UNIVERSIDAD DE ALICANTE

Tesis Doctoral

APLICACIÓN DEL MUESTREO BAYESIANO EN ROBOTS MÓVILES: ESTRATEGIAS PARA LOCALIZACIÓN Y ESTIMACIÓN DE MAPAS DEL ENTORNO

Presentada por
Domingo Gallardo López
Dirigida por
Dr. Ramón Rizo Aldeguer
Dr. Francisco Escolano Ruiz
Departamento de Tecnología Informática y Computación

Mayo de 1999

Para Conso

Agradecimientos

En cualquier proyecto de investigación, y en una tesis doctoral en especial, es fundamental tener alrededor buenos compañeros con los que compartir el trabajo y las experiencias. Esta tesis no hubiera sido posible sin las muchas personas que me han ayudado desde el principio, aportando ideas, propuestas, caminos a seguir o, simplemente, estando ahí para discutir y charlar.

Me gustaría agradecer sinceramente a mi director de tesis, *Ramón Rizo*, y a mi co-director, *Francisco Escolano*, por su continuo apoyo, ánimo y por todas sus sugerencias. A *Vicente Botti*, *Alfons Crespo*, *Federico Barber* y *Eva Onaindía* por lo mucho que aprendí en Valencia. A *Reid Simmons*, *Rich Goodwin* y *Hank Wan*, por su apoyo y amistad en los meses en CMU. Por último, a los buenos amigos y compañeros del grupo de investigación, *Faraón Llorens*, *Rosana Satorre*, *Isabel Alfonso*, *Pilar Arques* y, en especial a *Miguel Cazorla* y *Otto Colomina*, por compartir también proyectos, ideas y "pizarrón".

Índice General

Índice de Figuras

Índice de Tablas

Capítulo 1

Introducción

En esta tesis se estudian dos de los problemas fundamentales de la navegación de robots autónomos, el problema de la localización y el de la construcción automática de mapas del entorno. Proponemos abordar ambas cuestiones desde un punto de vista de estimación bayesiana del estado y de búsqueda del modelo de máxima verosimilitud, aplicando técnicas novedosas de modelado y de computación.

Para ilustrar el alcance de estas técnicas, describimos a continuación el comportamiento de un robot guía de un museo. Esta tarea es un ejemplo típico de aplicación de los robots móviles.

> El robot guía se encuentra localizado en una determinada posición de un gran salón en donde se distribuyen diversas obras, no sólo en las paredes sino también por el centro del mismo. Una pantalla táctil situada sobre el robot permite a los usuarios escoger la obra a la que se desea ir. Alguien pulsa la obra situada al otro extremo del salón y el robot, después de planificar una trayectoria factible que le llevará de la posición actual a la posición objetivo, comienza a moverse, invitando con un sintetizador voz a que le sigan. Utiliza los lectores de ultrasonidos y de láser para obtener lecturas de alcance del entorno, que procesa para moverse evitando obstáculos y para actualizar de forma correcta su localización, compensando los errores introducidos por la odometría. Al comenzar a moverse, el robot despierta una gran curiosidad y bastantes personas se agrupan a su alrededor, produciendo lecturas erróneas y bloqueando el camino que se había calculado previamente. El robot corrige la trayectoria, encontrando un camino alternativo y se mueve para evitar los nuevos obstáculos, llegando al objetivo sin más problemas. Durante el camino ha ido anunciando las obras que dejaba a derecha e izquierda. Una vez en el objetivo se detiene, listo para volver a comenzar.

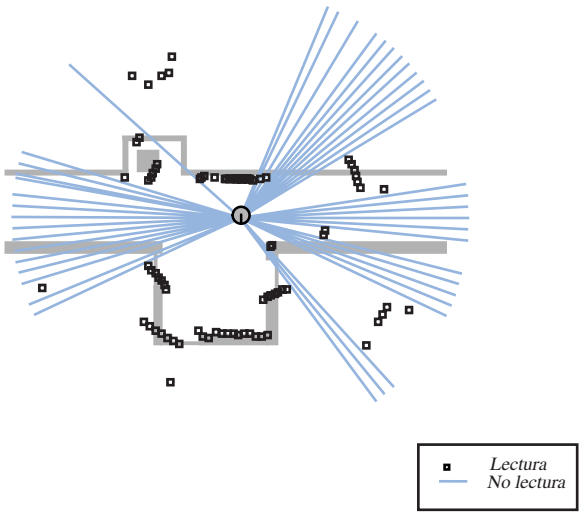


Figura 1.1: Ejemplo de lecturas de sensores de ultrasonidos en una habitación.

En este ejemplo, el robot es capaz de contestar en cada momento a las siguientes tres preguntas: ¿dónde estoy? ¿hacia dónde debo moverme? ¿cómo llegaré hasta allá?

Estas tres preguntas sugieren tres problemas fundamentales de la navegación: el problema de la localización, el problema de la planificación de conductas (trayectorias) y el problema de la utilización de esquemas de actuación.

En el problema de localización, en concreto, el robot se mueve en un entorno de oficina cuyo mapa se conoce de antemano, y cuenta con unos sensores de ultrasonidos con los que obtiene una percepción local (y ruidosa) del entorno en el que se encuentra (ver figura 1.1). También existen en el robot unos contadores mecánicos que proporcionan información métrica (también ruidosa) de los desplazamientos que realiza (*odometría*).

Se trata de encontrar, en todo momento, la posición del robot dentro del mapa conocido a partir de la información de las lecturas del sonar y de las lecturas de odometría. En la versión más complicada del problema, la *localización global*, se debe estimar la posición del robot *sin conocer su posición inicial*. Frente a este problema, el del *seguimiento de la localización* es más sencillo, ya que se conoce a priori la posición del robot.

Las técnicas de localización deben tratar correctamente el problema de los objetos no modelados en el entorno. Normalmente, el mapa que modela el entorno representa únicamente las características principales del mismo. Sin embargo, cuando el robot evoluciona por dicho entorno sus sensores se verán afectados por pequeños objetos no modelados (mobiliario, por ejemplo), por características que han cambiado (puertas que se abren o se cierran) o por personas que se mueven en los alrededores. Las técnicas de localización deben ser lo

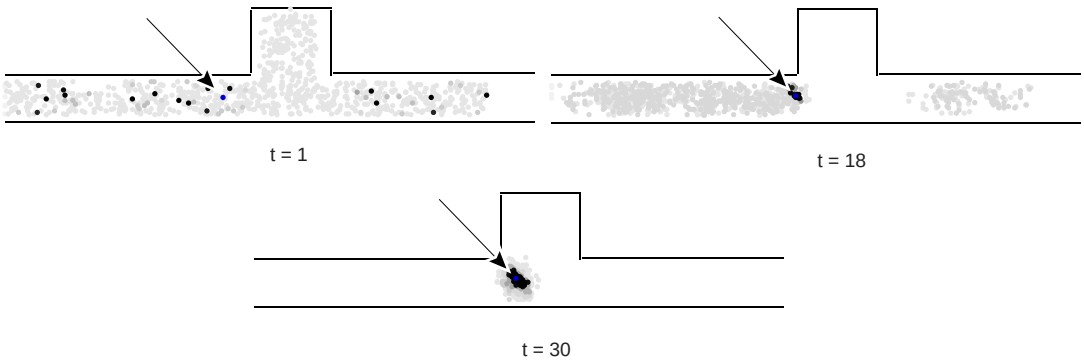


Figura 1.2: Muestras que representan la distribución de probabilidad de la posición del robot.

suficientemente robustas para contemplar estas diferencias.

En el problema de la construcción automática de mapas del entorno el robot explora un entorno y almacena las lecturas de los sensores de ultrasonidos y las lecturas de odometría. Una vez realizada la exploración, se debe determinar el modelo del entorno que mejor se adapta a las lecturas recogidas. Normalmente, en el mapeado también hay que resolver el problema de localización, ya que no se conocen las posiciones desde las que el robot ha realizado las lecturas.

En esta tesis se presenta una solución al problema de la localización global utilizando la estimación bayesiana. Este paradigma permite derivar una función de probabilidad *a posteriori* de la posición del robot, dado un mapa conocido del entorno, una secuencia de lecturas de los sensores del robot y una secuencia de sus movimientos. Las posiciones más probables serán aquellas que hagan más verosímiles dichas secuencias de lecturas y movimientos, dado el mapa del entorno.

Uno de los problemas más graves de la aplicación de este enfoque es que, debido a los problemas de obstáculos no modelados y de ambigüedad en la percepción del entorno, la función de probabilidad a posteriori es muy compleja y no puede aproximarse a una distribución normal (frecuentemente es multimodal). La solución novedosa que aportamos es la utilización del filtro *bootstrap* que representa la función de probabilidad mediante un conjunto de muestras extraídas de la misma. Por ejemplo, en la figura 1.2, se representan todas las muestras que definen la probabilidad de localización a posteriori en tres instantes de tiempo de un recorrido del robot por un pasillo.

Para calcular la función de probabilidad, y aplicar el algoritmo *bootstrap* es necesario formular un modelo de observación, que define la verosimilitud de las lecturas obtenidas por los sonares dada una posición en un determinado entorno, y un modelo de movimiento del robot. Presentamos en la tesis un modelo novedoso de observación, basado en un

Figura 1.3: PIXIE en un pasillo.

modelo realista del comportamiento del sensor de ultrasonidos, que tiene como características principales la robustez frente a obstáculos no modelados y la alta efectividad en situaciones que otros modelos más simples perciben como ambiguas.

En lo que se refiere al problema del mapeado, proponemos la utilización de modelos paramétricos del entorno y de un algoritmo de *estimación-maximización* (EM) para obtener los parámetros del mapa que maximizan la verosimilitud de las lecturas. Se utiliza como método de estimación el algoritmo de localización, al que se añade una fase posterior de suavizado de las muestras para reducir la ambigüedad en situaciones ruidosas. La maximización se utiliza mediante un sencillo algoritmo de búsqueda adaptativa.

Se han probado los métodos con datos obtenidos de un simulador propio (figura 1.4) y de PIXIE (figura 1.3), un robot móvil RWI B-21 con un anillo de 24 sensores de ultrasonidos.

Una de las características principales del simulador construido es su flexibilidad. Por ejemplo, es posible obtener datos de distintos modelos de entorno, modificar el comportamiento de las lecturas de los sensores de ultrasonidos, introduciendo distintos tipos de ruido aleatorio, o añadir ruido en las lecturas de odometría. La utilización del simulador permite, en la fase de diseño de los algoritmos, controlar mejor sus distintas variables y proporciona una enorme flexibilidad por la posibilidad de repetir de forma controlada los experimentos. El robot PIXIE se ha utilizado para tomar datos reales con los que validar el funcionamiento correcto de los algoritmos propuestos.

Tanto el filtro *bootstrap* como los enfoques bayesianos se han popularizado recientemente

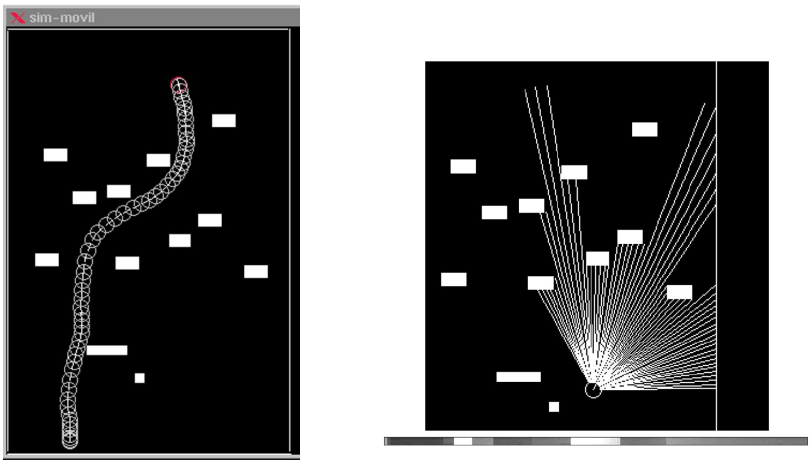


Figura 1.4: Simulador con el que se ha realizado parte de la experimentación.

en la comunidad científica dedicada a la Visión Artificial. Muchas de las ideas aportadas en esta tesis han surgido de la adaptación de técnicas recientes de este campo a los métodos de navegación.

El resto de la tesis se organiza de la siguiente forma. El capítulo 2 presenta los problemas de localización y mapeado, haciendo un repaso de las distintas técnicas propuestas. Se presenta también en el capítulo una formalización de los problemas en términos de estimación bayesiana que proporciona el marco conceptual del resto de la tesis. El capítulo 3 presenta la propuesta de modelo del sensor, junto con experimentos que muestran su correcto funcionamiento. En el capítulo 4 se proponen los modelos necesarios para utilizar el enfoque bayesiano. Estos modelos son los mapas paramétricos del entorno, el modelo de observación y el modelo de movimiento. El capítulo 5 está dedicado a la presentación del filtro *bootstrap* y a su aplicación a la solución del problema de localización global. Por último, el capítulo 6 propone un algoritmo EM para resolver el problema del mapeado. En este capítulo se presenta una técnica para suavizar la probabilidad asociada a las muestras resultantes del filtro *bootstrap* y se utiliza una búsqueda adaptativa para encontrar el modelo paramétrico que mejor se adapta a las lecturas obtenidas. En el capítulo 7 se resumen todas las aportaciones de la tesis y se comentan posibles líneas futuras de trabajo.

Los apéndices amplían algunos conceptos importantes y presentan trabajos previos relacionados con el tema de la tesis. El primero de ellos describe el método estadístico del muestreo por rechazo para muestrear funciones de densidad. El apéndice B describe el algoritmo EM. Los apéndices C y D presentan trabajos previos al desarrollo del cuerpo central de la tesis, relacionados con las conductas y las trayectorias de los robots móviles, y que sirvie-

ron para plantear algunas de las problemáticas de las que han surgido esta tesis. En concreto, el apéndice 3 propone la utilización de algoritmos genéticos para resolver el problema de la planificación de trayectorias, y el apéndice 4 utiliza estas trayectorias como modelos de los que extraer características comunes que definen conductas locales de navegación, utilizando el análisis de componentes principales (PCA).

Capítulo 2

Localización y mapeado en entornos de oficina

La localización de un robot móvil en un entorno conocido es uno de los problemas fundamentales de la robótica móvil, junto con la construcción automática de mapas del entorno. Se han propuesto un gran número de modelos, enfoques y técnicas para resolver ambos problemas. Muchas de estas técnicas presentan soluciones *ad-hoc* que sólo son útiles en situaciones muy específicas.

Frente a este tipo de técnicas, han aparecido recientemente algunas propuestas generales que utilizan enfoques bayesianos, como por ejemplo los modelos de Markov o las redes bayesianas.

Presentamos en este capítulo una formalización del problema de la localización y del mapeado que unifica las propuestas recientes y que proporciona una notación uniforme y coherente. Esta formalización sirve de marco conceptual en el que es posible desarrollar distintas implementaciones y técnicas.

Los elementos fundamentales de la formulación de ambos problemas son la definición de un *modelo de observación* que proporciona la probabilidad de las lecturas de los sensores dada una posición en el entorno definido y la definición de un *modelo de movimiento* que proporciona la probabilidad de la siguiente posición del robot, dada la posición actual y la acción ejecutada.

En capítulos siguientes presentaremos propuestas concretas para ambos modelos y algoritmos y técnicas de localización y mapeado basadas en esta formalización.

2.1 Introducción

Para navegar de forma robusta en entornos de oficina, un robot debe saber dónde se encuentra dentro de ese entorno. En los últimos años viene existiendo un gran interés en el desarrollo de algoritmos para estimar la localización del robot a partir de los datos percibidos por sus sensores. En el contexto de los robots móviles, el problema general de la localización puede ser formulado de la siguiente forma.

Dado: Un modelo del entorno, como una descripción geométrica, un mapa topológico o una rejilla de ocupación.

Tarea: Estimar la localización del robot en el modelo basándose únicamente en observaciones efectuadas por el robot. Dichas observaciones suelen consistir en información de odometría acerca de los movimientos realizados por el robot y en información de distancias a los obstáculos más cercanos obtenidas mediante sensores de alcance (sonar, láser)[1]. Las observaciones también pueden consistir en imágenes obtenidas por una cámara montada en el robot, en cuyo caso estamos ante un problema de *localización visual*.

Un problema fuertemente ligado al de localización es el de la construcción automática del mapa del entorno. También en los últimos años se han desarrollado métodos para estimar y mantener modelos del entorno de forma autónoma. Este problema se puede formular de la siguiente forma.

Dado: Una serie de observaciones realizadas por el robot evolucionando por el entorno. Las observaciones, al igual que en el problema de la localización, suelen consistir en información de odometría e información de lecturas de alcance.

Tarea: Construir un modelo del entorno que pueda ser utilizado por los algoritmos de localización. Posibles modelos son descripciones geométricas del entorno, mapas topológicos o rejillas de ocupación.

Los modelos y métodos propuestos para resolver los problemas de localización y mapeado deben tratar con ciertas limitaciones y restricciones prácticas del funcionamiento de los robots móviles. Algunas de ellas son las siguientes.

1. **Localidad de los sensores.** El rango de percepción de la mayoría de sensores (ultrasonidos, láser, cámaras) está limitado a una zona pequeña alrededor del robot. Para adquirir información global, el robot debe explorar activamente su entorno.

[1] Es importante resaltar que sólo se dispone de información local sobre la posición del robot. Evidentemente, si se contara con información global sobre su posición (mediante un sistema GPS, por ejemplo), el problema de la localización no existiría.

2. **Ruido en los sensores.** Las observaciones realizadas por los sensores son normalmente ruidosas, y la distribución estadística de este ruido no suele ser sencilla de modelar.

3. **Ruido en la posición.** Los movimientos del robot no suelen ser exactos, produciéndose los denominados errores de odometría. Estos errores son, además, acumulativos con la distancia recorrida. Por ejemplo, pequeños errores en la rotación del robot pueden tener efectos importantes en la estimación de los movimientos de traslación y en la determinación de la posición final del robot.

4. **Entornos complejos y dinámicos.** Los entornos de oficina en los que evoluciona el robot suelen ser complejos y dinámicos, haciendo prácticamente imposible mantener modelos consistentes de los mismos.

5. **Necesidad de tiempo real.** Los requisitos de la aplicación (control de un robot móvil) obligan a procesar la información en un tiempo real. Esto limita la complejidad del procesamiento realizado por los métodos de localización, así como los modelos del entorno.

En los siguientes apartados revisaremos los distintos aspectos de los problemas de localización y mapeado. Comenzaremos estudiando los distintos modelos del entorno propuestos en la literatura, a continuación presentaremos las aproximaciones al problema de la localización y al problema del mapeado. Se concluirá el capítulo definiendo formalmente el marco de estimación bayesiana, aplicado tanto a localización como a mapeado.

2.2 Modelos del entorno

Un elemento fundamental de la localización y el mapeado es el modelo de representación del entorno. Un modelo o mapa del entorno es una abstracción con la que se representan únicamente aquellas características del entorno que se consideran útiles para la navegación o la localización del robot. Al realizar esta abstracción se desechan características de grano fino que no se consideran útiles, debido a que pueden ser demasiado variables o no pueden ser detectadas con fiabilidad por los sensores.

La utilidad principal de un modelo del entorno es proporcionar el elemento fundamental para la localización del robot. En general, los algoritmos de localización suelen comparar las lecturas obtenidas por los sensores del robot con el modelo del entorno, actualizando la posición del robot de forma acorde con el resultado de esta comparación. La forma de realizar la comparación depende totalmente del tipo de modelo de entorno y de la propuesta realizada.

Se han desarrollado dos paradigmas fundamentales de modelado de los entornos de oficina: modelos métricos y modelos topológicos. A su vez, los modelos métricos pueden dividirse en modelos basados en rejillas y en modelos geométricos. En los siguientes apartados se revisarán los modelos de entorno más utilizados en la literatura, analizando las características y suposiciones de cada enfoque.

2.2.1 Mapas topológicos

La idea central de los mapas topológicos es representar las características esenciales del entorno percibidas por el robot móvil utilizando un grafo como un modelo de alto nivel. Los nodos se utilizan para representar lugares del entorno y los arcos caminos entre los lugares. Los lugares constituyen zonas del entorno (*landmarks*) con características sensoriales distinguibles de forma absoluta, o respecto a sus nodos vecinos.

Los nodos corresponden a la unidad elemental de localización, de manera que toda una zona geométrica del mapa real se representa por un único lugar. A partir del mapa topológico no es posible distinguir localizaciones más finas que las representadas por los lugares.

Veamos, como ejemplo, la propuesta de mapas topológicos de Kuipers (Kuipers y Byun 1991)(ver figura 2.1).

Los nodos corresponden a puntos distintivos del entorno y los arcos a caminos recorridos por el robot. Una posición del entorno correspondiente a un nodo debe distinguirse localmente de su vecindad mediante algún criterio definible en términos de los datos sensoriales. En el caso de los experimentos realizados por Kuipers, la función de distinción calcula el número de objetos cercanos que se encuentran a igual distancia del robot.

Los arcos entre los nodos representan caminos que se han seguido para llegar de un nodo a otro utilizando una determinada estrategia de control local (*seguir centro de pasillo*, *seguir pared a la derecha* o *seguir pared a la izquierda*).

Los mapas topológicos han sido utilizados, con múltiples variantes, a lo largo de los últimos años (Mataric 1992; Pierce y Kuipers 1994; Kortenkamp y Weymouth 1994; Shatkay y Kaelbling 1997; Nourbakhsh, Powers, y Birchfeild 1995; Ryu y Yang 1988; Koenig y Simmons 1998; Thrun 1998; Thrun, Burgard, y Fox 1998).

Un inconveniente de los mapas topológicos es que la necesidad de distinción sensorial entre lugares hace imposible la representación de zonas abiertas (habitaciones grandes, halls) en las que el alcance limitado de los sensores no obtiene información.

Otro punto débil es que la definición de lugares y la conexión entre los mismos es muy dependiente de la aplicación, no utilizándose normalmente ningún criterio formal para su construcción.

Por último, como se puede comprobar en el ejemplo de la figura 2.1, los mapas construidos dependen excesivamente de la historia de las percepciones del robot al construirlos (por ejemplo, el arco entre $P1$ y $P2$ está etiquetado *seguir pared izquierda* porque esa es

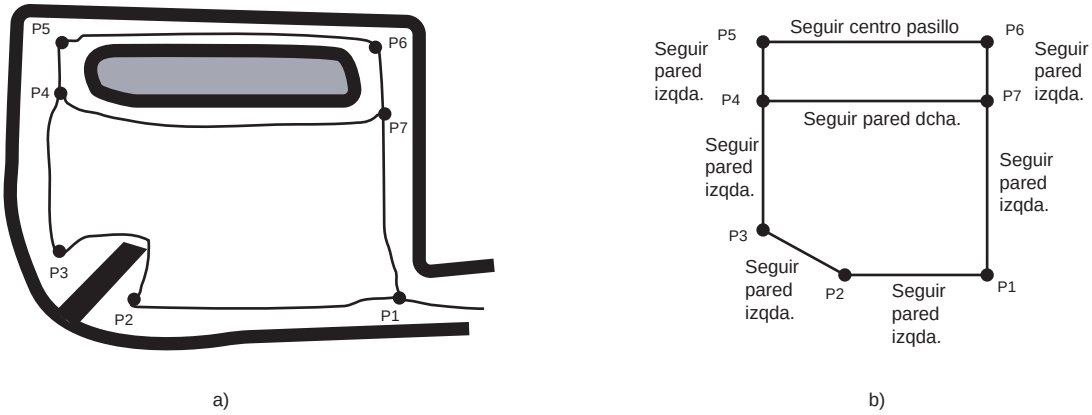


Figura 2.1: Ejemplo de mapa topológico de Kuipers.

precisamente la conducta local que el robot siguió para ir de un nodo a otro). Esto los hace muy sensibles a la aparición de elementos no modelados (personas, obstáculos imprevistos) que proporcionan información sensorial muy distinta de la modelada, haciendo que el robot pierda su localización.

Por otro lado, los modelos topológicos proporcionan ventajas a la hora de realizar una planificación de la trayectoria del robot, facilitan la interfaz con planificadores simbólicos y proporcionan un interfaz más natural para la interacción con instrucciones humanas (posibilitan órdenes del tipo *"ir a la habitación A"*).

