Universidad Rey Juan Carlos
Escuela Técnica Superior de Ingeniería de Telecomunicación
Departamento de Sistemas Telemáticos y Computación

Tesis Doctoral

Aplicación de los métodos secuenciales de Monte Carlo al seguimiento visual 3D de múltiples objetos

Pablo Barrera González
Ingeniero de Telecomunicación

Directores de Tesis:
José María Cañas Plaza
Doctor Ingeniero de Telecomunicación

Vicente Matellán Olivera
Doctor en Informática

Pablo Barrera González
Departamento de Sistemas Telemáticos y Computación
Escuela de Técnica Superior de Ingeniería de Telecomunicación
Universidad Rey Juan Carlos
Av. Tulipán s/n
Móstoles 28933 Madrid
Correo electrónico: pablo.barrera@urjc.es
Web: http://gsyc.es/∼barrera

Autorizamos la defensa de la tesis doctoral *Aplicación de los métodos secuenciales de Monte Carlo al seguimiento visual 3D de múltiples objetos* cuyo autor es D. Pablo Barrera González.

Los directores de la tesis

José María Cañas Plaza Vicente Matellán Olivera

Móstoles, 20 de diciembre de 2007

TESIS DOCTORAL:	Aplicación de los métodos secuenciales de Monte Carlo al seguimiento visual 3D de múltiples objetos
AUTOR:	D. Pablo Barrera González
DIRECTORES:	Dr. D. José María Cañas Plaza Dr. D. Vicente Matellán Olivera

El tribunal nombrado para juzgar la Tesis arriba indicada, compuesto por los siguientes doctores:

PRESIDENTE:

VOCALES:

SECRETARIO:

acuerda otorgarle la calificación de:

Móstoles, de de

El Secretario del Tribunal

A las mismas mujeres que siempre están ahí
Esther, Amaya, Tamara y Ruth

Agradecimientos

Al terminar este trabajo me gustaría agradecer a todos los que, día a día, me han ayudado a llegar hasta el punto final. Aunque sé que no ha sido fácil, he podido contar con ellos siempre que los he necesitado (y no han sido pocas veces). Espero que algún día sea capaz de devolverles todo lo que me dan.

Quiero empezar por ti, Ruth, porque sin ti el resto de las cosas no tienen sentido, porque cuando estoy a tu lado todo es mucho más fácil. Gracias por todo tu cariño, por todo tu tiempo, por saber aguantarme y por saber entenderme.

Me resulta difícil expresar en palabras el agradecimiento que tengo hacia mi familia porque siempre me han dado mucho más de lo podré darles nunca. Ellos me han enseñado lo poco que sé sobre las cosas que realmente importan. Gracias.

Quisiera agradecerles también su trabajo, tiempo y, sobre todo, comprensión a José María y Vicente. Ha sido un placer poder trabajar con vosotros y poder aprender algo nuevo cada día. También quisiera agradecer a toda la gente de GSyC, por acogerme en el grupo que se ha convertido en mi segunda casa.

Por último, querría agradecer su apoyo a mis compañeros y amigos. Los que mejor me conocen saben que la memoria no es mi principal característica, así que espero que sepáis perdonarme (otra vez) si no hago una lista con vuestros nombres. Sería larga y estaría llena de ausencias.

Tras todos esto lo único que puedo decir es gracias a todos.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE CUADROS

Abstract

The work for my PhD thesis is presented in this document. The research is focused on the 3D visual tracking of multiple objects using several cameras at the same time. The goal is to simultaneously track several indistinguishable objects but always maintaining real time performance and only using visual information. The system should also provide enough accuracy in large 3D areas, whenever the objects lie inside the scope of the cameras.

We use particle filters, which have been successfully used by the scientific community for the visual tracking problem. They are approximate solutions for the bayesian filtering problem using Monte Carlo techniques. It is straightforward to formulate the tracking problem using a bayesian approach, and so to use particle filter for it.

However particle filters have severe limitations. They are not able to track several objects at same time directly. To trail several objects it is convenient to maintain a multimodal probability density function. Particle filters represent multimodal functions but only for a short period of time. Eventually they lose all modes but one, in the long term they are inherently monomodal.

To overcome such limitation we propose a method that combines two different particle filters into an unified framework. First, a *non-sequential* part (importance sampling) is able to detect new objects and maintains current objects representing a multimodal probability density function. It generates new particles directly from ob-

servations. And second, a *sequential* part (condensation) is able to provide the necessary tracking features and to optimise current estimations, exploiting spatiotemporal continuity in order to provide more accuracy. This part generates new particles using particles from the previous iteration, both sequential and non-sequential.

Particles generated from each method are combined and shared in order to follow all objects at the same time. The combination fulfils the Monte Carlo assumptions. This provides the theoretical underlying support for the convergence and performance characteristics in the method proposed.

Regarding the non-sequential part we also present an *abduction technique*, inside Monte Carlo boundaries, that allows the method to place new particles in promising areas according to visual information. Other particle filters, like condensation algorithm, do not use visual information to place new particles. This is their main limitation to detect new objects.

In order to test and validate it several experiments have been performed. They expose the algorithm to situations involving one to five objects in static and dynamic setups. We provide performance and error measurements that show that the system works correctly with several objects at same time.

To carry on the experiments we have developed an implementation of the proposed system. The implementation has been made in C++ in a Linux machine and works in real time using commodity hardware.

Resumen

En el presente documento se resume el trabajo desarrollado durante mi tesis doctoral. El tema abordado se centra en el seguimiento visual en tres dimensiones de múltiples objetos, usando para ello varias cámaras.

Por seguimiento nos referimos a la habilidad de estimar de manera continua la posición tridimensional. Dicha posición puede ir cambiando a lo largo del tiempo, así que el proceso de seguimiento debe actualizar la estimación de manera continua.

Muchos trabajos han abordado el problema del seguimiento en general, y del seguimiento visual en particular con anterioridad, tanto para el caso de un objeto como el de varios objetos. En nuestro caso queremos centrarnos en el seguimiento visual, aquel que emplea imágenes como única fuente de información.

La ventaja de emplear imágenes es la cantidad de información a la que tenemos acceso empleando un sensor relativamente barato. Sin embargo, la información de las imágenes no es tridimensional, sino que únicamente nos proporciona una proyección 2D de la realidad observada por el sensor. El mundo real es tridimensional, por lo que si obtenemos información 3D nos encontraremos más cerca de la información real, simplificando el uso de ésta. Será necesario combinar varias cámaras para conseguir reconstruir la información 3D existente en la realidad. Además de esta forma podremos proporcionar estimaciones de la posición de los objetos precisas, incluyendo la distancia a la que están situados los objetos.

Existen multitud de aplicaciones posibles para este tipo de técnicas. Podemos pensar en sistemas que van desde la videovigilancia automática al control de procesos industriales. Por ejemplo, en los sistemas de videovigilancia será posible seguir a varias personas al mismo tiempo de manera automática, guardando las trayectorias que sigue cada uno y avisando cuando se encuentren en zonas restringidas o posiciones problemáticas (por ejemplo, caído en el suelo).

El sistema de seguimiento que hemos desarrollado obtiene la información 3D de la combinación de las imágenes obtenidas por dos o más cámaras estáticas colocadas mirando hacia una misma zona o región del espacio. Deseamos que el sistema de seguimiento consiga una buena precisión, incluso en grandes espacios, siempre que todo el área esté cubierta por el campo de visión de las cámaras.

El sistema desarrollado tiene como objetivo principal afrontar el problema del seguimiento de varios objetos al mismo tiempo, por lo que haremos un énfasis mayor en esta capacidad antes que en otras características. Para ello hemos empleado varios objetos indistinguibles entre sí, que es el caso más complejo a la hora de seguir varios objetos al mismo tiempo. Para resolver el problema hemos partido de dos algoritmos diferentes bien conocidos, como son el filtro de condensación y el filtro de muestreo enfatizado. Se han adaptado al problema de seguimiento visual 3D y hemos propuesto una extensión sobre los mismos.

Uno de los problemas a la hora de usar imágenes es el coste computacional que suele ir asociado a su análisis. El objetivo que nos propusimos en esta tesis era que los sistemas empleados deberían ser capaces de procesar las imágenes al mismo ritmo que son capturadas por la cámara. Para cumplir este requisito debemos elegir cuidadosamente la familia de métodos que podemos usar.

De entre todos los métodos de seguimiento posibles nos hemos centrado en los filtros de partículas. Éstos presentan una solución aproximada al problema del filtrado bayesiano secuencial empleando técnicas de Monte Carlo. Las técnicas de Monte Carlo nos permiten aproximar de forma no paramétrica las distribuciones de probabilidad usando para ello números aleatorios. En el caso de los filtros de partículas estas distribuciones serán las del filtrado bayesiano secuencial.

La ventaja de usar aproximaciones basadas en muestras es que resultan menos costosas que mantener una función paramétrica general. En lugar de evaluar las fun-

ciones en todos los puntos posibles sólo es necesario evaluarla en un subconjunto de puntos convenientemente elegidos. La ventaja de este método es clara si pensamos en la reducción de coste asociada. La teoría de los métodos de Monte Carlo nos dice que introduciendo pesos adecuados, los puntos colocados al azar pueden construir una aproximación válida de la distribución de probabilidad objetivo. Su ventaja es aún superior dado que el error de aproximación desciende en función del número de muestras, independientemente de la dimensión del problema.

La ventaja de emplear filtros de partículas frente a otros métodos es que no realizan asunciones sobre las distribuciones de probabilidad implicadas. Por ejemplo, los filtros de Kalman únicamente son capaces de emplear distribuciones de probabilidad gaussianas, que son incapaces de condensar la información acerca de las funciones de densidad de probabilidad como la de la posición de varios objetos.

De entre todas las posibles implementaciones de los filtros de partículas, hay una mucho más extendida que las demás, el muestreo enfatizado secuencial, también conocido como algoritmo de condensación. Su popularidad radica en su sencillez de implementación y sus buenos resultados. Para adaptarlo a nuestro problema en particular, el seguimiento de objetos de un color determinado, únicamente debemos desarrollar dos modelos probabilísticos diferentes: el modelo de observación y el modelo de movimiento.

Ambos modelos reflejan nuestro conocimiento *a priori* sobre el problema que pretendemos abordar. Por un lado, el modelo de observación es el encargado de relacionar la información que capturamos de las cámaras, lo que observamos, con la posición del objeto en un determinado momento. Para nuestro problema en particular usamos un filtrado de color en el espacio HSI. Aunque existen otras posibilidades para realizar dicho modelo que pueden proporcionar mejor precisión, empleamos un modelo tan simple por dos razones. En primer lugar para centrarnos en el desarrollo del algoritmo de seguimiento multiobjeto y, principalmente, en la dinámica del filtro de partículas. En segundo lugar, porque partimos de la hipótesis que resulta posible realizar el seguimiento de un objeto usando poca información visual o combinando estímulos relativamente simples.

El modelo de movimiento guarda la información relativa a la evolución de los objetos. En nuestro caso queremos reducir la información *a priori* necesaria al míni-

mo, por lo que hemos tomado decisiones encaminadas a usar los modelos más genéricos posibles. Partimos del supuesto de continuidad en el movimiento. De esta forma, entre dos instantes de tiempo consecutivos, el objeto estará en posiciones próximas. El modelo propuesto se corresponde a una gaussiana de tres dimensiones centrada en la posición anterior. Los puntos próximos a dicha posición serán más probables que los puntos alejados.

El problema del filtro de condensación es que resulta incapaz de realizar el seguimiento de *varios objetos* al mismo tiempo, por lo que resulta inútil para nuestros objetivos. Sus limitaciones provienen del paso de remuestreo que emplea. Una parte de las partículas colocadas por el algoritmo caerán en zonas de poco interés. Para optimizar el uso de los recursos disponibles, el filtro de condensación descartará las partículas menos relevantes, centrándose en aquellas que resulten más prometedoras. Éste es el proceso de remuestreo. Si aplicamos el descarte indiscriminadamente perderemos partículas que sí pueden ser relevantes a la hora de localizar un segundo, tercer o cuarto objeto. Es decir, se observa que el filtro tiende a converger en distribuciones de probabilidad monomodales a largo plazo. Este efecto es conocido como degradación del filtro de partículas.

Es posible evitar el problema de la degradación si al cambiar las partículas entre una iteración y la siguiente no guardamos información, es decir, tomamos las nuevas partículas fijándonos únicamente en las observaciones actuales. Ese es el funcionamiento del algoritmo de muestreo enfatizado no secuencial, que busca la mejor solución en cada instante de tiempo, sin tener en cuenta la información de tiempos pasados. Aunque esto impone una limitación dura a la calidad de los resultados esperados en cuanto a continuidad se refiere, proporciona una forma sencilla de localizar varios objetos al mismo tiempo. Para ello debe hacerse hincapié en el desarrollo de una función de propuesta, capaz de colocar las partículas en las zonas más relevantes del espacio de estados, a la vista de las observaciones actuales.

Nuestra función de propuesta, a la que denominamos función de abducción, se encarga de colocar las partículas sobre las líneas de visión en 3D de los objetos en cada cámara. Así cubriremos todas las zonas en las que podrían encontrarse los objetos que han producido las observaciones usadas. Al explorar todas las posibles opciones el algoritmo de muestreo enfatizado es capaz de proporcionar estimaciones para cada

objeto, aun empleando un número de partículas limitado.

Los problemas más destacados de este algoritmo aparecen con el movimiento de los objetos. Al descartar la información secuencial es necesario usar muchos recursos para volver a obtener información que ya estaba disponible en la iteración anterior. Es más, en muchos casos, no resulta suficiente, proporcionando mayores errores de seguimiento que los obtenidos por el algoritmo de condensación para un único objeto.

La solución que proponemos en esta tesis consiste en combinar estos dos algoritmos, el de condensación y el de muestreo enfatizado no secuencial, en un único sistema de seguimiento, compensando las deficiencias de cada uno con las ventajas del otro. La combinación se hace de tal forma que no abandonamos el marco probabilístico de las técnicas de Monte Carlo. Dicho marco proporciona una gran solidez teórica y medidas sobre la convergencia, a diferencia de otros algoritmos como pueden ser los genéticos. El algoritmo desarrollado es capaz de realizar un seguimiento suave, gracias a la utilización de la información secuencial por parte del filtro de condensación, sin perder ningún objeto debido al uso de la función de abducción. De hecho, la función de abducción permite detectar nuevos objetos tan pronto como aparecen en escena, dotando de mayor agilidad al sistema de seguimiento.

El método que proponemos para combinar los dos algoritmos es el uso conjunto de partículas colocadas por cada uno de los dos métodos. Al ser independientes entre sí es posible combinarlas en un único marco, ajustando sus pesos convenientemente. Este sistema colocará un mínimo de partículas en zonas que, de otra forma, desaparecerían por la evolución del filtro de condensación. Esta dinámica permite, incluso, descubrir nuevos objetos tan pronto como aparecen, al igual que hace el algoritmo de muestreo enfatizado. Por otra parte, el sistema consigue realizar un seguimiento suave de los objetos gracias a que emplea una parte de las partículas de una iteración en la siguiente, del mismo modo que hace el algoritmo de condensación.

El resto del documento está dividido de la siguiente forma. En primer lugar se realiza una introducción al tema tratado, haciendo énfasis en la relevancia del problema abordado. En el capítulo 2 se presentan una serie de trabajos que han abordado este mismo problema o problemas similares. Mostraremos aproximaciones que también hacen hincapié en la combinación de los dos algoritmos para resolver el problema de la monomodalidad de los filtros de partículas. El capítulo 3 contiene las bases teóricas

tanto del filtrado bayesiano óptimo como de las aproximaciones muestreadas basadas en modelos de Monte Carlo. Se estudiará, sobre todo, los algoritmos de condensación y de muestreo enfatizado no secuencial. Los capítulos 4 y 5 muestran los algoritmos propuestos para realizar el seguimiento tridimensional tanto de manera teórica como de manera práctica. Para ello se han implementado todos los algoritmos citados y se han realizado experimentos con imágenes reales en cada caso. Este trabajo nos ha permitido caracterizar el funcionamiento y, más importante, las limitaciones de los algoritmos de condensación y de muestreo enfatizado no secuencial. Una vez localizadas las limitaciones hemos podido validar el correcto funcionamiento del algoritmo propuesto, la combinación de condensación y muestreo enfatizado. Los experimentos realizados han cubierto los problemas de seguimiento sobre trayectorias controladas, seguimiento sobre movimiento aleatorio y estudios de precisión. Hemos hecho pruebas para un objeto y para un número variable de objetos, comprobando la capacidad del sistema para localizarlos y seguirlos. También hemos probado cómo reacciona el sistema según vamos añadiendo nuevos objetos, cuánto tarda en localizarlos y cómo se reparten los recursos entre ellos.

El capítulo 6 se centra en las cuestiones relacionadas con la implementación de los algoritmos y de la plataforma experimental utilizada. Para terminar, el capítulo 7 contiene un balance sobre el trabajo realizado en esta tesis.

CAPÍTULO 1

Introducción

El presente documento resume mi trabajo de investigación sobre el seguimiento visual multiobjeto en tres dimensiones usando más de una cámara.

Dicho tema ha sido estudiado en profundidad con anterioridad. Con esta nueva revisión del problema queremos aproximarnos a una aplicación práctica del mismo por lo que haremos especial hincapié en cuestiones como la eficiencia y el coste computacional, temas que abordaremos en profundidad en los siguientes capítulos.

La búsqueda de la eficiencia pone restricciones a la familia de técnicas que podemos usar para implementar el seguimiento. En particular hemos elegido los métodos secuenciales de Monte Carlo como base a la hora de buscar una forma ligera de realizar un seguimiento tridimensional. Al aplicar estos métodos hemos corroborado varios problemas anteriormente detectados a la hora de afrontar el seguimiento de varios objetos. Con esta tesis pretendemos resolver algunos de los problemas planteados y construir un sistema completo, basado en métodos de Monte Carlo, capaz de seguir a un número variable de objetos al mismo tiempo.

En este primer capítulo centraremos estos y otros temas dentro del marco en el que vamos a trabajar: la visión por computador. Desde ese campo genérico, que abarca multitud de problemas, avanzaremos a la visión 3D y, por último, a la localización y seguimiento, eje principal sobre el que vamos a desarrollar el presente trabajo. Una

vez centrado el problema pasaremos a perfilar tanto las hipótesis de partida como los objetivos buscados con esta tesis. Para cerrar el capítulo introductorio detallaremos la metodología de desarrollo seguida y la estructura del documento.

Para comenzar con esta introducción mencionaremos en primer lugar cuáles han sido las motivaciones que nos han llevado a abordarla.

1.1. Motivación

La visión por computador es un campo apasionante que cuenta ya con más de 40 años de historia. Siempre ha estado muy vinculado con el desarrollo de la potencia de cálculo de los ordenadores, siendo este factor limitante en muchas ocasiones. En los últimos años, a medida que prolifera el número de cámaras en nuestra vida cotidiana y aumenta la capacidad de cómputo de los ordenadores, se ha reavivado el interés en la visión por computador. Actualmente, los ordenadores disponen de la suficiente potencia para abordar aplicaciones que hasta ahora sólo aparecían en los sueños de muchos investigadores.

El campo de la visión por computador ha aparecido siempre ligado a la inteligencia artificial. La visión es, sin duda, nuestro principal sentido. Gracias a ella somos capaces de recibir información acerca del mundo que nos rodea, información que empleamos para desarrollar nuestra inteligencia y para interactuar con el mundo. La visión por computador se ha visto como una forma de aproximarnos al entendimiento de nuestra propia visión, y de ahí a nuestra propia inteligencia. Conseguir desarrollar sistemas de visión tan buenos, o más, como los nuestros implica conocernos un poco más a nosotros mismos, ser capaces de reproducir nuestras capacidades de manera artificial.

Sin embargo, desde los primeros momentos del desarrollo de la visión por computador se vio la dificultad intrínseca de estos sistemas. Tareas que los humanos realizamos de manera automática, sin dificultad aparente, se han demostrado como problemas titánicos para nuestras máquinas. Reconocer la cara de una persona es algo automático para nosotros, pero los ordenadores han tardado años en poder hacer esto de manera eficaz y aún así de una forma mucho más limitada a lo que nosotros somos capaces. Estos contratiempos detuvieron el ímpetu inicial de este campo y lo han

reconducido a nuevas aproximaciones.

En vez de intentar reproducir la visión humana de manera completa, la visión por computador redujo sus aspiraciones a la resolución de problemas aparentemente más sencillos. Con esto se esperaba comprender mejor la naturaleza del problema. La división de la visión por computador en tareas más sencillas tenía como objetivo estudiar el problema desde una perspectiva de *divide y vencerás* y construir aplicaciones más complejas basadas en los bloques básicos desarrollados.

El seguimiento visual, y de manera análoga la localización, son unas de estas tareas. El objetivo del seguimiento consiste en mantener una estimación de la posición de un objeto en una imagen a lo largo del tiempo. El interés de este problema viene en que es parte fundamental de muchos otros desarrollos, como pueden ser el reconocimiento de objetos o la autolocalización basada en visión. El seguimiento nos dirá dónde están los objetos que nos interesan, permitiéndonos centrarnos únicamente en esas zonas de la imagen, ignorando el ruido introducido por aquellas cosas que no nos interesan.

Sin embargo, uno de los problemas que plantea el seguimiento es cómo realizarlo de manera robusta mientras se mantiene el coste computacional acotado. El seguimiento normalmente forma parte de sistemas más completos que suelen depender de él para su robustez y con los que tiene que compartir los recursos computacionales. Mejorar estos dos puntos al mismo tiempo resulta interesante por sí mismo.

En el caso particular del seguimiento visual existen muchas y variadas técnicas, pero varias de ellas adolecen de un alto coste computacional aunque proporcionan resultados muy precisos. Nuestra motivación es proporcionar métodos mucho más ligeros que aún así sean suficientemente precisos y robustos.

Algunos de los métodos más rápidos de seguimiento son los basados en técnicas probabilísticas. Estamos hablando principalmente de filtros de Kalman y filtros de partículas (métodos secuenciales de Monte Carlo). Aunque estos métodos son muy robustos para el seguimiento de un único objeto fallan de manera estrepitosa al enfrentarse al seguimiento de *múltiples* objetos al mismo tiempo. Proporcionar alternativas, dentro del marco probabilístico, capaces de enfrentarse a este problema marca la motivación última de esta tesis.

1.2. Visión por computador

La visión por computador puede definirse como el estudio y la aplicación de métodos cuyo objetivo es extraer información sobre el mundo real empleando para ello imágenes. La visión por computador estudia y describe sistemas técnicos de visión implementados como *software* o *hardware*, en ordenadores o procesadores digitales de señales.

Éste es un campo que envuelve a muchos otros y que tiene mucha relación con la visión biológica, que ha usado como referencia y como fuente de inspiración. Aún así, en general, la visión por computador se aproxima al problema del procesado de información visual únicamente como un problema técnico. Aunque conseguir reproducir completamente un sistema de visión biológico puede resultar interesante, la visión por computador es mucho más pragmática e intenta buscar soluciones técnicas a problemas basados en visión. Comparándolo con la visión biológica, la visión por computador es un campo todavía muy inmaduro.

Existen grandes diferencias entre la visión por computador y la visión biológica. Aunque el origen de la visión por computador fuera realizar sistemas similares a los biológicos, la imposibilidad de obtener resultados adecuados modificó las aspiraciones de la visión por computador hasta convertirlo en el campo pragmático que es hoy en día. Aún así resulta útil seguir comparando estos dos campos para entender la importancia de la visión por computador.

Pensemos por un momento en la visión biológica. La visión es nuestro sentido principal. Esto no es una casualidad. La cantidad de información que recibimos a través de nuestra visión es inmensa. Otros sentidos nos proporcionan mucha menos información (aunque a veces muy útil). Por ejemplo el tacto sólo nos da información de presión en nuestras inmediaciones. El olfato nos lleva un poco más lejos de justo lo que tenemos al lado pero la información que recibimos es más bien limitada. Con esta información limitada podemos hacernos una idea aproximada de lo que tenemos a nuestro alrededor, del entorno en el que estamos. Esta es la forma en la que una persona invidente se construye una representación interna de un objeto, palpándolo con sus manos.

La visión, en cambio, nos proporciona mucha información acerca de nuestro

entorno con tan solo un vistazo. Esta información suele ser muy precisa, y es la combinación de su precisión y su velocidad la que le proporciona su destacado papel entre nuestros sentidos. Dada esta importancia hemos estructurado nuestro mundo de manera visual. Toda la información relevante aparece en nuestro entorno en forma de señales visuales. Avisos, señales de tráfico, indicaciones, anuncios, mapas, etc. tienen en común su naturaleza visual. Incluso nuestra comunicación tiene un componente visual indiscutible.

La visión está muy relacionada con el propio desarrollo de la inteligencia humana, dado que modela cómo podemos acercarnos al mundo que nos rodea. Comprender los procesos asociados a nuestra forma de ver el mundo y, sobre todo, de extraer información útil de lo que vemos, es un paso importante hacia el entendimiento de nuestra manera de pensar. Por este motivo la visión por computador ha estado ligada tradicionalmente con la inteligencia artificial. Desde el principio se pusieron muchas esperanzas en el desarrollo tanto de la visión por computador como de la inteligencia artificial, pero la complejidad tan grande asociada con estos dos campos echó un jarro de agua fría a las metas iniciales de los investigadores.

Por este motivo se tendió a reducir las pretensiones del desarrollo de la visión por computador, alejándola de la comprensión humana y aproximándola a una serie de problemas más acotados y definidos. Esto ha separado estos dos campos aunque siguen estando muy relacionados. Los ordenadores y los robots combinan técnicas de inteligencia artificial con técnicas de visión por computador para interactuar con el mundo real o con usuarios humanos. Son capaces de interactuar o trabajar entre nosotros gracias a esta combinación. Campos como el aprendizaje automático de formas o el reconocimiento de patrones caen en la intersección de la inteligencia artificial y la visión por computador.

Debido a su origen y a su estado actual podríamos clasificar a la visión por computador como parte de la inteligencia artificial o de las ciencias de la computación en general.

Independientemente del marco en que se coloque la visión por computador, parece obvio que es necesario proporcionar métodos para que las máquinas que nos rodean comprendan mejor la información visual. La comprensión de estos estímulos visuales es uno de los mayores obstáculos para la proliferación de dispositivos

autónomos en nuestra vida cotidiana.

Varios han sido los factores que han limitado el desarrollo de aplicaciones de visión computacional hasta hace relativamente poco tiempo. El aumento de la capacidad de cómputo y del número de ordenadores, junto con la caída de precios de las cámaras digitales, ha hecho que crezca la importancia de este campo dentro de la comunidad científica.

A día de hoy podemos encontrar cámaras en muchos lugares. Los teléfonos móviles están equipados con una, las webcams están conectadas, o incluso integradas, a casi cualquier ordenador moderno y las cámaras digitales de bolsillo son más baratas que nunca. La economía de escala y las nuevas tecnologías ha permitido fabricar cientos de miles de cámaras cada día, cada vez a un menor precio mientras sus prestaciones siguen aumentando.

Aún así las cámaras por si solas no son capaces de proporcionar ningún tipo de significado a las imágenes, son meros elementos de captura instantáneos. Es la visión por computador la que se encarga de proporcionar un significado a la gran cantidad de datos que nos dan los sensores visuales.

Hasta ahora hemos hablado de la extracción de información de las imágenes capturadas, pero más importante aún que sacar la información, es emplearla de manera correcta. Según sean nuestras necesidades, buscaremos determinado tipo de información y los métodos empleados en cada caso variarán. La información extraída se emplea para dos propósitos generales. Por un lado extraemos información con el fin de dársela a un operador humano y así aumentar su compresión de las imágenes. Por otro, podemos buscar información con el fin de construir sistemas autónomos, capaces de reaccionar ante estímulos variados sin intervención humana.

Como ejemplo del primer propósito podemos ver el procesamiento de imágenes médicas. El objeto en este caso es mejorar las imágenes de tal forma que, a la vista de un operador cualificado, sea más fácil diagnosticar. Será este operador el que, basándose en su conocimiento y en su experiencia, obtenga la información de la imagen y determine si existe algún tipo de anomalía o enfermedad. El objetivo del sistema de visión por computador consistirá en conseguir la mejor imagen posible, resaltando las áreas necesarias y eliminando las ruidosas, para que no confundan al operador.

El segundo suele servir para dar retroalimentación a algún proceso de control

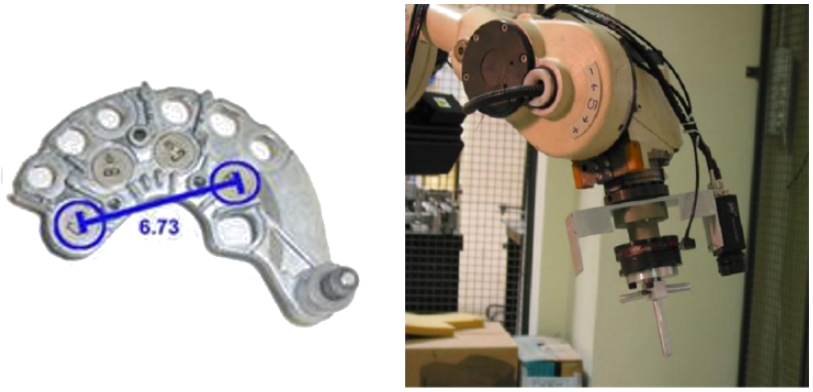

Figura 1.1: Visión por computador en la industria. Sistema de control de calidad que mide una pieza usando visión, con una cámara montada sobre un brazo robótico.

automático. En este apartado se engloban todos los sistemas de control autónomo basados en visión, como sistemas de vigilancia, robots móviles, controles industriales, etc. Un ejemplo de estas aplicaciones puede ser el control de calidad industrial mediante cámaras (figura 1.1). El objetivo consiste en detectar piezas defectuosas en una cadena de montaje. Una manera fácil de hacerlo es atender a su forma. Usando una cámara se pueden ir mirando todas las piezas y detectar aquellas que son defectuosas. Hacerlo de manera automática nos proporciona un menor número de errores y mayor velocidad de la que podría dar un operador humano dado que una máquina puede trabajar sin descanso alguno.

En ambos casos las técnicas a usar son similares aunque cada una hace énfasis en diferentes matices, sobre todo en las asunciones de partida y en las simplificaciones posteriores. Los tipos de datos usados en ambos casos son similares: imágenes estáticas o secuencias de vídeo tanto en color como en blanco y negro.

Áreas de aplicación

Resulta difícil describir todos los campos dónde se utiliza la visión por computador. Resulta aún más difícil diferenciar entre campos tan afines como la visión por computador, el procesado de imágenes, el análisis de imágenes y la visión en robots. Todos son campos muy próximos entre sí y, en muchos casos, están solapados. Si miramos dentro de libros con cualquiera de esos nombres en la portada, descubri-

remos que hay un solape significativo en los términos, en las técnicas e incluso en las aplicaciones que cubren. Esto implica que las técnicas básicas que se emplean y desarrollan en cada uno de esos campos son, más o menos, idénticas, lo que se puede interpretar como que existe un único gran campo pero que toma diferentes nombres.

En nuestro caso usaremos el término "visión por computador" para el campo en general, aunque a veces nos centraremos en un subcampo en particular: la visión robótica. De este subcampo tomaremos sobre todo los requisitos y restricciones de funcionamiento. Como hemos visto hasta ahora, parece difícil dar una descripción formal del campo de la visión por computador. En cambio intentaremos mostrar su estado actual en una serie de áreas de aplicación populares en este campo.

Sin duda una de las áreas de aplicación que resulta más representativa es la relacionada con la medicina o el procesamiento de imágenes médicas. En las aplicaciones médicas, el objetivo suele ser extraer información para facilitar el diagnóstico de un paciente. Normalmente las imágenes usadas son de microscopios, rayos X, angiografías, ultrasonidos o tomografías. Usando estas imágenes se pueden detectar tumores, arteriosclerosis o malformaciones dañinas, medir el flujo sanguíneo, el tamaño de los órganos, etc. Este área de aplicación es tan extensa e importante que incluso proporciona nuevos campos de investigación, como son el estudio de la estructura del cerebro o estudios sobre la calidad de los tratamientos médicos. Gracias al procesamiento de imágenes médicas existen investigaciones o aplicaciones que resultarían completamente imposibles de realizar de otra forma como, por ejemplo, el uso de microarrays para el estudio genético o los sistemas de recuento de células en tejidos.

La industria manufacturera es otra importante área de aplicación en la visión por computador. En este caso la información se extrae generalmente para controlar algún tipo de proceso de fabricación. Los sistemas de control de calidad son un claro ejemplo. Los detalles de los productos finales se inspeccionan de manera automática empleando cámaras para encontrar algún tipo de anomalía y así descartar las piezas defectuosas.

Otro área que podemos comentar son las aplicaciones militares. Quizá es una de las áreas dónde más trabajo se realiza a pesar de que tan sólo una pequeña parte de este trabajo se encuentre disponible al público. Las nuevas tácticas militares emplean multitud de sensores, incluyendo cámaras, para conseguir un rico abanico de informa-

ción sobre el terreno de combate. Su objetivo es tener información de primera mano y fiable con el fin de ayudar en la confección de la estrategia adecuada. El análisis automático combinado con la fusión de información entre diferentes tipos de sensores se emplea para reducir la complejidad de los datos obtenidos. Reduciendo la complejidad se llega a representaciones más manejables y fiables que pueden usar operadores humanos. De otra forma no se abarcaría tal cantidad de información. Ejemplos obvios son la detección de soldados enemigos o vehículos. Pero este no es el único uso de la visión por computador en este ámbito. En los diseños armamentísticos y en la construcción de vehículos se tiene cada vez más en cuenta el uso de estas técnicas. Como ejemplo podemos citar los nuevos sistemas de guiado de misiles. Los métodos clásicos usan receptores GPS o sistemas de posicionamiento similares para dirigir el misil hacia su objetivo en un área determinada. Los más avanzados emplean la visión por computador para realizar la selección del objetivo en los ordenadores colocados en el propio misil, procesando las imágenes que él mismo obtiene cuando alcanza el área deaseada. Esto se hace para conseguir mayor precisión en la elección del blanco dado que el misil tendrá acceso a información más completa cuando alcance el área objetivo.

Otro área, puede que más conocida que las anteriores, es la creación de efectos visuales. Estamos acostumbrados a ver este tipo de efectos en películas, series de televisión o incluso videojuegos. El tipo de efectos de los que estamos hablando van desde el retoque de imágenes para mejorar su contraste, hasta la creación de imágenes completamente sintéticas. Puede parecer que la creación de imágenes sintéticas es un problema fácil de abordar. Sin embargo esto no es así. A la hora de integrar estas imágenes con escenas reales aparecen muchos problemas que es necesario resolver. El movimiento de la cámara real, su orientación, el ajuste de condiciones de iluminación o incluso la interacción con objetos de la realidad son tareas que deben abordarse para buscar una apariencia verosímil en este tipo de efectos.

Gracias a los nuevos avances en la visión por computador este tipo de efectos no está confinado únicamente a las películas, dónde todo el procesado se realiza cuidadosamente a lo largo de varios meses. Sistemas como *Eyevision(TM)*[1] permiten modelar de manera fotorrealista una escena con el fin de construir nuevas imágenes

[1] http://www.ri.cmu.edu/events/sb35/tksuperbowl.html

Figura 1.2: Eyevision(TM) es un sistema capaz de generar nuevas vistas fotorrealistas de una real escena usando varias cámaras al mismo tiempo.

virtualizadas de la misma en tiempo real. Este sistema ha sido empleado en la Superbowl para mostrar la repetición de las jugadas desde cualquier ángulo. Para ello se colocan varias cámaras alrededor del campo de juego, grabando la escena continuamente. El sistema controla automáticamente la orientación y el foco de todas las cámaras para que apunten al mismo área. Las imágenes tomadas se envían a un conjunto de ordenadores que las procesa para construir un modelo 3D de la escena muy preciso. La resolución del modelo es tal que se pueden crear nuevas imágenes de la misma escena desde direcciones en las que no mira ninguna cámara. Con esas imágenes se construyen impresionantes repeticiones para mostrar tomas o movimientos de cámara imposibles de hacer de otra forma, proporcionando un resultado similar a los efectos especiales de la película *Matrix* pero, en este caso, en casi tiempo real (ver figura 1.2).

Tareas

Hasta ahora hemos descrito el campo de la visión por computador en términos de áreas de aplicación. Otra aproximación para intentar describir este campo es describirlo empleando las tareas que generalmente se abordan en las aplicaciones. Cada área de aplicación descrita anteriormente emplea un conjunto de tareas o técnicas similares aun cuando no exista una formulación común.

Lo que sí existe son abundantes métodos particulares para resolver varias tareas bien definidas. En algunos casos estos métodos se pueden generalizar y aplicar en otros problemas. A continuación se presentan algunos ejemplos típicos de tareas específicas dentro de la visión por computador que tienen cierta relevancia. Éstas se

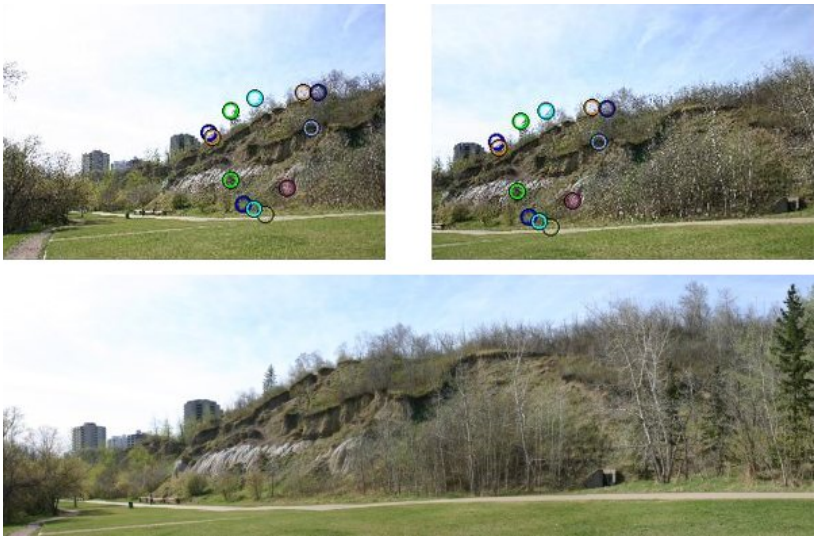

Figura 1.3: Imagen panorámica construida usando imágenes más pequeñas. Esta imagen ha sido creada usando Autopano (http://www.autopano.net/).

pueden dividir, simplificando, en dos grupos: tareas de bajo nivel y tareas de alto nivel.

En *bajo nivel* se engloban aquellas tareas que se realizan directamente sobre el valor de cada uno de los puntos de la imagen. Existen multitud de métodos de procesamiento para señales de una variable, en general para señales temporales. Estos se pueden extender de manera natural para procesar variables de dos dimensiones, como las imágenes. Transformadas de Fourier, correlación de señales, diferenciación, etc. son algunos ejemplos de este tipo de análisis. Otras técnicas usadas son mucho más específicas al uso de imágenes y no resulta tan fácil generalizarlas para otro tipo de señales. Un carácter distintivo para este tipo de métodos es que el proceso de creación de las imágenes (la proyección) implica relaciones no lineales que, junto con su carácter multidimensional, define un subconjunto en el campo de procesado de señal como parte de la visión por computador.

Un ejemplo en las tareas de bajo nivel es la búsqueda de correspondencias entre imágenes. Las correspondencias son puntos que pertenecen a la misma zona de la escena tomados por cada cámara. Esto se hace buscando correlaciones entre dos imágenes. Todo el análisis de bajo nivel de las imágenes se suele usar como parte

de un sistema más refinado. Las correspondencias se pueden usar, por ejemplo, para construir imágenes panorámicas con varias imágenes de menor tamaño. Captar este tipo de imágenes de manera directa implica la utilización de cámaras especiales con distancias focales cortas que permitan tener un gran campo de visión. Este tipo de cámaras son caras porque las ópticas que emplean son muy particulares. Además, este tipo de ópticas generalmente introducen una distorsión importante en la imagen resultante. Otra posible solución consiste en tomar varias fotografías con una cámara convencional cubriendo todo el campo visual sobre el que se desea construir la vista panorámica. Todas esas imágenes se pueden fundir en una sola buscando los puntos de correspondencia entre ellas y usándolos como guías a la hora de pegarlas todas juntas.

Conocer los puntos de correspondencia también permite averiguar las posiciones relativas desde las que se ha tomado cada imagen, lo que posibilita corregir las diferentes distorsiones introducidas en cada imagen con el fin de obtener un resultado más homogéneo (ver figura 1.3). Por último, se puede realizar un análisis estadístico de la luminosidad en cada imagen para que todas las zonas de la panorámica resultante muestren una iluminación uniforme.

Otra de las tareas de bajo nivel es la restauración de imágenes. La restauración intenta eliminar el ruido, suciedad o cualquier otra alteración de una imagen estática, secuencia de vídeo, que se haya visto degradada. El objetivo es devolver el aspecto original a los mismos. Los procesos ruidosos que normalmente se consideran en la restauración de imágenes son los producidos por las limitaciones de los sensores (por ejemplo, las cámaras o en las imágenes ultrasónicas), la distorsión por movimiento de la cámara (al realizar imágenes con poca luz) y la degradación por el paso del tiempo de negativos y positivos (por ejemplo al adaptar películas viejas a los nuevos soportes digitales).

En contraste con estas tareas de bajo nivel, más centradas en el análisis estadístico, agrupamos como tareas de *alto nivel* a aquellas que buscan encontrar algún tipo de significado dentro de los datos visuales. Esto abarca desde las relaciones geométricas en la proyección de la imagen hasta el descubrimiento de las pautas que encierran. Podemos dividir las tareas de alto nivel en otros tres grupos según qué es lo que buscan: tareas de reconocimiento, de reconstrucción y de localización.

Figura 1.4: Los sistemas de reconocimiento e identificación de caras usan una base de datos de caras para determinar la identidad de la persona de la imagen.

El reconocimiento es uno de los problemas más clásicos en la visión por computador. Se trata de determinar si una imagen contiene o no un objeto, característica o cualquier otro tipo de patrón. Los humanos somos capaces de realizar esta tarea prácticamente sin esfuerzo, pero aún no hemos conseguido resolverla de manera satisfactoria usando computadoras. Los métodos que existen para abordar este problema pueden, como mucho, resolverlo para un número limitado de objetos, ya sean formas geométricas u objetos más complejos como caras humanas o letras escritas y casi siempre en situaciones controladas o conocidas, descritas generalmente en términos de iluminación uniforme, fondo uniforme, orientación relativa a la cámara conocida, etc. Se suele hablar de este problema como de clasificación; clasificación de una imagen o de la figura en una imagen como perteneciente a un grupo o clase de entre varias posibilidades existentes. Una versión del mismo consiste en detectar si un objeto está o no presente en una imagen, si la imagen pertenece al grupo de imágenes con objeto o sin objeto. En este caso hablamos de detección. Por último, y para una única clase, se denomina identificación al proceso de determinar qué instancia de un objeto de una determinada clase se encuentra en una imagen.

Otro ejemplo típico de un sistema de reconocimiento es el que se usa en la iden-

tificación automática de matrículas de vehículos. Cámaras colocadas en las carreteras o en las entradas de los garajes graban a los vehículos que pasan. El sistema se encarga de localizar e identificar la matrícula en el vídeo y un módulo *OCR* (*Optical Character Recognition* o Reconocimiento Óptico de Caracteres) extrae el número de la matrícula que se usa para identificar el coche (ver figura 1.5). Una aplicación real de este sistema se puede encontrar en la ciudad de Londres, controlando el acceso a su zona centro. En esta ciudad existe una prohibición de circular con coche por la zona centro debido a los problemas de congestión y contaminación que sufría la ciudad en el pasado. Aún así se permite el acceso a vehículos que paguen una determinada tasa por cada día que quieran acceder a la zona centro. Para evitar fraudes es necesario controlar de alguna forma el acceso. El elevado número de vehículos que siguen circulando por la ciudad, incluyendo vehículos de uso público, hace imposible controlar el acceso de forma manual. La solución más fácil y extensible encontrada hasta el momento implica la utilización de cámaras en todos los accesos a la zona centro. Están encargadas de vigilar si los coches que pasan han pagado o no su tasa diaria. Para ello comprueban la matrícula del vehículo, contrastándola con una base de datos que almacena la contabilidad de los conductores que han abonado las tasas. De esta forma es posible detectar a los infractores y multarles automáticamente. Otro ejemplo, esta vez más cercano, se puede encontrar en los aparcamientos del aeropuerto de Barajas que, a menor escala, controla e identifica a los vehículos que acceden al mismo.

A parte del reconocimiento de objetos, otra tarea clásica es la reconstrucción de escenas. Ésta trata de cómo están distribuidas los objetos en una determinada escena. El objetivo principal de la reconstrucción es formar una representación de lo que estamos viendo, de dónde están colocadas todas las cosas y de cuál es la posición relativa entre ellas. Esta representación permitirá realizar análisis o transformaciones dentro de una computadora. La naturaleza exacta de la representación variará mucho según cuál sea el objetivo buscado. Cada una de estas posibles representaciones tiene requisitos diferentes, tanto en número de imágenes procesadas como en coste computacional, yendo desde mapas angulares creados con imágenes monoculares hasta reconstrucciones fotorrealistas detalladas. Representaciones aún más complejas pueden describir una escena como una combinación de objetos conocidos que se encuentran en ella.

Figura 1.5: Reconocimiento automático de matrículas. La matrícula se extrae de la primera imagen y posteriormente un OCR se encarga de convertir la matrícula a caracteres. Imagen extraída de http://www.platerecognition.info.

Similar a la reconstrucción encontramos los problemas de detección y localización, que comentaremos más profundamente en la sección 1.4. En esos casos estamos más interesados en la posición exacta de uno o varios objetos que en su colocación relativa de unos con respecto a otros.

1.3. Visión 3D

Cuando hablamos de visión 3D nos referirnos a métodos para conseguir información de las características tridimensionales de objetos o del entorno usando para ello cámaras. Las imágenes nos proporcionan únicamente información bidimensional, una proyección de la realidad. Dicha proyección implica una transformación no lineal que pierde una parte importante de la información. Las distorsiones que la perspectiva introduce en este proceso deben recuperarse para conseguir de nuevo la información tridimensional. Con las técnicas adecuadas y combinando varias imágenes se puede recuperar la información que se pierde con la proyección. La precisión de la información obtenida dependerá del número de imágenes usadas y, más importante aún, del método empleado. Con estas técnicas es posible obtener resultados con

Figura 1.6: Reconstrucción de modelo 3D fotorrealista. Usando varias cámaras, el espacio que rodea a la figura se va escarbando mientras se determina qué puntos 3D están libres y cuáles están ocupados y qué color tienen. La fotografía pertenece a [KS00].

calidad fotorrealista, muy diferentes a los modelos 3D a los que nos tienen acostumbrados los videojuegos o los programas de diseño asistido por ordenador (ver figura 1.6).

Una vez que estas distorsiones no lineales se han eliminado, trabajar con la información obtenida es mucho más fácil. Esto es lo que proporciona a la visión 3D su interés y, a la vez, su complejidad.

La visión 3D es un problema desafiante dentro de la visión por computador que ha levantado mucho interés en los últimos años. Definimos visión 3D como la capacidad de extraer información tridimensional de una o más imágenes bidimensionales. Las imágenes sólo proporcionan una parte de la información que existe en la realidad que observan. Ésta es tridimensional, así que toda la información 3D que seamos capaces de reunir estará más cerca del original que la información 2D.

La información obtenida incluye profundidad, posición y color. En imágenes 2D la primitiva principal se llama píxel (una abreviatura de *picture element* o elemento de

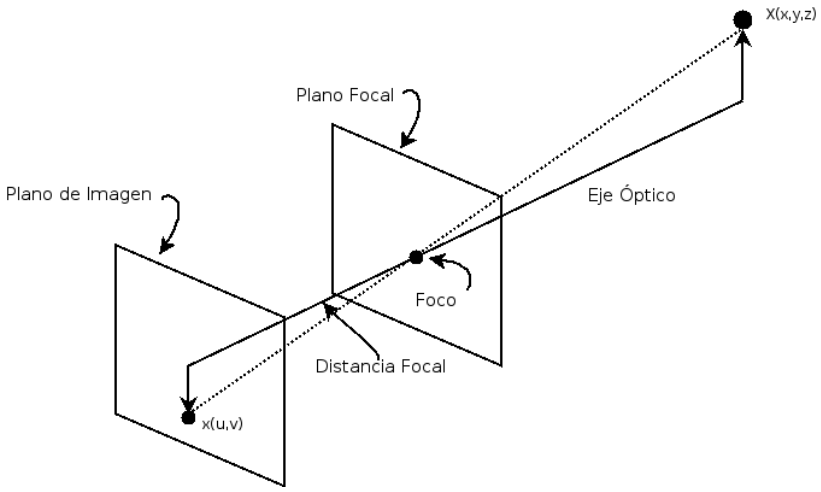

Figura 1.7: Modelo de cámara *Pin-Hole*.

imagen) y representa un punto en la imagen. En 3D existe una primitiva análoga que se llama *vóxel* (una contracción de las palabras volumen y píxel) que es un elemento volumétrico representando la posición dentro de una rejilla tridimensional en el espacio. Al igual que los píxeles, los vóxeles tienen asociado un color. Generalmente se pueden emplear en visualización y análisis de datos médicos o científicos, pero su nivel de información es tan bajo como el de un píxel. La información tridimensional de más alto nivel incluye primitivas como objetos geométricos (planos, segmentos, esferas, etc.) o modelos completos, en vez de simples puntos con color.

Aunque existen diferentes sensores que nos permiten obtener la información de profundidad de manera directa, como pueden ser los láser, las cámaras son mejores en varios sentidos. Tienen bastante precisión en color, una resolución angular muy buena, son muy robustas y no requieren ningún tipo de infraestructura adicional para funcionar. Pero su principal ventaja es sin duda su precio. Las cámaras son dispositivos baratos que son capaces de proporcionar una gran cantidad de información. Gracias a esto se pueden desarrollar aplicaciones que hasta ahora resultaban inimaginables.

Antes de continuar analizaremos el funcionamiento de las cámaras. Una cámara convencional sólo proporciona información 2D, esto es, el color o la intensidad de luz, en una determinada dirección. Esta información se almacena como un píxel, un

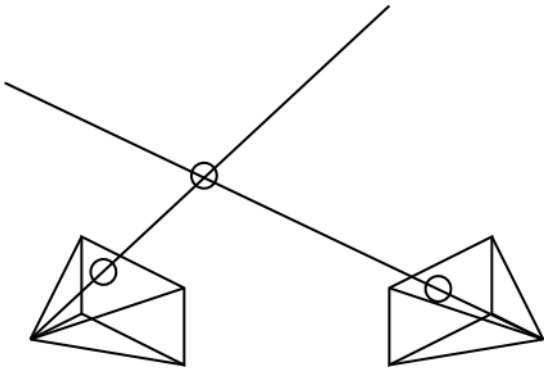

Figura 1.8: Triangulación de un objeto usando dos imágenes.

punto en el plano de la imagen. Desde este punto lo único que podemos saber es la dirección del punto original, pero no la posición exacta. El punto real estará a un distancia sin determinar sobre la línea formada por el punto en el plano de imagen y el *foco de la cámara*. Por este punto pasan todos los rayos que se proyectan en el plano de imagen. Esto puede verse en la figura 1.7 donde se muestra la relación entre los puntos reales y los puntos de la imagen.

Para definir la cámara se emplea su posición y orientación, los llamados *parámetros extrínsecos*, y una serie de características internas como la distancia focal, el tamaño de píxel, etc. que reciben el nombre de *parámetros intrínsecos*. Para terminar con nuestra descripción de cámara llamamos *línea de visión* al segmento que une el punto real con el punto proyectado. Esta línea, como hemos dicho, siempre pasará por el foco de la cámara.

Para conseguir reconstruir en 3D, o simplemente extraer información en 3D, se pueden relacionar estos segmentos o líneas de visión desde varias cámaras con los puntos reales en 3D. La forma más sencilla es triangular la posición 3D usando las líneas de visión del mismo objeto desde dos cámaras o desde dos imágenes tomadas desde una posición diferente. En la figura 1.8 se muestra un ejemplo de triangulación usando dos cámaras que ven el mismo objeto.

Las líneas de visión deben apuntar al mismo objeto, que debe localizarse en cada imagen. Es además necesario que, para objetos grandes con respecto a la imagen, ambas líneas de visión se dirijan a la misma zona del objeto. En caso contrario se producirán errores sensibles en la triangulación. A esta operación se la conoce como

emparejamiento entre las dos imágenes.

Existen multitud de técnicas para emparejar puntos o regiones entre diferentes imágenes. Las técnicas más generales siguen un procedimiento similar. En primer lugar se buscan los puntos de interés en una de las imágenes. Estos puntos suelen ser esquinas, bordes o cualquier zona de alto contraste. Generalmente es más sencillo encontrar un emparejamiento usando estas zonas porque resultan más llamativas que, por ejemplo, superficies de color uniforme. Una vez localizadas las regiones interesantes, se buscan zonas similares en la otra imagen.

Dependiendo del tamaño del patrón o del algoritmo usado para describirlo y buscar emparejamientos, el procedimiento será más o menos robusto, producirá más o menos falsos emparejamientos y, más importante, tendrá un coste computacional mayor. Las diferencias en las condiciones de iluminación, en el ángulo de visión de un mismo objeto, las rotaciones y cualquier otra distorsión, interfieren de manera significativa en el emparejamiento. Aunque existen métodos que proporcionan mayor robustez que otros (por ejemplo [SL04]), el procedimiento completo suele resultar bastante costoso.

Para mejorar las características del proceso de emparejamiento se pueden usar una serie de restricciones. La simplificación más sencilla, y la más usada, es la restricción epipolar. La recta epipolar parte de la línea de visión de un objeto desde una cámara. Dicha línea se proyecta en la otra cámara como una línea, conocida como recta epipolar. La recta epipolar cortará a la proyección del objeto en algún punto de la imagen (ver figura 1.8). De esta forma, conociendo la posición relativa de las cámaras es posible eliminar muchos de los falsos positivos que proporciona un método de emparejamiento no muy pesado. También es posible usar muchos puntos emparejados entre dos cámaras para calcular la posición relativa de ambas cámaras, por ejemplo para un sistema de autocalibración.

Aplicando este tipo de técnicas de manera densa podemos crear los conocidos como *mapas de disparidad* [BT99, MMHM02]. Los mapas de disparidad son imágenes espaciales donde el color de cada píxel representa la profundidad de dicho punto. Con estas imágenes resulta relativamente fácil navegar por un entorno dado, ya que es posible saber la distancia en cada una de las direcciones representadas en la imagen. También se pueden conseguir reconstrucciones de la escena captada por la imagen,

Figura 1.9: Mapa de disparidad. La imagen original se encuentra a la izquierda y el mapa de disparidad general está a la derecha. El color de la imagen de la derecha representa la profundidad de cada punto. La imagen pertenece a [BT99].

aunque sigue existiendo mucha información perdida en la escena. En la figura 1.9 puede verse un ejemplo de una imagen con un mapa de disparidad.

Una posible aplicación de técnicas como los mapas de disparidad es filtrar las imágenes en función de su profundidad. Esto puede ser útil, por ejemplo, en videoconferencias. Al usar una cámara en videoconferencia, nuestro entorno también queda grabado, lo que incluye el sitio en el que estamos, otra gente que puede encontrarse en la misma sala, etc. Tomando la información de profundidad podemos filtrar todos los puntos del fondo, es decir, aquellos que estén más lejos de un determinado umbral de la cámara. Con esto se puede crear fácilmente un velo invisible que oculta el fondo de la imagen de tal forma que el receptor de la videoconferencia sólo recibe un vídeo con su interlocutor, el primer plano de la imagen y un fondo neutro o incluso un fondo insertado para la videoconferencia. Esto aumentará nuestra intimidad al realizar la llamada.

La visión 3D es un interesante campo que expande las posibilidades de la visión por computador a un nuevo estado de integración con el mundo real y con la vida diaria. La información que nos puede resultar útil reside en un mundo 3D y tener computadoras capaces de ver directamente o de interactuar con esa información supone el primer paso para una nueva generación de aplicaciones. A pesar de que existen muchas posibles aplicaciones que pueden sacar partido de visión 3D, nosotros nos centraremos en un problema en particular, la localización y el seguimiento en 3D. Un lector interesado en cualquier otro tema relacionado con la visión 3D puede encontrar mucha más información en [HZ03].

1.4. Localización y seguimiento

En esta tesis abordamos el problema de la localización y el seguimiento visual en 3D. En la visión por computador localización implica determinar la posición instantánea de un objeto en una imagen o escena. Podemos hablar tanto de localización en 2D o 3D, en función del contexto. Casi todo el desarrollo es similar en ambos casos, pero nosotros nos centraremos exclusivamente en el seguimiento y la localización 3D.

En el fondo, determinar la posición de un objeto es un problema de estimación. Estimar la posición de un objeto en dos dimensiones implica localizarlo dentro de una imagen. Las coordenadas en este caso suelen ser relativas a la imagen, sin un marco de referencia externo. Por otro lado la localización en 3D implica la estimación de la posición usando un origen o marco de referencia. Determinar las coordenadas tridimensionales implica el conocimiento del marco de referencia o referenciar todos los puntos con respecto a la cámara. En ambos casos podemos estar interesados en sólo un punto, o en una región, podemos tener en cuenta su orientación o incluso su tamaño.

Podemos considerar más características en el proceso de estimación. Por ejemplo, podemos querer estimar la velocidad con la que se me mueve un objeto. Al incluir la velocidad, estaremos haciendo una estimación de 6 dimensiones, tres para describir la posición y tres para describir el vector de velocidad. Otras características pueden incluir la orientación, relaciones angulares o formas en objetos articulados. Es importante aclarar que cuando decimos que se estima la posición y la velocidad, la estimación de ambas se hace al mismo tiempo y no de manera separada. Cada uno será una dimensión más a ser estimada.

Para obtener información tridimensional, ya sea de posición, orientación o tamaño, será necesario emplear más de una cámara como comentamos anteriormente. Además las imágenes deberán estar sincronizadas (captadas en el mismo instante de tiempo) para poder relacionar la información proveniente de cada cámara.

El seguimiento es una tarea muy relacionada con la localización. Definimos seguimiento como la habilidad de localizar o estimar de manera continua la posición de un objeto. En vez de localizar un objeto en un instante de tiempo dado, lo que

se pretende es tenerlo localizado en todo momento, obteniendo la trayectoria del objeto. Aunque las técnicas empleadas son similares, para el seguimiento se utiliza la información sobre las últimas posiciones calculadas del objeto para tener una aproximación de hacia dónde se puede dirigir. Esto proporciona mucha información útil que facilita la estimación de la nueva posición porque permite aprovechar la continuidad espacio-temporal en el movimiento.

Cuando hablamos de secuencias de vídeo, localización y seguimiento normalmente se aplican juntas. En cambio en imágenes estáticas no tiene sentido hablar de continuidad o de seguimiento. El seguimiento implica una localización explícita y además permite emplear las estimaciones anteriores como base para las nuevas. Esto hace que, aunque nuestro objetivo sea únicamente la localización, realizar un seguimiento del objeto que deseamos localizar mejore los resultados con respecto a únicamente aplicar una estimación instantánea. El proceso de seguimiento podrá corregir y refinar continuamente la estimación con la llegada de nueva información.

La teoría detrás de la localización y el seguimiento suele emplear técnicas relacionadas con los métodos probabilísticos. Debido a la generalidad de estos métodos, existen multitud de campos que comparten técnicas similares. Uno de los ejemplos más obvios es la tecnología de los radares. Los radares utilizan la teoría de filtros adaptativos y otras técnicas de procesado de señales para localizar un objetivo empleando señales electromagnéticas.

Otros campos como la teoría de la información o las comunicaciones móviles también emplean el seguimiento con varias fuentes de señales diferentes, no únicamente imágenes.

Como ya se ha mencionado anteriormente, la localización y el seguimiento suponen un paso importante en las tareas de visión por computador. Estas técnicas resultan muy interesantes por sí mismas y, aún más como partes integrantes de sistemas más complejos. Resultan esenciales en muchas aplicaciones, como por ejemplo en la clasificación de objetos. En estos sistemas primero se localiza el objeto y luego, sobre esa zona de la imagen, se aplica el sistema de clasificación.

A continuación presentaremos varias aplicaciones que usan de manera intensiva técnicas de localización y seguimiento justificando con ello el interés de abordar este problema en particular.

Figura 1.10: Interfaz hombre-máquina GoMonkey usando visión. El usuario mueve sus brazos para controlar la interfaz de ventanas mostaradas.

El primer ejemplo que presentamos es un interfaz hombre-máquina que utiliza exclusivamente la visión. Mediante cámaras es capaz de detectar los movimientos del usuario, que se traducen en órdenes para el ordenador. Se trata del sistema comercial *GoMonkey*[2] (ver figura 1.10). Mientras el usuario mueve sus brazos y su cuerpo, GoMonkey detecta estos movimientos permitiendo seleccionar diferentes ventanas de una aplicación gráfica que se muestra en una gran pantalla. La principal ventaja de este método es que no precisa de ninguna infraestructura o hardware en especial y, aún más importante, de ningún entrenamiento para controlar el dispositivo. El usuario aproxima la mano hacia la ventana que desea acercar, hace un movimiento para agarrarla y la atrae hacia él. Resulta transparente y más natural que usar un ratón.

Otro ejemplo similar se usa en los videojuegos. La nueva generación de consolas permite una inmersión más profunda en mundos virtuales, siempre aumentando el grado de realismo. Eyetoy(TM) de Sony PlayStation 2(TM) (ver la figura 1.11) fue un producto pionero para el gran mercado que empleaba técnicas de visión por computador para introducir a los jugadores en la pantalla. Eyetoy utiliza una cámara conectada a la consola para seguir los movimientos de los jugadores. Estos movimientos se convierten en movimientos dentro del juego, con lo que se consigue una inmersión completa. Existen diversos juegos que emplean esta tecnología , relaciona-

[2]http://nooface.net/article.pl?sid=06/05/21/2249252, http://www.gomonkey.at/en/index2.htm

Figura 1.11: Eyetoy(TM) de Sony PlayStation 2(TM). La consola emplea una cámara para capturar el movimiento del jugador y trasladarlo al juego.

dos con los deportes, la lucha o la danza. Sin embargo, el principal límite del Eyetoy es la potencia de cómputo que tiene la PlayStation 2. La nueva generación de consolas más potentes, incluyendo la Xbox 360 de Microsoft y la PlayStation 3 de la propia Sony, soportan una serie de interesantes juegos y accesorios que incluyen realidad aumentada, fusionando figuras 3D con imágenes reales tomadas por una cámara.

Otra de las aplicaciones más populares del seguimiento visual se emplea en las películas y los videojuegos para crear efectos especiales, que de otra forma serían muy difíciles o completamente imposibles. Hablamos de los sistemas Vicon[3]. Los productos Vicon se centran en la captura de movimiento. Son capaces de seguir mediante cámaras una serie de puntos de interés o unos marcadores especiales colocados sobre una superficie (por ejemplo una persona). Varias cámaras se encargan de tomar imágenes desde muchas vistas para detectar estos marcadores y posteriormente un ordenador analiza todas las trayectorias extrayendo un esqueleto 3D animado del objeto. Cualquier movimiento que éste realice será capturado por el sistema. Éste fue el método empleado para crear el personaje de Gollum en la trilogía cinematográfica

[3]http://vicon.com/

Figura 1.12: Sistema de captura de movimiento Vicon. La imagen muestra a dos actores interpretando a personajes animados. El sistema Vicon sigue los movimientos de los actores y los traduce en movimientos de los personajes animados. Las tomas pertenecen a "Las Dos Torres" de New Line Cinema y a "Polar Express" de Castle Rock Entertainment.

del Señor de los Anillos (ver figura 1.12).

Sin embargo, ésta no es la única aplicación de la captura del movimiento. Otras aplicaciones incluyen la diagnosis médica, que analiza el esqueleto generado buscando algún tipo de enfermedad que se manifiesta produciendo anomalías en el movimiento. Esta misma idea se aplica en los entrenamientos deportivos de alto nivel. En este caso los movimientos del deportista se graban con el fin de analizarlos y conseguir perfeccionarlos hasta obtener un movimiento óptimo.

