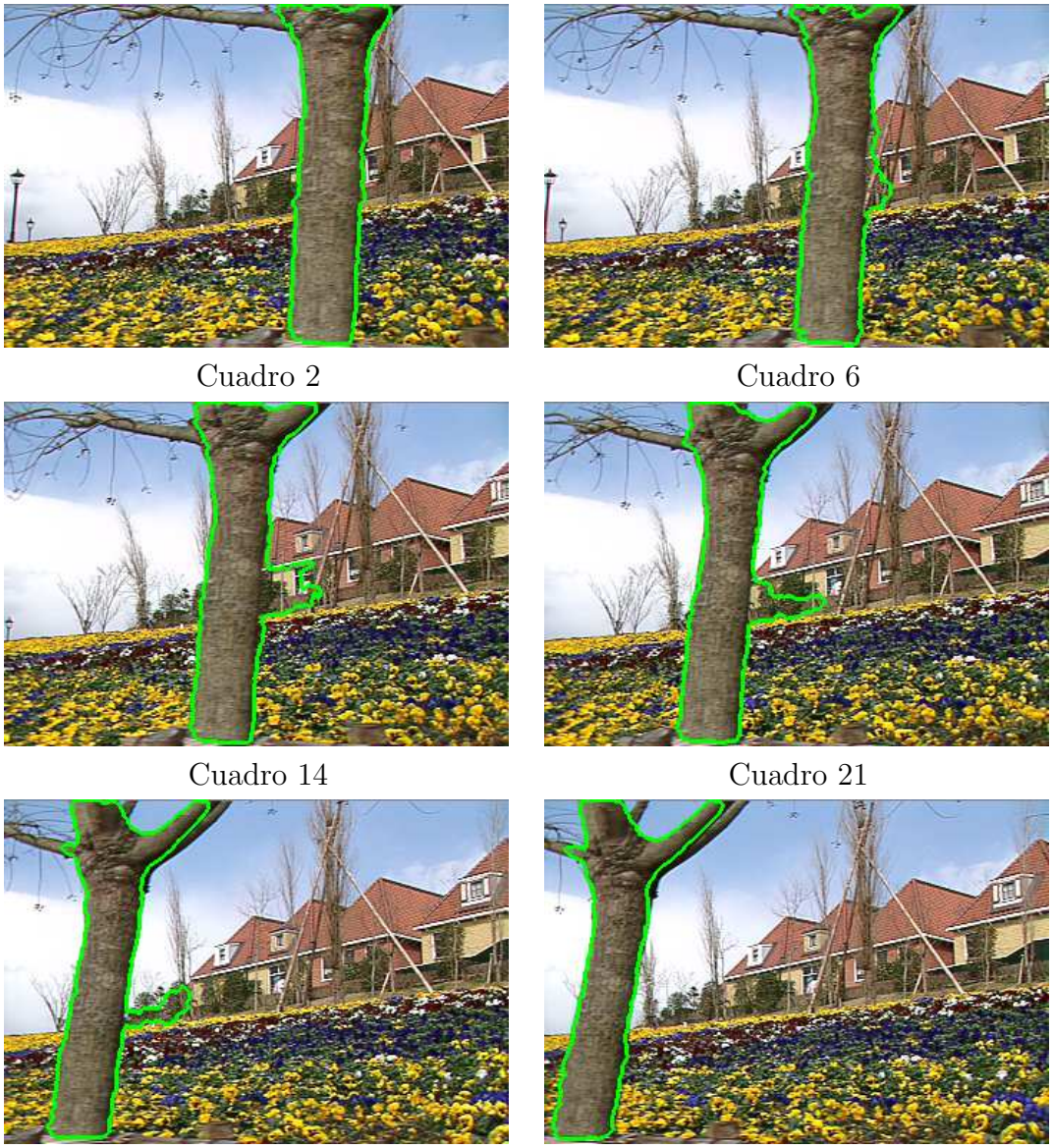
Cuadro 2 Cuadro 66

Cuadro 111 Cuadro 122



Cuadro 191 Cuadro 286

Figura 11.4: *Segmentación de la secuencia* `foreman`*, se muestran los cuadros 2, 66, 111, 122, 191 y 286.*

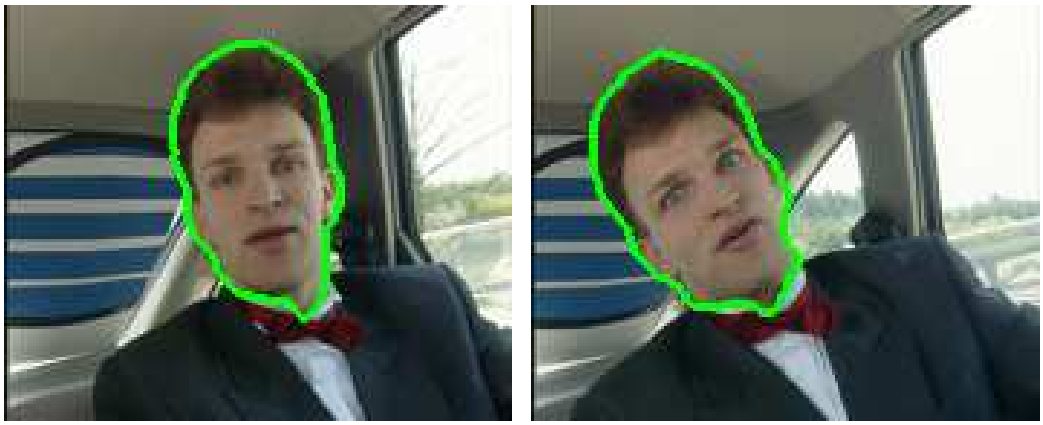


Cuadro 2 Cuadro 6

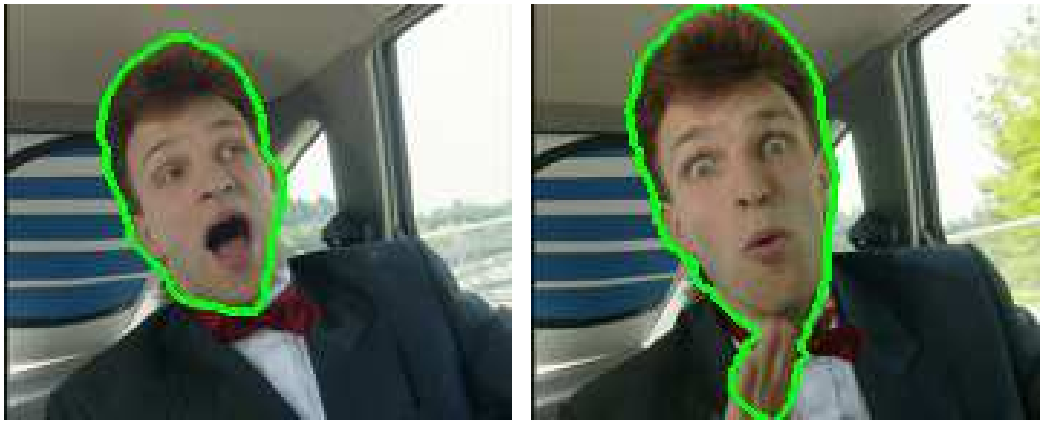
Cuadro 14 Cuadro 21

Cuadro 33 Cuadro 38

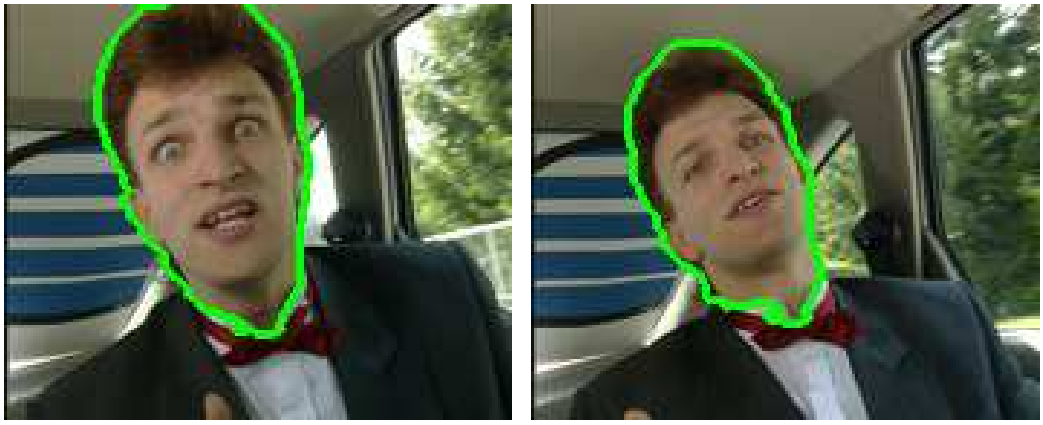
Figura 11.5: *Segmentación de la secuencia* **`flower garden`**, *se muestran los cuadros 2, 6, 14, 21, 33 y 38.*



Cuadro 2 Cuadro 89

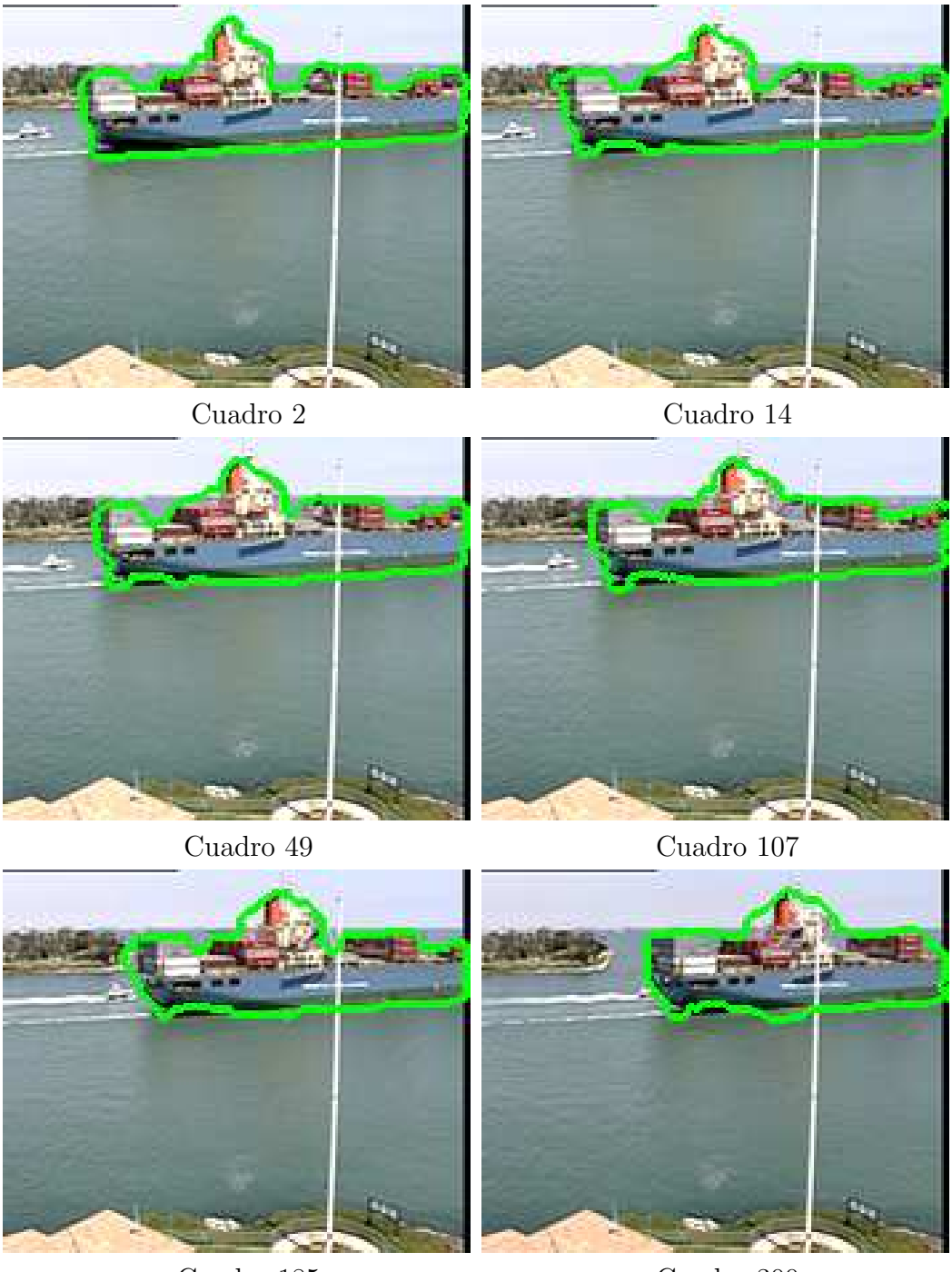


Cuadro 115 Cuadro 178



Cuadro 187 Cuadro 300

Figura 11.6: *Segmentación de la secuencia* `carphone`*, se muestran los cuadros 2, 89, 115, 178, 187 y 300.*