2.2.2 Rejillas de ocupación

Las rejillas de ocupación, inicialmente propuestas por Moravec y Elfes (Moravec y Elfes 1985), discretizan el entorno en celdillas de igual dimensión. Cada celdilla mantiene la probabilidad de que la zona del entorno asociada a ella esté ocupada. En la figura 2.2 se muestra un ejemplo de una rejilla de ocupación.

Es posible utilizar rejillas de ocupación definidas por el usuario, pero lo usual es que sea el propio robot móvil el que realize la construcción de la rejilla de forma autónoma, mediante algún algoritmo de exploración (Elfes 1987; Weigl, Siemiatkowska, Sikorski, y Borkowski 1993; Thrun, Burgard, y Fox 1998).

Las rejillas de ocupación se han utilizado desde entonces en numerosos enfoques de localización (Matthies y Elfes 1988; Courtney y Jain 1994; Schiele y Crowley 1994; Stevens, Stevens, y Durrant-Whyte 1995; Oriolo, Vendittelli, y Ulivi 1995; Daniel Pagac 1996;

Figura 2.2: Ejemplo de rejilla de ocupación (tomado de (Thrun, Burgard, y Fox 1998)).

Yamauchi 1996) normalmente para alinear mapas locales construidos mediante los datos de los sensores con el mapa de ocupación global.

En los últimos años ha resurgido el interés en las rejillas de ocupación al aparecer algoritmos que permiten manejar el problema de los errores de odometría en la construcción de rejillas de entornos de gran tamaño (ver (Thrun 1998; Thrun, Burgard, y Fox 1998)).

2.2.3 Modelos geométricos

Los modelos geométricos definen el entorno mediante sus características geométricas (distancias, dimensiones de los elementos que lo componen, posiciones absolutas). La ventaja principal de estos modelos es que, si se utilizan junto con un buen modelo del sensor, es posible simular los datos que los sensores del robot obtendrían en cualquier posición del entorno. Esto hace posible comparar los datos percibidos por el robot con los datos que se obtendrían en posiciones candidatas, calculándose una actualización de la probabilidad asociada a cada posición.

Existen distintos tipos de modelos geométricos. Un primer enfoque define el entorno mediante un conjunto de características geométricas (segmentos de rectas, esquinas) y mediante las relaciones geométricas entre ellas (distancia, posición, etc.). Ejemplos de utilización de estos modelos son los trabajos (Drumheller 1987; Neira, Horn, Tardos, y Schmidt 1997; Ohya, Nagashima, y Yuta 1994; Chong y Kleeman 1997; Leonard, Durrant-Whyte, y Cox 1992; Cox 1991; Ayache y Faugeras 1989).

Otro enfoque, los modelos geométricos basados en características, se relaciona directamente con implementaciones de modelos de sensor en las que se utilizan estas características geométricas como elementos base del modelado (Barshan y Kuc 1990; Kuc 1990; McKerrow 1993). En la figura 2.3 se muestra un ejemplo de mapa del entorno construido a base de las características geométricas definidas por Leonard (Leonard y Durrant-Whyte 1992): esquinas, aristas y segmentos de rectas.

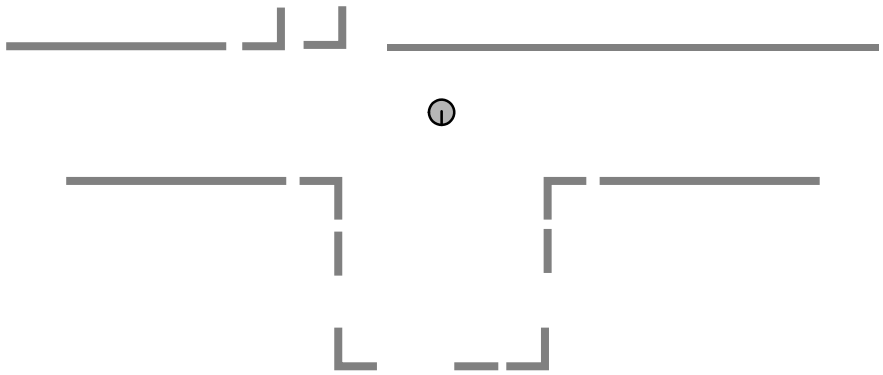


Figura 2.3: Ejemplo de modelo geométrico del entorno. Las características geométricas usadas en el modelo son: *esquinas*, *aristas* y *segmentos de rectas*.

Otro conjunto de modelos geométricos definen el entorno mediante un mapa CAD del mismo (Burgard, Fox, Henning, y Schmidt 1996; Burgard, Cremers, Fox, Hahnel, Lakemeyer, Schulz, Steiner, y Thrun 1998). Un mapa CAD refleja los elementos del entorno que se desea modelar, recogiendo sus dimensiones y posiciones. En la figura 2.4 se muestra un ejemplo de mapa CAD de un entorno.

Cuanto más detallado sea el modelo CAD mayor calidad tendrán las simulaciones de las lecturas de los sensores del robot en las posiciones candidatas. Evidentemente, para obtener simulaciones de lecturas de buena calidad es necesario utilizar un buen modelo del sensor. Burgard (Burgard, Cremers, Fox, Hahnel, Lakemeyer, Schulz, Steiner, y Thrun 1998) propone utilizar una simulación sencilla de sensores de rango, obteniendo la distancia angular con los obstáculos del entorno en una posición determinada.

Por último, los trabajos (Weib, Wetzler, y Puttkamer 1994; Lu y Milios 1994; Gutmann y Schlegel 1996; Lu y Milios 1997) proponen utilizar como modelo del entorno los propios datos percibidos por los sensores del robot (figura 2.5), aplicándoles el mínimo tratamiento posible (si acaso, una corrección de odometría). El problema principal de esta representación es que no se realiza ningún filtrado para eliminar ruido procedente de lecturas erróneas, por

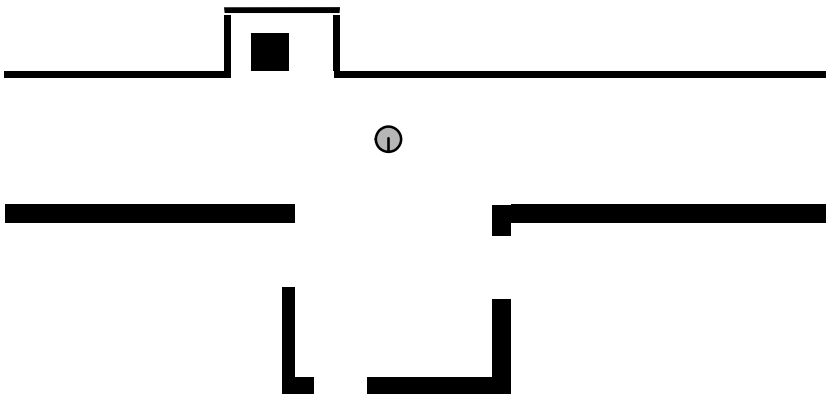


Figura 2.4: Ejemplo de modelo CAD del entorno.

Figura 2.5: Ejemplo de modelo sensorial del entorno, en el que el entorno es el mismo que el modelado por las figuras 2.3 y 2.4.

lo que sólo es aplicable a sensores muy precisos (sensores de alcance por láser).

2.3 Aproximaciones al problema de la localización

Una vez revisados los principales tipos de modelos de entorno, pasamos a tratar el problema de la localización. Localización es el proceso de estimar la posición de un robot móvil en un mapa que determina un sistema de coordenadas globales. Las fuentes de información disponibles para resolver el problema son las observaciones del entorno realizadas por el robot y sus lecturas de odometría (incrementos de posición registrados por el robot).

Otros enfoques más aplicados utilizan elementos externos incorporados artificialmente al entorno (como reflectores, marcas visuales, líneas, etc.) para triangular la posición del robot (ver el informe de Borenstein (J. Borenstein 1996) para un exhaustivo repaso de este tipo de técnicas). Descartamos la utilización de estos enfoques por la estructuración artificial que imponen en el entorno.

Es posible identificar dos variantes del problema general, dependiendo de si se supone conocida la posición inicial del robot o de si la desconocemos. Los enfoques de *localización local* suponen conocida la posición del robot y realizan un seguimiento (*tracking*) de la misma para estimar la siguiente posición. Por otro lado, los enfoques de *localización global* intentan estimar la posición del robot, sin tener un conocimiento a priori de la misma, utilizando la información de las lecturas en varios instantes de tiempo.

Revisaremos en este apartado distintas aproximaciones al problema de localización.

2.3.1 Localización local

El problema de la localización local se puede formular como el problema de realizar un seguimiento del estado del robot que compense los errores de odometría mediante el uso de observaciones del entorno.

Métodos basados en *landmarks*

Un conjunto de técnicas atacan el problema mediante la identificación de *landmarks* en las observaciones. Las posiciones de estos *landmarks* son conocidas, y con ellas puede corregirse la posición del robot. Ejemplos de algoritmos que han implementado con éxito estas técnicas son (Kuipers y Byun 1991; Kortenkamp y Weymouth 1994; Nourbakhsh, Powers, y Birchfeild 1995; Koenig y Simmons 1998; Ryu y Yang 1988).

Métodos basados en emparejamiento de características

Se han propuesto distintos algoritmos basados en el siguiente esquema genérico: (1) estimar características locales a partir de las últimas observaciones y (2) encontrar la mejor corrección a la posición actual del robot maximizando la correlación entre las características locales y el mapa del entorno.

Por ejemplo, Weib (Weib, Wetzler, y Puttkamer 1994) construye histogramas locales a partir de barridos de sensores láser, que después se correlacionan con histogramas almacenados. Yamauchi (Yamauchi 1996) aplica una técnica similar, pero utilizando rejillas de probabilidad (Moravec y Elfes 1985) como elementos de emparejamiento.

Métodos basados en el filtro de Kalman

El enfoque más usado para resolver el problema de la localización local es la utilización del *filtro de Kalman* (Sorenson 1970), una conocida técnica para integrar información en el tiempo. Este método fue inicialmente propuesto por Kalman (Kalman 1960) para estimar el estado de un proceso dinámico lineal arbitrario. Cada variable que describe el estado del proceso se representa mediante una distribución normal. Los parámetros de la distribución (media y varianza) se actualizan siempre que se aplica un comando de control al sistema y siempre que los sensores realizan nuevas mediciones. Estas dos actualizaciones del estado se suelen denominar *predicción* y *corrección*. En la fase de predicción, se modela el cambio del estado debido a las acciones de control. En la fase de corrección se combina la estimación del estado producida por la fase anterior con las lecturas realizadas por los sensores. Tal y como se verá, estas dos fases también están presentes en la estimación bayesiana. De hecho, se ha demostrado que el filtro de Kalman puede verse como un caso particular del enfoque de estimación bayesiana (Barker, Brown, y Martin 1994).

La aplicación del filtro de Kalman a la localización de robots móviles estima la posición (x, y, θ) del robot en el entorno mediante una distribución normal. La covarianza de esta distribución representa la incertidumbre local en la posición estimada. Siempre que se mueve el robot, la posición estimada se desplaza según la distancia medida por la odometría del robot. Las observaciones realizadas por los sensores se utilizan para actualizar la distribución de probabilidad de la localización, buscando la nueva distribución que maximiza la verosimilitud de las lecturas.

La mayoría de algoritmos que aplican el filtro de Kalman utilizan modelos de movimiento similares, pero difieren en cómo se calcula la verosimilitud de las lecturas de los sensores. Existen dos grupos principales de técnicas: basadas en características y basadas en rejillas de ocupación.

Entre las primeras, Leonard (Leonard, Durrant-Whyte, y Cox 1992) busca emparejar características extraidas de las lecturas de sonar con características predichas a partir de un

mapa geométrico del entorno. Las características son planos, cilindros y esquinas basados en *regiones de profundidad constante* estimadas a partir de las lecturas del sonar. Cox (Cox 1991) utiliza distancias medidas por sensores de rango de infrarrojos y las compara con una descripción del entorno basada en segmentos de rectas. Gutmann (Gutmann y Schlegel 1996) extiende este trabajo a modelos del mundo aprendidos en una fase de exploración previa.

Entre las segundas, Schiele (Schiele y Crowley 1994) compara distintas estrategias de seguimiento de la posición del robot basadas en rejillas de ocupación y en sensores de ultrasonidos. En esta propuesta se construye una rejilla de ocupación local que se se empareja con la rejilla global para producir una posición estimada que se combina con estimaciones previas mediante el filtro de Kalman.

Como conclusión, todas las implementaciones de técnicas basadas en el filtro de Kalman suponen que la posición del robot puede representarse mediante una función de densidad normal. Esta limitación hace que estas técnicas no sean robustas a ruido no modelado, ya que únicamente permiten representar una única posición posible del robot (función de densidad unimodal).

En condiciones normales, el ruido en las observaciones (*clutter*) producido por obstáculos no modelados hará de *distractor* del filtro y podrá ocasionar que el robot pierda totalmente su localización.

La suposición de localización gaussiana hace también difícil tratar el problema de la localización global, ya que no se dispone de una estimación inicial para la localización. Además, debido a la ambigüedad en la percepción del entorno (distintas zonas del entorno pueden generar mediciones similares), es necesario utilizar una función de distribución multimodal para representar la posible localización del robot.

2.3.2 Localización global

Tradicionalmente se ha supuesto que para solucionar el problema de la localización global son necesarias técnicas basadas en búsqueda, como la propuesta por Drumheller (Drumheller 1987).

El método de Drumheller obtiene un conjunto de características (segmentos de rectas) a partir de las lecturas de los sonares del robot y busca el mejor emparejamiento entre estas características y el modelo del entorno, utilizando un algoritmo de emparejamiento de características propuesto por Grimson (Grimson 1990). Como hace notar Leonard (Leonard y Durrant-Whyte 1992), el uso de técnicas de búsqueda no es lo suficientemente eficiente para un modo de funcionamiento continuo de un robot móvil. El enfoque de búsqueda fue abandonado por la comunidad de robótica móvil, y el problema de la localización global ha permanecido sin solución hasta la utilización de enfoques bayesianos (Nourbakhsh, Powers, y Birchfeild 1995; Simmons y Koenig 1995; Kaelbling, Cassandra, y Kurien 1996; Burgard,

Fox, Henning, y Schmidt 1996). Estos enfoques se pueden dividir en dos grandes grupos: los que utilizan modelos topológicos y los que usan rejillas de probabilidad. Revisaremos ambos métodos después de introducir los fundamentos de la localización bayesiana.

2.4 Aproximaciones al problema del mapeado

Al igual que los modelos del entorno, podemos distinguir dos enfoques fundamentales al problema del mapeado, a saber, enfoques métricos y enfoques topológicos.

Enfoques métricos

Uno de los métodos más antiguos y usados de construcción de mapas del entorno son las rejillas de ocupación. Las rejillas de ocupación fueron propuestas inicialmente por Elfes y Moravec (Moravec y Elfes 1985; Elfes 1987), y desde entonces se han adaptado en numerosos sistemas robóticos (Borenstein y Korem 1991; Yamauchi 1996; Burgard, Fox, Henning, y Schmidt 1996; Thrun, Bucken, Burgard, Fox, Frohlinghaus, Hennig, Hofmann, Krell, y Schmidt 1998). Constituyen uno de los primeros enfoques probabilísticos capaces de fusionar distintas observaciones realizadas por el robot, además de resaltar el papel fundamental del modelo del sensor en la construcción de los mapas. Su principal problema es la ausencia de mecanismos correctores de los errores de odometría, por lo que no es factible la construcción de mapas de tamaño medio. Este problema ha sido atacado por Thrun (Thrun 1998) mediante la utilización de la hipótesis de ortogonalidad y paralelismo de las paredes del entorno.

Otros enfoques métricos utilizan modelos geométricos del entorno. Por ejemplo, Chatila y Laumon (Chatila y Laumond 1985), en una propuesta similar a la planteada en nuestro trabajo, proponen representar el entorno mediante polígonos en un sistema de referencia global. En la propuesta se sugiere descomponer el espacio libre en un pequeño número de celdas correspondientes a habitaciones, pasillos, puertas, etc. Sin embargo, aunque el enfoque contiene elementos muy atractivos, no se detalla el mismo ni se presentan experimentos que demuestren su aplicabilidad. Leonard (Leonard y Durrant-Whyte 1992) propone la construcción iterativa, mediante un filtro de Kalman, de una interpretación del entorno basada en características elementales como segmentos y esquinas. Thrun (Thrun 1997) construye mapas geométricos de forma incremental a base de concatenar segmentos de rectas detectados en secuencias temporales de mediciones de sonar.

Por último, un conjunto de métodos suponen que se parte de ciertos modelos a priori del entorno e intentan ajustar distintos parámetros del modelo mediante las lecturas realizadas por el robot. Es el caso de los trabajos de Koenig y Simmons (Koenig y Simmons 1996) y Shatkay (Shatkay y Kaelbling 1997), que utilizan el algoritmo EM (también conocido

como Baum-Welch) (Rabiner y Juang 1986) para realizar la estimación. Recientemente, Thrun (Thrun 1998) ha formulado el problema de la construcción de mapas del entorno en términos bayesianos (como detallaremos en el apartado 2.6). Sin embargo, ha aplicado esta formulación al problema más restringido de encontrar el mejor mapa del entorno que explica una secuencia de observaciones de n tipos de landmarks, observaciones que han sido recogidas de forma manual.

Enfoques topológicos

Los enfoques topológicos definen los mapas como grafos, con los nodos correspondiendo a lugares y los arcos a acciones genéricas que mueven el robot de un lugar a otro. A menudo se añade a estos grafos información métrica que facilita la navegación de un lugar a otro. Intentan resolver, sobre todo, el problema del mapeado global.

Uno de los primeros trabajos en esta línea fue el de Kuipers y Byun (Kuipers y Byun 1988; Kuipers y Byun 1991). Los nodos de su propuesta se corresponden con lugares *distinguibles* del entorno mediante alguna función genérica aplicada a los datos percibidos por los sensores. En concreto, proponen utilizar como medida de distinción de los lugares el número de obstáculos equidistantes. De esta forma los nodos de sus grafos topológicos representan máximos locales de esta medida de distinción. Los arcos corresponden a conductas de navegación que el robot utiliza para moverse de un lugar a otro (como *"seguir pared"*, o *"seguir pasillo"*) junto con información métrica adicional sobre la conducta de navegación seguida. El robot explora el entorno y construye el mapa topológico de forma incremental, conforme va encontrando nuevos lugares *distinguibles*. Sin embargo, estas propuestas sólo han sido comprobadas en entornos simulados y, en estas simulaciones, el robot contaba con una brújula que eliminaba los errores de odometría en la orientación.

Un enfoque similar fue propuesto por Mataric (Mataric 1992). Su algoritmo utiliza como nodos topológicos *landmarks* predefinidos como segmentos rectos, puertas o esquinas. Los lugares topológicos vecinos que va encontrando el robot se conectan mediante aristas que representan también conductas de navegación junto con información métrica que ayuda a localizar al robot. La propuesta fue probada en un robot real evolucionando en una pequeña habitación. Los problemas del método propuesto residen en la dificultad de tratar mapas de mayor tamaño y en la sensibilidad del mismo a falsas detecciones de *landmarks*.

2.5 Fundamentos de la localización bayesiana

La localización bayesiana proporciona un potente marco probabilístico general para estimar la posición de un robot móvil en a partir de las observaciones realizadas por el robot y a las acciones realizadas.

Se han realizado distintas propuestas e implementaciones de este paradigma (Nourbakhsh, Powers, y Birchfeild 1995; Simmons y Koenig 1995; Kaelbling, Cassandra, y Kurien 1996; Burgard, Fox, Henning, y Schmidt 1996). En el capítulo 5 aportaremos una nueva propuesta, basada en la utilización de métodos estocásticos de muestreo, que mejora la eficiencia y la precisión de las implementaciones realizadas hasta el momento.

Formularemos en esta sección el enfoque bayesiano utilizando una notación general en la que tendrán cabida distintas implementaciones específicas. Estas implementaciones (filtros de Kalman, rejillas de probabilidad y métodos topológicos) se revisarán posteriormente.

2.5.1 Definiciones y consideraciones previas

Para definir formalmente la localización bayesiana, sea $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t)$ una secuencia de variables aleatorias que representan el estado del robot móvil en sucesivos instantes de tiempo. La variable aleatoria $\mathbf{x}$ puede estar definida sobre el espacio paramétrico de configuraciones (x, y, θ) del robot (siendo x e y coordenadas cartesianas en un mapa global del entorno y θ la orientación del robot) o puede tomar valores en el espacio discreto de nodos topológicos que representan el entorno.

En cada instante de tiempo t el robot realiza una observación $\mathbf{z}_t$ del entorno y realiza una acción $\mathbf{a}_t$. La variable aleatoria $\mathbf{z}$ puede ser una tupla con valores de distancias medidas por sonares o por sensores láser, o puede ser una imagen del entorno tomada por una cámara. La acción $\mathbf{a}_t$ proporciona información sobre el siguiente estado $\mathbf{x}_{t+1}$ del robot y puede tomar valores en el espacio de velocidades (v, ω) del robot (donde v es la velocidad lineal y ω la angular), puede también representar incrementos de posición $(\triangle x, \triangle y, \triangle\theta)$ obtenidos de los mecanismos de *odometría* del robot, o puede representar un valor tomado de un espacio discreto de comandos (*moverse en la dirección* θ, *seguir pared* o *girar a la derecha*).

El enfoque bayesiano nos permite estimar la función de densidad de la posición del robot $\mathbf{x}_t$ en el instante t, dadas las observaciones y acciones realizadas hasta ese instante y dada la probabilidad a priori $p(\mathbf{x}_1)$. Esta función de densidad representa la *probabilidad a posteriori* después de t instantes de tiempo, y se formula matemáticamente como

$$p(\mathbf{x}_t \mid \mathbf{z}_1, \ldots, \mathbf{z}_t, \mathbf{a}_1, \ldots, \mathbf{a}_{t-1}). \tag{2.1}$$

Veremos a continuación una formulación recursiva que permite la actualización de la función de densidad en el instante t, $p(\mathbf{x}_t)$ a partir de la densidad en el instante anterior, $p(\mathbf{x}_{t-1})$, de los datos medidos en el instante t, $\mathbf{z}_t$, y de la acción previa $\mathbf{a}_{t-1}$.

Para llegar a esta formulación es necesario considerar dos suposiciones, a saber, la condición de Markov del modelo dinámico y la independencia de las observaciones. Tratamos cada suposición por separado.

Condición de Markov

El modelo dinámico del robot determina la información que las acciones y los estados previos proporcionan sobre el estado actual. La formulación de este modelo se expresa como una función de probabilidad condicional

$$p(\mathbf{x}_t \mid \mathbf{x}_1, \ldots, \mathbf{x}_{t-1}, \mathbf{a}_1, \ldots, \mathbf{a}_{t-1}).$$

La condición de Markov sobre el modelo dinámico determina que el nuevo estado depende únicamente del estado y de la acción anterior. Esto es,

$$p(\mathbf{x}_t \mid \mathbf{x}_1, \ldots, \mathbf{x}_{t-1}, \mathbf{a}_1, \ldots, \mathbf{a}_{t-1}) = p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{a}_{t-1}). \tag{2.2}$$

La condición de Markov establece que el conocimiento de las acciones y posiciones previas, $(\mathbf{a}_1, \ldots, \mathbf{a}_{t-2}, \mathbf{x}_1, \ldots, \mathbf{x}_{t-2})$, no proporciona ninguna información adicional a la derivada de conocer la posición y acción inmediatamente previas.

Independencia en las observaciones

La segunda suposición se refiere a las medidas $\mathbf{z}$ observadas por el robot. Se supone que dichas observaciones son independientes con respecto al tiempo, esto es, que

$$p(\mathbf{z}_1, \ldots, \mathbf{z}_t \mid \mathbf{x}_1, \ldots, \mathbf{x}_t, \mathbf{a}_1, \ldots, \mathbf{a}_{t-1}) = \prod_{i=1}^{t} p(\mathbf{z}_i \mid \mathbf{x}_i, \mathbf{a}_{i-1}), \tag{2.3}$$

y que la probabilidad de la observación depende del estado y no de la acción previa

$$p(\mathbf{z}_i \mid \mathbf{x}_i, \mathbf{a}_i) = p(\mathbf{z}_i \mid \mathbf{x}_i). \tag{2.4}$$

Así,

$$p(\mathbf{z}_1, \ldots, \mathbf{z}_t \mid \mathbf{x}_1, \ldots, \mathbf{x}_t, \mathbf{a}_1, \ldots, \mathbf{a}_{t-1}) = \prod_{i=1}^{t} p(\mathbf{z}_i \mid \mathbf{x}_i). \tag{2.5}$$

Esta suposición permite formular la función de probabilidad conjunta de todas las observaciones como el producto de la *función de verosimilitud* de cada lectura. La suposición de independencia, a pesar de no ser estrictamente correcta, se aplica normalmente con éxito en muchos trabajos que utilizan estos enfoques (Pearl 1988) y, en concreto, en la construcción incremental de mapas de ocupación del entorno (Moravec 1998; Thrun 1998).