Para finalizar la revisión de estas aplicaciones de localización y seguimiento hablaremos de los sistemas de vigilancia, que son otra de las áreas donde la localización y el seguimiento tienen multitud de aplicaciones. Podemos hacernos una idea de la importancia del sector, considerando el número de cámaras de vigilancia instaladas en nuestras ciudades, que crece día a día. Sólo en la ciudad de Londres, una de las que cuenta con el mayor número, hay más de 500.000 cámaras instaladas. Esto supone que hay una cámara por cada 14 personas. Resulta obvio que con tal cantidad es imposible que operadores humanos las monitoricen de manera continua. Es nece-

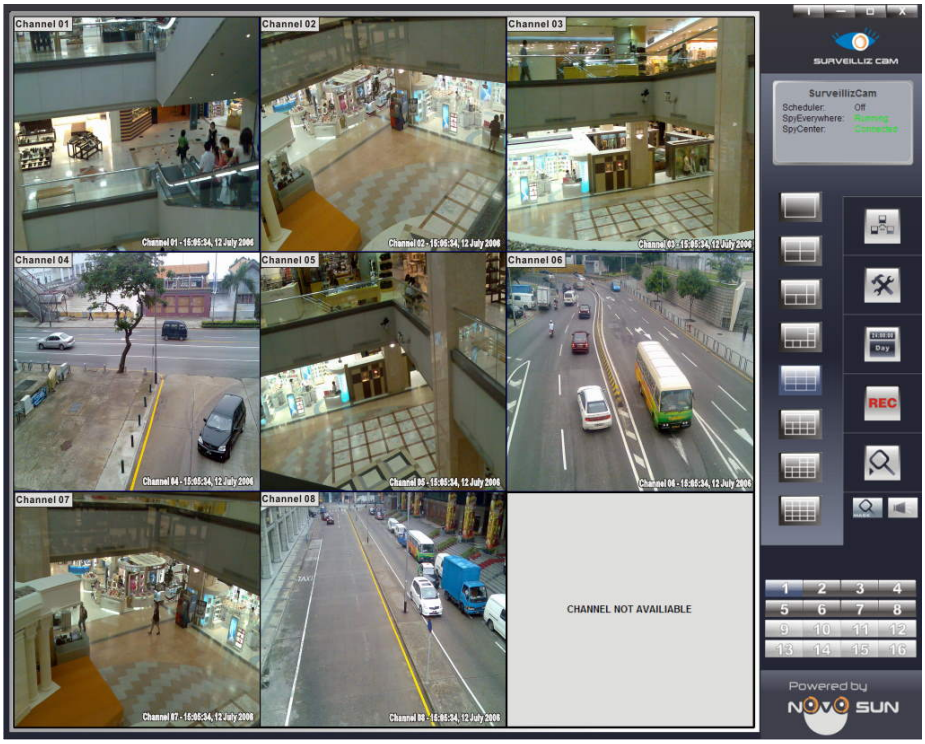

Figura 1.13: Aplicación de videovigilancia con varias imágenes. La imagen pertenece a uno de los productos de Novosun (http://www.novosun.com).

sario recurrir al procesamiento automático. Las técnicas de localización automática y seguimiento permiten a los ordenadores analizar todas esas imágenes buscando situaciones peligrosas, como puede ser una persona cruzando las vías del tren, contando personas, localizando intrusos en áreas restringidas o detectar asaltos. Sistemas más complejos se han mencionado anteriormente como el reconocimiento de personas o el reconocimiento de matrículas.

Dada su importancia, hemos seleccionado esta aplicación, la de vigilancia, como un campo de pruebas para los algoritmos desarrollados en esta tesis. Dentro de las aplicaciones de vigilancia nos centraremos en la localización de objetos o personas en el espacio, determinando su posición con respecto a un marco de referencia conocido. Dicha posición servirá para activar alarmas previamente configuradas. El escenario concreto se comentará en las secciones 4.7 y 5.5

1.5. Objetivos

Antes de adentrarnos en mayor profundidad en el desarrollo de la tesis es necesario presentar la objetivos de la misma. Como trata de reflejar el título, en esta tesis vamos a aplicar los métodos de Monte Carlo al problema de seguimiento visual en 3D. Este problema plantea algunas cuestiones interesantes que todavía no se encuentran resueltas, por lo que el estudio de la posible aplicación de los métodos de Monte Carlo resulta interesante en sí mismo.

Los métodos de Monte Carlo, revisados en profundidad en el capítulo 3, se han empleado en multitud de campos y con muchos objetivos diferentes. En nuestro caso, dado que estamos centrados en la visión 3D, abordaremos el problema de seguimiento visual. Por seguimiento entendemos la capacidad para estimar de forma continua las características que definen un objeto y su variación a lo largo del tiempo. La gran versatilidad de los métodos de Monte Carlo y sus excelentes resultados proporcionan el interés necesario para comprobar su funcionamiento en un entorno como el de visión 3D. Entre sus ventajas podemos mencionar su velocidad y su facilidad para adaptarse a gran cantidad de problemas diferentes. En este caso los algoritmos usados deben adaptarse para operar con funciones no lineales, como son las proyecciones visuales, y con los requisitos de tiempo impuestos por las aplicaciones en tiempo real.

A la hora de estudiar la posible aplicación de los métodos de Monte Carlo en este contexto, estamos interesados en la posibilidad de seguir *múltiples* objetivos al mismo tiempo. Un problema que suele plantearse con las implementaciones secuenciales de los métodos de Monte Carlo es la tendencia que tienen a representar únicamente distribuciones de probabilidad monomodales. En general, esto supone un inconveniente a la hora de abordar el problema del seguimiento multi-objeto. Esta tesis tiene como objetivo adentrarse en el estudio de sistemas que eviten esta tendencia perniciosa y proporcionar métodos robustos para realizar el seguimiento de múltiples objetos al mismo tiempo, usando técnicas de Monte Carlo.

Podemos abordar estos objetivos desde diferentes perspectivas. Esta tesis es una tesis de ingeniería por lo que, además de resolver los objetivos teóricos, buscamos conseguir una aplicación práctica. Por este motivo cuestiones como el coste de las

cámaras o la capacidad de cómputo deben tenerse en cuenta. Esto no se puede conseguir sin una vertiente experimental. En particular implementaremos todos los sistemas propuestos, lo que nos permitirá tanto realizar una serie de experimentos con los mismos como comprobar la viabilidad técnica del sistema propuesto. Un paso posterior consistiría en buscar una aplicación real dónde probar los métodos aquí propuestos, como pueden ser las aplicaciones de videovigilancia o de gestión operativa de procesos.

Los experimentos son esenciales para una tesis como ésta por lo que tendrán un gran peso dentro del desarrollo de la misma. Realizaremos dos tipos de experimentos bien diferenciados. Por un lado construiremos ejemplos estáticos que nos permitan comprobar la viabilidad del sistema en condiciones controladas, permitiéndonos medir variables tales como la velocidad de convergencia o la relación entre el error de medida y el ruido de observación. Por otro lado, el sistema se probará en condiciones reales, con imágenes tomadas en ambientes con luz variable y objetos en movimiento. Con estos dos tipos de experimentos esperamos caracterizar correctamente el sistema.

Requisitos

Además de los objetivos ya comentados, existen una serie de características deseables que nos autoimpondremos a la hora de abordar esta tesis, por su interés para el desarrollo posterior de aplicaciones. Estas características o restricciones no representan objetivos en sí, sino que marcan el camino a seguir a la hora de buscar dichos objetivos.

En primer lugar queremos desarrollar un sistema de seguimiento capaz de trabajar con la cantidad mínima de información *a priori*. No pretendemos estudiar cuál es la cantidad de información mínima necesaria para funcionar, ni emplear complicados sistemas que generen modelos sobre el comportamiento de los objetos. Nuestro objetivo es estudiar el comportamiento de los métodos de Monte Carlo por lo que usar el mínimo de información posible nos permitirá centrarnos en el desarrollo de mejores técnicas en vez de en la búsqueda de mejores modelos.

En nuestro caso supondremos que no sabemos nada acerca del patrón de movimiento del sistema. Aún así, podemos construir modelos de movimiento e interacción

genéricos, como suponer que los objetos se mueven de manera continua en el espacio. Estos modelos serán capaces de adaptarse a cualquier trayectoria y cambios en las dinámicas que rijan los movimientos.

Otra información necesaria *a priori* es algún método para saber qué es un objeto y qué no es un objeto. Para entrar en este punto debemos definir qué entendemos por objeto en el ámbito de este trabajo. Como objeto consideraremos cualquier entidad tridimensional que exista en el mundo real, comparta alguna característica distintiva y se mueva de manera compacta. Con compacta nos referimos a que todas las partes del objeto tendrán una alta correlación en sus posiciones espaciales. Esta suposición no entraña pérdida de generalidad para los objetos con los que podemos trabajar en la realidad. En caso de tener objetos más complejos será necesario abordar el problema de desarrollar modelos de movimiento *ad hoc*.

Podemos pensar en muchas y diferentes características distintivas. La primera y más sencilla de todas es el color. Un objeto puede ser una entidad del mundo real que tenga un color determinado. Esta simplificación nos permite centrarnos en el estudio del seguimiento de múltiples objetos de apariencia similar al mismo tiempo. Se puede ampliar la definición de objeto tanto como se quiera usando técnicas bien descritas en la literatura como pueden ser los histogramas de color, los autoespacios para describir objetos [BJ96, LRLY05] o las características invariantes a escala [SL04].

La precisión del sistema final es un requisito importante. Los métodos propuestos deben proporcionar sistemas suficientemente precisos o de lo contrario no resultarán útiles. Como precisión usaremos la medida del error en posición, no aceptando sistemas con errores relativos altos. Será necesario trabajar siempre con errores relativos dado que el error estará relacionado con el tamaño del objeto y la distancia. Esperamos lograr un error de estimación en posiciones tridimensionales menor a 10 cm al seguir una pelota de tenis en una zona de 150m^3.

Para conseguir esta precisión y estos resultados únicamente haremos uso de información visual. Todos los datos que usemos deben proceder de un conjunto de cámaras. Cualquier otro tipo de sensor, como pueden ser sónares o láseres, está totalmente descartado. Queremos emplear únicamente información visual por dos motivos. En primer lugar, las cámaras están cada día presentes en más lugares gracias a que su precio ha bajado continuamente a lo largo de los últimos años. Las cámaras

proporcionan mucha información por un precio muy ajustado, lo que las hace ideales como mecanismo de captura de información, siempre y cuando seamos capaces de procesarla. En segundo lugar, el uso de únicamente información visual es un problema todavía abierto para el seguimiento multiobjeto, aunque existe mucho trabajo en el área de visión por computador. El seguimiento visual es un campo desafiante en el que los métodos probabilísticos y el aumento de la potencia de los ordenadores están abriendo nuevas e interesante áreas de aplicación.

Partiremos de la utilización de al menos dos cámaras. De esta forma seremos capaces de extraer información tridimensional de manera directa. Para simplificar el problema a la hora de usarlas, supondremos que conocemos su posición y orientación en un marco de referencia común. Existen varios métodos de autocalibración que podrían emplearse para relajar este requisito. Sin embargo, no es el objetivo de esta tesis abordar este problema, por lo que suponemos que ya está resuelto. Un lector interesado puede consultar más información de calibración en cualquier libro o artículo de visión 3D [HZ03, MSKS04, HdARH99, dAHH99].

Así mismo consideramos que al igual que se conocen las posiciones y orientaciones de todas las cámaras, tenemos acceso a los parámetros de calibración de las mismas. Estos parámetros incluyen la distancia focal, el centro de la imagen y los parámetros de distorsión no lineal. Además de conocerlos suponemos que somos capaces de compensarlos. Con esto nos referimos a que somos capaces de relacionar un punto 3D del espacio con un píxel en cada cámara (proyección) y un píxel de una cámara con una línea del espacio. Las cámaras estarán fijas ya que no es un objetivo de esta tesis resolver el problema del movimiento de las cámaras.

En principio no ponemos ninguna restricción fuerte a la hora de colocar las cámaras, como podría ser el caso de un par estéreo. Aún así deben respetarse algunas mínimas restricciones. Por ejemplo, el objeto a estimar debe verse desde por lo menos dos cámaras o en caso contrario no podrá obtenerse información 3D real. Generalizando, toda la zona del espacio por la que se moverán los objetos debe ser cubierta, por lo menos, por dos cámaras. El campo de visión de las cámaras deberá estar, por lo tanto, solapado. Como ejemplo podemos pensar en que la posición de las cámaras debe ser tal que permita una correcta triangulación del objeto, si así fuera necesario. En el caso de que el objeto no se vea desde dos cámaras o más, por ejemplo debido a

una obstrucción, el sistema debe proporcionar la mejor estimación posible.

Todas las cámaras involucradas se tratarán de la misma forma. Otra alternativa, usada muchas veces en técnicas de visión estéreo, consiste en emplear una imagen como base y usar el resto para apoyar y añadir información a las estimaciones proporcionadas por la imagen principal. En nuestro caso, todas las imágenes tomadas se procesarán de la misma forma. Ninguna imagen será usada como referencia para nuestros cálculos o tendrá un procesamiento especial. Esta aproximación se conoce como un algoritmo verdaderamente multi-imagen (*"true multimage"*), tal y como se define en [Col95]. Debemos encontrar una forma de combinar todas las imágenes de una manera uniforme, lo que representa otro de los requisitos de la tesis.

Dentro del objetivo de seguir múltiples objetos al mismo tiempo y teniendo en cuenta nuestro requisito de minimizar la información *a priori*, supondremos que no conocemos cuántos objetos hay en la escena en un momento dado. Es más, supondremos que no sabemos cuál es el número máximo de objetos que puede aparecer en una determinada escena, hasta un límite prudencial (inferior a infinito). No presupondremos, como hacen otros métodos, ningún número fijo de objetos o número máximo. Los sistemas propuestos deben ser capaces de detectar la aparición de nuevos objetos y la desaparición de objetos antiguos sin perder los objetos seguidos. Muchos sistemas fallan a la hora de detectar nuevos objetos cuando sus capacidades de seguimiento están "enganchadas" a los objetos actuales. No son capaces de detectar la aparición o la sorpresa. Nuestro sistema debe detectar estos nuevos objetos tan rápido como sea posible y no debe perder ninguno debido a las oclusiones temporales. No obstante, no forzaremos al sistema a proporcionar un único identificador para cada objeto, de tal forma que si dos objetos se cruzan no es necesario distinguir sus trayectorias (esto podría hacerse en una capa superior con, por ejemplo, una red bayesiana, como en [NSC06]).

El sistema de seguimiento que vamos a usar debe funcionar en cualquier entorno en el que aparezcan varios objetos indistinguibles entre sí. El hecho de ser incapaces de distinguir dos objetos diferentes puede venir por dos motivos. Por un lado pueden tratarse de dos objetos realmente idénticos (por ejemplo dos pelotas iguales). Por otro lado puede darse el caso de seguir dos objetos diferentes pero que a través de nuestro modelo de observación simplificado se perciban de la misma forma. Esto puede

pasar si hay dos objetos del mismo color en una habitación y únicamente usamos información de color para seguirlos.

El hecho de que los objetos sean indistinguibles entre sí fuerza a realizar el seguimiento de todos los objetos al mismo tiempo. En caso contrario el problema podría abordarse separando los objetos entre sí y, posteriormente, realizando un seguimiento de cada uno de ellos de manera independiente. En nuestro caso buscamos un método que realice la estimación de las posiciones de todos los objetos al mismo tiempo, en un marco integrado.

El último de los requisitos que vamos a mencionar es quizá uno de los más importantes, el sistema debe trabajar en tiempo real. Cuando hablamos de tiempo real nos referimos a que el sistema debe funcionar lo suficiente rápido como para seguir de manera vivaz los movimientos en una escena. Esto elimina la posibilidad de emplear algún tipo de procesamiento en diferido, más allá de los asociados a la configuración y ajuste del sistema. La velocidad del sistema es difícil de medir por lo que lo haremos únicamente con el número de imágenes procesadas por segundo. En ningún caso deben ser menos de 30 por segundo, que en el caso de dos cámaras implicarían 15 por cámara. A parte debemos tener en cuenta el número de estimaciones de la posición, que debe ser, cuando menos, igual al número de imágenes capturadas por segundo. Estos resultados deben alcanzarse en un PC genérico de escritorio, sin ninguna característica especial, más allá de una cámara digital (por ejemplo con conexión Firewire). No usaremos ningún tipo de *hardware* dedicado, procesadores digitales de señal, ni ningún otro dispositivo especial.

Las imágenes se tomarán usando cámaras convencionales que podemos encontrar en cualquier tienda. En particular emplearemos cámaras de vídeoconferencia con una resolución variando entre 320x240 y 640x480 píxeles con una velocidad de entre 15 y 30 fotogramas por segundo. La cámara usará una conexión digital mediante USB 2.0 o puerto Firewire.

En la tabla 1.1 tenemos un breve resumen con los requisitos que debemos cumplir en esta tesis.

Requisito	**Descripción**
1	La posición y orientación de todas las cámaras se conoce.
2	El campo de visión de las cámaras se solapa sobre la zona de interés en la que se mueven los objetos.
3	Ninguna imagen será usada como referencia para nuestros cálculos.
4	El conocimiento *a priori* sobre la apariencia del objeto y su dinámica de movimiento debe ser mínimo.
5	No sabemos cuál es el número de objetos presentes en la escena en un momento dado. Tampoco conocemos el número máximo de objetos que pueden aparecer.
6	Los nuevos objetos se deben detectar sin perder los objetos ya conocidos.
7	El sistema debe funcionar en tiempo real.

Cuadro 1.1: Resumen de requisitos.

Hipótesis

A la hora de solucionar el problema que hemos presentado partiremos también de una serie de hipótesis. En nuestro caso podemos diferenciar dos conjuntos de hipótesis diferentes. En primer lugar haremos una serie asunciones acerca del entorno y del tipo de objetos que vamos a intentar seguir. El segundo grupo está relacionado con decisiones de diseño que iremos haciendo a lo largo de este trabajo.

Para extraer información de las imágenes podemos emplear diferentes técnicas de procesamiento. Otro de los objetivos de la tesis es definir dichas técnicas. Partimos del convencimiento de que es posible emplear sistemas de procesamiento relativamente simples que, combinados, pueden usarse para seguir cualquier objeto. Nos referimos como técnicas de procesamiento sencillas, por ejemplo, a filtros de color localizados en zonas de la imagen. Queremos comprobar que con métodos relativamente simples, como comprobar el color en un área dada de la imagen, se puede seguir sin mayor problema a una persona, pelota o cualquier otro objeto.

Otra hipótesis es que se puede seguir un objeto teniendo muy poca información *a priori* sobre él. De hecho la única información que consideramos es su color, suficientemente relevante como característica visual. La mezcla del color y el propio movimiento resulta bastante discriminante en condiciones normales. Ésta es la hipó-

tesis de la que partimos a la hora de diseñar los modelos de movimiento y los modelos de observación empleados tanto el capítulo 4 como en el 5.

En un caso sencillo tan sólo será necesario buscar el color de un objeto para ser capaz de localizarlo y seguirlo. Si esto no resulta suficientemente discriminante, el sistema tendrá múltiples falsos positivos. Al añadir el movimiento como segunda característica a buscar, estos falsos positivos irán desapareciendo. Esto se basa en el hecho de que cada uno de los objetos que sí se detectaban con el color no se mueven de la misma forma.

Así mismo, se podrán añadir cuantos estímulos diferentes sean necesarios, para hacer el sistema todo lo discriminante que se requiera. Consideramos que esta estructura de estímulos sencillos aditivos es más eficiente que desarrollar un modelo visual más complejo. El coste computacional y de desarrollo de modelos complejos tenderá a ser superior que el coste asociado a estos sencillos estímulos. Aún con esta diferencia de costes, la capacidad de discriminación será comparable. Podemos pensar, por ejemplo, en un modelo de persona para su seguimiento. Detectar una persona es una tarea relativamente compleja. Por otro lado podemos, simplemente, combinar información sobre movimiento, color y tamaño para ser bastante discriminante a la hora de seguir a una persona.

1.6. Estructura del documento

En este capítulo introductorio hemos presentado tanto el campo de la visión por computador como algunas de sus aplicaciones más relevantes o conocidas. Hemos pasado desde la visión bidimensional a la visión tridimensional, con las mejoras que ésta supone y los desafíos que presenta. Dentro de las posibles aplicaciones de la visión 3D, nos hemos centrado en una en concreto: las técnicas de localización y seguimiento. A continuación hemos enumerado los objetivos, requisitos y las hipótesis de partida de este trabajo.

En resto del documento está divido como sigue. El capítulo 2 hace un barrido por distintos trabajos de las áreas de la visión por computador y de la estimación y el seguimiento. Esta revisión tiene como fin presentar los trabajos en los que está basada esta tesis así como aproximaciones alternativas con objetivos similares.

El capítulo 3 se centra en las bases teóricas de las diferentes técnicas de estimación bayesiana que forman la base de los métodos propuestos en este trabajo. Partiendo del marco genérico de la estimación bayesiana se centra en los métodos muestreados, con énfasis en las técnicas de Monte Carlo. Se presentan diferentes métodos basados en estas técnicas, que tienen como base común el uso de números aleatorios o pseudoaleatorios. Entre los métodos presentados se incluyen, por ejemplo, el filtro de condensación y el filtro de partículas genérico.

Una vez que se han mostrado todos los conceptos relevantes sobre los métodos de Monte Carlo y la estimación secuencial bayesiana, se muestra cómo adaptarlas a nuestro problema en particular. En primer lugar, en el capítulo 4, se adaptan las técnicas para el seguimiento de un único objeto, mostrando sus limitaciones con una serie de experimentos. Así mismo se indica de manera teórica de dónde vienen dichas limitaciones.

El capítulo 5 generaliza el sistema de estimación para varios objetos, solucionando las limitaciones comentadas anteriormente. Para terminar, en dicho capítulo se muestran una serie de experimentos que avalan los métodos propuestos. Estos dos capítulos contienen las principales aportaciones de esta tesis.

El capítulo 6 describe la implementación de los algoritmos descritos en el resto del texto. Se hará especial hincapié en la problemática particular de la programación y tratará temas como la optimización o los espacios de color usados.

En el capítulo 7 están resumidas las conclusiones encontradas a la hora de realizar esta tesis, contrastando la propuesta con otros métodos alternativos.

Finalmente, en el apéndice A se añade una serie de información útil que por razones de brevedad y concreción no se han incluido en otros capítulos, como son unas notas sobre las relaciones de proyección y retroproyección o un repaso a las bibliotecas empleadas en la implementación de los algoritmos.

CAPÍTULO 2

Revisión Del Estado Del Arte

Para completar el contexto presentado en el capítulo anterior, a continuación realizaremos una revisión de la bibliografía relevante en las técnicas de seguimiento y localización visual en 3D. Este problema ha sido abordado con diferentes técnicas por parte de la comunidad científica, tanto para el caso de la visión 3D como para problemas similares, como veremos a continuación.

Nos centraremos en analizar diferentes técnicas para trabajar de manera eficiente en el problema de la localización de objetos. Muchas de las técnicas existentes no funcionan correctamente a la hora de enfrentarse al seguimiento de múltiples objetos simultáneamente. Este problema ha sido ampliamente explorado por la comunidad científica proponiendo diferentes técnicas para abordarlo. Así mismo podemos encontrar problemas que presentan unas limitaciones similares y las soluciones propuestas en esos casos. Mostraremos las más relevantes, estructurándolas en diferentes familias según las soluciones que empleen, y aquellas que han servido como base o inspiración para el desarrollo de esta tesis.

2.1. Visión 3D

El campo de la visión 3D ha sido siempre un campo muy activo desde el comienzo de la visión por computador. De hecho para muchos, éste ha sido considerado como el problema principal de la visión por computador [MN78, MP79]. El número de trabajos y propuestas es ingente en este área. A continuación pretendemos esbozar una pequeña muestra representativa de algunos trabajos principales.

Uno de los problemas de la visión 3D es el conocido como "reconstrucción visual". Partiendo de varias imágenes se intenta obtener una representanción 3D de la escena, ya sea localizando superficies, o bien puntos representativos en el espacio 3D real. Normalmente se suele abordar el problema en un escenario estático, por lo que la noción de seguimiento no tiene cabida, pero sí la localización 3D.

Las imágenes planas no proporcionan toda la información de la escena, en cuanto a posición tridimensional se refiere. La proyección de los objetos en las imágenes elimina la información de profundidad, por lo que será necesario algún proceso para reconstruir esta información perdida.

La aproximación más simplista y clásica para reconstruir esta información es emplear dos cámaras y alguna técnica basada en triangulación [HS97, Cre02]. En este caso, la forma de proceder consiste en localizar puntos interesantes en cada imagen y en encontrar correspondencia entre ellos, lo que permite obtener su profundidad triangulando. Es posible incluso emplear técnicas clásicas para localizar y seguir los puntos interesantes dentro de las imágenes planas, como [LK81, ST94].

Los resultados de esta aproximación clásica son muy interesantes como muestra el uso de los mapas de disparidad [BT99, MMHM02]. Estos mapas construyen una imagen en la que el color de cada punto representa la profundidad a la que se encuentra. Con esta información es sencillo realizar una reconstrucción de la escena que estamos viendo e incluso navegar usando esta información. Desgraciadamente, las aproximaciones basadas en mapas de disparidad no son lo suficientemente generales y potentes como para solucionar el problema de reconstrucción.

Aunque a primera vista pueda parecer que esta solución es sencilla, únicamente resuelve uno de los problemas planteados: la obtención de la posición 3D. Deja abiertos muchos otros problemas, como son las oclusiones o los fallos de emparejamiento,

que pueden dar lugar a estimaciones incorrectas. Pero sin duda, la mayor desventaja de estas aproximaciones es la dificultad para obtener más información que la posición 3D. Por ejemplo, si estamos buscando la posición y la orientación de un objeto articulado (como puede ser un brazo) tendremos más de tres variables que estimar, por lo que una simple triangulación no servirá para resolver el problema. Será necesario realizar más procesamiento para obtener toda la información sobre el objeto a seguir, lo que suele implicar realizar una reconstrucción de la escena.

Una aproximación más completa para la reconstrucción 3D, usando las mismas bases, se conoce como *Voxel Coloring* [SD99, CM99]. El objetivo de esta técnica consiste en obtener una rejilla densamente poblada que represente el objeto que se está viendo desde diferentes puntos de vista. Se parte de una rejilla 3D densa compuesta de vóxeles y, paulatinamente, se procede a eliminar aquéllos que no son compatibles con la información de color tomada desde cada imagen. Proyectando el vóxel en cada una de las imágenes se puede comprobar si presenta colores compatibles, lo que es indicio de su existencia en la realidad. Para evitar la correspondencia entre puntos se ajusta el orden de comprobación de los vóxeles, que ha de ser desde fuera hacia dentro. Esta restricción limita las posibles colocaciones de las cámaras, aunque es suficientemente genérico. Existen mejoras sobre este método, como son el *space carving* [KS00], que presenta un marco más preciso para combinar la información de cada cámara, y el *embedded voxel coloring* [LAS03], que obtiene mayor calidad al no eliminar los vóxeles vacíos hasta el final del procesamiento.

En una línea similar a la presentada por los algoritmos de *voxel coloring* está el *space-sweep* [Col95]. En este caso también se busca obtener una rejilla de ocupación, pero no con las características fotorrealistas que acompañan al *voxel coloring*. En este caso la construcción de la rejilla no se hace escarbando sobre una rejilla original, sino añadiendo información a una rejilla de probabilidad inicialmente vacía. Desde cada imagen se van tirando líneas de visión, aumentado la probabilidad de la rejilla en las zonas dónde dos líneas de diferentes cámaras se cruzan.

Una parte interesante de este algoritmo es que, a diferencia de los anteriores, emplea todas las imágenes de la misma forma. No existe una imagen de referencia de la que extraer patrones para buscarlos en las demás, sino que cada imagen vuelca su propia información sobre la rejilla. A este tipo de algoritmo se los define como

algoritmos “verdaderamente multi-imagen". Esta forma de abordar el problema tiene varias ventajas que la hacen interesante. Al no depender de una imagen en concreto es posible que el sistema sea menos sensible a oclusiones. Además, dado que las imágenes se tratan de la misma forma, se puede realizar una implementación en paralelo del algoritmo. Esta definición la usaremos a lo largo de la tesis puesto que exigimos que nuestros algoritmos funcionen de esta forma, evitando el uso de una imagen principal.

Una rejilla probabilística como la presentada en [Col95] puede extenderse para su uso en localización tridimensional usando únicamente visión. En esta dirección [MT98] presenta un sistema de localización 3D puramente visual usando una rejilla de probabilidad. El sistema tira líneas desde la cámara hasta un objeto cuyas características visuales ha aprendido previamente. La probabilidad de las celdas por las que pasa una línea de visión aumenta. Al ir moviendo la cámara a diferentes posiciones, la distribución de probabilidad representada por la rejilla tridimensional convergerá a una distribución monomodal sobre la localización exacta del objeto. Este sistema está pensado para robots móviles que se desplazan en entornos estáticos, pero no funcionará correctamente en entornos dinámicos dado que la información almacenada en la rejilla de probabilidad tiene cierta inercia.

Todos estos métodos basados en rejillas, ya sean de probabilidad o de vóxeles, tienen una restricción importante en común. Debe asegurarse que las imágenes estén tomadas sobre la misma escena. Esto se consigue únicamente si la escena es estática, no tiene cambios entre las diferentes capturas, o si las cámaras trabajan de manera sincronizada. En caso contrario no proporcionarán información compatible las unas con las otras. Aunque existen métodos capaces de trabajar con cámaras sin sincronizar, como el de [SFNT04], la aproximación más común consiste en emplear escenas estáticas o con pocos cambios. Esto no resulta un problema para modelar un objeto o extraer una representación tridimensional de un escenario, pero es una limitación importante para abordar problemas de seguimiento, como los que planteamos en esta tesis.

Otra característica común a este tipo de métodos es que la posición de las cámaras es conocida y referenciada a un marco común. Gracias a este marco es posible combinar la información obtenida por cada cámara. Si es necesario mover las cámaras, como en [MT98], la trayectoria debe ser conocida. Sistemas como [CAK02]

permiten mover las cámaras de forma activa para mejorar tanto el seguimiento como la reconstrucción. Sin embargo, la posición se conoce en todo momento.

En el caso más genérico de que no se conozca la posición de las cámaras es necesario abordar el problema desde un punto inicial diferente. Como primer paso será necesario estimar la posición de las cámaras. Existen multitud de trabajos que abordan la calibración de múltiples cámaras, ya sea en el problema de reconstrucción o en el de seguimiento. De hecho la autocalibración es un problema en sí mismo [Hem03]. Una vez estimada la posición el problema puede abordarse de la misma forma que el anterior.

Una aproximación existente que resulta muy interesante consiste en tratar al mismo tiempo los problemas de estimación de la posición de las cámaras y de reconstruir la escena. Este problema es conocido como *structure from motion*. Estos métodos enlazan de manera natural con los métodos de múltiples imágenes. Por ejemplo, [Dav03] usa una cámara móvil para extraer la posición de la cámara en todo momento e información acerca de la posición 3D de los objetos que está viendo. Este método hace uso de un filtro de Kalman para estimar el modelo de movimiento de la cámara y para corregir los puntos estimados pertenecientes a la escena. Otras aproximaciones emplean técnicas diferentes, como pueden ser los métodos de Monte Carlo [QC04, QCZ05] o los filtros de partículas [PC05, PC06], para obtener resultados similares. Métodos como estos últimos no intentan construir una representación densa de la escena, sino que construyen una representación liviana con unos cuantos puntos de interés. Dichos puntos se usarán para relacionar la información obtenida con su posición espacial o para interactuar con la formación de las imágenes en 3D. Los sistemas de realidad aumentada [KV98] hacen uso de estás técnicas.

Los métodos mencionados anteriormente parten de la base de que la estructura de la escena es fija, o por lo menos rígida. Esto permite utilizar puntos en el espacio real para calcular la posición de la cámara en cada instante de tiempo. Sin embargo, si esta estructura es dinámica o, incluso, no rígida no será posible emplearlos. Otras aproximaciones, como [dBdA06, dBdA07], son capaces de resolver este problema.

Técnicas como las presentadas en [Koc03] proporcionan una solución al problema de la reconstrucción densa sin calibración. Partiendo de una secuencia de vídeo tomada desde una cámara manual en movimiento, el sistema calcula la posición des-

de la que ha sido tomada cada imagen y la profundidad a la que se encuentra cada punto observado. Esto permite combinar las texturas tomadas desde cada posición para proporcionar una reconstrucción densa de la escena.

La característica común más importante de las técnicas basadas en rejillas que hemos comentado es su elevado coste computacional. Para obtener una representación completa de la escena es necesario hacer un barrido por todos los vóxeles. Si además se requiere precisión, esta rejilla será muy densa, agudizando aún más este problema. Otras aproximaciones emplean representaciones menos densas para trabajar de forma más eficaz. Por ejemplo, en [Fua97] se utilizan una serie de puntos 3D extraídos de imágenes estéreo como representación ligera de la escena. Una vez que se han extraído los puntos, se procede a aproximar por superficies. De esta forma se obtiene una representación compacta de la escena. Por otra parte, [Cre02] extrae los puntos 3D del emparejamiento de los puntos esquina entre ambas imágenes.

Usando mejor los recursos disponibles es posible reducir el coste manteniendo cierto grado de precisión. Dado que no todas las partes de la rejilla tendrán el mismo interés ni la misma cantidad de información es posible repartir mejor los recursos disponibles. En esta línea Louchet [Lou01] emplea un algoritmo genético combinado con una primitiva puntual, llamada mosca, para el problema de reconstrucción 3D visual. La colocación de las moscas en las zonas de interés permite optimizar el uso de los recursos disponibles.

2.2. Seguimiento de un objeto

A continuación nos centraremos en el problema del seguimiento de un objeto. En este caso estamos interesados no tanto en la reconstrucción completa de la escena como en la localización precisa de uno de los objetos para, a continuación, obtener su posición en todos los instantes de tiempo. El énfasis estará, a diferencia de los métodos vistos anteriormente, en el carácter dinámico del problema.

La manera que se ha demostrado más fiable para enfrentarse a este problema es emplear una aproximación probabilística. Para ello se representa la incertidumbre asociada a la posición del objeto mediante una función de densidad de probabilidad.

Aquellas áreas en las que tenemos más seguridad de que esté el objeto obtendrán valores más altos y viceversa.

Cualquier método capaz de estimar o de trabajar con dicha función de densidad de probabilidad nos ayudará a resolver el problema. En particular, las técnicas basadas en filtrado de Bayes se han empleado profusamente en el problema de localización y en el de seguimiento de uno o varios objetos [SBC99, Her02, NSC06, HE05, DDD04, Jen99].

Cada método emplea una forma de diferente para modelar las distribuciones de probabilidad involucradas. Los filtros de Kalman [Hay01], por ejemplo, modelan las funciones de distribución de probabilidad involucradas usando distribuciones gaussianas y relaciones lineales. Para este caso presentan la solución óptima, pero no pueden aplicarse para un caso más general. Los problemas en los que estamos interesados, la localización visual de múltiples objetos, son inherentemente multimodales por lo que un modelo gaussiano no es capaz de representarlos correctamente. Esto hace que los filtros de Kalman resulten de poca utilidad para nuestro objetivo.

Del mismo modo que los filtros de Kalman emplean un modelo gaussiano, otras técnicas emplean modelos analíticos para problemas particulares. La principal limitación a la que se enfrentarán es el desconocimiento que tenemos, en general, de la distribución de probabilidad, lo que hace difícil la obtención de un modelo analítico.

Por último, es posible emplear modelos no paramétricos para la descripción de la función de probabilidad. El planteamiento más extendido en este sentido es la utilización de representaciones muestreadas. La idea consiste en muestrear la distribución para conseguir un representación fiel. La aproximación más directa es realizar un muestreo uniforme, similar a las técnicas de rejilla anteriores. Al igual que en ese caso, el problema de estas técnicas uniformes es su alto coste computacional.

Otros métodos emplean las técnicas de Monte Carlo para obtener muestras representativas de una distribución de probabilidad. Estos métodos emplean números aleatorios, representativos estadísticamente, para reducir el número de muestras empleadas. La ventaja principal es que son capaces de reducir el coste computacional asociado a los métodos discretizados, manteniendo una tasa de fiabilidad independiente de las dimensiones del problema.

Quizá, el trabajo que más ha marcado el uso de los métodos probabilísticos

muestreados en la visión por computador ha sido el filtro de condensación [IB98a]. El objetivo de dicho filtro consiste en realizar seguimiento de formas en imágenes con muchos elementos de distracción. Para representar las formas a seguir emplea contornos activos. La adecuación de los parámetros de esos contornos es costosa, más aún en el entorno del seguimiento con ruido, donde es necesario descartar muchos contornos incorrectos. *Condensación* presenta un algoritmo probabilístico capaz de trabajar con múltiples contornos, siendo cada uno de ellos una hipótesis de la solución correcta. Esta forma de proceder permite reducir el coste computacional para encontrar la curva que mejor se adapte al objeto a seguir. Su aproximación consiste en evolucionar muchas curvas de manera independiente y que, de manera probabilística, alguna de ellas presente una solución adecuada. La implementación de condensación resulta muy sencilla lo que, junto a sus buenas prestaciones y su pocos requisitos, han convertido a este algoritmo en unos de los más exitosos para el seguimiento visual de objetos.

El filtro de condensación, o de manera más general los filtros de partículas, han sido empleados para resolver multitud de problemas de entre los que podemos destacar el de autolocalización de robots. Este es un problema formalmente idéntico al del seguimiento de un objeto y ha sido estudiado en profundidad desde la perspectiva probabilística. En este caso la distribución de probabilidad usada describe la posición de un robot o móvil dentro de un mapa. A medida que va añadiendo información sensorial, la estimación de su localización se irá actualizando.

Las aproximaciones tradicionales empleaban una rejilla de probabilidad para localizarse dentro de un escenario conocido. Los problemas de coste computacional hicieron que las soluciones fueran de manera paulatina hacia los métodos basados en filtros de partículas. Trabajos como [MSW02, GGB$^+$02, TFB00, TFBD01] nos muestran soluciones al problema de autolocalización usando filtros de partículas. Los problemas que aparecen a la hora de aplicar los filtros de partículas a la localización de robots son muy similares a los del seguimiento visual, de ahí nuestro interés en los mismos. Por ejemplo, [TFB00] muestra la problemática del mantenimiento de múltiples hipótesis, en este caso para la posición, y la importancia de la elección de una buena función de propuesta.

Las limitaciones encontradas en los filtros de condensación, o de manera gene-

ral en los filtros de partículas, suelen ser las mismas en cualquiera de estos ámbitos. La principal está relacionada con la degeneración de los pesos de las partículas a lo largo del tiempo, como se muestra en [Ber99, LC98]. Esta degeneración consiste en el aumento constante de la varianza de los pesos de las partículas. Esto implica que habrá partículas que no aportarán información a la estimación de la distribución objetivo, puesto que tienen asociados pesos muy bajos. Esta degeneración no puede evitarse, por lo que es necesario introducir alguna técnica de remuestreo que la reduzca cuando alcance un valor alto. Sin embargo, debe tenerse en cuenta que al aplicar el remuestreo también se pierde información sobre la distribución a estimar, lo que da lugar a un empobrecimiento de la población de partículas usadas, como se discute en [MMBS02]. Será necesario encontrar un compromiso entre las veces que aplicamos el remuestreo y el empobrecimiento de las partículas. Éste es un tema abierto, como indican los trabajos desarrollados tanto en la mejora de las *técnicas de remuestreo* [BDH03, DCM05, HSG06] como en la mejora de las *funciones de propuesta* [RC01, AFDJ03, OTF+04].

En particular, este último trabajo se centra en incorporar información sobre las observaciones actuales para mejorar la función de propuesta. Se suele aceptar que una función de propuesta que incluye información acerca de las observaciones recientes se comporta mejor en muchos casos que una función de transición genérica desde la población actual.

Aun con todas las ventajas en el rendimiento comentadas, los filtros de partículas, y el filtro de condensación en particular, no escalan correctamente con el número de dimensiones ni con el aumento del número de objetos a seguir, como veremos en la sección 2.3. Dada la naturaleza probabilística del muestreo, al aumentar el número de dimensiones hay cierta explosión dimensional. Dicha explosión no es comparable al caso del muestreo uniforme, pero aun así es perniciosa hasta el punto de convertir a los filtros de partículas en inútiles para altas dimensiones. Esto puede tener mucha importancia en, por ejemplo, el seguimiento de objetos articulados. Pensemos en el caso del movimiento de un cuerpo. Los modelos usados pueden llegar hasta 40 grados de libertad para describir la posición de todos los miembros del cuerpo. El filtro de partículas tradicional tiene poca eficacia al buscar en espacios de tan alta dimensión.

Por este motivo es necesario restringir la búsqueda para abordar estos problemas.

En [DBR00] se presenta el *annealed particle filter*, con el fin de introducir influencia entre los picos de la función de probabilidad. Este trabajo también se centra en la mortalidad de las partículas, que será de mayor interés si cabe en el seguimiento multiobjeto. Al eliminar una partícula de entre todas las partículas manejadas por el filtro estamos perdiendo información sobre una región del espacio de búsqueda. Si estamos en un problema de altas dimensiones repetir la misma región puede hacer al proceso muy lento y costoso.

Aparte del coste computacional, los filtros de partículas tienen otras ventajas por el hecho de emplear un marco probabilístico. Por ejemplo, [PVB04] presenta un método genérico para la combinación de diferentes fuentes de información de manera probabilística. De esta forma se pueden obtener mejores resultados al ser capaz de combinar información procedente de diferentes sensores. En particular, lo aplican para combinar información sonora con información visual, con el fin de realizar el seguimiento de hablantes en vídeo-conferencia. La información visual permite detectar a todas las personas presentes en la vídeo-conferencia y es la información sonora la que permite discriminar la dirección de la persona que habla, localizando únicamente a ésta. La ventaja del marco probabilístico es que permite una combinación limpia, sin interferencia entre los dos modelos de observación (dado que se suponen independientes). Otro ejemplo es el propuesto en [ZDD01]. Éste también fusiona diferentes fuentes de información empleando para ello un filtro de partículas y una red bayesiana. Usa modelos de observación basados en el sonido, que proporcionan una dirección de interés para corroborar la información obtenida a través de la visión. Gracias a esto es capaz de seguir a dos personas situadas frente a la cámara y focalizarse sobre la que está hablando. Este método lo emplea también en una aplicación de vídeoconferencia.

En la línea de combinar fuentes de información de diferentes sensores también es interesante el trabajo descrito en [KLF+02]. En este caso se emplean varios sensores de un robot móvil para obtener indicios parciales de la presencia de una persona. Cada uno de los sensores proporciona información parcial, como puede ser la localización de las piernas o la de la cara. El marco propuesto es capaz de funcionar aunque se pierda alguna de las observaciones necesarias. Por ejemplo, puede darse el caso de que una persona se encuentre detrás de una silla y no se vean sus piernas, pero si se

localiza la cara es posible continuar el proceso de seguimiento.

A parte del marco probabilítico bayesiano existen otras técnicas, que hemos mencionado anteriormente, que abordan el problema de manera similar. Los algoritmos genéticos, también basados en primitivas muestrales, pueden emplearse en éste problema. Por ejemplo, [PLLP04] aborda el problema de seguimiento e incluso [BMG07] se centra en la localización de un objeto manteniendo múltiples hipótesis con un algoritmo genético.

Sin embargo, los algoritmos genéticos no poseen unas bases teóricas tan sólidas y desarrolladas como los métodos de Monte Carlo, que se usan en los filtros de partículas. Por ejemplo, no es posible obtener garantías de convergencia como las que se dan en Monte Carlo. Por este motivo hemos descartado estas otras aproximaciones que podrían derivar en resultados igualmente válidos para el seguimiento.

También fuera del marco teórico de Monte Carlo y de la teoría de Bayes se encuentran las aproximaciones metaheurísticas. Las metaheurísticas son algoritmos aproximados que se han aplicado con éxito a los problemas de optimización. En particular, [Pan05] presenta un marco para la combinación de filtros de partículas y metaheurísticas que permite abordar con éxito el problema del seguimiento de uno o varios objetos.

2.3. Seguimiento de múltiples objetos

Si hay más de un objeto en la escena hablaremos de seguimiento multi-objeto. En éste tipo de seguimiento hay dos grandes familias de soluciones propuestas. Por un lado están aquellos sistemas que buscan objetos con características diferentes. En este caso, como los objetos tienen características distintas, producirán observaciones diferentes que podrán separarse de manera más o menos sencilla. Una vez separadas las observaciones, éstas se podrán usar como entrada a varios sistema de seguimiento de un único objeto, uno por cada objeto presente. Todos funcionarán en paralelo pero sin compartir información. Esto es posible si las observaciones son lo suficientemente diferentes de un objeto a otro. Por ejemplo, podemos seguir dos pelotas de colores distintos, un coche y una persona a la vez o dos caras identificando cada una como diferente.

Por otro lado, están los problemas de seguimiento de múltiples objetos indistinguibles unos de otros. En este caso no existe forma de separar las observaciones creadas por un objeto y las creadas por otro. Este problema presenta mayores desafíos, dado que toda la información disponible debe manejarse a la vez, creando de alguna forma hipótesis que expliquen toda las observaciones obtenidas hasta el momento.

Ejemplos de este problema pueden ser el seguimiento de varios coches al mismo tiempo, de las manos de una persona o el seguimiento de varias personas a la vez en un entorno concurrido [DT01, ZN04, SGPO05]. En este último caso, por ejemplo, lo que nos interesa es saber la posición de cada una de las personas que se mueven juntas por la escena. Éstas aparecen y desaparecen de manera continua, pasan cerca unas de otras, se esquivan y cambian su trayectoria. Aunque cada persona es diferente a los demás resulta complejo (y costoso) realizar un sistema capaz de seguir a cada persona como un objetivo distinguible de los demás. La forma común de abordar este problema consiste en interpretar cada persona como un objetivo indistinguible de los demás, por lo que el sistema se debe comportar como un verdadero sistema de seguimiento multiobjeto. Los problemas a resolver serán, entonces, la interacción entre las personas, los cambios de trayectoria o, incluso, la aparición y desaparición de personas.

Quizá el cambio más significativo al extender el seguimiento de un objeto al seguimiento de varios aparece al considerar múltiples soluciones igualmente válidas. En el caso del seguimiento de un objeto únicamente existe una solución válida, aunque de manera temporal pueda estar indeciso entre varias soluciones igualmente probables (hipótesis). Manejar varias hipótesis es una característica deseable de los sistemas de seguimiento y de los sistemas de autolocalización. El mantenimiento de múltiples hipótesis ha sido estudiado sobre todo en los sistemas de autolocalización para robots en entornos altamente simétricos, como por ejemplo edificios de oficinas o pasillos [MSW02, TFB00, TFBD01]. En esos casos lo importante es mantener las múltiples hipótesis el mayor tiempo posible hasta que una nueva observación permita desambiguar entre ellas.

A la hora de tratar con múltiples objetos estamos en una situación similar a la de mantener múltiples hipótesis de localización pero con una gran diferencia. En el

caso del seguimiento puede que todas las soluciones hipotéticas sean válidas (que representen cada una a un objeto real). Será necesario, entonces, no perder estas *posibles soluciones* y determinar cuáles de ellas son verdaderas soluciones. Incluso es posible que el número de soluciones válidas varíe con respecto al tiempo, al aparecer o desaparecer objetos.

Existen diferentes estrategias para lidiar con el problema de la existencia de múltiples soluciones. Podemos clasificar estás técnicas en varias familias según la forma en la que se enfrentan al problema. De esta manera distinguiremos las cuatro familias de algoritmos que detallamos en las siguientes secciones.

2.3.1. Incremento del espacio de estados

En primer lugar vamos a centrarnos en las soluciones que implican una modificación del espacio de estados. Una manera de representar de forma explícita la existencia de múltiples objetos es aumentar el tamaño del espacio de estados para que éste represente no sólo la posición de un único objeto, sino la colección de todas las posiciones de todos los objetos. De esta forma el estado $\boldsymbol{X}$ pasará a estar formado por $\boldsymbol{X} = \{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^n\}$, siendo n el número de objetos.

A este tipo de distribuciones de probabilidad se las conoce con el nombre genérico de distribución de probabilidad conjunta multiobjetivo o JMPD (del inglés *Joint Multitarget Probability Density*) [Kas97, KKH03, KHK04]. Al aumentar el espacio de estados se puede expresar la posición de cada uno de los objetos de manera desacoplada al resto, lo que fuerza a buscar la mejor solución para todos los objetos al mismo tiempo.

La ventaja de usar una distribución de probabilidad JMPD es que los algoritmos presentados anteriormente pueden funcionar sin grandes cambios. En este caso no existen múltiples soluciones válidas sino una única solución que aglutina todas las soluciones parciales. El proceso iterativo correcto irá perfilando la distribución hacia un único modo centrado en la solución global en un espacio de alta dimensionalidad.

Una de las limitaciones de este tipo de métodos es que el número de objetos a seguir debe conocerse *a priori*, o por lo menos se debe tener una cota superior. Sin embargo, técnicas como la presentada en [KKH03] permiten estimar tanto la posición

de los objetos, como el número de objetos presentes en la escena. Para ello trabaja con un espacio de estados de dimensión variable, permitiendo la entrada o desaparición de nuevos objetos en la descripción conjunta, formada por las dimensiones del espacio de estados.

El aumento del tamaño del espacio de estados conlleva la aparición de los problemas que mencionamos para el seguimiento de objetos articulados. Resumidamente, funcionan de manera más ineficiente según aumenta la dimensión del espacio de estados. Para reducir este problema [MB00] presenta un sistema capaz de reducir la complejidad asociada a la búsqueda en espacios de altas dimensiones, empleando una técnica de búsqueda jerárquica. Por ejemplo, si buscamos una estimación de tres dimensiones de dos objetos, tendremos un espacio de estados de seis dimensiones. El sistema propuesto se encargará de buscar primero en las tres primeras dimensiones asociadas a uno de los objetos y luego en las otras tres. De esta forma se consigue reducir la complejidad del sistema de búsqueda.

Otra importante limitación a la hora de usar JMPD se debe a la no interacción entre las partículas [OF02]. Cada partícula tiene información de la posición de todos los objetos de la escena. Puede que no todas las estimaciones sean correctas, es decir, una determinada partícula puede tener una estimación muy buena para la posición de un objeto pero estimaciones desastrosas para las posiciones del resto. Dado que la bondad de la partícula se evalúa teniendo en cuenta la solución global, es posible que ésta sea descartada en el proceso de evolución. Esto se debe a que las partículas no interactúan entre sí ni comparten información. La solución parcial para un objeto debería combinarse con soluciones parciales para otros objetos propuestas por otras partículas, lo que aumentaría la eficacia de esta aproximación.

En [KBD05] se proporciona otra solución para el caso de objetos que interactúan entre sí. Cuando dos objetos al acercarse modifican su modelo de movimiento, por ejemplo para no chocarse, las predicciones en las que se basa la estimación del movimiento no serán igualmente válidas. Para intentar acotar la complejidad, el sistema propuesto modela de manera explícita tanto el movimiento de los objetos como las interacciones entre los mismos, ajustando el modelo cuando éstas se producen. Así se consigue una simplificación del proceso de búsqueda, dado que las soluciones posibles están acotadas por las interacciones entre objetos.

El número de dimensiones de las partículas puede ser variable, como se presenta en [NLGP07]. Esto permite adaptarse a los cambios en el número de objetos. La solución presentada en este artículo emplea un modelo de movimiento basado den MRF (*Markov Random Fields*) que permite trabajar con trayectorias muy cercanas entre sí.

Para reducir el coste computacional es posible dirigir la búsqueda en el espacio de estados conjunto. En este sentido [MKK07] utiliza una función de propuesta para dirigir la búsqueda, con el fin de optimizar los recursos de búsqueda disponibles. Otras formas de reducir el coste computacional de la aproximación completa consiste en asociar las medidas con las trayectorias, como se hace en [Lee07], en una aproximación similar a las que veremos en la siguiente sección.

Aunque todas los trabajos que hemos presentado proporcionan soluciones aproximadas a este problema, es posible abordarlo de manera teórica. En [KF07] se presenta un marco teórico para conseguir una solución óptima al seguimiento con JMPD.

2.3.2. Segmentación de datos de entrada

Otra posible forma de enfrentarse al problema del seguimiento de múltiples objetos consiste en segmentar los datos de entrada. Si somos capaces de separar las entradas producidas por cada uno de los objetos a seguir podremos usarlas de manera aislada. Esto permite, de nuevo, usar varios sistemas de seguimiento de un único objeto, para cada uno de los objetos. Al usar técnicas conocidas es fácil configurar estos métodos y los resultados suelen ser satisfactorios. Además, el coste computacional crece de forma lineal con el número de objetos y no de forma exponencial, como sucedía con el uso de JMPD [KBD05].

El origen de estas técnicas es anterior a las demás aproximaciones que veremos. Esto se debe a que se trata de la aproximación más directa, usar varios sistemas en paralelo. Por ejemplo, [Rei79] ya usaba un sistema de segmentación de los datos de entrada para alimentar una serie de filtros de Kalman en paralelo. Al segmentar las entradas y alimentar sólo las entradas correspondientes a cada objeto tenemos varios filtros de Kalman funcionando en paralelo de manera independiente.

El principal problema de estas técnicas consiste en la asociación de los datos de

entrada con cada uno de los objetos a seguir. Dado que dichos objetos pueden ser indistinguibles entre sí, este problema resulta complejo. Por ejemplo, en [ORS04] se exponen técnicas capaces de asociar datos de entrada de forma probabilística para el problema del seguimiento multiobjeto, incluso capaces de estimar el número de objetos que aparecen en la escena. Para ello utiliza un sistema de Monte Carlo basado en cadenas de Markov (*Markov Chain Monte Carlo*).

La continuidad de las observaciones se puede emplear para mejorar la segmentación de los datos de entrada. En [NJS06] se muestra como la continuidad espacio-temporal de las observaciones permite una mejor separación de las mismas, lo que conduce a mantener durante más tiempo el seguimiento de los objetos.

Otro de los efectos perniciosos que pueden aparecer en estas técnicas, como se indica en [KBD05] y [MB00], es una convergencia de todos los módulos de seguimiento al mismo objeto de entrada. También cuando las trayectorias de dos objetos se cruzan, el sistema de segmentación puede no funcionar correctamente, mezclando la información de los sistemas de seguimiento. Esto puede hacer que más de uno sigan al mismo objeto, perdiendo la trayectoria de otros.

Para evitar los problemas con los cruces algunos sistemas se centran aprender, si es posibles, las características diferenciadoras de cada objeto. Por ejemplo, en [CdFL06] y en [YL06] se aprenden dinámicamente los colores particulares de cada objeto en movimiento con el fin de aislarlo del resto de los objetos presentes. Incluso en [DCSF06] separa por clases los objetos, con el fin de realizar la segmentación en diferentes pasos, uno por clase o tipo de objeto.

2.3.3. Interacción entre partículas

La siguiente familia de soluciones se enmarca dentro de marco estricto de los filtros de partículas. Hasta ahora hemos analizado el uso de las partículas como soluciones al problema del seguimiento. En el caso de los JMPD las partículas indicaban la solución completa a la posición de todos los objetos. Si aplicábamos una segmentación de los datos de entrada, cada uno de los filtros usados tendría sus propias partículas. Otra posibilidad consiste en describir sobre una única función de densidad todas las posibles soluciones. En ese caso la densidad de probabilidad será, necesaria-

mente, multimodal. Lo que buscamos no es un único punto en la función de densidad de probabilidad sino una o varias zonas de interés, zonas en las que estarán los objetos. De esta manera pasamos del problema de localizar una única partícula correcta a buscar "grupos" de partículas que reflejan los diferentes modos de la distribución de probabilidad. Cada grupo estará asociado a un objeto y será independiente del resto. La información que nos interesa estará, en este caso, en los grupos, no en las partículas. Éstas serán únicamente una herramienta para describir lo mejor posible la función de densidad de probabilidad y, por tanto, los grupos.

La principal dificultad con la que deben lidiar los trabajos de esta familia es la tendencia monomodal de los filtros de partículas. Cada una de las partículas se trata como una solución hipotética, independiente de las demás. Aquellas que parecen más verosímiles, a la luz de las observaciones recibidas, se mantienen por completo y aquellas que lo son menos se eliminan. Esto se realiza en un intento de optimizar los recursos existentes, priorizando las zonas prometedoras. Por ejemplo, [KKH03] ya menciona los problemas de los filtros clásicos, en los que algunos modos de la función de densidad de probabilidad pueden morir debido a que no hay suficientes partículas sobre ellos.

Muchos trabajos abordan este problema empleando algún método de interacción o repulsión entre partículas. La idea principal consiste en evitar que las partículas se centren únicamente en uno de los grupos, dejando al resto sin soporte y, por tanto, perdiendo objetos. Una vez que tenemos una zona con suficientes partículas, esto es, una zona ya explicada, no es necesario seguir añadiendo nuevas partículas. Con esto se consigue que las partículas compartan información de manera implícita.

En [TC02] se presenta un método capaz de seguir un número alto de objetos al mismo tiempo. Para ello presenta un modelo de observación que tiene en cuenta las oclusiones visuales producidas por unos objetos en otros. Dado que la posición real de los objetos no se conoce, se emplean las partículas como aproximación a las posiciones. De esta forma la existencia de una partícula influye en la existencia de las demás. Al colocar una partícula se elimina la información visual que dicha partícula proporciona, dejando únicamente aquella que la partícula no explica. Otras partículas serán las encargadas de dar significado, con nuevas hipótesis, a las observaciones visuales que queden. De esta forma es posible no centrar todas las partículas en una

única observación, lo que abre la posibilidad de seguir múltiples objetos al mismo tiempo.

En [MB00], como hemos comentado, se presenta un método para reducir el coste asociado a la búsqueda de objetos en espacios de estados de muchas dimensiones. Aunque trabaje en un espacio JMDP, también emplea técnicas de exclusión entre partículas. En este caso la exclusión no es exactamente entre las partículas, sino entre las partes de las partículas de cada subespacio. Cuando un objeto ha sido explicado por las primeras componentes de una partícula, asociadas a un objeto en un JMDP, las distribuciones de propuesta para las siguientes dimensiones se modifican atendiendo a estos valores. De esta forma la distribución usada tiene en cuenta únicamente las observaciones que siguen sin estar exploradas tras colocar las primeras componentes de las partículas. Vemos que se trata de un esquema similar al anterior.

Fuera del marco probabilístico de Monte Carlo existen otras técnicas que también emplean la interacción entre soluciones. En particular nos referimos a los algoritmos genéticos que hemos comentado anteriormente. Éstos no están limitados por un marco probabilístico estricto, por lo que es posible añadir operadores genéticos capaces de implementar interacción entre soluciones de manera directa. [LGLB02] presenta un operador de repulsión entre hipótesis para repartir los recursos disponibles sobre todo el espacio de estados.

2.3.4. Mezcla de distribuciones

Otra solución para abordar el problema anterior, la pérdida de multimodalidad de una distribución de probabilidad, consiste en centrarse en el sistema de elección de nuevas partículas. En lugar de dejar que unas influyan en otras, se modifican las técnicas de colocación de las nuevas partículas para conseguir un reparto algo más equitativo. De esta forma no hay una interacción directa entre las partículas, aunque sí la habrá indirecta.

En los filtros de partículas se coloca nuevas partículas en dos pasos: el remuestreo y la función de propuesta. La idea principal de este grupo de soluciones es ajustar estas dos funciones para que coloquen las partículas de una forma suficientemente distribuida entre todos los grupos existentes en el espacio de estados.

Vermaak [VDP03], por ejemplo, propone modelar la distribución objetivo como una mezcla no paramética de filtros de condensación. Este método recibe el nombre de *mixture of particle filters*. La distribución combinada tendrá de manera explícita un carácter multimodal, necesario para el seguimiento de múltiples objetos. La ventaja principal es que resulta posible realizar el paso de remuestreo a cada filtro de condensación de manera individual, asegurando un mínimo de partículas en cada grupo. Aunque cada filtro de condensación tenga características monomodales, la combinación de varios respetará la forma de la distribución objetivo.

Cada filtro de partículas individual en [VDP03] interactúa con los demás únicamente en el cálculo de los pesos de mezcla, lo que proporciona un algoritmo numéricamente eficiente y capaz de funcionar en paralelo. Las buenas características del algoritmo de condensación se mantienen, sin restricciones, en cada una de las componentes involucradas en la mezcla.

Debemos recalcar que aunque usemos varios filtros de partículas al mismo tiempo no existe ningún tipo de segmentación de los datos de entrada. Lo que sí tenemos es un sistema en dos capas, una a nivel de partículas y otra a nivel de componentes de la distribución objetivo. Será necesario algún método para asignar las partículas a estas componentes. Dicha separación se puede hacer con un algoritmo de segmentación cualquiera. Por ejemplo, [TC02] realiza esta segmentación empleando celdas de Voronoi sobre el espacio de estados. Una vez construidas las celdas, coloca un filtro de condensación por cada grupo obtenido.

Una limitación de este método es que no es capaz de gestionar la aparición y desaparición de objetos. Los filtros de partículas empleados son incapaces de localizar un nuevo objeto, a no ser que éste se encuentre en las inmediaciones de uno de los objetos que están siguiendo. Trabajos como el de Koller-Meier [KMA01] han tratado de resolver este problema. Su propuesta consiste en dividir el problema en dos partes difrenciadas: el seguimiento de objetos y el manejo de la aparición y desaparición de objetos. Para el seguimiento se emplea un sistema de combinación de filtros de partículas, cada uno de ellos siguiendo a un objeto diferente. Para gestionar la aparición se modifica la función de propuesta, que es la encargada de mover las partículas de una iteración a la siguiente. Koller-Meier propone emplear una función de propuesta formada por la combinación lineal de dos componentes. La primera moverá las par-

tículas siguiendo el modelo de movimiento, al igual que en el filtro de condensación. La segunda componente, llamada *densidad de inicialización*, será la encargada de colocar cierto número de partículas en nuevas zonas a la búsqueda de nuevos objetos. La densidad de inicialización empleará la secuencia de observaciones para colocar las nuevas partículas, aumentando la probabilidad de localizar los nuevos objetos tan pronto como aparezcan en éstas.

Simplificando, la función de propuesta puede verse como una modificación a la mezcla de filtros de partículas que divide las partículas en dos grupos. Tendremos un grupo de partículas que evolucionarán siguiendo la dinámica de los algoritmos de condensación. En cada iteración añadiremos un segundo grupo con un número fijo de partículas en las nuevas zonas extraídas directamente de las observaciones. Si estas nuevas partículas obtienen un peso significativo implicará que han localizado un nuevo objeto, que se seguirá con un nuevo algoritmo de condensación sobre él.

Similar a este método es el propuesto en [MSG+05], que se centra en mejorar el remuestreo de las partículas. En este caso no se hace el remuestreo de manera aislada en cada filtro de condensación, sino que se hace de manera conjunta entre todos. Para ajustar mejor las partículas usadas por cada uno de los filtros, se hace una segmentación de la población de partículas total. Una vez obtenidos los grupos, las muestras se pueden recolocar haciendo un reparto más uniforme entre los objetos.

[OTF+04], por su parte, añade al sistema la capacidad de adaptar los modelos de observación según van apareciendo nuevos objetos en la escena. En su caso lo utilizan para seguir a los jugadores de un partido de hockey. Tan pronto como se detecta un jugador nuevo, se aprenden sus características visuales y se añade un algoritmo de condensación centrado en él. Para ello emplea una función de propuesta compuesta por la combinación lineal de un algoritmo *adaboost* como modelo de observación y la dinámica propia de cada objeto.

En esta última familia es dónde podemos enmarcar el método propuesto en esta tesis. Nos centraremos concretamente en modificar la función de propuesta, haciendo énfasis en la distribución de las partículas para evitar perder objetos, pero sin modificar la función de remuestreo. Esto lo hacemos desacoplando totalmente los dos niveles del algoritmo, el de estimación de la distribución de probabilidad y el de segmentación de los grupos.

CAPÍTULO 3

Fundamentos De Los Filtros De Partículas

En capítulos anteriores hemos mostrado nuestra intención de abordar el problema del seguimiento visual 3D que puede estudiarse utilizando técnicas de filtrado recursivo bayesiano. Dicha teoría es capaz de proporcionar las herramientas necesarias para asegurar técnicas que funcionen de manera óptima usando la información de la que disponemos. En este capítulo pretendemos hacer un estudio de las técnicas más generales de estimación dinámica (localización y seguimiento) dentro del marco probabilístico, recorriendo las soluciones óptimas y las aproximaciones más usadas, abordando tanto los fundamentos básicos del filtrado bayesiano como las aproximaciones numéricas a sus soluciones.

Para ello comenzaremos exponiendo las bases del filtrado bayesiano y de su forma recursiva. Dado que esta solución, en la mayoría de los casos, resulta muy costosa de alcanzar o imposible, mostraremos una de las aproximaciones más usadas hoy en día, que consiste en el uso de integración numérica para solucionar el problema. En particular emplearemos métodos de Monte Carlo. Estos han demostrado una gran eficacia reduciendo el coste computacional de problemas complejos, a la vez que proporcionan cotas de error a las aproximaciones dadas. Los métodos secuenciales

de Monte Carlo se emplean como base de muchas técnicas usadas a día de hoy para la estimación bayesiana. Mostraremos algunas de estas técnicas y comentaremos sus limitaciones y ventajas. En particular, haremos especial hincapié en los filtros de partículas, que serán usados, con diversas variantes, en los capítulos siguientes.

En dichos capítulos usaremos estas técnicas generales para aplicarlas al problema concreto del seguimiento visual. Para ello será necesario adaptar los diferentes modelos que aquí presentamos a las particularidades de las señales visuales y de los sistemas de proyección óptica. El objetivo es mostrar las técnicas tal y como son, para comprender sus fundamentos y matices. Esto nos permitirá abordar mejor los problemas planteados a la hora de adaptar estos métodos al seguimiento visual.