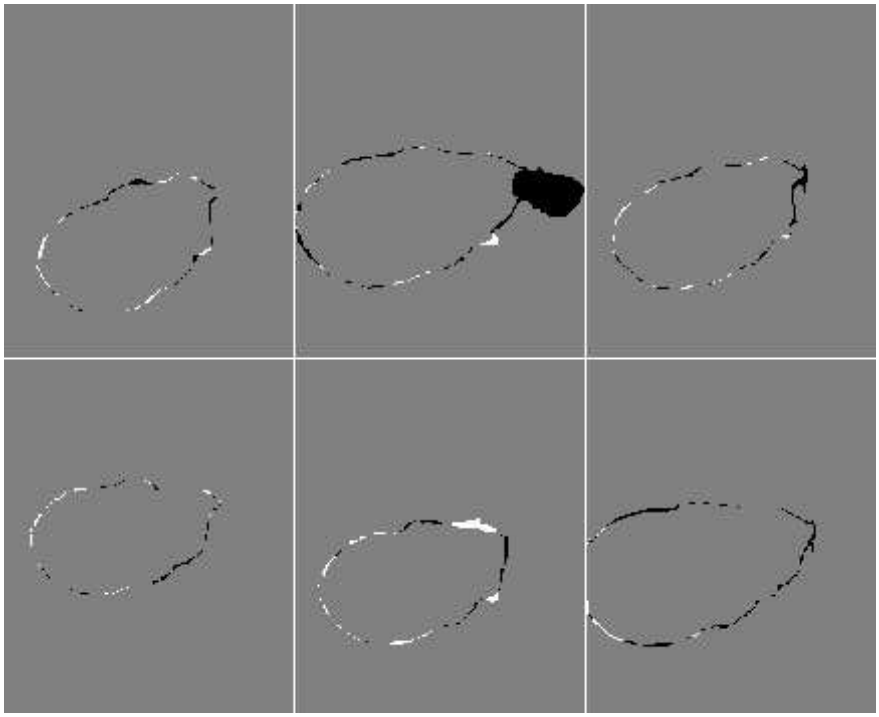


Cuadro 2 Cuadro 14

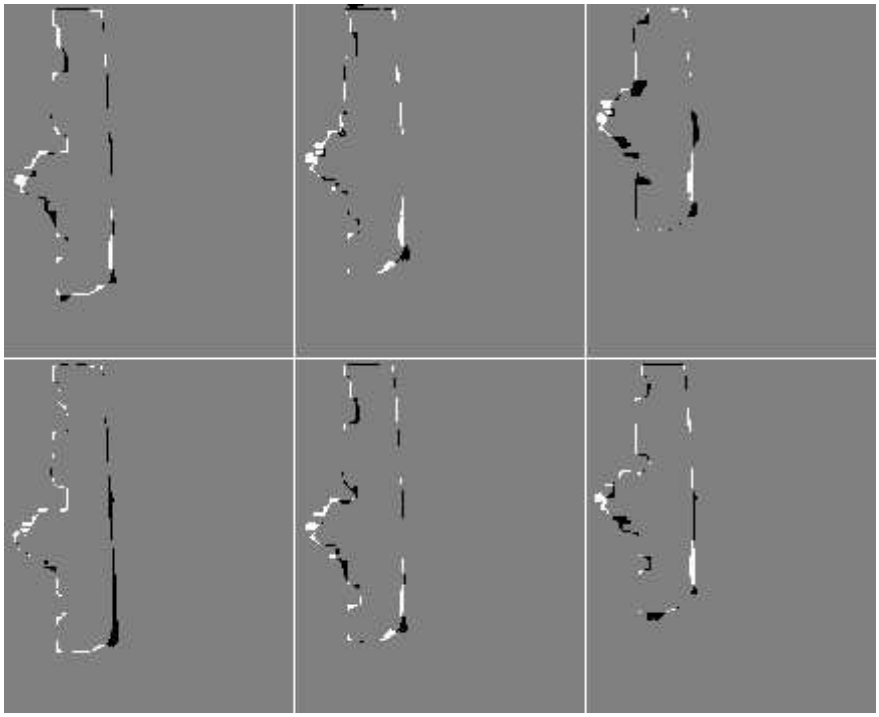
Cuadro 49 Cuadro 107

Cuadro 185 Cuadro 300

Figura 11.7: *Segmentación de la secuencia* `container`*, se muestran los cuadros 2, 14, 49, 107, 185 y 300.*

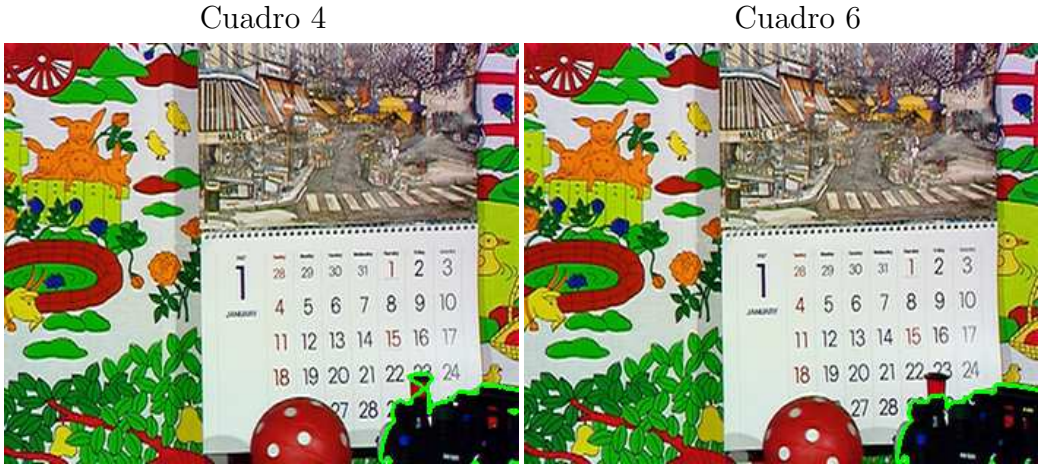


(a) *Rotar 90º en sentido horario para ver.*

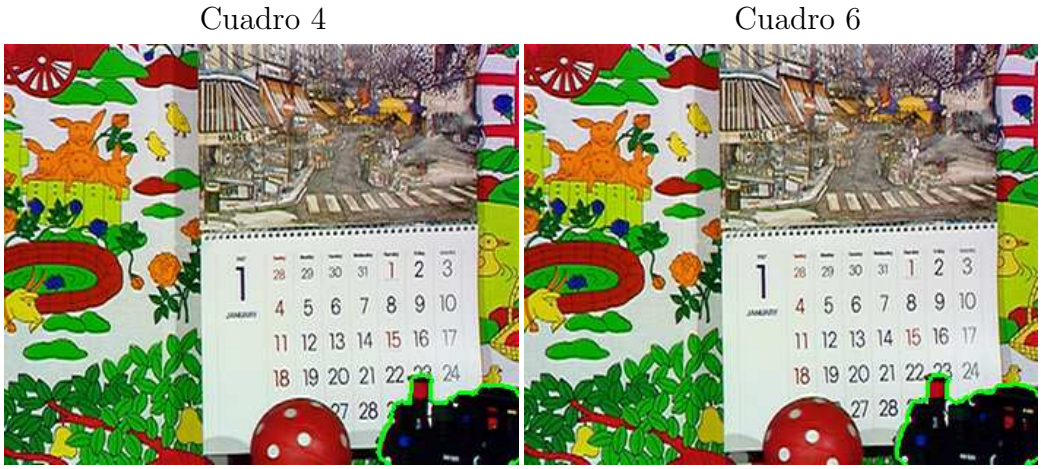


(b) *Rotar 90º en sentido horario para ver.*

Figura 11.8: *Falsos positivos (en negro) y Falsos negativos (en blanco) comparando con una segmentación manual de los cuadros mostrados en las figuras 11.6 y 11.7.* (a) *Carphone* (b) *Container*

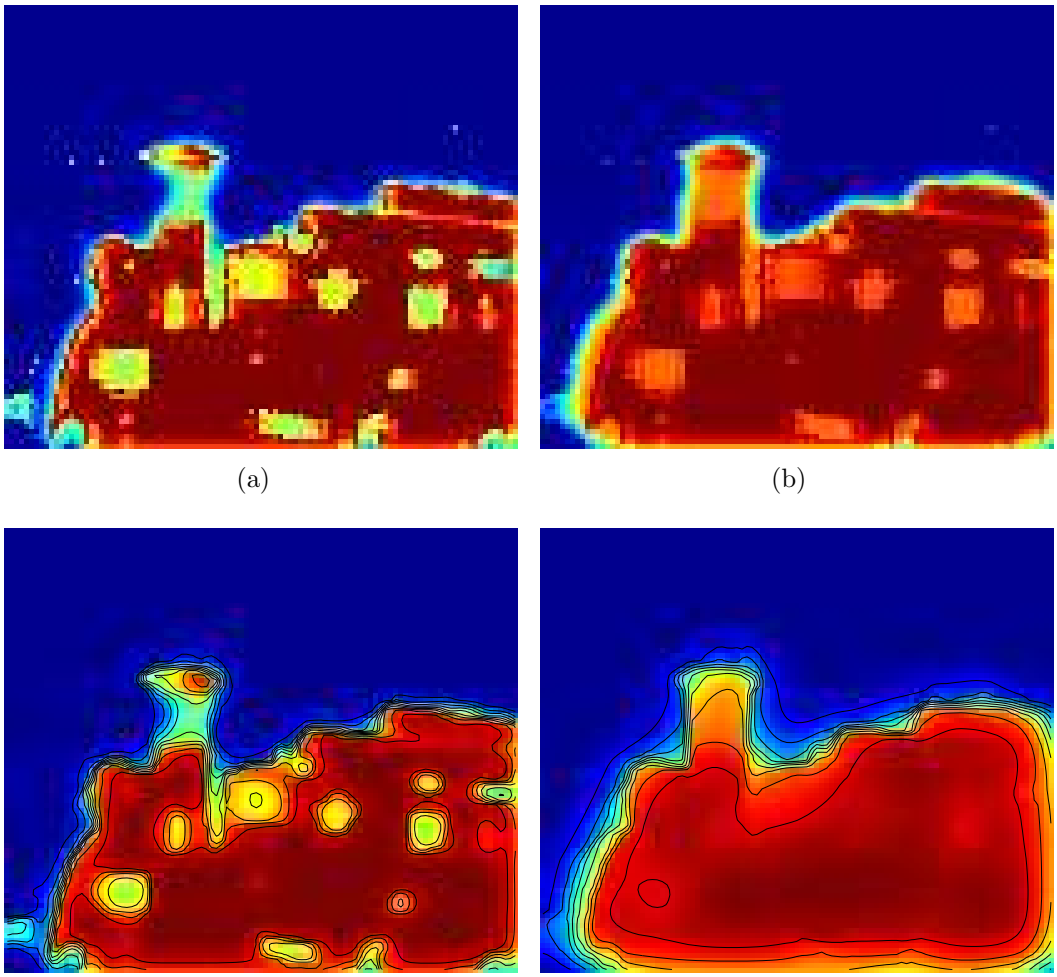


(a)