2.5.2 Actualización de la probabilidad a posteriori

Se utiliza la regla de Bayes para calcular la probabilidad a posteriori

$$p(\mathbf{x}_t \mid \mathbf{z}_1, \ldots, \mathbf{z}_t, \mathbf{a}_1, \ldots, \mathbf{a}_{t-1}) = \\ \alpha p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{z}_1, \ldots, \mathbf{z}_{t-1}, \mathbf{a}_1, \ldots, \mathbf{a}_{t-1}) p(\mathbf{x}_t \mid \mathbf{z}_1, \ldots, \mathbf{z}_{t-1}, \mathbf{a}_1, \ldots, \mathbf{a}_{t-1}) \quad (2.6)$$

Esto es, la probabilidad a posteriori puede expresarse como la verosimilitud de la última lectura, ponderada por la función de probabilidad a priori de la posición del robot.

La verosimilitud de la última lectura depende únicamente de la posición actual del robot

$$p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{z}_1, \ldots, \mathbf{z}_{t-1}, \mathbf{a}_1, \ldots, \mathbf{a}_{t-1}) = p(\mathbf{z}_t \mid \mathbf{x}_t). \quad (2.7)$$

Este término se denomina normalmente *modelo de observación* o *modelo del sensor*. Hay que hacer notar que en la expresión de probabilidad hay implícita una variable que define el mapa del entorno, ya que las observaciones dependen tanto de la posición del robot como del entorno en el que éste evoluciona. Esta variable se hace explícita en aquellos enfoques que pretenden realizar una estimación del mapa del entorno a partir de las lecturas realizadas por el robot (Thrun 1998; Koenig y Simmons 1996), tal y como haremos en la sección 2.6, utilizándose entonces la expresión

$$p(\mathbf{z}_t \mid \mathbf{x}_t, \phi),$$

donde ϕ es la variable que define el modelo del entorno.

El segundo término de la ecuación 2.6 describe la estimación a priori de la localización $\mathbf{x}_t$ inmediatamente *después* de la acción $\mathbf{a}_{t-1}$ y *antes* de realizar la observación $\mathbf{z}_t$. El modelo dinámico (ecuación 2.2) permite expresar esta densidad como

$$p(\mathbf{x}_t \mid \mathbf{z}_1, \ldots, \mathbf{z}_{t-1}, \mathbf{a}_1, \ldots, \mathbf{a}_{t-1}) = \\ \int_{\mathbf{x}_{t-1}} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{a}_{t-1}) p(\mathbf{x}_{t-1} \mid \mathbf{z}_1, \ldots, \mathbf{z}_{t-1}, \mathbf{a}_1, \ldots, \mathbf{a}_{t-2}). \quad (2.8)$$

Si observamos el último término de la ecuación anterior,

$$p(\mathbf{x}_{t-1} \mid \mathbf{z}_1, \ldots, \mathbf{z}_{t-1}, \mathbf{a}_1, \ldots, \mathbf{a}_{t-2})$$

podremos comprobar que representa la estimación a posterior anterior, por lo que podremos formular la ecuación 2.6 como

$$p(\mathbf{x}_t) = \alpha p(\mathbf{z}_t \mid \mathbf{x}_t) \int_{\mathbf{x}_{t-1}} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{a}_{t-1}) p(\mathbf{x}_{t-1}), \tag{2.9}$$

que proporciona una expresión recursiva de la estimación a posteriori de la localización del robot. El factor α es un factor de normalización que asegura que $\int_{\mathbf{x}_t} p(\mathbf{x}_t) = 1$.

La expresión anterior es una expresión genérica que se aplica a cualquier implementación concreta de la localización bayesiana, obtenida mediante la definición de un modelo de observación y de movimiento específicos.

Pero, sobre todo, lo que distingue los distintos enfoques de localización bayesiana es el método escogido para estimar computacionalmente la función de densidad anterior. Revisamos en el siguiente apartado los métodos más utilizados hasta el momento.

2.5.3 Estimación de la función de densidad a posteriori

En la sección anterior se ha derivado la expresión matemática de la función de densidad a posteriori. Revisaremos en esta sección las distintas técnicas propuestas para su estimación computacional.

Entre las más extendidas se incluyen: (1) la suposición de que $p(\mathbf{x})$ es una función normal y la estimación de sus parámetros (mediante el filtro de Kalman, (2) la discretización del espacio de la variable aleatoria $\mathbf{x}$ (métodos basados en la estimación de rejillas de probabilidad) y (3) la consideración de que $\mathbf{x}$ toma valores discretos (métodos topológicos).

Frente a estos enfoques proponemos en la tesis la utilización de *filtros de partículas* que representan la función de densidad mediante un conjunto de muestras.

La localización bayesiana mediante el filtro de Kalman ya ha sido comentada previamente. Revisamos a continuación los enfoques de rejillas de probabilidad y de métodos topológicos. En el capítulo 5 presentaremos nuestra propuesta de estimación bayesiana mediante un filtro de partículas.

Rejillas de probabilidad

Frente a la propuesta anterior, las rejillas de probabilidad (Burgard, Fox, Henning, y Schmidt 1996; Thrun, Burgard, y Fox 1998) permiten representar y actualizar funciones de probabilidad arbitraria. Para ello discretizan con una resolución fina todo el espacio de posibles localizaciones $\mathbf{x}$ del robot y formulan las funciones de densidad de la ecuación 2.9 como funciones constantes en los intervalos correspondientes a la discretización. De esta forma, la actualización de la función de densidad se completa iterando por todos los posibles valores discretos.

Es interesante recoger aquí el proceso de actualización de la función de densidad a posteriori, dada la similitud que tendrá el mismo con el método basado en muestreo que

propondremos más adelante. La formulación está basada en los trabajos de Burgard (Burgard, Fox, Henning, y Schmidt 1996; Burgard, Cremers, Fox, Hahnel, Lakemeyer, Schulz, Steiner, y Thrun 1998) y aparece, en forma de algoritmo, en la tabla 2.1. En el algoritmo, se utiliza la notación $P(\mathbf{x}_i)$ para referirse a la estimación de la probabilidad para la celdilla i del espacio paramétrico $\mathbf{X}$ de posibles configuraciones del robot.

Algoritmo: *Localización bayesiana con rejilla de probabilidad*

1. **Inicialización**

   Para cada celdilla $\mathbf{x}_i \in \mathbf{X}$

   $$P(\mathbf{x}_i) \leftarrow P_0(\mathbf{x}_i)$$

2. **Actualización de la acción a**

   Para cada celdilla $\mathbf{x}_i \in \mathbf{X}$

   $$P(\mathbf{x}_i) \leftarrow \sum_{\mathbf{x}_j} P(\mathbf{x}_i \mid \mathbf{x}_j, \mathbf{a}) P(\mathbf{x}_j)$$

3. **Actualización de la lectura z**

   Para cada celdilla $\mathbf{x}_i \in \mathbf{X}$

   $$\begin{aligned} P(\mathbf{x}_i) &\leftarrow P(\mathbf{z} \mid \mathbf{x}_i) P(\mathbf{x}_i) \\ P(\mathbf{x}_i) &\leftarrow \frac{P(\mathbf{x}_i)}{\sum_{\mathbf{x}_j} P(\mathbf{x}_j)} \qquad \text{(normalización)} \end{aligned}$$

4. **Saltar a 2**.

Tabla 2.1: Algoritmo de localización basado en rejillas de probabilidad.

Los métodos basados en este enfoque han demostrado su potencia en aplicaciones en robots reales (Burgard, Cremers, Fox, Hahnel, Lakemeyer, Schulz, Steiner, y Thrun 1998; Koenig y Simmons 1998), pero tienen ciertos problemas, entre los que se pueden citar la complejidad computacional y la necesidad de definir a priori el tamaño de la discretización del espacio de parámetros y, por ello, su precisión.

Métodos topológicos

Los métodos topológicos (Nourbakhsh, Powers, y Birchfeild 1995; Kaelbling, Cassandra, y Kurien 1996; Koenig y Simmons 1998) definen un espacio discreto de estados para el robot, distinto del espacio de configuraciones (x, y, θ). Este espacio discreto suele ser de un grano muy grueso (*pasillo,unión,final de pasillo*), en contraste con el grano fino usado en el enfoque anterior. Por ejemplo, Nourbakhsh (Nourbakhsh, Powers, y Birchfeild 1995) utiliza nodos topológicos que representan pasillos o uniones.

El grano grueso mejora la complejidad computacional del método anterior. Sin embargo, no se garantiza una localización precisa del robot y se producen con frecuencia errores de confusión de estados, debidos a la ausencia de información métrica en los nodos.

2.6 Fundamentos del mapeado bayesiano

Al igual que en el apartado de localización bayesiana (sección 2.5), denotamos por $(\mathbf{x}_1, \ldots, \mathbf{x}_T)$ la secuencia de variables aleatorias que definen el estado del robot móvil en sucesivos instantes de tiempo. En cada instante $t \leq T$ el robot ha realizado una observación $\mathbf{z}_t$ y ejecutado una acción $\mathbf{a}_t$. Llamaremos D a la secuencia de observaciones y acciones obtenidas

$$D = (\mathbf{z}_1, \mathbf{a}_1, \mathbf{z}_2, \mathbf{a}_2, \ldots, \mathbf{a}_{T-1}, \mathbf{z}_T). \tag{2.10}$$

La utilización de la variable T mayúscula hace notar que el número de observaciones es constante. El proceso de mapeado se aplica una vez que el robot ha realizado la exploración del entorno, y ha almacenado las observaciones realizadas. Todas ellas se van a utilizar para estimar los mejores parámetros que definen el mapa del entorno. Estos parámetros se representan con una variable aleatoria, ϕ, cuyos valores dependen del enfoque utilizado. Por ejemplo, en el trabajo de Thrun (Thrun, Burgard, y Fox 1998), ϕ es una asignación de coordenadas cartesianas (x, y) a un conjunto de *landmarks* que el robot ha ido registrando mientras navegaba. En la propuesta de Koenig (Koenig y Simmons 1996), se estima la distancia entre los nodos de un mapa topológico construido a priori. En nuestra propuesta, la variable ϕ representa un conjunto de parámetros utilizados en la definición de las coordenadas de los vértices del modelo poligonal construido a priori.

Siguiendo el enfoque bayesiano, se debe encontrar el mapa ϕ más probable dada la secuencia de datos observada D, esto es, el mapa *máximo a posteriori* (MAP). Aplicando la regla de Bayes, el MAP es el valor de ϕ que cumple

$$\begin{aligned} \phi^* &= \arg\max_{\phi} p(\phi \mid D) = \\ &= \arg\max_{\phi} p(D \mid \phi)\, p(\phi). \end{aligned} \tag{2.11}$$

El término $p(D \mid \phi)$ define la *verosimilitud* de la secuencia de datos dado el mapa ϕ, y el término $p(\phi)$ define la probabilidad a priori de ϕ. En muchos enfoques se supone que la probabilidad a priori de ϕ es uniforme. En este caso, podemos simplificar la ecuación anterior y considerar sólo el término $p(D \mid \phi)$ para encontrar el mapa más probable. El valor de ϕ que maximiza este término se denomina valor de *máxima verosimilitud* (MV)

$$\phi^* = \arg\max_{\phi} p(D \mid \phi). \tag{2.12}$$

Desarrollando el término de verosimilitud de la ecuación anterior, podemos incorporar en el mismo la secuencia de posiciones $(\mathbf{x}_1, \ldots, \mathbf{x}_T)$ en las que se ha encontrado el robot en los sucesivos instantes de tiempo. Estas variables se denominan *variables ocultas* en la terminología de los Modelos Ocultos de Markov (*Hidden Markov Models*, HMM) ya que el observador no tiene acceso directo a ellas (consultar (Rabiner y Juang 1986) para una revisión sobre los HMM y algoritmos asociados).

Si se conocieran estas posiciones, podríamos expresar el estimador de máxima verosimilitud como

$$\phi^* = \arg\max_{\phi} p(D, \mathbf{x}_1, \ldots, \mathbf{x}_T \mid \phi). \tag{2.13}$$

Aplicando la definición de la probabilidad condicional, se llega a la siguiente expresión de la función de densidad en la ecuación 2.13

$$p(D, \mathbf{x}_1, \ldots, \mathbf{x}_T \mid \phi) = p(D \mid \mathbf{x}_1, \ldots, \mathbf{x}_T, \phi)\, p(\mathbf{x}_1, \ldots, \mathbf{x}_T \mid D), \tag{2.14}$$

Dado que la observación $\mathbf{z}_t$ depende únicamente del mapa ϕ y de la posición del robot en el instante t, $\mathbf{x}_t$, y suponiendo independencia entre las observaciones, el primer término de la ecuación anterior puede reescribirse como

$$p(D \mid \mathbf{x}_1, \ldots, \mathbf{x}_T, \phi) = \prod_{t=1}^{T} p(\mathbf{z}_t \mid \mathbf{x}_t, \phi). \tag{2.15}$$

La localización del robot en el instante t, $\mathbf{x}_t$, depende únicamente de su localización en el instante $t-1$, $\mathbf{x}_{t-1}$, y de la acción $\mathbf{a}_t$ realizada por el robot en ese instante,

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_T \mid D) = p(\mathbf{x}_1) \prod_{t=2}^{T} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{a}_{t-1}). \tag{2.16}$$

El término $p(\mathbf{x}_1)$ determina la distribución a priori de la localización del robot en el primer instante de tiempo.

Sustituyendo las ecuaciones 2.15 y 2.16 en la ecuación principal 2.13 se llega a la formulación final del mapa de máxima verosimilitud

$$\begin{aligned}\phi^* &= \arg\max_{\phi} p(D, \mathbf{x}_1, \ldots, \mathbf{x}_T \mid \phi) = \\ &= \arg\max_{\phi} p(\mathbf{x}_1) \prod_{t=1}^{T} p(\mathbf{z}_t \mid \mathbf{x}_t, \phi) \prod_{t=2}^{T} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{a}_{t-1}).\end{aligned} \tag{2.17}$$

La expresión final, al igual que la expresión que determina la localización (2.9), depende únicamente del modelo de observación, $p(\mathbf{z}_i \mid \mathbf{x}_i, \phi)$, y del modelo dinámico del robot, $p(\mathbf{x}_i \mid \mathbf{x}_{i-1}, \mathbf{a}_{i-1})$.

El problema fundamental de la expresión 2.17 es que no se conoce el valor de las posiciones $(\mathbf{x}_1, \ldots, \mathbf{x}_T)$. Una solución, claramente ineficiente, sería integrar todos los posibles valores de estas variables, de forma que se calculara

$$\arg\max_{\phi} \int_{\mathbf{x}_1} \ldots \int_{\mathbf{x}_T} p(\mathbf{x}_1) \prod_{t=1}^{T} p(\mathbf{z}_t \mid \mathbf{x}_t, \phi) \prod_{t=2}^{T} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{a}_{t-1}). \tag{2.18}$$

Existe, afortunadamente, una técnica que realiza un descenso por gradiente en el espacio de verosimilitud: el método EM (Dempster, Laird, y Rubin 1977), que, en el contexto de los HMM se denomina algoritmo Baum-Welch (Rabiner y Juang 1986) (para aplicaciones del EM en el contexto del mapeado, consultar (Koenig y Simmons 1996; Shatkay y Kaelbling 1997; Thrun, Burgard, y Fox 1998)). Un algoritmo EM realiza un descenso por gradiente en el espacio de verosimilitud alternando dos pasos, un *paso de estimación* (*expection*) (paso E) y un *paso de maximización* (paso M). En el paso E, se estiman los valores más probables de las localizaciones del robot basándose en el mejor valor del mapa obtenido hasta el momento (en la primera iteración no hay ninguno). En el paso M se estima un mapa de máxima verosimilitud a partir de las localizaciones estimadas en el paso E. El paso E puede interpretarse como un procedimiento de localización dado un mapa fijo, mientras que el paso M implementa un proceso de mapeado bajo la suposición de que la localización del robot es conocida. La aplicación iterativa de ambos pasos conduce a un refinamiento sucesivo tanto de las posiciones estimadas como del mapa.

Un algoritmo EM de mapeado debe proporcionar implementaciones del paso E y del paso M. Dependiendo del modelo del entorno y de las funciones de densidad, será más o menos directo implementar ambos pasos. En nuestro caso, al ser ϕ un modelo paramétrico, y la función de verosimilitud una función multimodal no representable mediante una distribución normal, no es posible llegar a una solución cerrada de ninguno de los pasos. En el capítulo 7 proponemos un algoritmo estocástico que sigue la filosofía del EM para buscar el mapa de máxima verosimilitud.

2.7 Discusión

Se han presentado en este capítulo los problemas de la localización y el mapeado, que constituyen los problemas centrales de la tesis. Se han revisado los distintos enfoques y propuestas existentes en la literatura para tratar ambas cuestiones, haciendo un énfasis especial en los distintos modelos de entorno, en las propuestas de localización global y local y en los enfoques para resolver el problema del mapeado.

Se presenta una formalización de ambos problemas utilizando la teoría de estimación bayesiana. Esta formalización unifica las propuestas existentes y proporciona un marco general en el que se pueden formular muchas de las técnicas propuestas.

Capítulo 3

Un modelo estocástico del sonar

Un buen modelo del sonar proporciona una estimación correcta de las lecturas reales que el sensor realizaría dado un entorno conocido. Esta estimación nos va a permitir formular una función de verosimilitud, con la que, dado un modelo del entorno, unos datos leídos y una posición del robot, sea posible determinar la probabilidad de que los datos hayan sido realmente percibidos en ese entorno y en esa posición.

La función de verosimilitud de las lecturas del sonar será el elemento clave de los algoritmos de localización y mapeado.

3.1 Introducción

La palabra sonar deriva del inglés *SOund NAvigation and Ranging*, y se suele utilizar para denominar dispositivos que detectan y localizan objetos submarinos mediante la emisión de ondas de sonido y el cálculo del tiempo de recepción del eco rebotado. Por extensión, reciben el mismo nombre los *sensores de distancia* de uso en robots móviles, basados en la emisión de pulsos de ultrasonidos y en la medición de la distancia de los obstáculos por el tiempo de recepción del eco.

A pesar del alto nivel de ruido existente en sus lecturas, los sonares se han convertido en el sensor de distancia típico de los robots móviles. Se ha utilizado el sonar, por ejemplo, para detectar obstáculos (Arkin 1989; Borenstein y Korem 1991), construir mapas de ocupación del entorno (Elfes 1989) o localizar la posición del robot en un entorno previamente conocido (Drumheller 1987).

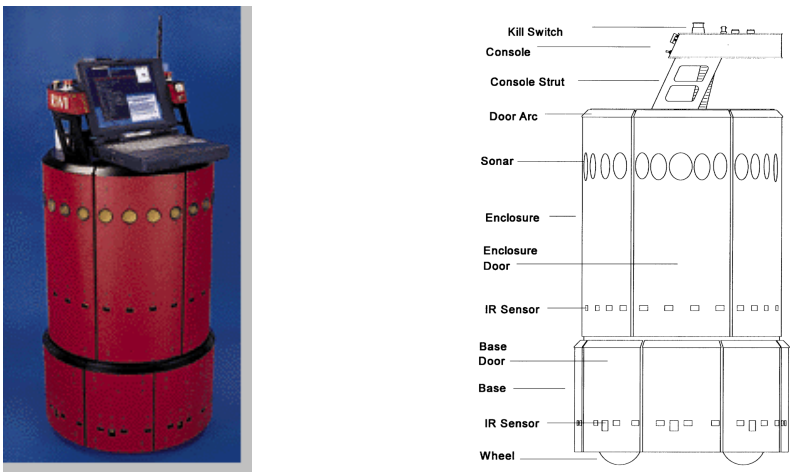
El alto nivel de incertidumbre de las lecturas del sonar aconseja tratarlas mediante alguna técnica basada en modelos probabilísticos bayesianos (Fukunaga 1990) (incluyendo el *filtro de Kalman* y sus variantes (Dean y Wellman 1991)). De hecho, este tipo de enfoques se han aplicado al sonar desde los comienzos de la investigación en robótica móvil (Moravec y Elfes

1985; Matthies y Elfes 1988), aunque es en la actualidad cuando comienza a reconocerse plenamente su utilidad (Koenig y Simmons 1998; Fox, Burgard, Thrun, y Cremers 1998b; Thrun 1998).

En los enfoques bayesianos, como el propuesto en esta tesis, tiene una importancia fundamental realizar un cálculo correcto de la verosimilitud, y para ello es indispensable un buen modelo del sensor.

Este trabajo propone una función multimodal de verosimilitud de un sonar genérico basada en un modelo novedoso y realista del sensor que incorpora características no contempladas hasta el momento. El modelo parte de la formulación de Barshan (Barshan y Kuc 1990), a la que añadimos la utilización de una técnica equivalente al trazado de rayos (Watt y Watt 1992) para simular todas las posibles trayectorias de los haces de ultrasonidos emitidos por el transductor. De esta forma es posible simular lecturas que, hasta el momento, eran despreciadas como errores del sonar y, sin embargo, pueden ser modeladas correctamente mediante *dobles rebotes*.

Todas las medidas y experimentos presentado en este trabajo han sido realizados con el anillo de 24 sonares de PIXIE (ver figura 3.1).



(a) *(b)*

Figura 3.1: PIXIE, robot móvil RWI B-21 con el que se ha realizado la experimentación: (a) fotografía, (b) esquema mostrando sus elementos.

En el apartado 2 describiremos las características principales de los transductores de ultrasonidos, presentando datos reales del anillo de sonares. En el apartado 3 se propone

un modelo empírico de interacción entre un haz de ultrasonidos y el entorno. Este modelo se utiliza como base del algoritmo de trazado de rayos especificado en el apartado 4. En este apartado también se detallan los distintos parámetros que permiten ajustar el modelo a las lecturas reales, y se estudian los resultados obtenidos por el algoritmo de simulación, examinándose cómo varían dichos resultados en función de los valores dados a parámetros del modelo. Por último se comparan dichos resultados con lecturas reales y se realiza el ajuste de los parámetros a los datos reales. En el último apartado se formula la verosimilitud multimodal de las lecturas de un sonar y de un anillo basándose en el modelo planteado.

3.2 Características del sonar

Veremos en este apartado las características de los sensores de ultrasonidos Polaroid y explicaremos el funcionamiento usual de los anillos de sonares usados habitualmente en los robots móviles. Enumeraremos los distintos problemas inherentes a los sensores de ultrasonidos de este tipo y, por último, presentaremos datos experimentales de lecturas realizadas en distintos entornos y condiciones.

3.2.1 Funcionamiento del transductor de ultrasonidos Polaroid

Un transductor Polaroid (Polaroid Corp. 1984) tiene una forma circular, con un diámetro de unos 5 cm. Una fotografía de un sensor de este tipo aparece en la figura 3.2.

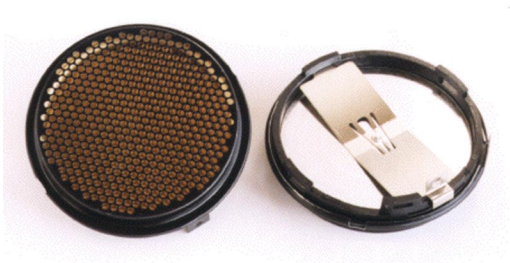


Figura 3.2: Fotografía de uno de los transductores Polaroid con los que se ha realizado la experimentación.

El transductor actúa como emisor y como receptor de señales de ultrasonidos. En su funcionamiento como emisor, envía frontalmente un breve tren de pulsos de ultrasonidos de unos 50 kHz. con el perfil de intensidad que aparece en la figura 3.3. Este perfil determina un haz principal de sonido en forma de cono con una extensión angular de unos 15° a ambos

lados del eje central en el que está orientado el transductor. A mayor distancia angular la intensidad del sonido decae exponencialmente.