3.1. Seguimiento bayesiano

Al hablar de seguimiento nos referimos a la habilidad de estimar continuamente la posición de un objeto. En un entorno más general podemos hablar de estimar algún valor que cambia a lo largo del tiempo, y ser capaces de mantener dicha estimación. Como ejemplos podemos citar el seguimiento de aviones usando radares o el seguimiento de la señal de un teléfono móvil para adaptar los receptores a las condiciones de la misma.

Usar un marco bayesiano para este tipo de problemas tiene varias ventajas. En particular las más relevantes para el contexto en el que nos encontramos son las siguientes:

- Permite estimaciones recursivas, lo que facilita tanto la implementación como la eficiencia.

- Este marco es capaz de proporcionar cotas de error probabilístico. Estas cotas nos aseguran el correcto funcionamiento del sistema, una vez que cumplamos una serie de requisitos previos. Sin embargo, estas cotas son probabilísticas, por lo que únicamente indican que el sistema funcionará correctamente en media. No hay garantías para una ejecución en concreto.

- Existen aproximaciones numéricas que reducen en gran medida el coste computacional. Gracias a esto es posible encontrar soluciones óptimas en cuanto a coste computacional y velocidad para muchos problemas.

- El marco probabilístico de Bayes se sustenta en una sólida base matemática. Ésta le proporciona una gran robustez, de la que podemos beneficiarnos en nuevos desarrollos.

En general, los valores que se intentan seguir no se pueden medir de forma directa o, en caso de poder hacerlo, se obtienen altos niveles de ruido. El seguimiento será el encargado de proporcionar una estimación precisa del estado real del objetivo. Para ello se utilizarán otras variables, relacionadas estadísticamente con el valor que no podemos medir, con el fin de extraer la información necesaria del sistema. A estas variables se las llama observaciones porque sí que pueden obtenerse (u observarse) de manera directa.

A pesar de que únicamente estamos interesados en aplicaciones visuales, las técnicas de seguimiento se han estudiado y aplicado en un rango muy amplio de problemas, empleando un conjunto de técnicas muy diversas. Estas técnicas incluyen filtros adaptados, filtros de Wiener, filtros de Kalman [Hay01], filtros de partículas [AMGC02] y redes bayesianas [NSC06]. También se han empleado otras técnicas que no están basadas en teoría probabilística, como pueden ser las técnicas metaheurísticas o los algoritmos genéticos, por citar algunos. Estos métodos, aunque han resultado muy útiles en algunos ámbitos, no presentan unas bases teóricas tan sólidas como las que han demostrado las aproximaciones probabilísticas.

La principal formulación del problema de seguimiento o estimación usando métodos probabilísticos es el filtro bayesiano y su extensión recursiva. En esta sección presentaremos el marco general para describir dicho filtrado Bayesiano no lineal y sus raíces. Nos centraremos en los conceptos generales usando variables aleatorias y funciones de densidad de probabilidad.

Muchos problemas en el procesado estadístico de señales pueden enunciarse como conocer el *estado* de un sistema dado un conjunto de medidas que están relacionadas estadísticamente con él. Llamamos *estado* al vector que contiene toda la información relevante requerida para describir completamente el sistema bajo inves-

tigación. En problemas de seguimiento esta información puede incluir posiciones o características cinemáticas del objetivo, como la velocidad o la aceleración. En otros tipos de problemas los valores necesarios para describir el problema variarán. En economía, por ejemplo, podemos hablar de flujo monetario, tipos de interés, inflación, etc. Sean cuales sean los valores representados, en el vector de estados se tienen todas las variables necesarias para describir la situación real actual.

El espacio de todos los posibles estados se llama *espacio de estados* y, en términos generales, el estado puede cambiar a lo largo del tiempo de un estado a otro. Por ahora definiremos el vector multidimensional $\boldsymbol{x}_k$ como el estado del sistema en el instante k, con $k = 1, 2, ...$ Los instantes de tiempo considerados son pasos discretos donde el subíndice k hace referencia al instante $t_k = k\Delta t$, donde Δt es un intervalo fijo de tiempo.

Empleando una notación similar a la usada en [AMGC02], la secuencia completa de estados desde el principio hasta el estado actual en k se representa por $\boldsymbol{x}_{1:k}$, que se refiere al conjunto $(\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_k)$. Generalmente podemos considerar que $\boldsymbol{x}_k$ es una variable aleatoria debido tanto a la presencia de ruido en el proceso de evolución, como a la incertidumbre del proceso de extracción, esto es, de la obtención de dicha variable. En resumen, no podemos conocer de forma directa y exacta esta variable, por lo que se asume que la secuencia de estados es desconocida. Toda la información sobre la misma se obtiene a través de un proceso de observación.

Por otro lado, definimos $\boldsymbol{z}_k$ como la medida tomada sobre el sistema en el instante k y, empleando la misma notación que antes, $\boldsymbol{z}_{1:k}$ representa la secuencia de medidas hasta el instante k. Estas medidas también se considerarán como variables aleatorias. Esto es debido, en primer lugar, a que dependen de una variable aleatoria (el estado), y en segundo lugar al ruido en el proceso de medida. No es necesario que los vectores $\boldsymbol{x}$ y $\boldsymbol{z}$ tengan la misma dimensión. En general la dimensión de $\boldsymbol{z}$ suele ser menor a la de $\boldsymbol{x}$, lo que suele implicar que $\boldsymbol{z}$ tiene menos información que $\boldsymbol{x}$, el verdadero estado del sistema. Recordemos que $\boldsymbol{x}$ alberga toda la información útil para describir el mismo.

Usando el teorema de Bayes podemos relacionar la variable observada $\boldsymbol{z}$ con la variable que buscamos $\boldsymbol{x}$. Para ello es necesario conocer, o inferir, las distribuciones de probabilidad condicionadas de $\boldsymbol{x}$ con respecto a $\boldsymbol{z}$ y viceversa:

$$p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})p(\boldsymbol{z}_{1:k}) = p(\boldsymbol{z}_{1:k}|\boldsymbol{x}_k)p(\boldsymbol{x}_k) \tag{3.1}$$

La estimación bayesiana consiste en calcular la probabilidad del estado actual ($\boldsymbol{x}_k$) en función de todas las observaciones. Para ello se puede expresar la anterior expresión como:

$$p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) = \frac{p(\boldsymbol{z}_{1:k}|\boldsymbol{x}_k)p(\boldsymbol{x}_k)}{p(\boldsymbol{z}_{1:k})} \tag{3.2}$$

Esto forma la base de la teoría de Bayes y será usada a partir de aquí para desarrollar el resto de las expresiones de este capítulo.

En resumen, un filtro bayesiano es cualquier técnica que produzca una estimación de $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$ a partir de $p(\boldsymbol{z}_{1:k}|\boldsymbol{x}_k)$ usando el teorema de Bayes, para cada instante de tiempo $k = 1, 2, ...$ Esto implica que en vez de proporcionar directamente una estimación de $\boldsymbol{x}_k$, un filtro de Bayes busca una estimación de la función de densidad de probabilidad *a posteriori* $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$. Una vez obtenida dicha densidad, resulta inmediato conseguir cualquier estadístico sobre $\boldsymbol{x}_k$, usando cualquiera de los estimadores probabilísticos conocidos. Normalmente se emplea el estimador de mínimo error cuadrático medio, que permite buscar la estimación del estado actual simplemente calculando la media condicional $\hat{\boldsymbol{x}}_k = \int \boldsymbol{x}_k p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})d\boldsymbol{x}_k$.

La estimación de la probabilidad $p(\boldsymbol{x}_k\,|\boldsymbol{z}_{1:k})$ se va haciendo más compleja según aumenta k. Para intentar aliviar el aumento de la complejidad se buscará una formulación recursiva, donde para calcular la estimación actual se pueda utilizar la del instante anterior.

3.2. Filtrado bayesiano recursivo

El filtrado bayesiano recursivo tiene como característica el uso de información de un instante de tiempo para calcular la mejor estimación en el siguiente. Hasta ahora las expresiones las hemos construido siempre usando la información disponible en un instante de tiempo determinado, sin atender a la evolución temporal. Esto, en general, se puede traducir en una pérdida de información, dado que la mayor parte de los procesos en los que estamos interesados se pueden modelar como procesos no

lineales que pueden variar a lo largo del tiempo. Nótese que esta forma de describir al proceso no pierde generalidad alguna.

Existirá una evolución tanto para el estado como para la observación, cada una sujeta a su propia dinámica. Por ahora expresaremos estos modelos como:

$$\boldsymbol{x}_k = f_k(\boldsymbol{x}_{k-1}, \boldsymbol{v}_k) \tag{3.3}$$

$$\boldsymbol{z}_k = h_k(\boldsymbol{x}_k, \boldsymbol{n}_k) \tag{3.4}$$

dónde f_k y h_k son los dos procesos no necesariamente lineales que pueden variar con el tiempo y $\boldsymbol{v}_k$ y $\boldsymbol{n}_k$ son dos variables desconocidas independientes e idénticamente distribuidas (i.i.d.) que se emplean para modelar la incertidumbre.

En estas expresiones se ha supuesto que la evolución del sistema se puede modelar por una secuencia de Markov de orden uno y que la observación se realiza en un canal sin memoria. Las suposiciones realizadas resultan bastante generalistas. Un proceso de Markov de orden uno implica que el único factor relevante para la evolución del sistema es el último estado por el que ha pasado, siendo totalmente irrelevante el resto de la historia pasada del proceso. Un canal sin memoria, por su parte, indica que la observación $\boldsymbol{z}_k$ únicamente depende del estado en ese instante, $\boldsymbol{x}_k$, y no de ningún otro estado anterior. Fenómenos como la histéresis o el eco no se consideran.

En este punto, debido a la naturaleza aleatoria de (3.3), (3.4) y las variables $\boldsymbol{x}_k$ y $\boldsymbol{z}_k$, podemos usar un marco probabilístico para caracterizar todo el proceso usando las funciones de densidad de probabilidad: $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$ y $p(\boldsymbol{z}_k|\boldsymbol{x}_k)$. La primera indica la evolución del sistema y la segunda el proceso de observación. A partir de ahora manejaremos únicamente estas funciones en el resto del desarrollo. Utilizándolas podemos describir el proceso de filtrado, que es la estimación de la secuencia de estados. Deseamos estimar un estado desconocido $\boldsymbol{x}_k$ usando la colección completa de observaciones hasta el instante actual k, $\boldsymbol{z}_{1:k}$. Ya que $\boldsymbol{x}_k$ es una variable aleatoria, toda la información que da $\boldsymbol{z}_k$ sobre ésta aparece en la función de densidad de probabilidad posterior $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$.

Las estimaciones del filtro bayesiano se pueden plantear de manera *recursiva*.

De esta forma la estimación de $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$ se basa únicamente en la función de densidad de probabilidad posterior en el instante anterior, $p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1})$ y la medida disponible más reciente de la observación $\boldsymbol{z}_k$. Esto evita el problema de almacenar la secuencia completa de medidas lo que presenta grandes ventajas computacionales.

La estimación recursiva se puede dividir en dos pasos: *predicción y actualización*. Supongamos por un momento que la densidad posterior buscada $p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1})$ se conoce en un instante $k-1$. El paso de predicción usa el modelo de sistema para obtener la densidad de probabilidad a priori del estado en el instante k usando la ecuación de Chapman-Kolmogorov:

$$p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}) = \int p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1})d\boldsymbol{x}_{k-1} \tag{3.5}$$

La predicción se realiza basándose únicamente en la información disponible en el instante $k-1$. Debe hacerse hincapié en que se ha considerado que el proceso de evolución sigue un proceso de Markov de orden uno. Conceptualmente, para un proceso de este tipo, el valor de la variable aleatoria $\boldsymbol{x}_{k-1}$ proporciona tanta información sobre $\boldsymbol{x}_k$ como el valor de estado en todos los instantes de tiempo hasta $k-1$. En este sentido el estado de un sistema $\boldsymbol{x}_k$ es cualquier colección de variables que hace que la trayectoria $\boldsymbol{x}_{0:k-1}$ sea totalmente intrascendente para estableces los futuros valores de la secuencia. En resumen, el paso de predicción únicamente depende del modelo de evolución y de la última densidad a posteriori disponible.

El segundo de los pasos del filtro actualiza la predicción a priori cuando se toma una nueva medida. La verosimilitud se combina con la probabilidad *a priori* para obtener la función de densidad de probabilidad *a posteriori* del estado actual, usando la conocida regla de Bayes.

$$p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) = \frac{p(\boldsymbol{z}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1})}{p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1})} \tag{3.6}$$

La constante de normalización

$$p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1}) = \int p(\boldsymbol{z}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1})d\boldsymbol{x}_k \tag{3.7}$$

depende de la función de verosimilitud $p(\boldsymbol{z}_k|\boldsymbol{x}_k)$ definida en el modelo de medida.

Aquí hemos asumido que el canal no tiene memoria, de tal forma que la medida únicamente depende del estado actual del sistema, no de estados anteriores ni de medidas anteriores.

Esta propagación recursiva de la densidad *a posteriori* es la solución óptima bayesiana al problema de filtrado. Sin embargo sólo se trata de una solución conceptual que, en general, no es posible aplicar. Esta solución requiere que todas las funciones de densidad de probabilidad se puedan representar de manera analítica, cuestión que, en condiciones generales, resulta complicado. Sólo en una serie de casos restringidos resulta posible abordar la solución exacta.

Un ejemplo de una solución óptima al problema de filtrado bayesiano es el filtro de Kalman [Hay01]. Este filtro aplica las fórmulas bayesianas al caso en el que todas las distribuciones de probabilidad son gaussianas y los modelos de actualización lineales. Con estas restricciones resulta posible abordar computacionalmente la evolución de los modelos analíticos, descritos únicamente por la media y la covarianza. Los filtros de Kalman tienen, por tanto, la ventaja de proporcionar una solución óptima a la vez que muestran una gran eficiencia.

En el resto de los casos no es posible hacer cálculos empleando las expresiones analíticas de las distribuciones de probabilidad. Esto puede ser debido a su complejidad o al hecho de que no siempre se conocen dichas distribuciones.

En la mayor parte de los casos se emplean diferentes tipos de aproximaciones al filtro óptimo bayesiano. Debido a que el espacio de estados es inherentemente continuo, la mayoría de los algoritmos subóptimos empleados están basados en el filtro de Kalman o en alguna de sus aproximaciones muestreadas.

Por un lado existen aproximaciones que intentan modelar las distribuciones de probabilidad involucradas en el sistema por procesos gaussianos y desarrollos en series de Taylor sencillos. Como ejemplo podemos mencionar el filtro extendido de Kalman [Hay01]. La idea principal consiste en obtener una representación analítica manejable de las distribuciones de probabilidad involucradas.

También existen aproximaciones basadas en representaciones muestrales de las distribuciones de probabilidad. La ventaja de estas representaciones es que son capaces de abordar cualquier tipo de problema, sea cual sea la distribución de probabilidad involucrada. La única limitación consiste en nuestra capacidad de añadir más mues-

tras a la representación, cuestión que dependerá de la capacidad de cómputo. Con las muestras se obtiene una representación no paramétrica de la distribución de probabilidad, fácilmente manejable por un ordenador. Esta forma es mucho más sencilla que emplear una representación analítica en el caso general (es decir, no gaussiano). Al trabajar con muestras resulta fácil implementar las dos etapas de un filtro bayesiano. Además, las representaciones muestrales convierten las integrales en sumatorios, reduciendo aún más la complejidad del sistema.

Sin embargo, existe una importante desventaja al usar esta aproximación. No tenemos una forma directa para seleccionar las mejores muestras para una función en particular, por lo que es necesario encontrar algún método para seleccionarlas.

Dentro de las aproximaciones muestradas podemos encontrar diferentes métodos para colocar las muestras a lo largo de la distribución de probabilidad. Por un lado tenemos las técnicas basadas en rejillas. Estas técnicas consisten en realizar un división uniforme del espacio, a modo de rejilla, y realizar el muestreo en cada una de las celdas de la rejilla. De esta formar tendremos un muestreo completo y, según disminuyamos el tamaño de la rejilla, cada vez más preciso. Estas técnicas presentan un problema importante de eficiencia dado que pueden colocar muestras en zonas del espacio sin interés alguno y no escalan cuando el tamaño del espacio de estados aumenta.

Otro de los métodos más populares es emplear los métodos de Monte Carlo para realizar las representaciones de las distribuciones. Los métodos de Monte Carlo son la base de los filtros de partículas, quizá el método subóptimo más conocido para los filtros bayesianos. A continuación realizaremos una breve introducción a los métodos de Monte Carlo.

3.3. Introducción a las técnicas de Monte Carlo

Como hemos comentado, resulta difícil conseguir una representación analítica de una función de probabilidad *a posteriori*. Una representación muestral de esta función suele ser más fácil de obtener, mantener y actualizar. Además, permite una representación compacta de dicha función. Una forma de obtener dichas representaciones consiste en emplear métodos basados en técnicas de Monte Carlo.

En esta sección vamos a presentar los métodos de Monte Carlo de manera general. Para más detalles sobre estas técnicas, sus teoremas, pruebas y una lista completa de referencias se pueden consultar [AFDJ03, DdFG01, Mac03].

Los métodos de Monte Carlo son técnicas computacionales que hacen uso extensivo de los números aleatorios para conseguir muestras representativas de una determinada distribución de probabilidad. Con un número suficiente de dichas muestras se puede obtener una estimación numérica de la distribución de probabilidad suficientemente fiable. Así mismo es posible obtener esperanzas o integrales sobre estas distribuciones, de la forma:

$$I(f) = \int f(\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x} \tag{3.8}$$

La anterior expresión es difícil de calcular a no ser que la distribución $p(\boldsymbol{x})$ sea sencilla. Nosotros partimos del caso en que $p(\boldsymbol{x})$ es lo suficientemente compleja como para no poder evaluar esta expresión usando métodos directos. Este es el caso en el que estamos interesados en emplear los métodos de Monte Carlo. Nos centraremos en el problema de muestreo. Esta es la llave para abordar el resto de problemas. El problema de estimación puede resolverse con un conjunto de muestras aleatorias tomadas de la distribución $p(\boldsymbol{x})$, $\{\boldsymbol{x}^i\}_{i=1}^N$. Con ellas se puede construir un estimador para (3.8) de la siguiente forma:

$$\hat{I}(f) = \frac{1}{N}\sum_{i=0}^{N-1} f(\boldsymbol{x}^i) \tag{3.9}$$

De esta forma convertimos la integral de (3.8) en un sumatorio sobre las muestras obtenidas de $p(\boldsymbol{x})$. Si los vectores $\{\boldsymbol{x}^i\}_{i=1}^N$ se generan como extracciones aleatorias de $p(\boldsymbol{x})$, entonces la esperanza de $\hat{I}(f)$ es $I(f)$, según la ley de los grandes números. Además, según aumenta el número de muestras N, la varianza de $\hat{I}(f)$ disminuirá según σ^2/N, dónde σ^2 es la varianza de f,

$$\sigma^2 = \int (f(\boldsymbol{x}) - I(f))^2 p(\boldsymbol{x})d\boldsymbol{x} \tag{3.10}$$

Esta es una de las características más importantes de Monte Carlo. La precisión de la estimación de Monte Carlo únicamente depende de la varianza de f, no de la

dimensión del espacio muestreado. Independientemente de la dimensión de $\boldsymbol{x}$, puede darse el caso en que unas pocas decenas de muestras independientes $\{\boldsymbol{x}^i\}$ resulten suficientes para estimar $I(f)$ de manera satisfactoria. La integración de Monte Carlo usará puntos en las regiones de mayor probabilidad a la hora de realizar la estimación de la integral, dado que $\boldsymbol{x}^i$ se obtiene de dicha distribución. Podemos decir que los métodos de Monte Carlo son capaces de focalizar la atención en aquellos puntos más relevantes, según $p(\boldsymbol{x})$ a la ahora de obtener la estimación de $I(f)$. Esto contrasta con una integración uniforme que colocará puntos en todas las regiones por igual sin tener en cuenta su probabilidad, consumiendo capacidad de cómputo en regiones de poco interés, como son las zonas de baja probabilidad.

En un caso general $p(\boldsymbol{x})$ es desconocida, o aunque se conozca, resulta imposible extraer muestras e ella. Una posibilidad es colocar muestras uniformes a lo largo del espacio de estados y evaluar $p(\boldsymbol{x})$ en esos puntos o, por lo menos, algún valor proporcional a éste, que llamaremos $p^*(\boldsymbol{x})$. Luego puede introducirse el factor de normalización Z, definido como:

$$Z = \sum_{i=1}^{N} p^*(\boldsymbol{x}^i) \tag{3.11}$$

y estimar $I(f) = \int f(\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x}$ de la siguiente forma:

$$\hat{I}(f) = \sum_{i=1}^{N} f(\boldsymbol{x}^i)\frac{p^*(\boldsymbol{x}^i)}{Z} \tag{3.12}$$

convirtiendo una integral en un simple sumatorio. Aunque teóricamente este método funciona de manera correcta, tiene un gran inconveniente. Generalmente la mayor parte de $p(\boldsymbol{x})$ se concentrará en una pequeña región del espacio, lo que afectará de manera importante a la estimación $\hat{I}(f)$ incluso si $f(\boldsymbol{x})$ es una función benigna y suave. La región dónde se concentra la mayor parte de $p(\boldsymbol{x})$ se conoce como *conjunto típico T*, cuyo tamaño viene dado por $|T| \approx 2^{H(\boldsymbol{X})}$, dónde $H(\boldsymbol{X})$ es la entropía de Shannon-Gibbs de la distribución de probabilidad $p(\boldsymbol{x})$ [Mac03]:

$$H(\boldsymbol{X}) = \sum_{\boldsymbol{x}} p(\boldsymbol{x}) \log_2 \frac{1}{p(\boldsymbol{x})} \tag{3.13}$$

Si casi toda la masa de probabilidad está concentrada en el conjunto típico y $f(\boldsymbol{x})$ es una función benigna, el valor de $I(f) = \int f(\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x}$ vendrá principalmente determinado por los valores de $f(\boldsymbol{x})$ tomados del conjunto típico. Un muestreo uniforme solo daría una buena estimación de $I(f)$ si hay un número suficientemente grande de muestras de tal forma que resulte probable que unas cuantas caigan en el conjunto típico. El número de muestras necesarias dependería del tamaño del conjunto típico y del tamaño del área de muestreo. La proporción de partículas que caen en el conjunto típico empeora en problemas con alta dimensionalidad. Si la distribución $p(\boldsymbol{x})$ no es verdaderamente uniforme, el muestreo uniforme probablemente resulte de poca utilidad [DdFG01].

Existen otras soluciones que se comportan mejor que el muestreo uniforme cuando se desconoce $p(\boldsymbol{x})$. Su objetivo es conseguir colocar el mayor número de muestras aleatorias sobre el conjunto típico, mejorando los resultados que el muestreo uniforme tiene en este sentido. Para ello se puede usar otra distribución diferente a $p(\boldsymbol{x})$ y, posteriormente, aplicar algún método de corrección para tener en cuenta que se ha muestreado de la distribución incorrecta. Así es como funcionan el muestreo con rechazo y el muestreo enfatizado.

3.4. Muestreo con rechazo

El muestreo "con rechazo" (del inglés *Rejection Sampling*) es el primero de los métodos que analizaremos para obtener muestras aleatorias de una distribución de probabilidad. Para ello partiremos de una distribución de probabilidad de una dimensión, $p(x) = p^*(x)/Z$, de la que resulta muy difícil obtener muestras directamente. Supongamos que tenemos otra función más sencilla, $q(x)$, a la que llamaremos *función de propuesta*, que sí podemos evaluar (hasta una factor de proporcionalidad Z_q) y de la que podemos obtener muestras sin problemas. Además asumiremos que conocemos el valor de una constante c tal que

$$\forall x, \;\; cq^*(x) > p^*(x) \tag{3.14}$$

El proceso de muestreo con rechazo genera dos números aleatorios. El primero,

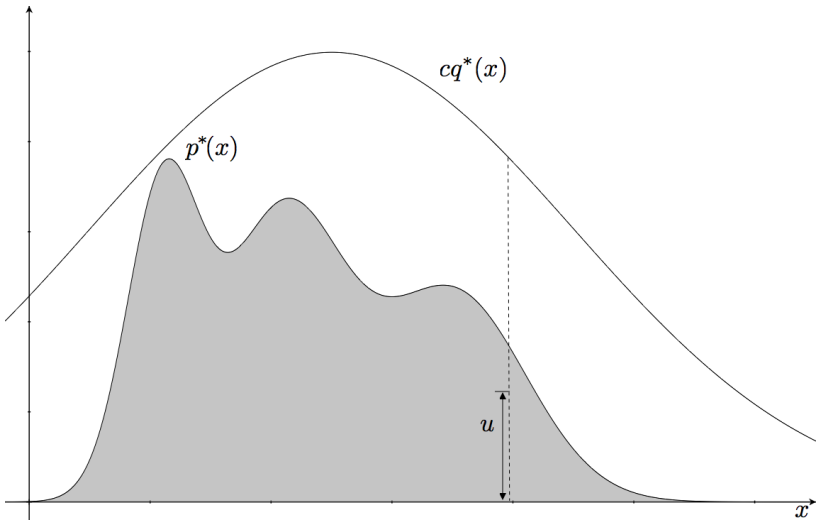

Figura 3.1: El muestreo con rechazo selecciona x de $cq^*(x)$, a continuación acepta x si la variable aleatoria $u \in [0, cq^*(x)]$ es más pequeña que $p(x)$. De esta forma todas las x seleccionadas se distribuirán según $p(x)$.

x se extrae de la función de propuesta $q(x)$. Entonces evaluamos $cq^*(x)$ y extraemos el segundo número aleatorio, u, usando una distribución uniforme en el intervalo $[0, cq^*(x)]$. Estos dos números se pueden interpretar como seleccionar un punto en el plano bidimensional de la figura 3.1.

Podemos evaluar $p^*(x)$ y aceptar o rechazar la muestra x comparando el valor de u con el valor de $p^*(x)$. Si $u > p^*(x)$, entonces se rechaza x; en caso contrario se acepta, lo que implica que añadimos x a nuestro conjunto de muestras $\{x^i\}$.

Este proceso resulta equivalente a muestrear directamente de $p(x)$ dado que las x aceptadas se distribuyen realmente según $p(x)$. Los puntos propuestos (x, u) proceden del área bajo la curva $cq^*(x)$ de la figura 3.1 con una probabilidad uniforme. La regla de rechazo excluye todos los puntos que caen sobre la curva $p^*(x)$. Así que los puntos (x, u) aceptados se distribuyen uniformemente en el área fuertemente sombreada bajo $p^*(x)$. Esto implica que la densidad de probabilidad de las coordenadas x de los puntos aceptados debe ser proporcional a $p^*(x)$, o lo que es lo mismo, muestras independientes de $p(x)$ [Mac03]. El algoritmo seguido se resume en el cuadro 3.1.

El muestreo con rechazo es capaz de trabajar con funciones de densidad de probabilidad genéricas, siempre y cuando podamos evaluar dichas funciones hasta un

$\left[\{x^i\}_{i=1}^N\right] = MUESTREO\_CON\_RECHAZO$

- Obtén $x^i \sim q(x)$
- Genera $u \sim [0, cp^*(x)]$
- Acepta x si $u \leq p^*(x)$
- Rechaza x en caso contrario
- Repite hasta obtener N muestras

Cuadro 3.1: Algoritmo de muestreo con rechazo.

factor de proporcionalidad. Sin embargo puede resultar difícil encontrar la constate c de (3.14) y el problema se agrava aún más con espacios de altas dimensiones. Normalmente esta constate será alta, a no ser que p y q sean similares. Si la constante c es alta, el sistema de rechazo descartará muchas muestras. El rechazo no es un fenómeno deseable, dado que se consumen recursos computacionales en puntos que no tendrán ningún efecto sobre el conjunto final. El muestreo con rechazo es, por tanto, útil en problemas unidimensionales pero no resulta práctico para generar muestras de distribuciones multidimensionales.

3.5. Muestreo enfatizado

El muestreo "enfatizado" (del inglés *Importance Sampling*), a diferencia del muestreo con rechazo, no es tanto un método para generar muestras de $p(\boldsymbol{x}|\boldsymbol{z})$, sino un método para estimar la esperanza de una función $f(\boldsymbol{x})$, a la vista de una serie de observaciones, o incluso obtener una estimación de la propia distribución de probabilidad $p(\boldsymbol{x}|\boldsymbol{z})$. Puede verse como una generalización del método de muestreo uniforme y se trata de una alternativa "clásica" que se remonta a los años 40 del siglo pasado.

Para introducir este tipo de muestreo partiremos, como hicimos en el caso de muestreo con rechazo, de la suposición de que la distribución $p(\boldsymbol{x}|\boldsymbol{z})$ resulta demasiado complicada como para muestrear directamente de ella. En caso contrario no sería necesario recurrir a este tipo de métodos. Para evitar muestrear directamente de

la distribución *a posteriori* se introduce, de nuevo, una distribución arbitraria $q(\boldsymbol{x}|\boldsymbol{z})$, más simple, de la que podemos sacar muestras fácilmente y a la que podemos evaluar hasta una constante de proporcionalidad arbitraria. De nuevo nos referimos a esta distribución como función o distribución de propuesta.

En vez de muestrear directamente de $p(\boldsymbol{x}|\boldsymbol{z})$, el muestreo enfatizado genera muestras independientes directamente de $q(\boldsymbol{x}|\boldsymbol{z})$. Si estas muestras fueran muestras independientes de $p(\boldsymbol{x}|\boldsymbol{z})$ podríamos estimar $I(f)$ directamente aplicando (3.9), como hicimos en el caso del muestreo con rechazo. Desgraciadamente en este caso las muestras no han sido generadas por $p(\boldsymbol{x}|\boldsymbol{z})$ sino por otra distribución, así que aplicar esta sencilla fórmula no es posible. Al utilizar $q(\boldsymbol{x}|\boldsymbol{z})$ para obtener las muestras éstas estarán localizadas en aquellas zonas en las que $q(\boldsymbol{x}|\boldsymbol{z})$ sea más probable. En concreto, aquellas zonas en las que $q(\boldsymbol{x}|\boldsymbol{z})$ sea mayor que $p(\boldsymbol{x}|\boldsymbol{z})$, estarán sobrerrepresentadas en el conjunto de partículas y aquéllas en las que $q(\boldsymbol{x}|\boldsymbol{z})$ sea menor que $p(\boldsymbol{x}|\boldsymbol{z})$ sucederá lo contrario. Para considerar el hecho de muestrear de una función "errónea", se utiliza la siguiente función correctora:

$$w^*(\boldsymbol{x}|\boldsymbol{z}) \equiv \frac{p(\boldsymbol{x}|\boldsymbol{z})}{q(\boldsymbol{x}|\boldsymbol{z})} \tag{3.15}$$

cuya misión es compensar el número de veces que aparece una muestra ($q(\boldsymbol{x}_k|\boldsymbol{z}_k)$) por el número de veces que realmente debería aparecer ($p(\boldsymbol{x}_k|\boldsymbol{z}_k)$).

Utilizando esta función, $w^*(\boldsymbol{x}|\boldsymbol{z})$, podemos remplazar $p(\boldsymbol{x}|\boldsymbol{z})$ por $q(\boldsymbol{x}|\boldsymbol{z})$ al calcular $I(f)$:

$$I(f) = \int f(\boldsymbol{x})p(\boldsymbol{x}|\boldsymbol{z})d\boldsymbol{x} = \int f(\boldsymbol{x})q(\boldsymbol{x}|\boldsymbol{z})w^*(\boldsymbol{x}|\boldsymbol{z})d\boldsymbol{x} = E_q[f(\boldsymbol{x})w^*(\boldsymbol{x}|\boldsymbol{z})|\boldsymbol{z}] \tag{3.16}$$

$E_q[\cdot|\boldsymbol{z}]$ es la esperanza con respecto a $q(\boldsymbol{x}|\boldsymbol{z})$. Las muestras también son llamadas *puntos soporte* puesto que se emplean como soportes a la hora de construir la estimación a través del sumatorio.

La ventaja principal que proporciona el muestreo enfatizado es que resulta, por diseño, más sencillo muestrear de $q(\boldsymbol{x}|\boldsymbol{z})$ que de $p(\boldsymbol{x}|\boldsymbol{z})$ para obtener la solución exacta del problema. En cuanto a las restricciones a la hora de elegir $q(\boldsymbol{x}|\boldsymbol{z})$ son

pocas. Al contrario que en el muestreo con rechazo, no es necesario conocer una constante c tal que $cq(\boldsymbol{x}|\boldsymbol{z}) \leq p(\boldsymbol{x}|\boldsymbol{z}), \forall \boldsymbol{x}$. La función de propuesta no tiene esta restricción. La única restricción que se pone sobre esta función es que el soporte de $q(\boldsymbol{x}|\boldsymbol{z})$ incluya al soporte de $p(\boldsymbol{x}|\boldsymbol{z})$, es decir, que $q(\boldsymbol{x}|\boldsymbol{z})$ sea distinta de cero en, por lo menos, todos los puntos en los que $p(\boldsymbol{x}|\boldsymbol{z})$ es diferente de cero. Este requisito es necesario o en caso contrario habría puntos de $p(\boldsymbol{x}|\boldsymbol{z})$ que no podrían alcanzarse por ninguna muestra extraída de la distribución de propuesta. Una vez que se cumpla esto se puede probar que el estimador $\hat{I}(f)$ converge a $I(f)$, el valor medio ponderado de $f(\boldsymbol{x})$, al aumentar el número de partículas.

Para hacer un estimador de Monte Carlo con el muestreo enfatizado vamos a utilizar una representación muestral para calcular la integral de (3.16). La esperanza de $I(f)$ se puede aproximar usando N muestras independientes $\{\boldsymbol{x}^i\}_{i=1}^N$, extraídas de $q(\boldsymbol{x}|\boldsymbol{z})$, como

$$\hat{I}_N^*(f) = \sum_{i=1}^{N} f(\boldsymbol{x}^i) w^{*i}, \tag{3.17}$$

donde w^{*i} es la abreviatura de $w^*(\boldsymbol{x}|\boldsymbol{z})$. Al conjunto de los pesos $\{w^{*i}\}_{i=1}^N$, que acompaña siempre al conjunto de partículas, se le suele llamar pesos de importancia. La definición explícita de estos pesos es

$$w^{*i} \equiv \frac{p(\boldsymbol{x}^i|\boldsymbol{z})}{q(\boldsymbol{x}^i|\boldsymbol{z})} = \frac{p(\boldsymbol{z}|\boldsymbol{x}^i)p(\boldsymbol{x}^i)}{p(\boldsymbol{z})q(\boldsymbol{x}^i|\boldsymbol{z})} \tag{3.18}$$

en la que hemos aplicado la regla de Bayes para reescribir $p(\boldsymbol{x}^i|\boldsymbol{z})$.

Como sucede en (3.9), la estimación $\hat{I}_N^*(f)$ es insesgada. Si la varianza de $f(\boldsymbol{x})$ con respecto a $q(\boldsymbol{x}|\boldsymbol{z})$ es finita, entonces se puede demostrar que, según la ley de los grandes números, $\hat{I}_N^*(f)$ converge a $I(f)$ con probabilidad igual a uno cuando $N \to \infty$.

Desgraciadamente, existe un problema práctico en (3.18) que es necesario resolver. Hasta ahora hemos supuesto que $p(\boldsymbol{x}|\boldsymbol{z})$ podía evaluarse aunque resultara imposible extraer muestras de dicha distribución. Para el tipo de modelos estocásticos que estamos considerando, es decir, secuencias de estados markovianas observadas a través de un canal sin memoria, éste no suele ser el caso. Mientras que suele ser

relativamente directo evaluar la verosimilitud $p(\boldsymbol{z}|\boldsymbol{x})$ y la distribución *a priori* $p(\boldsymbol{x})$ para este tipo de modelos, el cálculo del término de normalización del denominador (3.18) es, en muchos casos, inviable.

Para evitar la necesidad de evaluar $p(\boldsymbol{z})$, partimos de (3.16), reescribiéndola de la siguiente forma.

$$I(f) = \int f(\boldsymbol{x}) \frac{p(\boldsymbol{z}|\boldsymbol{x})p(\boldsymbol{x})}{q(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})} q(\boldsymbol{x}|\boldsymbol{z}) d\boldsymbol{x} \tag{3.19}$$

$$= \frac{1}{p(\boldsymbol{z})} \int f(\boldsymbol{x}) \frac{p(\boldsymbol{z}|\boldsymbol{x})p(\boldsymbol{x})}{q(\boldsymbol{x}|\boldsymbol{z})} q(\boldsymbol{x}|\boldsymbol{z}) d\boldsymbol{x} \tag{3.20}$$

Una vez alcanzado este punto se puede aplicar la regla de Bayes obteniendo:

$$I(f) = \frac{\int f(\boldsymbol{x}) \frac{p(\boldsymbol{z}|\boldsymbol{x})p(\boldsymbol{x})}{q(\boldsymbol{x}|\boldsymbol{z})} q(\boldsymbol{x}|\boldsymbol{z}) d\boldsymbol{x}}{\int \frac{p(\boldsymbol{z}|\boldsymbol{x})p(\boldsymbol{x})}{q(\boldsymbol{x}|\boldsymbol{z})} q(\boldsymbol{x}|\boldsymbol{z}) d\boldsymbol{x}} \tag{3.21}$$

$$= \frac{E_q[f(\boldsymbol{x})w(\boldsymbol{x})|\boldsymbol{z}]}{E_q[w(\boldsymbol{x})|\boldsymbol{z}]} \tag{3.22}$$

donde los nuevos pesos

$$w(\boldsymbol{x}) = \frac{p(\boldsymbol{z}|\boldsymbol{x})p(\boldsymbol{x})}{q(\boldsymbol{x}|\boldsymbol{z})} \tag{3.23}$$

sí pueden calcularse. El término de normalización $p(\boldsymbol{z})$ de (3.20) se ha expandido con la integral del denominador de (3.21). En este caso, un conjunto de N muestras independientes sacadas de $q(\boldsymbol{x}|\boldsymbol{z})$ pueden usarse para estimar las dos esperanzas presentes en (3.22), por lo que obtenemos una nueva aproximación de $I(f)$:

$$\hat{I}_N(f) = \frac{\frac{1}{N}\sum_{i=1}^{N} f(\boldsymbol{x}^i)w^i}{\frac{1}{N}\sum_{j=1}^{N} w^j} = \sum_{i=1}^{N} f(\boldsymbol{x}^i)\overline{w}^i, \tag{3.24}$$

donde w^i es una abreviatura para $w(\boldsymbol{x}^i)$ y $\overline{w}^i$ es

$$\overline{w}^i = \frac{w^i}{\sum_{j=1}^{N} w^j} = \frac{w(\boldsymbol{x}^i)}{\sum_{j=1}^{N} w(\boldsymbol{x}^j)} \tag{3.25}$$

De (3.25), podemos observar que los pesos $\{\overline{w}^i\}$ están normalizados para sumar uno. Con el fin de evitar confusiones, de aquí en adelante nos referiremos a $\{w^{*i}\}$ como los "verdaderos" pesos y los $\{w^i\}$ y $\{\overline{w}^i\}$ como los pesos sin normalizar y normalizados respectivamente. Por añadidura, todas las referencias a los pesos implicarán la aproximación $\hat{I}_N(f)$ tal y como aparece en (3.24), a no ser que se indique lo contrario (en contraposición a $\hat{I}^*_N(f)$ en (3.17)). Dado que el estimador $\hat{I}_N(f)$ es una relación entre estimaciones, estará generalmente sesgado. Además, la ley de los grandes número todavía podrá aplicarse y por lo tanto $\hat{I}_N(f)$ converge a $I(f)$, con probabilidad uno cuando $N \to \infty$. Comparando (3.17) y (3.24) se observa que los pesos verdaderos desconocidos $\{w^{*i}\}$ se aproximan por $\{N\overline{w}^i\}$ en $\hat{I}_N(f)$.

Antes de continuar, resulta interesante reescribir la aproximación $\hat{I}_N(f)$ dada por el muestreo enfatizado de una forma ligeramente diferente. Representando la delta de Dirac como $\delta(\boldsymbol{x} - \boldsymbol{x}^i)$ podemos escribir (3.24) de la siguiente forma:

$$\hat{I}_N(f) = \sum_{i=1}^{N} \left[f(\boldsymbol{x})\delta(\boldsymbol{x} - \boldsymbol{x}^i)d\boldsymbol{x} \right] \overline{w}^i = \int f(\boldsymbol{x}) \sum_{i=1}^{N} \left(\overline{w}^i \delta(\boldsymbol{x} - \boldsymbol{x}^i) \right) d\boldsymbol{x} \qquad (3.26)$$

Entonces, definiendo la medida empírica (aleatoria) $\hat{p}_N(\boldsymbol{x}|\boldsymbol{z})$ generada por las muestras $\{\boldsymbol{x}^i\}$ extraídas de $q(\boldsymbol{x}|\boldsymbol{z})$ como

$$\hat{p}_N(\boldsymbol{x}|\boldsymbol{z}) \triangleq \sum_{i=1}^{N} \overline{w}^i \delta(\boldsymbol{x} - \boldsymbol{x}^i), \qquad (3.27)$$

podemos relacionar (3.16) y (3.26) como

$$\hat{I}_N(f) = \int f(\boldsymbol{x})\hat{p}_N(\boldsymbol{x}|\boldsymbol{z})d\boldsymbol{x} \approx \int f(\boldsymbol{x})p(\boldsymbol{x}|\boldsymbol{z})d\boldsymbol{x}, \qquad (3.28)$$

donde la aproximación mejora cuando $N \to \infty$.

La expresión (3.28) resulta muy útil porque indica el sentido en el cuál el muestreo enfatizado se puede considerar como una técnica de estimación de densidades de probabilidad. Podemos pensar en los resultados del algoritmo de muestreo enfatizado tanto como una aproximación de Monte Carlo para la integral $\int f(\boldsymbol{x})p(\boldsymbol{x}|\boldsymbol{z})d\boldsymbol{x}$ o como una estimación empírica $\hat{p}_N(\boldsymbol{x}|\boldsymbol{z})$ de la distribución *a posteriori* $p(\boldsymbol{x}|\boldsymbol{z})$. Estas

$\left[\{\boldsymbol{x}_k^i, w_k^i\}_{i=1}^N\right] = MUESTREO\_ENFATIZADO\left[\boldsymbol{z}_k\right]$

- Para cada $i = 1 : N$
  - Obtén $\boldsymbol{x}^i \sim q(\boldsymbol{x}|\boldsymbol{z})$
  - Asigna w^i usando la expresión 3.25.

Cuadro 3.2: Algoritmo de muestreo enfatizado.

dos formas de ver el algoritmo son dos caras de la misma moneda. Como nos queremos centrar en el filtrado bayesiano y el objetivo de éste es la distribución *a posteriori* $p(\boldsymbol{x}|\boldsymbol{z})$, adoptaremos esta última interpretación del algoritmo.

En resumen, el algoritmo de muestreo enfatizado nos proporciona un método para construir una aproximación empírica a la verdadera distribución *a posteriori*, sin que sea necesario ser capaz de muestrear de $p(\boldsymbol{x}|\boldsymbol{z})$.

En el cuadro 3.2 puede verse un resumen de los pasos del algoritmo. Al contrario de lo que sucede en el muestreo con rechazo y con el algoritmo de Metropolis-Hastings [Mac03], donde los puntos $\{\boldsymbol{x}^i\}$ se distribuyen según la verdadera distribución *a posteriori*, las muestras generadas por el algoritmo de muestreo enfatizado se distribuyen según la distribución $q(\boldsymbol{x}|\boldsymbol{z})$. Sin embargo, al calcular los pesos normalizados $\{\overline{w}^i\}$, el algoritmo llega a una estimación empírica de $p(\boldsymbol{x}|\boldsymbol{z})$, como se indica en (3.27). Más aún, el muestreo enfatizado es la única técnica de este tipo que es capaz de enfrentarse a aplicaciones en tiempo real porque resulta eficiente (es decir, todas las muestras se usan, al contrario que en el muestreo con rechazo) y no requiere largos períodos de incubación como en el Metropolis-Hastings [AFDJ03].

Un problema práctico del muestreo enfatizado es que resulta difícil calcular cuan fiable es el estimador $\hat{I}(f)$. La varianza de $\hat{I}(f)$ es difícil de calcular porque las varianzas empíricas de las cantidades w_i y $w_i f(\boldsymbol{x}^i)$ no son necesariamente una buena indicación de las verdaderas varianzas del numerador y denominador en la ecuación (3.24). Si la distribución de propuesta $q(\boldsymbol{x}|\boldsymbol{z})$ es pequeña en una región en la que $|f(\boldsymbol{x})p(\boldsymbol{x}|\boldsymbol{z})|$ es grande, entonces es bastante probable que, incluso tras generar un número importante de muestras, ninguna de ellas haya caído en dicha región. Esto conduce a una estimación radicalmente incorrecta y no tendremos ningún indicador en la varianza empírica que nos diga que la verdadera varianza del estimador $\hat{I}(f)$ es

alta.

Hasta ahora hemos presentado los fundamentos del muestreo enfatizado para el caso de un único estado, $p(\boldsymbol{x}|\boldsymbol{z})$. A partir de ahora vamos a extender este algoritmo para la estimación secuencial de densidades del tipo $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$ para $k = 1, 2, \ldots$.

3.6. Muestreo enfatizado secuencial

En esta sección analizaremos la extensión secuencial del algoritmo de muestreo enfatizado. El algoritmo de muestreo enfatizado secuencial (del inglés *Sequential Importance Sampling* o SIS) nos permitirá aproximar las densidades *a posteriori* de la forma $p(\boldsymbol{x}_k|\boldsymbol{z}_k)$ en cada instante de tiempo usando las del instante anterior, necesarias para el filtrado bayesiano recursivo.

Para que el algoritmo de muestreo enfatizado sea recursivo tendremos que restringir la forma de las posibles funciones de propuesta.

En la anterior sección se mostró cómo el muestreo enfatizado podía usarse para generar una aproximación aleatoria $\hat{p}_N(\boldsymbol{x}|\boldsymbol{z})$ de la verdadera densidad a posteriori. De (3.27) podemos ver que $\hat{p}_N(\boldsymbol{x}|\boldsymbol{z})$ se encuentra completamente especificada por el conjunto de valores $\{\boldsymbol{x}^i, \overline{w}^i\}_{i=1}^N$, dónde $\overline{w}^i$ es el peso enfatizado correspondiente a $\boldsymbol{x}^i$. Así $\hat{p}_N(\boldsymbol{x}|\boldsymbol{z})$ consiste en una serie de puntos de soporte (aleatorios) y sus correspondientes pesos. En el contexto de filtrado bayesiano, cada punto de soporte es en realidad una secuencia de estados aleatoriamente generada, que denotamos por $\boldsymbol{x}_{0:k}^i$. En este caso nuestra medida aleatoria $\hat{p}_N(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k})$ viene dada por el conjunto $\{x_{0:k}^i, \overline{w}_k^i\}_{i=1}^N$, donde los pesos sin normalizar introducidos en (3.23) ahora satisfacen que

$$w_k^i = \frac{p(\boldsymbol{z}_{1:k}|\boldsymbol{x}_{0:k}^i)p(\boldsymbol{x}_{0:k}^i)}{q(\boldsymbol{x}_{0:k}^i|\boldsymbol{z}_{1:k})} \tag{3.29}$$

Por lo tanto, hasta que la medida actual $\boldsymbol{z}_k$ esté disponible, la extensión secuencial del algoritmo de muestreo enfatizado debe transformar la medida aleatoria $\{\boldsymbol{x}_{0:k-1}^i, \overline{w}_{k-1}^i\}_{i=1}^N$ en $\{\boldsymbol{x}_{0:k}^i, \overline{w}_k^i\}_{i=1}^N$. Si la distribución de propuesta elegida se puede factorizar como:

$$q(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k}) = q(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}, \boldsymbol{z}_{1:k})q(\boldsymbol{x}_{0:k-1}|\boldsymbol{z}_{1:k-1}) \tag{3.30}$$

entonces una muestra $\boldsymbol{x}^i_{0:k} \sim q(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k})$ puede obtenerse por el sencillo procedimiento de:

1. Obtener una nueva $\boldsymbol{x}^i_k \sim q(\boldsymbol{x}_k|x^i_{0:k-1}, \boldsymbol{z}_{1:k})$.

2. Ajustar $\boldsymbol{x}^i_{0:k} = \{\boldsymbol{x}^i_{0:k-1}, \boldsymbol{x}^i_k\}$.

Sin la factorización de (3.30), extraer una muestra de $q(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k})$ requeriría que toda la historia previa $\boldsymbol{x}_{0:k-1}$ sea remuestreada. Claramente, según crece k, esto se convierte en inmanejable para cualquier aplicación. Sin embargo, con la factorización introducida en (3.30), sólo es necesario extraer N vectores de estado en cada actualización. Esta factorización de la función de propuesta también simplifica el cálculo de los pesos, como podemos ver sustituyendo (3.30) en (3.29), de lo que se obtiene:

$$\begin{aligned} w^i_k &= \frac{p(\boldsymbol{z}_{1:k}|\boldsymbol{x}^i_{0:k})p(\boldsymbol{x}^i_{0:k})}{q(\boldsymbol{x}^i_k|\boldsymbol{x}^i_{0:k-1}, \boldsymbol{z}_{1:k})q(\boldsymbol{x}^i_{0:k-1}|\boldsymbol{z}_{1:k-1})} && (3.31)\\ &= \frac{p(\boldsymbol{z}_k|\boldsymbol{x}^i_k)p(\boldsymbol{x}^i_k|\boldsymbol{x}^i_{k-1})}{q(\boldsymbol{x}^i_k|\boldsymbol{x}^i_{0:k-1}, \boldsymbol{z}_{1:k})} \cdot \frac{p(\boldsymbol{z}_{1:k-1}|\boldsymbol{x}^i_{0:k-1})p(\boldsymbol{x}^i_{0:k-1})}{q(\boldsymbol{x}^i_{0:k-1}|\boldsymbol{z}_{1:k-1})} && (3.32)\\ &= \frac{p(\boldsymbol{z}_k|\boldsymbol{x}^i_k)p(\boldsymbol{x}^i_k|\boldsymbol{x}^i_{k-1})}{q(\boldsymbol{x}^i_k|\boldsymbol{x}^i_{0:k-1}, \boldsymbol{z}_{1:k})} \cdot w^i_{k-1} && (3.33) \end{aligned}$$

donde se hace uso de la independencia markoviana de $\boldsymbol{x}_{0:k}$ y de la independencia condicional de $\boldsymbol{z}_{1:k}$ dado $\boldsymbol{x}_{0:k}$ para llegar de (3.31) a (3.32).

Como deseamos realizar un filtrado recursivo bayesiano debemos modificar ligeramente la distribución de propuesta en el denominador de (3.33). En primer lugar la distribución de propuesta debe redefinirse como condicional únicamente a la medida actual $\boldsymbol{z}_k$, porque no queremos almacenar la secuencia de medidas $\boldsymbol{z}_{1:k}$ completa. En segundo lugar, como nos interesa obtener una estimación para la densidad filtrada $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$, sería preferible que no fuera necesario almacenar la historia completa de la secuencia de estados $\boldsymbol{x}^i_{0:k-1}$ para obtener $\boldsymbol{x}^i_k$. Esto implica eliminar la dependencia

de $\boldsymbol{x}_{0:k-2}^i$ en la distribución de propuesta de (3.33). Aplicando estas dos restricciones de manera recursiva a $q(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k})$ llegamos a la siguiente distribución de propuesta:

$$q(\boldsymbol{x}_{0:k}^i|\boldsymbol{z}_{1:k}) = q(\boldsymbol{x}_0^i) \prod q(\boldsymbol{x}_j^i|\boldsymbol{x}_{j-1}^i, \boldsymbol{z}_j). \tag{3.34}$$

Los pesos sin normalizar $\{w_k^i\}$ de (3.33) pasan a tener la siguiente sencilla formulación:

$$w_k^i = w_{k-1}^i \cdot \frac{p(\boldsymbol{z}_k|\boldsymbol{x}_k^i)p(\boldsymbol{x}_k^i|\boldsymbol{x}_{k-1}^i)}{q(\boldsymbol{x}_k^i|\boldsymbol{x}_{k-1}^i, \boldsymbol{z}_k)} \tag{3.35}$$

Como la distribución de propuesta $q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^i, \boldsymbol{z}_k)$ ya no se encuentra condicionada a la historia pasada de las medidas $\boldsymbol{z}_{1:k-1}$, la ecuación (3.35) resulta adecuada para una implementación recursiva. Esta es la forma más usada en casi todas las implementaciones de este tipo de filtros.

Sin embargo, independientemente de que se use (3.33) o (3.35) para calcular los pesos $\{w_k^i\}$, el algoritmo del muestreo enfatizado secuencial proporciona la siguiente estimación empírica de la distribución de probabilidad buscada:

$$\hat{p}_N(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) = \sum_{i=1}^{N} \overline{w}^i \delta(\boldsymbol{x}_k - \boldsymbol{x}_k^i) \approx p(\boldsymbol{x}_k|z_{1:k}) \tag{3.36}$$

donde la aproximación es en el sentido indicado por (3.28) y $\{\overline{w}^i\}$ son los pesos normalizados.

Aunque existen mejoras importantes a este algoritmo que mostraremos en las próximas secciones, podemos resumir el desarrollo hasta este punto en la descripción en pseudocódigo del cuadro 3.3. Puede verse la simplicidad de implementación de este algoritmo. Este es el principal motivo por el que este algoritmo ha alcanzado su gran popularidad para realizar un filtro recursivo bayesiano aproximado. El nombre con el que generalmente es conocido es el de *filtro de partículas*, denotando por partícula a cada una de las muestras o puntos de soporte utilizados.

$\left[\{\boldsymbol{x}_k^i, w_k^i\}_{i=1}^N\right] =$
$MUESTREO\_ENFATIZADO\_SECUENCIAL\left[\{\boldsymbol{x}_{k-1}^i, w_{k-1}^i\}_{i=1}^N, \boldsymbol{z}_k\right]$

- Para cada $i = 1 : N$
  - Saca $\boldsymbol{x}_k^i \sim q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^i, \boldsymbol{z}_k)$
  - Asigna w_k^i usando la expresión (3.35).

Cuadro 3.3: Algoritmo de muestreo enfatizado secuencial.

3.7. Degeneración del muestreo enfatizado secuencial

Un problema común en los filtros de muestreo enfatizado es el fenómeno conocido como *degeneración*. Dicho fenómeno consiste en que, tras unas pocas iteraciones del filtro, la mayor parte de las partículas pasan a tener unos pesos despreciables, de tal forma que únicamente unas pocas muestras son representativas. Esto es indeseable en varios niveles. En primer lugar afecta al número de partículas realmente disponibles a la hora de realizar la estimación. La calidad de la medida $\sum_{i=1}^N \overline{w}^i \delta_{\boldsymbol{x}_k^i}(\boldsymbol{x}_k)$, así como cualquier estimación basada en ella, disminuirá a la vez que más pesos van tendiendo a cero. Las partículas con pesos ínfimos no aportan información al estimador $\hat{I}(f)$ ni a la estimación de $\hat{p}_N(\boldsymbol{x}|\boldsymbol{z})$.

Por otro lado, estaremos consumiendo recursos computacionales en partículas que no dan ninguna información sobre la evolución del sistema. Ese esfuerzo computacional se malgastará en actualizar partículas cuya contribución a la aproximación de $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$ es prácticamente nula.

El origen principal de la degeneración es la diferencia entre $p(\boldsymbol{x}|\boldsymbol{z})$ y $q(\boldsymbol{x}|\boldsymbol{z})$. La evolución de $q(\boldsymbol{x}|\boldsymbol{z})$ puede llevar a las partículas a sitios muy diferentes a $p(\boldsymbol{x}|\boldsymbol{z})$ por lo que se obtendrán pesos ínfimos. Se puede demostrar, como aparece en [DdFG01], que la varianza de los pesos del filtro de partículas genérico, tal y como se muestra en el algoritmo 3.3, sólo puede aumentar con el tiempo, por lo que resulta imposible evitar el fenómeno de degeneración.

Una manera de medir el grado de degeneración de un filtro de partículas es ver cuántas partículas de la población inicial siguen siendo útiles. Para esto se utiliza el tamaño efectivo del conjunto de partículas, N_{eff}, introducido en [Ber99] y [LC98].

Éste se define como:

$$N_{eff} = \frac{N}{1 + var(w_k^{*i})} \tag{3.37}$$

donde $w_k^{*i} = p(\boldsymbol{x}_k^i|\boldsymbol{z}_{1:k})/q(\boldsymbol{x}_k^i|\boldsymbol{x}_{k-1}^i, \boldsymbol{z}_k)$ se refiere al "verdadero peso", definido en (3.15). Esta medida no se puede evaluar de manera exacta ya que es necesario calcular la varianza de los verdaderos pesos, que no se conocen. Sin embargo podemos estimar dicho valor de manera aproximada introduciendo $\hat{N}_{eff}$, que se define de la siguiente forma:

$$\hat{N}_{eff} = \frac{1}{\sum_{i=1}^{N_s}(w_k^i)^2} \tag{3.38}$$

donde w_k^i es el peso normalizado usando (3.35). Siempre se cumple que $N_{eff} \leq N$ y una baja N_{eff} indica una degeneración severa. El problema de la degeneración es claramente indeseable en cualquier filtro de partículas. Es necesario encontrar alguna forma de solventarlo.

La aproximación de fuerza bruta, para reducir el efecto pernicioso de esta degeneración, consiste en emplear un número muy grande de partículas. Esto no soluciona el problema, únicamente lo retrasa. En determinados casos podría ser suficiente, pero no es una solución real. Implica un alto coste computacional innecesario e inútil que, en general, no se puede aceptar.

Los filtros de partículas se caracterizan por ser una alternativa de bajo coste computacional comparados con otras alternativas, por lo que resulta imprescindible proporcionar métodos para solventar la degeneración. En particular, hay dos formas principales para abordar este problema:

- una buena elección de la función de propuesta q que minimice la degeneración
- la utilización de un proceso de remuestreo que corrija el problema de degeneración cuando éste sea insostenible.

La primera de las soluciones suele ser más difícil de implementar porque implica un conocimiento sobre la distribución *a posteriori* $p(\boldsymbol{x}|\boldsymbol{z})$ que puede que no se

tenga. En cambio, la segunda opción siempre será posible, aunque introduce nuevos problemas, como veremos a continuación.

3.8. Técnicas de remuestreo

El primer método que vamos a analizar para combatir los efectos perniciosos de la degeneración de pesos consiste en utilizar un sistema de remuestreo.

Partimos, como ya se ha mencionado, del hecho de que la degeneración del filtro de partículas es inevitable, siempre se va a producir. La degeneración consiste en una pérdida efectiva de partículas útiles. La formulación incremental del filtro de partículas emplea los pesos en un instante de tiempo dado para calcular los pesos en el instante siguiente. De manera intuitiva podemos ver que si un peso w_{k-1}^i es bajo comparado con los demás, será difícil que se recupere significativamente en $t = k$. Esto provoca una perniciosa tendencia, que desemboca en una pérdida efectiva de esa partícula. Este efecto se sigue produciendo con el resto de partículas, llevando al filtro a una situación degenerada.

La idea básica del remuestreo consiste en detener esta tendencia perniciosa, eliminando aquellas partículas cuya evolución ha llevado a obtener pesos bajos y que ya no proporcionan información útil. En cambio, se concentra en aquellas que tienen pesos más altos y por lo tanto contribuyen en mayor medida en el estimador $\hat{I}(f)$.

Una forma de recuperarlas, sin perder el formalismo que estamos usando, consiste en volver a obtener un conjunto de partículas independientes de $p(\boldsymbol{x}_k|\boldsymbol{z}_k)$. De esta manera, al tomar las partículas de la verdadera distribución de probabilidad, todos los pesos se normalizarán al mismo valor, $w_k^i = 1$.

Como ya se ha comentado, la distribución $p(\boldsymbol{x}_k|\boldsymbol{z}_k)$ es demasiado compleja como para muestrear de ella directamente. Este es el principal motivo por el que usamos muestreo enfatizado. Sin embargo tenemos una distribución estimada, formada por las partículas: $\hat{p}_N(\boldsymbol{x}_k|\boldsymbol{z}_k)$.

En teoría cualquier conjunto de muestras tomadas independientemente de $p(\boldsymbol{x}_k|\boldsymbol{z}_k)$ sería igualmente válido para formar este estimador. Si se obtiene de manera independiente de $p(\boldsymbol{x}_k|\boldsymbol{z}_k)$, sus pesos tendrán varianza mínima. El paso de remuestreo implica obtener este conjunto totalmente nuevo de partículas $\{x_k^{i*}\}_{i=1}^{N_s}$, muestreando

(con remplazamiento) de $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$ N veces. En nuestro caso la mejor representación disponible de $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$ será la aproximación discreta formada por las propias partículas, siguiendo la siguiente expresión:

$$\hat{p}_N(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) = \sum_{i=1}^{N} w_k^i \delta(\boldsymbol{x}_k - \boldsymbol{x}_k^i) \tag{3.39}$$

de tal forma que

$$Pr(\boldsymbol{x}_k^{i*}) = w_k^i \tag{3.40}$$

La muestra resultante es de hecho una muestra independiente de la densidad discreta (3.39). Por lo tanto los pesos se ajustan ahora como $w_k^i = 1/N$, es decir, de manera uniforme. Como vemos, el paso de remuestreo evita que los pesos degeneren, construyendo un nuevo conjunto de muestras donde todas tienen el mismo peso. Si nos fijamos en la expresión (3.40), el paso de remuestreo tiende a multiplicar aquellos estados con pesos significativos mientras que descarta aquéllos con pesos bajos. En este sentido podemos interpretar el paso de remuestreo como una focalización de nuestra capacidad de atención. Fijaremos esta atención, es decir, las partículas, en las zonas más prometedoras del espacio de estados siguiendo nuestra estimación actual de la función de distribución de probabilidad.

El remuestreo que hemos definido se puede implementar en $O(N)$ operaciones muestreando de una distribución uniforme N veces. En el cuadro 3.4 puede verse uno de los algoritmos de remuestreo más usados, también conocido como remuestreo por ruleta.

Existen otros métodos de remuestreo eficientes, en términos de la reducción de la varianza de Monte Carlo, como el muestreo estratificado y el muestreo residual [LC98], que no trataremos aquí. En este trabajo nos centraremos en el uso del algoritmo de la ruleta para implementar el remuestreo cuando sea necesario.

El algoritmo de remuestreo se puede aplicar cada vez que el conjunto de los pesos llegue a un umbral alto de degeneración. Para medir la degradación podemos usar $\hat{N}_{eff}$, según la expresión (3.38). Combinando el algoritmo del muestreo enfatizado secuencial y el paso de remuestreo podemos escribir el algoritmo para el filtro de

$[\{\boldsymbol{x}_k^{j*}, w_k^i, i^j\}_{j=1}^N] = REMUESTREO\left[\{\boldsymbol{x}_k^i, w_k^i\}_{i=1}^N\right]$

- Inicializa $c_1 = 0$.
- Para $i = 2 : N$
  - $c_i = c_{i-1} + w_k^i$
- Comienza con $i = 1$
- Obtén un punto de comienzo $u_1 \sim U[0, N^{-1}]$
- Para $j = 1 : N$
  - Desplázate por la función de densidad acumulada: $u_j = u_1 + N^{-1}(j-1)$
  - Mientras $u_j > c_i$
    - i = i + 1
  - Asigna la partícula: $\boldsymbol{x}_k^{j*} = \boldsymbol{x}_k^i$
  - Asigna el peso: $w_k^j = N^{-1}$
  - Asigna el parentesco: $i^j = i$

Cuadro 3.4: Algoritmo de remuestreo por ruleta.

partículas genérico, que se muestra en el cuadro 3.5.

Aún cuando el remuestreo alivia los efectos de la degeneración de los pesos, introduce nuevos problemas [AMGC02, Dou98]. En primer lugar, desde un punto de vista práctico, el remuestreo limita la capacidad de paralelizar el algoritmo del filtro de partículas, dado que es necesario sumar todos los pesos w_k^i durante el paso de normalización. En segundo lugar, desde una perspectiva teórica, las muestras $\boldsymbol{x}_k^i$ no serán estadísticamente independientes tras el remuestreo. Siendo así, perdemos los resultados de convergencia que se discutieron en la sección 3.5. No obstante, bajo unas suposiciones genéricas, todavía es posible garantizar una convergencia casi segura de las distribuciones empíricas generadas por el filtro de partículas hacia las verdaderas, a pesar de que las demostraciones se vuelven más complejas [CD02, Cri01].