(b)

Figura 11.9: **Limitaciones del algoritmo en la segmentación de la secuencia** `mobile & calendar`**.** *La semejanza de los modelos de color en la componente del rojo de la chimenea genera una mala segmentación.* (a) *Se muestran los cuadros 4 y 6 de la combinación con igual peso* ($\alpha_p = \alpha_c$) *de las probabilidades a posteriori dado el color y dada la posición.* (b) *Los mismos cuadros dando mayor peso a la probabilidad a posteriori dada la posición* ($\alpha_p = 3\alpha_c$).



(a) (b)

(c) (d)

Figura 11.10: **Probabilidad del objeto en `mobile & calendar`.** (a) *Probabilidad con la combinación con iguales confianzas* ($\alpha_p = \alpha_c$) (b) *Probabilidad de la combinación con mayor confianza en la posición* ($\alpha_p = 3\alpha_c$) (c)-(d) *Probabilidades luego de la difusión.*

11.2. La disparidad como característica para la segmentación

En esta segunda parte se han utilizado las expresiones «objeto» para denominar la región de interés en la imagen y «fondo» como denominación del resto de la imagen. Esto viene del uso de las expresiones del inglés *object* y *background*. Sin embargo, no hay hipótesis en el algoritmo presentado que implique un orden dentro de la escena, y que el objeto esté delante del fondo. En la secuencia `foreman` se ha segmentado la cabeza y el casco, dejando el torso «en el fondo». Sí existen dificultades cuando los regiones del fondo ocluyen el objeto, como ser el caso de la mano en el ejemplo de `carphone`, o las regiones vecinas de color semejante al objeto en la secuencia `flower garden`.

Cuando se puede recuperar la estructura de la escena es posible utilizar ésta como información para la segmentación. En el caso de poder obtener una secuencia estéreo es posible obtener la disparidad de los objetos en la escena, y utilizarlo como característica [36, 37, 38].

En la secuencia `flower garden` hay un desplazamiento horizontal de la cámara, de izquierda a derecha. Esto, junto que la escena es estática[2], genera que cuadros consecutivos de la secuencia tengan un desplazamiento horizontal entre ellos. La correspondencia entre *scanlines* no está garantizada, pero es muy cercana[3]. Asumiendo esta hipótesis, se tomó el cuadro t como imagen izquierda y el cuadro $t+1$ como imagen derecha de un par estéreo y se calculó la disparidad de cada uno de los cuadros de la secuencia, con los algoritmos presentados en la parte I. A la disparidad obtenida con el algoritmo de Bobick e Intille, se le aplicó un filtro de mediana en las columnas para agregar coherencia *inter-scanline*, como se explicó en la sección 4.1.4.

Para modelar la disparidad se utilizan histogramas con los valores de la disparidad en el objeto y en el fondo, generando una probabilidad para el píxel X de pertenecer al objeto, O, dado el valor de disparidad calculado en el píxel, f_d,

$$P(X \in O | f_d)$$

y de forma similar la probabilidad de pertenecer al fondo, B, dado el valor de disparidad. Este modelo es actualizado en cada uno de los cuadros a partir de la segmentación gruesa $\tilde{S}(t+1)$.

[2]Los objetos de la escena no se deforman ni cambian su posición en la escena.

[3]Los resultados de los algoritmos de cálculo de disparidad entre cuadros consecutivos verifican que se puede hacer esta consideración.

En el primer experimento que se presenta se utilizaron como características la posición y la disparidad, sin hacer uso de la información de color. En la figura 11.11 se muestran las disparidades calculadas con el algoritmo de Kolmogorov y Zabih, y la segmentación obtenida. En la figura 11.12 se muestran las disparidades calculadas con el algoritmo de Bobick e Intille, y la segmentación obtenida. La única diferencia en la configuración de los algoritmos es el mapa de disparidad utilizado.

Los resultados obtenidos con esta variante son razonables, dadas las disparidades estimadas. Se observa que ambos mapas de disparidad definen bien los bordes del objeto, excepto en la parte superior donde se producen errores. En el caso de Kolmogorov y Zabih los resultados son mejores en la parte superior del árbol. En esa región el algoritmo de Bobick e Intille presenta errores clásicos en este algoritmo. Las ramas finas (1 o 2 píxeles de ancho) a la izquierda del tronco, generan un matcheo correcto, y una disparidad mayor que los píxeles vecinos, que corresponden a un cielo uniforme, donde no se logra calcular la disparidad correcta y *se mantiene* la disparidad del matcheo anterior en la *scanline*. Esto produce las regiones de disparidades erróneas delante del objeto en la disparidad calculada con el algoritmo de Bobick e Intille.

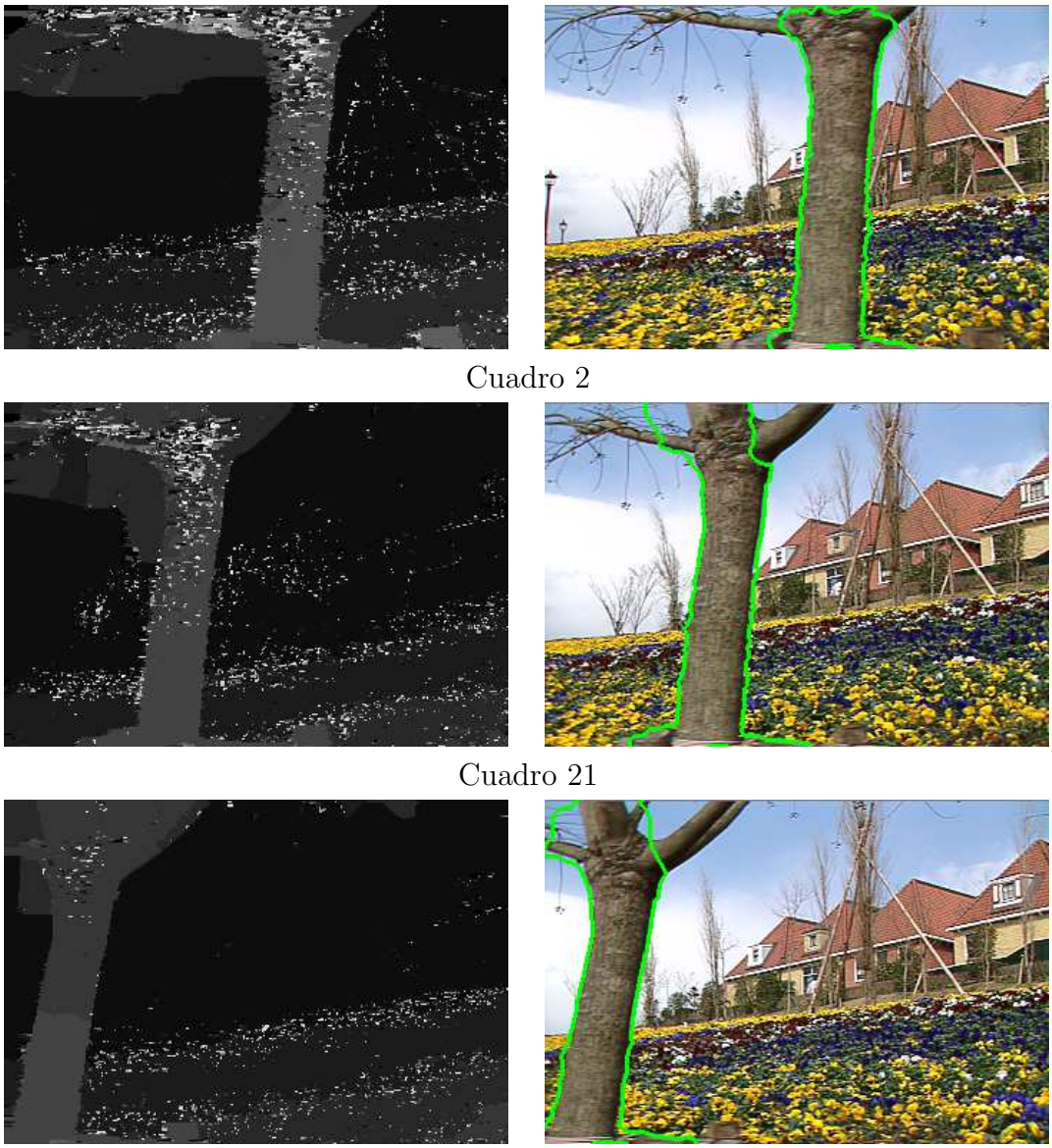
Los errores que se generan (fenómeno de «adherencia») utilizando posición y color (figura 11.5), en este caso no se presentan, pues la disparidad diferencia perfectamente las regiones donde el color no discrimina correctamente. Sin embargo, se introducen errores en otras regiones donde el color sí tiene un buen desempeño (en las ramas superiores).

En el último experimento que se presenta, se combinan las características de posición, color y disparidad para segmentar esta secuencia. Los resultados se muestran en la figura 11.13.

Se observa que la disparidad y el color se complementan, corrigiendo errores que se producen cuando se utilizan las características por separado. La segmentación del tronco es correcta, evitando que la misma se «adhiera» al fondo, debido al aporte de la disparidad. Mientras tanto, la segmentación de las ramas en la parte superior del árbol no sufre las deformaciones que se tienen en la segmentación considerando la disparidad y la posición. Aunque no se extiende tanto hacia las ramas superiores como en el caso de color y posición, debido al aporte de la disparidad que no contempla estas regiones.

Este último experimento muestra que la combinación con nuevas características mejora la segmentación y genera una solución a un problema complejo que la configuración inicial del algoritmo no puede resolver. Sin

embargo se sigue utilizando información en cada píxel para la clasificación y no se utiliza ninguna curva para la definición del borde, sobre la cual se pueda incorporar otras restricciones en su deformación.