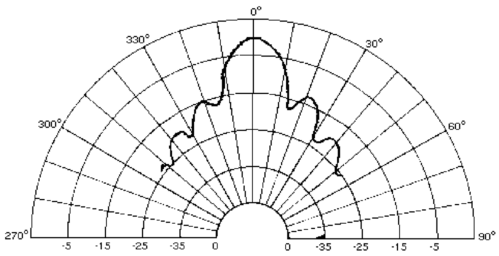


Figura 3.3: Patrón de intensidad de emisión de ultrasonidos (en dB) en función de la distancia angular al eje de orientación del transductor.

La duración del tren de ultrasonidos suele ser de alrededor de 1 ms. Después de haber emitido los pulsos, el sensor pasa a modo receptor, esperando los ecos procedentes de los obstáculos. Un amplificador va aumentando de forma calculada su ganancia para eliminar el efecto de atenuación del sonido con la distancia. Los ecos recibidos (ver figura 3.4) se filtran mediante un sencillo algoritmo de umbralización, y aquellos que sobrepasen el umbral de intensidad especificado se interpretan como ecos procedentes de un obstáculo. Tanto el umbral de intensidad como la ganancia se modifican manualmente en el proceso de calibración del sensor. Debido a ello es bastante probable que existan variaciones importantes en las lecturas proporcionadas por distintos sonares.

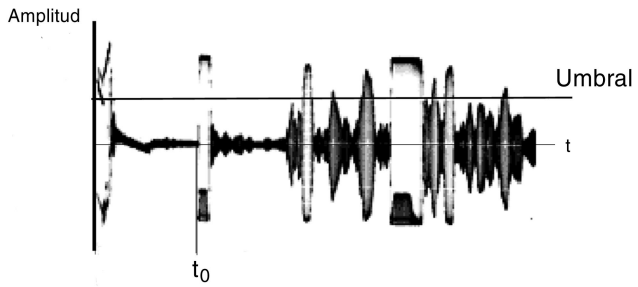


Figura 3.4: Eco de un pulso de ultrasonidos con múltiples reflexiones.

Una vez detectado un eco se calcula la distancia al objeto que lo ha producido mediante el tiempo transcurrido (*TOF* – **T**ime **O**f **F**light–) desde la emisión del pulso hasta la recepción

del eco. En la figura 3.4 esto sucede, para el primer obstáculo, en el instante marcado como t_0. A partir del TOF se obtiene, mediante la siguiente fórmula, la distancia R_0 a la que está situada el obstáculo.

$$R_0 = \frac{ct_0}{2}, \tag{3.1}$$

donde c es la velocidad del sonido en el aire [1].

El sensor puede funcionar detectando el obstáculo más cercano o bien puede devolver los *TOF* de las lecturas que superan el umbral en un determinado intervalo de tiempo después de emitido el pulso. Todos los experimentos se han realizado con el sensor funcionando en el primer modo. Si, después de un intervalo de tiempo predefinido, no se detecta ningún eco, el sensor devuelve un valor máximo arbitrario.

3.2.2 Errores de medida en las lecturas del sensor de ultrasonidos

En muchas ocasiones las lecturas realizadas por un sensor de ultrasonidos contienen errores debido fundamentalmente a dos factores: la extensión angular del haz de sonido y la inclinación de los obstáculos frente al sensor.

Estos dos factores producen tres tipos fundamentales de errores, que enumeramos a continuación.

1. **No recepción del eco por un ángulo de incidencia demasiado grande**.

   Como se representa en la figura 3.5 (a), un obstáculo demasiado inclinado con respecto al eje principal del sensor hace que el eco del pulso de sonido se pierda y no sea recogido por el transductor. Las inclinaciones a partir de las que este error comienza a producirse dependen principalmente de la extensión angular del haz. Cuando se produce este error, el sensor detecta un espacio vacío frente a él.

2. **Incertidumbre de la distancia debido a la extensión angular del haz**.

   Cuando el ángulo entre el transductor y el obstáculo no es tan grande como para que se produzca el error anterior, se produce otro error de medición debido a que el eco devuelto por el obstáculo no es el procedente de la zona central del cono de sonido, sino de una zona periférica del mismo. En la figura 3.5 (b), se puede observar que el primer eco que llega al transductor es el procedente del lado izquierdo del haz, realizándose una medida de la pared menor de la existente en realidad.

3. **Dobles rebotes**.

[1] 340 m/s.

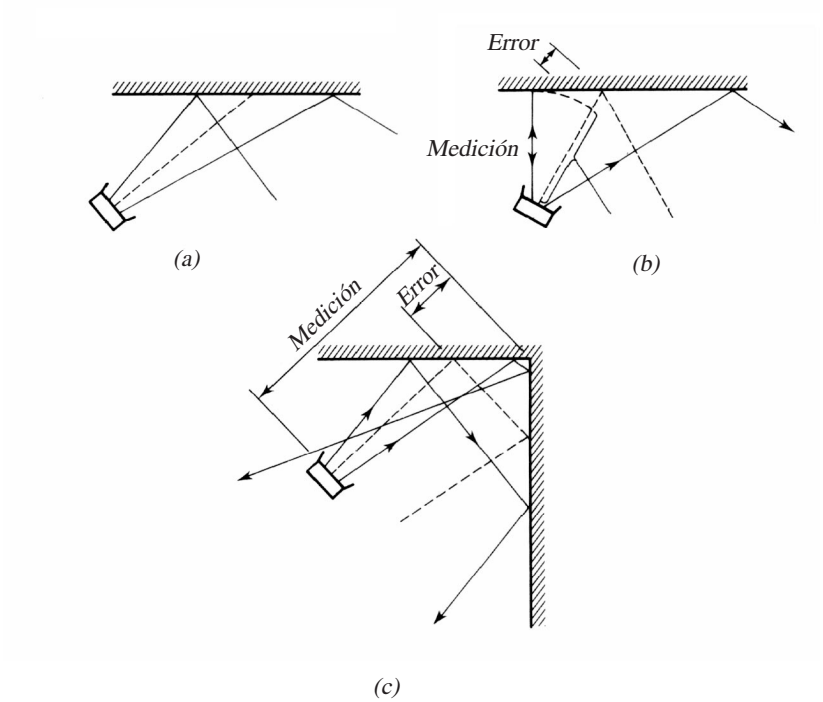


Figura 3.5: Tres tipos de errores de medición del sensor del sensor de ultrasonidos. En (a) el eco no se recibe debido a que rebota alejándose del transductor. En (b) se realiza una medida menor de la que existe en realidad debido a la extensión angular del haz de sonido. En (c) se realiza una medida mayor de la que existe en realidad debido a la recepción de un doble rebote producido por una esquina.

En el caso de la primera situación, en la que el eco es reflejado fuera del transductor, si la onda de sonido reflejada incide en otro obstáculo, es posible que éste refleje a su vez el sonido en la dirección del transductor. Este es el caso denominado como de un *doble rebote* y sucede principalmente en lecturas realizadas en configuraciones de esquina (ver figura 3.5 (c)). En este caso la lectura devuelta por el sensor es mayor que la existente en la realidad.

Estos errores de medición no se producen aleatoriamente, sino que se deben a factores que pueden ser cuantificados y modelados (extensión angular del haz, ángulo entre el eje del transductor y el obstáculo frente al transductor y disposición de los obstáculos en el entorno). En las secciones siguientes veremos cómo realizar esta modelización.

3.2.3 Anillo de sonares

Debido a la baja resolución de los sonares, lo normal es que no se utilicen individualmente, sino agrupados en ciertas disposiciones típicas. La más habitual de ellas, sobre todo para robots móviles circulares, es la de anillo con 12 o 24 sonares. Un anillo de 24 sonares determina que cada uno de ellos va a vigilar una zona angular de 15°. Esto es consistente con la resolución de los sensores y garantiza que todo el espacio alrededor del robot estará cubierto.

Un primer problema de los anillos de sonares es la posible existencia de lecturas cruzadas (*crosstalks* en inglés) entre sonares cercanos. Esto es, si disparamos dos transductores cercanos simultáneamente, es posible que los pulsos de sonido de uno de ellos activen el otro. Una manera de evitar los *crosstalks* consiste en activar los sensores en grupos de 4 unidades situadas en posiciones opuestas. De esta forma, después de 6 lecturas se habrá realizado una lectura completa de todo el anillo de 24 sonares. Todo el proceso dura unos 250 ms., dando tiempo a completar hasta 4 lecturas cada segundo. Es posible registrar de forma independiente el instante de tiempo en el que se ha realizado la lectura de cada uno de los sensores, lo que será de gran utilidad cuando se utilicen estas lecturas para realizar un seguimiento de la posición real del robot en movimiento, o de las características del entorno frente a él.

Existen enfoques mucho más sofisticados para temporizar las lecturas de los transductores de un anillo de sonares, como el propuesto por Borenstein y Koren en (Borenstein y Korem 1995). Con este método se consigue mejorar la velocidad de las lecturas entre 5 y 10 veces con respecto al método presentado previamente. Sin embargo, para la experimentación y para las aplicaciones que presentaremos, no hemos necesitado aumentar la velocidad de lectura del anillo.

Otro importante problema de los anillos, que no suele citarse en la literatura, es el problema de la distinta calibración de los sonares que lo componen. Es muy difícil ajustar todos los potenciómetros de todos los transductores para que proporcionen las mismas lecturas. En el siguiente apartado presentamos algunas lecturas realizadas por distintos sonares de un mismo anillo que evidencian el problema. La solución que proponemos pasa por considerar cada uno de los sonares de forma independiente, ajustando el modelo de sensor a cada uno de ellos.

3.2.4 Experimentación

Todos los resultados presentados en esta sección han sido obtenidos a partir de lecturas del anillo de sonares del robot móvil mencionado en la introducción.

En la figura 3.6 se muestran 130 lecturas de un mismo sonar del anillo obtenidas mediante un giro de 360° del robot.

La interpretación de las lecturas de la figura anterior se realiza en la figura 3.7, en donde se puede comprobar los errores ya mencionados propios de los sensores de ultrasonidos. Se han redondeado las lecturas producidas por dobles rebotes.

En la figura 3.8 se muestran las lecturas de distancia obtenidas por el anillo de sonares con el robot moviéndose a lo largo de un pasillo.

Por último, en la figura 3.9 se puede observar el problema de la calibración del anillo. Se puede comprobar que las lecturas tomadas por dos sonares distintos en diferentes posiciones de la habitación son bastante dispares, sobre todo en aquellos casos en las que el ángulo entre el sonar y el obstáculo es bastante pronunciado.

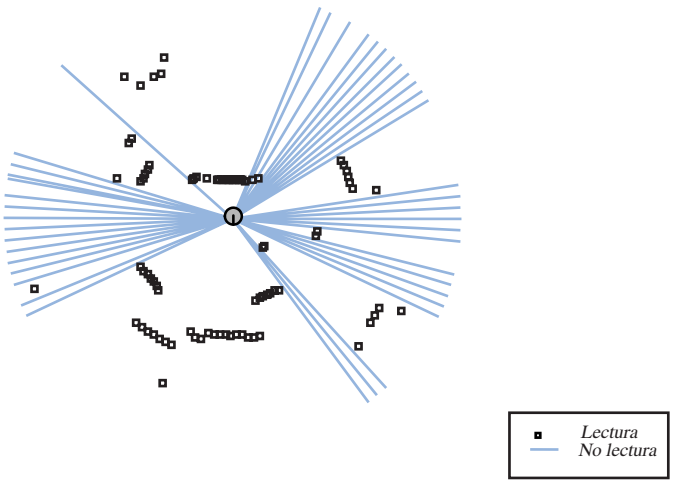


Figura 3.6: 130 lecturas de un sonar del anillo con el robot girando a una velocidad de $5°/s$. Se dibujan como rectas aquellas lecturas que no detectan ningún obstáculo.

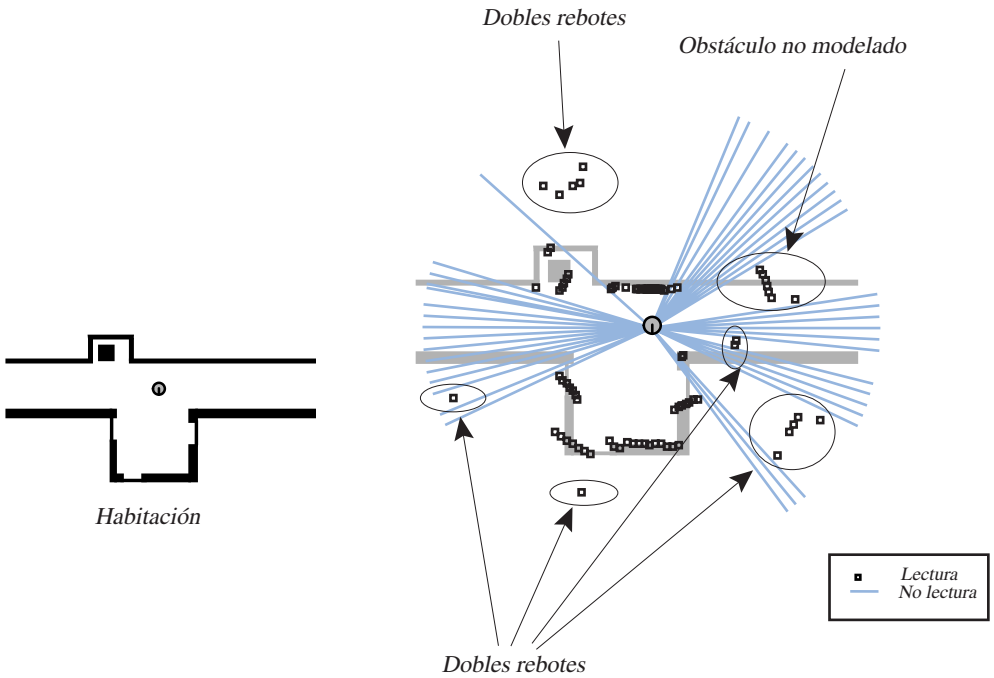


Figura 3.7: Representación de las lecturas de la figura 3.6 sobre la habitación en la que se realizó el experimento.

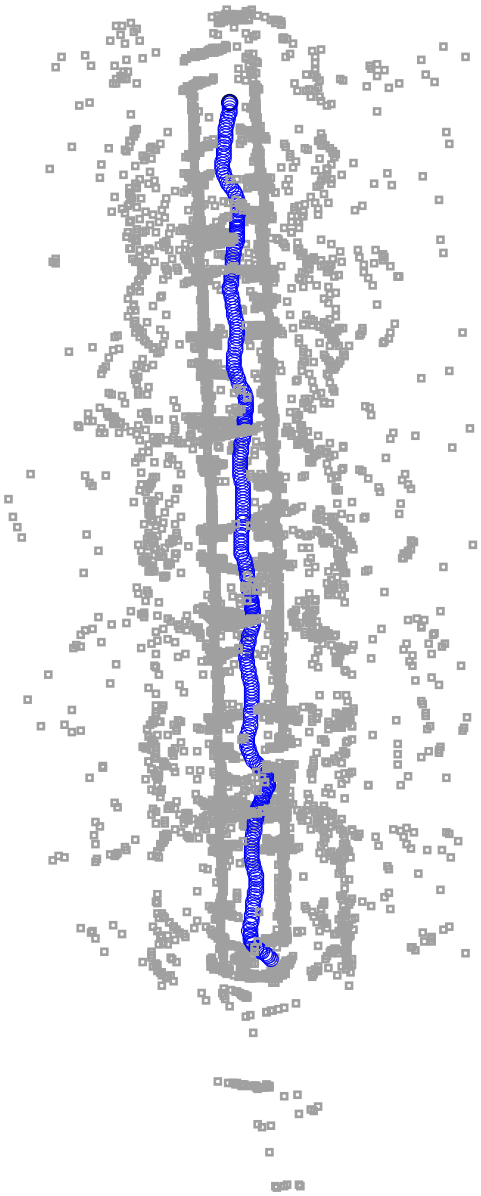


Figura 3.8: Lecturas del anillo de sonares de PIXIE obtenidas con el robot moviéndose a lo largo de un pasillo de 26 metros de largo y 1.6 metros de ancho.

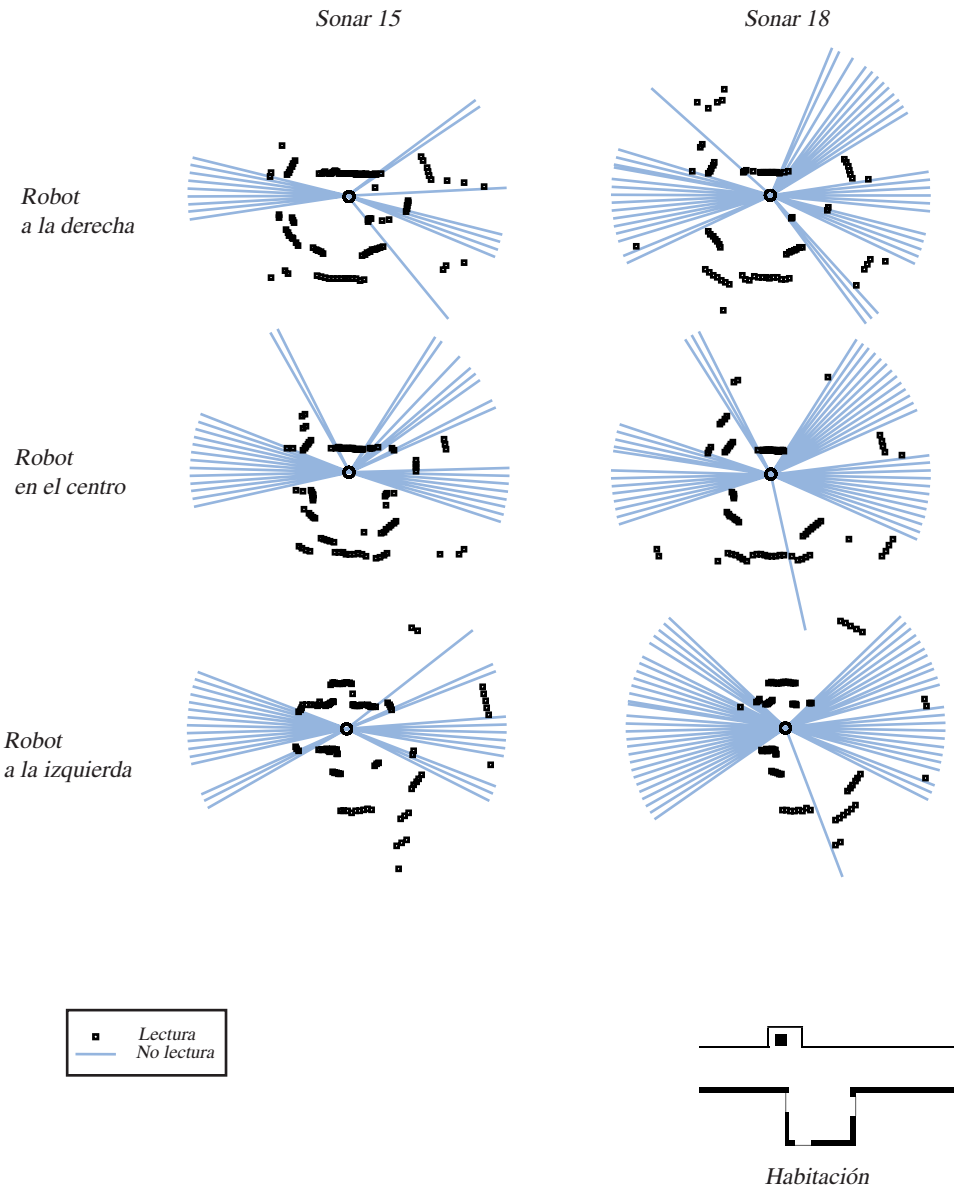


Figura 3.9: Lecturas de los sonares 15 (izquierda) y 18 (derecha) del anillo, con el robot situado en distintas posiciones (de arriba a abajo) de un recinto.

3.3 Modelo de interacción del sensor de ultrasonidos

Kuc (Kuc y Siegel 1987) formula, basándose en conceptos de acústica y teoría de sistemas lineales, un modelo físico que simula la interacción de las ondas del pulso de ultrasonidos con planos y esquinas del entorno. Los resultados son correctos, pero la complejidad computacional del modelo es muy grande.

En la propuesta de Barshan (Barshan y Kuc 1990), de amplia aceptación en la actualidad (Henderson, Bruderlin, Dekhil, Schenkat, y Veigel 1996; Ko, Kim, y Chung 1996; Ayrulu, Barshan, y Utete 1997), se simplifican las ecuaciones anteriores, desarrollándose expresiones analíticas más sencillas y eficientes de computar. Leonard (Leonard y Durrant-Whyte 1992) adapta el modelo de Barshan a diferentes tipos de obstáculos, como son planos, esquinas, aristas y cilindros. El modelo simula el sonar frente a un obstáculo plano, modelando correctamente los problemas de no recepción del eco y de incertidumbre en la distancia, pero dejando sin tratar los efectos de los dobles rebotes. Todos estos modelos tienen un carácter local ya que descomponen el entorno en el que se realiza la simulación en elementos individuales y modelan el comportamiento del sonar con cada uno de los elementos (ver figura 3.10).

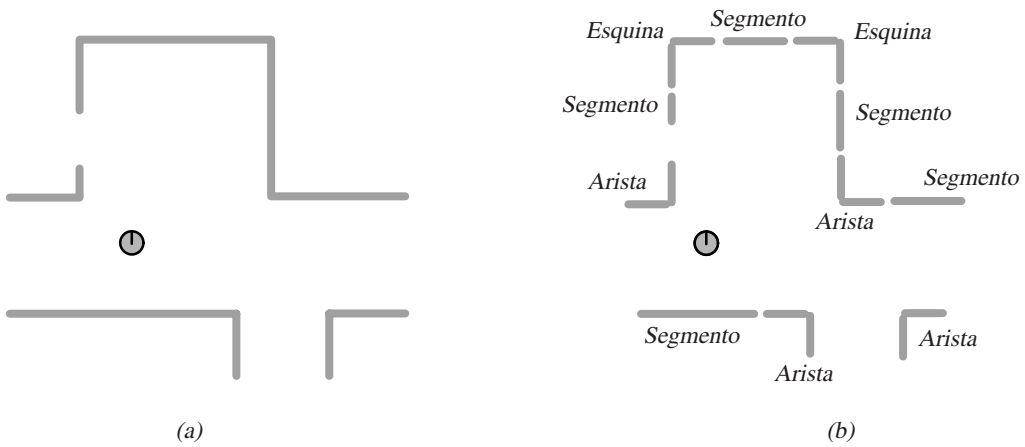


Figura 3.10: Los modelos del sonar de Barshan (Barshan y Kuc 1990) o Leonard (Leonard y Durrant-Whyte 1992) descomponen un entorno (a) en elementos individuales (b) (aristas,esquinas y segmentos) y simulan la interacción del sonar con cada uno de los elementos.

Frente a estos modelos locales existen propuestas globales, procedentes del campo de la acústica, que simulan correctamente el comportamiento del sonido en una sala. En particular, se ha propuesto con éxito la utilización del trazado de rayos (*ray-tracing*) (Watt y Watt 1992)

y de variantes suyas como el *cone tracing* y el *pyramid tracing* para resolver el problema de los rebotes indirectos de los ecos con el entorno (Ondet y Babry 1989; Stephenson 1990; Vian y van Maercke 1986).

Formulamos en este apartado una extensión del modelo de Barshan que, con una implementación eficiente, contempla los rebotes de los ecos con el entorno mediante una variante del algoritmo básico del trazado de rayos.

3.3.1 Modelo físico del sensor de ultrasonidos

Cuando el sonar emite un pulso de sonido, éste se comporta como un haz con forma de cono centrado en el transductor y que tiene como eje de simetría la línea perpendicular a la superficie del sonar. Experimentalmente se comprueba que la amplitud de la presión del sonido en un punto p situado en el extremo del cono varía de forma exponencial en función del ángulo $\triangle\theta$ entre la perpendicular al sonar y la recta que va del centro del sonar a p, formando el típico lóbulo emisor del sonar (figura 3.11).