Por ahora destacaremos el problema del *empobrecimiento de las partículas*. Este

$\{\boldsymbol{x}_k^i, w_k^i\}_{i=1}^N = FILTRO\_PARTICULAS\left[\{\boldsymbol{x}_{k-1}^i, w_{k-1}^i\}_{i=1}^N, \boldsymbol{z}_k\right]$

1. Para cada $i = 1 : N$
   - Obtén una nueva $\boldsymbol{x}_k^{'i} \sim q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^i, \boldsymbol{z}_k)$.
   - Calcula w_k^i usando la expresión (3.35)
2. Para cada $i = 1 : N$
   - Calcula el peso normalizado: $\overline{w}_k^i = \frac{w_k^i}{\sum_{j=1}^N w_k^j}$
3. Calcula $\hat{N}_{eff}$ usando la expresión (3.38)
4. Si $\hat{N}_{eff} < N_{umbral}$
   - $\{x_k^i, w_k^i\}_{i=1}^N = REMUESTREO\left[\{x'^i_k, \overline{w}_k^i\}_{i=1}\right]$

   En caso contrario
   - $\{x_k^i, w_k^i\}_{i=1}^N = \{x'^i_k, \overline{w}_k^i\}_{i=1}$

Cuadro 3.5: Algoritmo para el filtro de partículas genérico.

empobrecimiento ocurre cuando un estado $\boldsymbol{x}_k^i$ (o de manera equivalente, una secuencia de estados $\boldsymbol{x}_{0:k}^i$) con un peso alto, se selecciona muchas veces durante el paso de remuestreo. Esto se traduce en una pérdida de diversidad entre los puntos soporte, disminuyendo la independencia entre ellos. A pesar de que en la siguiente iteración del filtro se "vuelva a diversificar" el conjunto de puntos $\{x_{k+1}^i\}$, el empobrecimiento puede tener efectos muy perjudiciales, perdiendo información valiosa que puede no volver a recuperarse.

Sin embargo, a pesar de estos tres problemas, la necesidad de prevenir el pernicioso efecto de la degeneración de los pesos en el filtro de partículas es más importante y los métodos de remuestreo se han convertido en un componente de casi todas las implementaciones del filtro de partículas.

3.9. Muestreo enfatizado secuencial con remuestreo

Al usar el remuestreo debemos llegar a un compromiso sobre el número de veces que lo aplicamos. Al aplicarlo se reduce de la degeneración de los pesos, volviendo a una situación de varianza mínima de los mismos, en la que todas las partículas son igualmente significativas. Desgraciadamente, al mismo tiempo, se pierde variedad entre los puntos soporte del filtro.

Esta afirmación contrasta con el que es, sin duda alguna, el filtro de partículas más conocido que emplea el remuestreo en todas y cada una de la iteraciones del filtro. Dicho filtro es el muestreo enfatizado con remuestreo (del inglés *Importance Sampling Resampling* o SIR) [Rub98], también conocido con el nombre de filtro de condensación [IB98a]. Dicho filtro ha tenido un gran éxito en los últimos años gracias, sobre todo, a dos factores:

- realiza unas suposiciones muy fáciles de cumplir y
- es muy sencillo de programar.

Su éxito ha sido tal que en muchos casos se ha considerado como máximo exponente de la familia de los filtros de partículas, eclipsando otras variantes.

El muestreo enfatizado con remuestreo parte del conocimiento de la dinámica del estado y de la función de medida, funciones $f_k(\cdot,\cdot)$ y $h_k(\cdot,\cdot)$ en (3.3) y (3.4) respectivamente. Además, debemos ser capaces de muestrear de la distribución del proceso de ruido v_k y de la distribución *a priori*. Por último, la función de verosimilitud $p(\boldsymbol{z}_k|\boldsymbol{x}_k)$ debe poder ser evaluada en un punto arbitrario hasta una constante de proporcionalidad.

Este algoritmo se puede derivar fácilmente del algoritmo general de muestreo enfatizado secuencial, que introdujimos anteriormente. Para ello únicamente es necesario la correcta elección de dos factores:

1. La función de propuesta. En este caso $q(\boldsymbol{x}_k|\boldsymbol{x}^i_{k-1}, \boldsymbol{z}_{1:k})$ se elige de la forma de la distribución *a priori* $p(\boldsymbol{x}_k|\boldsymbol{x}^i_{k-1})$, lo que simplifica tanto la extracción de nuevas partículas como el cálculo de los pesos.

$\left[\{\boldsymbol{x}_k^i, w_k^i\}_{i=1}^N\right] = CONDENSACION\left[\{\boldsymbol{x}_{k-1}^i, w_{k-1}^i\}_{i=1}^N, \boldsymbol{z}_k\right]$

- Para cada $i = 1 : N$
  - Saca $\boldsymbol{x}_k^i \sim p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^i)$
  - Calcula $w_k^i = p(\boldsymbol{z}_k|\boldsymbol{x}_k^i)$
- Calcula la suma total de los pesos: $t = \sum_{i=1}^N w_k^i$
- Para cada $i = 1 : N$
  - Normaliza $w_k^i = t^{-1} w_k^i$
- $\left[\{x_k^i, w_k^i, -\}_{i=1}^N\right] REMUESTREO\left[\{x_k^i, w_k^i, -\}_{i=1}^N\right]$

Cuadro 3.6: Filtro de condensación.

2. El paso de remuestreo. En este caso se aplica en todas las iteraciones del algoritmo un remuestreo por ruleta.

El pseudocódigo para este algoritmo se encuentra en el cuadro 3.6. Su funcionamiento es un proceso continuo de elección de puntos prometedores, diversificándolos posteriormente para buscar nuevos puntos interesantes. En la figura 3.2 se muestra gráficamente este proceso. Se parte de un conjunto de partículas de la iteración anterior. Se calcula el peso de cada una de estas partículas en función de la distribución $p(\boldsymbol{z}_k|\boldsymbol{x}_k)$. Estos pesos se emplean durante el remuestreo, colocan un número mayor de muestras en las zonas con mayores pesos. Los nuevos puntos se diversifican usando la distribución $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$. Este proceso se repite iterativamente, colocando las partículas en las zonas de interés.

Esta elección de la función de propuesta implica que debemos ser capaces de muestrear de $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^i)$. Las muestras $x_k^i \sim p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^i)$ pueden obtenerse generando en primer lugar una muestra del proceso de ruido $v_k^i \sim p_v(v_k)$ y ajustando $x_k^i = f_k(\boldsymbol{x}_k^i, v_k^i)$, donde $p_v(\cdot)$ es la función de densidad de probabilidad de v_k. Para esta elección en particular de la función de propuesta resulta evidente que los pesos se obtienen simplificando 3.33 de la siguiente forma:

$$w_k^i \propto w_{k-1}^i p(\boldsymbol{z}_k|x_k^i) \tag{3.41}$$

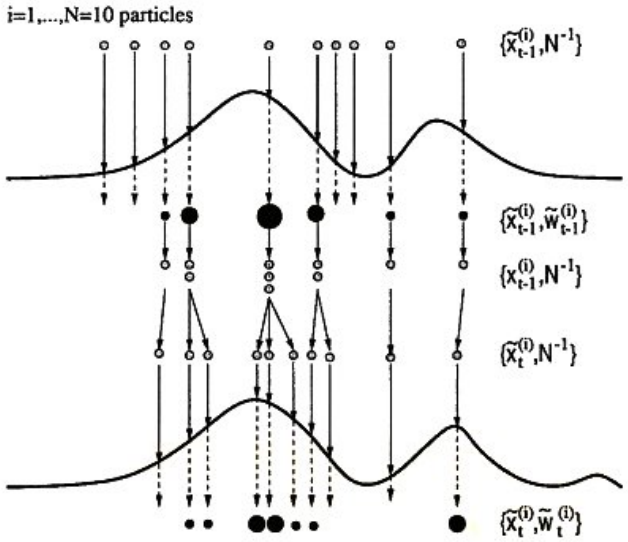

Figura 3.2: Evolución del filtro de condensación. La imagen pertenece a [IB98a].

Sin embargo, como se aplica el remuestreo en todas las iteraciones del algoritmo, tenemos que $w_{k-1}^i = 1/N \; \forall i$; y por lo tanto:

$$w_k^i \propto p(\boldsymbol{z}_k | x_k^i) \tag{3.42}$$

Como la función de propuesta para este filtro es independiente de la medida actual $\boldsymbol{z}_k$, el espacio de estados se explora sin ningún conocimiento sobre la observación. Esta exploración ciega puede resultar muy ineficiente y sensible a medidas espurias. Por otro lado, como el proceso de remuestreo se aplica en cada paso del algoritmo, puede darse el caso en que el efecto del empobrecimiento del conjunto de las partículas se maximice. Sin embargo, este filtro tiene como clara ventaja el hecho de que los pesos se calculan de manera muy sencilla y que es fácil muestrear de la función de propuesta.

3.10. Función de propuesta

Una elección adecuada de la función de propuesta puede ser otra alternativa para mitigar los perniciosos efectos de la degeneración de los pesos en el filtro de partículas. Aunque la aparición de la degeneración es inevitable, como ya hemos mencionado en la sección 3.7, la correcta elección de la función $q(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k})$ retrasará la aparición de la misma y limitará el número necesario de remuestreos.

Llegados a este punto la pregunta que nos surge es qué función elegir y cómo hacerlo.

Al introducir el algoritmo de muestreo enfatizado en la sección 3.5 se habló de que casi cualquier función podía emplearse como función de propuesta. Hasta ahora la única restricción que habíamos puesto a su elección era que el soporte de la función de propuesta debía incluir el soporte de la verdadera distribución *a posteriori* $p(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k})$. Al hablar de la formulación secuencial del filtro se añadió la restricción de que la función de propuesta pudiera factorizarse de la forma

$$q(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k}) = q(\boldsymbol{x}_0)\prod_{j=1}^{k} q(\boldsymbol{x}_j|\boldsymbol{x}_{j-1}, \boldsymbol{z}_j) \quad \forall k \tag{3.43}$$

Se simplificaba así la formulación secuencial del filtro de muestreo enfatizado (sección 3.6). Aparte de estos requisitos, no hemos añadido ninguna restricción en la elección de una función de propuesta a la hora de resolver el problema de filtrado bayesiano.

Sea cual sea la elección de la función de propuesta, siempre respetando las dos restricciones mencionadas, el sistema debe funcionar según la ley de los grandes números y dar un estimador correcto. Sin embargo, dicha ley también establece que los resultados se aplican cuando $N \to \infty$. Así que si queremos obtener unas prestaciones aceptables para los casos en los que no usamos un número infinito de partículas, debemos poner algo más de cuidado en la elección de la función de propuesta $q(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k})$.

Algunas funciones de propuesta serán, obviamente, mejores que otras. Un criterio interesante a la hora de elegir una u otra es escoger aquella que minimice la varianza condicionada de los pesos del estimador $\hat{I}_N(f)$. Este estimador tiene la siguiente forma

$$\hat{I}_N(f) = \sum f(\boldsymbol{x}_k) w_k(\boldsymbol{x}_k) \tag{3.44}$$

Si reducimos la varianza del producto $f(\boldsymbol{x}_k)w_k(\boldsymbol{x}_k)$ estaremos repartiendo los pesos de manera adecuada para la función f. La varianza de $f(\boldsymbol{x})w(\boldsymbol{x})$ con respecto a $q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{z})$ viene dada por:

$$var_{q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1},\boldsymbol{z}_k)}\left[f(\boldsymbol{x}_k)w(\boldsymbol{x}_k))\right] = E_{q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1},\boldsymbol{z}_k)}\left[(f^2(\boldsymbol{x}_k)w^2(\boldsymbol{x}_k)) - I^2(f)\right] \tag{3.45}$$

donde hemos utilizado la distribución de propuesta $q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{z}_k)$ obtenida de la formulación incremental tal y como aparece en la expresión 3.43.

El segundo término de la derecha es independiente de $q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{z}_k)$ y por lo tanto sólo resulta necesario minimizar el primer término. Para ello podemos emplear la desigualdad de Jensen. Dicha desigualdad nos dice que si g es una función cóncava y x una variable aleatoria, entonces:

$$E(g(x)) \geq g(E(x)) \tag{3.46}$$

Aplicando dicha desigualdad a la anterior expresión, podemos obtener el siguiente límite inferior:

$$E_{q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1},\boldsymbol{z})}\left[f^2(\boldsymbol{x}_k)w^2(\boldsymbol{x}_k))\right] \geq \left(E_{q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1},\boldsymbol{z}_k)}\left[|f(\boldsymbol{x}_k)|w(\boldsymbol{x}_k)\right]\right)^2 \tag{3.47}$$

Aplicando la expresión incremental de los pesos

$$w_k = w_{k-1}\frac{p(\boldsymbol{z}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})}{q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{z}_k)} \tag{3.48}$$

podemos obtener el límite inferior de la expresión 3.47:

$$\left(E_{q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1},\boldsymbol{z}_k)}\left[|f(\boldsymbol{x}_k)|w(\boldsymbol{x}_k)\right]\right)^2 = \left(w_{k-1}\int |f(\boldsymbol{x}_k)|p(\boldsymbol{x}_k,\boldsymbol{z}_k|\boldsymbol{x}_{k-1})d\boldsymbol{x}_k\right)^2 \tag{3.49}$$

Este límite inferior se alcanza cuando usamos la siguiente función de propuesta óptima [AFDJ03]:

$$q_{opt}(\boldsymbol{x}_k|\boldsymbol{x}_{k-1},\boldsymbol{z}_k) = \frac{|f(\boldsymbol{x}_k)|p(\boldsymbol{x}_k,\boldsymbol{z}_k|\boldsymbol{x}_{k-1})}{\int |f(\boldsymbol{x}_k)|p(\boldsymbol{x}_k,\boldsymbol{z}_k|\boldsymbol{x}_{k-1})d\boldsymbol{x}_k} \tag{3.50}$$

La función óptima no resulta muy útil, en el sentido de que tiene un problema grave que impide su aplicación práctica. Para usar esta función es necesario que seamos capaces de muestrear directamente de $|f(\boldsymbol{x}_k)|p(\boldsymbol{z}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$, que puede ser una distribución genérica de la que resulte difícil sacar muestras. Precisamente uno de los motivos para usar un muestreo enfatizado era que resultaba difícil muestrear directamente de $p(\boldsymbol{x}_k|\boldsymbol{x}^i_{k-1},\boldsymbol{z}_k)$.

Sin embargo, sí que resulta útil dado que nos indica como se puede conseguir una gran eficiencia de muestreo para el estimador $\hat{I}(f)$. Al muestrear se puede hacer un especial énfasis en aquellas zonas que resultan interesantes, aquéllas que tienen un valor de $|f(\boldsymbol{x})|p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1},\boldsymbol{z}_k)$ relativamente grande. De ahí viene el nombre de muestreo enfatizado, pues hace énfasis en esas zonas.

Este resultado implica que las estimaciones del muestreo enfatizado pueden resultar *super-eficientes*. Es decir, para una función determinada $f(\boldsymbol{x})$, resulta posible encontrar una distribución $q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1},\boldsymbol{z}_k)$ que produzca una estimación con una varianza menor que usando un método de Monte Carlo ideal, es decir cuando $q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1},\boldsymbol{z}_k) = p(\boldsymbol{x}|\boldsymbol{x}_{k-1},\boldsymbol{z}_k)$. Esta propiedad se suele emplear, por ejemplo, para evaluar la probabilidad de eventos poco comunes en redes de comunicación.

Cuando se proponen candidatos en aquellas regiones que no tienen utilidad, resulta un completo desperdicio a la hora de evaluar la estimación $\hat{I}_N(f)$. En muchas aplicaciones el objetivo puede ser diferente en el sentido de que lo que se busca es obtener una buena aproximación de $p(\boldsymbol{x})$ y no de una integral en concreto con respecto a $p(\boldsymbol{x})$. En estos casos lo que se intenta buscar es que $q(\boldsymbol{x}) \approx p(\boldsymbol{x})$. Se puede definir otros criterios a la hora de buscar una q adecuada según nuestra aplicación en

concreto. En [RC01, OTF$^+$04, TFB00] se muestran algunas alternativas.

Con esto concluimos el capítulo dedicado a los fundamentos matemáticos del filtro de partículas. En este capítulo hemos realizado un repaso al filtrado bayesiano óptimo, viendo las limitaciones que éste tiene. Dentro de las aproximaciones a éste filtro nos hemos centrado en los métodos basados en técnicas de Monte Carlo. Éstos emplean números aleatorios para representar distribuciones de probabilidad. En los siguientes capítulos haremos uso principalmente de dos de estas técnicas: el filtro de condensación y el muestreo enfatizado.

CAPÍTULO 4

Seguimiento De Un Objeto

El objetivo de esta tesis, como se expuso en la sección 1.5, es la estimación de la posición 3D de varios objetos al mismo tiempo usando para ello únicamente información visual. El funcionamiento debe ser eficiente, por lo que es necesario hacer un énfasis en la rapidez de los algoritmos y las técnicas empleadas. Podemos resumir nuestro objetivo como localización y seguimiento 3D visual eficiente.

Interpretamos el seguimiento como un caso particular de un problema más general de estimación, dentro del campo de la estadística. Usando la estimación, nuestro objetivo se puede reducir a proporcionar un estimador que utilice como entrada las imágenes de varias cámaras y que como salida obtenga los valores estimados de la posición de todos los objetos que estén en la escena.

En este trabajo nos hemos decantado por la utilización de filtros de partículas. Una de sus ventajas es que no tiene límite en cuanto a las distribuciones *a posteriori* que pueden representar, por lo que parece más adecuada para nuestros objetivos. A parte de sus ventajas teóricas y de su sencillez de implementación, el principal motivo que nos ha llevado a usarlos viene de la idea principal en la que se sustenta esta tesis. En vez de abordar la construcción directa de soluciones 3D para seguir un objeto, resulta más sencillo realizar múltiples hipótesis de dónde puede encontrarse ese objeto y comprobar lo acertado de cada una de ellas, usando la información que se toma de

la realidad. Esto combinado con la utilización de modelos visuales sencillos, como se expuso en el capítulo 1, busca una reducción en el coste computacional que permita seguir a varios objetos a la vez.

Para desarrollar el sistema de seguimiento empleando filtros de partículas, vamos a reformular los objetivos de la tesis atendiendo a la naturaleza de estos métodos. En nuestro caso queremos adaptar un filtro de partículas para el problema particular del seguimiento en tres dimensiones, proponiendo mejoras en el algoritmo para trabajar con varios objetos indistinguibles entre sí a la vez. Este problema no se encuentra resuelto, aunque existen varias aproximaciones al mismo, como las propuestas en [KKH03, KBD05]. Nuestra aproximación hará uso de la *función de propuesta* como método para mantener una distribución *a posteriori* capaz de mostrar varios modos al mismo tiempo.

Antes de encontrar una solución capaz de seguir múltiples objetos, debemos ser capaces de seguir un único objeto, que es el objetivo de este capítulo. Plantearemos el problema de seguimiento de un único objeto, presentado formas eficientes de hacerlo, con diferentes modificaciones del filtro de partículas. Una vez mostradas las diferentes propuestas veremos cuáles son las limitaciones de las mismas para seguir múltiples objetos al mismo tiempo. En el siguiente capítulo se intentarán solucionar los problemas aquí descritos.

4.1. Seguimiento 3D y espacio de estados

Las técnicas de seguimiento empleadas se basan en la definición de un *espacio de estados*. Cada posición de dicho espacio, cada punto, representa una solución completa para el problema al que nos enfrentemos. Para nuestro problema en particular sólo estamos interesados en la posición del objeto, por lo que cuando hablemos del estado del sistema nos referiremos a esta posición. La información necesaria es únicamente la de las coordenadas x, y, z del objeto, pero podría darse el caso de que no fuera así. Variables como la velocidad en cada eje, la velocidad angular o la orientación, por citar algunas, pueden proporcionar información muy relevante para la correcta descripción del sistema. Incluso es posible que al añadir esta nueva información resulte más sencillo resolver el problema de seguimiento. Sin embargo, en

esta tesis queremos hacer hincapié en el seguimiento de múltiples objetos, por lo que no nos centraremos en la elección del espacio de estados.

Aún así haremos un pequeño resumen de las aportaciones más relevantes en cuanto a la elección de un correcto espacio de estados. Cualquier información que nos permita definir el sistema o ayudar en su seguimiento, debe incluirse en el estado según el formalismo bayesiano. De esta forma, las partículas son soluciones completas al problema, autocontenidas. Debe notarse que la elección del espacio de estados tiene implicaciones importantes tanto en el modelo de movimiento como en el modelo de observación, dado que ambos trabajan sobre $\boldsymbol{x}_k$. El modelo de movimiento debe adecuarse para relacionar los nuevos estados $\boldsymbol{x}_k$ en función de $\boldsymbol{x}_{k-1}$. Debemos relacionar la posición actual del objeto con la posición en el siguiente instante de tiempo. Una de las ventajas de añadir más información al espacio de estados es que podemos perfeccionar el modelo de movimiento. Por ejemplo, en vez de usar únicamente la posición puntal x, y, z podemos añadir las velocidades en cada eje v_x, v_y, v_z. El modelo de movimiento podrá basarse en expresiones de cinemática básica. Esto proporcionará mejores estimaciones para un objeto móvil que simplemente la posición puntual, siempre y cuando la estimación de la velocidad sea adecuada.

La información que se incluye en el espacio de estados no está limitada únicamente a la velocidad. Pueden añadirse otras variables interesantes a la hora de aplicar el modelo de movimiento. Por ejemplo, en [IB98c] se emplea una variable extra en el espacio de estados para referenciar el tipo de movimiento que tiene el objeto. De esta forma, es sencillo combinar diferentes modelos de movimiento controlados por la propia evolución de la partícula. En particular [IB98c] emplean dos modelos, uno para movimientos bruscos y otro para movimientos suaves. Cada partícula puede tener su propio movimiento asociado, independiente al que siga el resto, y así, reaccionar mejor a cambios bruscos en la dinámica del objeto.

Otro posible uso de un espacio de estados mayor es el seguimiento de objetos articulados [DBR00]. Al enfrentarnos a este tipo de objetos no es suficiente conocer su posición sino que también debemos conocer cómo están articulados, cuál es su forma. Esto tiene dos motivos principales. Por un lado puede que estemos interesados en obtener realmente la forma que ha adquirido el objeto, además de su posición. Por otra parte si no conocemos la forma del objeto puede que no seamos capaces de obte-

ner un modelo de observación preciso para el mismo, lo que implicaría una reducción de las prestaciones del sistema de seguimiento. Otra consideración que es necesario hacer al aumentar el tamaño del espacio de estados es la necesidad de emplear un número superior de partículas para seguir los objetos. En [DBR00] también se proporciona un método para elegir el número de partículas en función de la dimensión del espacio de estados.

Por último, podemos añadir otra utilidad más: el seguimiento de múltiples objetos. Al aumentar el espacio de estados podemos añadir la posición de todos y cada uno de los objetos de la escena de manera explícita [KKH03, KBD05]. De esta forma, el estado contiene toda la información referente a la escena, a todos los objetos. Este método constituye la manera más directa de obtener estimación de múltiples objetos con los filtros de partículas, pero como vimos en la sección 2.3.1 no está exenta de problemas. El principal de todos, que aparece también para el seguimiento de objetos articulados, es la pérdida de eficacia de los filtros de partículas según aumenta el número de dimensiones del espacio de estados.

En nuestro caso no emplearemos un espacio de estados como éstos, sino únicamente la posición tridimensional. El estado $\boldsymbol{x}_k$ tendrá la información de la posición actual (x, y, z) en el instante k. Esta aproximación, aunque más limitada, resulta suficiente para poder apreciar el problema de la pérdida de variedad en las partículas, responsable del mal funcionamiento del seguimiento de múltiples objetos. Este es justo el problema que pretendemos atacar. Las observaciones $\boldsymbol{z}_k$, en nuestro, son un conjunto de imágenes capturadas en el instante k.

Usando la notación que introdujimos en la sección 3.1 podemos formalizar nuestro objetivo de la siguiente forma: queremos encontrar la mejor estimación posible de la posición del objeto, $\hat{\boldsymbol{x}}_k$, en todo instante k. Para ello podemos usar la última estimación disponible, $\hat{\boldsymbol{x}}_{k-1}$ y las observaciones obtenidas $\boldsymbol{z}_k$. La distribución buscada será, por tanto, $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{z}_k)$.

$\left[\{\boldsymbol{x}_k^i, w_k^i\}_{i=1}^N\right] = CONDENSACION\left[\{\boldsymbol{x}_{k-1}^i, w_{k-1}^i\}_{i=1}^N, \boldsymbol{z}_k\right]$

- Para cada $i = 1 : N$
  - Genera un número aleatorio $r \in [0, 1]$ distribuido uniformemente.
  - Encuentra por división binaria el índice j más pequeño que cumpla $c_{t-1}^j \geq r$.
  - $x_t^i = x_{t-1}^j$
  - Obtiene $\boldsymbol{x}_k^i \sim p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^i)$
  - Calcula $w_k^i = p(\boldsymbol{z}_k|\boldsymbol{x}_k^i)$
- Calcula la suma total de los pesos: $t = \sum_{i=1}^N w_k^i$
- Para cada $i = 1 : N$
  - Normaliza $w_k^i = t^{-1} w_k^i$
  - $c_t^i = c_{t-1} + w_k^i$, con $c_0 = 0$
- Estima $\hat{\boldsymbol{x}}_k$: $\hat{\boldsymbol{x}}_k = \sum_{i=0}^{N-1} w_k^i \boldsymbol{x}_k^i$

Cuadro 4.1: Algoritmo de condensación detallado.

4.2. Filtro de condensación para el seguimiento de un objeto

Como hemos visto, el filtro de partículas es una buena herramienta para realizar estimaciones y puede resultar útil para el problema de seguimiento. Existen varias aproximaciones a este problema que ya han utilizado el filtro de partículas. Quizá la más conocida de todas sea el filtro condensación de Isard y Blake [IB98a], pero en el capítulo 2 se mencionan otras alternativas igualmente válidas.

En el capítulo 3 hemos presentado una introducción en profundidad a este tipo de técnicas. Como hemos comentado, los filtros emplean una serie de puntos soporte, conocidos como partículas, para estimar la distribución *a posteriori*. Todas las partículas son independientes entre sí, por lo que podemos garantizar que la aproximación a la distribución *a posteriori* será correcta y convergerá hacia la verdadera distribución según aumente el número de partículas.

El filtro de partículas sigue un funcionamiento iterativo, cuyos pasos se resumen en el cuadro 4.1.

Debe notarse en este punto que el hecho de buscar la mejor estimación posible para la posición actual, $\boldsymbol{x}_k$, es sustancialmente idéntico al de encontrar una correcta estimación de la distribución de probabilidad $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{z}_k)$. A partir del conocimiento de la distribución de probabilidad será posible obtener una correcta estimación de $\boldsymbol{x}_k$. Los métodos de Monte Carlo, base sobre la que se construyen los filtros de partículas, son técnicas especialmente indicadas para obtener una representación correcta de $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{z}_k)$. A la hora de afrontar el problema del seguimiento multiobjeto haremos especial uso de la capacidad de representación de distribuciones de probabilidad de estos métodos. Hasta ese momento consideraremos las partículas como hipótesis de la posición del único objeto existente independientes entre sí. La combinación adecuada de dichas hipótesis, como se vio en el capítulo anterior, permite obtener una estimación de la distribución de probabilidad *a posteriori* $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{z}_k)$.

Al observar los pasos necesarios para aplicar el filtro de partículas al seguimiento visual se hacen evidentes sus ventajas. Por un lado, el proceso es completamente inductivo, en lugar en construir de manera directa la posición 3D del objeto se hipotetiza sobre el mismo y se comprueba si dichas hipótesis son correctas o no en las observaciones siguientes. Esto supone, en problemas como el de visión, un gran ahorro de cómputo dado que la comprobación de si un objeto está o no en una posición dada, resulta más liviana que la construcción directa de la solución. Dicha construcción implicaría, como mínimo, una segmentación, un emparejamiento y un proceso de triangulación. Usando este método lo único necesario es obtener el modelo de movimiento y el de observación. Estas tareas, como veremos a continuación, pueden resultar sencillas.

4.3. Modelo de movimiento

El modelo de movimiento es el encargado de relacionar el último estado conocido con el estado actual. Para representarlo podemos usar una función de distribución de probabilidad del tipo $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$. En este modelo es dónde reside nuestro cono-

cimiento sobre la dinámica del sistema. La correcta elección del mismo hará que el sistema funcione o no. Si las predicciones acerca de la nueva posición del objeto son incorrectas las partículas abandonarán las zonas probables de $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{z}_k)$ o en caso de estar en ellas sus pesos no serán los adecuados.

Analicemos en primer lugar cuáles deben de ser las características del modelo. Debemos recordar que a la hora de aplicar el filtro de partículas hemos supuesto que el sistema puede modelarse como un modelo de Markov de orden uno. Este hecho simplifica de manera importante el modelo de movimiento sin perder por ello demasiada generalidad. En el problema que queremos abordar, el seguimiento de objetos físicos, parece razonable suponer que la trayectoria pasada condiciona la posición en el siguiente instante de tiempo. Resulta suficiente suponer únicamente dependencia de la última posición. De esta forma, estaremos ignorando la trayectoria pero aplicando un principio de continuidad en el movimiento.

Uno de los problemas a la hora de elegir un modelo de movimiento es la necesidad de obtener información *a priori*. Cuanto más conozcamos acerca de los posibles movimientos del objeto, de su dinámica, más preciso será el modelo y, probablemente, mejores resultados obtendremos. Pensemos, por ejemplo, en el movimiento de un péndulo. Si conocemos la trayectoria resulta trivial acotar las posibles hipótesis para el siguiente instante de tiempo. Esto nos dará mucha información, simplificando el trabajo del filtro de partículas. Las hipótesis hechas serán más precisas y obtendrán mejores pesos, lo que lleva a una mejor estimación de la posición del objeto. Esto se debe a que la mayor parte de las partículas tendrán pesos correctos y estarán colocadas en las partes interesantes de $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{z}_k)$, en sus máximos.

Introducir demasiada información en el modelo de movimiento podría limitar la generalidad del sistema de seguimiento. En el caso de aplicarlo a otro entorno o de que la dinámica del movimiento cambie súbitamente, el sistema fallará.

Uno de los objetivos que propusimos fue realizar un seguimiento genérico, sin demasiado conocimiento *a priori* sobre cómo podría moverse el objeto. Por este motivo el modelo de movimiento elegido debe ser lo más general posible, sin hacer suposiciones que no resulten completamente necesarias, aún perdiendo precisión en algún caso.

El modelo de movimiento que hemos elegido tiene forma de gaussiana isótro-

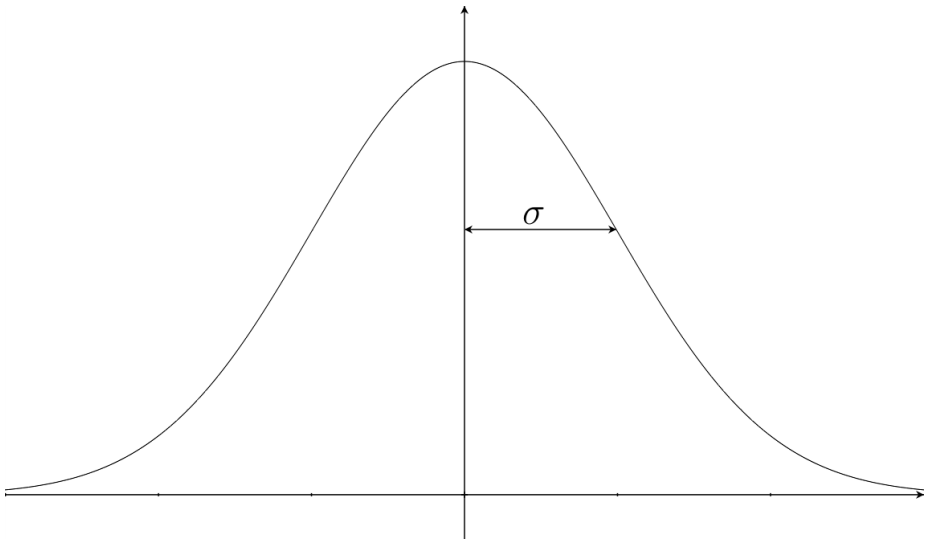

Figura 4.1: Modelo de movimiento.

pa. Este modelo, como puede verse en la figura 4.1, marca como más probables las zonas alrededor de la última posición conocida. Cualquier dirección de movimiento es igualmente probable, de ahí la naturaleza isótropa. La expresión del modelo es la siguiente:

$$p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) = \frac{1}{(2\pi)^{3/2}\left|\Sigma\right|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}_k - \boldsymbol{x}_{k-1})^\top \Sigma^{-1}(\boldsymbol{x}_k - \boldsymbol{x}_{k-1})\right) \qquad (4.1)$$

donde Σ la matriz de covarianza diagonal. Dicha matriz será de la forma $\Sigma = \sigma I$, donde σ es un parámetro que debemos ajustar y I es la matriz identidad.

En principio este sistema es lo suficientemente general, dado que no tiene direcciones privilegiadas. Lo único que expresa es una continuidad física en el movimiento: entre dos instantes de tiempo el objeto se encontrará en zonas del espacio próximas entre sí. Por ejemplo, en el seguimiento de una persona no sabemos hacia dónde se dirige. Es posible que cambie de forma abrupta su trayectoria, pero sí podemos suponer que su posición no cambiará sustancialmente entre un fotograma y el siguiente.

La distancia máxima entre dichas posiciones que identificaremos como parte del

movimiento viene marcada por la anchura de la gaussiana del modelo de movimiento, controlada por el parámetro σ. Dicho de otra forma, este parámetro marcará la velocidad máxima a la que podemos seguir un objeto, pero también tendrá repercusiones en el error residual de la estimación.

Durante las iteraciones del filtro de partículas se seleccionan nuevas muestras haciendo uso del modelo de movimiento. Si la anchura de la gaussiana usada es grande, las nuevas partículas se separarán mucho de la posición actual. Aunque esto sea bueno para el seguimiento, cuando el objeto está quieto se traduce en un ruido constante en la estimación de la posición. Para solucionar este problema se puede emplear una combinación de diferentes modelos de movimiento al mismo tiempo [IB98c], unos para movimientos bruscos y otros para situaciones en las que el objeto se mueva más despacio. En los experimentos veremos de forma más clara esta influencia.

4.4. Modelo de observación y proyección 3D

El modelo de observación es otra pieza clave del filtro de partículas. Su cometido es relacionar el estado actual con las observaciones que produce. Especifica la probabilidad de una observación $\boldsymbol{z}_k$ cuando el sistema se encuentra en el estado $\boldsymbol{x}_k$ y se representa mediante la distribución de probabilidad $p(\boldsymbol{z}_k|\boldsymbol{x}_k)$.

Al igual que sucedía en el modelo de movimiento, toda la información que incluyamos en el modelo de observación servirá para mejorar el seguimiento. En condensación al realizar las hipótesis mediante el modelo de movimiento es el modelo de observación el que se encarga de descartar aquellas hipótesis que no sean verosímiles a la luz de las nuevas observaciones.

Por este motivo, el modelo debe recibir la posición de un posible objeto y comprobar si la observación obtenida podría haber sido generada desde dicho punto. Cuanto más preciso sea el modelo de observación a la hora de seleccionar las hipótesis correctas, mejores prestaciones se obtendrán del filtro. Las comprobaciones del modelo de observación pueden ser todo lo complicadas que se quiera, añadiendo información acerca de la forma del objeto, de sus características, etc.

En las hipótesis que presentamos al comienzo de esta tesis indicamos que aun usando modelos de observación muy simples podríamos obtener unos buenos resulta-

dos en el seguimiento del objeto. La ventaja de usar modelos de observación sencillos es que son rápidos y fáciles de calcular. Al tener que probar muchas hipótesis, el tiempo en comprobar cada una de ellas será crítico, por lo que reducir la complejidad del modelo de observación aumentará la velocidad del filtro.

Otra ventaja de usar un sistema probabilístico basado en Bayes, como ya adelantamos en el capítulo 2 es que podemos combinar diferentes modelos. De esta forma, si un modelo de observación simple no es suficientemente bueno, se puede combinar con otro para obtener mejores resultados. Esta combinación probabilística también se puede ver como una combinación aditiva de estímulos. Por ejemplo, si vemos algo marrón y negro, que se mueve y además está cerca del suelo es probable que se trate de un perro. La imagen puede tener otras zonas marrones o negras, o incluso otros objetos en movimiento, pero es la combinación de todos los estímulos la que nos proporciona la intuición de que lo que vemos es un perro. Incluso si se detectan erróneamente objetos que no existen o que no son los que buscamos, el proceso secuencial los irá descartando paulatinamente dado que no recibirán el suficiente soporte para cada característica buscada a largo plazo.

La utilización de estímulos aditivos también nos permite abordar el problema de la vista parcial de un objeto. Es probable que al perder parte del objeto se pierdan parte de los estímulos que lo definen. Sin embargo, la implementación secuencial permitirá mantener la estimación correcta de la posición del objeto con los estímulos que siguen observándose.

Aunque, como hemos comentado, se pueden usar modelos de observación complejos, a partir de ahora únicamente consideraremos modelos sencillos, como un filtro de color o un filtro de movimiento. En el caso de usar objetos articulados o con orientación, se pueden desarrollar modelos que tengan en cuenta esta información extra a la hora de buscar color o forma en las imágenes.

La forma más sencilla de buscar un objeto de un color conocido es realizar un filtrado de la imagen buscando píxeles de ese color. En vez de filtrar toda la imagen, nosotros sólo necesitamos comprobar la validez de cierta hipótesis. Cada hipótesis, o lo que es lo mismo, cada partícula representa una posición tridimensional que podemos proyectar en las cámaras. En caso de que la partícula o punto tridimensional pertenezca al objeto, la zona de la imagen en la que proyecta debería ser del color de

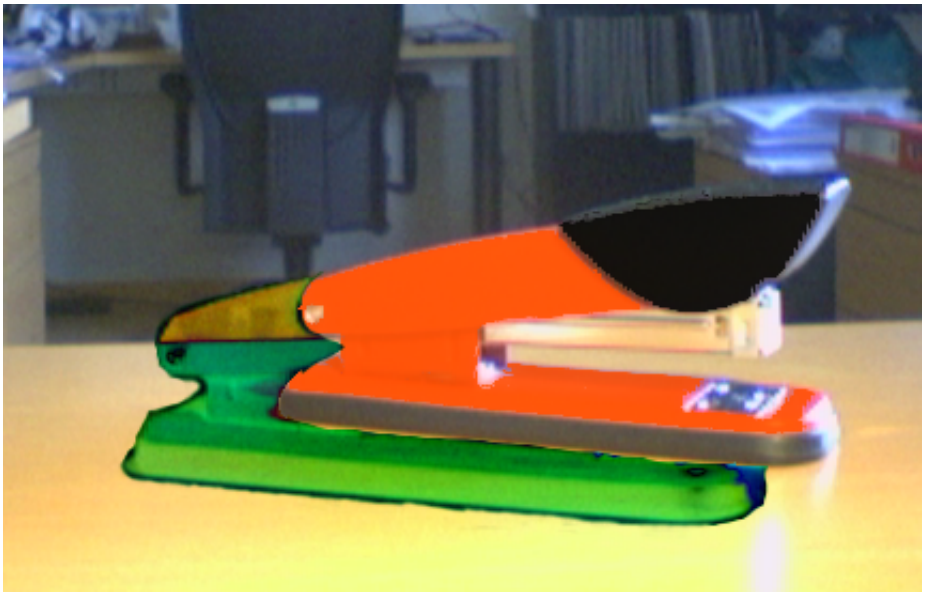

Figura 4.2: Estímulos visuales aditivos.

dicho objeto. Para nuestro modelo de observación, por tanto, sólo resulta necesario filtrar esta pequeña zona de la imagen para comprobar si la hipótesis resulta verosímil o no. De esta forma tomaremos una ventana de tamaño W ($5x5$ píxeles es un valor típico) en torno al punto de proyección de esa partícula en la imagen y realizaremos un filtrado de color de esa zona. Una vez que el filtrado se ha completado, se cuenta el número de píxeles que han pasado el filtro y cuántos no, devolviendo este valor normalizado a uno. El valor de de probabilidad obtenido puede calcularse de la siguiente forma:

$$p(\boldsymbol{x}_k|\boldsymbol{z}_k) = \frac{r}{W^2} \tag{4.2}$$

donde r es el número de píxeles que pasa el filtro en la vecindad. Podemos observar este cálculo en la figura 4.3). El punto verde representa la posición de la proyección de la partícula que estamos considerando. Sobre dicho punto se construye una ventana de vecindad de tamaño $5x5$ y se aplica un filtro de color, marcando aquellos píxeles que pasan los umbrales de filtrado.

Una primera ventaja de usar este método aparece cuando contamos el número

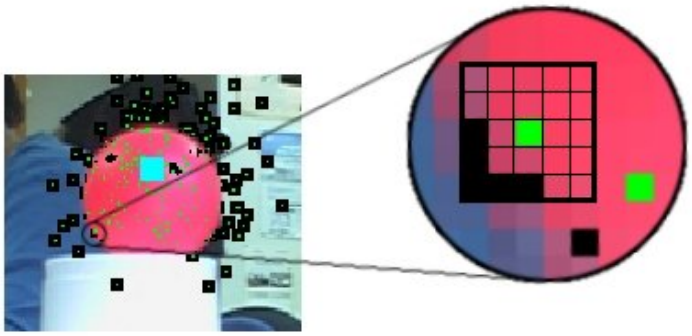

Figura 4.3: Modelo de observación usando un filtro de color en una ventana de vecindad.

de píxeles que debemos filtrar siguiendo esta técnica y el número de píxeles totales. Con ventanas de $5x5$ y 200 partículas en dos imágenes de 320x240 sólo tendríamos que filtrar 10.000 píxeles, frente a los 153.600 totales. Esto representa un orden de magnitud menos de puntos a filtrar. Si subimos el número de partículas hasta 1.000, el número de píxeles que debemos filtrar aumenta hasta 50.000, todavía lejos del número total de píxeles. La ventaja es aún más evidente cuando consideramos imágenes de mayor resolución, como pueden ser 640x480. El número de píxeles totales aumenta, mientras que el número de píxeles que necesitamos filtrar se mantiene igual al no ser necesario explorar toda la imagen. El número de píxeles a filtrar se mantiene en 10.000 y 50.000 para cada caso, pero el total de píxeles asciende a 614.400. O lo que es lo mismo, únicamente deberíamos filtrar el 8 % y el 1,6 % de los píxeles en cada caso.

Puede darse el caso que el número de píxeles que pasen el filtro en la ventana de vecindad sea cero. Esto puede darse por cuatro causas.

- En primer lugar puede deberse a que la hipótesis sea incorrecta, por lo que no hay puntos del color buscando en esa región del espacio.
- En segundo lugar puede que exista una oclusión del objeto y que por ese motivo no veamos el color buscado.
- En tercer lugar, puede que la hipótesis sea parcialmente mala, es decir, que la partícula proyecte correctamente en una cámara pero no en la otra. Esto implica

una triangulación implícita incorrecta del objeto que únicamente se solucionará cuando el objeto se vea de manera correcta en ambas imágenes.

- En cuarto y último lugar, el punto podría proyectar fuera de la cámara por lo que no tendríamos información para juzgarlo.

En el primer y último caso no resulta preocupante el hecho de que dicha muestra se pierda, pero en los otros casos estaríamos perdiendo información muy valiosa. Recordemos que la distribución de probabilidad $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{z}_k)$ que estamos estimando únicamente tiene información en aquellas regiones del espacio en las que colocamos partículas. Al eliminar estas partículas también estamos eliminando parte del soporte para nuestra estimación de la distribución *a posteriori*.

Por este motivo, el sistema de combinación probabilístico no debe verse influido por el hecho de que una cámara no vea ningún píxel del color buscado. Si una de las cámaras proporciona un valor de cero, la probabilidad total también será cero. Para evitar este problema se introduce un límite en el valor mínimo que puede devolver el modelo de observación, evitando los pozos de probabilidad asociados con la probabilidad cero. El modelo de observación corregido queda como:

$$p^c_{col}(\boldsymbol{x}_k|\boldsymbol{z}_k) = \begin{cases} \frac{1}{W^2}, & P_c\boldsymbol{x}_k \notin \texttt{imagen} \\ \frac{max(1,r)}{W^2}, & P_c\boldsymbol{x}_k \in \texttt{imagen} \end{cases} \tag{4.3}$$

donde P_c es la matriz de proyección de la cámara y $P_c\boldsymbol{x}_k$ es la proyección de la partícula en la cámara c.

Análogo a este modelo de observación vamos a usar información de movimiento. Para ello, usando el mismo sistema de ventanas que en el caso del filtrado de color, contamos el número de píxeles que han tenido un cambio significativo en el paso de la imagen anterior a la imagen actual. De esta forma se detectarán objetos que se muevan de un fotograma a otro. También es necesario aplicar la misma restricción que en (4.3) para evitar perder un objeto que estamos viendo cuanto éste se encuentra parado. El modelo de observación para detectar movimiento quedará como:

$$p^c_{mov}(\boldsymbol{x}_k|\boldsymbol{z}_k) = \begin{cases} \frac{1}{W^2}, & P_c\boldsymbol{x}_k \notin \texttt{imagen} \\ \frac{max(1,r')}{W^2}, & P_c\boldsymbol{x}_k \in \texttt{imagen} \end{cases} \tag{4.4}$$

donde r' es el número de píxeles en los que se ha detectado movimiento dentro de la ventana de vecindad, W.

Una modificación a este sistema consiste en usar un tamaño de ventana adaptable según la distancia de la cámara a la partícula. Cuando el objeto está demasiado lejos, el tamaño real visto en la imagen puede ser menor al tamaño de una ventana fija, lo que llevaría a dar un valor bajo en la probabilidad de observación. Lo contrario puede pasar con objetos grandes o demasiado cercanos, se aceptarán por buenas todas las ventanas que correspondan a alguna parte de objeto. Esto puede llevar a una estimación de posición errónea, como veremos en detalle durante la realización de los experimentos.

Una solución para ambos casos es calcular la distancia de la partícula a la cámara, lo que es factible dado que la partícula representa una posición 3D concreta. Una vez que tenemos esa distancia, podemos ajustar la ventana de vecindad según el tamaño esperado del objeto en píxeles.

En resumen, para que estos filtros funcionen correctamente es necesario conocer dos datos.

- Por un lado, debemos conocer el *color*, o colores, del objeto a seguir para poder ajustar los filtros de color.
- Por otro lado, debe conocerse el *tamaño* del objeto para así poder estimar el valor de la ventana de vecindad si usamos la aproximación adaptativa.

Antes de pasar a la combinación de cámaras debemos tener en cuenta que el modelo de observación para cada cámara puede ser diferente. Hasta ahora no hemos establecido restricción alguna a la elección de las cámaras, por lo que, en principio, podemos querer combinar cámaras de diferentes características. Por ejemplo, una puede tener una resolución más alta que las demás o puede ser una cámara infrarroja. Esto nos llevará a tener modelos de observación obligatoriamente diferentes. Aunque el marco bayesiano de combinación de cámaras que veremos en la siguiente sección no pone ningún tipo de restricción a la combinación de modelos de observación diferentes, para simplificar, supondremos que ambas cámaras son iguales y que empleamos el mismo modelo de observación en las dos. Esto tiene una implicación en el tipo de objetos que podemos seguir. En particular, supondremos que estamos

ante superficies lambertianas, esto es, superficies que se ven del mismo color desde cualquier dirección desde la que se miren. En otras palabras, no esperamos tener brillos o superficies tornasoladas. Si este fuera el caso, deberíamos tener en cuenta los cambios del color en función de la posición de la cámara y añadir esta información a cada modelo de observación. Cada cámara, al estar colocada en posiciones diferentes, debería configurarse de manera independiente.

En la literatura podemos encontrar técnicas de procesado visual más complejas y más exactas. Por ejemplo los histogramas de color, los autoespacios para describir objetos [BJ96, LRLY05] o las características invariantes a escala [SL04]. En esta tesis no pretendemos utilizar tales técnicas por dos motivos. En primer lugar, como ya hemos dicho, creemos que resulta suficiente emplear una colección de filtros simples para seguir un objeto y en segundo lugar queremos centrar este trabajo en el algoritmo para gestionar un número variable de objetos.

Hasta ahora hemos hablado únicamente de la utilización de una cámara. Para nuestra aplicación en particular no resulta suficiente, dado que no proporcionan información sobre la profundidad. El objetivo del modelo de observación es indicar cuáles de las hipótesis realizadas son verosímiles a la vista de las imágenes disponibles. Los puntos verosímiles serán aquéllos que proyectan correctamente en todas las imágenes al mismo tiempo y que son compatibles con los modelos de observación usados para cada una de ellas. Es decir, si estamos usando información de color y de movimiento con dos cámaras, los puntos interesantes serán aquellos que proyecten en las dos imágenes sobre puntos del color deseado y sobre los que, además, se haya detectado movimiento. Deben darse todas las condiciones al mismo tiempo para que consideremos que la partícula es buena.

La combinación de la información de cada cámara debe proporcionarnos una única medida de lo buena o mala que es la partícula. La manera más directa de combinar la información de todas las cámaras es emplear la probabilidad conjunta, esto es, el producto de todas (suponemos que todas las probabilidades son independientes entre sí). Ésta no es una aproximación exacta, dado que ambas cámaras están observando la misma realidad en instantes de tiempo muy próximos. Aún así, suponer independencia nos proporciona un método adecuado y simplifica mucho los cálculos con respecto al caso en el que no consideráramos esta independencia. La probabilidad

final para cada cámara quedará como:

$$p_c(\boldsymbol{x}_k|\boldsymbol{z}_k) = \prod_{obs} p^c_{obs}(\boldsymbol{x}|\boldsymbol{z}_k) \tag{4.5}$$

dónde p^c_{obs} es la probabilidad de una cámara (c) empleando un modelo de observación (obs).

Obsérvese que para evitar los pozos de probabilidad será necesario siempre tener en cuenta una probabilidad mínima, como expusimos en las ecuaciones (4.3) y (4.4).

Una vez que tenemos la probabilidad según el modelo de observación para cada cámara es necesario combinarla para obtener la probabilidad de observación total. Dado que queremos obtener información sobre profundidad y que las cámaras que usaremos son estáticas, necesitamos usar dos o más cámaras, con el fin de eliminar la ambigüedad en cuanto a la profundidad cuando se usa una única cámara.

Las restricciones que tiene este sistema a la hora de colocar las cámaras son las mismas que las de una triangulación. De hecho el procedimiento es completamente análogo a esta, aunque se realiza de manera inversa. En vez de buscar los puntos en las imágenes y triangular, lo que estamos haciendo es buscar el punto en 3D y calcular sus proyecciones. En cualquiera de los dos casos se forma el mismo triángulo en tres dimensiones.

La limitación de la triangulación implica que sólo seremos capaces de proporcionar una verdadera estimación de la posición 3D en aquellas zonas del espacio en las que se tenga línea de visión directa desde por lo menos dos cámaras. En principio el uso de más de dos cámaras no mejorará la precisión, siempre que las cámaras tengan orientaciones lo suficientemente diferentes y estén correctamente calibradas. Lo que sí se mejorará es tanto el espacio cubierto como la robustez frente a oclusiones o pérdidas del objeto debido a movimientos bruscos. Durante los experimentos volveremos sobre estas limitaciones.

De nuevo, podemos combinar la información procedente de diferentes cámaras usando un producto como muestra 4.6. Para calcular p_c también pusimos un límite mínimo. Por este motivo también se asegura que $p(\boldsymbol{x}_k|\boldsymbol{z}_k)$, la probabilidad de observación, sea mayor que cero en cualquier caso.

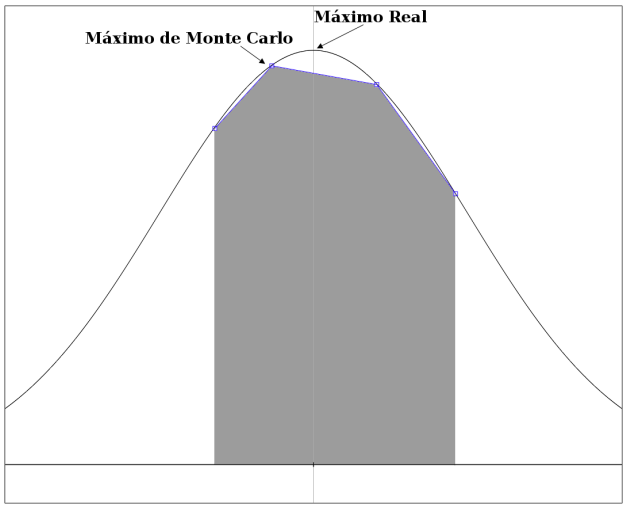

Figura 4.4: Diferencias entre el máximo de Monte Carlo y el máximo real de las distribuciones de probabilidad.

$$p(\boldsymbol{x}_k|\boldsymbol{z}_k) = \prod_i p_c(\boldsymbol{x}|\boldsymbol{z}_k) \tag{4.6}$$

Debe recordarse que la posición de todas las cámaras (ya sea la posición absoluta o relativa entre ellas) debe conocerse. En caso contrario no sería posible proyectar las partículas en las cámaras, paso imprescindible para poder calcular la probabilidad del modelo de observación.

4.5. Estimación de la posición del objeto

En cada iteración del filtro, ya sea usando el filtro de condensación o el filtro enfatizado, obtenemos una estimación de la densidad de probabilidad $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{z}_k)$. Sin embargo estamos interesados en extraer una estimación sobre la posición actual del objeto. Para ello podemos adoptar diferentes estrategias.

Por un lado podríamos elegir la partícula con mayor peso como la hipótesis más probable, al ser los pesos proporcionales a la distribución $p(\boldsymbol{z}_k|\boldsymbol{x}_k^i)$. Sin embargo, estos pesos únicamente nos indican cuál es el pico de la distribución discreta esti-

mada $p_N(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{z}_k)$ y puede que eso no sea suficiente. En la figura 4.4 podemos ver cómo esta estimación puede no corresponderse con el máximo de la distribución buscada, introduciendo un sesgo a la estimación.

La forma más directa de proceder consiste en emplear la media ponderada de las partículas, como vimos en la expresión 3.24:

$$\hat{\boldsymbol{x}}_k = \sum_{i=1}^{N} w_k^i \boldsymbol{x}_k^i \tag{4.7}$$

Este es el estimador de error cuadrático medio mínimo. Para el caso de distribuciones gaussinas será el mejor estimador posible. Dado que no tenemos más conocimiento *a priori* sobre el tipo de distribuciones que tendremos es una elección general adecuada.

4.6. Abducción con muestreo enfatizado

Hasta aquí hemos mostrado cómo adaptar el filtro de partículas más conocido, el filtro de condensación, para el seguimiento de un objeto en 3D usando únicamente información visual. Este sistema, como veremos durante los experimentos, funciona de manera correcta. Es capaz de seguir un objeto de manera robusta y de recuperarse frente a oclusiones parciales o totales. Sin embargo, podemos fijarnos en un detalle antes de continuar: El filtro de condensación realiza la búsqueda de manera totalmente ciega. Las hipótesis se toman usando únicamente el modelo de movimiento $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$, dándonos las mejores posiciones posibles atendiendo únicamente a la última posición conocida, $\boldsymbol{x}_{k-1}$.

Hay otra fuente de información que este filtro ignora a la hora de colocar las nuevas hipótesis: las observaciones. Las observaciones nos pueden dar pistas acerca de dónde colocar las mejores hipótesis [RC01]. Podemos ver ésto como un sesgo a la hora de realizar nuevas hipótesis. Este sesgo, al que denominaremos abducción, está a medio camino entre la reconstrucción y la búsqueda ciega. No se trata de una reconstrucción dado que no pretendemos extraer toda la información sobre la escena. Únicamente queremos encontrar cuáles son las zonas prometedoras del espacio de

estados, con el fin de realizar las búsquedas en esas regiones.

Como vimos en la sección 3.3 existen otras variedades de filtros, basados igualmente en técnicas de Monte Carlo, que nos permiten añadir cierto sesgo sobre las hipótesis que queremos crear. Se trata del muestreo enfatizado. En este tipo de muestreo las hipótesis se toman de una función $q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{z}_k)$, conocida como función de importancia o función de propuesta. La correcta elección de esta función puede mejorar los resultados dramáticamente sin aumentar, necesariamente, la complejidad del sistema.

En el caso del seguimiento visual en 3D es posible añadir una restricción muy importante a la hora de crear nuevas hipótesis. Sabemos que los objetos estarán colocados sobre la intersección de las líneas de visión de las cámaras. Podemos tomar esta información a la hora de crear nuevas hipótesis. Si colocamos todas las hipótesis sobre la línea de visión de una cámara, todas las partículas proyectarán, por lo menos, correctamente en una cámara. Eventualmente alguna se situará bien en todas las cámaras, indicándonos la verdadera posición del objeto.

Procediendo de esta forma obtenemos dos ventajas importantes. Por una parte, no perdemos recursos computacionales en zonas sin ningún interés. Ninguna de las partículas que obtenemos de esta forma caerán en zonas que el modelo de observación no vaya a premiar de alguna forma. Dado que el principal límite que tenemos es la capacidad de cómputo, al hacer un uso más racional de la colocación estamos mejorando el funcionamiento del sistema. Por otra parte, tenemos un efecto muy importante en la búsqueda. La búsqueda que realizaba el filtro de condensación era una búsqueda ciega, moviéndose incrementalmente desde la posición actual. Las partículas se mueven de forma errática siguiendo el modelo de movimiento hasta que alguna, por azar, cae sobre una zona de interés. En ese momento el modelo de observación nos dirá que esa partícula es interesante y el filtro comenzará a colocar un mayor número de partículas en esa región del espacio. El problema es que este proceso puede ser muy lento, dependiendo de la velocidad a la que las partículas se dispersan de acuerdo al modelo de movimiento. En cambio, al añadir una función de importancia que tiene en cuenta las observaciones directamente, la convergencia es mucho más rápida. Tan pronto como un objeto sea captado por la cámara el filtro comenzará a colocar hipótesis en las zonas más probables del espacio.

$\left[\{\boldsymbol{x}_k^i, w_k^i\}_{i=1}^N\right] = MUESTREO\_ENFATIZADO\,[\boldsymbol{z}_k]$

- Para cada $i = 1 : N$
  - Obtiene $\boldsymbol{x}^i \sim q(\boldsymbol{x}_k|\boldsymbol{z}_k)$
  - Asigna w^i usando la expresión 3.25.
  - Calcula $w_k^i \propto \frac{p(\boldsymbol{z}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k)}{q(\boldsymbol{x}_k|\boldsymbol{z}_k)}$
  - Calcula la suma total de los pesos: $t = \sum_{i=1}^N w_k^i$
  - Para cada $i = 1 : N$
    - Normaliza $w_k^i = t^{-1} w_k^i$

Cuadro 4.2: Algoritmo de muestreo enfatizado detallado.

Para añadir la abducción podemos modificar el filtro de condensación del cuadro 4.1 y emplear el algoritmo de muestreo enfatizado del cuadro 4.2. Los pasos son similares. La diferencia principal radica en que hacemos un uso explícito de la distribución de propuesta $q(\boldsymbol{x}_k|\boldsymbol{z}_k)$.

Este algoritmo tiene una naturaleza no secuencial dado que la función de propuesta depende únicamente de las imágenes y no de la posición actual de las partículas. Hemos tomado esta decisión para evitar la influencia de la posición actual de la partícula sobre la información que nos proporcionan las observaciones. Nuestro objetivo es detectar zonas nuevas tan pronto como aparecen y no hacer converger las trayectorias existentes hacia estas posiciones prometedoras.

En nuestro caso, como hemos comentado, vamos a tomar exclusivamente las líneas de visión para crear esta función de propuesta. El proceso puede verse en la figura 4.5. Los pasos que se siguen son los siguientes:

- En primer lugar se filtra completamente la imagen por color. De esta forma se extraen los píxeles interesantes en la imagen, aquellos que, probablemente, serán puntuados con probabilidades altas por el modelo de observación. Con los píxeles extraídos se construye una máscara binaria.

- A continuación se realiza un histograma de color sobre cada eje de la máscara. De esta forma sabemos qué coordenadas x e y tienen más componentes del

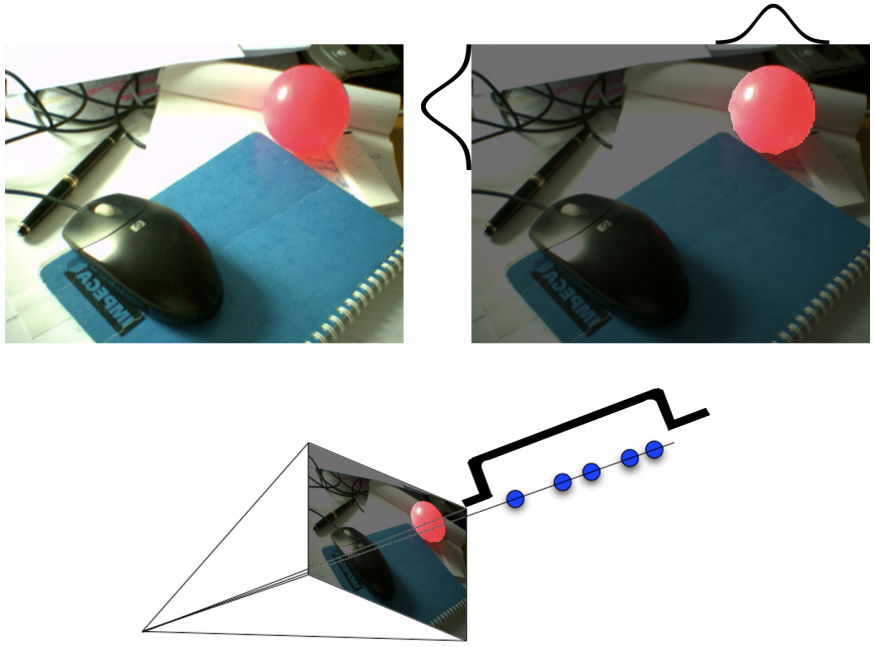

Figura 4.5: Extracción de las líneas de visión para la abducción.

color que buscamos. Obteniendo el máximo de los histogramas, podemos conseguir las coordenadas x, y de la proyección del objeto. El histograma puede verse dibujado al borde de la captura en la figura 4.5.

- Tras obtener las coordenadas, resulta directo trazar una línea desde el foco de la cámara que pase por ese punto en la imagen. Las nuevas hipótesis se colocarán sobre dicha línea de forma uniforme, entre una distancia mínima y una distancia máxima. Dichas distancias vienen marcadas por nuestro conocimiento de la escena que queremos seguir. Sin embargo, sería posible tomar valores genéricos en el caso de no tener dicha conocimiento. La distancia mínima vendrá marcada por la distancia a la cual las cámaras no son capaces de enfocar. La distancia máxima será aquella para la que un objeto ya no se puede distinguir en la imagen debido a la resolución de ésta. Estas distancias estarán relacionadas con el tamaño de la zona de seguimiento.

Dado que estamos usando una distribución uniforme sobre una línea fija, todas

las partículas $\boldsymbol{x}_k$ generadas por esta función de propuesta serán igualmente probables. Esto simplifica el cálculo de la expresión (3.25), dejándolo como sigue:

$$w^i = \frac{p(\boldsymbol{z}_k|\boldsymbol{x}_k)p(\boldsymbol{x})}{\sum_{j=1}^{N} w^j} \tag{4.8}$$

El hecho de considerar todas las partículas generadas por la distribución de importancia, viene determinado porque no usamos ningún tipo de información sobre a qué distancia podría encontrarse el objeto. La distribución de importancia podría añadir información acerca de cuál es el tamaño del objeto y así ajustar más las posiciones posibles en las que se encuentra el objeto. Sin embargo, como veremos, algo tan sencillo como realizar hipótesis sobre toda la línea de visión funciona correctamente.

Una vez obtenida la población de partículas con este algoritmo podemos emplear el mismo método de la sección 4.5 para obtener una estimación de la posición actual del objeto. De esta forma podemos implementar un segundo sistema de seguimiento, en este caso no secuencial, que también emplearemos durante los experimentos. Comprobaremos las diferencias entre las dos aproximaciones al problema: por un lado la secuencial, que coloca las partículas de forma ciega, y por otro la no secuenial, que lo hace extrayendo la información directamente de las observaciones (abducción).

4.7. Experimentos

En este apartado presentamos los resultados de los experimentos realizados con los sistemas de seguimiento basados tanto en el filtro de condensación como en el muestreo enfatizado. Los experimentos que vamos a mostrar intentan reflejar el correcto funcionamiento de los sistemas descritos. Por ese motivo hemos dividido los experimentos en unos estáticos, encargados de comprobar la precisión y la correcta convergencia del sistema y otros dinámicos, encargados de comprobar características como el seguimiento o el error residual de estimación.