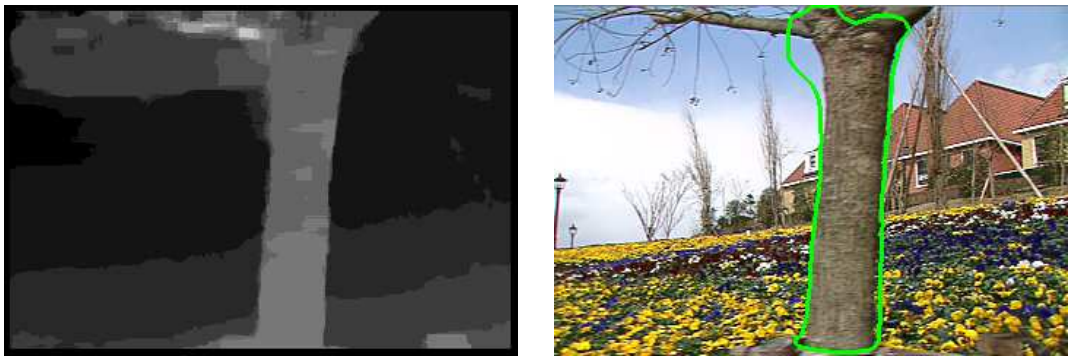


Cuadro 2

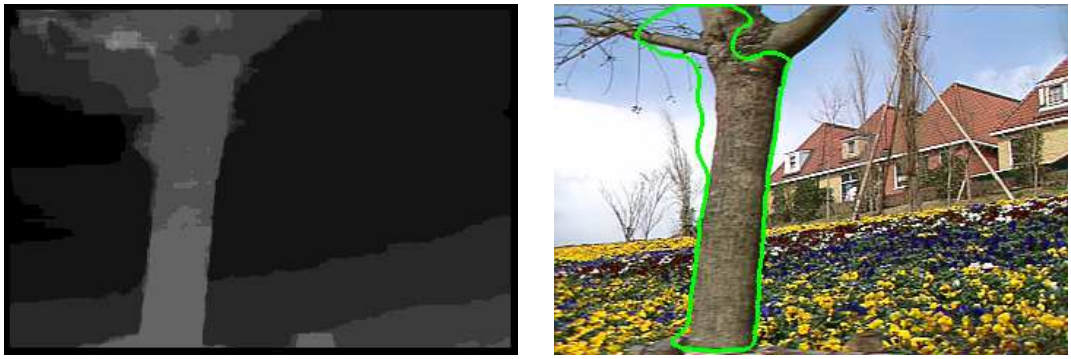
Cuadro 21

Cuadro 38

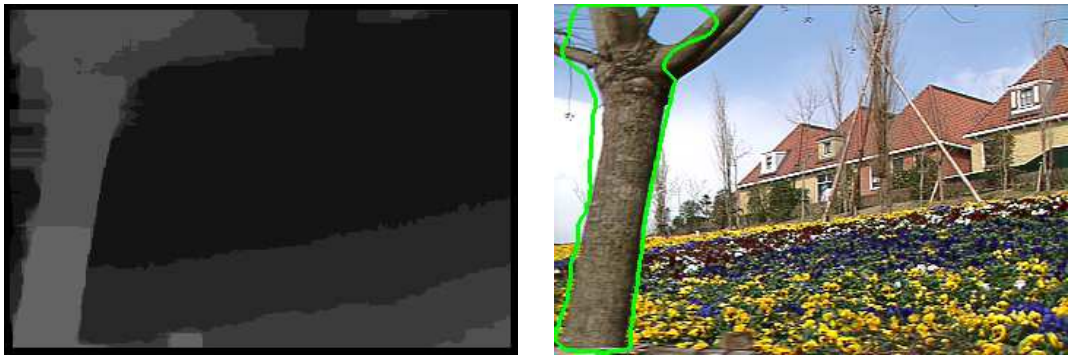
Figura 11.11: **Segmentación de la secuencia `flower garden`, utilizando la posición y la disparidad.** *A la izquierda la disparidad calculada con el algoritmo de Kolmogorov y Zabih, a la derecha la segmentación lograda. Se muestran los cuadros 6, 21 y 38.*



Cuadro 6

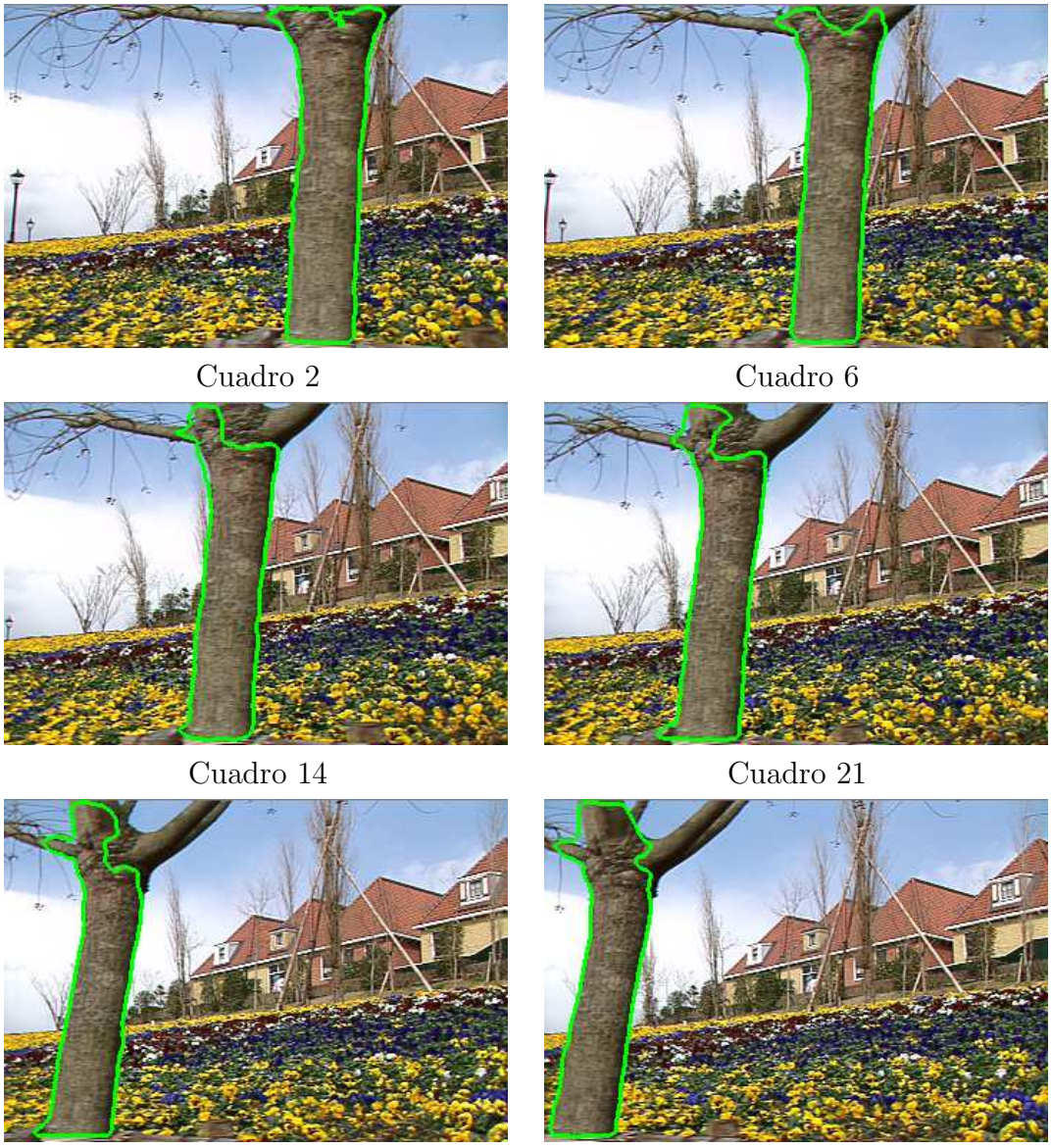


Cuadro 21



Cuadro 38

Figura 11.12: **Segmentación de la secuencia `flower garden`, utilizando la posición y la disparidad.** *A la izquierda la disparidad calculada con el algoritmo de Bobick e Intille, a la derecha la segmentación lograda. Se muestran los cuadros 6, 21 y 38.*



Cuadro 2 Cuadro 6

Cuadro 14 Cuadro 21

Cuadro 33 Cuadro 38

Figura 11.13: **Segmentación de la secuencia `flower garden`, utilizando la posición, el color y la disparidad.** *La disparidad es calculada con el algoritmo de Bobick e Intille. Se muestran los cuadros 2, 6, 14, 21, 33 y 38.*

12 Conclusiones y trabajo futuro

Se presentó un algoritmo para la segmentación semiautomática de objetos basado en el uso de múltiples características con el agregado de un método de difusión de probabilidades (modificación del método existente de VPD). Las hipótesis que plantea el sistema son fuertes para los objetos a segmentar en las secuencias. Sin embargo las pruebas que se realizaron con secuencias que los cumplen han dado resultados aceptables con buena precisión en la localización de los contornos del objeto. Esto muestra la validez de un esquema de clasificación simple basado en características espaciales y coherencia temporal dada por el movimiento, junto con una etapa de difusión.

La implementación del algoritmo desarrollado en MatLab© realiza la segmentación de secuencias (tamaño QCIF) a una razón de un cuadro por segundo. Por lo que se puede esperar que la implementación final en C++ tenga una performance cercana al tiempo real, sin hacer actualización del modelo de color en cada cuadro.

El esquema de segmentación (clasificación) puede ser extendido para contemplar la segmentación de más de un objeto sobre el fondo. En este caso, hay que considerar alguna las restricciones para manejar las posibles oclusiones entre los objetos.

De igual forma, la incorporación de nuevas características a las usadas en el algoritmo actual es sencilla; lo cual permitiría obtener mejores resultados con características más ajustadas a las diferencias entre los objetos. Se muestra un caso particular con uso de la disparidad en la segmentación de la secuencia `flower garden` donde los resultados de la combinación mejoran sensiblemente los resultados obtenidos. El uso de la disparidad como característica no siempre es posible, pues es necesario que se cumplan ciertas hipótesis en la secuencia o utilizar una secuencia estéreo.

Los resultados son aceptables en la mayoría de las secuencias probadas, y en aquellas donde se producen errores los resultados presentados serán mejorables con la inclusión de métodos más sofisticados. En este sentido, el método propuesto está basado en regiones y no impone ningún tipo de restricción a los bordes en la segmentación del objeto, a pesar de esto las segmentaciones que se generan son ajustadas a los mismos. La precisión de

los bordes depende de la capacidad de discriminación de las características. La aparición de regiones del fondo con características similares al objeto no están contempladas en el algoritmo y es una fuente de error.

Uno de los agregados para mejorar la segmentación del algoritmo es el uso de *snakes* [122] para la definición del borde como una curva. Esto permite imponer restricciones sobre la deformación que se le permite tener al objeto, además de dar al usuario un forma de controlar «el borde» para las correcciones.

La segmentación de objetos no rígidos, o cuya deformación no pueda ser contemplada por el modelo de propagación de la segmentación entre cuadros no es posible. Para contemplar este hecho una solución posible puede ser la aplicación de la misma propagación con un esquema basado en regiones, por ejemplo dividiendo el objeto en regiones que puedan ser propagadas con este esquema, y luego agruparlas.

La adaptación del modelo de color es necesaria para obtener una segmentación robusta a través de un gran número de cuadros si el objeto presenta muchas variaciones, en su color o iluminación. Para atacar este último punto, una opción es utilizar como característica en la clasificación únicamente las componentes de croma del color.