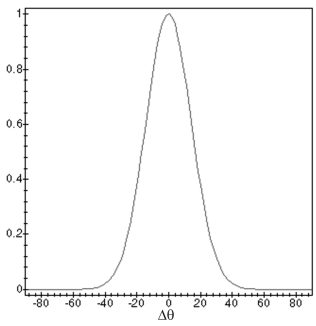
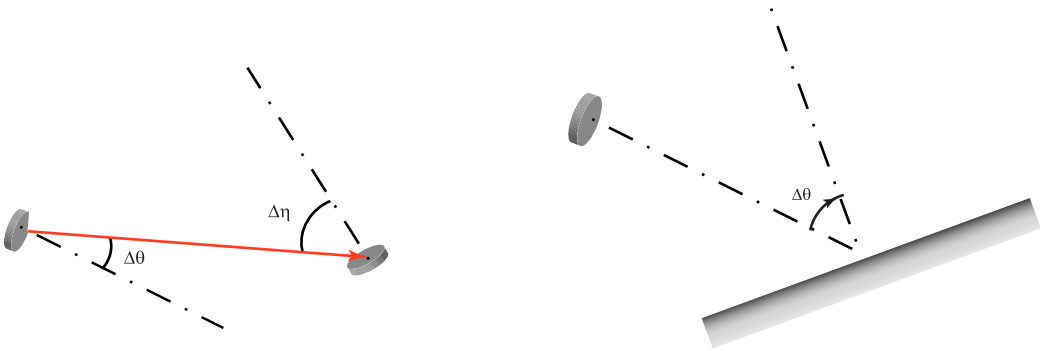


Figura 3.11: Representación de la ecuación $e^{(-2\triangle\theta^2/\theta_0^2)}$ que define la amplitud del haz de ultrasonidos emitido por un transductor en función de la desviación con respecto a la normal del propio transductor.

La presión máxima se obtiene cuando p está situado justo en la perpendicular del sonar y disminuye de forma exponencial conforme aumenta $\triangle\theta$. La ecuación

$$a(\triangle\theta) = a_{\max} e^{(-2\triangle\theta^2/\theta_0^2)} \tag{3.2}$$

modela esta presión, siendo $a_{\max}$, la amplitud máxima observada. Esta ecuación representa una distribución normal con una desviación estándar igual a $\theta_0/2$, siendo θ_0 una constante que



(a) *(b)*

Figura 3.12: (a) Un par de transductores enfrentados. (b) Un transductor enfrentado a una pared con una inclinación $\triangle\theta$.

depende del tipo de emisión del pulso de sonido y que denominamos constante de dispersión del haz.

Para un par de transductores idénticos, uno actuando como emisor y el otro como receptor, la amplitud de la señal detectada se computa multiplicando las dos presiones del pulso:

$$a(\triangle\theta, \triangle\eta) = a_{\max} e^{(-2\triangle\theta^2/\theta_0^2)} e^{(-2\triangle\eta^2/\eta_0^2)}, \tag{3.3}$$

siendo $\triangle\theta$ y $\triangle\eta$ los ángulos de inclinación del transductor emisor y receptor respectivamente (ver figura 3.12 (a)), y $a_{\max}$ la amplitud máxima observada, esto es, cuando los transductores están enfrentados directamente y $\triangle\theta = \triangle\eta = 0$.

En el modelo propuesto por Barashan y Kuc se supone que el entorno está formado por paredes planas que actúan como reflectores especulares del sonido. El transductor actúa al mismo tiempo como emisor y como receptor. Se deriva la siguiente expresión para la amplitud de la señal recibida por el transductor

$$a(\triangle\theta) = a_{\max} e^{(-4\triangle\theta^2/\theta_0^2)}, \tag{3.4}$$

siendo $\triangle\theta$ el ángulo formado por la normal al transductor y la normal al plano de la pared (ver figura 3.12 (b)).

La mayoría de los sonares no proporcionan la amplitud de la señal recibida, sino el TOF del primer rebote. Este se obtiene umbralizando la señal, de forma que se considera que se ha recibido un reflejo de la señal cuando la amplitud recibida supera un determinado umbral a_0. De esta forma, sustituyendo a_0 en la ecuación 3.4, consideramos que un obstáculo se

detecta correctamente cuando el ángulo $\triangle\theta$ que forman la normal al transductor y la normal al plano del obstáculo es menor que un umbral de sensibilidad $\triangle\theta_d$, donde

$$\triangle\theta_d = \frac{\theta_0\sqrt{-\ln(a_0/a_{\max})}}{2} \tag{3.5}$$

Normalizando la ecuación anterior, considerando $a_{\max} = 1$, entonces $0 < a_0 < 1$ y la ecuación anterior queda como sigue

$$\triangle\theta_d = \frac{\theta_0\sqrt{-\ln(a_0)}}{2}, \tag{3.6}$$

con lo que el sector angular de sensibilidad del sonar es (ver figura 3.13)

$$\left[-\frac{\theta_0\sqrt{-\ln(a_0)}}{2}, +\frac{\theta_0\sqrt{-\ln(a_0)}}{2}\right]. \tag{3.7}$$

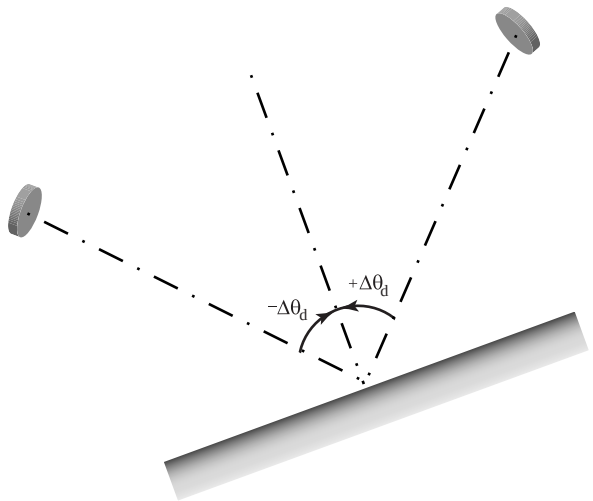


Figura 3.13: Zona de sensibilidad del haz de ultrasonidos.

3.3.2 Extensión del modelo para contemplar múltiples reflexiones

La limitación más importante del modelo anterior es que únicamente contempla la interacción entre un sonar y un único obstáculo, no considerando el sonido indirecto debido a los reflejos del mismo entre los propios obstáculos. Lecturas como las que se marcan en la figura 3.14

(a), no se detectarían como obstáculos con una aplicación directa del modelo de Barshan. Esto es debido a que el ángulo de la superficie en la que incide directamente el sonar supera el umbral angular de sensibilidad, $\triangle\theta_d$. Sin embargo, como se muestra en la figura, el sonar recibe sonido rebotado por esa superficie. Una interpretación plausible de dichas lectura, que se utiliza con éxito en las simulaciones de acústica, es que ese sonido ha llegado al sonar después de varios rebotes con las paredes del entorno, como se muestra en la figura 3.14 (b).

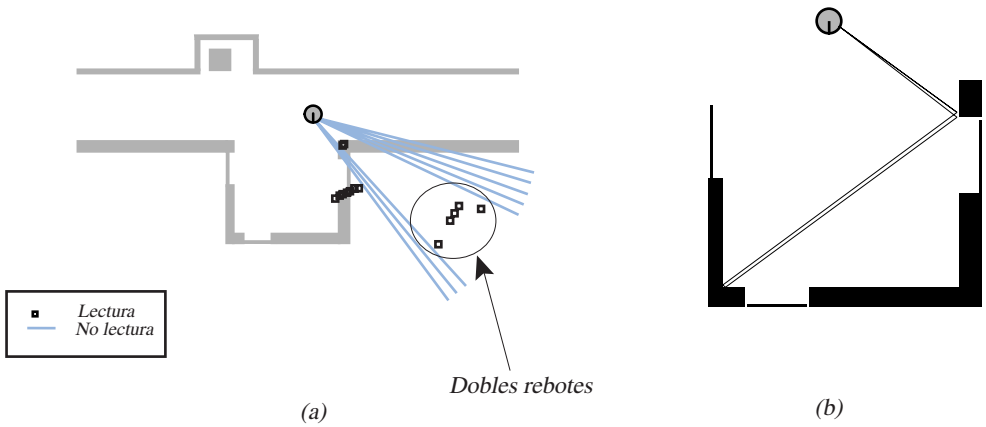


Figura 3.14: (a) Lecturas procedentes de dobles rebotes. (b) Trayectorias del sonido que han producido uno de los dobles rebotes.

A continuación presentamos una extensión del modelo de Barshan que realiza una simulación global del comportamiento del sonar, en la que todos los elementos del entorno interactúan entre sí, reflejándose en las ondas de sonido.

Amplitud

Cuando un pulso de ultrasonidos rebota en una superficie consideramos que su reflejo sigue el mismo patrón definido por la ecuación 3.2, sufriendo una atenuación exponencial conforme el ángulo se aparta del ángulo principal de reflexión. Así, supongamos un rayo puntual con amplitud a_i que incide en una superficie. Consideramos que el reflejo de este rayo se comporta como un haz cuya presión de sonido máxima corresponde al ángulo de reflexión especular y decae exponencialmente según la ecuación

$$a(\triangle\kappa) = a_i e^{(-2\triangle\kappa^2/\kappa_0^2)}, \tag{3.8}$$

siendo $\triangle\kappa$ el incremento con respecto al ángulo de reflexión especular (ver figura 3.15), a_i la presión del rayo incidente y κ_0 la varianza de la distribución de reflexión, que depende principalmente del tipo de material del obstáculo.

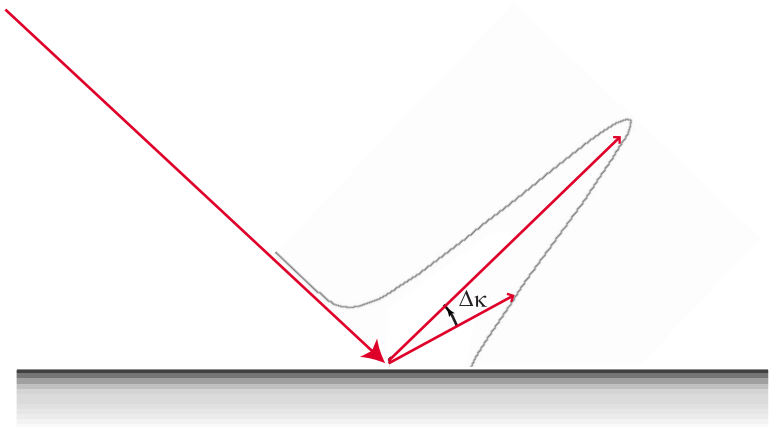


Figura 3.15: El reflejo de un pulso de sonido incidente en un obstáculo con una amplitud a_i genera un haz reflejado cuya amplitud a_r decae exponencialmente alrededor del rayo reflejado ideal según la ecuación $a_r = a_i e^{(-2\triangle\kappa^2/\kappa_0^2)}$, siendo $\triangle\kappa$ el ángulo de desviación con respecto al rayo reflejado ideal.

La amplitud del eco percibido por el sonar, en el caso de un haz que rebota en n superfices, se calcula multiplicando la expresión anterior tantas veces como superficies existan. Al final la señal rebotada será percibida por el sonar si el último haz rebota en su dirección.

Por ejemplo, en la figura 3.16, vemos un rayo que se emite con un ángulo $\triangle\theta$ con respecto a la normal del transductor, que se refleja con una trayectoria definida por los ángulos $\triangle\kappa_1, \triangle\kappa_2$ con respecto a los ángulos de reflexión especular y que incide en el transductor con ángulo $\triangle\eta$.

En general, denotamos por Γ la trayectoria seguida por un rayo emitido por el transductor, con un ángulo de emisión $\triangle\theta$, ángulos de reflexión $\triangle\kappa_1, \ldots, \triangle\kappa_n$ y ángulo de incidencia $\triangle\eta$. La amplitud final de un rayo emitido que sigue la trayectoria Γ se puede calcular siguiendo la idea de Barshan de multiplicar las sucesivas disminuciones de la amplitud causadas por las desviaciones angulares de la trayectoria, lo que nos lleva a la siguiente ecuación.

$$a(\Gamma) = e^{-2(\triangle\theta^2/\theta_0^2 + \triangle\kappa_1^2/\kappa_0^2 + \ldots + \triangle\kappa_n^2/\kappa_0^2 + \triangle\eta^2/\theta_0^2)}. \tag{3.9}$$

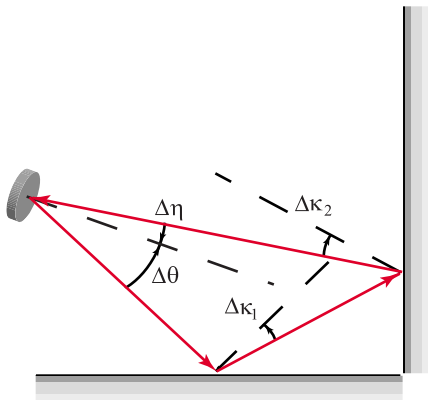


Figura 3.16: Ejemplo de un rayo de ultrasonidos que se refleja en una superficie con un ángulo $\triangle\kappa_1$ con el rayo reflejado ideal y en otra con $\triangle\kappa_2$.

Normalmente se utilizan amplificadores de ganancia para compensar la pérdida de amplitud de la señal debida a la difracción y a la atenuación, por lo que la amplitud del rayo reflejado puede considerarse independiente de la distancia a la que se encuentra el obstáculo y, por tanto, este factor no se considera en la ecuación anterior.

TOF

Para obtener el primer TOF del haz debemos encontrar el rayo que realice el recorrido menor, esto es, el que primero llegue rebotado al transductor, y que cumpla la propiedad de incidir en el sensor con una amplitud mayor que su umbral de sensibilidad, que hemos denominado a_0. Si llamamos D_Γ a la distancia recorrida por un rayo que sigue la trayectoria angular Γ, entonces la expresión que define el rango R_0 medido por nuestro modelo de sonar es la siguiente

$$R_0 = \arg\min_{\Gamma} D_\Gamma/2 \qquad \text{sujeto a } a(\Gamma) > a_0. \tag{3.10}$$

Para simular n sensores de un robot móvil consideraremos que son independientes unos de otros (como de hecho sucede en la realidad) y aplicaremos el modelo en la orientación definida por cada sensor. Será necesario ajustar en cada caso los parámetros del modelo a las características particulares de cada uno de los sonares.

3.4 Simulación del sonar mediante trazado de rayos

La simulación del modelo del sonar se basa en el algoritmo de *trazado de rayos* (consultar (Watt y Watt 1992) para una revisión completa), adaptado para incorporar el tratamiento del TOF.

3.4.1 Trazado de rayos

En la técnica del trazado de rayos, utilizada para construir imágenes sintéticas hiper-realistas, se lanza un conjunto de rayos, uno por cada uno de los elementos de imagen de la pantalla, desde el punto de vista desde el que se toma la imagen hacia la escena.

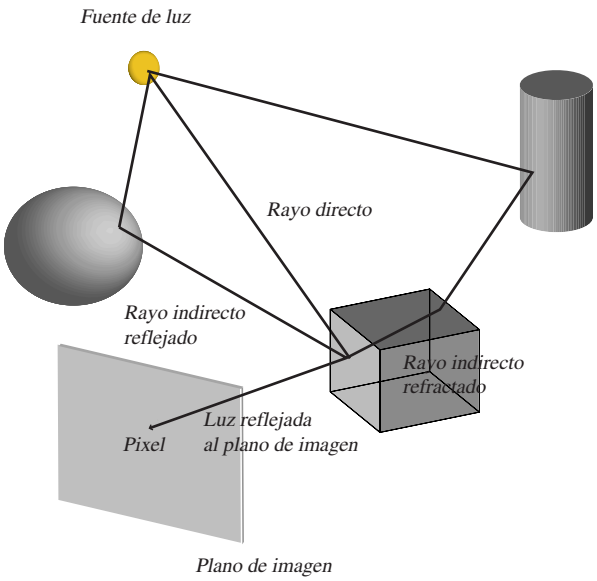


Figura 3.17: Funcionamiento básico del trazado de rayos. El rayo que llega desde la escena es una composición de un rayo reflejado ambiental, un rayo reflejado especular y un rayo refractado.

Para cada rayo se calcula la intersección más cercana con un objeto de la escena y se computa el color del mismo. Para ello se supone que el color del objeto depende de la luz recibida por el mismo, calculándose de forma recursiva las tres posibles formas por las que puede recibir luz el objeto: rayo de luz indirecto reflejado procedente de otro objeto, rayo de luz indirecto refractado si el objeto tiene algún grado de transparencia y rayo directo procedente de la fuente de luz (ver figura 3.17).

```
Algoritmo TrazadoDeRayos
Entrada: PuntoComienzo, Dirección del rayo de la pantalla hacia la escena, Profundidad
Salida: Color resultante

    Si Profundidad > PROFUNDIDAD_MAXIMA devolver COLOR_NEGRO
    Sino
        Calcular el Objeto y el Punto donde intersecta el rayo que va de PuntoComienzo a Dirección
        Si existe Objeto intersectado
            ColorLocal := CalcularLuzDirecta(Objeto,Punto)
            DirecciónReflejada := CalcularReflexión(Objeto,Punto)
            DirecciónRefractada := CalcularRefracción(Objeto,Punto)
            ColorReflejado := TrazadoDeRayos(Punto,DirecciónReflejada,Profundidad +1)
            ColorRefractado := TrazadoDeRayos(Punto,DirecciónRefractada,Profundidad +1)
            devolver Combinar(Objeto,ColorLocal,ColorReflejado,ColorRefractado)
        sino devolver COLOR_NEGRO
```

Tabla 3.1: Algoritmo recursivo base del trazado de rayos

Hay que hacer notar que, aunque el fundamento del modelo es el que se ha enunciado previamente, el trazado de rayos se realiza de atrás hacia adelante, desde el punto de vista hacia la escena, en vez de hacerlo desde la fuente de luz hacia el punto de vista. Esto es porque no estamos interesados en todos los rayos emitidos desde la fuente de luz, sino sólo en un subconjunto reducido de éstos, los que, después de reflejarse en elementos de la escena, terminan pasando a través del plano de imagen.

La forma de implementar el trazado de los rayos es recursiva, utilizándose una estructura de árbol binario como soporte lógico. Cada nodo representa una llamada recursiva del procedimiento general de trazado de rayos, teniendo normalmente una rama producida por el rayo reflejado y otra por el refractado. Si un rayo intersecta un objeto, éste produce otros dos rayos, uno reflejado y otro refractado. En ambos se aplica el algoritmo de trazado de rayos para calcular con qué objetos intersectan y, para cada intersección, volver a dividirlo en una componente reflejada y otra refractada. El proceso continúa hasta que se llega a un nivel de recursión predeterminado o hasta que el rayo no intersecta con ningún objeto, en cuyo caso se le asigna un color de fondo. El color de cada intersección se calcula sumando la componente de luz ambiental debida a la fuente de luz con las componentes debidas a los rayos reflejados y refractados. En la tabla 3.1 se encuentra una versión de alto nivel del algoritmo recursivo del trazado de rayos (adaptado de (Watt y Watt 1992)).

3.4.2 Simulación del sonar

A diferencia del algoritmo del trazado de rayos no se trazarán los rayos hacia atrás, sino de forma inversa (como en la técnica del *backward raytracing* (Glassner 1989)). Esto se debe

a que en el caso del sonar coinciden la fuente y el sensor de sonido, con lo que es indiferente trazar los rayos hacia atrás o hacia adelante.

En la figura 3.18 se puede ver un ejemplo de recorrido de los rayos de sonido según el algoritmo de simulación del sonar.

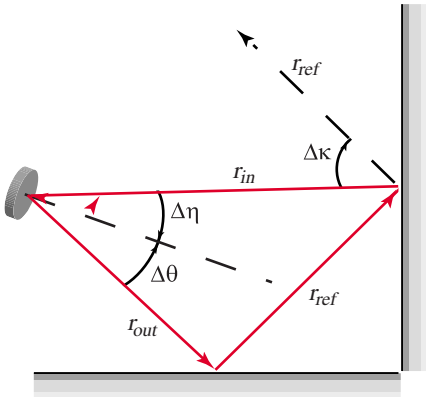


Figura 3.18: Ejemplo del funcionamiento del algoritmo de simulación del sonar.

El algoritmo de simulación (detallado en la tabla 3.2) funciona como sigue.

- Suponemos que el sonar está orientado en la dirección θ. En primer lugar se calcula el umbral de sensibilidad del sonar $\triangle\theta_d$, y se emite un número δ de rayos desde $\theta - \triangle\theta_d$ hasta $\theta + \triangle\theta_d$. Se recorren todos estos rayos emitidos y se calcula el alcance devuelto por cada uno de ellos, devolviendo el menor alcance.

- Para calcular el alcance de un rayo emitido r_{out}, primero se determina el punto de intersección p con el entorno y la dirección de reflejo del rayo en ese punto r_{ref}. A continuación se comprueba si r_{ref} incide en el sensor de ultrasonidos con amplitud suficiente. Para ello se traza el rayo r_{in} del transductor al punto p y se calcula el ángulo $\triangle\kappa$ entre r_{in} y r_{ref}, y el ángulo $\triangle\eta$ entre r_{in} y la orientación del transductor θ. Una vez calculados estos ángulos ya se puede aplicar la ecuación 3.9 para calcular la amplitud del sonido entrante en el transductor. Si ésta es mayor que el umbral, entonces se almacena el rayo y la distancia recorrida en una lista.

- Si la amplitud no es mayor que a_0 comprobamos entonces el siguiente reflejo. Se considera que el pulso se refleja en la dirección del rayo reflejado r_{ref} y se calcula la intersección de éste con el entorno, volviendo a calcular la amplitud del sonido entrante en el transductor debida al rayo reflejado con el método anterior.

Algoritmo Alcance de sonar
Entrada: Ángulo θ en el que se realiza la lectura del sonar
Salida: Alcance del sonar en la orientación θ
Parámetros: θ_0, κ_0, a_0

Sea *ListaRayos* una lista vacía
$\triangle\theta_d := \theta_0\sqrt{\ln(a_0)}/2$
Sean $r_{out_1} \ldots r_{out_n}$ rayos con orientación $\theta_i = \theta - \triangle\theta_d \ldots \theta + \triangle\theta_d$ con incremento δ
Para cada r_{out} con desviación $\triangle\theta_{in} = \theta_i - \theta$
    Inicializar *Profundidad* a 0
    Mientras (*Profundidad* < PROFUNDIDAD_MAXIMA)
        Calcular p el punto de intersección de r_{out} con el objeto más cercano
        Calcular r_{ref} el rayo reflejado por r_{out} en p
        Calcular r_{in} el rayo del transductor a p
        Calcular $\triangle\kappa$ el ángulo entre r_{ref} y r_{out}
        Calcular $\triangle\eta$ el ángulo entre la orientación del transductor y r_{in}
        $a := e^{-2(\triangle\theta^2/\theta_0^2 + \triangle\eta^2/\theta_0^2 + \triangle\kappa^2/\kappa_0^2)}$
        Si $a > a_0$
            Calcular d distancia recorrida por r_{out}
            Añadir (r_{out}, d) a *ListaRayos*
        Sino
            $r_{out} := r_{ref}$
            Profundidad := *Profundidad*+1
    FinMientras
FinParaCada
Devolver el alcance de r_{out} con menor recorrido d en *ListaRayos*

Tabla 3.2: Algoritmo de simulación del sonar

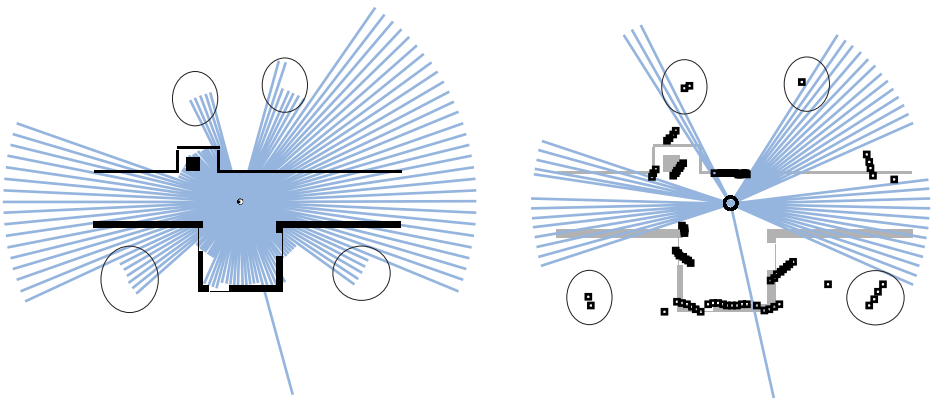
Aquí estamos realizando una importante simplificación para hacer tratable el algoritmo. En el modelo del sonar consideramos que el reflejo de un pulso de sonido es un haz, sin embargo en el algoritmo de simulación esto lo tenemos en cuenta únicamente a efectos del cálculo del rayo entrante emitido hacia el transductor y no para seguir trazando rayos reflejados. El único rayo reflejado que se considera es el principal, el que coincide con la dirección de reflejo.