Como queremos comparar los resultados entre las dos alternativas expuestas, realizaremos los mismos experimentos con los dos algoritmos. Nuestro objetivo es comprobar qué características de cada filtro son más interesantes en cada situación. Además, también queremos comprobar cómo se enfrenta cada filtro en situaciones en

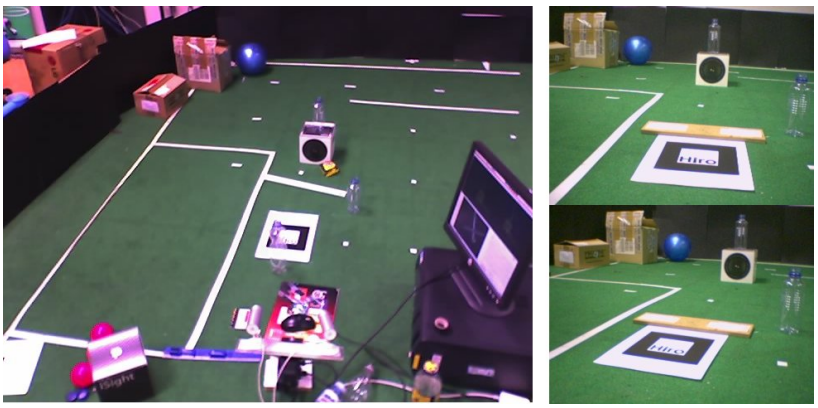

Figura 4.6: Capturas desde el montaje para los experimentos.

las que aparezcan más de un objeto al mismo tiempo en la escena y qué limitaciones tiene.

Para la realización de los experimentos los algoritmos han sido implementados usando la plataforma software METS que se describe en el capítulo 6. Esta plataforma está diseñada para ser modular, permitiendo probar diferentes algoritmos usando las estructuras comunes que proporciona. En particular, los experimentos aquí mostrados hacen uso de los algoritmos de condensación y de muestreo enfatizado resumidos en los cuadros 4.1 y 4.2.

Para la realización de los experimentos se han colocado dos cámaras sobre un soporte fijo, evitando así que se muevan. Ambas cámaras se colocan mirando hacia la misma dirección, situadas una cerca de la otra. En la figura 4.6 se pueden ver un par de capturas realizadas con las mismas. En dichas imágenes se puede apreciar el patrón de calibración común para los dos imágenes con el que se han obtenido los parámetros extrínsecos. Dicho patrón sirve también como referencia de la posición de los ejes, tal y como se observa en el diagrama de la figura 4.7.

Los parámetros extrínsecos obtenidos indican que la separación entre las cámaras es de algo más de 15 centímetros. Las coordenadas exactas de los centros de cada cámara con respecto al patrón visual colocado en el suelo son:

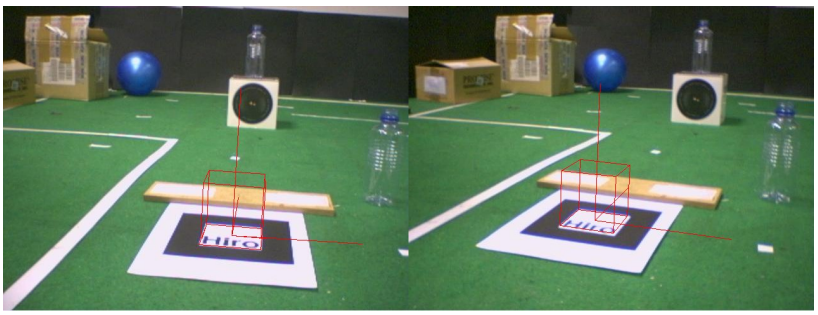

Figura 4.7: Esquema del montaje para los experimentos.

$$[12,2377; -347{,}1779; 121{,}1890]$$
$$[109,7155; -357,3541; 125{,}9411]$$

La orientación, representada como un cuaternión, de cada cámara es la siguiente:

$$[0,768453; 0,057495; -0,017651; 0,637073]$$
$$[0,764558; 0,123050; -0,074303; 0,628321]$$

Las cámaras elegidas son dos Apple iSight con conexión Firewire. La matriz de calibración intrínseca que usamos, obtenida mediante ARToolkit, está resumida en la siguiente matriz de proyección:

$$\begin{pmatrix} 822{,}31 & 0 & 315 \\ 0 & 838{,}46 & 235 \\ 0 & 0 & 1 \end{pmatrix}$$

Para los experimentos hemos usado una resolución de 640x480 píxeles con una velocidad de captura de 15 fotogramas por segundo. Empleamos la máxima resolución que nos proporciona la cámara para forzar las capacidades de los sistemas desarrollados. El funcionamiento a 320x240 resulta mucho más liviano por lo que resultaría más complejo encontrar las limitaciones al usar estos algoritmos en tiempo

real, como se comenta en la sección 4.4.

La máquina empleada en los experimentos es un ordenador de escritorio convencional, un Pentium IV con Hyper-threading a 3GHz con 2 GB de memoria RAM. La cantidad de memoria necesaria es alta para grabar los vídeos con las secuencias usadas, como se comenta en la sección 6.5. En caso de que no se necesite esta característica el consumo de memoria es muy inferior.

4.7.1. Experimentos de precisión

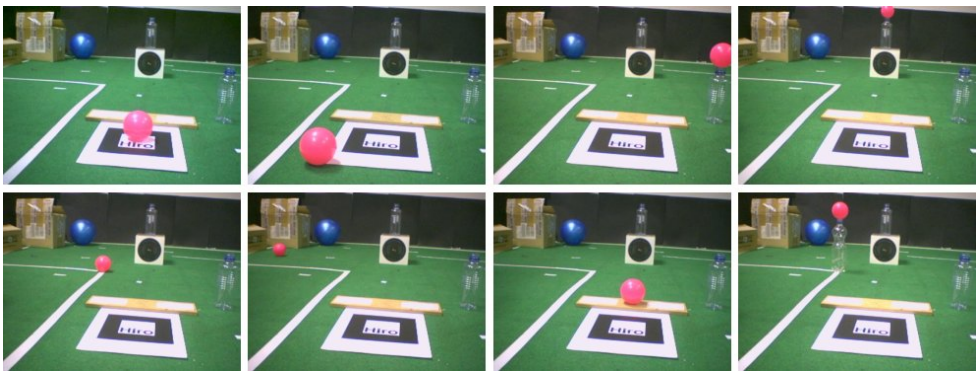

Figura 4.8: Colocación de objetos en posiciones estáticas.

La primera tanda de experimentos va orientada a comprobar la precisión del sistema de localización. Para ello hemos colocado una pelota de un color llamativo en diferentes posiciones, tal y como se ve en la figura 4.8.

Las coordenadas reales de cada una de las posiciones se miden, empleando una regla, con respecto a los ejes fijados por el patrón de referencia. En el cuadro 4.3 se pueden ver las posiciones en las que se coloca cada uno de los objetos que corresponden a las imágenes de la figura 4.8 de izquierda a derecha. De esta forma tenemos una referencia para ser capaces de estimar el error entre la medida real y la que proporciona el sistema de seguimiento. El volumen de seguimiento en el que se encuentran los objetos es de aproximadamente 270 litros. La pelota que usamos tiene un tamaño de unos 7 centímetros de diámetro, lo que nos da unos 18 centilitros de volumen a seguir.

ID	X(cm)	Y(cm)	Z(cm)
1	0	0	3,5
2	-10,5	-14,85	3,5
3	24,7	22,5	20,2
4	1	99	35,2
5	-20	65	3,5
6	-64,5	96	3,5
7	0	17	3,5
8	-20	65	26,5

Cuadro 4.3: Posiciones estáticas de los objetos de prueba.

Usando únicamente un objeto a la vez se mide el resultado proporcionado por el sistema para cada posición. Los resultados pueden verse en la figura 4.4.

En las tablas mostramos los resultados divididos por componentes. La media del error para el caso de condensación es de unos 6,10 cm y en el del muestreo enfatizado de 6,56 cm. Estos números incluyen los errores de calibración de las cámaras. Aún así los resultados son satisfactorios en cuanto al tamaño del espacio que estamos manejando supera el metro cuadrado.

La falta de precisión viene dada por el tamaño de la pelota y el modelo de observación. Este último no tiene en cuenta el tamaño del objeto a seguir y considera igual de probable los puntos en el borde del mismo que los del centro. La falta de precisión se produce al usar emparejamientos de los puntos que no se corresponden a la misma parte de la pelota. Aún con estos problemas y usando un modelo de observación tan sencillo, los resultados son suficientemente buenos.

También podemos notar cómo existen diferencias en los errores de cada eje. En los ejes X y Z los errores son muy bajos. Éstos se corresponden a los ejes perpendiculares a la orientación de las cámaras, el eje Y. En el Z, que se corresponde con la altura desde el suelo, se han realizado un menor número de medidas. Pero en el eje X, completamente análogo, hemos obtenido un error máximo menor a 3cm a 1 m de distancia manteniendo un valor medio por debajo del centímetro.

El eje Y se ha alineado con la profundidad de la escena para poder estudiar más fácilmente los resultados con la profundidad. Éste eje es el que peor resultados, en cuando a errores, proporciona. Esto se debe a que cualquier partícula que caiga en la

Algoritmo de Condensación

X(cm)	Y(cm)	Z(cm)	Error X (cm)	Error Y (cm)	Error Z (cm)	Error (cm)
-0.46	3.66	2.11	-0.46	3.66	-1.39	3.67
-9.87	-14.98	4.69	0.63	-0.13	1.19	1.75
23.28	23.02	18.88	-1.42	0.52	-1.32	2.10
0.98	84.71	36.33	-0.02	-14.29	1.13	15.15
-17.33	55.43	5.66	2.67	-9.57	2.16	9.59
-65.42	95.09	4.95	-0.92	-0.91	1.45	2.88
1.08	7.71	6.81	1.08	-9.29	3.31	9.92
-20.51	67.89	28.11	-0.51	2.89	1.61	3.74
Media			0,96	5,16	1,70	6,10

Algoritmo de Muestreo Enfatizado

X(cm)	Y(cm)	Z(cm)	Error X (cm)	Error Y (cm)	Error Z (cm)	Error (cm)
-0.42	3.41	2.25	-0.42	3.41	-1.25	3.59
-9.85	-15.09	4.75	0.65	-0.24	1.25	1.45
23.23	22.81	18.92	-1.47	0.31	-1.28	2.00
1.00	81.85	36.16	-0.00	-17.15	0.96	17.21
-17.13	54.48	5.88	2.87	-10.52	2.38	11.21
-64.12	92.30	5.31	0.38	-3.70	1.81	4.36
1.11	7.36	6.94	1.11	-9.64	3.44	10.25
-20.28	66.73	28.15	-0.28	1.73	1.65	2.38
Media			0,90	5,84	1,75	6,56

Cuadro 4.4: Precisión de localización con filtro de condensación y con muestreo enfatizado para un objeto.

zona marcada en la figura 4.9 será aceptada por el modelo de observación como una partícula correcta e indistinguible de las demás. Nos referiremos a éste área como *zona de incertidumbre* y usaremos este concepto en otros experimentos. Cuanto más cerca estén las cámaras entre sí, más se alargará esa región. Podemos suponer que las partículas tenderán a distribuirse uniformemente sobre dicho área dado que el modelo de observación las evaluará todas de la misma forma. Como el área no es simétrica, la medida ponderada de las partículas no se corresponderá con la verdadera posición del objeto. Esto se traduce en unos resultados menos precisos en el eje de profundidad.

Para solucionar esta limitación podemos jugar con dos factores: el modelo de

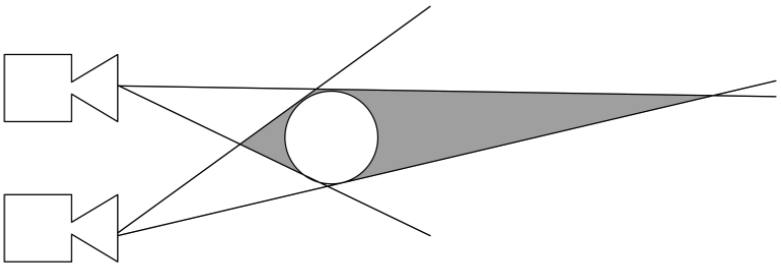

Figura 4.9: Problemas con la estimación de profundidad del par estéreo.

observación y el número y la posición de las cámaras. En ambos casos el objetivo buscado es similar: reducir la zona del espacio marcada en la figura 4.9. Si modificamos el modelo de observación, por ejemplo, para que tenga en cuenta el tamaño del objeto no todos los puntos de la zona marcada serán válidos. En vez de usar una ventana de filtrado se pueden usar ventanas que se adapten a la profundidad esperada del objeto. De esta forma no todas las partículas de la zona indicada proyectarán bien, sino sólo las que pasen por la línea central, reduciendo significativamente el área. Este método también tiene la contrapartida de que es más sensible a los errores de calibración de las cámaras.

El otro método para reducir este problema consiste en colocar las cámaras de otra forma. Dado que las hemos puesto como un par estéreo mirando hacia el eje Y las dos cámaras refuerzan la información obtenida en los ejes X y Z, pero proporcionan poca información sobre el eje Y. Con otra colocación o añadiendo nuevas cámaras, podemos reducir significativamente el área de incertidumbre, como se muestra en la figura 4.10. Todo ello sin cambiar el modelo de observación usado.

Una opción que no hemos implementado en los experimentos consiste en usar objetos más pequeños, lo que aumentaría significativamente la precisión del sistema. A primera vista puede parecer extraño que reducir el tamaño del objeto pueda mejorar el error, el motivo está en las simplificaciones hechas por el modelo de observación. Si reducimos el tamaño del objeto estaremos así mismo reduciendo el tamaño de la zona de incertidumbre de la figura 4.9. Esto mejorará la precisión de la estimación puntual del objeto.

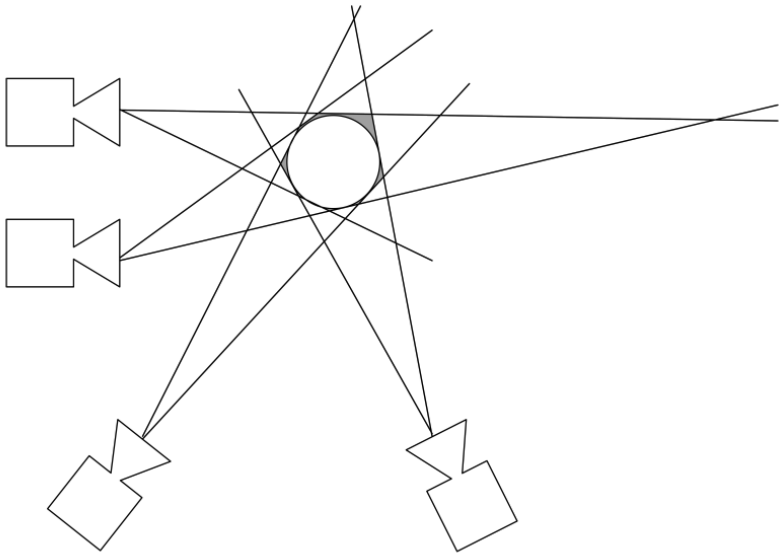

Figura 4.10: Colocación de nuevas cámaras para reducir la zona de incertidumbre.

4.7.2. Experimentos dinámicos

Tras comprobar los resultados de precisión podemos pasar a estudiar la dinámica del sistema. A diferencia de la precisión, donde los resultados de condensación y de muestreo enfatizado son similares, las dinámicas de estos dos sistemas son radicalmente diferentes. Para comprender las diferencias, antes de mostrar los resultados de las pruebas, explicaremos cómo funciona cada uno de los sistemas desde el punto de vista de la dinámica de la población de partículas.

El algoritmo de condensación es un algoritmo secuencial de búsqueda ciega por el espacio de estados. Esto quiere decir que coloca partículas usando únicamente el modelo de movimiento desde las mejores posiciones encontradas hasta el momento. En un primer momento puede que no encuentre ninguna posición correcta por lo que todas las partículas tendrán pesos similares. Lo que produce esto es que, probabilísticamente, todas promocionen a la siguiente generación desplazándose según lo indicado por el modelo de movimiento. Dicho de otra forma, nuestro modelo de movimiento se corresponde con un paseo aleatorio gaussiano, o lo que es lo mismo, una búsqueda ciega por todo el espacio. Las nubes de partículas se expanden desde

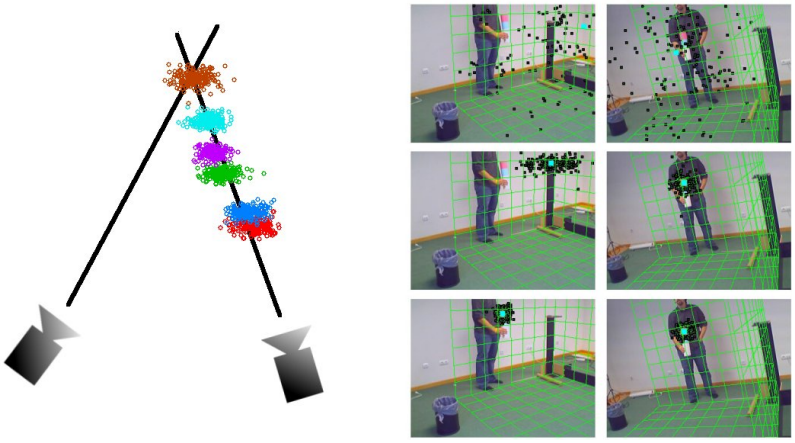

Figura 4.11: Evolución espacial de las partículas en el filtro de condensación.

sus posiciones iniciales hasta cubrir la escena completa.

Eventualmente una de las partículas caerá sobre una de las proyecciones del objeto a seguir. El tiempo que tarda en hacerlo no está acotado y depende del tamaño relativo del objeto frente al espacio a recorrer, del modelo de observación y del número de partículas usadas.

Cuando la proyección de una de las partículas cae sobre el objeto en una de las cámaras, es de esperar que su peso sea mucho más alto que el del resto de las partículas que no proyectan bien en ninguna de las cámaras. En ese momento el paso de remuestreo se encargará de crear muchas partículas en las cercanías del punto prometedor, que estará en la línea de visión del objeto desde una de las cámaras. Una parte de las partículas creadas caerá fuera de dicha línea, por lo que el modelo de observación volverá a bajar su peso. La porción de la nube de partículas que se mantenga dentro de la línea de visión se irá desplazando por esta hasta toparse con la línea de visión desde la otra cámara. En la figura 4.11 podemos ver cómo se desplaza la nube por la línea epipolar de la otra cámara.

Al realizar los experimentos hemos observado que el movimiento de las partículas sobre la línea de visión no es aleatorio, como podría pensarse en un primer momento. La nube de partículas tiende a alejarse de las cámaras. Esta deriva sistemática puede resultar dañina si el objeto se encuentra más cerca de las cámaras que la nube de partículas. Este efecto se produce por la forma cónica del modelo de obser-

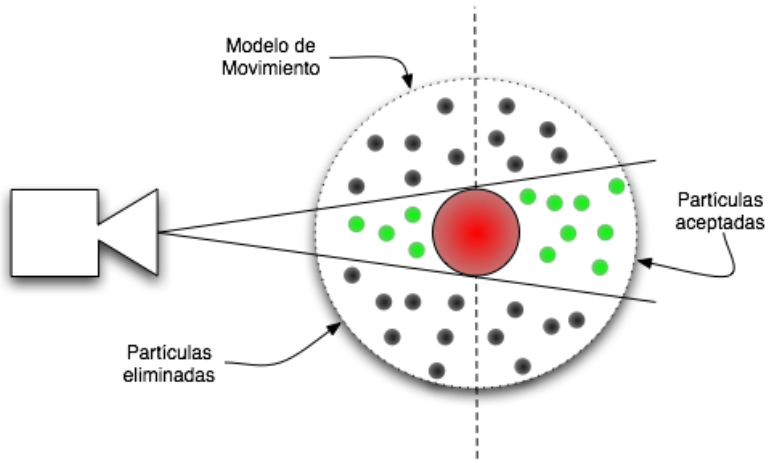

Figura 4.12: Deriva sistemática de las nubes de partículas.

vación proyectivo. Después del paso de predicción, el modelo de movimiento tiene la misma probabilidad de colocar una partícula más cerca de la cámara que más lejos de la cámara. Sin embargo, el hecho de caer más cerca hace más improbable que la partícula caiga sobre la proyección del objeto. En ese caso, la partícula será descartada en el siguiente remuestreo haciendo que la población de partículas se vaya alejando lentamente de las cámaras. Este efecto puede verse en la figura 4.12. Las partículas que proyectan correctamente en la cámara son aquellas dentro del cono de visión. Podemos observar cómo hay más partículas que proyectan correctamente detrás de la posición actual de la partícula que las que hay delante.

Las pruebas realizadas también muestran que la dinámica del seguimiento con el muestreo enfatizado es radicalmente diferente. En este caso no usamos una aproximación secuencial por lo que cada iteración es completamente independiente de la anterior. El funcionamiento para cada instante de tiempo será como si fuera el primero, por lo que hablar de convergencia en este caso no tiene sentido. Según aparezca un objeto en la escena el sistema tendrá la misma probabilidad de encontrarlo o no encontrarlo en todas las iteraciones siguientes. En los resultados de las pruebas de convergencia podremos ver cómo el tiempo de convergencia es instantáneo.

Nuestros siguientes experimentos se centran en estudiar la convergencia de los algoritmos. Para ello usamos la misma secuencia de vídeo en ambos casos. Gracias

a que METS permite emplear vídeos, podemos comparar los algoritmos *exactamente* con los mismos datos de entrada. La secuencia de vídeo está compuesta por 5 fotogramas en los que no aparece ninguna pelota y 100 en los que sí aparece. Se ha realizado concatenando dos secuencias de vídeo reales, para poder trabajar con imágenes reales pero al mismo tiempo forzar a que la aparición del objeto en el campo visual sea instantánea.

Cuando usamos condensación el sistema puede no converger en el tiempo que estamos usando (120 fotogramas a 15fps). Sería necesario tener vídeos más largos, pues como hemos comentado el tiempo para localizar la pelota no está acotado.

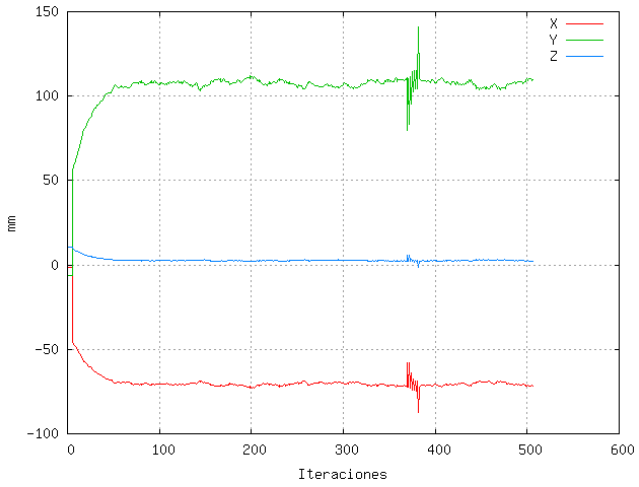

Figura 4.13: Convergencia de la estimación de la posición para un único objeto.

En la figura 4.13 podemos ver la convergencia para el objeto situado en la posición 6 de la tabla 4.3. Hemos elegido esta posición al encontrarse la pelota suficientemente alejada de la cámara, porque proporciona cierto grado de complejidad a la localización de la misma. En el eje de abscisas se muestran las iteraciones del filtro y en el eje de ordenadas la estimación para cada uno de los ejes, en tres colores diferentes.

Aproximadamente en 80 iteraciones el filtro ha convergido a la posición correcta. También observamos que cada eje converge a una velocidad diferente y con un error residual distinto, como comentamos en los resultados de precisión.

Una anomalía que conviene resaltar, son las fluctuaciones que aparecen sobre

la iteración 150, 300 y 450 aproximadamente. Observamos cómo la estimación tiene un pico de ruido en esos instantes de tiempo. Dicho error de estimación no aparece por el algoritmo de seguimiento empleado, sino por el algoritmo de segmentación. La implementación de METS incluye un algoritmo de segmentación que trabaja sobre las partículas, como se discutirá en profundidad en la sección 5.4. El objetivo de este algoritmo es proporcionar estimaciones correctas para el caso del seguimiento de varios objetos. Sin embargo, como se verá, su simplicidad puede causar la aparición de objetos espúreos. Básicamente el funcionamiento del algoritmo puede colocar dos grupos para cubrir el área de un objeto localizado. Al tener dos estimaciones, estás se reparten el área que ocupa el objeto sobre el espacio de estados, moviendo la estimación de su posición actual. Tras unas iteraciones el algoritmo de segmentación elimina uno de estos dos grupos, al no tener suficiente base para el mismo. Al desaparecer uno, el otro vuelve a cubrir todas las partículas asociadas al objeto obteniendo de nuevo una estimación correcta.

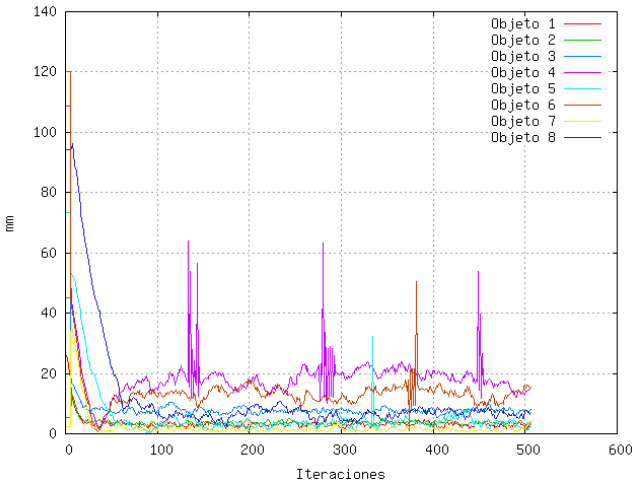

Figura 4.14: Convergencia del error de estimación usando condensación.

La figura 4.14 contiene las gráficas de error para los 8 objetos de la tabla 4.3. Podemos fijarnos en cómo las dinámicas para converger a las 8 posiciones son similares en cuanto a tiempo y en cuanto a forma del error residual. Sin embargo, vemos cómo los objetos que más tardan en converger son los que se encuentran más alejados del centro de coordenadas, o lo que es mismo, del punto de inicialización de la

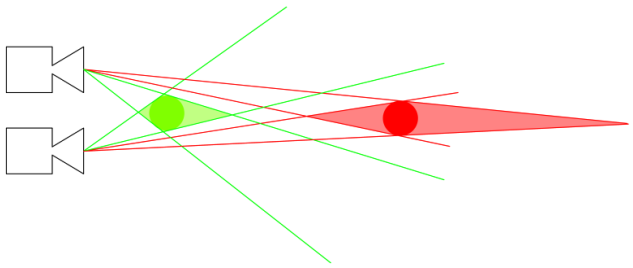

Figura 4.15: Variación de la zona de incertidumbre en función de la posición del objeto.

nube de partículas. La curva de error del objeto 4, el más alejado y el que más tarda en converger a una posición estable, alcanza un mínimo antes de estabilizarse. Esta posición sería más deseable que la posición que alcanza posteriormente, con mayor error residual. Esto se produce porque la estimación final estable se encuentra por detrás de la posición real del objeto. La estimación irá alejándose de las cámaras hasta que encuentre la posición estable, aquella en la que alejarse más implicaría perder peso en las partículas.

El error de estimación proviene del tamaño de la zona de incertidumbre, como vimos en la figura 4.9. Según nos alejamos de las cámaras esta zona se va incrementando dado que las líneas de visión se van haciendo más paralelas, como muestra la figura 4.15. La estimación de profundidad queda sesgada hacia atrás porque el área de incertidumbre es más grande detrás del objeto que delante.

Aun contando con este problema, la dinámica en esta posición es similar a la del resto como podemos ver en la figura 4.16.

Sobre los resultados de convergencia influirá la dinámica propia de las partículas. Dicha dinámica está controlada por el modelo de movimiento, cuyo único parámetro libre es la desviación típica de la gaussiana. Este parámetro será el encargado de alejar las partículas de la posición actual o de concentrarlas sobre dicha posición. Usar una desviación típica alta permitirá cubrir más espacio mientras la nube de partículas realiza la búsqueda ciega del objeto.

Una vez localizado el objeto el sistema es capaz de seguir movimientos bruscos o rápidos del mismo. Como contrapartida produce un mayor rizado del error generado ya que los continuos saltos de posición hacen que la estimación varíe mucho, como

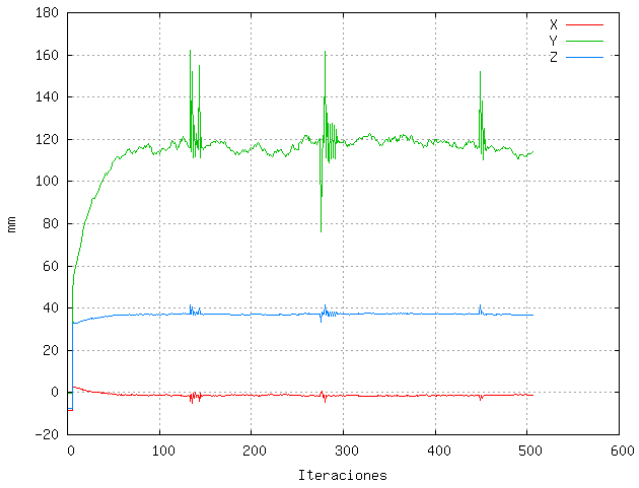

Figura 4.16: Convergencia de la estimación de la posición para un único objeto lejano.

puede apreciarse en la figura 4.17. En cambio, una desviación típica baja puede hacer que el sistema tarde mucho más en converger pero proporcionará mejores resultados en cuanto a error.

En la figura 4.17 podemos ver las curvas de error para un único objeto usando diferentes desviaciones típicas en cada caso. Empezando por una desviación típica igual a 1 centímetro vemos que, tras localizar el objeto (en las primeras iteraciones) el error converge muy lentamente. Por otro lado con una desviación típica de 40 centímetros vemos cómo aumenta el error residual de la estimación. Debemos recordar que los gráficos usados corresponden a casos en los que el sistema sí ha convergido, lo que sucede aproximadamente una vez de entre cada cinco. En resumen, cuanto más baja sea la desviación típica más probable será que o no converja o tarde mucho en hacerlo.

La desviación típica que hemos usado en los experimentos anteriores es de 20 centímetros. Esta también será la que usemos en el resto de experimentos de este capítulo. El motivo para utilizar este valor es porque proporciona un buen compromiso entre capacidad de seguimiento, convergencia y los errores de estimación.

El número de partículas es otro factor que influye, aunque no de manera tan directa, en la convergencia. En primer lugar, cuantas más partículas haya involucradas más probable será que se encuentre el objeto dentro del espacio de estados. Sin em-

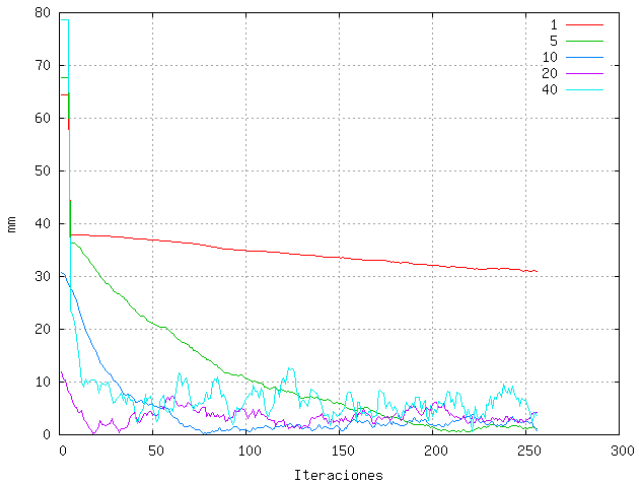

Figura 4.17: Error de convergencia con respecto a la desviación típica del modelo de movimiento.

bargo, también hay que tener en cuenta que según aumenta el número de partículas, disminuye el número de iteraciones por segundo que podemos procesar en el computador. En el cuadro 4.5 podemos ver cómo al aumentar el número de partículas se reducen las iteraciones por segundo. Esto se debe a que es necesario más tiempo para procesar todas las partículas existentes. Podemos ver que al aumentar en un orden de magnitud el número de partículas se reduce en un orden de magnitud la velocidad a la que funciona el filtro.

Número de partículas	Iteraciones por segundo
100	1130
200	647
500	250
750	151
1000	132

Cuadro 4.5: Velocidad de funcionamiento del algoritmo de condensación en función del número de partículas.

Sin embargo, aunque se reduzca el número de iteraciones por segundo, el número total de partículas procesadas por segundo se mantiene casi constante, algo más de 110.000 partículas. Aunque el número de partículas procesadas por segundo sea

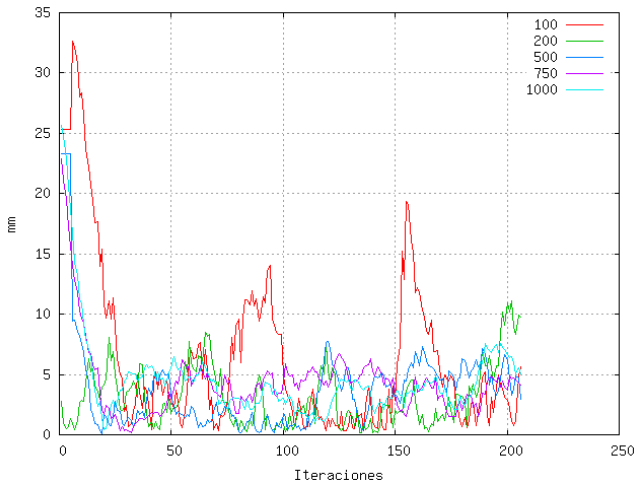

Figura 4.18: Error de convergencia con respecto al número de partículas.

similar, en el caso de reducir el número de partículas también se reduce la variedad de las partículas en juego por lo que se pueden perder zonas de interés.

En la figura 4.18 se muestran las curvas de error en función del número de partículas. Los valores probados van desde 100 partículas hasta las 1.000. Podemos ver que según disminuye el número de partículas, el sistema de seguimiento es más propenso a perder los objetos seguidos. Por lo demás, los cambios no son tan relevantes como los ajustes en la desviación típica del modelo de movimiento. Para el resto de experimentos hemos usado un número de partículas fijo igual a 1.000, dado que proporciona una mayor estabilidad y a un coste computacional aceptable.

Para comparar estos resultados hemos realizado los mismos experimentos usando el filtro de muestreo enfatizado. Las gráficas del error se muestran en la figura 4.19. En ellas puede verse cómo los errores son más o menos estables desde el principio. En algunos casos puede tardar unas pocas iteraciones para localizar el objeto correctamente, pero esto está relacionado con el tiempo que tarda el algoritmo de segmentación en considerar que existe un objeto en la escena, a la vista de las partículas colocadas sobre el mismo. Es interesante comparar estos resultados con la gráfica 4.14. Ambas gráficas están dibujadas con respecto a las iteraciones del algoritmo, pero los tiempos absolutos en cada una son diferentes. El filtro de condensación resulta

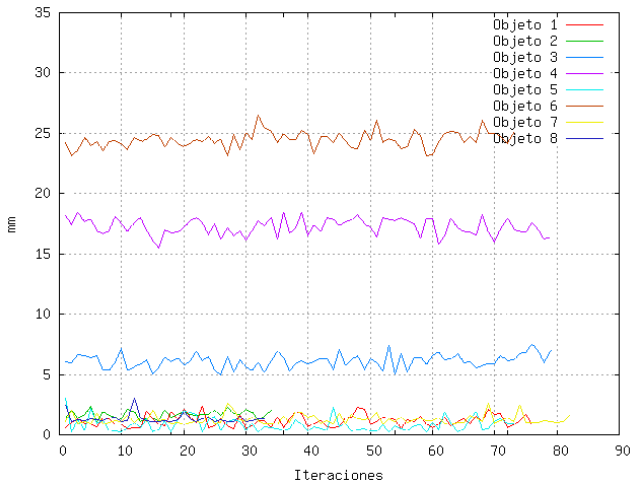

Figura 4.19: Evolución del error de estimación usando muestreo enfatizado.

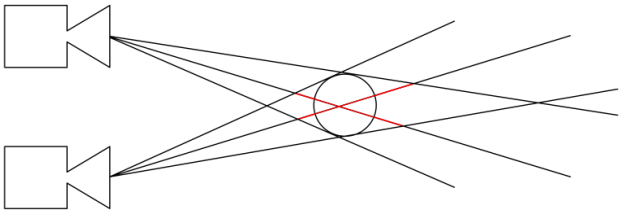

Figura 4.20: Zona de incertidumbre usando abducción con el muestreo enfatizado.

mucho más rápido que el de muestreo enfatizado por lo que, aunque necesite más iteraciones, los tiempos reales no resultan tan diferentes.

De nuevo el error medio está relacionado, principalmente, con la zona de incertidumbre asociada a cada uno de los objetos. El rizado del error se mantiene constante a lo largo del resto del experimento. De hecho sólo dos factores influirán en dicho rizado. El primero, y principal, la posición del objeto. Como hemos visto la posición del objeto influye sobre las partículas que caen en la zona de incertidumbre. En la figura 4.20 podemos ver cómo se modifica dicha zona en el caso de usar nuestra función de abducción. En este caso los puntos de interés caen sobre los segmentos, representados en rojo, de las líneas de proyección que se encuentran dentro de la zona de incertidumbre.

El número partículas usadas también influye en la velocidad del filtro, aunque

Número de partículas	Iteraciones por segundo
100	40
200	35
500	20
750	16
1000	15

Cuadro 4.6: Velocidad de funcionamiento del algoritmo de muestreo enfatizado en función del número de partículas.

no de manera tan grande como en el caso del filtro de condensación. En la tabla 4.6 podemos ver cómo cambia la velocidad de funcionamiento al ir cambiando el número de partículas. El cambio no es tan brusco porque la mayor parte del tiempo se pierde en el proceso de abducción que obliga a filtrar las imágenes completas (o una parte importante de las mismas).

En la figura 4.21 aparece la evolución del error de estimación. Vemos cómo al bajar el número de partículas aumentan la amplitud del rizado.

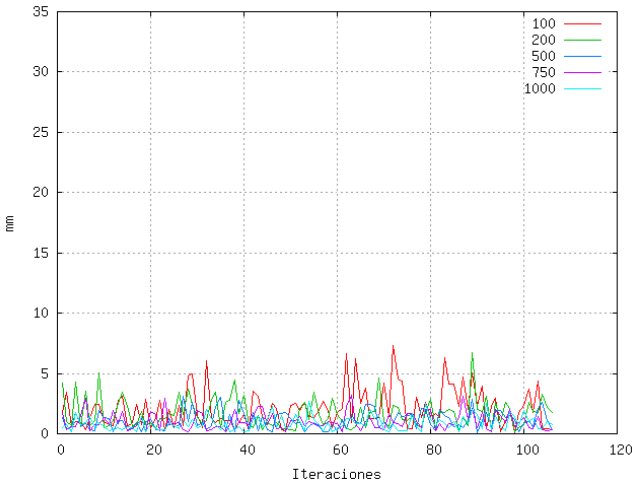

Figura 4.21: Evolución del error de estimación variando el número de partículas.

4.7.3. Experimentos de movimiento

Hasta ahora hemos mostrado experimentos estáticos. Estos tienen dos objetivos. En primer lugar, entender las dinámicas de funcionamiento de los algoritmos sin complicarlas con los problemas asociados al movimiento. En segundo lugar, las pruebas estáticas nos permiten realizar medidas de precisión sobre los resultados obtenidos con facilidad. Para completar estos resultados a continuación vamos a mostrar una serie de experimentos usando secuencias de vídeo con objetos en movimiento.

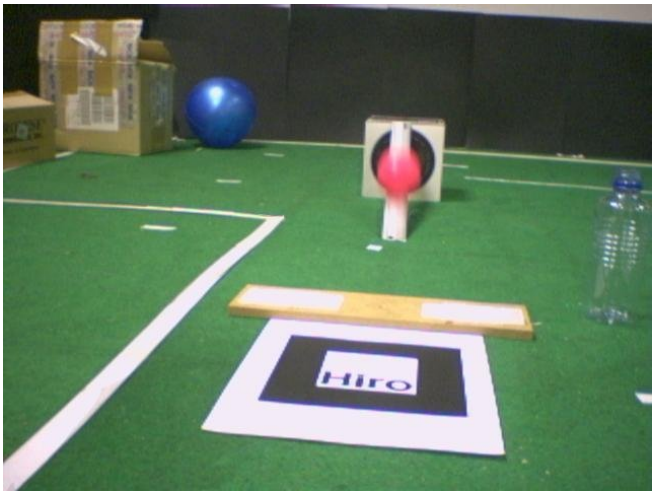

Figura 4.22: Captura del experimento de movimiento sobre una rampa.

La primera secuencia que vamos a considerar consiste en colocar una rampa orientada hacia las cámaras y dejar caer una pelota sobre ella. Repetiremos esta operación dos veces, cogiendo la pelota del suelo tras bajar de la rampa. El montaje se puede ver en la figura 4.22, tal y como se ve desde una de las cámaras. El objetivo es que el objeto a seguir tenga un movimiento continuo con aceleración constante. En la figura 4.23 podemos ver los resultados del seguimiento, separados por coordenadas. El eje Y, alineado parcialmente con la rampa, es el menos preciso de los tres, como vimos anteriormente. Sin embargo, el seguimiento resulta suave en esa coordenada. Los movimientos de las otras coordenadas se deben a que la rampa no se encuentra perfectamente alineada con el eje Y.

El seguimiento usando el algoritmo de condensación puede calificarse como suave. En la figura 4.24 aparecen los resultados del mismo experimento usando mues-

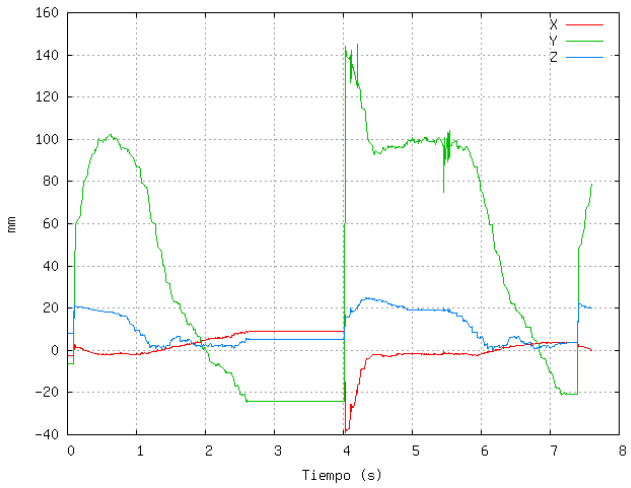

Figura 4.23: Seguimiento de un objeto sobre una rampa con filtro de condensación.

treo enfatizado. En este caso aparece mucho más ruido, haciendo el seguimiento más abrupto. Esta es una de las características importantes de los dos métodos de seguimiento. El primer algoritmo, debido a su naturaleza secuencial, presenta menos errores y rizados cuando sigue la pelota. El segundo algoritmo no tiene esta naturaleza secuencial y realiza la estimación únicamente con la información instantánea disponible. Esto le hace mucho más sensible a los errores que el anterior, dado que cada vez debe volver a localizar el objeto.

Para seguir comprobando la diferencia entre los dos métodos de seguimiento vamos a emplear otra secuencia de vídeo. En este caso movemos una pelota sobre una superficie plana, en la dirección del eje X. Para simplificar la comprobación de los resultados vamos a ir moviendo la pelota con una mano y deteniéndola en unas posiciones fijas separadas, aproximadamente, unos cinco centímetros entre sí, como ilustra la figura 4.25. Tras recorrer todas las posiciones volvemos a colocar la pelota en el principio y nos movemos con saltos más grandes. La duración de la secuencia total es más larga que la anterior, unos 30 segundos.

Los resultados del seguimiento con el algoritmo de condensación se muestran en la figura 4.26. De nuevo vemos cómo los movimientos son suaves en el eje X, deteniéndose en las posiciones fijas marcadas. Durante el paso de una posición a la siguiente el objeto está parcialmente tapado por la mano que lo coge. Además vemos

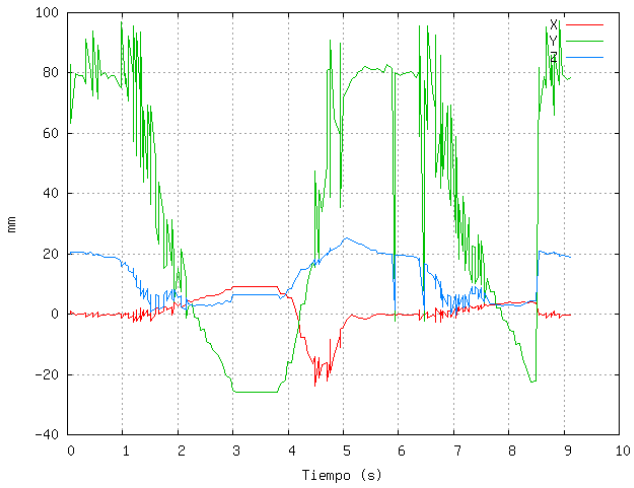

Figura 4.24: Seguimiento de un objeto sobre una rampa con muestreo enfatizado.

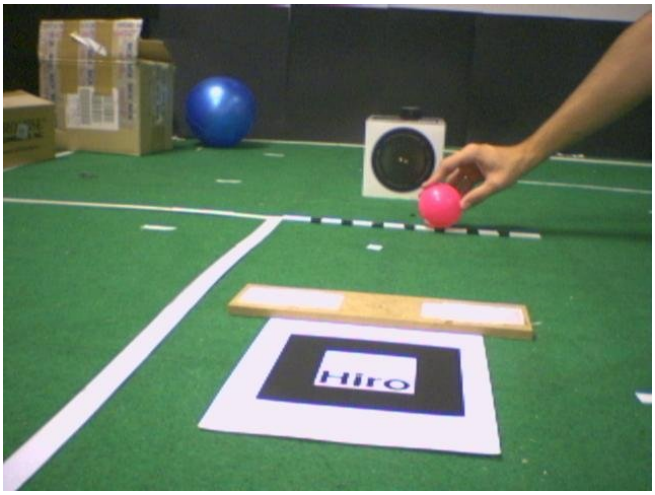

Figura 4.25: Captura del experimento de movimiento sobre una superficie plana.

cómo los movimientos largos sufren de la deriva sistemática. Un movimiento brusco puede hacer que unas cuantas partículas pierdan el objeto en una de las cámaras y comience a alejarse de las cámaras. Esto aparece en la figura como los picos altos (llegando hasta 1,6 metros) en el eje Y.

También observamos ciertos picos en el eje Z. La explicación en este caso el movimiento brusco ha hecho que la nueva posición en una de las dos cámaras se coloque delante de la posición correcta. El objeto se localizará navegando por la epipolar hasta

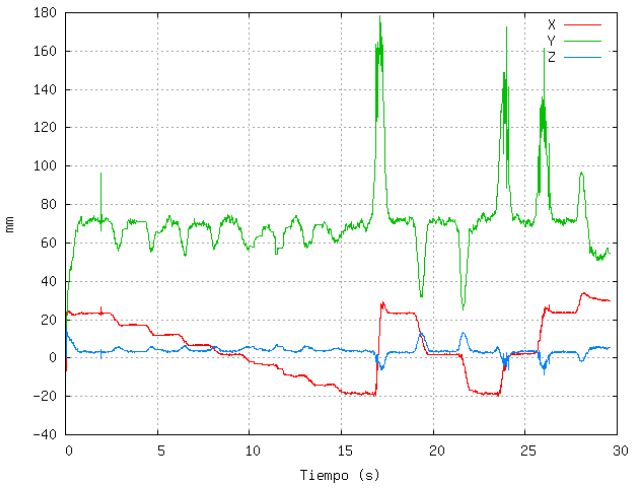

Figura 4.26: Seguimiento de un objeto sobre una superficie plana con filtro de condensación.

la posición correcta.

El seguimiento con muestreo enfatizado se emplea en la misma secuencia de vídeo. Los resultados aparecen en la figura 4.27. Al igual que pasaba durante el seguimiento en la rampa, los resultados son mucho más ruidosos que en el caso del filtro de condensación. Aunque hay errores de estimación instantáneos, la posición media es, de nuevo, correcta.

Podemos concluir que este filtro es capaz de localizar correctamente las zonas de interés para el seguimiento, pero que es incapaz de proporcionar unos resultados estables y suaves por sí mismo.

Para terminar con los experimentos de movimiento libre, hemos realizado otro dejando caer una pelota sobre el suelo. Al botar la pelota crea un movimiento rápido para la tasa de captura usada. En la figura 4.28 podemos ver una secuencia de imágenes usadas en el vídeo. Los fotogramas mostrados son consecutivos tomados con la misma cámara. Podemos ver como el rápido movimiento de la pelota al saltar hace que de un fotograma al siguiente haya recorrido un espacio importante. Además se puede apreciar cómo la pelota aparece difuminada en las imágenes. Otro dato interesante de la secuencia de vídeo es que durante unos instantes la pelota sale del campo de visión de las cámaras durante uno de los botes. En este caso la secuencia es más corta que las anteriores dado que únicamente nos queremos centrar en los botes de la

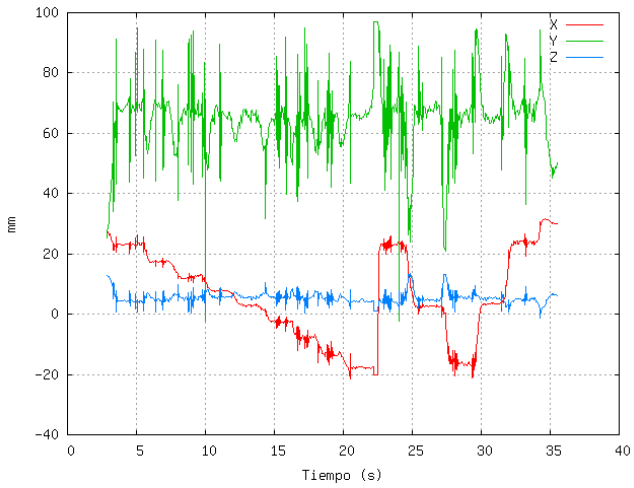

Figura 4.27: Seguimiento de un objeto sobre una superficie plana con muestreo enfatizado.

pelota. La duración es de unos dos segundos aproximadamente.

La figura 4.29 muestra los resultados de seguir los saltos usando el filtro de condensación. En las gráficas podemos ver cómo la pelota baja y vuelve a subir varias veces, rebotando en el suelo (eje Z en azul). La primera subida que aparece en las gráficas se corresponde con la convergencia del algoritmo, no con un verdadero bote. La secuencia de vídeo comienza con la pelota apareciendo por la parte de arriba de las imágenes (coordenada Y alta), pero las partículas están inicializadas cerca el origen de coordenadas.

También vemos cómo la pelota cae hacia atrás y hacia la izquierda (Y positivas y X negativas). De los gráficos es interesante señalar que los saltos de estimación son escalonados. Estos escalones se producen cada vez que hay una nueva imagen disponible. El algoritmo es capaz de ejecutar varias iteraciones por cada fotograma, estabilizando la estimación hasta que hay una nueva imagen disponible. En ese momento se produce el salto hasta la siguiente posición estable. El funcionamiento del filtro es mucho más rápido que la velocidad de captura, siendo ésta el verdadero límite para la velocidad de seguimiento. También vemos cómo aproximadamente al medio segundo del vídeo, la estimación se estabiliza durante varios fotogramas. Es en este momento cuando la pelota desaparece del campo de visión de las cámaras.

Los resultados con muestreo enfatizado son más ruidosos que los anteriores,

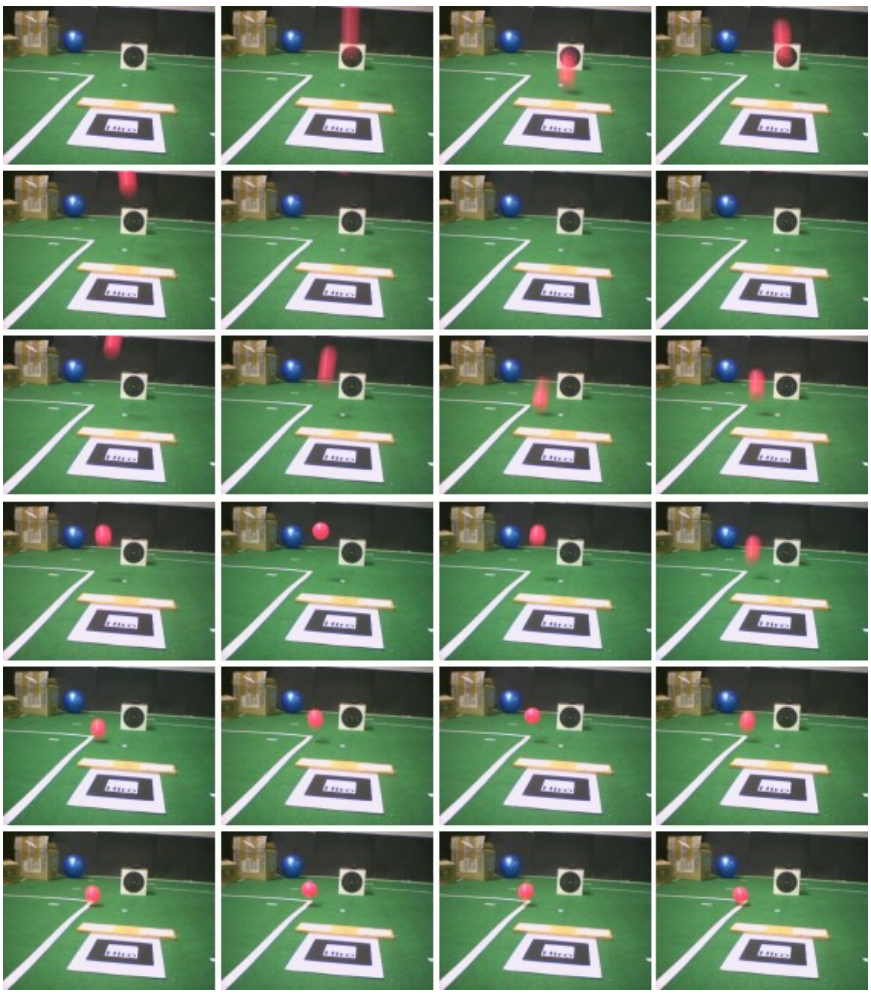

Figura 4.28: Secuencia de imágenes correspondientes al bote de una pelota sobre el suelo.

como puede apreciarse en la figura 4.30, aunque en la coordenada Z podemos apreciar los diferentes saltos que hace la pelota, el ruido de estimación es mucho más alto en este caso. La estimación en Y salta continuamente y el seguimiento no es suave.

4.7.4. Experimentos con múltiples objetos

En el siguiente experimento vamos a comprobar cómo se enfrentan los algoritmos estudiados al problema del seguimiento de múltiples objetos. Para ello hemos diseñado un sencillo experimento. Colocamos dos pelotas cualesquiera sobre el suelo

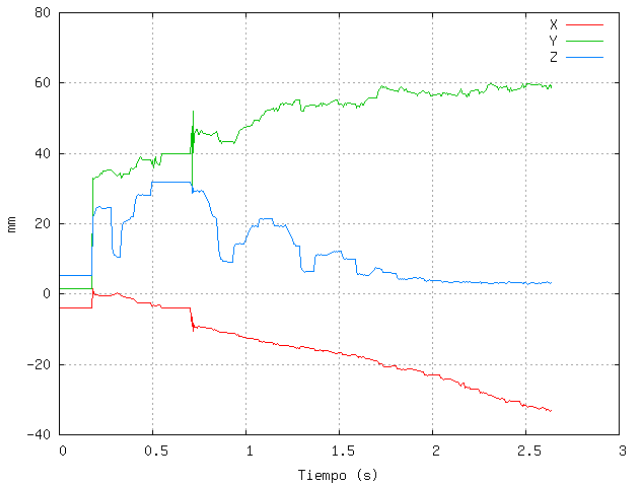

Figura 4.29: Seguimiento de un objeto moviéndose rápidamente con filtro de condensación.

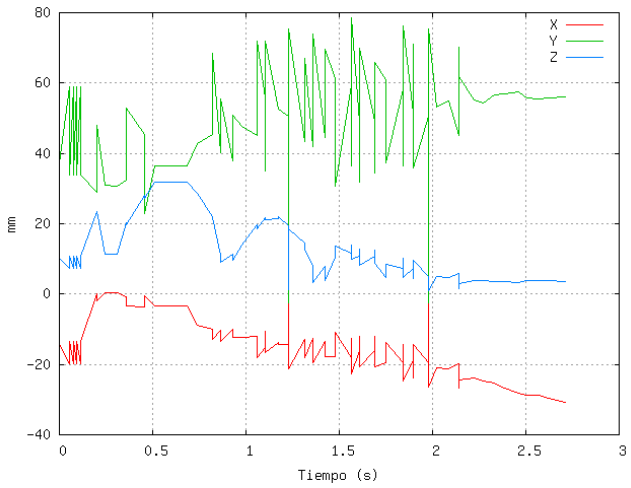

Figura 4.30: Seguimiento de un objeto moviéndose rápidamente con muestreo enfatizado.

y observamos cuáles son los resultados moviéndolas por el espacio. Lo que queremos ver es cómo se distribuyen las partículas a la vista de los dos objetos.

En la figura 4.31 podemos ver la estimación que hace el algoritmo de condensación de los objetos de la escena, que se representa por el cuadrado verde que aparece sobre la pelota de la izquierda. Únicamente es capaz de detectar uno de los dos. La nube de partículas se abre hasta que una de ellas proyecta bien en una de las cámaras. En ese momento el proceso de remuestreo descartará todas las otras partículas

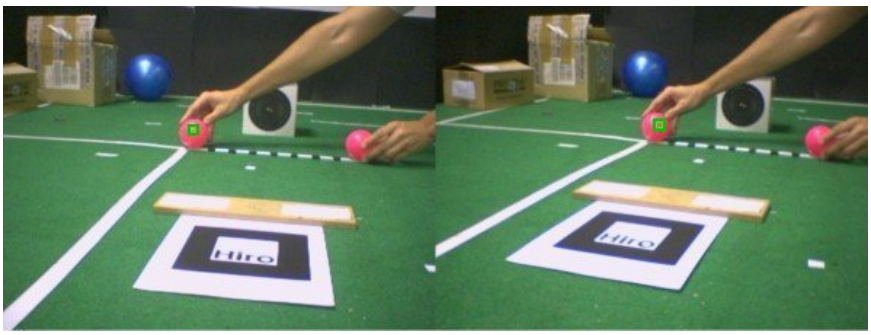

Figura 4.31: Resultados de localización de dos objetos usando el algoritmo de condensación.

para centrarse en esa única posición. La pelota concreta que se localiza depende de la evolución aleatoria que hayan seguido las partículas desde su inicialización. Será más probable caer sobre el objeto que se encuentre más cerca del punto de inicialización. Nunca se mantienen dos poblaciones a largo plazo, una sobre un objeto y la otra sobre el otro. El proceso de remuestreo y el número limitado de partículas hace que se converja únicamente a uno de los dos. Además, el sistema no saldrá de esa posición, dado que la solución es estable de por sí. En el caso de que los objetos se crucen al moverse el sistema podrá cambiar de objeto, pero no será capaz de mantener las dos poblaciones de modo permanente. En resumen, la naturaleza aleatoria de la localización y el estancamiento del sistema en sólo uno de los objetos, hacen del filtro de condensación un algoritmo totalmente inapropiado para el seguimiento de múltiples objetos.

En cambio el algoritmo de muestreo enfatizado sí que es capaz de localizar las dos pelotas al mismo tiempo. Esto se debe a que coloca partículas en todas las regiones del espacio que podrían contener un objeto, incluyendo las dos posiciones reales. En la figura 4.32 podemos ver cómo es capaz de, incluso, seguir los movimientos de las dos pelotas al mismo tiempo. El gráfico muestra únicamente la coordenada X del movimiento mientras desplazamos las dos pelotas sobre el suelo. A modo de comparación también se muestran los resultados que proporciona condensación, que sólo es capaz de seguir a uno de los dos objetos.

El problema del muestreo enfatizado es que realiza un seguimiento abrupto, como vimos, por ejemplo, en la figura 4.30. Esto hace que sea poco adecuado pa-

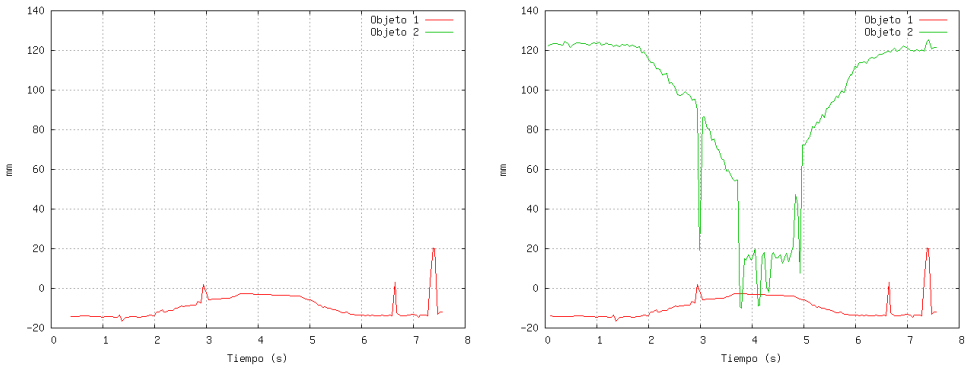

Figura 4.32: Posición estimada (eje X) de dos objetos con filtro de condensación (izquierda) y con muestreo enfatizado (derecha).

ra un problema de seguimiento, ya sea de uno o varios objetos.

A continuación, en el capítulo 5, abordaremos este problema en profundidad, presentando una alternativa a estos algoritmos que combina las ventajas de cada uno de ellos sin los inconvenientes mencionados.

CAPÍTULO 5

Seguimiento De Varios Objetos

Este capítulo recoge el objetivo principal de esta tesis, que es mostrar cómo se pueden modificar los filtros de partículas básicos, con los que hemos trabajado anteriormente, para que sean capaces de seguir varios objetos al mismo tiempo. Estamos interesados en el caso particular de que los objetos sean indistinguibles entre sí. En caso contrario, como hemos visto en el capítulo 2, podríamos emplear otras técnicas diferentes a las que vamos a tratar a continuación.

5.1. Función de densidad de probabilidad para varios objetos

Existen varias aproximaciones para abordar el problema del seguimiento multiobjeto, como comentamos en el capítulo 2. Nosotros estamos interesados en las soluciones que representan varios objetos empleando el mismo espacio de estados que se usaría para un único objeto. Es decir, la solución de múltiples objetos será un conjunto de puntos independientes tomados del espacio de estados. De esta forma se pueden representar a la vez tantos objetos como sean necesarios, siendo cada uno un punto (o región) en el espacio de estados.

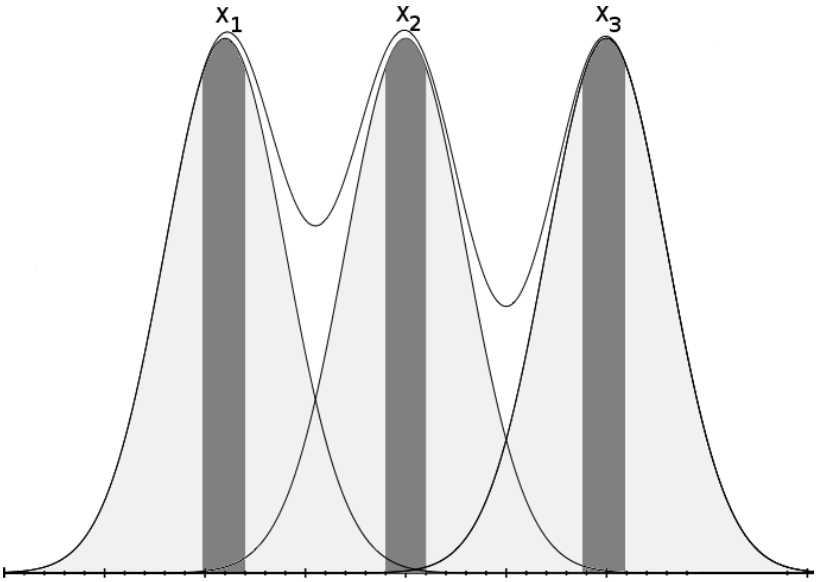

Figura 5.1: Distribución de ocupación $p(\boldsymbol{x})$ con múltiples objetos.

Teniendo esto en cuenta podemos pensar en la distribución $p(\boldsymbol{x})$ como una distribución de probabilidad de ocupación a lo largo de una determinada área del espacio de estados, $\boldsymbol{x}$. Allá dónde exista ocupación tendremos indicios de la presencia de un objeto. La estimación de la posición de cada uno de los objetos estará relacionada con la búsqueda de los máximos locales o regiones compactas en la distribución *a posteriori*, $p(\boldsymbol{x}|\boldsymbol{z})$.

A modo de ejemplo podemos fijarnos en la figura 5.1, que muestra una distribución de probabilidad $p(\boldsymbol{x})$ con tres máximos locales. Cada uno de estos máximos corresponde a un objeto en una determinada posición, $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ y $\boldsymbol{x}_3$. La distribución $p(\boldsymbol{x})$ indica la ocupación del espacio en función de $\boldsymbol{x}$ y es la combinación de las distribuciones de probabilidad de la posición de los tres objetos, como se muestra en la figura.

Si interpretamos de esta forma el problema, nuestro objetivo consistirá en obtener la mejor representación de la distribución de probabilidad *a posteriori* $p(\boldsymbol{x}|\ \boldsymbol{z})$. En ella estará toda la información que necesitamos. Hemos pasado de un problema de estimación de posición a un problema de estimación de distribución de probabilidad.