Bibliografía

[1] Leonardo da Vinci. *Tratado de la Pintura. Versión castellana de Mario Pitaluga.* Buenos Aires: Losada, 1943.

[2] F.R.S. Charles Wheatstone. Contributions to the Physiology of Vision. Part the First. On some remarkable, and hitherto unobserved, Phenomena of Binocular Vision. *Philosophical Transactions of the Royal Society of London*, 128:371 – 394, 1838.

[3] D. Marr y T. Poggio. A computational theory of human stereo vision. *Proc R Soc Lond*, pp. 301–328, May 1979.

[4] Daniel Scharstein y Richard Szeliski. Stereo Vision Research Page. WWW, 6 de diciembre de 2004. `http://cat.middlebury.edu/stereo/`.

[5] Visual Perception Library. WWW, 12 de noviembre de 2004. `http://viperlib.york.ac.uk/`.

[6] The Lord of the Rings – The Fellowship of the Ring. WWW, 16 de enero de 2001. `http://www.lordoftherings.net/`.

[7] Daniel Scharstein y Richard Szeliski. High-Accuracy Stereo Depth Maps Using Structured Light. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1:195–202, January 2003.

[8] Richard Hartley y Andrew Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge, UK, Cambridge University Press, 2000.

[9] Olivier Faugeras, Quang-Tuan Luong y Theo Papadopoulo (cont.). *The Geometry of Multiple Images: The Laws That Govern the Formation of Multiple Images of a Scene and Some of Their Applications.* MIT Press, 2001.

[10] Myron Z. Brown, Darius Burschka y Gregory D. Hager. Advances in computational stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):993–1008, August 2003.

[11] Aaron F. Bobick y Stephen S. Intille. Large Occlusion Stereo. *International Journal of Computer Vision*, 33(3):181–200, 1999.

[12] Peter N. Belhumeur. A Bayesian Approach to Binocular Stereopsis. *International Journal of Computer Vision*, 19(3):237–260, August 1996.

[13] Vladimir Kolmogorov y Ramin Zabih. Multi-camera Scene Reconstruction via Graph Cuts. *European Conference on Computer Vision (3)*, pp. 82–96, 2002.

[14] Sébastien Roy y Ingemar J. Cox. A Maximum-Flow Formulation of the N-Camera Stereo Correspondence Problem. *International Conference on Computer Vision*, pp. 492–502, 1998.

[15] Daniel Scharstein y Richard Szeliski. A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. *International Journal of Computer Vision*, pp. 7–42, April-June 2002.

[16] Yuri Boykov y Vladimir Kolmogorov. An Experimental Comparison of Min-cut/Max-flow Algorithms for Energy Minimization in Vision. *Energy Minimization Methods in Computer Vision and Pattern Recognition, Third International Workshop*, pp. 359–374, 2001.

[17] Yuri Boykov y Vladimir Kolmogorov. An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004.

[18] Yuri Boykov, Olga Veksler y Ramin Zabih. Fast Approximate Energy Minimization via Graph Cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

[19] Vladimir Kolmogorov y Ramin Zabih. Computing Visual Correspondence with Occlusions via Graph Cuts. *International Conference on Computer Vision*, pp. 508–515, 2001.

[20] Olivier Faugeras y Renaud Keriven. Variational principles, surface evolution, PDEs, level set methods, and the stereo problem. *IEEE Transactions on Image Processing*, 7(3):336–344, March 1998.

[21] Jian Sun, Nan-Ning Zheng y Heung-Yeung Shum. Stereo Matching Using Belief Propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(7):787–800, July 2003.

[22] John (Juyang) Weng. Image matching using the windowed Fourier phase. *International Journal of Computer Vision*, 11(3):211–236, 1993.

[23] Yuichi Ohta y Takeo Kanade. Stereo by Intra- and Inter-Scanline Search Using Dynamic Programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(2):139–154, March 1985.

[24] Pascal Fua. Combining Stereo and Monocular Information to Compute Dense Depth Maps that Preserve Depth Discontinuities. *International Joint Conferences on Artificial Intelligence*, pp. 1292–1298, 1991.

[25] Davi Geiger, Bruce Ladendorf y Alan Yuille. Occlusions and Binocular Stereo. *European Conference on Computer Vision*, pp. 425–433, 1992.

[26] Chun-Jen Tsai y Aggelos K. Katsaggelos. Dense Disparity Estimation with a Divide-and-Conquer Disparity Space Image Technique. *IEEE Transactions on Multimedia*, 1(1):18–29, 1999.

[27] Ingemar J. Cox, Sunita L. Hingorani, Satish B. Rao y Bruce M. Maggs. A Maximum Likelihood Stereo Algorithm. *Computer Vision and Image Understanding*, 63(3):542–567, May 1996.

[28] Stan Birchfield y Carlo Tomasi. A Pixel Dissimilarity Measure That Is Insensitive to Image Sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(4):401–406, 1998.

[29] Stan Birchfield y Carlo Tomasi. Depth Discontinuities by Pixel-to-Pixel Stereo. *International Conference on Computer Vision*, pp. 1073–1080, 1998.

[30] Stephen S. Intille y Aaron F. Bobick. Disparity-Space Image and Large Occlusion Stereo. *European Conference on Computer Vision*, volumen 2, pp. 179–186, May 1994.

[31] Chris Buehler, Steven J. Gortler, Michael F. Cohen y Leonard McMillan. Minimal Surfaces for Stereo. *European Conference on Computer Vision*, volumen 3, pp. 885–899, 2002.

[32] Vladimir Kolmogorov y Ramin Zabih. What Energy Functions Can Be Minimized via Graph Cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):147–159, 2004.

[33] Christos H. Papadimitriou y Kenneth Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, 1982.

[34] D. Demirdjian y T. Darrell. Using Multiple-Hypothesis Disparity Maps and Image Velocity for 3-D Motion Estimation. *International Journal of Computer Vision*, 47(1-3):219–228, April-June 2002.

[35] D. Demirdjian y T. Darrell. Motion Estimation from Disparity Images. *International Conference on Computer Vision*, 1:213–218, July 2001.

[36] Kiran Challapali, Tomas Brodsky, Yun-Ting Lin, Yong Yan y Richard Yi Chen. Real-time object segmentation and coding for selective-quality video communications. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(6):813–824, Junio 2004.

[37] Christopher K. Eveland, Kurt Konolige y Robert C. Bolles. Background Modeling for Segmentation of Video-Rate Stereo Sequences. *Computer Vision and Pattern Recognition*, pp. 266–273, 1998.

[38] Ebroul Izquierdo. Disparity/Segmentation Analysis: Matching with Adaptive Windows and Depth Driven Segmentation. *IEEE Transactions on Circuits and Systems for Video Technology, Special Issue on Image and Video Processing for Emerging Interactive Multimedia Services*, 7(4):589–607, June 1999.

[39] L. Di Stefano, M. Marchionni, S. Mattoccia y G. Neri. A fast area-based stereo matching algorithm. *Image and Vision Computing*, 22(12):983–1005, 2004.

[40] Takeo Kanade, Kazuo Oda, Atsushi Yoshida, Masaya Tanaka y Hiroshi Kano. Video-Rate Z Keying: A New Method for Merging Images. Technical Report CMU-RI-TR-95-38, The Robotic Institute, Carnegie Mellon Institute, December 1995.

[41] Kurt Konolige. Small Vision System: Hardware and Implementation. *International Symposium on Robotics Research, Proceedings of*, pp. 111–116, October 1997. `http://www.ai.sri.com/~konolige/svs/`.

[42] TYZX Incorporated. Real-time Stereo Vision for Real-world Object Tracking. `http://www.tyzx.com/`, 15 de diciembre de 2004.

[43] Larry Matthies, Richard Szeliski y Takeo Kanade. Kalman Filter-based Algorithms for Estimating Depth from Image Sequences. Technical Report CMU-RI-TR-88-01, Pittsburgh, PA, Robotics Institute, Carnegie Mellon University, June 1988.

[44] Karsten Mühlmann, Dennis Maier, Jürgen Hesser y Reinhard Männer. Calculating Dense Disparity Maps from Color Stereo Images, an Efficient Implementation. *International Journal of Computer Vision*, 47(1-3):79–88, 2002.

[45] Richard Szeliski y Daniel Scharstein. Symmetric Sub-Pixel Stereo Matching. *European Conference on Computer Vision*, volumen 2, pp. 525–540, May 2002.

[46] Q. Tian y M.N. Huhns. Algorithms for subpixel registration. *Computer Vision Graphics and Image Processing*, 35:220–233, August 1986.

[47] Christoph Strecha, Rik Fransen y Luc Van Gool. Wide-Baseline Stereo from Multiple Views: A Probabilistic Account. *Computer Vision and Pattern Recognition (1)*, pp. 552–559, 2004.

[48] Tinne Tuytelaars y Luc Van Gool. Wide Baseline Stereo Matching based on Local, Affinely Invariant Regions. *British Machine Vision Conference*, 2000.

[49] Takeo Kanade, Atsushi Yoshida, Kazuo Oda, Hiroshi Kano y Masaya Tanaka. A Stereo Machine for Video-Rate Dense Depth Mapping and Its New Applications. *Computer Vision and Pattern Recognition*, pp. 196–202, 1996.

[50] Vladimir Kolmogorov. Vladimir Kolmogorov's Home Page. WWW, 17 de marzo de 2004. `http://www.cs.cornell.edu/People/vnk/software.html`.

[51] Roberto Castagno, Touradj Ebrahimi y Murat Kunt. Video Segmentation based on Multiple Features for Interactive Multimedia Applications. *IEEE Transactions on Image Processing*, 8(5):562–571, September 1998.

[52] Ahmed Elgammal, Ramani Duraiswami, David Harwood y Larry S.Davis. Background and Foreground Modeling Using Nonparametric Kernel Density Estimation for Visual Surveillance. *Proceedings of the IEEE*, 90(7):1151–1163, July 2002.

[53] Touradj Ebrahimi y Caspar Horne. MPEG-4 natural video coding: An overview. *Signal Processing: Image Communication*, 1(15):365–385, 2000.

[54] Paulo L. Correia y Fernando M. Pereira. Classification of Video Segmentation Application Scenarios. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(5):735–741, 2004.

[55] Chad Fogg, Didier J. LeGall, Joan L. Mitchell y William B. Pennebaker. *MPEG Video Compression Standard (Digital Multimedia Standards Series)*. New York, John Wiley & Sons, ©1996.

[56] A. Murat Tekalp. *Digital Video Processing*. New Jersey, Prentice Hall PTR, ©1995.

[57] MPEG ORG. MPEG Pointers & Resources. `http://www.mpeg.org/`, 7 de julio de 2005.

[58] B. S. Manjunath, P. Salembier y T. Sikora (editors). *Introduction to MPEG 7: Multimedia Content Description Language*. John Wiley & Sons, 2002.

[59] Walter Phillips III, Mubarak Shah y Niels da Vitoria Lobo. Flame Recognition in Video. *Pattern Recognition Letters*, 23(1-3):319–327, 2002.

[60] Dengsheng Zhang y Guojun Lu. Segmentation of Moving Objects in Image Sequence: A Review. *Circuits, Systems and Signal Processing*, 20(2):143–183, 2001.