El trazado del rayo reflejado termina cuando se alcanza un número de rebotes determinado.

- Por último, una vez realizado el trazado de todos los rayos emitidos, se devuelve la menor distancia recorrida. En el caso en el que ningún rayo haya vuelto al sensor, se devuelve un valor arbitrario que indica que no hay rebote.

Como ejemplo de funcionamiento, en la figura 3.19 se muestra la simulación del sonar en un entorno idéntico al recinto en el que se realizaron las mediciones de la figura 3.7. Como dato curioso se acompaña la figura de las lecturas reales, haciendo notar que no se ha realizado ningún proceso de ajuste de los parámetros del modelo.

Resaltamos la correcta simulación de los dobles rebotes, como puede comprobarse en la figura, no conseguida por otros modelos.



Simulación *Lecturas reales*

Figura 3.19: Resultado, a la izquierda, de la simulación de 130 lecturas de sonar en un entorno idéntico al recinto en el que se han realizado las mediciones. A la derecha se muestran los resultados reales de las mediciones. Tanto en la simulación como en los datos reales aparecen rodeadas las lecturas producto de dobles rebotes.

3.4.3 Parámetros del modelo y del algoritmo

Si revisamos el algoritmo y el modelo del sensor, comprobaremos que existen tres parámetros de cuyo valor va a depender el comportamiento del sonar. Se trata de θ_0, κ_0 y a_0. Los dos primeros determinan, respectivamente, la extensión angular del haz emitido por el sonar y del haz de sonido reflejado en un obstáculo. Ambos se comportan igual, cuanto mayores sean, mayor será la extensión angular del haz y mayor sensibilidad (e imprecisión) tendrá el sonar. El último parámetro determina el umbral del transductor. En la figura 3.20 se muestra el resultado de simular 130 lecturas de sonar en el mismo entorno con distinto valores de parámetros.

Hay que hacer notar que, para una lectura con un determinado ángulo, la variación de la medida no es continua con la variación de los parámetros. Por el contrario, se manifiesta en la simulación un *efecto de escalón* que hace que la distancia calculada por un sonar cambie bruscamente frente a una pequeña modificación de los parámetros.

Para demostrar este efecto, se recogieron los resultados de la simulación de un sonar en una configuración de esquina, variando los parámetros θ_0 entre 0.05 y 1.5 y κ_0 entre 0.05 y 1.5. La representación de los resultados se muestra en la figura 3.21. Como se observa, los valores medidos por el sonar simulado son, alternativamente, 100, 210 y 350 centímetros, definiéndose un comportamiento marcadamente multimodal del modelo. Existe también una combinación de parámetros que hace que el sonar no detecte obstáculos. En la simulación el alcance máximo del sonar se fijó en 500 cm.

Modificando estos parámetros es posible ajustar la simulación a los valores reales proporcionados por cada uno de los sonares. Debido a que, como ya mencionamos, el comportamiento de los sonares del anillo es muy dispar se deberá realizar un ajuste de parámetros independiente para cada uno de los sonares. En el siguiente apartado se presenta el proceso de ajuste y los resultados del mismo.

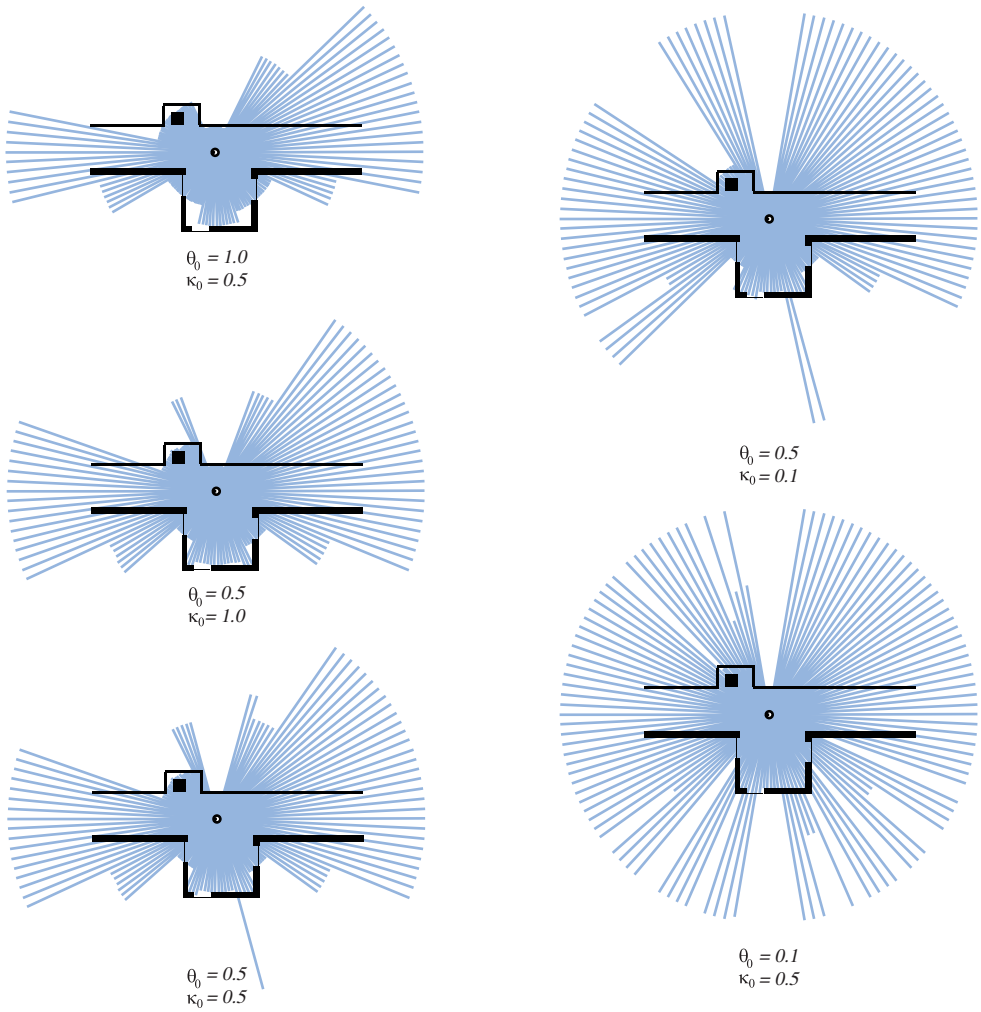


Figura 3.20: Resultado de la simulación de 130 lecturas de sonar en el mismo entorno, variando los valores de los parámetros del algoritmo de simulación.

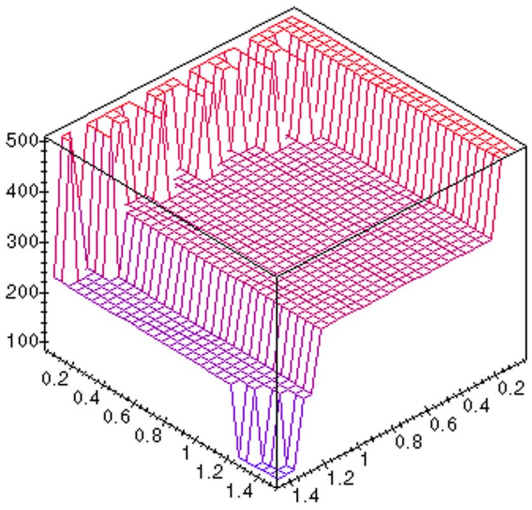


Figura 3.21: Variación de la lectura del sensor para un ángulo determinado cuando se varía θ_0 entre 0.05 y 1.5 y κ_0 entre 0.05 y 1.5.

3.4.4 Ajuste *off-line* de los parámetros del modelo

Para ajustar el modelo debemos encontrar los valores de sus parámetros que optimizan la simulación, haciendo que ésta se acerque lo máximo posible a los valores tomados por el sensor en el mismo entorno.

Formalmente, buscaremos para cada sonar aquellos valores de los parámetros que minimicen la siguiente expresión

$$\chi^2 = \sum_{i=1}^{N} (x_i - f(x_i, \theta_0, \kappa_0, a_0))^2, \tag{3.11}$$

siendo N el número de lecturas realizadas para el ajuste, x_i los valores de alcance medidos en las lecturas y $f(x_i; y)$ el valor medido por el simulador correspondiente a la lectura i con los valores de los parámetros θ_0,κ_0 y a_0.

La ecuación del modelo impide una resolución analítica del ajuste, ya que no es posible calcular las derivadas de la ecuación 3.10. Además, en el apartado anterior se comprobó que el comportamiento del modelo tiene forma de función escalón, debido a que las desviaciones en los parámetros del modelo hacen que una lectura dada varíe entre dos o tres valores muy distintos (100, 210, 350 cm. y *no obstáculo*, por ejemplo, en la figura 3.21) sin tomar valores intermedios. Por ello, tampoco será posible calcular de forma numérica las derivadas de la ecuación 3.11.

Dado que el número de parámetros es pequeño, hemos optado por realizar una búsqueda exhaustiva de los valores de los parámetros, variando todos ellos desde 0.0 hasta 1.0. Si el modelo hubiera tenido un número mayor de parámetros la búsqueda exhaustiva no hubiera sido eficiente, teniendo que recurrir a métodos estocásticos como el *simulated annealing* o la búsqueda genética.

Los resultados obtenidos para cada uno de los sonares se muestran en la tabla 3.3, y un ejemplo de dos sonares ajustados se muestra en la figura 3.22.

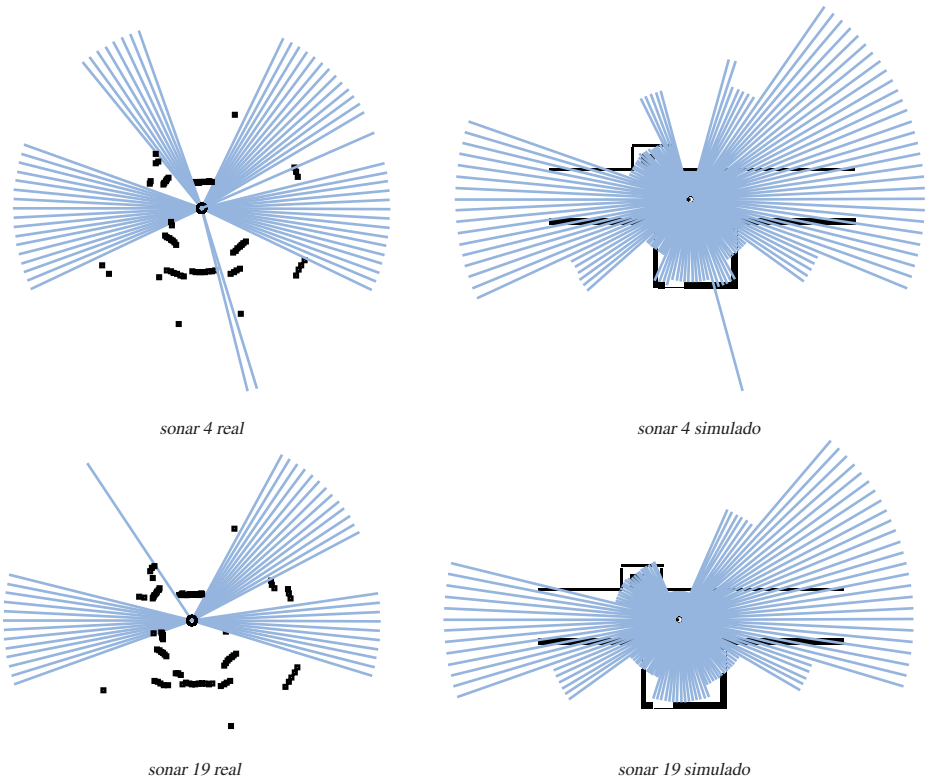


Figura 3.22: Ejemplo del resultado del ajuste de dos sonares, comparando su simulación con las lecturas reales.

Sonar	θ_0	η_0	a_0	*Sonar*	θ_0	η_0	a_0
0	0.65	0.05	0.00	12	0.50	0.50	0.60
1	0.90	0.50	0.60	13	0.50	0.25	0.40
2	0.85	0.05	0.00	14	0.65	0.95	0.80
3	0.70	0.15	0.20	15	0.85	0.20	0.40
4	0.50	0.25	0.40	16	0.25	0.95	0.20
5	0.75	0.25	0.60	17	0.85	0.05	0.00
6	1.00	0.20	0.40	18	0.80	1.00	0.60
7	0.50	0.35	0.40	19	0.90	0.95	0.60
8	0.80	0.10	0.20	20	0.70	0.15	0.20
9	0.70	0.15	0.20	21	0.80	0.55	0.60
10	0.45	0.85	0.00	22	0.75	0.45	0.60
11	0.70	0.15	0.20	23	0.85	0.80	0.60

Tabla 3.3: Resultados del ajuste de parámetros para los 24 sonares de PIXIE.

3.5 Modelo estocástico

Una vez detallado el modelo del sonar y el algoritmo que lo simula, pasamos a definir el cálculo de la función de verosimilitud $p(\mathbf{z} \mid \mathbf{x})$ del sensor. En primer lugar daremos una formulación para el caso de un único sonar, y a continuación la ampliaremos al caso de un anillo.

3.5.1 Sonar único

Sea $\mathbf{x} = (o_1, \ldots, o_n, x, y, \theta)$ un entorno definido por las características geométricas de los objetos que lo forman $(o_1, \ldots, o_n)$ y por la posición del sonar (x, y), y la orientación del mismo θ. Sea $\mathbf{z} = r_{real}$ una lectura de distancia r_{real} realizada por un sonar.

En una primera versión, para calcular la verosimilitud de que la distancia r_{real} haya sido producida en el entorno $\mathbf{x}$, se aplica en este entorno el modelo del sonar, calculando la distancia r_{sim} devuelta por un sonar en la posición (x, y) y con la orientación θ. La verosimilitud se calcula entonces como una distribución normal truncada:

$$p(r_{real} \mid o_1, \ldots, o_n, x, y, \theta) = e^{-(\phi(r_{real}, r_{sim})/2\sigma^2)}, \tag{3.12}$$

donde

$$\phi(r_{real}, r_{sim}) = \begin{cases} (r_{real} - r_{sim})^2 & \text{si } r_{real} < r_{max} \text{ y } r_{sim} < r_{max} \\ \rho_1 & \text{si } r_{real} = r_{max} \text{ y } r_{sim} = r_{max} \\ \rho_2 & \text{en otro caso} \end{cases}, \tag{3.13}$$

y ρ_1 es una constante cercana a 1 para el caso en el que ambas lecturas devuelvan una lectura máxima (no existe obstáculo frente al sensor) y ρ_2 es una constante de penalización para el caso en que la lectura simulada o la real devuelvan una lectura máxima cuando la otra no lo hace.

El problema de esta versión inicial es que asume un modelo gaussiano del ruido, cuando esto no es correcto. Recordemos que pequeñas variaciones de los parámetros del modelo o de la orientación del sensor producen grandes cambios en el valor devuelto por el sensor (el *efecto de escalón* que aparecía en la figura 3.20).

En la versión final de la función de verosimilitud introducimos una componente estocástica en forma de ruido gaussiano añadido a los parámetros del modelo y a la orientación del sonar. El algoritmo para calcular la verosimilitud es el siguiente

1. Generar las lecturas $r_1, \ldots, r_n$ repitiendo la simulación añadiendo ruido gaussiano a los parámetros del modelo y a la orientación del sonar.

2. Buscar el valor r_i más próximo a r_{real}.

3. Calcular la verosimilitud aplicando la ecuación 3.12 a las lecturas r_{real} y r_i

$$p(r_{real} \mid o_1, \ldots, o_n, x, y, \theta) = e^{-(\phi(r_{real}, r_i)/2\sigma^2)}. \tag{3.14}$$

Este algoritmo, a diferencia de la primera versión, produce una función de verosimilitud multimodal, mucho más acorde con las características del sensor de ultrasonidos.

En la figura 3.23 se puede comprobar el resultado de esta formulación. En ella se supone que el robot y su anillo de sonares se encuentra centrado en la habitación. Se han generado uniformemente 40000 posiciones p_i alrededor del robot y se ha supuesto que cada una de ellas define una lectura real r_i. La distancia r_i es la distancia entre la posición p_i y el robot, y la orientación θ_i correspondiente a la lectura se define por la orientación de la recta que une p_i con el centro del robot. Una vez definidas cada lectura y cada orientación, se ha calculado su verosimilitud dado el entorno y la posición del robot, y se ha dibujado la posición p_i de la lectura con un tono de gris proporcional a la verosimilitud. De esta forma, cuanto más oscura aparece una posición, mayor es la verosimilitud de su lectura asociada.

3.5.2 Anillo de sonares

Para calcular la verosimilitud de las lecturas de un anillo aplicamos el modelo anterior a cada uno de los sonares que lo componen, y calculamos la probabilidad conjunta de todas las lecturas.

Al ser lecturas independientes, podemos aplicar la siguiente expresión

$$p(z_1, \ldots, z_n \mid o_1, \ldots, o_n, x, y, \theta) = \prod_{i=1}^{n} p(z_i \mid o_1, \ldots, o_n, x, y, \theta_i), \tag{3.15}$$

donde θ es la orientación de referencia del anillo y θ_i la orientación individual del sonar que ha producido la lectura z_i. Para calcular las lecturas simuladas actualizamos el modelo del sonar con los parámetros propios del sonar que ha realizado la lectura.

3.6 Discusión

Se ha presentado en esta sección un modelo del sonar realista, que es capaz de simular con notable fidelidad el comportamiento de los sensores de ultrasonidos, modelándose correctamente lecturas que previamente eran despreciadas como errores. La simulación se basa en una adaptación del algoritmo de *trazado de rayos*, con el que se sigue el recorrido de los haces de ultrasonidos y sus rebotes con el entorno.

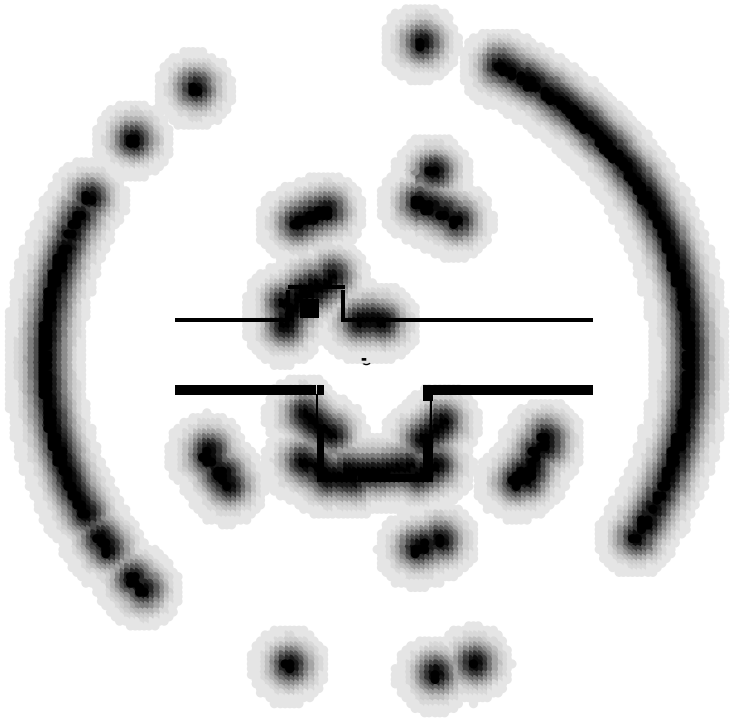


Figura 3.23: Verosimilitud de 40000 posiciones distribuidas uniformemente alrededor del anillo de sonares. Cuanto más oscura aparece una posición, mayor es la verosimilitud de su lectura asociada.

Se ha realizado un ajuste de los parámetros del modelo a partir de mediciones obtenidas por los sonares de PIXIE, y se han presentado numerosos experimentos que muestran la corrección de la simulación.

Por último se ha presentado un algoritmo estocástico con el que, variando aleatoriamente los valores de los parámetros del modelo, se obtiene una función multimodal de verosimilitud de las lecturas del sensor.

Capítulo 4

Modelos para la estimación bayesiana

El paradigma de estimación bayesiana para los robots móviles se basa en la definición probabilística de un modelo del entorno, un modelo de observación y un modelo de movimiento.

El tipo de modelo del entorno (topológico o métrico) influye directamente en la formulación del modelo de observación, así como en el tratamiento del problema del mapeado. El uso de modelos métricos parametrizables, como los que definimos en este capítulo, permite estimar posiciones absolutas del robot, así como buscar, mediante algoritmos de mapeado, los parámetros del mapa que mejor explican una secuencia de movimientos y observaciones del robot.

El modelo de observación de las lecturas de sonares de un robot permite evaluar la probabilidad (verosimilitud) de que unas lecturas se hayan realizado en una determinada posición del entorno. Para que un modelo de observación se ajuste a la realidad hay que considerar una cierta probabilidad de que las lecturas hayan sido producidas por obstáculos o características no modeladas del entorno.

Por último, el modelo de movimiento evalúa la probabilidad de que el robot se encuentre en una posición nueva, dada la posición anterior del mismo y la acción (estimada a partir de la lectura de odometría) realizada.

La calidad de los modelos de observación y de movimiento es la clave de una estimación bayesiana robusta y fiable.

4.1 Mapas del entorno

Un modelo métrico de mapa de entorno permite que los algoritmos de localización determinen la posición absoluta (coordenada x, y y orientación θ) del robot, frente a un modelo topológico en donde la localización no es tan exacta (robot en pasillo, robot al final del pasillo, etc.).

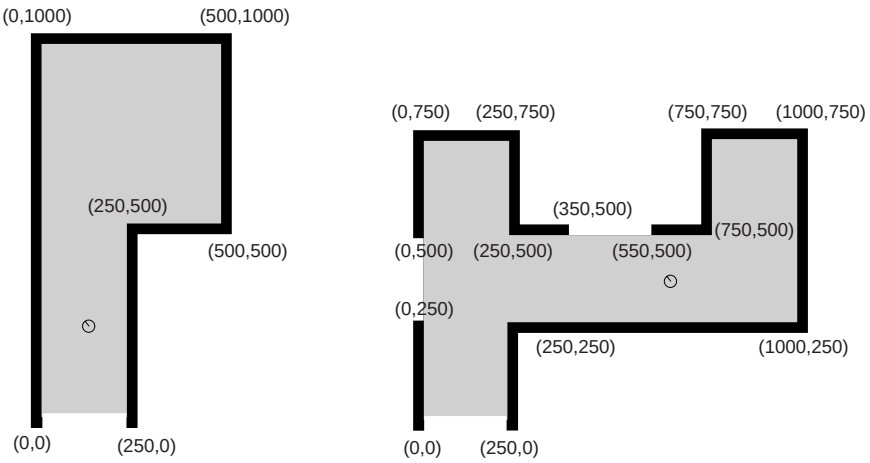


Figura 4.1: Ejemplos de mapas poligonales.

En el capítulo 2 revisamos los distintos modelos de entorno utilizados en la literatura. El modelo que proponemos se acerca a los modelos CAD utilizados por Burgard (Burgard, Fox, Henning, y Schmidt 1996; Burgard, Cremers, Fox, Hahnel, Lakemeyer, Schulz, Steiner, y Thrun 1998), aunque formalizamos su definición utilizando regiones poligonales, que van a permitir construir modelos paramétricos del entorno.

Un mapa del entorno se define mediante una región poligonal (polígono simple cerrado) cuyos vértices $(p_1, p_2, \ldots, p_n)$ representan coordenadas en el plano $(p_i = (x, y))$. Las aristas $(a_1 = \overline{p_1 p_2}, a_2 = \overline{p_2 p_3}, \ldots, a_n = \overline{p_n p_1})$ representan los límites de la región poligonal y están etiquetadas con ABIERTO o CERRADO, dependiendo de si definen una zona abierta o cerrada (pared). Algunos ejemplos de mapas del entorno se muestran en la figura 4.1.

Las coordenadas de los vértices pueden utilizar un conjunto $\phi = (d_1, \ldots, d_n)$ de parámetros para definir modelos genéricos paramétricos. Por ejemplo, en la figura 4.2 se define un modelo con tres parámetros que permite modelar desde un final de pasillo hasta una esquina.