Como vimos en la sección 3.3, los filtros de partículas son también un método

propicio para realizar la estimación de distribuciones de probabilidad. Sin embargo, la convergencia a la distribución correcta sólo se asegura cuando el número de partículas tiende a infinito. Por este motivo será necesario estudiar un poco más a fondo su comportamiento, para que funcione con un número limitado de partículas.

Podríamos haber optado por otras alternativas, como puede ser la utilización de filtros de Kalman para representar la distribución de probabilidad o de modelos analíticos o polinómicos. Estas aproximaciones tienen la desventaja de ser mucho menos flexibles que los filtros de partículas. Por ejemplo, los filtros de Kalman únicamente son capaces de representar distribuciones gaussianas, por lo que no son adecuadas para representar varios objetos al mismo tiempo. Los modelos polinómicos necesitan saber *a priori* el orden del modelo, con el fin de poder representar el número correcto de máximo locales, información que, en principio, desconocemos y será variable con respecto al tiempo.

Los métodos de Monte Carlo no sufren de estas limitaciones. Al utilizar una aproximación muestreada, resultan mucho más flexibles para representar distribuciones de probabilidad con un número cambiante de máximos locales, como las que nos interesan. De esta manera podremos desarrollar un marco integrado para representar las distribuciones de probabilidad que nos permite la búsqueda eficiente de los máximos locales, repartiendo las partículas entre las diferentes zonas de interés que vayan apareciendo.

La información que nos interesa de la distribución de probabilidad de ocupación es la colocación de los máximos locales, las zonas dónde pueden existir objetos. Si sólo estamos interesados en dichos máximos, únicamente necesitamos representar con precisión estas zonas de la distribución de probabilidad. Dicho de otra forma, queremos localizar y estimar correctamente las zonas de las distribución de probabilidad que son interesantes, es decir, aquellas que contienen información sobre las posiciones de los objetos.

Cuantas más partículas coloquemos en esas zonas, mejor serán las representaciones de los objetos situados en cada posición. Al mismo tiempo será necesario mantener un número mínimo de partículas en cada máximo con el fin de no perderlos, evitando que se concentren en un subconjunto de máximo. De esta forma podemos interpretar las partículas como recursos de búsqueda, que se irán repartiendo para

obtener buenas estimaciones o para descubrir nuevos máximos locales.

5.2. Limitaciones de los filtros de partículas

Las dos aproximaciones estudiadas en el capítulo 4 presentan importantes limitaciones a la hora de enfrentarse al problema del seguimiento de varios objetos al mismo tiempo tal y como lo hemos descrito. En particular, la sección 4.7.4 muestra con un experimento real en el que se ve cómo los dos sistemas fallan al enfrentarse a este problema.

En el caso del filtro de condensación la limitación reside en cómo se gestiona su tendencia monomodal. Al aparecer varios objetos indistinguibles entre sí, habrá varias zonas del espacio de estados que resultan igualmente válidas para el filtro. El modelo de observación no será capaz de separarlos, por lo que siempre existirá esta discrepancia sobre qué posición es la correcta. Al haber realmente varios objetos, varias posiciones lo serán. El filtro de condensación podrá mantener varias hipótesis de manera temporal, mientras se encuentran indicios para elegir una solución entre todas las alternativas. Pero a la larga se quedará únicamente en una de las posiciones.

Esto se debe a que el algoritmo de condensación coloca las partículas de manera ciega a la nuevas observaciones. Para colocarlas únicamente emplea el modelo de movimiento, $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$, ignorando por completo la información visual. El modelo de movimiento produce un paseo aleatorio por el espacio de estados. Al no tener información sobre la verdadera posición de los objetos, las partículas pueden tanto alejarse como acercarse a las posiciones correctas. El modelo de observación, posteriormente, se encargará de ajustar los pesos de las partículas para descartar aquellas mal colocadas.

Como vimos en el capítulo 3, el algoritmo de condensación realiza un remuestreo de las partículas en cada iteración. El remuestreo escoge las partículas en función de sus pesos, con el fin de elegir aquellas que parecen más relevantes y así optimizar los recursos disponibles.

Sin embargo, el proceso de remuestreo se hace a costa de perder partículas en zonas del espacio que, aún resultando interesantes, el modelo de movimiento no ha seguido correctamente. Al ir perdiendo partículas en uno de los objetos, el remuestreo

tomará otro como más probable. Esto sucede aún partiendo de diferencias pequeñas debido al carácter discreto del número de partículas. Nuestro objetivo consiste en colocar partículas en todas esas zonas y no únicamente en una de ellas.

Para hacer seguimiento de múltiples objetos debemos evitar perder esta información de la población de partículas, pero asegurando una degeneración mínima. La información se debe mantener con una población estable de partículas en cada una de las zonas que resulten interesantes en el espacio de estados. Estas poblaciones, o subconjuntos de partículas, estarán relacionadas con cada uno de los modos de la distribución de probabilidad $p(\boldsymbol{x}|\boldsymbol{z})$.

Este sistema además impide la exploración de zonas desconocidas. La exploración se hace a través del paseo aleatorio, pero en cuanto una partícula abandone la posición del objeto seguido, será eliminada por el paso de remuestreo. Así será imposible encontrar ningún objeto que no se cruce en la trayectoria del que estamos siguiendo.

Aparte de colocar las partículas en las zonas correctas, el sistema debe ser capaz de mantener un grupo de ellas de forma estable sobre cada objeto. Con esto queremos decir que las regiones de interés deben estar representadas de manera consistente de una iteración a la siguiente. En caso de que esto fuese así se perderá información sobre la posición real de los objetos. Una vez asegurada esta estabilidad, es posible obtener una representación de la distribución *a posteriori* suficientemente fiel. De esta distribución se puede inferir la posición de cada uno de los objetos buscando los máximos locales. Los recursos de búsqueda sobrantes (partículas) se pueden emplear para buscar nuevos objetos que aparezcan en la escena.

Para estabilizar la población de partículas podemos modificar tanto la distribución de propuesta como el método de remuestreo. Técnicas como el filtro de condensación no nos sirven, porque usan una búsqueda ciega en el espacio de estados, desperdiciando muchos recursos, necesarios para tener una población estable. La utilización del remuestreo en este algoritmo es útil, dado que permite ir buscando las zonas más prometedoras de manera rápida hasta llegar a una situación de estabilidad. En caso de que existan varios objetos en la escena este filtro no asegura que se encuentre uno en concreto ni que se localice el más grande de ellos.

El muestreo enfatizado, por otro lado, sí que es capaz de enfrentarse este pro-

blema en particular con cierto éxito. Gracias a la función de abducción que hemos incluido, detecta todos los objetos presentes y es capaz de añadir, incluso, nuevos objetos según vayan apareciendo proporcionando estimaciones suficientemente buenas para cada una de las posiciones. Sin embargo, la función de propuesta está basada únicamente en información visual, lo que nos lleva a una formulación no secuencial del filtro. Esto hace que el seguimiento usando este algoritmo no sea tan suave como en el caso anterior.

En realidad, ni siquiera podemos hablar de "seguimiento" como tal, dado que el algoritmo realiza un descubrimiento para cada instante de tiempo, construyendo la estimación sin tener en cuenta la información almacenada hasta ese momento. Aunque los resultados en seguimiento no sean buenos, los errores de estimación son aceptables en media. La ventaja principal de este método es su capacidad de convergencia instantánea y la ausencia de cualquier tipo de deriva perniciosa.

El problema principal de este algoritmo aparece a la hora de emparejar la información extraída de cada una de las cámaras. El emparejamiento resultará complejo dado que no sabemos, *a priori*, cuáles son las correspondencias entre las dos cámaras. Será necesario explorar todas las posibles combinaciones para buscar la solución correcta, con la consiguiente complejidad que esto conlleva.

Sin embargo, aunque busquemos con todas las posibles combinaciones, puede darse el caso en el que no seamos capaces de desambigüar entre varias soluciones posibles. Por ejemplo, en la parte izquierda de la figura 5.2, sin tener más información, no podemos saber cuántos objetos hay realmente (2, 3 o 4), ni cuáles son las posiciones reales en las que se encuentran. Esto se debe a que no somos capaces de encontrar el emparejamiento correcto entre una imagen y otra.

Como estamos trabajando con objetos indistinguibles entre sí, esta ambigüedad no podrá ser eliminada sin añadir información extra. Por este motivo necesitamos un sistema de seguimiento que sea capaz, por lo menos, de lidiar con estas incertidumbres en cuanto al emparejamiento.

Una posible solución al problema de estos "objetos fantasma" consiste en colocar soluciones hipotéticas y esperar a confirmarlas o desecharlas a lo largo del tiempo en una aproximación secuencial. Si tomamos la figura 5.2, en la que realmente sólo hay dos pelotas del mismo color, al moverse una de las pelotas sólo 2 de las 4 posi-

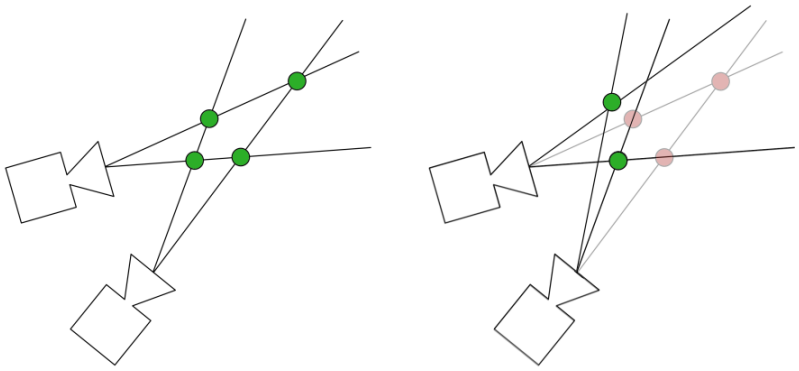

Figura 5.2: Ambigüedad en la posición de varios objetos en 3D. La continuidad en los movimientos elimina dicha ambigüedad.

bles posiciones tienen sentido. En la parte derecha de la figura vemos cuáles son las posiciones correctas tras el movimiento de la pelota. Esto puede hacerse gracias a la continuidad espacial del movimiento, pero el filtro de muestreo enfatizado no la tiene en cuenta. porque de una iteración a la siguiente no guarda la información sobre las posibles posiciones actuales de los objetos.

El filtro de condensación sí resuelve este problema porque realiza una serie de hipótesis que se van desechando y actualizando siguiendo un proceso iterativo, usando la información que está disponible en cada instante para resolver las ambigüedades que existían anteriormente.

5.3. Combinación de filtro secuencial y no secuencial

La solución que proponemos para seguir varios objetos al mismo tiempo consiste en combinar estas dos técnicas, el filtro de condensación y el muestreo enfatizado, complementando las carencias de cada una con la otra. De esta forma, podremos mantener las zonas de interés en la función de distribución *a posteriori* estimada. El muestreo enfatizado nos proporciona información acerca de dónde se encuentran los objetos, añadiendo los nuevos tan pronto como aparecen. El algoritmo de condensación será capaz de desechar los errores del muestreo enfatizado, centrándose en las

zonas de interés.

Como intuición podemos pensar que el muestreo enfatizado realiza una búsqueda global y el algoritmo de condensación una búsqueda local sobre las zonas de interés.

La estructura de los dos filtros es similar. Ambos mantienen una población de partículas en cada iteración. Dicha población es la que alberga toda la información acerca de la distribución objetivo. La forma que hemos elegido para combinar la información de los dos filtros consiste en combinar, precisamente, las partículas. Creamos para ello una nueva población que contenga las partículas de cada filtro, obtendremos una distribución que aproxime tanto las zonas nuevas como las zonas ya conocidas.

La combinación de los dos permite que el algoritmo de condensación tenga siempre un número mínimo de muestras en cada objeto, evitando que se pierdan durante el remuestreo. Condensación se encargará de explotar la búsqueda en cada zona, encontrado la mejor estimación posible. Cuando aparezca un nuevo objeto, la abducción colocará partículas sobre él y, al tener un número suficiente de partículas, el algoritmo de condensación optimizará la estimación para ese nuevo objeto.

La idea de combinar varios filtros de partículas no es nueva, como hemos comentado en el capítulo 2. Por ejemplo, Blake indica en [IB98c, IB98b] diferentes formas de combinar modelos de movimiento diferentes y distribuciones de probabilidad. Por otro lado Koller-Meier presenta en [KMA01] un marco para combinar el filtro de condensación y el filtro de muestreo enfatizado similar al que vamos a usar a continuación. En este caso la combinación de las distribuciones se realiza en la función de propuesta, sin integrarla en la función *a posteori*, como hacemos nosotros, lo que hace que no sea posible calcular los pesos de importancia de manera directa. Okuma [OTF+04] también emplea una combinación de dos funciones de probabilidad en la función de propuesta, una para colocar nuevos objetos y otra para seguir los objetos actuales. Estos trabajos sirven como inspiración al que presentamos a continuación y en ellos se presentan algunas de las expresiones que comentaremos.

En el capítulo 3 vimos los diferentes desarrollos matemáticos de los filtros de partículas. El muestreo enfatizado y condensación pretenden aproximar, en las zonas de interés, la función de densidad de probabilidad $p(\boldsymbol{x}_k \mid \boldsymbol{z}_k)$. A continuación nos referiremos como $\hat{p}_{NS}(\boldsymbol{x}_k \mid \boldsymbol{z}_k)$ a la aproximación muestral no secuencial que nos da

el muestreo enfatizado y $\hat{p}_S(\boldsymbol{x}_k|\ \boldsymbol{z}_k)$ a la aproximación obtenida con condensación. Usando estas dos aproximaciones podemos construir una tercera aproximación a la verdadera distribución de probabilidad $p(\boldsymbol{x}_k|\boldsymbol{z}_k)$ como una combinación lineal de la siguiente forma:

$$\hat{p}(\boldsymbol{x}_k|\boldsymbol{z}_k) = (1-\alpha)\hat{p}_S(\boldsymbol{x}_k|\boldsymbol{z}_k) + \alpha\hat{p}_{NS}(\boldsymbol{x}_k|\boldsymbol{z}_k) \tag{5.1}$$

$\hat{p}(\boldsymbol{x}_k|\boldsymbol{z}_k)$ estará formada por partículas que proceden de cada uno de los filtros. El peso α, $0 \leq \alpha \leq 1$, sirve para ajustar los pesos de cada una de la dos aproximaciones, convirtiendo a $\hat{p}$ en una verdadera distribución de probabilidad. Los casos en los que α sea igual a 0 o a 1 son los del filtro de condensación y el muestreo enfatizado respectivamente.

Dado que las dos partes de esta estimación están compuestas por partículas, lo que obtenemos es un nuevo conjunto cuyas partículas están colocadas siguiendo los dos algoritmos al mismo tiempo. Por un lado, tendremos partículas en todos los máximos locales, colocadas por el filtro no secuencial, y por otro, partículas que explotarán las zonas conocidas usando el filtro de condensación.

La parte no secuencial de la distribución *a posteriori*, $\hat{p}_{NS}$, puede aportar partículas a $\hat{p}$ sin impedimento alguno. Esto se debe a que en cada iteración no necesita información sobre la iteración anterior, sino que las coloca únicamente en función de la observación en ese instante de tiempo, $\boldsymbol{z}_k$.

La parte secuencial de la distribución *a posteriori*, p_S, en cambio, requiere de alguna consideración adicional. Para evitar los problemas con la degeneración de los pesos de importancia, el filtro de condensación realiza un remuestreo de los pesos desde su estimación de p. Esto permite que los pesos se normalicen todos al mismo valor. En nuestro caso podemos tomar la estimación de p que hemos creado, $\hat{p}$. Si se trata de una aproximación válida de p es posible tomar partículas de ella sin interferir en el comportamiento del filtro de condensación. De esta forma se consigue que este filtro emplee muestras que proceden tanto de la iteración pasada (optimización local) como partículas propuestas por la función de abducción (búsqueda global).

Empleando las partículas obtenidas podemos estimar $\hat{p}$ al igual que hacíamos anteriormente:

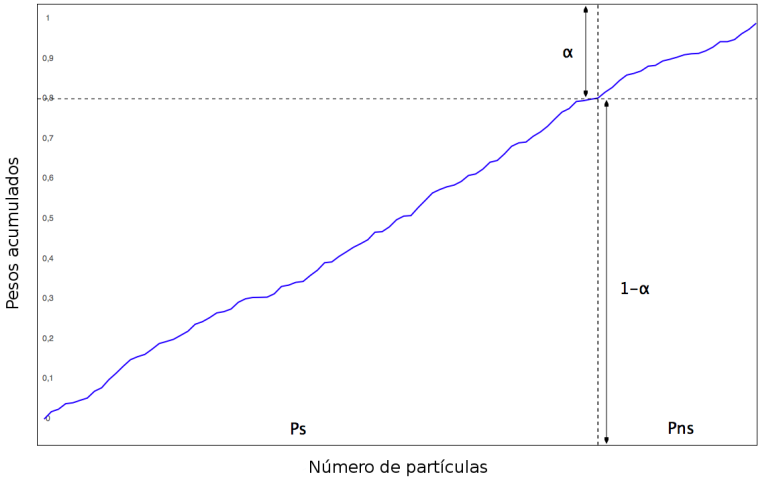

Figura 5.3: Diagrama de pesos acumulados para emplear en el remuestreo.

$$\hat{p} = (1-\alpha)\sum w_S^i \delta(\boldsymbol{x} - \boldsymbol{x}_S^i) + \alpha \sum w_{NS}^i \delta(\boldsymbol{x} - \boldsymbol{x}_{NS}^i) \tag{5.2}$$

Para muestrear de $\hat{p}$ usaremos el método de la ruleta. En el funcionamiento habitual los pesos del filtro de condensación y del muestreo enfatizado suman uno. El factor de corrección α se puede emplear para corregir los pesos, introduciéndolo en los sumatorios. De esta forma obtendríamos la siguiente combinación:

$$\hat{p} = \sum w_S^i \delta(\boldsymbol{x} - \boldsymbol{x}_S^i) + \sum w_{NS}^i \delta(\boldsymbol{x} - \boldsymbol{x}_{NS}^i) \tag{5.3}$$

dónde w_S sumaría $(1-\alpha)$ y w_{NS} sumaría α. Una vez corregidos los pesos sería directo aplicar el método de la ruleta sobre el nuevo conjunto de partículas para obtener la nueva población. En la figura 5.3 podemos ver una explicación gráfica a este proceso. Pueden verse las zonas que proceden del filtro de condensación y del muestreo enfatizado. La probabilidad de tomar una partículas de cada parte es $(1-\alpha)$ y α respectivamente.

El hecho de emplear los pesos $(1-\alpha)$ y α hace que, siguiendo el método de la ruleta, obtengamos de media $(1-\alpha)M$ partículas procedentes de la parte secuencial del algoritmo y αM de la parte no secuencial, siendo M el número de partículas de la parte secuencial. Por lo tanto el parámetro α nos servirá para modular el algoritmo

entre sus dos extremos: optimización local y obtención de nuevos objetos. Durante los experimentos (sección 5.5) realizaremos diferentes pruebas para ver cómo afecta realmente el valor de α al comportamiento del filtro.

Podemos tener cualquier número de partículas en cada componente de $\hat{p}$, $\hat{p}_S$ y $\hat{p}_{NS}$. Sin embargo, el número máximo de partículas que podemos mantener depende de la capacidad de cómputo que tengamos disponible. Será necesario repartir las partículas entre las dos partes del algoritmo de alguna manera. Nosotros las repartimos teniendo en cuenta el remuestreo que usaremos en la parte de condensación. De esta forma colocaríamos $M = (1-\alpha)N$ partículas en la parte $\hat{p}_S$ y el resto, esto es αN partículas, en la parte $\hat{p}_{NS}$. Aunque otra elección podría ser igualmente válida. Manteniendo esta distribución de partículas podemos ver las diferentes fases del sistema en la figura 5.4.

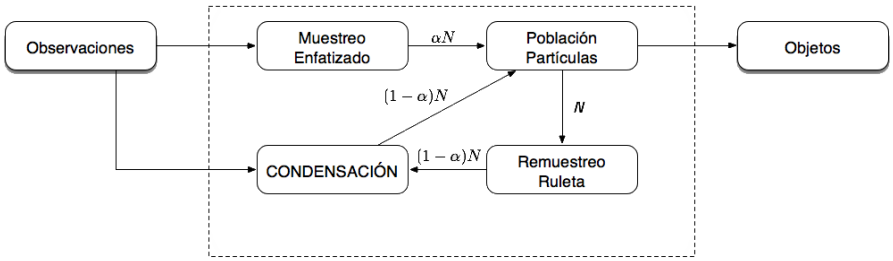

Figura 5.4: Diagrama de funcionamiento del algoritmo de estimación multimodal.

Hasta ahora no hemos abordado el cálculo de los pesos de cada componente. Dicho cálculo será análogo al empleado en el filtro de condensación y en el de muestreo enfatizado, siguiendo el formalismo de Monte Carlo, dado que dicho formalismo no se ha modificado. El único cambio será añadir el factor de corrección $(1-\alpha)$ y α en cada caso.

Por un lado los pesos de la parte secuencial se calculan según la expresión 3.29:

$$w_k^i = \frac{p(\boldsymbol{x}_k^i|\ \boldsymbol{z}_k)}{q_S(\boldsymbol{x}_k^i\ |\ \boldsymbol{x}_{k-1}^i, \boldsymbol{z}_k)} = \frac{p(\boldsymbol{z}_k|\ \boldsymbol{x}_k^i)}{p(\boldsymbol{z}_k)} \cdot w_{k-1}^i \propto p(\boldsymbol{z}_k|\ \boldsymbol{x}_k^i) \cdot w_{k-1}^i \tag{5.4}$$

La distribución de muestreo, $q_S(\boldsymbol{x}_k^i\ |\ \boldsymbol{x}_{k-1}^i, \boldsymbol{z}_k)$, para este caso es igual al modelo de movimiento, $p(\boldsymbol{x}_k|\ \boldsymbol{x}_{k-1})$. Para evitar la dependencia de w_{k-1}^i se remuestrea en todos los instantes de tiempo. Esto hace que los pesos finales se simplifiquen a

$$w_k^i = (1-\alpha)\frac{p(\boldsymbol{z}_k|\ \boldsymbol{x}_k^i)}{\sum_j^M w_k^j} \tag{5.5}$$

normalizados a $1-\alpha$, la parte de probabilidad que les corresponde según la expresión 5.1.

Para el cálculo de los pesos de la parte no secuencial partiremos de la expresión 3.23:

$$w_k^i = \frac{p(\boldsymbol{x}_k^i|\ \boldsymbol{z}_k)}{q_{NS}(\boldsymbol{x}_k^i\ |\ \boldsymbol{z}_k)} = \frac{p(\boldsymbol{z}_k|\ \boldsymbol{x}_k^i)p(\boldsymbol{x}_k)}{p(\boldsymbol{z}_k)q_{NS}(\boldsymbol{z}_k|\ \boldsymbol{x}_k)} \tag{5.6}$$

Dado que no tenemos información *a priori* sobre $p(\boldsymbol{x}_k)$ la supondremos uniforme, eliminándola de la anterior expresión. Esto nos lleva a

$$w_k^i \propto \frac{p(\boldsymbol{z}_k|\ \boldsymbol{x}_k^i)}{q_{NS}(\boldsymbol{z}_k|\ \boldsymbol{x}_k)} \tag{5.7}$$

Nuestra elección de la función de abducción, q_{NS}, es tal que todas las partículas que proporciona tienen la misma probabilidad de ser elegidas. De esta manera podemos simplificar el cálculo de los pesos a exactamente la misma expresión que usamos para la parte secuencial:

$$w_k^i = \alpha\frac{p(\boldsymbol{z}_k|\ \boldsymbol{x}_k^i)}{\sum_j^{M^*} w_k^j} \tag{5.8}$$

ponderada esta vez por α.

En el cuadro 5.1 podemos ver un resumen de los pasos principales a seguir en el algoritmo de estimación multimodal que hemos presentado.

El algoritmo presentado respeta los formalismos de Monte Carlo y de filtrado bayesiano que hemos presentado en los capítulos 3 y 4. La combinación de las dos partes, la secuencial y la no secuencial, en una única representación también respeta las suposiciones de convergencia que se presentaron en la introducción a los métodos de Monte Carlo en la sección 3.3. Allí comentamos que la aproximación de Monte Carlo cumple que:

$$\lim_{N\to\infty} |I - \hat{I}| = 0 \tag{5.9}$$

$\left[\{\boldsymbol{x}_k^i, w_k^i\}_{i=1}^N\right] = ESTIMACION\_MULTIMODAL\left(\left[\{\boldsymbol{x}_k^i, w_k^i\}_{i=1}^N\right]\right)$

- Remuestrea con ruleta $(1-\alpha)N$ veces.
- Parte secuencial:
  - Repetir con i desde 1 hasta $(1-\alpha)N$
    - $x_S^i \sim p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$
    - $w_S^i \propto p(\boldsymbol{z}_k|\ \boldsymbol{x}_k)$
  - Normaliza pesos para que $\sum_{i=1}^{(1-\alpha)N} w_S^i = 1-\alpha$
- Parte no secuencial:
  - Repetir con i desde $(1-\alpha)N+1$ hasta N
    - $x_{NS}^i \sim q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{z}_k)$
    - $w_{NS}^i = 1$
  - Normaliza pesos para que $\sum_{i=(1-\alpha)N+1}^{N} w_{NS}^i = \alpha$

Cuadro 5.1: Algoritmo de estimación de densidad de probabilidad multimodal.

La bondad de las aproximaciones de Monte Carlo se miden en función del error de aproximación de la integral I (expresiones 3.8 y 3.9). Si usamos el estimador propuesto en la expresión 5.1 podemos calcular el error de aproximación de la siguiente forma:

$$\hat{I} = (1-\alpha)\sum w_S^i \boldsymbol{f}(x_S^i) + \alpha \sum w_{NS}^i \boldsymbol{f}(x_{NS}^i) \tag{5.10}$$

$$|I-\hat{I}| = |I - (1-\alpha)\sum w_S^i \boldsymbol{f}(x_S^i) + \alpha \sum w_{NS}^i \boldsymbol{f}(x_{NS}^i)| \tag{5.11}$$

$$= |(1-\alpha)\left(I - \sum w_S^i f(x_S^i)\right) + \alpha\left(I - \sum w_{NS}^i f(x_{NS}^i)\right)| \tag{5.12}$$

$$= |(1-\alpha)(I-\hat{I}_S) + \alpha(I-\hat{I}_{NS})| \tag{5.13}$$

El límite de este error será la combinación lineal de los límites de $|I-\hat{I}_S|$ e $|I-\hat{I}_{NS}|$ que, como hemos visto, tienden a cero cuando el número de partículas tiende a infinito. Por lo tanto, el estimador $\hat{I}$, combinación de otros dos estimadores muestrales, es también un estimador muestral válido según los criterios de Monte

Carlo.

La velocidad de convergencia será, por lo menos, igual a la más lenta de los dos filtros. Por otra parte, el error de convergencia también estará, como mínimo, por debajo del error máximo. La intuición que podemos extraer de esta combinación es que las partículas están mejor colocadas con la combinación de los dos algoritmos de lo que lo están de manera independiente.

5.4. Estimación de la posición de los objetos usando una distribución de probabilidad multimodal

Una de las limitaciones que aparecen con el uso de un estimador de distribuciones de probabilidad como el que vamos a usar es que no proporciona una estimación de la posición de cada objeto de manera directa. El algoritmo es un método de estimación multimodal de funciones de densidades de probabilidad. Se centra en localizar y representar correctamente los máximos de dichas funciones.

Sin embargo, aunque localice y represente correctamente los máximos, no es un obtiene directamente la posición de cada objeto. Esta tarea se debe resolver en otro nivel del algoritmo, encargado de localizar y segmentar la información que tenemos para poder asociar las partículas con los objetos a los que pertenecen.

Algunos de los métodos que presentamos en la sección 2.3 asocian de manera explícita las partículas a los objetos, separando la evolución de cada subconjunto de partículas. Aunque son capaces de seguir los objetos correctamente, adolecen de otros problemas. En primer lugar la división hay que hacerla de manera explícita, lo que resta flexibilidad al uso de nuestros recursos de búsqueda. Además, la aparición y desaparición de objetos tiende a considerarse como un caso especial y no como algo normal y que forma parte del algoritmo.

En nuestro caso añadimos un segundo nivel totalmente independiente del primero, capaz de estimar tanto el número de objetos como sus posiciones. Este nivel recibe la población de partículas y estima el número y la posición de los objetos presentes, pero no influye para nada en la colocación futura de las partículas. Hay varias alternativas para hacer esto.

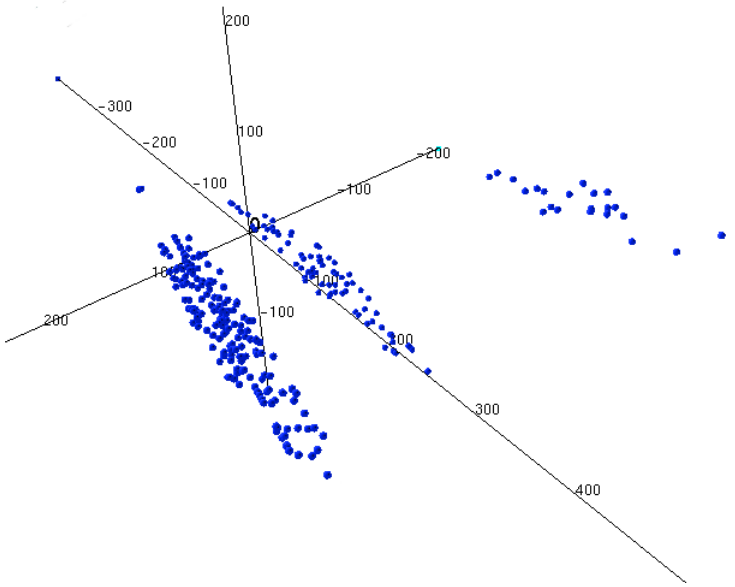

Figura 5.5: Representación de partículas a la entrada del algoritmo de segmentación.

Podríamos buscar las partículas de mayor peso en un determinado área y considerarlas como máximos locales. Esta aproximación no resulta adecuada dado que, con nuestro modelo, todas las partículas tendrán pesos similares, lo que nos interesa son las áreas de interés y no directamente la partícula de mayor peso, como comentamos en la figura 4.4.

En nuestro caso la opción que hemos elegido consiste en buscar las zonas de interés del espacio de estados, segmentando la distribución de partículas. En la figura 5.5 podemos ver cómo se distribuyen las partículas en el espacio de estados, agrupándose en las zonas correspondientes a cada objeto. El algoritmo de segmentación deberá separar los tres grupos que existen, agrupando las partículas según su posición en el espacio de estados.

Debe notarse que esta segmentación no es de los datos de entrada, las imágenes, sino de las partículas en el espacio de estados. Resulta más sencillo seguir esta vía dado que es posible definir una métrica de similitud entre partículas en el espacio de estados: la distancia entre ellas.

Hay muchos algoritmos de segmentación que pueden usarse para esta tarea, desde el KNN (K Vecinos cercanos) hasta los mapas autoorganizativos [Bis95]. El al-

goritmo que nosotros hemos empleado es el EM, o de maximización de esperanza (*expectation maximization*). Este algoritmo es la base para sistemas de segmentación muy conocidos como K-Medias o mezcla de gaussianas. Esta última será la variante que usaremos. Es posible encontrar más información sobre el algoritmo EM o el de mezcla de gaussianas en [Bis95, Mac03].

El algoritmo funciona de la siguiente forma. Parte de un conjunto de datos $D = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, distribuidos según una determinada distribución de probabilidad $p(\boldsymbol{x})$ y tomados de manera independiente. Suponemos que la distribución de probabilidad $p(\boldsymbol{x})$ está formada por varias componentes gaussianas, o por lo menos puede aproximarse por:

$$p(\boldsymbol{x}|\Theta) = \sum_{k=1}^{K} \alpha_k p(\boldsymbol{x}|\ \theta_k) \tag{5.14}$$

K indica el número de componentes, que debe conocerse *a priori*, y Θ representa los parámetros del modelo, que regulan tanto las componentes como los pesos de combinación entre ellas. Dichos pesos son tales que:

$$\sum_{k=1}^{K} \alpha_k = 1 \tag{5.15}$$

Cada una de estas componentes será una gaussiana de la forma:

$$p(\boldsymbol{x}|\ \theta_k) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_k)^T \Sigma_k^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_k)\right\} \tag{5.16}$$

dónde los parámetros θ_k estarán formados por la media y la matriz de covarianza de la distribución gaussiana:

$$\theta_k = \{\boldsymbol{\mu}_k, \Sigma_k\} \tag{5.17}$$

El algoritmo EM es un proceso iterativo que está definido por dos pasos, el de cálculo de la esperanza y el de maximización. El primero de los dos pasos, conocido como paso *E*, calcula la probabilidad de pertenencia de cada punto del conjunto de

datos D, $\boldsymbol{x}_i$, al conjunto c definido por la gaussiana k como:

$$\omega_{ik} = p(c = k|\boldsymbol{x}_i, \theta_k) = \frac{p_k(\boldsymbol{x}_i|\theta_k)\alpha_k}{\sum_{m=1}^{K} p_m(\boldsymbol{x}_i|\ \theta_k)\alpha_m} \tag{5.18}$$

Para obtener esta expresión simplemente hay que aplicar la regla de Bayes de forma directa. Es necesario calcular todas las combinaciones de partículas con gaussianas, creando así una matriz $N * K$. Dicha matriz mantiene la información de la pertenencia de cada muestra a cada grupo.

En el siguiente paso, el paso ***M***, recalculamos los parámetros de cada grupo, usando la información de pertenencia redefinidos por los datos próximos a su situación actual. Las expresiones usadas para la actualización son las siguientes:

$$\begin{aligned}
\alpha_k^{new} &= \frac{1}{N}\sum_{i=1}^{N}\omega_{ik} \quad 1 \leq k \leq K \\
\boldsymbol{\mu}_k^{new} &= \left(\frac{1}{\sum_{i=1}^{N}\omega_{ik}}\right)\sum_{i=1}^{N}\omega_{ik}\boldsymbol{x}_i \quad 1 \leq k \leq K \\
\Sigma_k^{new} &= \left(\frac{1}{\sum_{i=1}^{N}\omega_{ik}}\right)\sum_{i=1}^{N}\omega_{ik}(\boldsymbol{x}_i - \boldsymbol{\mu}_k^{new})(\boldsymbol{x}_i - \boldsymbol{\mu}_k^{new})^T
\end{aligned} \tag{5.19}$$

Este proceso se repite hasta que las partículas no cambien de grupo entre una iteración y la siguiente o hasta que se supere el máximo número de iteraciones. La inicialización se puede hacer tomando K partículas al azar y centrando en ellas sendas gaussianas.

La propia media de las gaussianas nos servirá como estimación de la posición de los objetos. En el capítulo 4 ya empleamos un método similar para calcular las posiciones de los objetos.

Para completar la funcionalidad del algoritmo EM debemos ajustar el número de objetos K, dado que dicho algoritmo no es capaz de hacerlo. Existen muchas estrategias dentro del grupo de algoritmos de clasificación sin supervisión. Algunas de ellas funciona de manera constructiva (añadiendo cada vez más objetos) y otras destructiva (partiendo de un número de objetos elevado y eliminando algunos). En nuestra

propuesta usaremos un algoritmo basado en el algoritmo ISODATA (del inglés *Iterative Self-Organizing Data Analysis Techniques*) [BH65]. Este algoritmo permite que el número de grupos se ajuste automáticamente durante las iteraciones, uniendo grupos similares y partiendo grupos con altas desviaciones típicas. Para conseguir mejores resultados, el algoritmo que usamos está muy integrado con el EM en nuestra implementación, por lo que lo comentaremos en la sección 6.4, referente a la implementación del mismo.

La segmentación de las partículas es necesaria para proporcionar una estimación sobre la posición de todos los objetos presentes en la escena. Sin embargo, no constituye el foco principal del trabajo de esta tesis. El objeto consiste en proporcionar una distribución de probabilidad capaz de representar todos los objetos presentes en la escena, por lo que no entraremos en mayor profundidad en el algoritmo de segmentación.

5.5. Experimentos

Una vez presentado el marco formal, en esta sección nos centraremos en comprobar el funcionamiento del sistema combinado propuesto para el seguimiento de múltiples objetos, mediante la realización de varios experimentos. La plataforma experimental que vamos a emplear es exactamente igual a la que se utilizaba en el capítulo 4 y que se describe en el capítulo 6, incluyendo las mismas cámaras y el mismo equipo.

5.5.1. Seguimiento de un único objeto

Comenzamos comprobando que los cambios que hemos introducido en el algoritmo no invalidan, de hecho mejoran, las capacidades del sistema para seguir un único objeto. Para ello colocamos un objeto en una posición fija y dejamos que el sistema lo localice. Las posiciones que usamos son las mismas que las que empleamos en el capítulo 4. En la tabla 4.8 se encuentran las posiciones de cada uno de los objetos.

En la tabla 5.2 podemos ver los errores de estimación para las ocho posiciones

ID	α							
	0,0	0,1	0,2	0,4	0,6	0,8	0,9	1,0
1	3,67	3,57	3,41	3,72	3,99	3,74	3,73	3,59
2	1,75	1,55	1,59	1,47	1,53	1,47	1,49	1,44
3	2,10	2,40	2,27	2,20	2,14	2,11	2,04	2,00
4	15,15	15,17	15,04	15,74	16,58	16,85	16,92	17,21
5	9,59	10,69	10,43	11,18	11,13	10,99	11,25	11,21
6	2,88	2,07	3,12	3,50	3,38	3,72	4,02	4,36
7	9,92	10,30	10,22	10,20	10,26	10,02	10,40	10,25
8	3,74	3,79	2,85	2,78	2,89	2,47	2,60	2,38

Cuadro 5.2: Errores en centímetros para cada objeto para diferentes valores de α.

consideradas en el capítulo anterior. Los valores máximo y mínimo (correspondientes a $\alpha = 1$ y a $\alpha = 0$) se corresponden con los algoritmos de muestreo enfatizado y de condensación, respectivamente.

Los errores de estimación con el algoritmo modificado son del mismo orden a los que obteníamos con los algoritmos de condensación y de muestreo enfatizado por separado, que aparecen en el cuadro 4.4. Debemos recordar que en este error no sólo influye el error del algoritmo de seguimiento, también se hereda siempre el error de la calibración inexacta de las cámaras reales. Por otra parte los resultados de convergencia, en las técnicas de Monte Carlo, tienen carácter probabilístico por lo que puede haber variaciones de una ejecución a otra.

Podemos observar cómo las variaciones en el parámetro α no suponen cambios radicales en las prestaciones en cuanto al error medio se refiere.

Sin embargo, sí tiene importancia en el ruido residual de las estimaciones. En la figura 5.6 podemos ver la evolución del error con el tiempo. El cambio que se produce tras, aproximadamente, un segundo se corresponde con la aparición del objeto en la escena. Vemos cómo el cambio en el parámetro α tiene implicaciones en cuanto al rizado de la estimación. Los resultados son similares a los que se obtenían en el capítulo anterior para $\alpha = 0$ y $\alpha = 1$. Para los casos intermedios los resultados se ajustan a lo esperado, esto es, una transición suave entre los casos extremos expresados por el muestreo enfatizado y el filtro de condensación.

La velocidad de convergencia es similar para todos los valores de α. Esto se debe

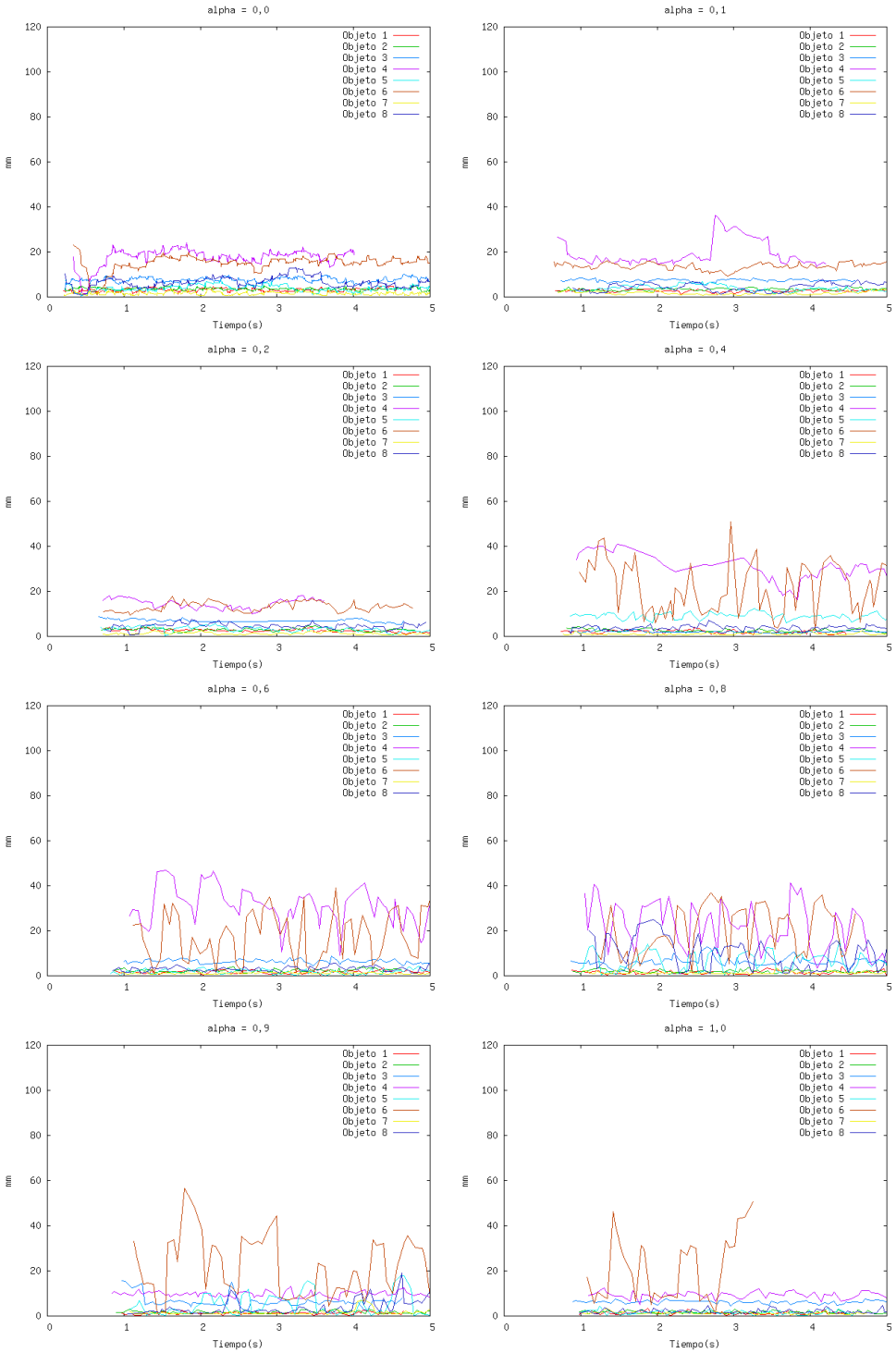

Figura 5.6: Evolución del error en función del parámetro α para un único objeto estático.

a que el rango de tiempo en el que nos estamos moviendo es suficientemente grande como para que no apreciemos diferencias entre un caso y otro. Pero más importante aún, hemos seleccionado sólo los casos en los que el filtro de condensación ($\alpha = 0$) ha convergido. Si el punto de partida es incorrecto, condensación solo no se recupera muy bien de esta situación, haciendo que su convergencia sea mucho más lenta de lo que aquí hemos mostrado. En los experimentos que hemos realizado el sistema convergía únicamente una vez de cada cinco pruebas, aproximadamente. Esta es una de las limitaciones del algoritmo, que no aparece para el resto de los valores de α considerados. En esos casos, el objeto aparece en una posición, sólo las partículas de la abducción tienen pesos significativos (recordemos que estamos en el caso de un único objeto). Estas partículas se reproducirán en el paso de remuestreo hasta llenar casi completamente el número de partículas de la parte secuencial. Este proceso es el que confiere al algoritmo que presentamos su capacidad para converger en todas las situaciones y hacerlo rápidamente.

Los resultados en cuanto a velocidad (medidas en número de iteraciones del algoritmo en un ordenador personal) de estas pruebas aparecen en el cuadro 5.3. Los resultados son similares a los obtenidos para el caso del muestreo enfatizado, que vimos en la tabla 4.6. En el algoritmo de seguimiento multiobjeto la parte más costosa corresponde, al igual que en el muestreo enfatizado, a la función de abducción, que requiere una filtrado completo de la imagen. En la tabla 5.3 vemos cómo el número de iteraciones por segundo cambia en función del parámetro α. El salto que se produce al pasar de $\alpha = 0$ a $\alpha = 0,1$ es debido a que es necesario realizar el filtrado de la imagen completa, reduciendo drásticamente la velocidad del sistema. A partir de ese momento el incremento del número de partículas abductivas, más costosas que las partículas de condensación, conllevará un descenso paulatino en la velocidad, bajando hasta unas 20 iteraciones por segundo. Aún así, el aumento del coste computacional viene acompañado con una mejora en las prestaciones del seguimiento para el caso multiobjeto, como hemos mostrado en la tabla 5.2.

Parámetro α	Iteraciones/segundo
0	143
0,1	32
0,2	29
0,4	28
0,6	24
0,8	21
0,9	20
1,0	20

Cuadro 5.3: Iteraciones por segundo del algoritmo en función del parámetro α.

5.5.2. Seguimiento de dos objetos

Nuestra siguiente prueba entra en la localización, al mismo tiempo, de dos objetos diferentes. En este caso el filtro de condensación, como vimos en el capítulo 4, no era capaz de mantener las dos hipótesis y convergía a uno de los dos objetos.

En este experimento vamos a emplear únicamente 2 de las 8 posiciones de la tabla 4.8, pero ahora simultáneamente, colocando sendos objetos en las posiciones 3 y 4. Hemos elegido estas por ser dos posiciones suficientemente diferentes, una con un error de estimación sensiblemente mayor a la otra.

En el cuadro 5.4 se muestran estimaciones medias y los errores para los dos objetos, con diferentes valores de α. Puede verse cómo para el valor $\alpha = 0$, análogo al filtro de condensación, el sistema únicamente detecta uno de los dos objetos. En el resto de los casos se detectan correctamente ambos objetos y los errores de estimación son comparables a los obtenidos en el cuadro 4.4.

El mecanismo fundamental para que el sistema sea capaz de estimar dos objetos al mismo tiempo es el reparto de las partículas. El sistema debe encargarse de asignar un número suficiente de partículas a cada objeto. En el cuadro 5.5 se muestra el reparto de partículas entre los dos objetos. Cuando el parámetro α es cero el sistema únicamente converge a uno de lo objetos, el primero que encuentra, situando todas las partículas en él. A partir de ese momento, al incrementar el número de partículas procedentes de la abducción, hay una división más equitativa entre los objetos. Esto se debe a que el método de abducción, tal y como aparece descrito en la sección 6.3, se encarga de colocar partículas por igual en todos los objetos existentes.

α	Objeto 1				Objeto 2			
	X(cm)	Y(cm)	Z(cm)	Error (cm)	X(cm)	Y(cm)	Z(cm)	Error (cm)
0,0	23,28	22,90	18,90	1,97	—	—	—	—
0,1	23,28	22,81	18,95	1,92	1,02	84,00	36,28	15,04
0,2	23,26	23,21	18,87	2,09	0,90	85,03	36,37	14,01
0,4	23,32	23,28	18,92	2,04	1,04	82,86	36,15	16,17
0,6	23,25	22,86	18,89	1,98	0,98	83,29	36,28	15,74
0,8	23,19	22,62	18,93	1,97	1,03	82,28	36,18	16,75
0,9	23,24	22,90	18,93	1,97	1,02	82,09	36,17	16,93
1,0	23,23	22,65	18,93	1,95	1,00	81,92	36,16	17,11

Cuadro 5.4: Errores en centímetros para cada objeto para diferentes valores de α.

1.000 partículas				2.000 partículas			
α	Objeto 1	Objeto 2	Total	α	Objeto 1	Objeto 2	Total
0,0	777	0	777	0,0	1529	0	1529
0,1	80	771	851	0,1	141	1395	1536
0,2	79	714	793	0,2	131	1357	1488
0,4	124	609	733	0,4	227	1171	1398
0,6	156	562	718	0,6	321	1010	1322
0,8	200	511	711	0,8	359	937	1296
0,9	190	501	691	0,9	369	920	1288
1,0	203	497	700	1,0	377	888	1265

Cuadro 5.5: Reparto del número de partículas entre dos objetos estáticos en función del parámetro α usando 1.000 partículas y 2.000 partículas respectivamente.

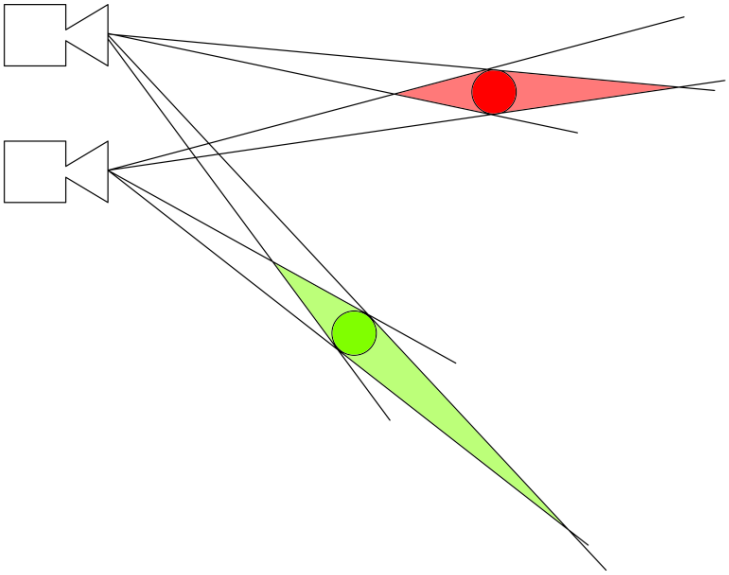

Figura 5.7: Diferencias en las áreas correspondientes a los objetos en función de la posición en el espacio de estados.

Pero aun en el caso en que todas las partículas las coloca la función de abducción ($\alpha = 1$), existe una diferencia entre el número de partículas situadas sobre cada objeto. Dicha diferencia proviene de la zona de incertidumbre asociada a cada objeto, es decir, del área asignada en el espacio de estados a cada objeto por el modelo de movimiento. El tamaño de ese área dependerá del tamaño real del objeto y de su posición relativa con respecto a las cámaras, como comentamos en los experimentos del capítulo anterior (ver figura 4.15). En el caso de la tabla 5.5 el "objeto 2" es el que tiene asociado un área de incertidumbre mayor.

Cuanto mayor sea este área mayor será el número de partículas asignadas al objeto. Por ejemplo, en el caso en el que sólo tengamos partículas de abducción y coloquemos el mismo número de partículas en cada línea de visión posible, el área más grande acogerá mayor número de partículas válidas. El resto de las partículas serán consideradas "parias" por el algoritmo de segmentación. En la figura 5.7 podemos ver cómo para dos objetos del mismo tamaño se obtienen zonas de incertidumbre de diferente volumen al tener posiciones diferentes con respecto a las cámaras usadas.

Este efecto es también el responsable del reparto asimétrico para el caso en que

el parámetro α sea menor que 1. Las partículas no tendrán las mismas probabilidades de reproducirse de una iteración a la siguiente, es decir, de superar el proceso de remuestreo, depende del área de incertidumbre de cada objeto. Un objeto con un área asociada grande partirá con un número superior de partículas procedentes de la función de abducción. Dichas partículas serán seleccionadas por el proceso de remuestreo, formando así parte de las partículas de la parte secuencial del algoritmo. Una vez aplicado el modelo de movimiento las partículas obtendrán buenos pesos si caen en el área asignada al objeto. Esto será más probable si el área de incertidumbre del objeto es más grande (para un modelo de movimiento en concreto). La cantidad total de partículas no influye en el reparto entre objetos de las mismas, dado que las proporciones entre las áreas no varían, como puede apreciarse en la tabla 5.5.

Las partículas que caen fuera de las áreas marcadas en la figura 5.7 obtendrán valores de probabilidad bajos en el modelo de observación. La cantidad de partículas que no influyen en la estimación de los objetos podemos verla en la tabla 5.5. De las 1.000 partículas usadas más de 300 no se emplean para la estimación de las posiciones de los objetos en ningún momento. El proceso de abducción resulta muy útil porque es capaz de colocar cerca de 700 partículas en zonas correctas del espacio, aunque se haga a costa de colocar 300 partículas en zonas de poco interés *a posteriori*.

El algoritmo de estimación de la posición tan solo necesita unas pocas partículas bien colocadas para proporcionar estimaciones suficientemente correctas, como pudo verse en la tabla 5.4. Colocar un número suficiente de partículas sobre cada objeto resulta más relevante que realizar un reparto lo más simétrico posible. Será la parte no secuencial del algoritmo propuesto la que se encargue de mantener un mínimo de partículas en cada objeto. De esa forma se consiguen los dos objetivos buscados: localizar todos los objetos presentes en la escena, reaccionando rápidamente a las nuevas incorporaciones, y mantener una población estable en cada uno de ellos.

5.5.3. Seguimiento frente a incorporaciones de nuevos objetos

Hasta ahora hemos considerado el caso en el que el sistema comenzaba a funcionar cuando los objetos ya existían. Nuestro siguiente experimento se centra en estudiar el funcionamiento del sistema frente a la aparición de nuevos objetos. Para

ello colocaremos una única pelota en una posición determinada y esperaremos hasta que el sistema converja a dicha posición. Tras ello colocaremos una segunda pelota en otra posición. Las pelotas usadas son las mismas que en el caso anterior, en primer lugar colocamos el "objeto 1" y a continuación el "objeto 2".

En la figura 5.8 vemos cómo se reparten las partículas a lo largo del tiempo para diferentes valores del parámetro α. En las gráficas puede verse que, aun partiendo de una situación ventajosa para uno de los objetos, el algoritmo converge a una situación estable similar a la expuesta en la tabla 5.5. Vemos cómo a lo largo del tiempo hay grandes diferencias en el número de partículas asociadas a un mismo objeto. Esto se debe a la naturaleza probabilística del proceso de remuestreo y colocación de nuevas partículas, ya sean colocadas por el modelo de movimiento o por la función de abducción. Lo importante es que el número de partículas para un objeto no desaparezca. En ese caso el objeto no sería localizado y no podría estimarse su posición. Aún cuando esto ocurra, al remuestrear partiendo de un número pequeño de partículas, la parte no secuencial del algoritmo será capaz de recolocar un número mínimo de partículas para reestimar de nuevo la posición de dicho objeto.

Para continuar comprobando el funcionamiento frente a la aparición de múltiples objetos vamos a colocar, de manera consecutiva, 5 objetos en posiciones diferentes. De esta forma el sistema tendrá que localizar primero un objeto, luego dos, etc. Las posiciones elegidas y el orden de aparición pueden verse en la figura 5.9.

En la figura 5.10 podemos ver el reparto de partículas según van apareciendo los objetos para un valor de α de 0,4. La elección de este parámetro de α, como hemos visto en los experimentos anteriores no resulta especialmente relevante para los resultados de seguimiento del algoritmo.

Al ir incrementando el número de objetos, la función de abducción añade nuevas zonas en las que colocar las partículas disponibles. De esta forma vemos como las partículas se van distribuyendo sobre cada objeto hasta cubrir las cinco posiciones. Las subidas que aparecen al principio de cada curva se corresponden a la localización del objeto. Al encontrar un objeto nuevo el sistema coloca unas pocas partículas en él. El paso de remuestreo irá colocando más partículas en cada objeto localizado, aumentando el número de partículas rápidamente.

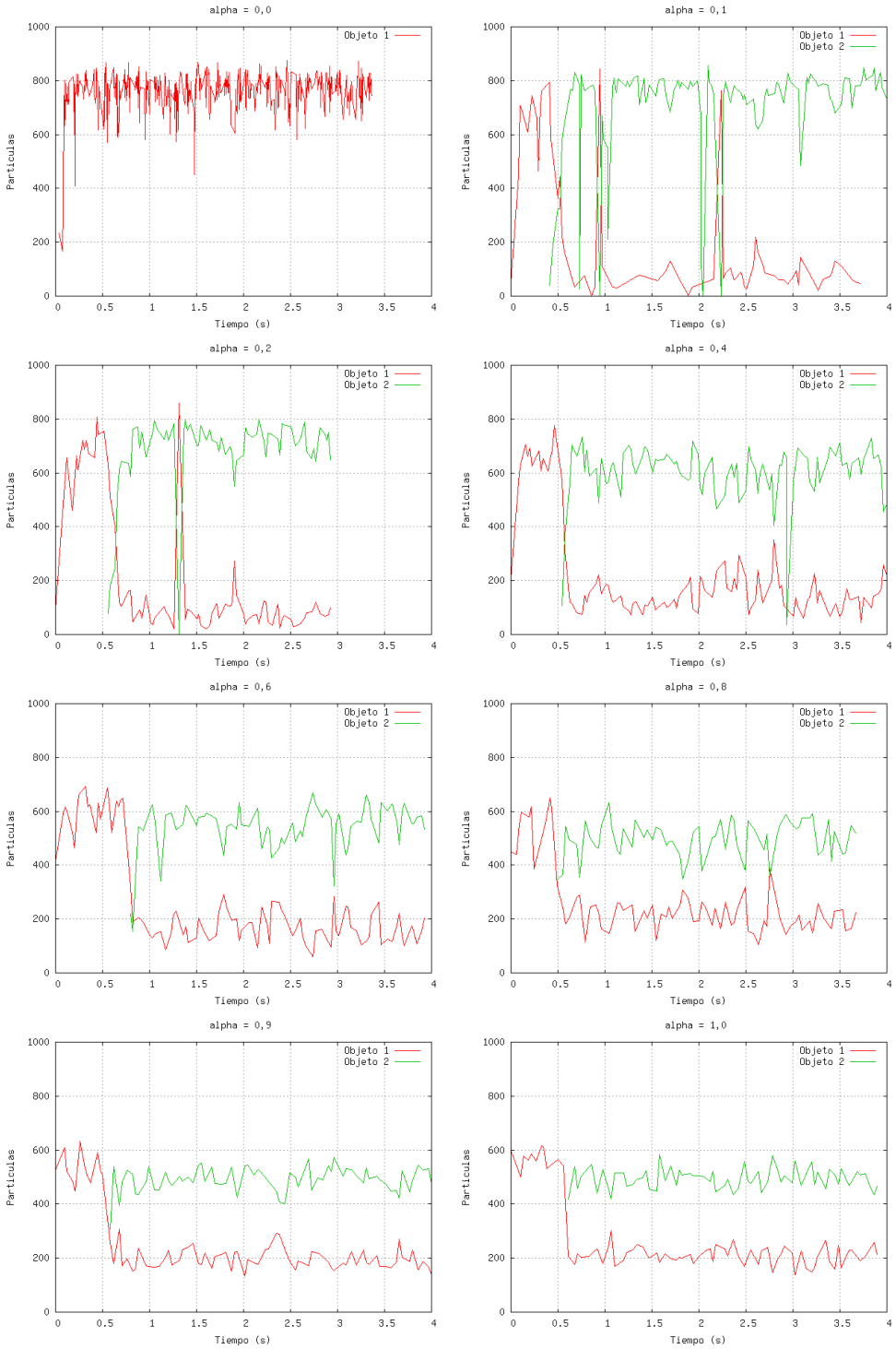

Figura 5.8: Evolución del reparto de partículas para dos objetos en función del parámetro α.

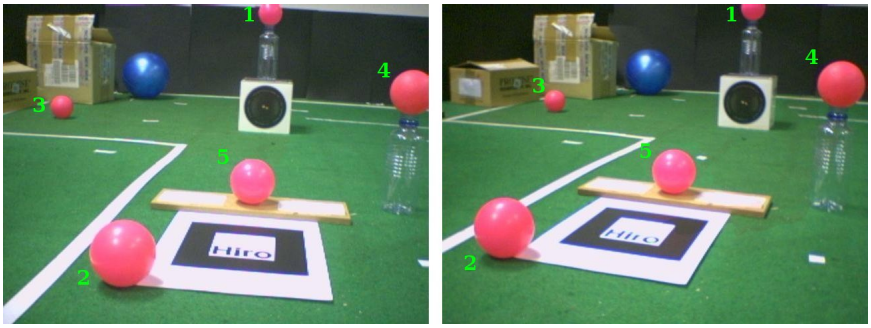

Figura 5.9: Colocación inicial de cinco objetos y orden de aparición.

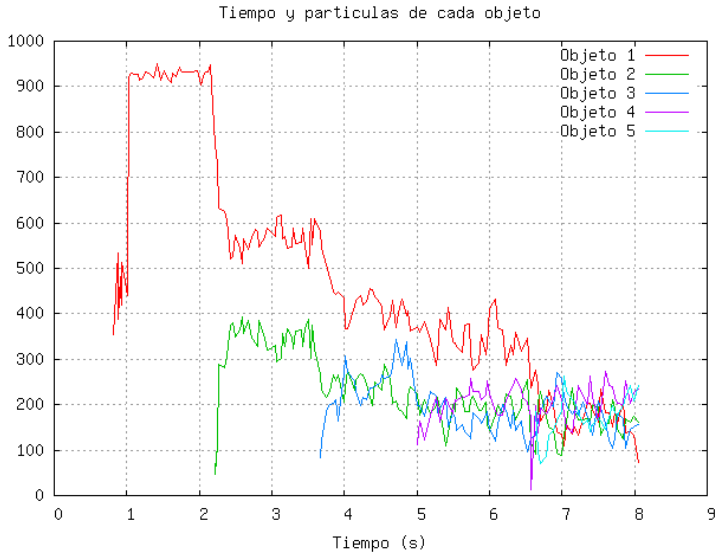

Figura 5.10: Reparto de partículas a lo largo del tiempo para cinco objetos.

Una vez que mantenemos una población mínima en un objeto seremos capaces de proporcionar una estimación para su posición. En la figura 5.11 podemos ver la evolución del error de seguimiento para cada objeto. Vemos cómo los errores se mantienen bastante estables aunque el número de partículas en cada objeto es decreciente según aumenta el número de objetos.

Debe notarse el repunte del error del objeto 1 sobre el sexto segundo. Dicho incremento del error se debe a la aparición del objeto 5. Los resultados que mostramos en la gráfica son fruto de aplicar el algoritmo de segmentación sobre la población de

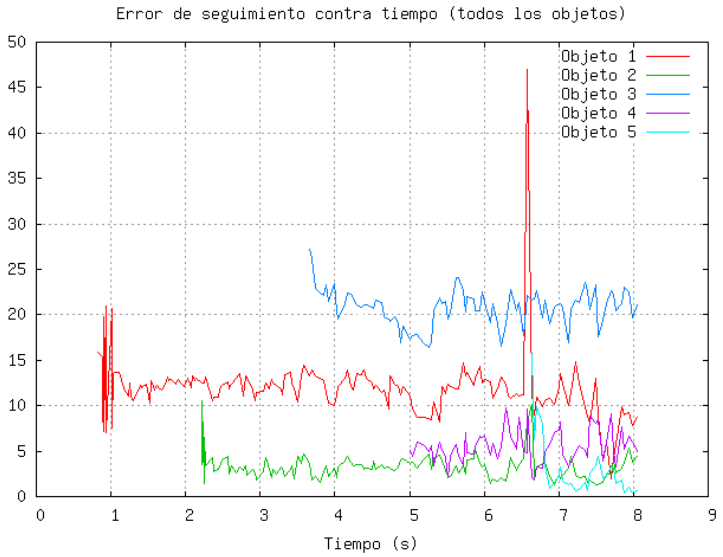

Figura 5.11: Error de seguimiento para cinco objetos.

partículas. Cuando dos partículas, o grupos de partículas, se encuentran muy cerca entre sí, el algoritmo de segmentación que empleamos las asignará al mismo grupo. Para ajustar la sensibilidad del algoritmo a la aparición de nuevos objetos tenemos el parámetro *distorsión máxima*. La distorsión es una medida de la varianza de las partículas de un mismo grupo. Cuando dicha varianza aumenta sabemos que el grupo comienza a hacerse más grande. Esto es lo que le sucede al objeto 1 cuando aparecen las primeras partículas del grupo 5. Cuando aparecen más partículas, la varianza supera el umbral de distorsión máxima y el algoritmo de segmentación coloca dos objetos diferentes. En ese momento el error de las dos estimaciones vuelve a colocarse en los valores usuales.