[61] Remi Megret y Daniel DeMenthon. A Survey of Spatio-Temporal Grouping Techniques. Technical Report LAMP-TR-094,CS-TR-4403,UMIACS-TR-2002-83,CAR-TR-979, University of Maryland, College Park, 2002.

[62] Roberta Piroddi y Theodore Vlachos. Multiple-Feature Spatiotemporal Segmentation of Moving Sequences using a Rule-based Approach. *British Machine Vision Conference*, pp. 353–362, 2002.

[63] Sohaib Khan y Mubarak Shah. Object Based Segmentation of Video Using Color, Motion and Spatial Information. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volumen 2, pp. 746–751, December 2001.

[64] Edmon Chalom y Jr. V. Michael Bove. Segmentation of an Image Sequence using Multi-Dimensional Image Attributes. *International Conference on Image Processing*, volumen 2, pp. 525–528, 1996.

[65] D. J. Thirde, G. A. Jones y J. Flack. Spatio-Temporal Semantic Object Segmentation using Probabilistic Sub-Object Regions. *British Machine Vision Conference*, pp. 163–172, September 2003.

[66] Richard O. Duda, Peter E. Hart y David G. Stork. *Pattern Classification*, 2da. edición. New York, Wiley-Interscience, November ©2001.

[67] David Thirde y Graeme Jones. Hierarchical Probabilistic Models for Video Object Segmentation and Tracking. *International Conference on Pattern Recognition*, volumen 1, pp. 636–639, 2004.

[68] Alexander J. Smola, Peter L. Bartlett, Bernhard Schölkopf y Dale Schuurmans. *Advances in Large Margin Classifiers*. The MIT Press, October 2000.

[69] Yoav Freund y Robert E. Schapire. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Second European Conference on Computational Learning Theory*, pp. 23–37, 1995.

[70] Josef Kittler, Mohamad Hatef, Robert P. W. Duin y Jiri Matas. On Combining Classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998.

[71] Ludmila I. Kuncheva, Christopher J. Whitaker, Catherine A. Shipp y Robert P. W. Duin. Is Independence Good For Combining Classifiers? *International Conference on Pattern Recognition*, volumen 2, pp. 2168–2171, 2000.

[72] David M. J. Tax, Martijn van Breukelen, Robert P. W. Duin y Josef Kittler. Combining multiple classifiers by averaging or by multiplying? *Pattern Recognition*, 33(9):1475–1485, 2000.

[73] J. Kittler, M. Hatef y R. P. W. Duin. Combining Classifiers. *International Conference on Pattern Recognition*, volumen 2, pp. 897–901, 1996.

[74] Mario Figueiredo y Anil Jain. Unsupervised Learning of Finite Mixture Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):381–396, March 2002.

[75] Mark Everingham y Barry Thomas. Supervised Segmentation and Tracking of Nonrigid Objects Using a "Mixture of Histograms" Model. *IEEE International Conference on Image Processing*, volumen 1, pp. 62–65, October 2001.

[76] Alessandro Rizzi, Carlo Gatta y Daniele Marini. A New Algorithm for Unsupervised Global and Local Color Correction. *Pattern Recognition Letters*, 24(11):1663–1677, 2003.

[77] David H. Hubel. *Eye, Brain, and Vision*. W.H. Freeman & Company, 1995.

[78] Christian Van den Branden Lambrecht. *Perceptual Models And Architectures For Video Coding Applications*. PhD thesis, École Polytechnique Fédérale de Lausanne, 1996.

[79] Rafael Gonzalez y Richard Woods. *Tratamiento digital de imágenes*, volumen 1. Addison-Wesley Iberoamericana, 1996.

[80] Gernot Hoffmann. CIE Color Space. WWW, 9 de febrero de 2005. `http://www.fho-emden.de/~hoffmann/ciexyz29082000.pdf`.

[81] Pantone matching system©. WWW, 9 de febrero de 2005. `http://www.pantone.com/`.

[82] John C. Russ. *The Image Processing Handbook*, 2da. edición. Boca Raton, CRC Press, 1994.

[83] Vezhnevets V., Sazonov V. y Andreeva A. A Survey on Pixel-Based Skin Color Detection Techniques. *Proc. Graphicon*, pp. 85–92, September 2003.

[84] J. L. Barron, D. J. Fleet y S. S. Beauchemin. Performance of Optical Flow Techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.

[85] Simon Baker y Iain Matthews. Lucas-Kanade 20 Years On: A Unifying Framework. *International Journal of Computer Vision*, 56(3):221–255, 2004.

[86] Milan Sonka, Vaclav Hlavac y Roger Boyle. *Image Processing: Analysis and Machine Vision*, 2da. edición. Thomson-Engineering, September 1998.

[87] Hayit Greenspan, Jacob Goldberger y Arnaldo Mayer. Probabilistic Space-Time Video Modeling via Piecewise GMM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(3):384–396, March 2004.

[88] P. Wayne Power y Johann A. Schoonees. Understanding Background Mixture Models for Foreground Segmentation. *Proceedings of Image and Vision Computing New Zeland*, 2002.

[89] Ning Xu, Ravi Bansal y Narendra Ahuja. Object Segmentation Using Graph Cuts Based Active Contours. *Computer Vision and Pattern Recognition*, volumen 2, pp. 46–53, 2003.

[90] David Hasler y Sabine Süsstrunk. Using Colour to Model Outliers. *Color Imaging Conference*, pp. 107–114, 2003.

[91] Stephen J. McKenna, Yogesh Raja y Shaogang Gong. Tracking colour objects using adaptive mixture models. *Image and Vision Computing*, 17(3-4):225–231, 1999.

[92] L. Lucchese y S. K. Mitra. Color image segmentation: a state of the art survey. *Proceedings of the Indian National Science Academy (INSA-A)*, volumen 67, pp. 207–221, March 2001.

[93] John Y. A. Wang y Edward H. Adelson. Representing Moving Images with Layers. *IEEE Transactions on Image Processing*, 3(5):625–638, September 1994.

[94] Michal Irani, Benny Rousso y Shmuel Peleg. Computing Occluding and Transparent Motions. *International Journal of Computer Vision*, 12(1):5–16, February 1994.

[95] Michael J. Black y Allan D. Jepson. Estimating optical flow in segmented images using variable-order parametric models with local deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(10):972–986, 1996.

[96] Michael J. Black y P. Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 63(1):75–104, January 1996.

[97] Noel Brady y Noel O'Connor. Object Detection and Tracking Using an EM-Based Motion Estimation and Segmentation Framework. *Proceedings of the 8th IEEE International Conference on Image Processing*, volumen 1, pp. 925–928, September 1996.

[98] Yu-Pao Tsai, Chih-Chuan Lai, Yi-Ping Hung y Zen-Chung Shih. A Bayesian Approach to Video Object Segmentation via Merging 3-D Watershed Volumes. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(1):175–180, 2005.

[99] Y. H. Yang y M. D. Levine. The Background Primal Sketch: An Approach for Tracking Moving Objects. *Machine Vision Application*, 5:17–34, 1992.

[100] Yaakov Tsaig. Automatic Segmentation of Moving Objects in Video Sequences. Master's thesis, Tel-Aviv University, 2001.

[101] Lucia Ballerini. Multiple Genetic Snakes for People Segmentation in Video Sequences. *13th Scandinavian Conference Image Analysis*, volumen 2749/2003, pp. 275–282. Springer-Verlag Heidelberg, 2003.

[102] Automatic Segmentation of Moving Objects in Video Sequences: A Region Labeling Approach. Yaakov Tsaig and Amir Averbuch. *Circuits and Systems for Video Technology, IEEE Transactions on*, 12(7):597–612, July 2002.

[103] G. Gordon, T. Darrell, M. Harville y J. Woodfill. Background Estimation and Removal Based on Range and Color. *Proceedings of the Computer Vision and Pattern Recognition*, volumen 2, pp. 459–464, June 1999.

[104] Michael Harville, Gaile Gordon y John Woodfill. Adaptive Video Background Modeling Using Color and Depth. *IEEE International Conference on Image Processing*, volumen 3, pp. 90–93, October 2001.

[105] Bruce D. Lucas y Takeo Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. *Proceedings of the 1981 DARPA Image Understanding Workshop*, pp. 121–130, April 1981.

[106] Ahmed M. Elgammal, Ramani Duraiswami y Larry S. Davis. Efficient Kernel Density Estimation Using the Fast Gauss Transform with Applications to Color Modeling and Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(11):1499–1504, 2003.

[107] Ahmed Elgammal, David Harwood y Larry Davis. Non-parametric Model for Background Subtraction. *European Conference on Computer Vision*, volumen 2, pp. 751–767, June 2000.

[108] Nir Friedman y Stuart J. Russell. Image Segmentation in Video Sequences: A Probabilistic Approach. *Uncertainty in Artificial Intelligence*, pp. 175–181, 1997.

[109] B. Stenger, V. Ramesh, N. Paragios y F. Coetzee. Topology Free Hidden Markov Models: Application to Background Modeling. *International Conference on Computer Vision*, pp. 294–301, 2001.

[110] Chris Stauffer y W. E. L. Grimson. Adaptive Background Mixture Models for Real-Time Tracking. *Computer Vision and Pattern Recognition*, pp. 2246–2252, 1999.

[111] W. E. L. Grimson, C. Stauffer, R. Romano y L. Lee. Using Adaptive Tracking to Classify and Monitor Activities in a Site. *Computer Vision and Pattern Recognition*, pp. 22–31, 1998.

[112] Takashi Matsuyama, Takashi Ohya y Hitoshi Habe. Background Substraction for Nonstationary Scenes. *Proceedings 4th Assian Conferencein Computer Vision*, volumen 1, pp. 662–667, 2000.

[113] Kentaro Toyama, John Krumm, Barry Brumitt y Brian Meyers. Wallflower: Principles and Practice of Background Maintenance. *International Conference on Computer Vision*, pp. 255–261, 1999.

[114] J. Rittscher, J. Kato, S. Joga y A. Blake. A Probabilistic Background Model for Tracking. *European Conference on Computer Vision*, volumen 2, pp. 336–350, 2000.

[115] Michal Irani y P. Anandan. A Unified Approach to Moving Objects Detection in 2D and 3D Scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(6), June 1998.

[116] Michal Irani, Steve Hsu y P. Anandan. Video Compression Using Mosaic Representations. *Signal Processing: Image Communication, special issue on Coding Techniques for Low Bit-rate Video*, 7(4-6):529–552, November 1995.

[117] Michal Irani y P. Anandan. Video Indexing Based on Mosaic Representations. *Proceedings of the IEEE*, 86(5):905–921, May 1998.

[118] Yaron Caspi y Michal Irani. Spatio-Temporal Alignment of Sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(11):1409–1424, 2002.

[119] Jun Wang, Mohan S Kankanhalli, Weiqi Yan y Ramesh Jain. Experiential Sampling for video surveillance. *First ACM Special Interest Group on Multimedia International Workshop on Video Surveillance*, pp. 77–86. ACM Press, 2003.

[120] E. Stringa, C. Sacchi y C. S. Regazzoni. A Multimedia System for Surveillance of Unattended Railway Stations. *Proceedings of Eupsico*, pp. 1709–1712, 1998.

[121] Alvaro Pardo y Guillermo Sapiro. Vector Probability Diffusion. *IEEE International Conference on Image Processing*, volumen 1, pp. 884–887, Vancouver, BC, Canada, September 10-13 2000.