La definición paramétrica de los modelos del entorno permite formular el problema del mapeado como un problema de estimación de parámetros. La utilización de estos parámetros para definir las posiciones de los vertices proporciona libertad suficiente para definir un amplio conjunto de modelos y para introducir restricciones *ad-hoc* que acoten la búsqueda.

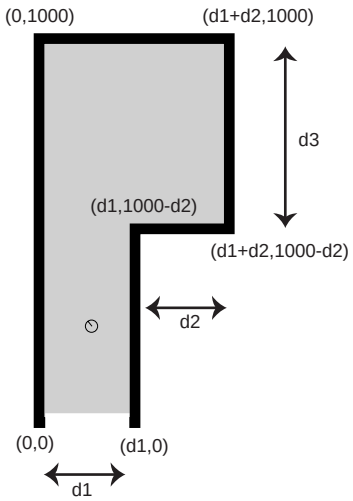


Figura 4.2: Ejemplo de mapa poligonal definido mediante los parámetros d_1, d_2 y d_3.

4.2 Modelo de observación

Un modelo de observación proporciona una estimación de la función de densidad $p(\mathbf{z} \mid \mathbf{x}, \phi)$. Recordemos que en el enfoque bayesiano esta función de densidad mide la verosimilitud de que unas medidas $\mathbf{z} = (z_1, \ldots, z_n)$ hayan sido producidas en una configuración $\mathbf{x}$ del modelo que se está estimando definido por los parámetros ϕ. En nuestro caso $(z_1, \ldots, z_n)$ son las lecturas tomadas por un barrido del anillo de sonares del robot móvil, $\mathbf{x}$ son las posiciones del robot en el modelo del entorno y ϕ son los parámetros que definen el mismo.

En adelante supondremos un modelo fijo de entorno, por lo que utilizaremos $p(\mathbf{z} \mid \mathbf{x})$ en lugar de $p(\mathbf{z} \mid \mathbf{x}, \phi)$.

Un buen modelo de observación debe adecuarse lo más posible a las probabilidades reales de $p(\mathbf{z} \mid \mathbf{x})$. Sería ideal construir este modelo a partir de datos estadísticos de las lecturas reales del anillo en diferentes configuraciones del entorno. Sin embargo, el alto número de configuraciones posibles (todas las posiciones del robot en todos los posibles modelos de entorno) y la alta variabilidad de las lecturas hacen, en principio, que este enfoque no sea factible.

Usaremos el modelo del sensor formulado en el capítulo 3 como base de construcción de la función de verosimilitud. Dada una posición del robot en el entorno, podemos obtener lecturas simuladas de los sonares y compararlas con las lecturas reales, dando más verosimilitud a las reales cuanto más se parezcan a las simuladas.

Para simplificar la formulación supongamos que deseamos estimar la verosimilitud de una única lectura z_i obtenida desde la posición $\mathbf{x}$. Formalmente, el modelo del sensor define una función de probabilidad para las lecturas $\hat{p}(z \mid \mathbf{x})$, a partir de la que formulamos la verosimilitud de z_i como $\hat{p}(z_i \mid \mathbf{x})$. Al ser $\hat{p}$ una función de probabilidad sin una fórmula analítica, no es posible evaluar esta verosimilitud directamente. Si embargo, es posible muestrear $\hat{p}$, aplicando los algoritmos de simulación del modelo del sonar y generar M lecturas simuladas $l_1(\mathbf{x}), \ldots, l_M(\mathbf{x})$. Utilizamos, entonces, estas lecturas simuladas para representar $\hat{p}$ como una suma de distribuciones normales centradas en cada una de estas lecturas resultantes. De esta forma, es posible calcular $\hat{p}(z_i \mid \mathbf{x})$ a partir de estas distribuciones normales. Recordemos que el modelo del sensor ha sido ajustado mediante lecturas reales, por lo que podemos afirmar que la función de verosimilitud se estima, de forma indirecta, a partir de datos reales.

El modelo de observación que se formula en esta sección contempla además la posibilidad de que algunas lecturas hayan sido causadas por obstáculos no modelados.

En los siguientes apartados se formula de forma general la función de probabilidad condicional, considerando las dependencias entre variables aleatorias. Después se concreta el modelo de observación, definiendo cada una de las distribuciones que intervienen en su formulación. Por último se proporcionan ejemplos y resultados del modelo definido.

4.2.1 Formulación general

Las lecturas $\mathbf{z} = (z_1, \ldots, z_n)$ son lecturas contiguas de un anillo circular de sonares (las lecturas z_i y z_{i+1} son contiguas, así como la lectura z_1 y la z_n) y todas se realizan en el mismo instante de tiempo t. La orientación asociada a cada lectura de sonar z_i la denominamos θ_i.

Para modelar la posible presencia de obstáculos, se define una variable aleatoria $\Phi = \{o, \overline{o}\}$ con una probabilidad $p(o) = q$ de presencia de un obstáculo y una probabilidad $p(\overline{o}) = 1 - q$ de que no exista obstáculo en la dirección de la lectura de sonar que se está considerando.

De forma general, la verosimilitud del conjunto de lecturas la podemos formular como $p(z_1, z_2, \ldots, z_n \mid \mathbf{x}, \Phi)$, dependiendo del entorno $\mathbf{x}$ y de la existencia de algún obstáculo Φ. Desarrollando la expresión, se obtiene

$$\begin{aligned} &p(z_1, z_2, \ldots, z_n \mid \mathbf{x}, \Phi) = \\ &\quad p(z_1 \mid \mathbf{x}, \Phi) \cdot p(z_2 \mid z_1, \mathbf{x}, \Phi) \cdot \ldots \cdot p(z_n \mid z_1, z_2, \ldots, z_{n-1}, \mathbf{x}, \Phi). \end{aligned} \tag{4.1}$$

El caso de que no haya obstáculos en el entorno, consideramos que todas las lecturas son independientes entre sí, dependiendo únicamente de la configuración del entorno

$$p(z_i \mid z_{j \neq i}, \mathbf{x}, \overline{o}) = p(z_i \mid \mathbf{x}). \tag{4.2}$$

Sin embargo, las lecturas producidas por obstáculos no las consideramos independientes, ya que suponemos obstáculos con una cierta extensión angular. Por ello, lecturas cercanas angularmente tendrán valores similares, ya que van a ser afectadas por el mismo obstáculo.

Esto lo formulamos definiendo un entorno G_i de lecturas afectadas por un obstáculo detectado por la lectura z_i, y restringiendo a dicho entorno la dependencia entre las variables

$$p(z_i \mid z_{j\neq i}, \mathbf{x}, o) = p(z_i \mid z_{j\in G_i}, \mathbf{x}, o), \tag{4.3}$$

donde consideramos como entorno de z_i a las lecturas anterior y posterior

$$G_i = \begin{cases} \{i-1, i+1\} & \text{si } i \in \{2, n-1\} \\ \{n, 2\} & \text{si } i = 1 \\ \{n-1, 1\} & \text{si } i = n \end{cases}$$

Aplicando estas consideraciones a la ecuación (4.1), se puede formular la función de verosimilitud como

$$\begin{aligned} &p(z_1, z_2, \ldots, z_n \mid \mathbf{x}, \Phi) = \\ &\quad p(z_1 \mid \mathbf{x}, \Phi) \cdot p(z_n \mid z_1, z_{n-1}, \mathbf{x}, \Phi) \cdot \prod_{i=2}^{n-1} p(z_i \mid z_j, \mathbf{x}, \Phi). \end{aligned} \tag{4.4}$$

Para calcular todos los términos de esta expresión, debemos formular las probabilidades condicionales que aparecen en la misma.

Primero detallamos algo más la ecuación (4.3), en donde se formula la probabilidad condicional de una lectura z_i dadas las lecturas contiguas, la representación del entorno y dada la existencia de un obstáculo frente a z_i. Se deben contemplar dos casos: o bien el obstáculo frente a z_i ha sido detectado por alguna lectura contigua, o bien ninguna lectura contigua lo ha detectado. Para ello se descompone el evento o (existencia de un obstáculo frente a z_i) en dos eventos: o_1, que denota el caso en que el obstáculo ha afectado alguna lectura contigua a z_i y o_2, que denota el caso contrario. Las probabilidades de ambos eventos deben sumar q:

$$p(o_1) = q_1, \qquad p(o_2) = q_2 \qquad \mid \qquad q_1 + q_2 = q.$$

En el caso o_1 la probabilidad de z_i depende de los valores de las lecturas contiguas, ya que se supone que alguna de ellas ha detectado el obstáculo que hay frente a z_i y que z_i deberá dar un valor similar a ésta

$$p(z_i \mid z_{j\neq i}, \mathbf{x}, o_1) = p(z_i \mid z_{j\in G_i}) \tag{4.5}$$

En el caso o_2, en el que el obstáculo frente a la lectura no ha sido detectado por ninguna lectura contigua, estas lecturas no aportan ninguna información sobre la posición del obstáculo, por lo que la probabilidad de z_i no depende de ninguna de ellas, sino de la presencia del obstáculo y del entorno

$$p(z_i \mid z_{j\neq i}, \mathbf{x}, o_2) = p(z_i \mid \mathbf{x}, o) \tag{4.6}$$

Tras alguna derivación, utilizando teoría básica de probabilidad y las ecuaciones (4.2), (4.5) y (4.6) se llega a las siguientes expresiones (4.7), (4.8) y (4.9) que definen las probabilidades condicionales usadas en la función de verosimilitud anterior (4.4)

$$p(z_i \mid \mathbf{x}, \Phi) = qp(z_i \mid \mathbf{x}, o) + (1-q)p(z_i \mid \mathbf{x}) \tag{4.7}$$

$$p(z_i \mid z_{i-1}, \mathbf{x}, \Phi) = q_1 p(z_i \mid z_{i-1}) + q_2 p(z_i \mid \mathbf{x}, o) + (1-q)p(z_i \mid \mathbf{x}) \tag{4.8}$$

$$p(z_i \mid z_{i-1}, z_{i+1}, \mathbf{x}, \Phi) = q_1 p(z_i \mid z_{i-1}, z_{i+1}) + q_2 p(z_i \mid \mathbf{x}, o) + (1-q)p(z_i \mid \mathbf{x}) \tag{4.9}$$

La ecuación (4.7) establece que la verosimilitud de z_i depende de la verosimilitud de que dicha lectura haya sido producida por un obstáculo en el entorno $\mathbf{x}$, ponderada por la probabilidad de que exista un obstáculo, y de la verosimilitud de la lectura dado únicamente el entorno $\mathbf{x}$, ponderada por la probabilidad de que no exista ningún obstáculo.

Las expresiones (4.8) y (4.9) definen las probabilidades condicionales de una lectura z_i en función de lecturas contiguas, el entorno y la presencia de un obstáculo.

4.2.2 Funciones de probabilidad condicional

En esta sección se definen las funciones de probabilidad condicional en las que se basa la función de verosimilitud (4.4). Estas funciones son $p(z_i \mid \mathbf{x})$, $p(z_i \mid \mathbf{x}, o)$, $p(z_i \mid z_{i-1})$ y $p(z_i \mid z_{i-1}, z_{i+1})$.

La primera de ellas define la verosimilitud de una lectura z_i dada una posición $\mathbf{x}$ en el entorno.

$$p(z_i \mid \mathbf{x}) = \sum_{l_i(\mathbf{x})} \frac{1}{\sqrt{2\pi}\sigma_1} \exp\left(-\frac{(l_i(\mathbf{x}) - z_i)^2}{2\sigma_1^2}\right) p(l_i(\mathbf{x})) \tag{4.10}$$

Esta densidad se formula como una suma de distribuciones normales con desviaciones típicas σ_1 centradas en las lecturas $l_i(\mathbf{x})$ proporcionadas por el modelo simulado del sonar en la posición $\mathbf{x}$. Recordemos que $l_i(\mathbf{x})$ es una variable aleatoria definida por el modelo del sonar y que modela la lectura realizada por el sonar en la dirección θ_i en una posición $\mathbf{x}$ del entorno.

La segunda función de densidad define la verosimilitud de una lectura dado un obstáculo y una posición.

$$p(z_i \mid \mathbf{x}, o) = \begin{cases} 1/(K_s - r_i(\mathbf{x})) & K_s \leq z_i \leq r_i(\mathbf{x}) \\ 0 & \text{para cualquier otro valor} \end{cases} \tag{4.11}$$

Esta densidad se define como una distribución uniforme en el intervalo $(K_s, r_i(\mathbf{x}))$, siendo K_s la distancia de seguridad de los métodos de navegación local (distancia en la que es seguro que no existirá un obstáculo) y $r_i(\mathbf{x})$ la menor distancia desde $\mathbf{x}$ a un segmento del entorno en la dirección θ_i.

Por último, las distribuciones condicionales $p(z_i \mid z_j)$ y $p(z_i \mid z_j, z_k)$ se modelan, respectivamente, como una distribución normal centrada en z_j y como una suma de dos normales centradas en z_j y z_k en el segundo.

$$p(z_i \mid z_j) = \frac{1}{\sqrt{2\pi}\sigma_2} \exp\left(-\frac{(z_i - z_j)^2}{2\sigma_2^2}\right) \tag{4.12}$$

$$p(z_i \mid z_j, z_k) = \frac{1}{2\sqrt{2\pi}\sigma_3} \exp\left(-\frac{(z_i - z_j))^2}{2\sigma_3^2}\right) + \frac{1}{2\sqrt{2\pi}\sigma_3} \exp\left(-\frac{(z_i - z_k))^2}{2\sigma_3^2}\right). \tag{4.13}$$

4.2.3 Experimentos

En esta sección presentamos un ejemplo representativo del funcionamiento del modelo de observación.

Figura 4.3: Representación de la función de densidad de las lecturas del sonar, $p(z_i \mid \mathbf{x})$, para una posición (x, y) fija del robot en un entorno de final de pasillo centrado en la posición en la que se han tomado las lecturas. A la izquierda nuestro modelo, a la derecha el modelo de (Burgard, Cremers, Fox, Hahnel, Lakemeyer, Schulz, Steiner, y Thrun 1998).

Figura 4.4: Situación de PIXIE en el experimento con el modelo de observación.

Utilizamos la función de verosimilitud $p(\mathbf{z} \mid \mathbf{x})$ para encontrar la posición de máxima verosimilitud de PIXIE en un final de pasillo, a partir de lecturas realizadas por su anillo de sonares. Comprobaremos que la función de probabilidad propuesta mejora mucho los resultados que se obtienen utilizando un modelo más sencillo del sensor, como el propuesto por (Fox, Burgard, Thrun, y Cremers 1998b). En este modelo se simula la distancia obtenida por un sonar como la distancia al elemento del entorno más cercano en la orientación del sonar. En la figura 4.3 se muestran las funciones de densidad producidas por cada uno de los modelos.

Es importante resaltar, para colocar los resultados en su justa medida, que el barrido se han extraído del conjunto de lecturas de la figura 3.8, lecturas con una gran cantidad de ruido producido por la presencia en el pasillo de múltiples columnas y por la abundancia de puertas (ver fotografía en la figura 4.4).

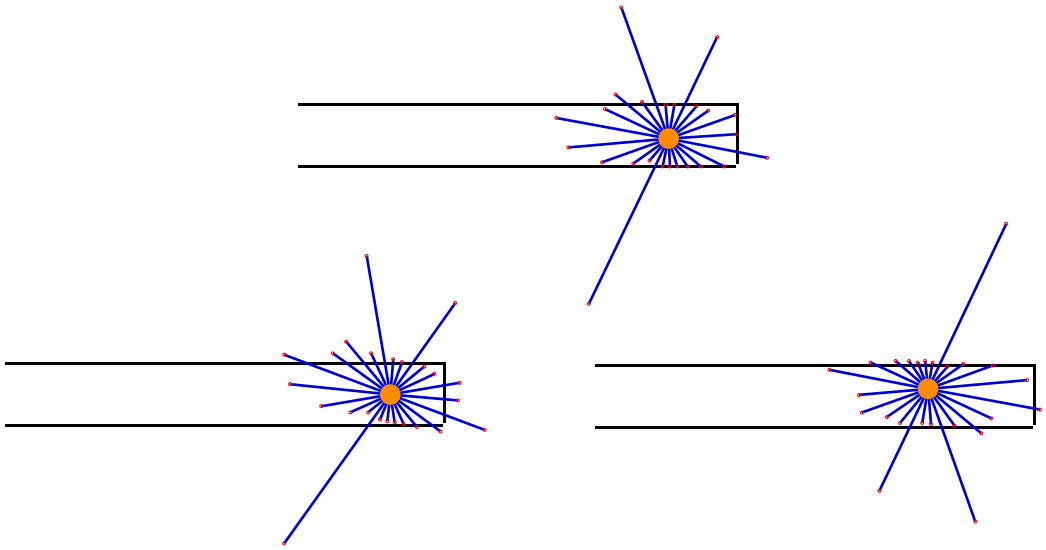


Figura 4.5: Superior: Barrido de 24 lecturas del anillo de sonares en la posición del robot en la que se tomó el barrido . Inferior izquierda: posición de máxima verosimilitud con la función propuesta. Inferior derecha: posición de máxima verosimilitud con la función simplificada.

El experimento ha consistido en, dadas unas lecturas de los sonares (ver figura 4.5), buscar la localización $\mathbf{x} = (x, y, \theta)$ de *máxima verosimilitud* ($\mathbf{x}_{MV}$) suponiendo conocido las dimensiones del final del pasillo.

Esta localización es aquella para la que el modelo de observación devuelve una probabilidad máxima, esto es

$$\mathbf{x}_{MV} = \arg\max_{\mathbf{x}} p(\mathbf{z} \mid \mathbf{x}).$$

En este ejemplo no se considera el modelo de movimiento del robot, ni las probabilidades a priori de las configuraciones, ya que se pretende mostrar el comportamiento aislado de la función de verosimilitud.

Las tres componentes de la posición se han discretizado para poder llevar a cabo la experimentación. Se ha calculado, para el barrido de lecturas $\mathbf{z}$, su verosimilitud $p(\mathbf{z} \mid x, y, \theta)$, con

$$H_x = \{20, 40, 60, \ldots, 420\} \quad \text{(21 hipótesis)}$$

Parámetros	Valores reales	Máxima verosimilitud 1	Máxima verosimilitud 2
x	190 cms.	150 cms.	290 cms.
y	80 cms.	90 cms.	110 cms.
θ	180 grad.	170 grad.	0 grad.

Tabla 4.1: Comparación de la posición real de PIXIE (primera columna) con de las posiciones de máxima verosimilitud de nuestro modelo de observación (segunda columna) y del modelo de observación de Fox (tercera columna).

$$\begin{aligned} H_y &= \{20, 40, 60, 80, 100\} \quad \text{(5 hipótesis)} \\ H_\theta &= \{9, 18, 27, \ldots, 360\} \quad \text{(40 hipótesis)} \end{aligned}$$

El resumen de la comparación entre ambos modelos de observación se muestra en la tabla 4.1. Se puede comprobar que la corrección del modelo propuesto es, en este caso, mucho mayor que la del propuesto por Fox. La posición real del robot es (190,90,180), la calculado con nuestro modelo es (150,90,170) y la calculada con el modelo de Fox es (290,110,0).

Es interesante observar qué valores de probabilidad calculan los modelos de observación para el todas las posibles posiciones del robot. Para ello representamos (figuras 4.6 y 4.7) los las probabilidades marginales de dos de los parámetros variando el tercero. Podemos comprobar que el modelo de observación propuesto es más sensible y selectivo (además de exacto) que el de Fox.

Por último, en la figura 4.8 se representa la verosimilitud marginal de x e y con respecto a θ,

$$p(\mathbf{z} \mid t_x, t_y) = \sum_{\theta \in H_\theta} p(\mathbf{z} \mid t_x, t_y, \theta).$$

Esta marginal puede entenderse como la información que proporciona la función de verosimilitud sobre la localización de PIXIE en el pasillo.

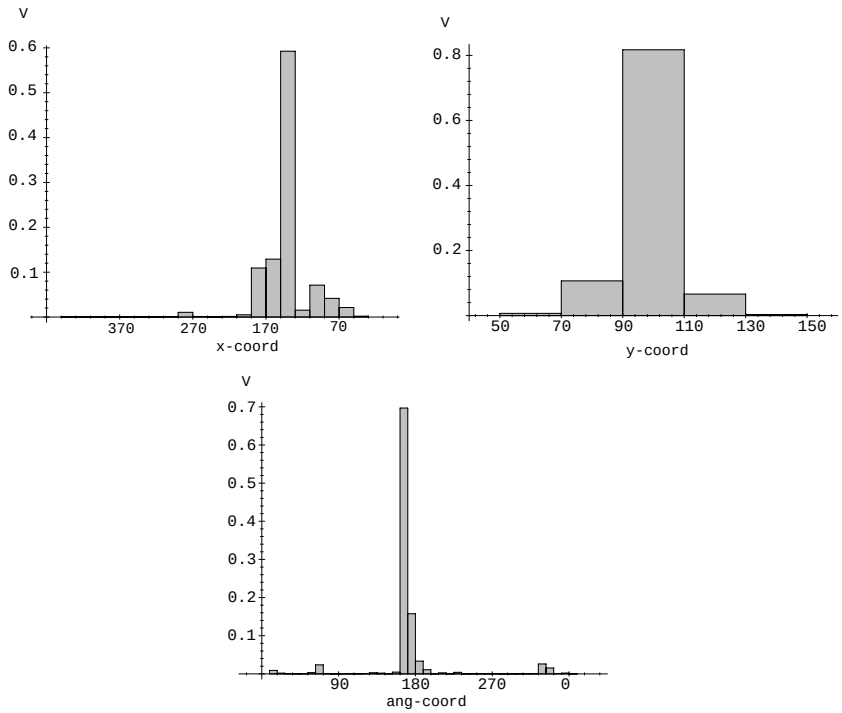


Figura 4.6: Función de verosimilitud propuesta. Verosimilitudes marginales de x, y y θ. Posición real del robot: $x = 190$, $y = 80$, $\theta = 180$.

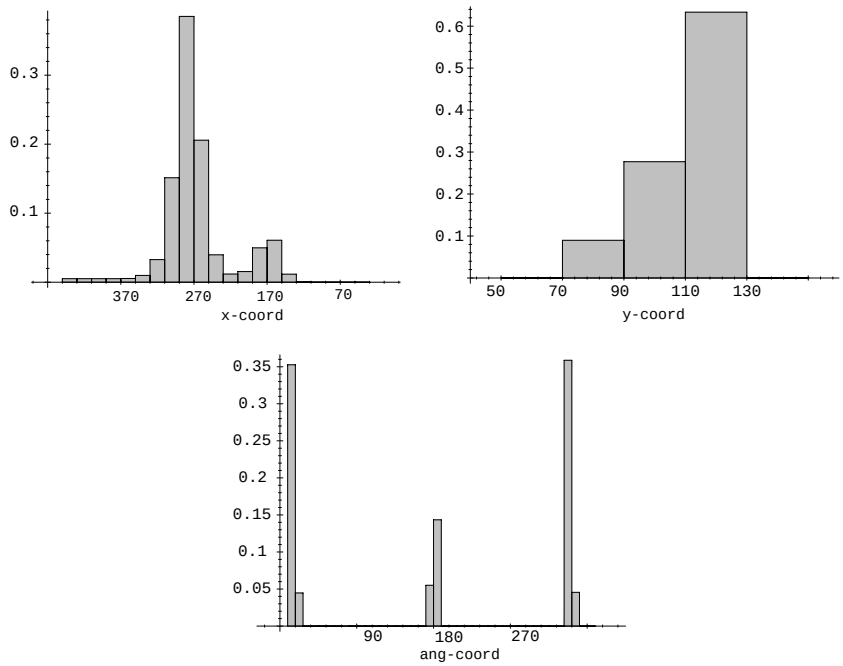


Figura 4.7: Función de verosimilitud de Fox. Verosimilitudes marginales de x, y y θ. Posición real del robot: $x = 190$, $y = 80$, $\theta = 180$.