Esto error aparece por el algoritmo de segmentación empleado, no por la colocación de las partículas que es correcta. Para solucionarlo sería necesario refinar el algoritmo de segmentación, dado que su simplicidad resulta insuficiente en este caso. Aumentar el valor del parámetro distorsión máxima no es suficiente. Si lo hacemos las partículas de los objetos más lejanos, que tendrán zonas de incertidumbre más grandes (ver figura 5.7), se fragmentarán en más grupos. Para evitar este problema el algoritmo de segmentación debería incluir información acerca del tamaño de la zona

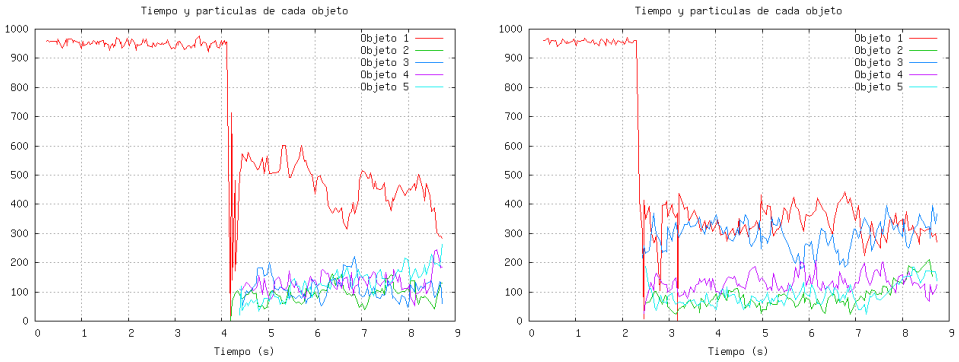

Figura 5.12: Reparto de partículas a lo largo del tiempo para cinco objetos que aparecen al mismo tiempo.

de incertidumbre, que estará relacionado con la varianza máxima que debe permitirse para ese objeto. Esto se encuentra fuera del objetivo de esta tesis, centrada en la correcta colocación de las partículas.

Por lo demás vemos que el algoritmo de seguimiento detecta de forma correcta la aparición de los cinco objetos, manteniendo un error de seguimiento acotado, con las consideraciones realizadas anteriormente.

Para terminar las pruebas de incorporación de objetos podemos comprobar cómo se enfrenta el algoritmo a la aparición de varios objetos al mismo tiempo. La diferencia principal en este caso radica en la aparición al mismo tiempo de los objetos, lo que obliga al algoritmo a detectarlos al mismo tiempo, o lo que es lo mismo, a colocar partículas en varios sitios nuevos a la vez. Para realizar este experimento empleamos las mismas cinco posiciones que en el caso anterior. En primer lugar mostramos únicamente uno de los objetos. A continuación hacemos aparecer los demás al mismo tiempo. En la figura 5.12 podemos ver el reparto de las partículas usando dos valores diferentes de α, $0,2$ en la imagen de la izquierda y $0,4$ en la imagen de la derecha, una con menos componente abductivo que la otra. Valores por debajo de $0,2$ incluyen muy pocas partículas abductivas y valores por encima de $0,4$, debido a la realimentación del proceso de remuestreo, tendrían mucha influencia de la abducción.

En el reparto podemos ver cómo se colocan las partículas en todos los objetos tan pronto como aparecen. El sistema es capaz de hacerlo al mismo tiempo dado que las partículas se colocan siguiendo la función de propuesta abductiva. Dicha función

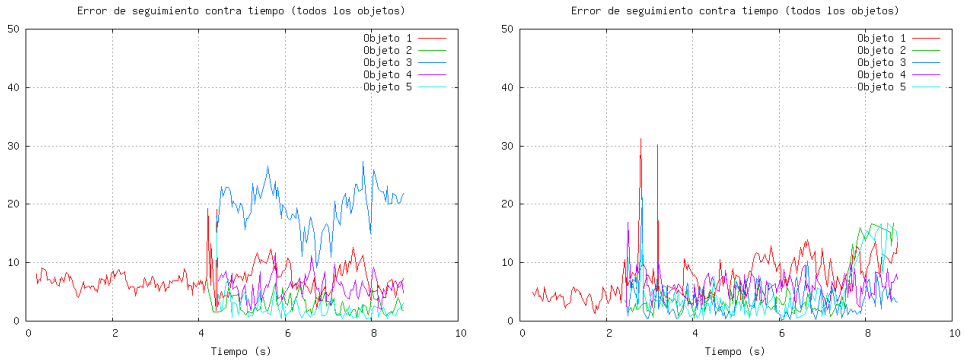

Figura 5.13: Error de seguimiento para cinco objetos que aparecen al mismo tiempo.

detecta los objetos tan pronto como aparecen en las observaciones.

Por otra parte podemos ver que al igual que en el caso de los dos objetos estudiados anteriormente, existe un objeto con mayor número de partículas que los demás. Las variaciones en el parámetro α, como comentamos en el caso de dos objetos, sí modifican el número de partículas sobre cada objeto aunque no cambian de manera significativa el número mínimo de partículas.

Gracias a este reparto es posible seguir todos los objetos al mismo tiempo. Sin embargo tiene un inconveniente: la función de abducción que empleamos no coloca el mismo número de partículas sobre cada objeto, como comentamos en la sección 5.5.2. Mejorar el reparto resulta interesante cuando aumentamos el número de objetos. Las partículas disponibles se repartirán entre todas las posibles líneas de visión y, en función del tamaño del objeto, obtendrán unos pesos elevados o no. Partiendo de un número de partículas fijo, la parte mínima de partículas que caen sobre cada objeto irá descendiendo en función del número de objetos.

Si el reparto fuera más justo podrían optimizarse los recursos al seguir un gran número de objetos, pero esto implicaría desarrollar tanto un modelo de observación más complejo como una función de abducción más exacta. Aunque dichas funciones implican un diseño más elaborado puede ser una solución mejor que aumentar el número de partículas, por el coste computacional que esto conlleva. En ambos casos el objetivo es el mismo, aumentar el número de partículas mínimo sobre cada objeto.

El error de seguimiento para los cinco objetos es similar al de los experimentos anteriores, como podemos ver en la figura 5.13. En ella se muestran los resultados

para los casos de $\alpha = 0,2$ (imagen de la izquierda) y $\alpha = 0,4$ (imagen de la derecha). Los errores son comparables, en valor, a los que mostrábamos en las figuras 5.6 o en el capítulo anterior. El hecho de tener en este caso varios objetos que aparecen al mismo tiempo no cambia de manera significativa los resultados en cuanto a precisión se refiere.

5.5.4. Seguimiento de varios objetos en movimiento

Para terminar vamos a presentar un experimento de seguimiento de nuestro algoritmo combinado en el caso en de varios objetos en movimiento. En este caso, al igual que sucedía en el capítulo anterior, no tendremos las posiciones reales de los objetos en todos los momentos por lo que calcular el error de estimación no nos resultará posible. Lo que sí podremos hacer es extraer las trayectorias y comprobar si dichas trayectorias siguen las físicas de los movimientos de los objetos.

Figura 5.14: Montaje para el seguimiento de múltiples objetos al mismo tiempo.

El montaje para este experimento puede verse en la figura 5.14. Las imágenes que aparecen en la figura son las que se capturan desde las dos cámaras empleadas. En ellas podemos ver cuatro pelotas rosas colgadas de diferentes soportes. Las pelotas se moverán siguiendo una trayectoria pendular sobre el plano XY. Cada una se encuentra a una distancia diferente desde las cámaras (alineado con el eje Z). Aunque las alineaciones no son perfectas, sí que nos darán una idea de cómo se están moviendo los objetos y, además, nos permite separar cada una de las componentes del movimiento para su análisis.

Figura 5.15: Movimiento de péndulo de cuatro objetos al mismo tiempo.

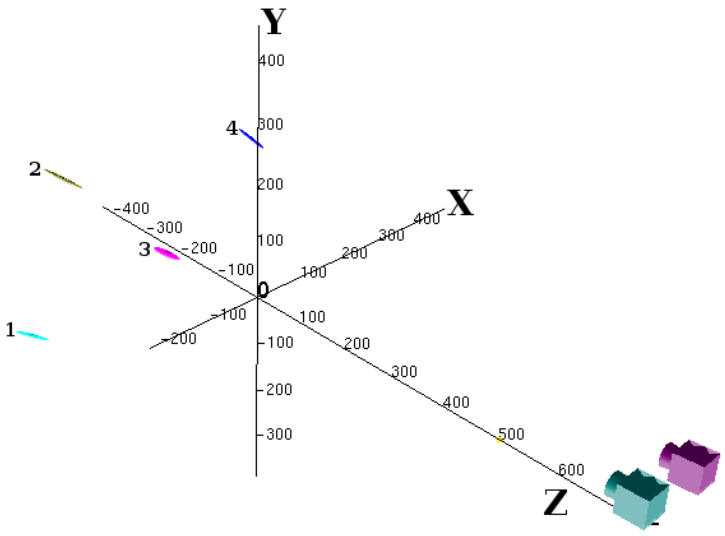

Figura 5.16: Representación 3D de la localización de cuatro objetos en movimiento.

En la figura 5.15 podemos ver un fragmento de la secuencia usada. En ella pueden verse las cuatro pelotas realizando un movimiento de péndulo, cada una de ellas a una velocidad y amplitud diferentes.

El experimento se ha realizado fijando el número de partículas a 1.000 y el parámetro $\alpha = 0,2$. Con estos parámetros el sistema es capaz de localizar los cuatro objetos al mismo tiempo, obteniendo su posición 3D. En la figura 5.16 podemos ver un gráfico en 3D con la posición de las dos cámaras y la posición de los cuatro objetos. Cada uno de los objetos está dibujado con un elipsoide que marca la zona de incertidumbre sobre la posición de estos objetos. De nuevo, la incertidumbre es más grande en la dirección de la distancia.

Podemos proyectar la posición 3D estimada de cada objeto sobre las dos imágenes usadas para comprobar si la estimación del sistema es correcta. En la figura 5.17 se muestran unos recuadros verdes sobre las proyecciones de los cuatro objetos. Podemos ver cómo coinciden con las posiciones de los objetos en cada imagen. Recordemos que la estimación realizada se calcula en 3D y, desde ahí, se proyecta en las imágenes. Por este motivo si la estimación en 3D no fuera correcta las proyecciones

Figura 5.17: Localización de los cuatro objetos en un determinado instante. Proyección sobre las dos imágenes capturadas por las cámaras.

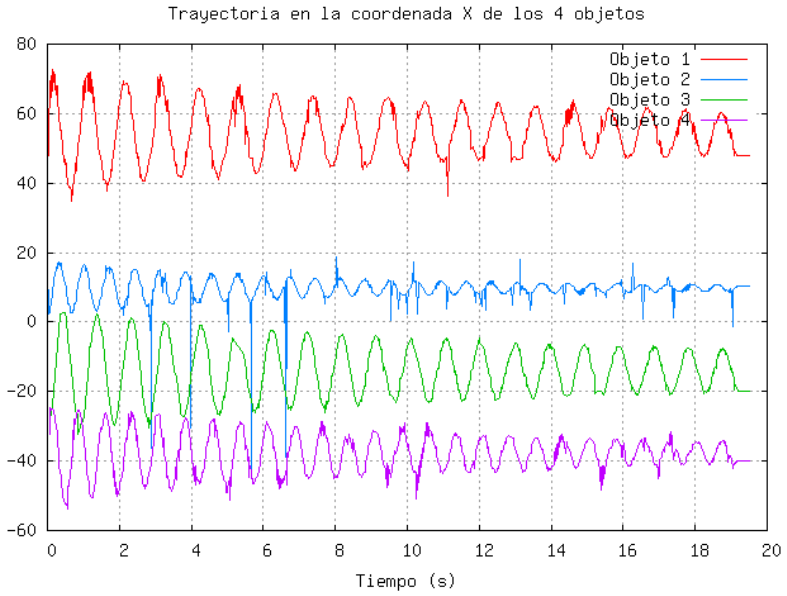

Figura 5.18: Seguimiento de cuatro objetos para un movimiento oscilatorio. Coordenada X en centímetros.

en las dos cámaras no podrían ser correctas al mismo tiempo.

Las trayectorias pueden verse en las figuras 5.18 y 5.19, para los ejes X e Y. El eje X es el eje horizontal de las figuras. En la figura 5.18 podemos ver cómo el sistema de seguimiento registra un movimiento oscilante que sigue fielmente el modelo real. A lo largo del tiempo la amplitud del movimiento se va reduciendo debido al rozamiento, hasta ir deteniéndose paulatinamente. El sistema de seguimiento mues-

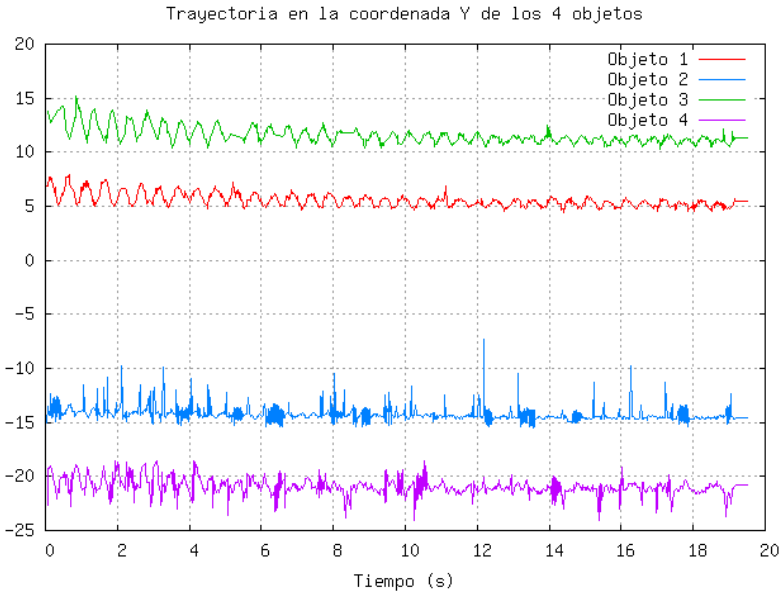

Figura 5.19: Seguimiento de cuatro objetos para un movimiento oscilatorio. Coordenada Y en centímetros.

tra, así mismo, esta tendencia. Los objetos están numerados de acuerdo a su orden de aparición de izquierda a derecha en la figura 5.17.

En la figura 5.19 se muestran las trayectorias en el eje Y, que sigue la dirección vertical de las imágenes. De nuevo pueden verse las variaciones ondulatorias correspondientes al movimiento pendular. El objeto 1 y el objeto 3 muestran claramente las trayectorias. Por el contrario, el objeto 2 y 4 tienen una componente de ruido más importante en este eje. Esto se debe a su colocación en áreas donde las zonas de incertidumbre asociadas a los objetos son más grandes. Esto hace que la estimación de este objeto contenga un error mayor.

También podemos notar el error con respecto a la calibración de las cámaras. Cuanto más nos acerquemos a los bordes de las imágenes tanto más importantes serán las desviaciones fruto de los errores de calibración. En nuestro caso tenemos un movimiento de unos pocos centímetros (unos 5 centímetros) sobre una distancia de, aproximadamente, dos metros. Dicho movimiento se sigue de manera precisa, con la componente de ruido mencionada, achacable a la calibración y la colocación de los objetos.

Con este experimento hemos demostrado que el sistema de seguimiento que hemos desarrollado es capaz de localizar y seguir varios objetos dinámicos al mismo tiempo, objetivo central de esta tesis. La precisión, como hemos visto tanto en los experimentos de un objeto como en los de varios objetos, es del orden de centímetros dentro de una habitación de unos 50 metros cuadrados. Todo ello funciona con una velocidad de más de 20 iteraciones por segundo en un PC de escritorio.

Con esto ponemos punto final a la descripción de los experimentos de funcionamiento de la tesis. En el siguiente capítulo nos centraremos en las cuestiones relativas a la implementación que hemos realizado para llevar a cabo estos experimentos.

CAPÍTULO 6

Plataforma Experimental

Esta es una tesis de ingeniería por lo que, desde el principio, se ha puesto un especial énfasis en la implementación y prestaciones de todos los algoritmos aquí propuestos. La implementación ha tenido como fin tanto la realización de experimentos como el estudio de los resultados obtenidos y la generación de los gráficos o tablas mostradas en las secciones 4.7 y 5.5, referentes a los experimentos. Sin el desarrollo de todo el software usado no hubiera sido posible abordar la realización de una tesis como ésta, en la que la parte de estudio teórico y de prueba real son inseparables.

Este capítulo está dedicado a describir la implementación del software desarrollado, con especial énfasis en las decisiones de diseño que se han realizado. Entre el software que comentaremos se encuentra la aplicación encargada del seguimiento de uno o varios objetos en 3D. También describiremos otras herramientas auxiliares que hemos desarrollado, principalmente para el análisis de resultados, o para comprobar el funcionamiento de las técnicas descritas en algún problema real. Para más información puede consultarse el apéndice A.2, donde se presenta un resumen de otras herramientas o bibliotecas software que hemos usado durante el desarrollo de esta tesis, pero no han sido directamente programadas por el autor.

6.1. Sistema de seguimiento multicámara

METS (Multi-Eye Tracking System) es el programa encargado de probar todos los algoritmos desarrollados en esta tesis. METS gestiona la adquisición de imágenes, la elección del algoritmo de seguimiento a usar y el ajuste de los parámetros de éste. Además incorpora una interfaz gráfica, útil tanto para la realización de experimentos como para la depuración.

METS está programado completamente en C++ sobre el sistema operativo Linux (distribución Ubuntu 6.06.1 LTS Dapper). Cuenta con algo más de 7.500 líneas de código, dedicadas únicamente a los algoritmos y la gestión de imágenes. Hace uso de multitud de bibliotecas auxiliares como pueden ser VW (biblioteca dedicada a la visión por computador, descrita en los apéndices), OpenGL, GTKMM, libglademm y otras muchas.

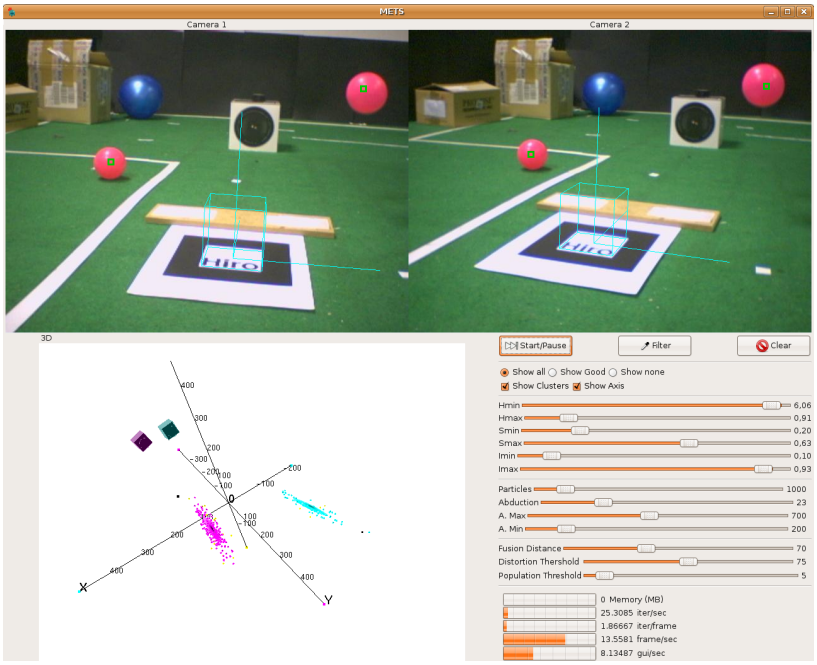

Figura 6.1: Captura de *METS (Multi-Eye Tracking System)* en funcionamiento.

En la figura 6.1 puede verse una captura de METS en funcionamiento. En ella se aprecian las imágenes capturadas por las dos cámaras conectadas al ordenador. También puede verse la escena 3D observada desde una cámara virtual en la que apa-

recen tanto las cámaras reales como la posición estimada del objeto, todo ello con respecto al mismo marco de referencia que definen los ejes que aparecen dibujados. Dichos ejes también se muestran superpuestos sobre las imágenes reales. A la derecha se encuentran los controles de funcionamiento y barras de desplazamiento para ajustar los parámetros de los filtros. Los controles permiten activar y detener el filtro o, si es necesario, reiniciarlo. Entre los parámetros ajustables tenemos los umbrales del filtro de color, el número de partículas, el parámetro de abducción o los parámetros del algoritmo de segmentación. Por último en la parte derecha, bajo las barras de desplazamiento que controlan los parámetros del algoritmo, pueden verse algunas estadísticas sobre la velocidad de ejecución y de captura.

El usuario puede configurar los parámetros del filtro desde la interfaz gráfica. Es posible modificar la posición de la cámara virtual para estudiar diferentes zonas de la escena, haciendo ampliaciones si es necesario. Se puede quitar o añadir información a la representación gráfica, como la posición de las partículas o los objetos localizados, permitiendo una descripción menos saturada de la escena. Todo ello se realiza con los botones que aparecen a la derecha de la imagen. También es posible mostrar las imágenes recibidas tras pasarlas por el filtro de color. De esta forma resulta más sencillo ajustar los parámetros de dicho filtro. Junto a la imagen filtrada se muestran las proyecciones de todas las partículas, permitiendo estudiar la evolución del filtro.

El diseño interno de METS está orientado hacia la modularidad. Su objetivo es ser una plataforma completa y de fácil modificación en la que probar diferentes algoritmos de seguimiento de manera sencilla. Por ese motivo METS separa las tareas comunes, como pueden ser la captura o el filtrado de imágenes, de las tareas específicas de cada algoritmo, como son las técnicas de remuestreo o los procesos de abducción.

En la figura 6.2 puede verse un diagrama de los componentes software de METS. Se muestra la separación entre los módulos comunes, como pueden ser los de captura de imágenes o filtrado por color, y los propios de cada método de seguimiento, marcados como “algoritmos”. Para añadir un nuevo modo de seguimiento únicamente es necesario crear un módulo software capaz de procesar las observaciones de entrada y devolver el conjunto de partículas representando la función de densidad de probabilidad. Los métodos de segmentación posteriores se encargarán de proporcionar las

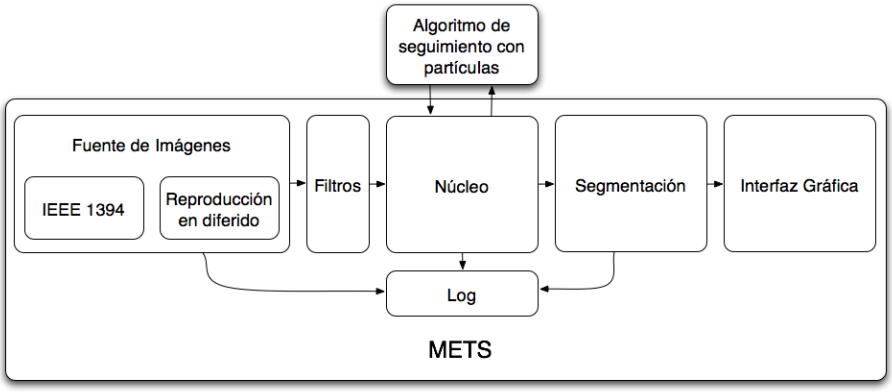

Figura 6.2: Diagrama software de METS.

mejores estimaciones de posición a partir de la nube de partículas.

El funcionamiento de METS es el siguiente. En primer lugar captura las imágenes desde algún dispositivo de entrada al ritmo impuesto por dicho dispositivo, preparando la secuencia de observaciones y aplicando los filtros necesarios. A continuación METS ejecuta tantas iteraciones del algoritmo de seguimiento como sea posible, usando las "observaciones" que acaba de crear hasta que capture una nueva imagen y pueda crear unas nuevas. Las observaciones incluyen toda la información que podría necesitar el algoritmo de seguimiento, como la imagen sin filtrar y filtrada, la posición de las cámaras y todos sus parámetros y el tiempo de captura. Los resultados se van mostrando por la pantalla continuamente, pero de tal forma que el coste computacional asociado con la interfaz gráfica no penalice el correcto desarrollo del algoritmo a estudiar, foco principal de METS.

Las imágenes se pueden adquirir de secuencias de ficheros guardadas en el disco o directamente de cámaras conectadas al bus IEEE 1394 (*Firewire*) en tiempo real. La utilización de cámaras pretende comprobar el funcionamiento del sistema en situaciones reales, donde el coste computacional viene limitado por la necesidad de velocidad del sistema. Esto es importante para el desarrollo de esta tesis, la cual gira en torno a aplicaciones en tiempo real en ordenadores convencionales.

Al usar secuencias de ficheros también se intentan respetar las restricciones de tiempo. El objetivo es que los resultados al usar ficheros en el disco duro sean comparables con las imágenes tomadas desde las cámaras directamente. Para evitar los

problemas de velocidad, las imágenes se precargan en memoria, para no depender de la velocidad de acceso al disco. Esto permite probar exactamente el mismo conjunto de imágenes con diferentes algoritmos y comparar sus resultados en cuanto a tiempo de cómputo, velocidad y precisión. Además, podemos emplear esta característica para ejecutar los algoritmos contra imágenes sintéticas generadas por algún programa de diseño 3D, como puede ser Blender[1], con el fin de conocer la verdadera posición de los objetos en todo momento y así poder realizar medidas fiables de calidad en un entorno simulado.

Actualmente sólo soportamos dos resoluciones diferentes, 320x240 y 640x480, ambas con 24 bits de color, tomadas tanto desde archivos como desde las cámaras reales. En el caso de las cámaras las usamos a 15 o a 30 fotogramas por segundo. Esta limitación viene de las características en cuanto a resolución y velocidad del modelo de cámara que usamos[2] y del propio bus Firewire, que funciona a 400Mbps. Preferimos usar la resolución más alta, 640x480, para comprobar los límites de funcionamiento del algoritmo, centrándonos en problemas que no aparecen en resoluciones más bajas. Al usar una secuencia de archivos la velocidad de captura no está tan limitada y es posible adaptarla a voluntad para probar las prestaciones a otras velocidades.

Una vez que se han capturado las imágenes METS realiza una o varias iteraciones del algoritmo de seguimiento sobre ellas. Los algoritmos soportados por METS son todos los comentados en los capítulos 4 y 5. Entre ellos podemos destacar los siguientes:

- Filtro de condensación,
- Filtro de partículas con una función de propuesta abductiva,
- Filtro de partículas con una función de propuesta abductiva y agrupamiento (*clustering*) de las partículas resultantes.

Cada uno de ellos tiene parámetros comunes, como pueden ser la configuración del filtro de color o el número de partículas, y parámetros propios, como el porcentaje

[1] http://www.blender.org/
[2] Las cámaras usadas son Apple iSight.

de abducción o la distancia de fusión para la segmentación. La mayor parte de estos parámetros se pueden controlar desde la interfaz gráfica.

Junto con las características comunes de METS, se han implementado una serie de módulos y funciones para el correcto funcionamiento de los algoritmos de seguimiento. En particular podemos mencionar el filtro de color, el sistema de abducción y los algoritmos de segmentación de partículas para el caso multiobjeto. Son los que se describen en el resto de este capítulo.

6.2. Filtrado de color

El filtro de color es posiblemente uno de los módulos con más consumo de CPU. Su objetivo consiste en filtrar todos los píxeles de la imagen para quedarse únicamente con aquellos que pertenecen a un rango de color determinado. Una de las ventajas de usar un método muestreado como el que planteamos es que no resulta necesario filtrar toda la imagen. Únicamente es necesario filtrar los puntos sobre los que queremos evaluar el modelo de observación, es decir, las zonas donde proyectan partículas. Esta característica reduce de manera importante el coste computacional. Aún así, la implementación de este filtrado debe hacerse lo más eficiente posible para los casos en los que:

- filtremos toda la imagen para mejorar la abducción del sistema,
- aumentemos mucho el número de partículas.

Por este motivo se ha puesto un especial énfasis en la optimización del filtro de color, como comentaremos a continuación.

Al usar el color como principal distinción entre objetos de la escena es necesario hacer hincapié en las condiciones de iluminación de la escena. La mayor parte de las cámaras de bajo coste, como las que pretendemos utilizar, trabajan en los espacios de color RBG o YUV. El primero emplea las componentes primarias de color, pero no trabaja de manera directa con magnitudes relacionadas explícitamente con la iluminación. Por dicho motivo, realizar un filtrado de color robusto frente a cambios

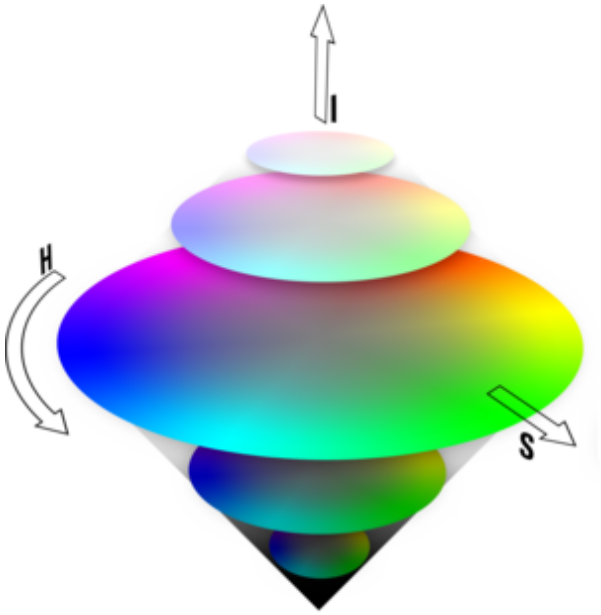

Figura 6.3: Espacio de color HSI.

de iluminación en este espacio se torna complicado, al estar todas las componentes influídas por esta magnitud.

El caso del espacio YUV es algo mejor. Aunque una de las componentes de YUV sí está relacionado con esta magnitud, la división de la componente de color en U y V no resulta tan intuitiva como RGB.

En nuestro caso hemos elegido realizar el filtrado de color en el espacio HSI. A diferencia de los otros modelos mencionados, el espacio HSI separa la componente de iluminación del resto de la información de color. Las tres componentes que utiliza este modelo son (ver figura 6.3):

1. el *tinte* (*hue* o H), o color puro propiamente dicho, se corresponde con el ángulo del círculo de color de la figura 6.3;

2. la *saturación* (*saturation* o S), muestra la viveza de un color determinado y se representa en la figura 6.3 como el radio dentro del cono; y

3. la *intensidad* (*intensity* o I), que es la iluminación del color y se representa como la altura en el cono de color.

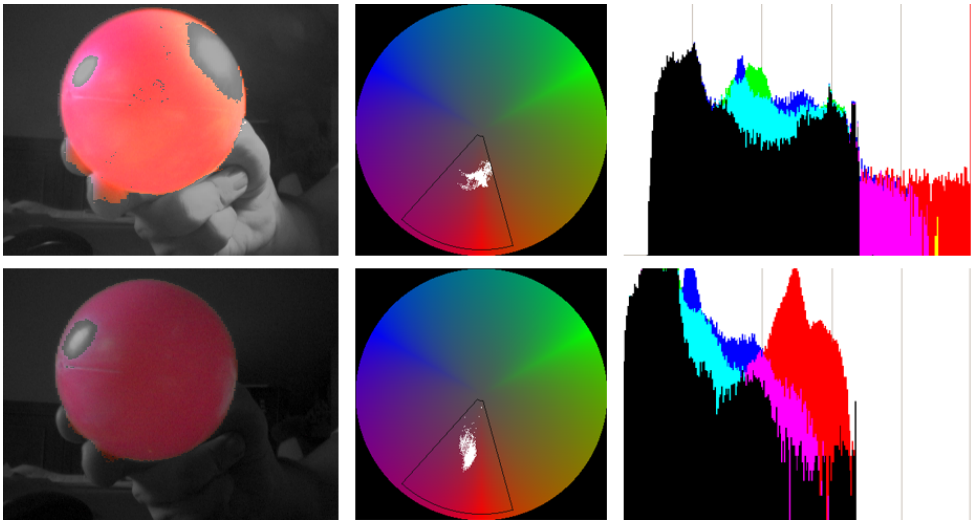

Figura 6.4: Filtrado de color frente a cambios de iluminación de la escena.

Toda la información de luminosidad se concentra en la componente I, dejando la información propia del color en las componentes H y S. El filtro de color se centrará únicamente en estas dos componentes para aceptar o descartar un color. Por ese motivo resulta inmune, hasta cierto límite, a las variaciones de iluminación. Estas variaciones son muy comunes y se pueden dar por cambios ambientales o por diferentes configuraciones de las cámaras. En el espacio de color RGB un cambio de iluminación afecta por igual a las tres componentes de color: rojo, verde y azul. Por ese motivo resulta más difícil realizar un filtro sencillo capaz de resistir los cambios de iluminación en RGB. En la figura 6.4 podemos ver dos ejemplos de filtrado de la misma escena con dos condiciones diferentes de iluminación. En la figura se muestra el disco de color del espacio HSI (componentes H y S). Vemos cómo dichas componentes no cambian de manera significativa de una imagen a otra. Como contraste podemos ver el histograma en las componentes RGB y cómo el cambio en sus valores resulta más difícil de seguir.

El principal problema a la hora de realizar el filtrado en el espacio de HSI es la conversión de la imagen entre cada uno de los espacios de color usados. Como ya se ha indicado, las cámaras de bajo coste suelen trabajar en formato RGB o YUV por lo que será necesaria una conversión entre estos espacios de color y HSI para poder

disfrutar de las ventajas de éste. En particular, las cámaras que usamos proporcionan las imágenes en formato YUV y de ahí se convierten a RGB. Este paso es necesario puesto que la interfaz gráfica que empleamos, basada en GTKMM y OpenGL, trabaja únicamente con imágenes RGB. La conversión entre YUV y RGB la realiza el software de captura de imágenes, implementado en la biblioteca VW (ver sección A.2). La conversión es software, lo que pone un límite a la velocidad máxima a la que se pueden capturar imágenes, sobre todo a resoluciones altas. Una vez capturadas las imágenes y convertidas a RGB, METS almacena una copia para aplicarla los diferentes filtros disponibles.

La relación existente entre RGB y HSI está resumida en las expresiones 6.1, 6.2 y 6.3, donde R, G y B son la cantidad de componente roja, verde y azul, respectivamente, medidas entre 0 y 1. I nos da la intensidad luminosa, entre 0 y 1, H el ángulo de color, entre 0 y 2π y S, la saturación entre 0 y 1.

$$H = \cos^{-1}\left(\frac{\frac{1}{2}[(R-G)+(R-B)]}{\sqrt{(R-G)^2+(R-B)(G-B)}}\right) \tag{6.1}$$

$$S = 1 - \frac{3\,\text{mín}(R,G,B)}{R+G+B} \tag{6.2}$$

$$I = \frac{1}{3}(R+G+B) \tag{6.3}$$

El proceso de conversión es costoso, implica varias operaciones en coma flotante y una relación trigonométrica por cada píxel de la imagen. Para aumentar la velocidad usaremos una tabla precalculada. La tabla funciona como una *look up table* o LUT y estará indexada por el color RGB, de forma que sabiendo el color del punto la conversión resulta directa. El ahorro de tiempo al filtrar la imagen completa es considerable al cambiar el cálculo de expresiones complejas como las anteriores por la consulta en una tabla.

Para evitar que el tamaño de la tabla sea demasiado grande se descartan los dos bits menos significativos de cada canal de color. De esta forma se construye un índice de 18 bits, en lugar de los 24 bits del píxel RGB. Esto reduce el tamaño de la tabla desde 16 millones de posiciones a 260.000. Descartando los bits menos significativos

de cada canal de color, podemos construir una tabla de tamaño manejable sin perder información relevante (de color[3]). Por cada índice de la tabla se filtra el color RGB usando la conversión HSI anterior y se guarda únicamente el resultado del filtrado. De esta forma evaluar el modelo de observación para cada partícula será más sencillo.

Consideramos que un color pasa el filtro si tanto sus componentes de tinte como de saturación, H y S, están dentro del rango definido por los valores máximos y mínimos del filtro. Esto define una sección de arco en el círculo HS, como se ve en la figura 6.4. Los umbrales se pueden poner como un entorno centrado en el color buscado con cierto margen de tolerancia.

Sin embargo, estas componentes no son del todo independientes a la iluminación, por lo que ésta también tendrá que tenerse en cuenta. Cuando los puntos están próximos al blanco o al negro (valores extremos de I), el rango de variación de la saturación baja y la precisión es cada vez más pobre en cuanto al color (nótese el estrechamiento del círculo de color en la figura 6.3). En estos casos no seremos capaces de discernir el color del objeto. Por este motivo eliminaremos los puntos con valores de I cercanos a los límites, es decir, los colores muy oscuros o los excesivamente claros. Si no tomáramos esta precaución la conversión entre colores RGB y HSI puede no ser correcta, haciendo que el filtro en H y S acepte píxeles que en realidad pertenecen a puntos cercanos al negro o al blanco.

6.3. Abducción de nuevos objetos

Como vimos en las secciones 4.6 y 5.3, los algoritmos usados emplean un método de abducción para localizar nuevos objetos según aparecen en la escena. METS también implementa dichas funciones, permitiendo agregar los objetos de forma muy rápida.

Esto se debe a su capacidad de buscar en regiones de interés tan pronto como el filtro de color encuentra zonas de la imagen que pueden corresponderse a un objeto. En los capítulos 4 y 5 se han explicado las bases teóricas de esta característica. En particular en la sección 4.6 introducimos el concepto de abducción en el marco de

[3]Las cámaras de bajo coste empleadas tienen un margen de error parecido a estos 2 bits de precisión.

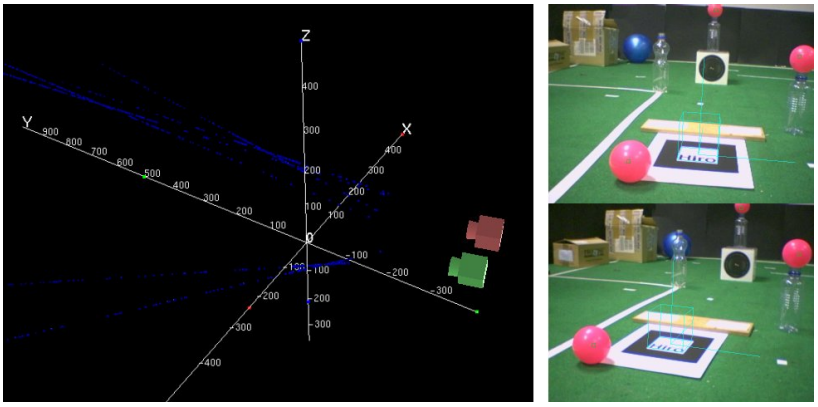

Figura 6.5: Partículas colocadas por METS usando la técnica de abducción.

los filtros de partículas, haciendo uso de la función de propuesta. A continuación discutiremos alguno de los detalles de la implementación.

El objetivo de la abducción es proporcionar nuevas hipótesis para buscar en regiones prometedoras del espacio de estados. Para ello localiza las zonas de la imagen que pueden provenir de un objeto y, posteriormente, coloca nuevas partículas sobre la línea de retroproyección de los nuevos puntos localizados en cada imagen. Dado que la proyección de estos puntos se corresponde con la proyección del objeto localizado, es probable que alguno de éstos se corresponda con la verdadera posición 3D de dicho objeto. En la figura 6.5 podemos ver una captura de METS donde aparecen partículas procedentes únicamente de la abducción de varios objetos.

En primer lugar, para buscar los puntos que pasan el filtro impuesto por el modelo de observación, será necesario filtrar completamente las imágenes de entrada. Esto es un proceso bastante costoso, aunque la mejora de la calidad de los resultados compensa este coste extra.

Sería posible rebajar el coste computacional aplicando la abducción cada cierto intervalo temporal, en lugar de hacerlo con cada imagen recibida. Es necesario que dicho intervalo esté en relación con el tiempo de reacción que requiera la aplicación, puesto que hasta que no ejecutemos la abducción es posible que no se localicen los objetos que estén presentes en la escena. Podemos ver la abducción como el ajuste de grano grueso del sistema, el que marca las direcciones en las que buscar. El algoritmo de condensación se encargará de realizar el ajuste de grano fino. De esta forma será

suficiente ejecutar la función de abducción con una periodicidad tal que nos permita reaccionar correctamente a la aparición de nuevos objetos. Si estamos trabajando a 30 fps quizá baste con ejecutar el proceso de abducción una de cada 15 o 30 veces, es decir, cada medio segundo o cada segundo. Estos tiempos parecen razonables para detectar un nuevo objeto en un entorno natural, en el que los objetos aparecerán cada cierto tiempo.

Tampoco es necesario aplicar el filtrado a todas las cámaras a la vez. Las hipótesis generadas en la línea de visión de cada cámara, se cruzarán en las verdaderas posiciones de los objetos. Pero esta característica no se tiene en cuenta a la hora de crear hipótesis. Al no ser necesario buscar los puntos de cruce, bastará con elegir una cámara y usarla como base para obtener las líneas de visión. Para evitar las oclusiones que afecten a una de las cámaras, se puede cambiar la cámara usada para la abducción de una iteración a la siguiente.

Por último, no es necesario filtrar la imagen en su totalidad. El objetivo es localizar las zonas de la imagen que resulten relevantes para el modelo de observación, con el fin de construir una línea de proyección. Pretendemos encontrar zonas de interés, no la posición de cada uno de los píxeles. Dado que los objetos que buscamos tienen un tamaño fijo, su proyección ocupará una región de la imagen mayor a un píxel (si no está demasiado lejos). Si reducimos la resolución de la imagen todavía podremos encontrar alguno de esos puntos de interés. De esta forma podemos pasar a filtrar, por ejemplo, 1 de cada 4 píxeles o 1 de cada 9, con la reducción de coste computacional que esto conlleva y localizando aún la información sobre las zonas de interés.

Todas las reducciones de coste computacional que presentamos son a costa de reducir la precisión del sistema a la hora de localizar las proyecciones de los objetos. Es posible reducir la precisión gracias a que la abducción funciona correctamente con indicios parciales sobre la posición real del objeto. Para funcionar correctamente la abducción emplea información parcial para colocar nuevas partículas sobre las zonas más prometedoras del espacio de estado. Será la propia dinámica del filtro de partículas la que se encargará de explotar esas zonas para obtener las estimaciones correctas. Mejorar la precisión de la abducción tendría unas consecuencias perjudiciales para el sistema. El aumento del coste computacional que esto conlleva restaría tiempo a la dinámica del filtro de partículas. Esto reduciría el número de fotogramas por segundo

que podemos usar, obligando a emplear modelos de movimiento más anchos. Todo ello disminuiría la precisión y la frecuencia de las estimaciones.

Una vez que tenemos filtrada la imagen, debemos segmentar las zonas que resultan de interés. Existen muchos algoritmos de segmentación 2D que podrían usarse en este caso. Muchos de ellos requieren varias pasadas o buscan soluciones que crecen desde unos puntos iniciales. El problema que suelen plantear es, de nuevo, su coste computacional. En nuestro intento de reducir el coste computacional al máximo, estamos dispuestos a sacrificar algo de calidad en la abducción. Para ello simplemente calculamos el histograma acumulado en cada eje de la imagen obtenida tras el proceso de filtrado. Los picos del histograma de color nos dan información sobre la posición, en cada eje, de los objetos en la imagen. Para evitar los puntos aislados que pasen el filtro, pasaremos la información del histograma por una función umbral. De esta forma ignoraremos las zonas en las que aparezcan muy pocos píxeles. También puede darse el caso en el que en un objeto aparece un brillo o una zona que no pase el filtro. Con el fin de no partir esta zona del espacio como dos objetos diferentes, implementaremos un cierto grado de histéresis que una regiones próximas del mismo color. Una vez que tenemos los máximos de cada eje obtendremos la posición aproximada de del objeto.

Este proceso no deja de ser una simplificación, por lo que puede llevarnos a errores en algún caso. Por ejemplo, en el caso de tener más de un objeto en la misma imagen, aparecerán varios máximos locales en los histogramas. No sabemos de manera directa qué combinaciones de los máximos de un eje y los del otro se corresponden con objetos reales. En la figura 6.6 vemos los histogramas correspondientes a una imagen con dos objetos, aplicando la función de umbral antes mencionada. La combinación de los máximos nos proporciona cuatro posibles localizaciones para los objetos, pero únicamente dos son posibles. Para refinar la segmentación de manera correcta sería necesario realizar una segunda pasada.

Gracias a la robustez del sistema sería suficiente con proporcionar estas cuatro líneas de visión. Sería el modelo de observación el que se encargaría de eliminar los puntos incorrectos y hacer que el filtrado funcione correctamente. Aún así, para mejorar las prestaciones, en la implementación actual comprobamos que los puntos usados para generar las líneas de proyección tengan el color adecuado. En caso con-

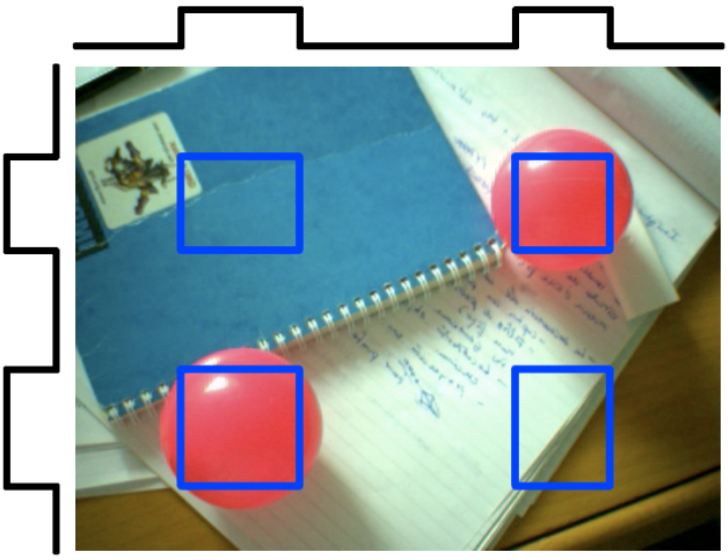

Figura 6.6: Problemas con objetos fantasma en la extracción de líneas de abducción. El sistema detecta incorrectamente 4 objetos en la primera pasada.

trario serán rechazados. De esta forma los rectángulos pintados en rojo de la figura 6.6 serán descartados, dejando únicamente las dos líneas de visión correctas.

6.4. Segmentación del conjunto de partículas y estimación de posiciones

El conjunto de puntos que proporciona el filtro de partículas representa una distribución de probabilidad, en la que pueden aparecer varios máximos locales. METS incluye también los algoritmos necesarios para realizar una segmentación entre las partículas, de tal forma que se asocien con cada uno de los objetos presentes en la escena. En un paso posterior se estimará la posición de cada objeto utilizando la información de las partículas de cada grupo creado.

Los algoritmos de segmentación usados han sido descritos en la sección 5.4 del anterior capítulo por lo que no entraremos en el detalle de su funcionamiento. Nos centraremos aquí en describir las características de su implementación y estudiar su

eficiencia.

El algoritmo de segmentación implementado se compone de dos elementos: uno que agrupa las partículas usando un algoritmo de maximización de la esperanza o *expectation maximization* (EM) [Bis95, Mac03] y otro que ajusta el número de grupos a buscar dado por el anterior algoritmo. Esto es así porque el algoritmo EM necesita conocer *a priori* el número de objetos que se encuentran en la escena, o lo que es lo mismo, el número de grupos que debe formar.

El problema de estos algoritmos es que son de naturaleza iterativa. Para dar resultados satisfactorios deben ejecutarse cierto número de veces hasta converger a un valor adecuado. Así, cada vez que ejecutemos una iteración de METS será necesario ejecutar unas cuantas de EM y el algoritmo de ajuste del número de objetos. Esto puede penalizar fuertemente el funcionamiento del sistema completo.

Para evitar los problemas del coste computacional, en METS la implementación del algoritmo EM está muy acoplada con el algoritmo de ajuste del número de grupos. Además se hace uso de la naturaleza secuencial del filtro de partículas. Si el sistema funciona lo suficientemente rápido, los grupos de una iteración a la siguiente serán muy similares. Aunque las partículas cambien de una iteración a la siguiente, los grupos se mantendrán, más o menos, en las mismas zonas del espacio. De esta forma, se pueden usar los resultados de EM de una iteración como puntos de partida para la iteración siguiente, reduciendo significativamente el número de iteraciones necesarias del algoritmo de segmentación.

La implementación consta de tres pasos diferenciados. Por un lado están los dos referentes al algoritmo EM: el paso E y el paso M. Éstos siguen las expresiones 5.18 y 5.19, de la sección 5.4. El tercer paso es el correspondiente al ajuste del número de grupos en el algoritmo EM, al que llamamos paso K. Este paso es similar al algoritmo ISODATA [BH65]. Los tres pasos se combinan intentando que el coste computacional asociado con el proceso completo no crezca demasiado.

La implementación actual de METS realiza tres pasos EM y un paso de ajuste del número de grupos. De esta forma se da tiempo a EM para perfilar los segmentos aunque se permiten cambios con el paso K. Repetimos esta operación dos veces por cada iteración de METS. Mantenemos el número de iteraciones bajo y fijo. Esto puede hacer que el algoritmo no converja en un momento dado, pero usaremos sus

resultados como base para la siguiente iteración de METS. De esta forma terminará convergiendo a las verdaderas posiciones de todos los objetos.

Los pasos E y M no trabajan sobre el conjunto total de partículas, sino únicamente sobre aquéllas suficientemente buenas. Para ello descartamos las partículas con un valor de $p(\boldsymbol{z}_k \mid \boldsymbol{x}_k^i)$ por debajo de un umbral. Las partículas con valores altos de $p(\boldsymbol{z}_k|\boldsymbol{x}_k^i)$ serán aquellas que se encuentren cerca de los máximos de la distribución de probabilidad $p(\boldsymbol{x}_k|\boldsymbol{z}_k)$ y por ende de las posiciones de cada objeto. Si no se descartaran las partículas con baja verosimilitud estaríamos introduciendo ruido en el algoritmo de segmentación, con los problemas que esto conlleva a la hora de realizar las agrupaciones. Esto es aún más evidente cuando se emplean partículas procedentes de la abducción, como se muestra en la figura 6.7.

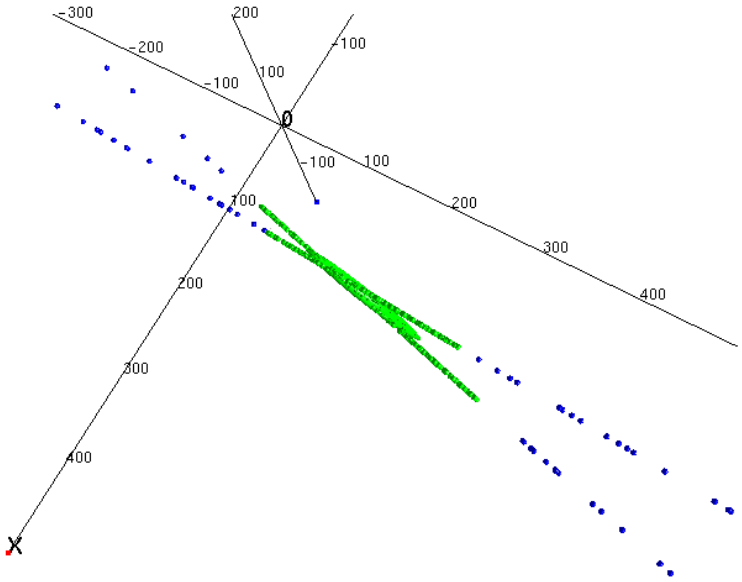

Figura 6.7: Las partículas que proceden de la abducción pueden suponer un problema a la hora de realizar la segmentación. Por ese motivo se eliminan si no tienen un peso suficientemente alto.

Cada grupo se representa por una gaussina tridimensional. El algoritmo optimizará tanto la media como todos los valores de la matriz de covarianza (orientación) para cada uno de los grupos. Los parámetros de cada gaussiana serán independientes a los del resto de gaussianas. A parte de estos valores, los pasos E y M también

calculan los siguientes parámetros:

- Relación de pertenencia de cada partícula a cada grupo,
- grado de distorsión de cada grupo,
- población de cada grupo,
- partículas que no están asociadas a ningún grupo.

El último valor está relacionado con la distorsión máxima que permitimos. La distorsión a la que hacemos referencia será un valor que nos indica cuán compactas están las partículas entre sí. Podemos tomar varios valores para representar la distorsión. En nuestro caso hemos elegido emplear la suma de los valores de la matriz de covarianza. Dichos valores están relacionados con la distancia de todas las partículas a la media del grupo.

Para evitar que un grupo aumente su distorsión y cubra el espacio ocupado por dos objetos próximos, no consideramos las partículas que se encuentran muy lejos de un grupo para redefinir éste. De esta forma aparecen una serie de partículas que no están asociadas a ningún grupo, demasiado alejadas de cualquiera de éstos como para formar parte de uno de ellos. A estas partículas las denominamos partículas *parias* y sirven para descubrir zonas del espacio que no están explicadas por ninguno de los grupos actuales.

El paso K es el encargado de modificar el número de grupos que toman los pasos EM como parámetro. Para ello tiene en cuenta los valores calculados en los pasos E y M de la siguiente forma:

- Si hay dos grupos suficientemente cerca, éstos se fusionan en un único grupo en punto medio de los mismos.
- Si la población de un grupo está por debajo de un determinado umbral (hay muy pocas muestras) el grupo se elimina.
- Si la distorsión de un grupo es mayor a un determinado umbral, el grupo se divide en dos subgrupos. Éstos se colocan en dos puntos en sentidos opuestos

siguiendo el eje de máxima varianza, desde la media del grupo eliminado, y separados por una distancia igual a una desviación típica.

- Si hay más partículas parias de las permitidas se crea un nuevo grupo eligiendo como centroide una partícula paria al azar.

La inicialización del algoritmo se hace a cero grupos. Cuando aparezca un número suficiente de partículas con verosimilitud alta se creará un nuevo grupo para ellas y así continuamente.

En un primer lugar este algoritmo convergerá al número de objetos presentes en la escena. Su generalidad permite que también sea capaz de seguir los cambios en el número de objetos, por aparición o desaparición de objetos, como han mostrado los experimentos descritos en 5.5. La dinámica de funcionamiento habitual es la siguiente: Cuando aparezca un nuevo objeto, el sistema de abducción colocará nuevas partículas en las zonas del espacio prometedoras. Aquellas que caigan sobre un objeto verdadero obtendrán valores altos de probabilidad en el modelo de observación. Si no existe un objeto en esa zona del espacio, por ejemplo porque el objeto acaba de aparecer, las partículas se marcarán como parias. Si aparece un número suficiente de partículas parias se creará un nuevo objeto, que convergerá a la posición correcta del objeto.

El proceso de desaparición de objetos es simétrico a éste. En este caso, al desaparecer el objeto, el modelo de observación penalizará a todas las partículas que estuvieran anteriormente relacionadas con el objeto. Éstas serán eliminadas por el proceso de remuestreo y, al llegar al paso de ajuste de grupos, este grupo será eliminado por no tener suficientes partículas que lo soporten.

Para finalizar, la estimación de la posición de cada uno de los objetos se toma de la media de cada una de las gaussianas. De esta forma no es necesario realizar más cálculos sobre las partículas segmentadas para obtener las posiciones de los objetos.

6.5. Imágenes y reproducción en diferido

Para terminar con la descripción de la plataforma METS comentaremos otras características del programa.

METS soporta tanto el uso de cámaras en tiempo real, como de secuencias de vídeo pregrabadas. Aunque la aplicación está orientada al funcionamiento con cámaras reales, la utilización de vídeos aporta una gran ventaja: podremos simular distintos algoritmos usando exactamente los mismos datos de entrada por lo que los resultados podrán compararse entre sí. Igualmente, podemos usar el mismo algoritmo parametrizado de varias formas diferentes, estudiando así el efecto de cada parámetro de manera aislada. Si no empleáramos exactamente los mismos datos de entrada, no podríamos asegurar que los cambios en los resultados se debiesen a la variación de los parámetros o a los cambios en las imágenes.

La velocidad de reproducción está sincronizada con la velocidad de captura, con el fin de obtener resultados comparables tanto en las ejecuciones en directo como en diferido. No obstante, también es posible variar la velocidad de ejecución, si así se desea. De esta forma se pueden emular, vía software, cámaras más rápidas de las que disponemos, capaces de proporcionar 30 o 60 fps a 640x480 píxeles a partir de secuencias de vídeo.

Para completar la capacidad de reproducción en diferido, METS cuenta con un completo sistema de *log*, capaz de almacenar tanto la información acerca de los algoritmos usados (partículas, estimaciones), como la de configuración, o incluso las imágenes usadas. Todo ello sin afectar a las prestaciones globales del sistema. Las imágenes se guardan temporalmente en memoria, con el fin de no gastar recursos y tiempo en la codificación y volcado a disco. El limite para la secuencia de vídeo a grabar viene marcado por la memoria de la máquina que usemos (el PC de prueba contaba con 2 GB de memoria RAM). Una ver terminada la grabación, METS guardará todas las imágenes en el disco duro. De esta forma podremos realizar un análisis en diferido del funcionamiento de METS y de su evolución, midiendo precisión, tiempo de convergencia o reparto de partículas entre diferentes objetos. La mayor parte del análisis en diferido se realiza con otra herramienta, llamada *ParticlePlayer*. Esta herramienta está desarrollada completamente en Python usando la biblioteca gráfica

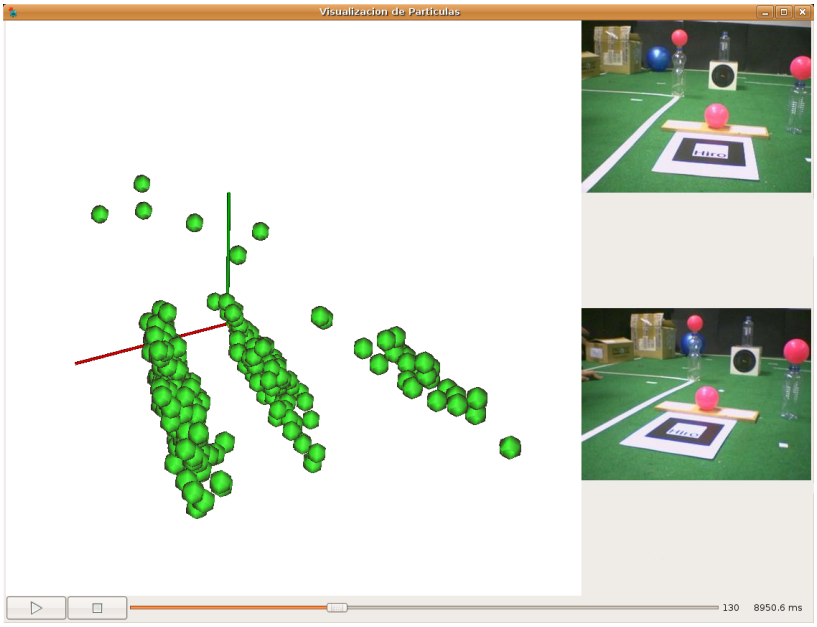

Figura 6.8: Captura de *ParticlePlayer* con tres objetos.

VTK. Podemos ver una captura de la aplicación en funcionamiento en la figura 6.8.

Como ya hemos mencionado, el objetivo de METS es ejecutarse con la suficiente rapidez, por lo que la eficiencia es uno de sus principales requisitos. Con la implementación actual el sistema funciona a unas 30 iteraciones por segundo en un PC de escritorio convencional (Pentium(R) 4 CPU 3.00GHz). Estos tiempos incluyen la captura, el filtrado de las dos imágenes a 640x480 y el proceso de segmentación para 1.000 partículas. Realizando alguna de las optimizaciones comentadas en el capítulo 5, como la abducción con una única imagen, se pueden alcanzar hasta 60 iteraciones por segundo. El límite de velocidad viene dado principalmente por el filtrado de la imagen, usado por la abducción, y por el proceso de segmentación de partículas. En el algoritmo de condensación para un único objeto, que no usa un filtrado exhaustivo, ni aplica el algoritmo EM, se pueden superar las 130 iteraciones por segundo en el mismo hardware.

6.6. Aplicaciones

Otra característica interesante de METS es que puede ser empotrado dentro de otras aplicaciones. Una vez empotrado proporciona información 3D de la posición de uno o varios objetos. De esta forma se puede usar un joystick o ratón visual basado en METS dentro de otra aplicación. La funcionalidad se ofrece como una biblioteca que el usuario puede importar a su proyecto. Entre las características que ofrece, se incluye una pequeña ventana con los controles básicos de METS que se puede añadir a la nueva aplicación (figura 6.9). Hemos desarrollado esta biblioteca con el fin de realizar aplicaciones que ilustren el funcionamiento del seguimiento visual frente a algún problema en particular. Dicha biblioteca proporciona interfaces en C++ y Python, sin sufrir penalización alguna en cuanto a las prestaciones del sistema se refiere.

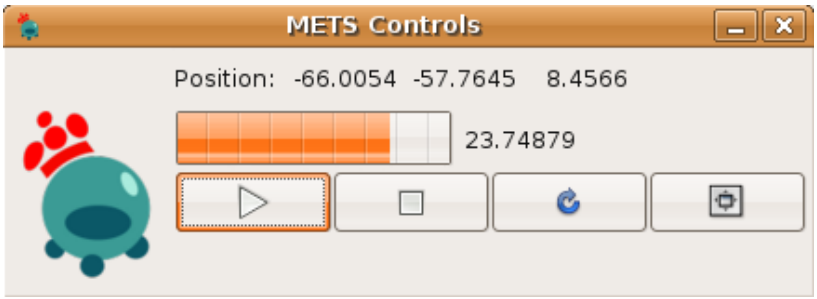

Figura 6.9: Controles de METS embebido.

Empleando la biblioteca de METS con tecnología derivada de esta tesis, se han desarrollado unas aplicaciones, haciendo énfasis en facetas diferentes a la cubiertas por METS. Éste es el caso de *Watcher*, una aplicación de videovigilancia que emplea 4 cámaras al mismo tiempo. En la figura 6.10 se puede ver una captura del programa en ejecución. El algoritmo de seguimiento empleado es el descrito en el capítulo 4, sin incluir la abducción. La parte relevante de este programa es la utilización de 4 cámaras y de varias fuentes de información al mismo tiempo: color y movimiento. El objetivo del programa era probar las hipótesis sobre el funcionamiento del marco bayesiano para fusionar la información extraída de cada cámara y de cada fuente de información. Su funcionamiento está descrito en la sección 4.4.

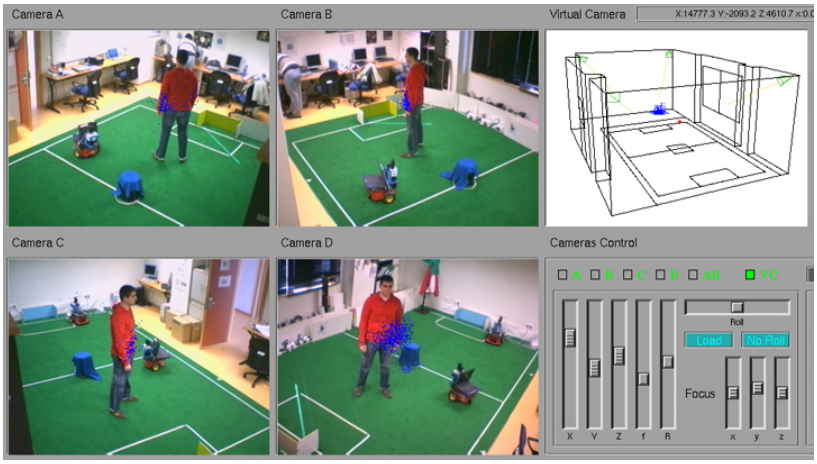

Figura 6.10: Captura de *Watcher*.

Watcher se ha empleado para probar el funcionamiento de los algoritmos descritos en un entorno real. Su utilidad se centra en las aplicaciones de videovigilancia. En particular, es capaz de vigilar una habitación y disparar una alarma, por ejemplo, cuando alguien o algo se acerque a un determinado punto de la misma (descrito en 3D).

Figura 6.11: Captura de *ElderCare*.

La capacidad de localizar la posición 3D exacta permite desarrollar otras aplicaciones. Como ejemplo hemos desarrollado la aplicación *ElderCare* para el cuidado de personas mayores, en particular la vigilancia de personas que vivan solas. Instalando

varias cámaras en la vivienda de una persona mayor podremos seguir sus movimientos y detectar cuando se ha caído al suelo. La manera de hacerlo es estimar la posición de su cuerpo y detectar cuando está muy cerca del plano del suelo. Si la persona permanece demasiado tiempo en el suelo el sistema activa una alarma para los cuidados sanitarios acudan a ayudarle, como se muestra en la figura 6.11. Al ser un sistema automático tenemos dos grandes ventajas. Por un lado, es posible instalar el sistema en muchas viviendas o residencias de ancianos por un coste bajo. Por otro lado, la intimidad de las personas no tiene porqué verse perjudicada dado que el vídeo solo es analizado por un ordenador y las imágenes no quedan grabadas en ningún lado.

Actualmente estas dos aplicaciones, *Watcher* y *ElderCare*, se han implementado como prototipos para mostrar la utilidad de los algoritmos en los que trabajamos y únicamente se han implementado para seguir a un objeto. La extensión a multi-objeto únicamente necesita añadir los algoritmos descritos en el capítulo 5, incluidos ya en METS.

CAPÍTULO 7

Conclusiones

En este documento se ha resumido el trabajo realizado en esta tesis doctoral, centrada en el seguimiento visual en 3D de múltiples objetos. Hemos descrito la cuestión genérica del seguimiento de uno o varios objetos, dentro del campo de la visión por computador y cómo la hemos abordado, validando nuestra propuesta experimentalmente.