[122] Andrew Blake y Michael Isard. *Active Contours*. Springer, 2000.

[123] Olivier Faugeras. *Three-Dimensional Computer Vision : a geometric viewpoint*. Artificial Intelligence. Cambrigde Mass., MIT Press, ©1993.

[124] Richard E. Bellman. *Dynamic programming*. Princeton Press, 1957.

[125] Michael A. Trick. A Tutorial on Dynamic Programming. WWW, 4 de diciembre de 2004. `http://mat.gsia.cmu.edu/classes/dynamic/dynamic.html`.

[126] Jr. L. R. Ford y D. R. Fulkerson. *Flows in Networks*. Princeton University Press, 1962.

Índice de tablas

Índice de figuras

Contenido del CD y URL

El CD que se adjunta contiene los siguientes archivos:

- `leerme.txt`: Archivo de texto describiendo el contenido del CD.
- `tesis/tesis.pdf`: esta documentación en formato PDF.
- `presentacion/presentacion_tesis.pdf`: Diapositivas utilizadas en la defensa de la tesis.
- `presentacion/secuencias/01-difusion.avi`: Comparación de los resultados de la difusión VPD y MVPD.
- `presentacion/secuencias/02-foreman.avi`: Segmentación de la secuencia `foreman` utilizando el color y la posición como características.
- `presentacion/secuencias/03-container.avi`: Segmentación de la secuencia `container` utilizando el color y la posición como características.
- `presentacion/secuencias/04-carphone.avi`: Segmentación de la secuencia `carphone` utilizando el color y la posición como características.
- `presentacion/secuencias/05-flower.poscol.avi`: Segmentación de la secuencia `flower garden` utilizando el color y la posición como características.
- `presentacion/secuencias/06-flower.dppos.avi`: Segmentación de la secuencia `flower garden` utilizando la disparidad y la posición como características.
- `presentacion/secuencias/07-flower.dpposcol.avi`: Segmentación de la secuencia `flower garden` utilizando el color, la disparidad y la posición como características.

Requerimientos del sistema

Los requerimientos mínimos para acceder al contenido del CD son:

- PC Pentium 233MHz o superior
- Display de 800×600 16 bits de profundidad de color
- 32MB de RAM
- Lectora de CD 8×.
- Programa para la visualización de los archivos PDF, por ejemplo *Adobe Reader* 4.0 o superior.
- Programa para la visualización de secuencias de video, por ejemplo *Media Player Classic* o *mplayer*.

En `http://iie.fing.edu.uy/~fefo/tesis/` se puede obtener el contenido del CD vía http.

Índice

Apéndices

A Geometría epipolar

La geometría epipolar es la geometría proyectiva que representa la geometría de dos vistas o geometría estéreo. La configuración general se muestra en la figura A.1. Esta geometría depende únicamente de los parámetros intrínsecos de las cámaras y de la posición relativa entre las mismas.

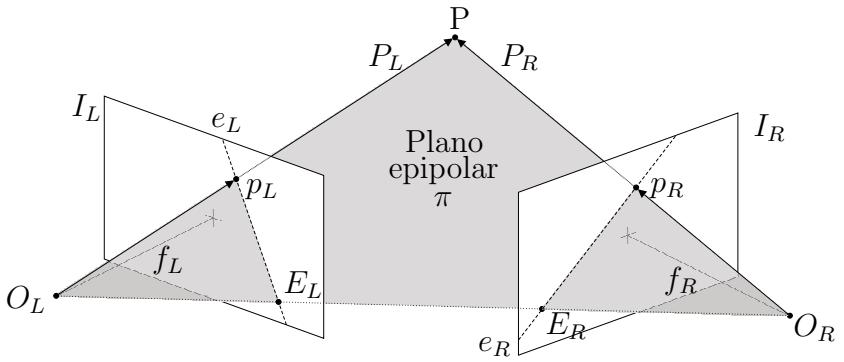


Figura A.1: **Geometría epipolar.**

En la figura A.1 se muestra la configuración de dos cámaras (par estéreo) que cumplen un modelo «pinhole». Sus centros de proyección son O_L y O_R para la cámara izquierda y derecha respectivamente. Los planos I_L y I_R son los planos de la imagen donde se forma la imagen de la escena dada por la proyección por O_L y O_R. Las distancias focales de cada cámara son f_L y f_R, que en el caso de la figura son iguales.

Cada cámara representa un sistema de coordenadas tridimensional cuyo centro es el centro de proyección de la misma, y el eje Z coincide con el eje óptico[1] de la cámara. En estos sistemas las coordenadas de un punto P de la escena se escriben como los vectores $P_L = (X_L, Y_L, Z_L)^\top$ para la cámara izquierda, y $P_R = (X_R, Y_R, Z_R)^\top$ para la cámara derecha.

Los vectores $p_L = (x_L, y_L, z_L)^\top$ y $p_R = (x_R, y_R, z_R)^\top$ definen las proyecciones del punto P en la imagen izquierda y derecha respectivamente, expre-

[1]El eje óptico de la cámara es la recta ortogonal al plano de la imagen por el centro de la cámara.

sados en su correspondiente sistema de referencia. En este caso, dado que los puntos p_L y p_R están en el plano de la imagen respectiva, la coordenada z es igual a la distancia focal $z_L = f_L$ y $z_R = f_R$.

Los sistemas de coordenadas de ambas cámaras están relacionados por los parámetros extrínsecos del par estéreo. Éstos son, una traslación entre los centros ópticos $T = O_R - O_L$, y una rotación R. Así, la relación entre las coordenadas del punto P en ambos sistemas de coordenadas se relacionan mediante

$$P_R = R\,(P_L - T)$$

La recta que une los centros de las cámaras $O_L O_R$ se denomina línea base (*baseline*), y corta a los planos de las imágenes I_L e I_R en los puntos E_L y E_R. Estos puntos son los *epipolos* izquierdo y derecho respectivamente.

Dado un punto P de la escena su proyección en la cámara izquierda p_L se encuentra en la intersección de la recta $O_L P$ con el plano de la imagen izquierda I_L. De forma similar se halla p_R para la cámara derecha. Los puntos O_L, O_R y P definen un plano π que se denomina *plano epipolar*. La línea base pertenece a este plano pues O_L y O_R pertenecen a él por construcción. Más aún, considerando todos los puntos de la escena, cada uno de ellos define junto con O_L y O_R un plano epipolar, de forma que todos los planos epipolares contienen a la línea base. Así, la familia de los planos epipolares forman un haz de planos con eje en la línea base, y todo plano que contenga a la línea base es un plano epipolar.

El plano epipolar interseca a los planos de la imagen I_L e I_R en cada cámara en una recta que se denomina *recta epipolar*, e_L y e_R respectivamente. Todas las rectas epipolares pasan por el epipolo en la imagen respectiva formando un haz de rectas de vértice en el epipolo.

Para un punto P, las proyecciones p_L y p_R pertenecen al plano epipolar que define. Por lo tanto p_L y p_R pertenecen a las rectas epipolares e_L y e_R respectivamente. Ésta es la propiedad fundamental de la geometría epipolar definida, y se utiliza de la siguiente manera. Suponiendo que se conoce sólo p_L, P puede ser cualquiera de los puntos del rayo $O_L p_L$. Este rayo y la línea base definen el plano epipolar de P, π. La intersección de π con I_R la recta epipolar e_R, a la cual pertenece p_R. De esta forma la búsqueda del correspondiente de p_L no es necesario hacerla en toda la imagen I_R, se restringe a una búsqueda lineal en e_R.

Dependiendo de la posición relativa de las cámaras hay tres posibles configuraciones diferentes de los epipolos. La primera es la que se muestra en la figura A.1 con los dos epipolos con coordenadas finitas en la imagen respec-

tiva. La segunda es la que se muestra en la figura 2.1 con ambos epipolos en el infinito. La tercera es un caso intermedio con un epipolo con coordenadas finitas y el otro en el infinito.

El caso que se plantea en la primera parte de esta tesis es el segundo, con ambos epipolos en el infinito, pues la línea base es paralela a los planos de las imágenes. De esta forma el haz de rectas epipolares en cada imagen se transforma en un conjunto de rectas paralelas en la dirección dada por el epipolo. De esta forma, en la configuración que se muestra en la figura 2.1 las rectas epipolares son las filas de las imágenes izquierda y derecha.

Para profundizar en este tema y en las relaciones geométricas que se definen se cita y deja como referencias los libros de Faugeras [123] y de Hartley y Zisserman [8].

B Programación dinámica

El algoritmo de Programación Dinámica fue introducido en 1957 por R.E. Bellman [124] y permite resolver problemas de optimización combinatoria en una secuencia de N decisiones. Se basa en el concepto de que si la secuencia de decisiones $\{D_1, D_2, \cdots, D_N\}$ es óptima (respecto a alguna medida), entonces las últimas k decisiones $\{D_{N-k+1}, D_{N-k+2}, \cdots, D_N\}$ deben ser óptimas. O sea, si cada una de las decisiones que se realizan es óptima el total de las decisiones lo será.

Las características generales de todo algoritmo de Programación Dinámica se pueden resumir en [125]:

1. El problema de toma de una decisión óptima puede dividirse en etapas donde se requiere tomar una decisión en cada etapa.

2. Cada etapa tiene asociada un número de estados.

3. La decisión en una etapa toma uno de los posibles estados y altera los posibles estados de la etapa siguiente.

4. Dado un estado, la decisión a tomar no depende de los estados o decisiones pasadas.

El uso de la Programación Dinámica en la resolución de problemas de cálculo de disparidad ha sido ampliamente utilizada [12, 28, 11, 25, 27] debido a su facilidad de programación y velocidad. Los diferentes algoritmos varían la construcción y asignación de costos para luego usar la Programación Dinámica para la minimización del costo global.

C Corte de grafos (Max-Flow/Min-Cut)

Para comprender la nomenclatura y terminología de los algoritmos basados en corte de grafos, presentamos una breve introducción a esta teoría.

Grafos. Un grafo G es un par $G = (V, E)$, donde V es un conjunto finito de *vértices* o *nodos*, y E tiene como elementos subconjuntos de V de cardinalidad dos, que forman los *enlaces* entre los *vértices*. Es común asignar una dirección a los enlaces entre los nodos, por lo cual el conjunto E pasa a ser un *conjunto ordenado de enlaces*, que suelen llamarse *arcos*. Estos arcos además tienen un *peso* asignado, interpretable como el costo de recorrer ese arco. Asimismo se añaden dos vértices más con características particulares, *la fuente*, s, y *el terminal*, t. Los arcos de la fuente son todos con dirección saliente hacia el conjunto de nodos; mientras que los arcos del terminal son todos con dirección entrante desde el conjunto de nodos. El conjunto $N = (s, t, V, E, b)$ se conoce como una *red*, donde b es una *cota* para el valor que toma el peso en cada uno de los arcos. [33]

Un *corte* C es una partición $C = (V^s, V^t)$ del conjunto de nodos en dos subconjuntos, V^s y V^t, de forma que $s \in V^s$ y $t \in V^t$. El *costo* asociado a un corte, $|C|$, es la suma de los pesos de los arcos que unen V^s con V^t.

Max-Flow/Min-Cut. El problema de Max-Flow/Min-Cut se basa en hallar un corte del grafo con costo mínimo. Este problema es similar (igual solución) al de hallar el máximo flujo entre la fuente y el terminal, como fue planteado por Ford y Fulkerson [126].