A continuación vamos a centrarnos en las conclusiones y aportaciones de cada uno de los capítulos. Posteriormente señalaremos publicaciones relacionadas con este trabajo e indicaremos una serie de posibles líneas futuras a seguir.

7.1. Resumen de aportaciones

En el capítulo 1 presentamos el campo de la visión por computador y el seguimiento de objetos. Se trata de un campo en continuo avance, gracias a las mejoras en la capacidad de cómputo de los ordenadores y la reducción de precios de las cámaras. Dentro de un campo tan grande como la visión por computador, nos hemos centrado en el seguimiento visual, delimitando el área al que dirigir nuestro trabajo.

En particular hemos trabajado sobre el seguimiento visual en tres dimensiones de *varios* objetos al mismo tiempo. El seguimiento de un único objeto ha sido resuelto

satisfactoriamente con diferentes técnicas en la comunidad científica. Sin embargo, el seguimiento de varios objetos al mismo tiempo todavía está abierto. Las dificultades aparecen al gestionar, al mismo tiempo, información de varios objetos. Resulta más complejo cuando los objetos tienen una apariencia visual similar.

En el capítulo 2 hemos presentado otros trabajos relevantes que han abordado previamente este campo. Una contribución que hemos realizado es formalizar y analizado las aproximaciones que sirven para el seguimiento, estructurando los métodos en cuatro grupos en función de las técnicas usadas.

En el capítulo 3 hemos presentado los fundamentos matemáticos sobre los que se sustentan las técnicas que hemos empleado: los métodos de Monte Carlo y los filtros de partículas. Se trata de una revisión extensa con el fin de marcar tanto las características principales de estos algoritmos como sus fortalezas y sus debilidades. En particular hemos hecho hincapié en el problema de la degeneración de los pesos de las partículas y de la degradación de las partículas. Ambos problemas aparecen en las soluciones tradicionales para los filtros de partículas y han sido abordados en muchas aproximaciones. También hemos visto cómo las soluciones propuestas, a pesar de ser correctas para el seguimiento de un único objeto, resultan insuficientes para el seguimiento de varios objetos. En ese caso la degeneración de los pesos o el empobrecimiento de las partículas desembocan en una pérdida de información sobre la posición de los objetos a seguir.

En el capítulo 4 se ha trabajado sobre el seguimiento de un único objeto. Nuestra contribución aquí consiste en adaptar un filtro de partículas para el seguimiento visual 3D de manera satisfactoria, combinando la información visual de varias cámaras en el marco probabilístico que proporcionan los filtros de partículas. Gracias a esta naturaleza hemos comprobado cómo resulta posible combinar fácilmente tanto la información procedente de varias cámaras como diferentes tipos de información, como pueden ser el color y el movimiento. Las ventajas de presentar una combinación probabilística son varias. En primer lugar, nos proporciona un marco de trabajo ampliamente estudiado y con sólidas bases matemáticas. Además, resulta sencillo

trabajar con modelos de probabilidad para el problema del seguimiento visual.

Los modelos que usamos, el modelo de observación y el modelo de seguimiento, se han diseñado de tal forma que sean lo más genéricos posibles. Su conocimiento sobre los objetos a seguir radica en su color y la suposición de la continuidad espacial en su movimiento. Una contribución realizada es que hemos comprobado que el sistema es capaz de seguir un objeto de manera correcta empleando únicamente esta información. Esto corrobora nuestra hipótesis, presentada al inicio de la tesis, acerca de la información necesaria para seguir un objeto. Es posible localizar y seguir a un objeto empleando poca información, únicamente mediante indicios generales de su presencia. La localización y compatibilidad de esos indicios a lo largo del tiempo son pruebas suficientes para la localización correcta del objeto. Además hemos comprobado experimentalmente este funcionamiento.

En cuanto a la suposición de la continuidad del movimiento, resulta correcta para movimientos en el mundo real, siempre y cuando la velocidad de captura de las cámaras sea suficientemente rápida en comparación con la velocidad de movimiento. En caso contrario existirán discontinuidades en el movimiento observado del objeto, que deben ser modeladas de una manera diferente.

Los algoritmos empleados en los sistemas de seguimiento mono-objeto que hemos desarrollado son dos: el algoritmo de condensación y el filtro de muestreo enfatizado. El primero de ellos ha sido empleado profusamente en este tipo de problemas por los buenos resultados que proporciona. En nuestro caso hemos adaptado el modelo de observación y de movimiento para el problema de seguimiento en 3D. Los experimentos llevados a cabo sobre nuestra implementación muestran que se puede conseguir un seguimiento correcto con un coste computacional bajo. El sistema es capaz de funcionar a unas 130 iteraciones por segundo con 1.000 partículas y empleando dos imágenes de 640x480 píxeles de resolución y 32 bits de información de color. Esto nos da una idea de la velocidad de funcionamiento del sistema.

La principal causa del funcionamiento desahogado del sistema es que no resulta necesario realizar un filtrado completo de las imágenes. En vez de analizar completamente las imágenes y reconstruir la realidad desde ahí, las técnicas propuestas usan la aproximación contraria. La solución se hipotetiza y se comprueba su validez contra las observaciones reales. Si repetimos este proceso muchas veces con hipótesis

independientes entre sí, por probabilidad, alguna de ellas será parcial o totalmente correcta. Aunque este proceso parezca muy laborioso resulta mucho menos costoso que utilizar la aproximación directa (construcción de soluciones 3D) cuando no existe una relación fácil de invertir entre las observaciones y el estado y es posible reutilizar mucha información de una iteración a la siguiente.

Las partículas en este escenario pueden considerarse como recursos de búsqueda. Nos permiten comprobar o buscar hipótesis dentro del espacio de estados. Ésta es la principal característica para elegir este tipo de aproximaciones frente a otras. Sin embargo, debemos ser conscientes de sus limitaciones. Durante los experimentos hemos mostrado cómo la búsqueda aleatoria o muy dependiente de las observaciones actuales, es decir, la colocación de hipótesis sin emplear las observaciones actuales, es la causa de problemas tales como la convergencia lenta o la pérdida de objetos tras movimientos bruscos o nuevas incorporaciones. Para evitar esto, es conveniente no emplear un sistema completamente ciego a la hora de colocar las partículas sobre el espacio de estados.

Por ello hemos desarrollado la función de abducción que, empleada junto al algoritmo de muestreo enfatizado, es capaz de colocar las partículas en aquellas zonas que resultan prometedoras a la vista de las observaciones actuales. Las partículas propuestas por la función de abducción recibirán un peso de acuerdo a lo correctas que son, empleando para ello el modelo de observación.

La función de abducción que hemos propuesto tiene en cuenta la naturaleza tridimensional del problema, proporcionando puntos sobre las líneas de proyección desde los objetos a las cámaras. Aunque no conozcamos la profundidad a la que se encuentra el objeto, al colocar hipótesis sobre las líneas de visión las partículas podrían converger rápidamente hacia la estimación correcta de la posición.

A pesar de que emplear las imágenes actuales para colocar las partículas tiene un coste computacional muy superior a colocarlas de manera ciega, los resultados muestran una mejora importante ante los problemas de convergencia y de movimientos bruscos. En la implementación que hemos realizado, el funcionamiento ha alcanzado una velocidad de 15 iteraciones por segundo con 1.000 partículas y usando dos cámaras con imágenes de 640x480 píxeles de resolución y 32 bits de color en el mismo hardware que en el caso anterior. Con estos resultados vemos que nos mante-

nemos dentro de las restricciones de funcionamiento en tiempo real que impusimos al comenzar el trabajo.

Además, hemos propuesto diferentes métodos para aumentar aún más la velocidad de funcionamiento de nuestra función de abducción. La forma principal para reducir la carga computacional asociada con el proceso de abducción proviene de la posibilidad de no aplicarlo en todos los instantes de tiempo. La abducción es una forma potente de obtener información acerca de qué zonas del espacio pueden resultar interesantes. Si suponemos que existe continuidad espacial entre un instante de tiempo y el siguiente, podemos suponer que los alrededores de las zonas de interés en un instante dado serán también zonas de interés del siguiente instante. De esta forma, podremos reutilizar la información obtenida por el proceso de abducción durante varios instantes de tiempo consecutivos, sin disminuir la capacidad de búsqueda de manera sensible.

Durante los experimentos realizados al sistema de seguimiento de un objeto, con el filtro de condensación, se ha descubierto la existencia de una deriva perniciosa. Ésta afecta a la nube de partículas, alejándola de las cámaras durante los pasos de búsqueda cuando el objeto se ve en una única cámara. Este efecto se achaca a la combinación de la proyección no lineal de las cámaras con un modelo de observación excesivamente sencillo, puesto que las partículas más alejadas tienden a caer con más probabilidad dentro del objeto. De esta forma existe una descompensación entre la parte de delante y de detrás del objeto que empuja la nube hacia atrás [BCM05].

Los resultados de precisión que hemos obtenido en ambos casos, para el algoritmo de condensación y muestreo enfatizado, son satisfactorios. Como hemos mostrado en la sección 4.7, la precisión alcanzada es de unos 6 cm en un entorno de unos 270 litros con objetos situados a un metro de la cámara. Como hemos indicado este error incluye los errores debidos a la calibración de las cámaras. Los resultados son mucho más precisos si los analizamos por cada eje, siendo el eje alineado con la profundidad de la cámara el más afectado por los errores. Esto se debe, como hemos mostrado, a la colocación de las cámaras.

Las diferencias entre ambos algoritmos resultan más llamativas cuando se comparan con el problema del seguimiento. El algoritmo de condensación proporciona estimaciones más suaves a la hora de realizar el seguimiento de un objeto. El mues-

treo enfatizado, al no mantener información de una iteración a la siguiente, presenta mayores temblores en las trayectorias estimadas.

Otra de las diferencias que hemos descrito entre los resultados de los algoritmos es cómo se enfrentan al problema del seguimiento de múltiples objetos. El algoritmo de condensación es incapaz de mantener la información de los dos objetos al mismo tiempo, por lo que no puede usarse para el seguimiento de múltiples objetos. Las partículas tienden a colocarse únicamente sobre uno de los objetos presentes en la escena. El algoritmo, con el fin de optimizar los recursos disponibles, elimina las partículas no prometedoras al pasar de una iteración a la siguiente. En los experimentos hemos comprobado cómo este proceso elimina en el largo plazo todas las partículas excepto las colocadas sobre uno de los objetos.

El filtro de muestreo enfatizado, por otro lado, recoloca todas las partículas en todos los instantes de tiempo. En cada paso las coloca en las zonas prometedoras que proporciona la función de abducción, por lo que es capaz de localizar más de un objeto al mismo tiempo. Sin embargo, a la hora de enfrentarse al problema del seguimiento, los resultados, como hemos comprobado, no resultan satisfactorios. Al igual que sucede para el seguimiento de un único objeto, las trayectorias estimadas contienen mucho ruido y discontinuidades por lo que no resultan útiles para una aplicación de seguimiento en general. Por otro lado, no es capaz de enfrentarse al problema de los "objetos fantasma" que mostramos en la sección 5.2.

Estos resultados se han corroborado en el montaje experimental desarrollado y justifican la necesidad de desarrollar un método capaz de seguir a varios objetos al mismo tiempo. Las limitaciones han sido debidamente documentadas en la sección 4.7.4.

El capítulo 5 recoge la contribución principal de esta tesis. Hemos propuesto una solución para combinar los dos algoritmos anteriores y resolver con ello correctamente el seguimiento de múltiples objetos. La base del sistema consiste en emplear partículas procedentes de los dos algoritmos al mismo tiempo. Las partículas del muestreo enfatizado proporcionan un mínimo de partículas en cada objeto, de tal forma que la evolución del algoritmo de condensación no pierda zonas interesantes en el espacio de estados. Además, la abducción será capaz de localizar nuevos objetos

según aparecen en la escena. Por otro lado, la naturaleza secuencial del algoritmo de condensación buscará localmente las mejores estimaciones posibles para cada objeto. Además, proporciona suavidad al seguimiento, compensando las deficiencias que aparecen en el muestreo enfatizado no secuencial.

La combinación de partículas se ha resuelto de manera teórica dentro del marco de Monte Carlo. En concreto, ajustando los pesos de cada grupo de partículas de manera diferente. De esta forma se puede tener en cuenta en cada caso la utilización de distribuciones de propuesta diferentes para extraer las partículas. Para permitir que se mezclen las partículas, es necesario remuestrear en todos los instantes de tiempo, como hacía el algoritmo de condensación. Al hacer esto conseguimos eliminar la dependencia de los pesos de instantes anteriores, simplificando el cálculo de los nuevos pesos, sea cuál sea la procedencia de la partícula.

Con este nuevo método que hemos propuesto se obtienen unos resultados de precisión similares a los que teníamos para el seguimiento de un único objeto. Esto era de esperar dado que estamos combinando dos soluciones con prestaciones similares y, como hemos mostrado en el capítulo 5, los errores de estimación deberían ser, como mínimo, similares al peor de los dos métodos por separado, lo que nos da un umbral mínimo de funcionamiento. La convergencia de las estimaciones a los valores correctos es, en este caso, mucho mejor que la del algoritmo de condensación, haciendo uso de las ventajas que proporciona la función de abducción.

A nuestro juicio estos resultados confirman que nuestra propuesta proporciona una solución factible al problema del seguimiento de múltiples objetos al mismo tiempo. La combinación de partículas de los dos métodos es una buena forma de conseguir mantener de manera estable la multimodalidad de la distribución de probabilidad estimada, principal limitación del algoritmo de condensación. Las ventajas asociadas a este algoritmo también están presentes en la propuesta realizada, tales como la velocidad o la capacidad de seguimiento. El reparto de las partículas entre los diferentes objetos permite mantener sistemas similares al de condensación por cada uno de ellos de manera implícita, sin necesidad de indicarlo explícitamente.

Sin embargo, hemos visto cómo el reparto de partículas no es uniforme entre los objetos. Un reparto uniforme sería deseable dado que un mayor número de partículas permite una estimación más precisa de la posición del objeto. La diferencia

en el reparto viene dada tanto por el proceso de remuestreo como por la función de abducción. El remuestreo, como hemos mostrado, tiende a ser injusto en el reparto de partículas pero también lo es nuestro modelo de observación. Aunque la función de abducción está diseñada para repartir uniformemente las partículas entre los objetos detectados en las imágenes, la asignación de pesos no lo es. Los objetos que tengan zonas de incertidumbre mayores en el espacio de estados, esto es, aquellos que tengan asociadas más soluciones válidas según el modelo de observación, obtendrán mayor número de partículas con pesos altos, por lo que predominarán en el remuestreo. En nuestro caso serán los objetos mayores los que tendrán mayores áreas y, a mismo tamaño, los que estén situados más lejos en profundidad y más cerca de los ejes ópticos de las cámaras ocuparán mayores zonas en el espacio de estados. Éste es el motivo por el que el parámetro α (ver expresión 5.1), encargado de modular el balance entre la parte secuencial y no secuencial del algoritmo, no tiene una influencia notable en estos cambios.

Hemos comprobado, tanto en el capítulo 4 como en el 5, que el reparto de partículas resulta suficiente para estimar la posición de todos los objetos de manera satisfactoria. Al ir aumentando el número de objetos resulta más relevante que el reparto de partículas sea equitativo entre los objetos existentes, con el fin de que el número de partículas en el objeto más perjudicado no se coloque por debajo del umbral de utilidad. Aumentar el número total de partículas puede ser una solución parcial al problema. Sin embargo, como hemos discutido, el hecho de que el reparto no sea equitativo puede hacer que sea necesario añadir muchas partículas para asegurar un número mínimo en cada objeto de la escena. Hemos propuesto una solución para paliar este problema, consistente en modificar el modelo de observación con el fin de conseguir igualar las áreas de incertidumbre entre todos los objetos presentes en la escena.

Otra ventaja que debemos señalar del modelo que hemos desarrollado es que la aparición y desaparición de objetos no es un hecho aislado o especial que debe considerarse de manera particular. El sistema es capaz de detectar los nuevos objetos igual que detecta los objetos presentes en la escena, por medio de la función de abducción. Tan pronto como aparece un nuevo objeto, la abducción colocará unas pocas partículas en las regiones prometedoras para él, permitiendo localizarlos rápidamente. No se

trata de un procedimiento especial dado que la abducción está continuamente buscando zonas del espacio donde colocar nuevas partículas, ya sea para objetos existentes o nuevos.

La función de densidad de probabilidad que conseguimos no estima por sí sola la posición de los múltiples objetos. Por ese motivo hemos propuesto añadir un segundo nivel al algoritmo que trabaje sobre la distribución de probabilidad multimodal estimada. Dicha distribución nos indica dónde parece probable que se encuentre un objeto. Esta información se proporciona "en crudo", como un conjunto de partículas con pesos que indican la probabilidad en un punto determinado del espacio de estados. El segundo nivel que hemos propuesto será el encargado de extraer desde las partículas la información sobre el número de objetos presente en la escena o la posición exacta de cada uno de ellos. Dicho de otra forma, será el encargado de convertir las nubes de partículas con pesos altos, que se encuentran alrededor de cada objeto, en grupos de partículas. Cada uno de estos grupos estará asociado con un objeto, permitiendo estimar su posición correcta. Este algoritmo resulta necesario para completar el sistema de seguimiento multiobjeto, pero no es nuestro objetivo ahondar en el funcionamiento de los sistemas de segmentación, ampliamente conocidos. Por este motivo hemos empleado un algoritmo basado en EM, una solución clásica al problema de la segmentación, añadiendo unos métodos para cambiar el número de grupos en función de las características de los ya creados y de las partículas no asociadas a ningún grupo. Así mismo hemos hecho hincapié en sus prestaciones, fijándonos sobre todo en su rapidez de funcionamiento. Aunque el algoritmo presenta varias limitaciones, nos permite ver cuál es el potencial del algoritmo de estimación probabilística, foco de este trabajo.

La limitación principal que hemos encontrado al emplear este algoritmo de segmentación está relacionada con las simplificaciones del modelo de observación. Las zonas de incertidumbre asociadas a los grupos de partículas sobre cada objeto no serán iguales, sino que dependerán del tamaño de la zona de incertidumbre asociada a dicho objeto. Al aplicar un algoritmo de segmentación que no tiene esto en cuenta resulta difícil llegar a un compromiso entre la división de grupos y la fusión de los mismos.

Durante los experimentos realizados hemos comprobado el correcto funciona-

miento del sistema combinado, tanto para el caso de un objeto como para el caso de varios objetos. Las pruebas se han centrado en estudiar tanto la precisión como el reparto de partículas para varios objetos. En los experimentos hemos probado de dos a cinco objetos al mismo tiempo y en diferentes montajes, usando dos o cuatro cámaras. La velocidad de la aplicación alcanza las 30 iteraciones por segundo en un caso medio con 1.000 partículas y dos imágenes de 640x480 píxeles de resolución y 32 bits de profundidad de color.

El capítulo 6 hemos descrito las principales características de la implementación que hemos desarrollado de los algoritmos propuestos. Nuestra implementación recibe el nombre de METS y la hemos realizado en C++ sobre el sistema operativo GNU/Linux. En este capítulo se hace un balance sobre las principales características de este software, con énfasis en la eficiencia o la implementación de determinadas partes del algoritmo. Nuestro objetivo era conseguir una implementación que funcionara en tiempo real por lo que ha sido necesario prestar una especial atención a los módulos que más coste computacional requerían, como el filtro de color o la función de abducción.

Así mismo hemos añadido funciones útiles para simplificar el proceso de realizar los experimentos, como son la grabación de vídeos y la reproducción de los mismos. Con estas funciones podemos repetir exactamente el mismo experimento empleando diferentes parámetros de los algoritmos usados, como puede ser el número de partículas o la anchura de los modelos de movimiento.

Sin estas herramientas no hubiera sido posible abordar una tesis como ésta de la forma que se ha hecho. Gracias al tiempo empleado en mejorar y depurar las aplicaciones hemos sido capaces de validar los algoritmos propuestos, así como comprender las dinámicas subyacentes en ellos.

Así mismo hemos probado algunos de los algoritmos propuestos desarrollando aplicaciones específicas. A diferencia de METS estas aplicaciones están centradas en un uso particular, no en el estudio de los algoritmos. Dichas aplicaciones son la de *Watcher*, para la videovigilancia de una habitación, y *ElderCare*, centrada en el cuidado de personas mayores, capaz de disparar una alarma cuando alguien cae al suelo. Estas dos aplicaciones emplean montajes diferentes a los presentados en los

experimentos, extendiendo así los escenarios donde se han probado los algoritmos. En particular tanto *Watcher* como *ElderCare* funcionan con 4 cámaras al mismo tiempo e imágenes de 320x240 píxeles de resolución.

7.2. Publicaciones relacionadas

Parte del trabajo que aquí se muestra ha sido publicado en artículos de revistas internacionales y presentado en diferentes conferencias:

- *Automatic detection of high risk situations for elder persons care*. J. M. Cañas, A. Pineda, P. Barrera. Aceptado en el II Congreso Internacional de Domótica, Robótica y Teleasistencia para Todos, DRT4ALL-2007.
  Este artículo se centra en la aplicación *ElderCare* y sus aplicaciones para el cuidado de personas mayores.

- *Multicamera 3D tracking using particle filter.* P. Barrera, J.M. Cañas, V. Matellán, F. Martín. IADAT Journal of Advanced Technology on Imaging and Graphics, Volumen 1, Número 1, pág. 13-15, Septiembre de 2005, ISSN 1885-6411. Aceptado previamente en las actas de la Int. Conf. on Multimedia, Image processing and Computer Vision, celebrado del 30 de Marzo al 1 de Abril de 2005 en Madrid. ISBN: 84-933971-5-6, pág. 193-197.
  En este artículo se presenta el sistema de combinación probabilístico para varias fuentes de información. En particular emplea una combinación de color y un modelo de movimiento simple para seguir a un objeto en el entorno de una habitación.

- *Visual object tracking in 3D with color based particle filter*. P. Barrera, J.M. Cañas, V. Matellán. Int. Journal of Information Technology, Vol 2, Número 1, pág. 61-65, 2005. ISSN: 1305-2403. Una versión más corta aparece en las actas de Int. Conf. on Pattern Recognition and Computer Vision, celebrado del 25 al 27 de Febrero de 2005, Estambul (Turquía). ISBN: 975-98458-3-0, pág. 200-203.
  Este artículo presenta la adaptación de un filtro de partículas para el caso del se-

guimiento de un único objeto empleando solamente información de color. Otra de las contribuciones es la explicación de la deriva perniciosa al recorrer el espacio de estados con el algoritmo de condensación y un modelo de movimiento genérico.

- *Seguimiento tridimensional usando dos cámaras.* P. Barrera, J.M. Cañas. Informe técnico TR-GSYC-2004-7, Universidad Rey Juan Carlos, Octubre 2004.
  El informe técnico se centra en las técnicas de triangulación y los filtros de partículas. Plantea las bases teóricas de estas dos técnicas y resuelve el problema estático de la localización de un único objeto.

7.3. Líneas futuras de trabajo

Como líneas de investigación y desarrollo futuras podemos recopilar algunas de las que hemos mencionado a lo largo de los capítulos anteriores:

- En primer lugar sería interesante probar un modelo de observación más potente y combinarlo con el sistema de seguimiento multiobjeto que hemos desarrollado. Esto permitiría obtener mejores resultados, sobre todo si el nuevo modelo aborda el problema de las zonas de incertidumbre en el espacio de estados, reduciendo, al mismo tiempo, la deriva perniciosa. Las posibilidades van desde cambios menores, como puede ser añadir el tamaño del objeto o más información acerca de su forma, a incluir características más complejas. Podemos usar técnicas como las empleadas con las características tipo SIFT [SL04], el aprendizaje incremental de las características de los objetos [LRLY05] o el uso de autoespacios para representar las características visuales [BJ96].

- También sería interesante realizar una implementación capaz de trabajar con objetos articulados con el fin de incrementar la dimensión del espacio de estados. La principal limitación que esperamos encontrar con este problema es la función de propuesta. En el caso de que el objeto a seguir sea complejo, la función de propuesta no nos proporcionará información en todas las dimensiones del problema. Para el seguimiento en 3D la función de propuesta nos daba

información en dos dimensiones limitando la búsqueda a una única dimensión. En el caso de tener más dimensiones la disminución puede ser menos significativa, lo que incrementaría la necesidad de realizar una búsqueda más eficiente que la búsqueda ciega directa. Sin embargo, una de las ventajas presentes es que la dimensión del problema no depende del número de objetos a seguir, lo que nos permite enfocar los recursos disponibles para seguir varios objetos en un espacio de dimensión menor que el de otras aproximaciones.

- Podría añadirse un nuevo sistema de segmentación menos simple que permita tener en cuenta las diferencias de tamaño de los objetos a la hora de gestionar la aparición de nuevos objetos. El actual sistema de segmentación pone una limitación a la hora de obtener estimaciones estables. Con un algoritmo de segmentación más elaborado sería posible conseguir que la extracción de información desde la distribución de probabilidad multimodal estimada sea más estable.

- El sistema funciona correctamente cuando un número mínimo de partículas se colocan sobre cada objeto. Podría estudiarse con mayor profundidad cuál es este número mínimo para el correcto funcionamiento del sistema, con el fin de seleccionar el número total de partículas en función de este valor, intentando optimizar al máximo los recursos disponibles.

- Una posible vía de estudio es ver cómo cambia el funcionamiento del sistema cuando añadimos más cámaras pero no todas apuntan en la misma dirección. Actualmente trabajamos sobre una escena cubierta por varias cámaras, entre dos y cuatro según el montaje. Sin embargo, todas ellas apuntan a la misma escena. Al incluir cámaras que apunten a áreas diferentes podríamos aumentar el área de seguimiento. Sería posible incluso realizar un seguimiento en varias habitaciones al mismo tiempo. En este mismo contexto se puede estudiar a fondo la influencia de las oclusiones parciales o totales de los objetos en la escena, siguiendo las líneas propuestas en [HE05].

- Las aplicaciones concretas que hemos desarrollado, *Watcher* y *ElderCare* no empleaban todo el potencial de los algoritmos aquí mostrados, centrándose úni-

camente en el seguimiento de un único objeto. Es posible añadir la funcionalidad descrita en el capítulo 5 a estas aplicaciones, lo que permitiría comprobar el funcionamiento para aplicaciones reales de las técnicas propuestas en este tesis.

- Por último, podemos señalar como posible línea futura el uso de los algoritmos propuestos frente a problemas que no empleen la visión como fuente principal de información. Las técnicas propuestas de combinación de algoritmos de seguimiento y de combinación de estímulos son técnicas generales, capaces de enfrentarse a otros tipos de problemas. Sin embargo, sería necesario estudiar las particularidades de cada uno de estos problemas con el fin de poder adaptar los modelos usados a cada uno de ellos. El concepto de función de abducción también tendría cabida para otros tipos de aplicaciones, aunque con un nuevo diseño que permitiera extraer la información de las fuentes disponibles.

Aparte de estudiar estas nuevas cuestiones, entre el trabajo futuro previsto está la escritura de varios artículos centrados en explicar a fondo la propuesta de combinación de distribuciones en el filtro de partículas para conseguir una estimación estable de la función de probabilidad objetivo multimodal.

APÉNDICE A

Apéndices

A.1. Relaciones geométricas de proyección y retroproyección

A continuación resumiremos el funcionamiento del proceso geométrico de proyección y retroproyección. Las expresiones aquí mostradas se emplean en el modelo de observación del filtro y en el proceso de abducción. En el primer caso se emplean las relaciones geométricas entre las posiciones en 3D de las partículas o de los objetos reales y sus proyecciones en cada una de las imágenes. En el segundo caso sirven para construir las líneas de proyección en 3D desde los puntos en cada una de las imágenes. Puede encontrarse una introducción más profunda en cualquier libro sobre geometría proyectiva o geometría para visión por computador, como por ejemplo [HZ03].

Todas estas relaciones están definidas por el tipo de cámara, más concretamente, por la forma que tiene ésta de captar la realidad. Para plasmar este proceso se suele emplear el modelo de cámara *pin-hole*, que es el que nosotros utilizamos. En la sección 1.3 presentamos una breve introducción a los parámetros más significativos de dicho modelo. Aquí nos centraremos en las expresiones de proyección y en cómo

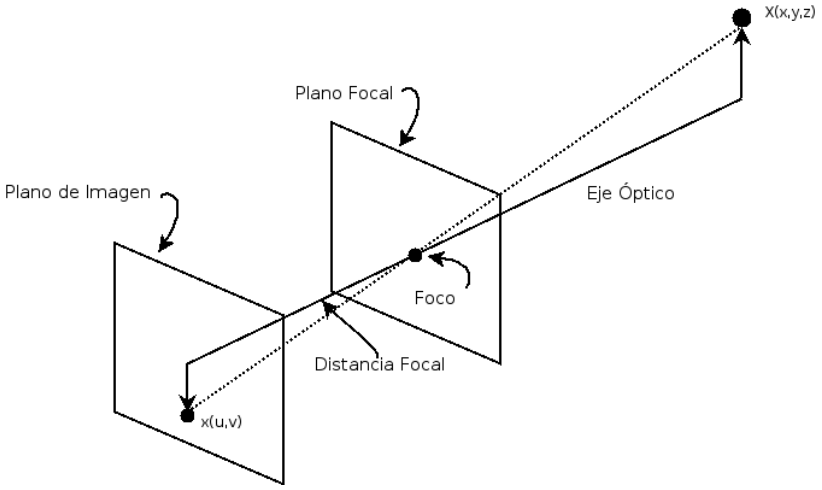

Figura A.1: Modelo *Pin-Hole* de cámara.

relacionarlas con los parámetros del modelo.

El modelo *pin-hole* (ver figura A.1) asume que un punto tridimensional $\boldsymbol{X}(x, y, z)$ se proyecta en el plano de imagen pasando a través de un único punto llamado *Centro óptico* o *Foco*. La recta que une el punto $\boldsymbol{X}$ y el centro óptico se llama *línea de proyección* e intersecta al *plano imagen* justo en el píxel $\boldsymbol{x}(u, v)$ que es la proyección de $\boldsymbol{X}(x, y, z)$. El centro óptico está situado a la *distancia focal* del plano imagen. Este modelo lo completan el *eje óptico*, que es una línea perpendicular al plano de imagen que atraviesa el centro óptico, y el *plano focal*, que es el plano perpendicular al eje óptico cuyos puntos no se proyectan en el plano de la imagen, incluyendo al foco.

La forma en la que podemos expresar este proceso es a través de una serie de parámetros que definen el comportamiento de la cámara en cuanto a la proyección se refiere. Dichos parámetros se pueden dividir en dos grupos, los intrínsecos y los extrínsecos. Los primeros son los parámetros internos de la cámara, como su distancia focal y el tamaño del píxel. Los parámetros extrínsecos, por otro lado, indican información externa a la cámara, como su posición en el mundo o su orientación.

La expresión A.2 relaciona la posición de un punto 3D con su proyección para una cámara situada en el origen de coordenadas:

$$\boldsymbol{x} = P\boldsymbol{X} \tag{A.1}$$

$$\begin{pmatrix} f_x u' \\ f_y v' \\ Z \end{pmatrix} = \begin{bmatrix} f_x & 0 & u_0 & 0 \\ 0 & f_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{A.2}$$

La relación emplea coordenadas homogéneas para simplificar el cálculo. Los parámetros usados se corresponden con los parámetros intrínsecos. Estos son la distancia focal medida en píxeles en los dos ejes (f_x, f_y) y la posición del centro de la imagen (u_0, v_0). Para obtener las coordenadas exactas de la proyección basta con aplicar que cualquier punto en coordenadas homogéneas cumple que $\boldsymbol{x} = \alpha\boldsymbol{x}$ para $\alpha \neq 0$. De esta forma obtenemos las coordenadas en la imagen como

$$u = \frac{f_x u'}{Z} \tag{A.3}$$

$$v = \frac{f_y v'}{Z} \tag{A.4}$$

De la expresión A.2 podemos aislar los parámetros intrínsecos de la cámara en la siguiente matriz, que usaremos más adelante:

$$K = \begin{pmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{A.5}$$

Las expresiones que hemos mostrado se encargan de realizar la proyección cuando la cámara está colocada en el origen de coordenadas. En el caso de que esté en un punto cualquiera del espacio será necesario aplicar una traslación y una rotación antes de proyectar usando las mismas expresiones. De esta forma la matriz de proyección queda como sigue:

$$\begin{pmatrix} f_x u' \\ f_y v' \\ Z \end{pmatrix} = \begin{bmatrix} f_x & 0 & u_0 & 0 \\ 0 & f_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & -RC \\ 0 & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{A.6}$$

donde R es la matriz de rotación $3x3$ y C es la posición del centro de la cámara en coordenadas del mundo. Podemos expresar la matriz de proyección de una manera más compacta de la siguiente forma:

$$P = KR[I| - C] \tag{A.7}$$

donde I es la matriz identidad de $3x3$. La matriz de proyección P tendrá una dimensión de $3x4$, relacionando coordenadas homogéneas del espacio real (cuatro dimensiones) con los de puntos proyectados (tres dimensiones). El número de grados de libertad, para una cámara real como las que estamos considerando, es de 10. Cuatro corresponden a la matriz K, tres a la matriz R y los últimos tres a la posición del centro de la cámara.

Una vez que hemos presentado todas las expresiones de proyección, podemos pasar a abordar el problema inverso, la retroproyección. En este caso queremos relacionar puntos del plano de imagen con puntos tridimensionales. El proceso de proyección pierde la información de profundidad del punto por lo que no es posible recuperarla. En cambio lo que podemos obtener es la línea de todos los posibles puntos que proyectan en un determinado punto del plano de imagen.

Para obtener dicha línea de proyección será suficiente con que localicemos dos puntos en 3D que proyecten sobre $\boldsymbol{x}$. Resulta directa la elección del foco o centro de la cámara como el primero de ellos. Como ya hemos comentado todas las líneas de proyección pasan por el foco de la cámara, así que este punto formará parte de todas las líneas de proyección. El centro C también se puede obtener de la matriz de proyección de la siguiente forma. Descompongamos primero la matriz P de la siguiente forma, $P = [M|P_4]$, siendo M la submatriz $3x3$ con las tres primeras columnas y P_4 el vector formado por la última columna de P. El centro de la cámara se puede

despejar de la matriz P de la siguiente forma:

$$C = -M^{-1}P_4 \tag{A.8}$$

El segundo punto puede ser cualquier otro que cumpla que $P\boldsymbol{X} = \boldsymbol{x}$. En particular podemos coger $\boldsymbol{X}_0 = M^{-1}(\boldsymbol{x} - P_4)$, de tal forma que

$$P\boldsymbol{X}_0 = [M|P_4] \begin{bmatrix} M^{-1}(\boldsymbol{x} - P_4) \\ 1 \end{bmatrix} = MM^{-1}(\boldsymbol{x} - P_4) + P_4 = \boldsymbol{x} \tag{A.9}$$

por lo que $\boldsymbol{X}_0$ es un punto válido.

A.2. Bibliotecas auxiliares

Para la realización METS, la implementación de los algoritmos descritos en esta tesis, y el resto de las herramientas software comentadas en el capítulo 6 se han empleado varias herramientas y bibliotecas disponibles en Internet. A continuación comentaremos brevemente algunas de las más relevantes.

VW

Esta es una biblioteca para la creación de aplicaciones de visión por computador desarrollada principalmente por el grupo de Visión Activa de la Universidad de Oxford[1]. Su tamaño ronda las 80.000 líneas de código, con capacidades que van desde la conversión de formatos de imágenes hasta el uso de primitivas geométricas.

El paquete de software incluye VNL (*Vision Numeric Library*), una biblioteca de álgebra genérica con especial énfasis en la eficiencia. VNL es parte de VXL[2] una biblioteca de visión por computador muy completa que incluye infinidad de componentes. El objetivo de VXL es proporcionar un marco de desarrollo para cualquier tipo de aplicación de visión por computador. La aproximación que sigue VW es dife-

[1] http://www.robots.ox.ac.uk/ActiveVision/
[2] http://vxl.sourceforge.net/

rente. En vez de intentar abarcar el mayor número de aplicaciones se centra en hacer sólo un número limitado de tareas pero manteniendo una buena eficiencia. Su orientación está sobre todo centrada en el desarrollo de aplicaciones en tiempo real. VW está desarrollada íntegramente en C++ (a diferencia de VXL que está implementada en C). Esta biblioteca hace uso extenso de plantillas y biblioteca estándar de plantillas (STL) con el fin de hacer el código lo más reusable posible.

Junto a VW se incluyen en la distribución varias bibliotecas auxiliares encargadas, entre otras cosas, de la conexión con OpenGL y del soporte para cámaras Firewire.

GTKMM

GTK+ es una biblioteca para crear interfaces gráficas en varios sistemas operativos. Permite un desarrollo muy rápido y tiene recubrimientos en infinidad de lenguajes. GTKMM[3] es uno de esos recubrimientos, en este caso en C++. Es el que hemos empleado para su integración en METS dado que este programa también está desarrollado en C++.

Glade es una herramienta que acompaña a GTK+ y que es capaz de diseñar las interfaces gráficas de manera visual, almacenando el diseño en un archivo XML. Dicho archivo se carga posteriormente desde otro programa que es capaz de enlazarse con ese diseño en tiempo de ejecución. De esta forma se puede independizar bastante el desarrollo de la interfaz gráfica del programa con su lógica interna.

VWGTK2

VW tiene una serie de bibliotecas auxiliares para facilitar la integración con bibliotecas de interfaces gráficos. Sin embargo, la versión que emplea de GTK+ se corresponde con la versión 1.2, actualmente casi en completo desuso. Por este motivo, para emplear VW en el marco de esta tesis, hemos desarrollado la biblioteca VWGTK2, encargada de realizar la integración entre VW y una versión reciente de GTK+ (concretamente la versión 2.8).

[3] http://gtkmm.org/

Entre las funcionalidades de VWGTK2 se incluye la integración con OpenGL, el soporte de Glade, la capacidad de cargar imágenes, widgets de control específicos para las cámaras y secuencias de imágenes, entre otras.

VWGTK2 está compuesto por unas 2.500 líneas de código C++ y forma el núcleo de la interfaz gráfica de METS, distribuyéndose junto a él.

ARToolkit

ARToolkit[4] es una biblioteca diseñada para el desarrollo de aplicaciones basadas en *realidad aumentada*. La realidad aumentada consiste en mezclar imágenes sintéticas con otras tomadas de la realidad. Para realizar esta tarea de manera satisfactoria es necesario ser capaces de estimar los parámetros de calibración de las cámaras de manera robusta y en tiempo real. La biblioteca ha sido desarrollada por la Universidad de Osaka (Japón) con ayuda del laboratorio HITLab (*Human Interface Technology*) de la Universidad de Washington (Estados Unidos) y el HITLab de la Universidad de Canterbury (Nueve Zelanda).

ARToolkit proporciona métodos para realizar la calibración de cámaras basándose en el uso de patrones. El patrón es una imagen conocida de alto contraste que se emplea como marcador en el mundo real, con el fin de extraer los parámetros de calibración extrínseca de la cámara (posición y orientación) en relación a la posición del patrón. Así mismo ARToolkit incluye funciones para usar diferentes tipos de cámaras en plataformas Linux, como son Video4Linux o conexiones Firewire.

En nuestro caso hemos empleado ARToolkit para calibrar las cámaras en cada experimento. La naturaleza de nuestros experimentos implicaban el cambio de la posición de la cámara con relativa frecuencia por lo que la facilidad de uso, rapidez y precisión de ARToolkit han sido una gran ayuda.

Hemos empleado una versión ligeramente modificada de ARToolkit para que funcionara en nuestros sistemas. En la figura A.2 podemos ver una captura del programa de autolocalización de cámaras que hemos emplado (se distribuye junto a ARToolkit).

[4] http://www.hitl.washington.edu/artoolkit/

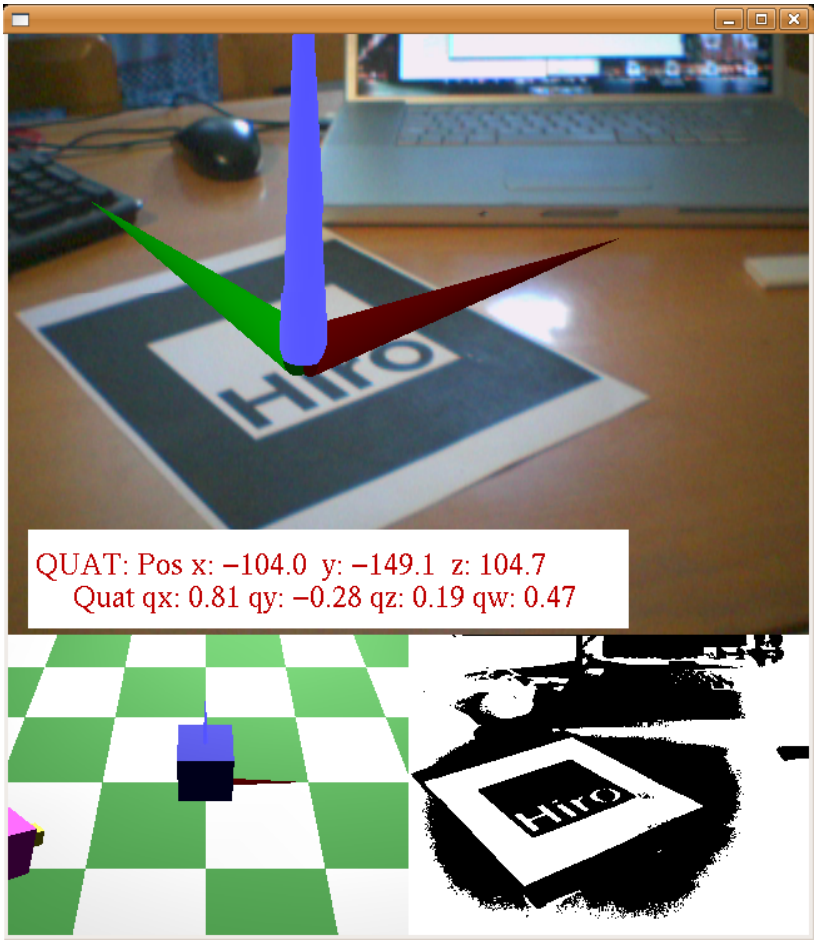

Figura A.2: Aplicación de autolocalización de cámaras hecha con ARToolkit.

OpenGL

Las bibliotecas de gráficos OpenGL son el estándar de facto de la industria en cuanto a gráficos 3D se refiere. Sencillas de usar y soportadas por la mayor parte de las tarjetas con aceleración gráfica, son capaces de reducir considerablemente el coste de cómputo en lo que a visualización se refiere relegando muchas operaciones directamente en el hardware específico. Como se ha comentado anteriormente, tanto VW como VWGTK2 hacen un uso intensivo de OpenGL.

Para nuestros desarrollos OpenGL presenta múltiples ventajas de las que podemos señalar principalmente dos. En primer lugar hace que nos despreocupamos de la

mayor parte del desarrollo gráfico de nuestras aplicaciones. En segundo lugar hace que el coste computacional dedicado a la visualización baje considerablemente, permitiendo que los recursos del ordenador se centren principalmente en los cálculos de nuestros diferentes algoritmos.

VTK

VTK[5] es una biblioteca de visualización científica desarrollada sobre OpenGL. Su objetivo principal es proporcionar un completo conjunto de herramientas para visualizar todo tipo de datos, en dos, tres o más dimensiones.

Aunque VTK está desarrollado íntegramente en C++ posee multitud de recubrimientos en muchos otros lenguajes de programación. En particular nosotros hemos usado PyVTK[6], el recubrimiento para el lenguaje de script Python.

En nuestro caso empleamos VTK para realizar el visualizador *ParticlePlayer*, descrito en la sección 6.5. La ventaja con respecto a otras bibliotecas gráficas es su alto nivel, que permite trabajar a nivel de dato u objeto en vez de punto o píxel.

[5] http://vtk.org/
[6] http://cens.ioc.ee/projects/pyvtk/

BIBLIOGRAFÍA

[AFDJ03] CHRISTOPHE ANDRIEU, NANDO DE FREITAS, ARNAULD DOUCET Y MICHAEL JORDAN. An introduction to mcmc for machine learning. *Machine Learning* **50**, 5 – 43 (Feb. 2003).

[AMGC02] SANJEEV ARULAMPALAM, SIMON MASKELL, NEIL GORDON Y TIM CLAPP. A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing* **50**(2), 174 – 188 (Febrero 2002).

[BCM05] PABLO BARRERA, JOSÉ MARIA CAÑAS Y VICENTE MATELLÁN. Visual object tracking in 3d with color based particle filter. *International Journal of Information Technology* **2**(1), 61 – 65 (2005).

[BDH03] MIODRAG BOLIC, PETAR M. DJURIC Y SANGJIN HONG. New resampling algorithms for particle filters. En "Proceedings of IEEE International Conference on Acoustics, Speech, and Signal", tomo 2, páginas 589 – 592 (Abril 2003).

[Ber99] NICLAS BERGMAN. "Recursive Bayesian estimation: Navigation and tracking applications". Tesis Doctoral, University of Linköping (Suecia) (1999).

[BH65] GEOFFREY H. BALL Y DAVID J. HALL. Isodata, a novel method of data analysis and pattern classification. Informe técnico AD-699616, Standord Research Institute (1965).

[Bis95] CHRISTOPHER M. BISHOP. "Neural Network for Pattern Recognition". Oxford University Press (1995).

[BJ96] MICHAEL J. BLACK Y ALLAN D. JEPSON. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. En "European Conference on Computer Vision", páginas 329 – 342 (1996).

[BMG07] MOMOTAZ BEGUM, GEORGE K.I. MANN Y RAYMOND G. GOSINE. Integrated fuzzy logic and genetic algorithmic approach for simultaneous localization and mapping of mobile robots. *Journal on Applied Soft Computing* (2007).

[BT99] STAN BIRCHFIELD Y CARLO TOMASI. Depth discontinuities by pixel-to-pixel stereo. *International Journal of Computer Vision* **35**(3), 269 – 293 (1999).

[CAK02] ROBERT T. COLLINS, OMEAD AMIDI Y TAKEO KANADE. An active camera system for acquiring multi-view video. En "Proceedings of the 2002 International Conference on Image Processing (ICIP '02)", páginas 517 – 520 (Septiembre 2002).

[CD02] DAN CRISAN Y ARNAULD DOUCET. A survey of convergence results on particle filtering methods for practitioners. *IEEE Transactions on Signal Processing* **50**, 736 – 746 (Marzo 2002).

[CdFL06] YIZHENG CAI, NANDO DE FREITAS, Y JAMES J. LITTLE. Robust visual tracking for multiple targets. *Computer Vision ECCV 2006* **3954**, 107 – 118 (2006).

[CM99] W. BRUCE CULBERTSON Y THOMAS MALZBENDER. Generalized voxel coloring. En "Vision Algorithms: Theory and Practice" (1999).

[Col95] ROBERT T. COLLINS. A space-sweep approach to true multi-image matching. Informe técnico UM-CS-1995-101, UM-CS (1995).

[Cre02] JOSÉ VICENTE CRESPO. "Métodos visuales de reconstrucción 3D en robots móviles". Tesis Doctoral, Universidad Politécnica de Madrid (2002).

[Cri01] DAN CRISAN. "Sequential Monte Carlo Methods in Practice", capítulo Particle filters – A theoritical perspective, páginas 17 – 41. Springer (2001).

[dAHH99] LOURDES DE AGAPITO, RICHARD I. HARTLEY Y ERIC HAYMAN. Linear calibration of a rotating and zooming camera. En "Proceedings on Computer Vision and Pattern Recognition", páginas 15 – 20 (1999).

[Dav03] ANDREW J. DAVISON. Real-time simultaneous localisation and mapping with a single camera. En "Proceedings of the 9th IEEE International Conference on Computer Vision", tomo 2, páginas 1403 – 1410 (2003).

[dBdA06] ALESSIO DEL BUE Y LOURDES DE AGAPITO. Non-rigid stereo factorization. En "Proceedings of International Journal of Computer Vision (IJCV)" (2006).

[dBdA07] ALESSIO DEL BUE Y LOURDES DE AGAPITO. "Scene Reconstruction, Pose Estimation and Tracking", capítulo Non-rigid Stereo-Motion. I-Tech Education and Publishing, Viena, Austria (2007).

[DBR00] JONATHAN DEUTSCHER, ANDREW BLAKE Y IAN REID. Articulated body motion capture by annealed particle filtering. En "Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)" (2000).

[DCM05] RANDAL DOUC, OLIVIER CAPPÉ Y ERIC MOULINES. Comparison of resampling schemes for particle filtering. En "Proceedings of the 4th

International Symposium on Image and Signal Processing and Analysis", páginas 64 – 69 (Septiembre 2005).

[DCSF06] SIMON DENMAN, VINOD CHANDRAN, SRIDHA SRIDHARAN Y CLINTON FOOKES. A multi-class tracker using a scalable condensation filter. En "Proceedings of the IEEE International Conference on Video and Signal Based Surveillance", página 25 (2006).

[DDD04] FADI DORNAIKA, FRANCK DAVOINE Y MO DANG. 3d head tracking by particle filters. http://citeseer.ist.psu.edu/698811.html (2004).

[DdFG01] ARNAULD DOUCET, NANDO DE FREITAS Y NEIL GORDON, editores. "Sequential Monte Carlo Methods in Practice". Ed. Springer (2001).

[Dou98] ARNAULD DOUCET. On sequential simulation-based methods for bayesian filtering. Informe técnico, Cambridge University Engineering Department (1998).

[DT01] SHILOH L. DOCKSTADER Y A. MURAT TEKALP. Multiple camera fusion for multi-object tracking. En "Proceedings of the IEEE Workshop on Multi-Object Tracking (WOMOT'01)", página 95, Washington, DC, USA (2001). IEEE Computer Society.

[Fua97] PASCAL FUA. From multiple stereo views to multiple 3-d surfaces. *International Journal of Computer Vision* **24**(1), 19 – 35 (1997).

[GGB+02] FREDRIK GUSTAFSSON, FREDRIK GUNNARSSON, NICLAS BERGMAN, URBAN FORSSELL, JONAS JANSSON, RICKARD KARLSSON Y PER-JOHAN NORDLUND. Particle filters for positioning, navigation, and tracking. *IEEE Transactions on Signal Processing* **50**, 425 – 435 (2002).

[Hay01] SIMON HAYKIN. "Adaptive Filter Theory". Prentice Hall, 4th edition edición (2001).

[HdARH99] RICHARD I. HARTLEY, LOURDES DE AGAPITO, IAN D. REID Y ERIC HAYMAN. Camera calibration and the search for infinity. En "International Conference on Computer Vision", páginas 510 – 517 (1999).

[HE05] YAN HUANG Y IRFAN ESSA. Tracking multiple object through occlusions. En "Proceedings in Computer Vision and Pattern Recongnition", tomo 2, páginas 1051 – 1058 (2005).

[Hem03] ELSAYED E. HEMAYED. A survey of camera self-calibration. *IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS'03)* **00**, 351 (2003).

[Her02] SHAWN MICHAEL HERMAN. "A Particle Filtering Approach to Joint Passive Radar Tracking and Target Classification". Tesis Doctoral, University of Illinois (2002).

[HS97] RICHARD I. HARTLEY Y PETER STURM. Triangulation. *Computer Vision and Image Understanding: CVIU* **68**(2), 146 – 157 (1997).

[HSG06] JEROEN HOL, THOMAS SCHÖN Y FREDRIK GUSTAFSSON. On resampling algorithms for particle filters. En "Nonlinear Statistical Signal Processing Workshop, Cambridge" (Septiembre 2006).

[HZ03] RICHARD HARTLEY Y ANDREW ZISSERMAN. "Multiple View Geometry". Cambridge University Press (2003).

[IB98a] MICHAEL ISARD Y ANDREW BLAKE. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision* **29(1)**, 5 – 28 (1998).

[IB98b] MICHAEL ISARD Y ANDREW BLAKE. Icondensation: Unifying low-level and high-level tracking in a stochastic framework. *Lecture Notes in Computer Science* **1406**, 893 – 908 (1998).

[IB98c] MICHAEL ISARD Y ANDREW BLAKE. A mixed-state condensation tracker with automatic model-switching. En "Sixth International Conference on Computer Vision", páginas 107 – 112 (1998).

[Jen99] CULLEN JENNINGS. Robust finger tracking with multiple cameras. En "Proceedings of RATFG99", páginas 152 – 160 (1999).

[Kas97] KEITH D. KASTELLA. Joint multitarget probabilities for detection and tracking. En "SPIE Proceedings, Acquisition, Tracking and Pointing" (Abril 1997).

[KBD05] ZIA KHAN, TUCKER BALCH Y FRANK DELLAERT. Mcmc-based particle filtering for tracking a variable number of interacting targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence archive* **27(11)**, 1805 – 1918 (2005).

[KF07] NEZAMODDIN N. KACHOUIE Y PAUL FIEGUTH. Toward an optimal solution for multitarget tracking. En "Proceedings of IEEE International Conference on Image Processing", tomo 6, páginas 329 – 332 (2007).

[KHK04] CHRIS M. KREUCHER, ALFRED O. HERO Y KEITH D. KASTELLA. Multiple model particle filtering for multitarget tracking. En "The Proceedings of The Twelfth Annual Conference on Adaptive Sensor Array Processing (ASAP)" (Marzo 2004).

[KKH03] CHRIS M. KREUCHER, KEITH D. KASTELLA Y ALFRED O. HERO. Tracking multiple targets using a particle filter representation of the joint multitarget probability density. En "Proceedings of SPIE-The International Society for Optical Engineering", páginas 258 – 269. International Society for Optical Engineering (2003).

[KLF+02] MARCUS KLEINEHAGENBROCK, SEBASTIAN LANG, JANNIK FRITSCH, FRANK LÖMKER, GERNOT A. FINK Y GERHARD SAGERER. Person tracking with a mobile robot based on multi-modal anchoring. En "Proc. IEEE Int. Workshop on Robot and Human Interactive Communication (ROMAN)", Berlin, Alemania (Septiembre 2002).

[KMA01] ESTHER B. KOLLER-MEIER Y FRANK ADE. Tracking multiple objects using the condensation algorithm. *Robotics and Autonomous Systems* **34**(2-3), 93 – 105 (2001).

[Koc03] REINHARD KOCH. 3d-scene modeling from image sequences. En "ISPRS Archives", tomo 34 (Septiembre 2003).

[KS00] KIRIAKOS KUTULAKOS Y STEVEN M. SEITZ. A theory of shape by space carving. *International Journal of Computer Vision* **38**(3), 199 – 218 (2000).

[KV98] KIRIAKOS KUTULAKOS Y JAMES R. VALLINO. Calibration-free augmented reality. *IEEE Transactions on Visualization and Computer Graphics* **4**(1), 1 – 20 (1998).

[LAS03] CARLOS LEUNG, BEN APPLETON Y CHANGMING SUN. Embedded voxel colouring. En "Digital Image Computing - Techniques and Applications", páginas 97 – 106 (2003).

[LC98] JUN S. LIU Y RONG CHEN. Sequential monte carlo methods for dynamical systems. *Journal American Statistic Association* **93**, 1032 – 1044 (1998).

[Lee07] YANG WEON LEE. Development of the multi-target tracking scheme using particle filter. *Advances in Neural Networks* **4493**, 1192 – 1201 (2007).

[LGLB02] JEAN LOUCHET, MAUD GUYON, MARIE-JEANNE LESOT Y AMINE BOUMAZA. Dynamic flies: a new pattern recognition tool applied to stereo sequence processing. *Pattern recognition letters* **23**, 335 – 345 (2002).

[LK81] BRUCE D. LUCAS Y TAKEO KANADE. An iterative image registration technique with an application to stereo vision. En "Proceedings of Imaging Understanding Workshop", páginas 121 – 130 (1981).

[Lou01] JEAN LOUCHET. Using an individual evolution strategy for stereovision. *Genetic Programming and Evolvable Machines* **2**, 101 – 109 (2001).

[LRLY05] JONGWOO LIM, DAVID A. ROSS, RUEI-SUNG LIN Y MING-HSUAN YANG. Incremental learning for visual tracking. En LAWRENCE K. SAUL, YAIR WEISS Y LÉON BOTTOU, editores, "Advances in Neural Information Processing Systems 17". MIT Press, Cambridge, MA (2005).

[Mac03] DAVID J.C. MACKAY. "Information Theory, Inference, and Learning Algorithms". Cambridge University Press (2003).

[MB00] JOHN MACCORMICK Y ANDREW BLAKE. A probabilistic exclusion principle for tracking multiple objects. *International Journal of Computer Vision* **39**(1), 57 – 71 (2000).

[MKK07] MARK R. MORELANDE, CHRIS M. KREUCHER Y KEITH D. KASTELLA. A bayesian approach to multiple target detection and tracking. *IEEE Transactions of Signal Processing* **55**(5), 1589 – 1604 (2007).

[MMBS02] ALAN MARRS, SIMON MASKELL Y YAAKOV BAR-SHALOM. Expected likelihood for tracking in clutter with particle filters. En "Proceedings of SPIE Signal and Data Processing of Small Targets", tomo 4728, páginas 230 – 239 (2002).

[MMHM02] 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).

[MN78] DAVID MARR Y KEITH NISHIHARA. Representation and recognition of the spatial organization of three dimensional structure. En "Proceedings of the Royal Society of London, Biological Sciences", tomo 200, páginas 269 – 294 (1978).

[MP79] DAVID MARR Y TOMASSO POGGIO . A computational theory of human stereo vision . En "Proceedings of the Royal Society of London, Biological Sciences", tomo 204, páginas 301 – 328 (1979).

[MSG+05] MARTA MARRÓN, MIGUEL A. SOTELO, JUAN C. GARCÍA, DAVID FERNÁNDEZ Y DANIEL PIZARRO. "xpfcp": an extended particle filter for tracking multiple and dynamic objects in complex environments. En "International Conference on Intelligent Robots and Systems" (Agosto 2005).

[MSKS04] YI MA, STEFANO SOATTO, JANA KOSECKA Y S. SHANKAR SASTRY. "An Invitation to 3-D Vision". Springer-Verlag New York (2004).

[MSW02] ADAM MILSTEIN, JAVIER NICOLÁS SÁNCHEZ Y EVAN WILLIAMSON. Robust global localization using clustered particle filtering. En "Eighteenth national conference on Artificial intelligence", páginas 581 – 586, Menlo Park, CA, USA (2002). American Association for Artificial Intelligence.

[MT98] DIMITRIS MARGARITIS Y SEBASTIAN THRUN. Learning to locate an object in 3D space from a sequence of camera images. En "Proc. 15th International Conf. on Machine Learning", páginas 332 – 340. Morgan Kaufmann, San Francisco, CA (1998).

[NJS06] HIEU T. NGUYEN, QIANG JI Y ARNOLD W. M. SMEULDERS. Robust multi-target tracking using spatio-temporal context. En "Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition", tomo 1, páginas 578 – 585 (2006).

[NLGP07] WILLIAM NG, JACK LI, SIMON GODSILL Y SZE KIM PANG. Multitarget initiation, tracking and termination using bayesian monte carlo methods. *The Computer Journal* **50**(6), 674 – 693 (2007).

[NSC06] PETER NILLIUS, JOSEPHINE SULLIVAN Y STEFAN CARLSSON. Multi-target tracking – linking identities using bayesian network infe-

rence. En "In Proc. IEEE Computer Vision and Pattern Recognition (CVPR06), New York City" (2006).

[OF02] MATTHEW ORTON Y WILLIAM FITZGERALD. A bayesian approach to tracking multiple targets using sensorarrays and particle filters. *IEEE Transactions on Signal Processing* **50**(2), 216 – 223 (2002).

[ORS04] SONGHWAI OH, STUART RUSSELL Y SHANKAR SASTRY. Markov chain monte carlo data association for general multiple-target tracking problems. En "43rd Conference on Decision and Control", tomo 1, páginas 735 – 742 (Diciembre 2004).

[OTF+04] KENJI OKUMA, ALI TALEGHANI, NANDO DE FREITAS, JAMES LITTLE Y DAVID LOWE. A boosted particle filter: Multitarget detection and tracking (2004).

[Pan05] JUAN JOSÉ PANTRIGO FERNÁNDEZ. "Resolución de Problemas de Optimización Dinámica mediante la Hibridación entre Filtros de Partículas y Metaheurísticas Poblacionales". Tesis Doctoral, Universidad Rey Juan Carlos (2005).

[PC05] MARK PUPILLI Y ANDREW CALWAY. Real-time camera tracking using a particle filter. En "Proceedings of the British Machine Vision Conference", páginas 519 – 528 (2005).

[PC06] MARK PUPILLI Y ANDREW CALWAY. Real-time visual slam with resilience to erratic motion. *IEEE Computer Vision and Pattern Recognition* (2006).

[PLLP04] OLIVIER PAUPLIN, JEAN LOUCHET, EVELYNE LUTTON Y MICHEL PARENT. Obstacle detection by evolutionary algorithm: the Fly algorithm. En "Proc. 2nd Int. Conf. on Autonomous Robots and Agents, ICARA 2004", páginas 136 – 140, Palmerston North (Nueva Zelanda) (Diciembre 2004).

[PVB04] PATRCIK PÉREZ, JACO VERMAAK Y ANDREW BLAKE. Data fusion for visual tracking with particles. *Proc. IEEE* **92**(3), 495 – 513 (2004).

[QC04] GANG QIAN Y RAMA CHELLAPPA. Structure from motion using sequential monte carlo methods. *International Journal of Computer Vision* **59**, 5 – 31 (2004).

[QCZ05] GANG QIAN, RAMA CHELLAPPA Y QINGEN ZHENG. Bayesian algorithms for simultaneous structure from motion estimation of multiple independently moving objects. *IEEE Transactions on Image Processing* **14**(1) (2005).

[RC01] YONG RUI Y YUNQIANG CHEN. Better proposal distributions: Object tracking using unscented particle filter. En "Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Kauai, Hawaii", tomo II, páginas 786 – 793 (2001).

[Rei79] DONALD B. REID. An algorithm for tracking multiple targets. *IEEE Tran. on Automatic Control* **24**(6) (1979).

[Rub98] DONALD B. RUBIN. "Bayesian Statistics 3", capítulo Using the SIR algorithm to simulate posterior distributions, páginas 395 – 402. Oxford University Press (1998).

[SBC99] LAWRENCE D. STONE, CARL A. BARLOW Y THOMAS L. CORWIN. "Bayesian Multiple Target Tracking". Arthec House (1999).

[SD99] STEVEN M. SEITZ Y CHARLES R. DYER. Photorealistic scene reconstruction by voxel coloring. *International Journal of Computer Vision* **35**(2), 151 – 173 (1999).

[SFNT04] SHOICHI SHIMIZU, HIRONOBU FUJIYOSHI, YASUNORI NAGASAKA Y TOMOICHI TAKAHASHI. A pseudo stereo vision method for unsynchronized cameras. En "Proceedings of Asian Conference on Computer Vision", tomo 1, páginas 575 – 580 (Enero 2004).

[SGPO05] KEVIN SMITH, DANIEL GATICA-PEREZ Y JEAN-MARC ODOBEZ. Using particles to track varying numbers of interacting people. En "Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition", tomo 1, páginas 962 – 969 (2005).

[SL04] IRYNA SKRYPNYK Y DAVID G. LOWE. Scene modelling, recognition and tracking with invariant image features. En "IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)", páginas 110–119 (2004).

[ST94] JIANBO SHI Y CARLO TOMASI. Good features to track. En "IEEE Conference on Computer Vision and Pattern Recognition" (1994).

[TC02] DAVID TWEED Y ANDREW CALWAY. Tracking many objects using subordinated condensation. En "British Machine Vision Conference", páginas 283 – 292 (2002).

[TFB00] SEBASTIAN THRUN, DIETER FOX Y WOLFRAM BURGARD. Monte carlo localization with mixture proposal distribution. En "Proceedings of the Innovative Applications of Artificial Intelligence Conference", páginas 859 – 865 (2000).

[TFBD01] SEBASTIAN THRUN, DIETER FOX, WOLFRAM BURGARD Y FRANK DELLAERT. Robust monte carlo localization for mobile robots. *Artificial Intelligence* **128**, 99 – 141 (2001).

[VDP03] JACO VERMAAK, ARNAUD DOUCET Y PATRICK PÉREZ. Maintaining multi-modality through mixture tracking. En "International Conference on Computer Vision" (2003).

[YL06] XIAOTONG YUAN Y STAN Z. LI. Learning feature extraction and classification for tracking multiple objects: A unified framework. En "Proceedings of the IEEE International Conference on Video and Signal Based Surveillance", página 22 (2006).

[ZDD01] DMITRY N. ZOTKIN, RAMANI DURAISWAMI Y LARRY S. DAVIS. Multimodal 3-d tracking and event detection via the particle filter. En "IEEE Workshop on Detection and Recognition of Events in Video", páginas 20 – 27 (2001).

[ZN04] TAO ZHAO Y RAM NEVATIA. Tracking multiple humans in crowded environment. *Journal on Computer Vision and Pattern Recognition* **2**, 406 – 413 (2004).