Este problema se puede interpretar como el de hallar la máxima cantidad de agua que puede enviarse desde la fuente hasta el terminal, si se piensan los arcos como tuberías con pesos dados por el caudal que pueden transportar. Ford y Fulkerson demostraron que esto se logra cuando se *satura* un conjunto de arcos (tuberías) que divide el grafo en dos –un corte– y la suma de sus pesos (caudales) es mínima. Además, el valor del máximo flujo que se envía coincide con la suma del caudal de las tuberías saturadas (costo

del corte mínimo).

D Difusión de probabilidades

Resultado 1 Primero demostraremos que la difusión planteada en la ecuación (10.6) no difunde a través de los bordes de la imagen, donde su gradiente es alto. La ecuación a estudiar es

$$\frac{\partial p}{\partial t} = \vec{\nabla}.\left(\frac{\overrightarrow{\nabla p} - \langle \vec{u}, \overrightarrow{\nabla p}\rangle \vec{u}}{\left\| \overrightarrow{\nabla p} - \langle \vec{u}, \overrightarrow{\nabla p}\rangle \vec{u} \right\|_2}\right) \quad \text{donde} \quad \vec{u} = \frac{\overrightarrow{\nabla L}}{\left\| \overrightarrow{\nabla L} \right\|} = \frac{(L_x, L_y)^\top}{\sqrt{L_x^2 + L_y^2}}$$

siendo L la componente de luminancia de la imagen. Se usará la notación $\vec{u} = (u_x, u_y)^\top$, y se omite el uso del $\vec{\cdot}$ en los vectores para tener una notación más clara. Entonces

$$\frac{\partial p}{\partial t} = \nabla.\left(\frac{\nabla p - \langle u, \nabla p\rangle u}{\|\nabla p - \langle u, \nabla p\rangle u\|_2}\right) =$$

$$\nabla.\left(\frac{\nabla p}{\|\nabla p - \langle u, \nabla p\rangle u\|_2}\right) - \nabla.\left(\frac{\langle u, \nabla p\rangle u}{\|\nabla p - \langle u, \nabla p\rangle u\|_2}\right)$$

Notando $g = \left\|\nabla p - \langle u, \nabla p\rangle u\right\|_2^{-1}$, la ecuación anterior, queda

$$\frac{\partial p}{\partial t} = \nabla.\big(g \nabla p\big) - \nabla.\big(g \langle u, \nabla p\rangle u\big) \tag{D.1}$$

Por otro lado, notando $\nabla p = (p_x, p_y)^\top$

$$\langle u, \nabla p \rangle u = (u_x\, p_x + u_y\, p_y)(u_x, u_y)^\top = (u_x^2 p_x + u_y u_x p_y, u_x u_y p_x + u_y^2 p_y)^\top$$

Esta última ecuación se puede escribir de forma matricial como

$$\langle u, \nabla p \rangle u = \begin{pmatrix} u_x^2 & u_x u_y \\ u_x u_y & u_y^2 \end{pmatrix} (p_x, p_y)^\top = A\ (p_x, p_y)^\top \tag{D.2}$$

Donde se define la matriz

$$A = \begin{pmatrix} u_x^2 & u_x u_y \\ u_x u_y & u_y^2 \end{pmatrix} = (u_x, u_y)^\top (u_x, u_y)$$

Sustituyendo (D.2) en la ecuación (D.1)

$$\frac{\partial p}{\partial t} = \nabla.\Big(g(Id - A)\nabla p\Big) = \nabla.\left(g\tilde{A}\nabla p\right) \tag{D.3}$$

donde $\tilde{A} = (Id - A)$ e Id es la matriz identidad 2×2. La matriz A (y por lo tanto $\tilde{A}$) depende de los valores de los gradientes de la imagen en cada pixel de la misma.

La ecuación (D.3) plantea una difusión *similar* a la ecuación original de VPD (ecuación (10.3)), agregando la matriz $\tilde{A}$ en el cálculo de la difusión *en cada pixel* de la imagen.

Para analizar la influencia de $\tilde{A}$ se calculan sus valores propios como

$$|\tilde{A} - \lambda I_d| = \begin{vmatrix} (1 - u_x^2) - \lambda & -u_x u_y \\ -u_x u_y & (1 - u_y^2) - \lambda \end{vmatrix} = \lambda\,(\lambda - 1)$$

donde se usó que $\|u\|^2 = u_x^2 + u_y^2 = 1$. Así, $\tilde{A}$ tiene dos valores propios $\lambda_1 = 0$ y $\lambda_2 = 1$ *independientes del pixel* de la imagen donde se aplique. Los valores propios asociados son:

$$\begin{array}{lll} \lambda_1 = 0: & v_1 = (u_x, u_y)^\top & \Rightarrow \quad v_1 \parallel u \\ \lambda_2 = 1: & v_2 = (u_y, -u_x)^\top & \Rightarrow \quad v_2 \perp u \end{array}$$

Esto implica que en cada pixel de la imagen la difusión se hace en la dirección del borde ($\vec{v}_2$) y no se difunde en la dirección normal a los mismos ($\vec{v}_1$).

Resultado 2 Ahora consideremos la minimización del funcional planteado en la ecuación (10.2)

$$\min_{p \in \mathcal{P}} J,\, J = \int_\Omega \left\|\nabla p - \langle u, \nabla p\rangle u\right\|_2 \,\mathrm{d}\Omega \tag{D.4}$$

donde $\Omega (\in I\!R^d)$ es el espacio de la imagen.

$$\begin{aligned} \frac{\partial J}{\partial t} &= \frac{\partial}{\partial t}\left(\int_\Omega \left\|\nabla p - \langle u, \nabla p\rangle u\right\|_2 \,\mathrm{d}\Omega\right) \\ &= \int_\Omega \frac{\partial}{\partial t}\sqrt{\langle \nabla p - \langle u, \nabla p\rangle u, \nabla p - \langle u, \nabla p\rangle u\rangle}\,\mathrm{d}\Omega \\ &\overset{\frac{\partial u}{\partial t}=0}{=} \int_\Omega \frac{\langle \frac{\partial \nabla p}{\partial t} - \langle u, \frac{\partial \nabla p}{\partial t}\rangle u, \nabla p - \langle u, \nabla p\rangle u\rangle}{\left\|\nabla p - \langle u, \nabla p\rangle u\right\|_2}\,\mathrm{d}\Omega \end{aligned} \tag{D.5}$$

Haciendo $g = \left\|\nabla p - \langle u, \nabla p\rangle u\right\|_2^{-1}$ y tomando la notación de $\dot{a} = \frac{\partial a}{\partial t}$

$$\frac{\partial J}{\partial t} = \int_\Omega g\langle \nabla \dot{p} - \langle u, \nabla \dot{p}\rangle u, \nabla p - \langle u, \nabla p\rangle u\rangle \, \mathrm{d}\Omega$$

Aplicando distributiva en el producto interno

$$\begin{aligned}\frac{\partial J}{\partial t} = &\int_\Omega g\Big(\langle \nabla \dot{p}, \nabla p\rangle - \langle \nabla \dot{p}, \langle u, \nabla p\rangle u\rangle\Big) \, \mathrm{d}\Omega + \\ &+ \int_\Omega g\Big(\langle u, \nabla \dot{p}\rangle\langle u, \nabla p\rangle \underbrace{\langle u, u\rangle}_{=1} - \langle u, \nabla \dot{p}\rangle\langle u, \nabla p\rangle\Big) \, \mathrm{d}\Omega\end{aligned} \tag{D.6}$$

Dado que $\langle u, u\rangle = 1$ la segunda integral de la igualdad anterior se anula, y agrupando en el producto interno la integral restante

$$\frac{\partial J}{\partial t} = \int_\Omega g\Big(\langle \nabla \dot{p}, \nabla p - \langle u, \nabla p\rangle u\rangle\Big) \, \mathrm{d}\Omega = \int_\Omega g\langle \nabla \dot{p}, \theta\rangle \, \mathrm{d}\Omega$$

con $\theta = \nabla p - \langle u, \nabla p\rangle u$.

Por otro lado

$$\int_\Omega g\nabla \cdot \big(\langle \dot{p}, \theta\rangle\big) \, \mathrm{d}\Omega = \int_\Omega g\langle \nabla \dot{p}, \theta\rangle \, \mathrm{d}\Omega + \int_\Omega g\langle \dot{p}, \nabla \cdot \theta\rangle \, \mathrm{d}\Omega = 0$$

Donde la igualdad con cero se da si se supone que la variación de $\dot{p}$ en los bordes de Ω es nula, y se aplica el teorema de la Green en $\partial\Omega$. Entonces

$$\frac{\partial J}{\partial t} = -\int_\Omega \langle \dot{p}, g\nabla \cdot \theta\rangle \, \mathrm{d}\Omega$$

Lo cual muestra que si la integral es positiva, la dirección de esta minimización será hacia un mínimo. Eligiendo

$$\dot{p} = g\nabla.\theta = \nabla.\left(\frac{\nabla p - \langle u, \nabla p\rangle u}{\|\nabla p - \langle u, \nabla p\rangle u\|_2}\right) \tag{D.7}$$

se hace que el integrando sea $\|g\nabla\theta\|^2$ lo cual logra al dirección de descenso «más rápido».

Resultado 3 La probabilidad p debe permanecer en $\mathcal{P}$, sea $\vec{1}_m = (1, \cdots, 1)$ la dirección normal a $\mathcal{P}$, veremos que la evolución es ortogonal a esta dirección, o sea:

$$\frac{\partial p}{\partial t} \cdot \vec{1}_m = 0$$

Entonces

$$\frac{\partial p}{\partial t} \cdot \vec{1}_m = \left(\nabla \cdot \left(\frac{\nabla p - \langle u, \nabla p \rangle u}{\|\nabla p - \langle u, \nabla p \rangle u\|_2} \right) \right) \cdot \vec{1}_m =$$

$$= \nabla \left(g \left(\nabla p - \langle u, \nabla p \rangle u \right) \right) \cdot \vec{1}_m + g \left(\nabla p - \langle u, \nabla p \rangle u \right) \cdot \underbrace{\nabla \vec{1}_m}_{=0} =$$

$$= \nabla \left(g \left(\nabla p \cdot \vec{1}_m - \langle u, \nabla p \cdot \vec{1}_m \rangle u \right) \right) = 0$$

pues $\nabla p \cdot \vec{1} = \sum_i \frac{\partial p_i}{\partial t} = 0$

Implementación numérica La implementación numérica de la ecuación se realizó basándose en las técnicas de aproximación numérica clásicas para ecuaciones en derivadas parciales, como se plantean en [121].

El estudio completo de la estabilidad de la implementación numérica aún no ha sido terminado, lo cual se planea realizar en trabajo futuro. La estabilidad se verifica empíricamente en la implementación realizada utilizando un paso de discretización en el tiempo Δt pequeño (del mismo orden que el utilizado en [121]).

En la implementación también se realiza un paso de proyección de la probabilidad difundida en cada iteración $p_{ij}^{t+\Delta t}$, garantizando que todas las componentes de $p_{ij}^{t+\Delta t}$ son no negativas y $\|p_{ij}^{t+\Delta t}\|_1 = 1$.

Esta es la última página de la tesis.
Versión del 24 de octubre de 2005.
http://iie.fing.edu.uy/∼fefo/tesis/