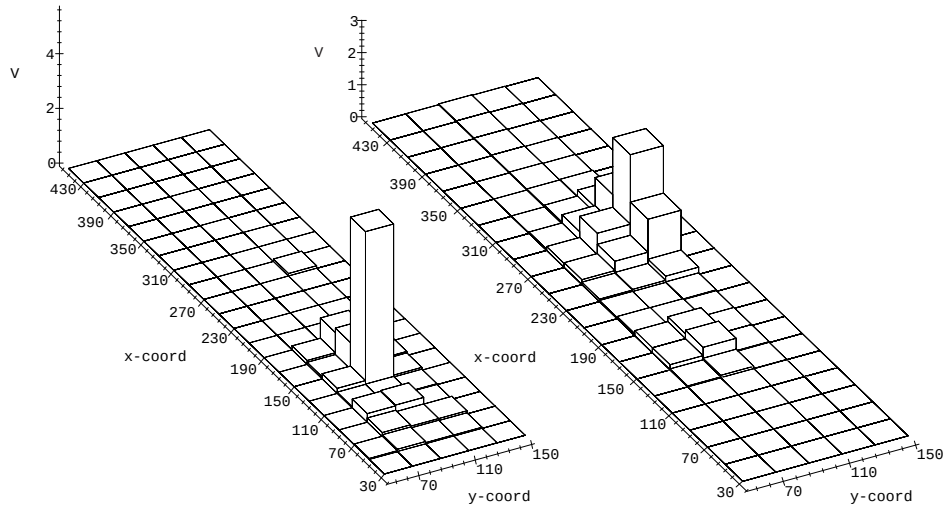


Figura 4.8: Funciones de verosimilitud marginal de x e y con respecto a la orientación. Izquierda: función propuesta. Derecha: función de Fox. Posición real del robot: $x = 190$, $y = 80$, $\theta = 180$.

4.3 Modelo dinámico

En muchos problemas de estimación temporal (seguimiento de objetos en imágenes, por ejemplo) el sistema no proporciona ninguna información sobre su evolución o es muy complicada la obtención de la misma. Por ejemplo, supongamos la primera instantánea de una película de una persona caminando. En ausencia de más información es muy complicado dar una estimación de hacia dónde se moverá la imagen de la persona. Es necesario entonces suponer un modelo a priori de la evolución temporal del sistema. Por ejemplo, se puede suponer que en el tipo de secuencias que se analizan las personas se mueven hacia la derecha, o se mueven con una velocidad constante.

En el caso de los robots móviles, sin embargo, se tiene la ventaja de que el sistema proporciona información de su evolución, mediante lo que se denomina *odometría*. El robot va integrando los incrementos de posición x, y y de orientación θ medidos con contadores (*shaft encoders*) situados en sus partes móviles. Estas medidas permiten estimar incrementos pequeños de posición del robot, pero no es aconsejable su uso para localizarlo en periodos largos, debido al alto error acumulativo de las mismas.

El modelo de movimiento del robot se puede formular, pues, a partir de las posiciones del robot proporcionadas por la odometría. Supongamos que las posiciones estimadas por odometría en el instante anterior son $(\hat{x}_{t-1}, \hat{y}_{t-1}, \hat{\theta}_{t-1})$, y en el instante actual $(\hat{x}_t, \hat{y}_t, \hat{\theta}_t)$. El desplazamiento lineal del robot estimado por odometría $\triangle_{t-1}\hat{d}$ entre el instante $t-1$ y el t y el desplazamiento angular del mismo $\triangle_{t-1}\hat{\theta}$ se pueden estimar como

$$\triangle_{t-1}\hat{d} = \sqrt{(\hat{x}_t - \hat{x}_{t-1})^2 + (\hat{y}_t - \hat{y}_{t-1})^2} \tag{4.14}$$

$$\triangle_{t-1}\hat{\theta} = \hat{\theta}_t - \hat{\theta}_{t-1} \tag{4.15}$$

Podemos definir entonces una variable aleatoria, $\triangle_{t-1}d$ que define el desplazamiento real del robot dado un desplazamiento $\triangle_{t-1}\hat{d}$ medido mediante la odometría. Su función de densidad se define como una distribución normal de media $\triangle_{t-1}\hat{d}$ y desviación típica σ_d

$$p(\triangle_{t-1}d \mid \triangle_{t-1}\hat{d}) = \frac{1}{\sqrt{2\pi}\sigma_d} \exp\left(-\frac{(\triangle_{t-1}d - \triangle_{t-1}\hat{d})^2}{2\sigma_x^2}\right) \tag{4.16}$$

Una vez definida la variable aleatoria $\triangle_{t-1}d$ es posible formular las posiciones del robot en el instante t, x_t e y_t, como variables aleatorias calculadas a partir de $\triangle_{t-1}d$, y de las posiciones en el instante anterior

$$x_t = x_{t-1} + d\cos(\theta_{t-1}) \tag{4.17}$$

$$y_t = y_{t-1} + d\sin(\theta_{t-1}). \tag{4.18}$$

Las funciones de densidad de x_t e y_t se pueden calcular y muestrear a partir de las ecuaciones 4.15,4.16 y 4.18.

La orientación del robot en el instante t, θ_t, la formulamos también como una variable aleatoria con función de densidad normal centrada en la orientación estimada

$$p(\theta_t \mid \triangle_{t-1}\hat{\theta}) = \frac{1}{\sqrt{2\pi}\sigma_\theta} \exp\left(-\frac{(\theta_t - (\theta_{t-1} + \triangle_{t-1}\hat{\theta}))^2}{2\sigma_\theta^2}\right) \tag{4.19}$$

4.4 Discusión

En este capítulo se han propuesto distintos modelos necesarios para la localización y el mapeado bayesianos.

En primer lugar, se ha presentado un modelo paramétrico de mapa de entorno, basado en regiones poligonales. La posibilidad de definir las posiciones de sus vértices de forma paramétrica dota al modelo de gran flexibilidad y permite formular de forma sencilla restricciones geométricas en los mapas de entorno.

En segundo lugar, se ha propuesto un modelo de observación robusto basado en el modelo del sonar del capítulo anterior y en una formulación probabilística que contempla la posible presencia en el entorno de obstáculos no modelados. Se ha comprobado, con datos obtenidos del robot PIXIE, que el modelo es más robusto y fiable que otros modelos más simples presentados en la literatura.

Por último, se ha presentado un modelo de movimiento que define la probabilidad de localización del robot, dado una localización anterior y unos datos de odometría.

Capítulo 5

Localización basada en filtros de partículas

La formulación bayesiana del problema de localización global choca con un problema fundamental: ¿cómo representar la función de probabilidad a posteriori?. Esta función de probabilidad estima las posiciones más probables del robot. Normalmente, debido al ruido producido por obstáculos no modelados o a ambigüedades en la percepción del entorno, esta función será multimodal, esto es, existirán posiciones muy distintas con alta probabilidad de que el robot se encuentre simultáneamente en ellas.

La mejor solución planteada hasta el momento ha sido la utilización de una *rejilla de probabilidad*, una rejilla que discretiza el espacio de estados (todas las posibles posiciones del robot en el entorno). Los algoritmos de localización que utilizan esta solución calculan la probabilidad de que el robot esté situado en cada una de las celdillas utilizando el modelo bayesiano. Sin embargo, esta solución tiene un alto coste espacial, computacional y obliga a definir las dimensiones de las celdillas de forma *ad-hoc*.

En este capítulo se presenta el filtro *bootstrap*, el cual resuelve el problema representando la función de densidad por un conjunto de muestras extraídas de la distribución.

5.1 Introducción

En este capítulo se describe un algoritmo de estimación muestral de la densidad de probabilidad a posteriori $p(\mathbf{x}_t \mid Z^t, A^{t-1})$ que define el estado del robot.

Tal y como se formuló en el capítulo 2, esta densidad de probabilidad representa el estado del robot en el instante actual, $\mathbf{x}_t$, dadas unas observaciones realizadas por sus sensores, $Z^t = \{\mathbf{z}_1, \ldots, \mathbf{z}_t\}$, y una secuencia de acciones realizadas, $A^{t-1} = \{\mathbf{a}_1, \ldots, \mathbf{a}_{t-1}\}$.

Si resumimos la formulación del capítulo 2, podemos descomponer el problema del cálculo de $p(\mathbf{x}_t \mid Z^t, A^{t-1})$ en dos pasos: predicción e integración de las observaciones.

1. **Predicción**.

   En este paso se utiliza el *modelo de movimiento* (ver sección 4.3) para predecir la posición del robot en el instante actual, a partir de la densidad de probabilidad del instante anterior. Esta densidad se puede calcular mediante la integración

$$p(\mathbf{x}_t \mid Z^{t-1}, A^{t-1}) = \int_{\mathbf{x}_{t-1}} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{a}_{t-1}) p(\mathbf{x}_{t-1} \mid Z^{t-1}, A^{t-1}). \tag{5.1}$$

2. **Integración de las observaciones**.

   En el segundo paso se integran las observaciones realizadas en el instante actual, $\mathbf{z}_t$. Para ello se utiliza el *modelo de observación* (ver sección 4.2) y se ponderan las probabilidades obtenidas por la fase anterior en función de la verosimilitud de las lecturas $\mathbf{z}_t$, para obtener la densidad a posteriori

$$p(\mathbf{x}_t \mid Z^t, A^{t-1}) = \alpha_t \, p(\mathbf{z}_t \mid \mathbf{x}_t) \, p(\mathbf{x}_t \mid Z^{t-1}, A^{t-1}) \tag{5.2}$$

   La constante α_t es un factor de escala que asegura que $\int_{\mathbf{x}_t} p(\mathbf{x}_t) = 1$.

Tal y como se revisó en la sección 2.5.3, existen varios enfoques para obtener una representación de la densidad 5.2 (filtro de Kalman, modelos topológicos y rejillas de probabilidad), pero todos ellos presentan distintos problemas.

La solución que planteamos es la representación de la densidad mediante un conjunto de muestras extraídas de dicha distribución. En el siguiente apartado se presenta el filtro *bootstrap* que realiza este muestreo.

5.2 Filtro *bootstrap*

Presentamos en este apartado el filtro *bootstrap*, propuesto por Gordon et. al. (Gordon, Salmond, y Smith 1993). Se trata de un filtro en el que la densidad a posteriori se representa mediante una distribución de partículas en el espacio de estados. Este enfoque se ha desarrollado de forma independiente en los últimos años en campos como la estadística, la economía o la visión artificial (Kitawa 1987; West 1992; Gordon, Salmond, y Smith 1993; Isard y Blake 1996; Kitawa 1996; Carpenter, Clifford, y Fernhead 1997; Pitt y Shephard 1997). Los nombres con los que se ha denominado este enfoque son *Monte Carlo*, *CONDENSATION* e *Importance Resampling Filters*, aunque últimamente se está utilizando el término *filtros de partículas* para todos ellos.

Son propuestas similares que propagan las partículas (muestras de la función de densidad a posteriori) utilizando el modelo de movimiento $p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{a}_t)$ y el modelo de verosimilitud $p(\mathbf{z}_t \mid \mathbf{x}_t)$, de forma que el peso combinado de las partículas de una región aproxima la integral de la función de densidad a posteriori en esa región.

En concreto, el filtro *bootstrap* representa la densidad a posteriori mediante un conjunto de N muestras $(\mathbf{m}_1, \ldots, \mathbf{m}_N)$ y sus probabilidades asociadas $(\pi_1, \ldots, \pi_N)$.

Inicialmente, el conjunto de muestras se escoge a partir de la distribución a priori $p(\mathbf{x}_0)$. Si no existe información a priori, entonces las muestras se distribuyen uniformemente por el espacio de estados. Posteriormente, en cada instante de tiempo t, se actualizan las N muestras en función de la acción anterior $\mathbf{a}_{t-1}$ y la observación actual $\mathbf{z}_t$.

Para ello, (fase 1 del algoritmo) se aplica el modelo de movimiento $p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{a}_{t-1})$ a cada una de las N muestras, generando un nuevo conjunto de muestras. Estas muestras representan la predicción de la variable de estado, sin considerar la observación. Para considerar la observación (fase 2 del algoritmo), se obtiene el peso π^i asociado a cada muestra según la verosimilitud de que la observación haya sido realizada en un estado del sistema definido por la muestra, $p(\mathbf{z}_t \mid \mathbf{x}_t)$.

En un último paso (fase 3 del algoritmo), se remuestrea el conjunto de muestras, extrayendo (con reemplazo) N muestras del conjunto actual, con probabilidad proporcional al peso de cada una. En este nuevo conjunto, por ejemplo, desaparecerán las muestras para las que no hay evidencia de verosimilitud. Una vez construido el nuevo conjunto de muestras, se escalan los pesos asociados a cada una para que representen la probabilidad asociada a cada muestra. Este nuevo conjunto de muestras constituye una representación muestral de la probabilidad a posteriori. En la tabla 5.1 se detalla este algoritmo y en la figura 5.1 se explica gráficamente la evolución de las muestras en cada fase.

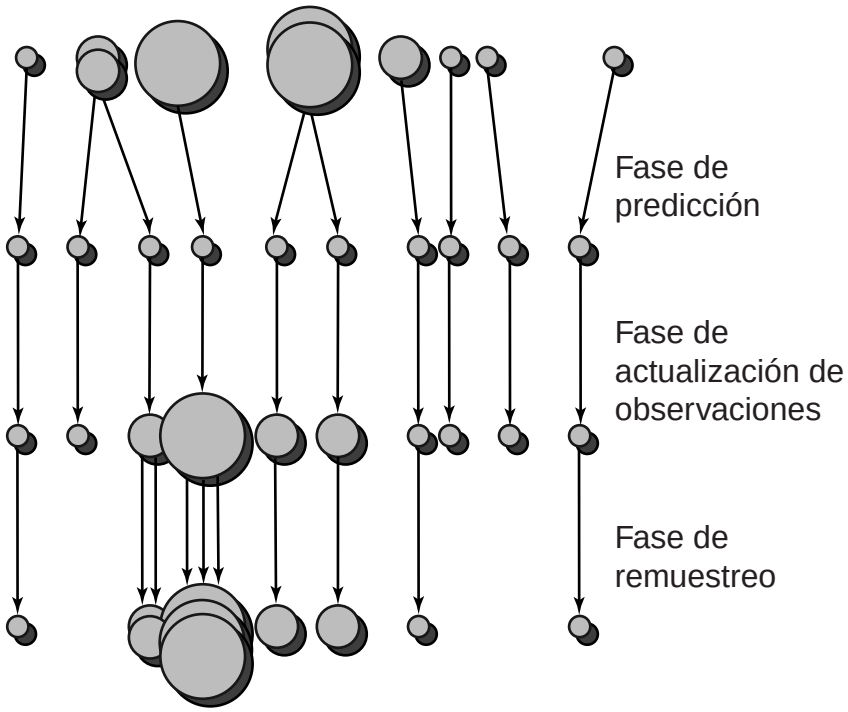


Figura 5.1: Funcionamiento del algoritmo *bootstrap*. La figura supone que las muestras están estimando un único parámetro, distribuido en el eje horizontal. Las muestras se representan por círculos centrados en el valor del parámetro que representan. El área de los círculos representa el peso de cada muestra.

Algoritmo BOOTSTRAP

A partir del conjunto anterior de muestras $M_{t-1} = \{(\mathbf{m}_{t-1}^1, \pi_{t-1}^1), \ldots, (\mathbf{m}_{t-1}^N, \pi_{t-1}^N)\}$ en el instante $t-1$, construir un nuevo conjunto de muestras $M_t = \{(\mathbf{m}_t^1, \pi_t^1), \ldots, (\mathbf{m}_t^N, \pi_t^N)\}$ para el instante actual t.

1. **Fase de predicción**:

   Sea $\mathbf{a}_{t-1}$ la acción ejecutada por el sistema en el instante $t-1$.

   Para cada muestra $\mathbf{m}_{t-1}^i \in M_{t-1}$, predecir su nuevo estado en el instante t, muestreando la densidad que define el modelo de movimiento

   $$\tilde{\mathbf{m}}_t^i \leftarrow \text{Muestra de } p(\mathbf{x}_t \mid \mathbf{x}_{t-1} = \mathbf{m}_{t-1}^i, \mathbf{a}_{t-1}).$$

   De esta forma, se construye un nuevo conjunto de N muestras $\tilde{M}_t$ para el instante actual.

2. **Fase de actualización de observaciones**:

   Sea $\mathbf{z}_t$ la observación realizada por el sistema en el instante actual t.

   Para cada muestra $\tilde{\mathbf{m}}_t^i \in \tilde{M}_t$, actualizar su peso asociado π_t^i según la verosimilitud de que los datos $\mathbf{z}_t$ hayan sido observados en el estado $\tilde{\mathbf{m}}_t^i$

   $$\tilde{\pi}^i \leftarrow p(\mathbf{z}_t \mid \mathbf{x}_t = \tilde{\mathbf{m}}_t^i).$$

3. **Fase de remuestreo**:

   Construir el nuevo conjunto de N muestras M_t remuestreando (con sustitución) el conjunto $\tilde{M}_t$, de forma que se escoge cada muestra $\tilde{\mathbf{m}}_t^i$ con probabilidad proporcional a la verosimilitud de la misma $\tilde{\pi}_t^i$.

   Desde $i = 1$ hasta N:
   $(\mathbf{m}_t^i, \pi_t^i) \leftarrow$ Escoger muestra de $\tilde{M}_t$

   Normalizar todas las probabilidades π_t^i de las muestras de M_t de forma que $\sum_{i=1}^T \pi_t^i = 1$. Para ello

   $$\pi_t^i \leftarrow \frac{\pi_t^i}{\sum_{i=1}^T \pi_t^i}.$$

Tabla 5.1: Filtro bootstrap.

5.3 Justificación teórica

Una justificación teórica del funcionamiento del algoritmo se formula en (Carpenter, Clifford, y Fernhead 1997; Pitt y Shephard 1997). Presentamos a continuación esta justificación.

- **Fase de predicción**

  En esta fase se obtiene un conjunto de muestras de la función de densidad $p(\mathbf{x_t} \mid Z^{t-1}, A^{t-1})$, que representa el estado del sistema, una vez actualizada la ultima acción $\mathbf{a}_{t-1}$. Para ello usamos el modelo de movimiento y el conjunto de muestras M_{t-1} del instante anterior para construir la siguiente función de densidad (Pitt y Shephard 1997)

$$\hat{p}(\mathbf{x}_t \mid Z^{t-1}, A^{t-1}) = \sum_{i=1}^{N} p(\mathbf{x}_t \mid \mathbf{x}_{t-1} = \mathbf{m}_{t-1}^i, \mathbf{a}_{t-1}). \tag{5.3}$$

  Esta ecuación proporciona una mezcla de distribuciones que aproxima $p(\mathbf{x_t} \mid Z^{t-1}, A^{t-1})$. Para muestrear esta distribución es posible utilizar el método de *muestreo estratificado* en el que se extrae una muestra de cada componente de la mezcla. Para ello se aplica el modelo de movimiento a cada una de las N muestras de M_{t-1}, para obtener $\tilde{M}_t$.

- **Fase de actualización**

  En la segunda fase se debe utilizar el modelo de observación para obtener las muestras M_t de la distribución a posteriori $p(\mathbf{x}_t \mid Z^t, A^{t-1})$. La siguiente distribución aproxima esta densidad

$$p(\mathbf{x}_t \mid Z^t, A^{t-1}) = \alpha p(\mathbf{z}_t \mid \mathbf{x}_t)\hat{p}(\mathbf{x}_t \mid Z^{t-1}, A^{t-1}). \tag{5.4}$$

  La técnica del *muestreo por rechazo* (ver apéndice A) puede utilizarse para muestrear esta distribución. Esto es debido a que la densidad $\hat{p}(\mathbf{x}_t \mid Z^{t-1}, A^{t-1})$ puede muestrearse (de hecho, el conjunto $\tilde{M}_t$ es un conjunto de muestras correctas de esta distribución) y la probabilidad $p(\mathbf{z}_t \mid \mathbf{x}_t)$ puede calcularse (mediante el modelo de observación).

  El conjunto de muestras resultantes y sus pesos, M_t, representa la distribución a posteriori.

5.4 Aplicación a la estimación de elementos topológicos

Presentamos en esta sección resultados experimentales que muestran cómo se puede aplicar el enfoque de estimación bayesiana temporal y el filtro *bootstrap* a la estimación y seguimiento

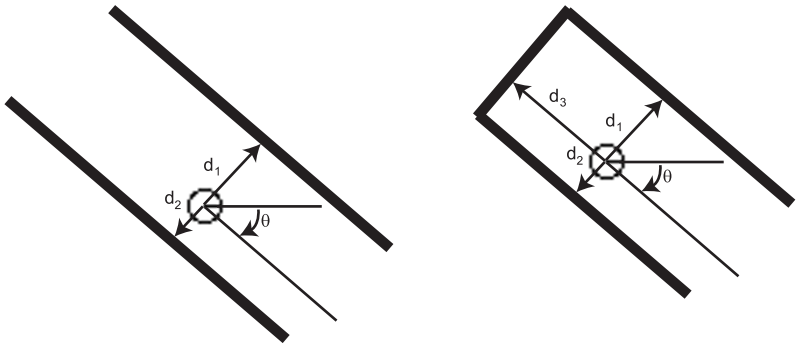


Figura 5.2: Características topológicas usadas en el trabajo: pasillos y finales de pasillo. Los pasillos quedan definidos con tres parámetros (distancia a una y otra pared y orientación) y los finales de pasillo con cuatro (distancias a las paredes y al final del pasillo y orientación).

robusto de estas características topológicas, en concreto, pasillos y finales de pasillos (ver figura 5.2). Los modelos de observación y movimiento que se utilizan son los que se plantean en el capítulo 4. Los parámetros del sistema son distancias a las paredes y orientación local del robot con respecto al pasillo. Esta metodología es genérica y permite ser aplicada a otro tipo de características topológicas, como conexiones entre pasillos, etc.

La obtención de características estables y robustas del entorno en un robot móvil es el paso previo para una posterior extracción autónoma de mapas del entorno, localización en el mismo o navegación de una localización a otra (Kortenkamp, Bonasso, y Murphi 1998). Existe una amplia colección de trabajos en los que se proponen métodos para filtrar las lecturas de sonares y obtener características geométricas elementales (Drumheller 1987; Barshan y Kuc 1990; McKerrow 1993). Sin embargo, las características obtenidas en todos ellos son muy locales, como aristas, esquinas o segmentos, y esta propia localidad hace que sean muy sensibles al ruido, poco robustas y poco estables. Esto último se acentúa en entornos dinámicos y variables, del tipo en los que suelen evolucionar estos robots.

Para comprobar la técnica propuesta se han realizado una serie de experimentos en los entornos simulados que aparecen en la figura 5.3. En todos los experimentos el robot se mueve evitando obstáculos a una velocidad de 25 cm/s y realiza una lectura de sensores cada 0.25 segundos. Cada vez que se realiza una lectura se ejecuta un paso del filtro *bootstrap*. La velocidad del algoritmo es aceptable, funcionando a $1/4$ del tiempo real con $N = 300$ muestras en un procesador Pentium II. Futuras optimizaciones del código harán posible la ejecución del algoritmo en tiempo real.

En los siguientes apartados veremos ejemplos del funcionamiento del algoritmo en distintos instantes de tiempo. Dibujaremos las muestras generadas por el algoritmo sobre el

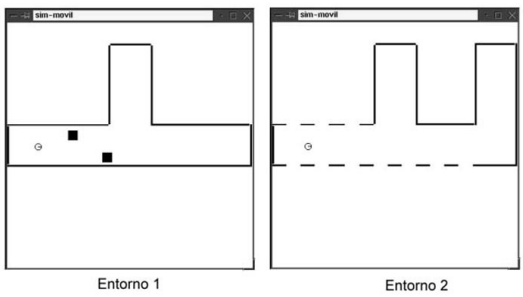


Figura 5.3: Algunos de los entornos de prueba en los que se han realizado los experimentos. El entorno 1 consiste en un pasillo con dos obstáculos y el 2 un pasillo con múltiples puertas.

entorno real, y cada muestra tendrá un tono de gris proporcional a su probabilidad, siendo más oscuras cuanto mayor probabilidad tengan.

- **Experimento 1**

  Fase de inicialización

  En la figura 5.4 se puede observar el proceso de inicialización del conjunto de muestras de la característica *pasillo*. Recordemos que cuanto más oscura es la muestra mayor probabilidad asociada tiene. El número de muestras utilizadas es $N = 300$. El tiempo de una instantánea a otra es de 1 segundo.

  Seguimiento de pasillos con obstáculos

  En la figura 5.5 se puede ver la continuación de la situación anterior. Una vez centrado el conjunto de muestras alrededor del pasillo real todas las muestras bajan en verosimilitud al pasar el robot frente a un obstáculo (instantánea 8). En la instantánea 9, el ruido gaussiano del modelo de movimiento genera algunas muestras de pasillos más cercanos al obstáculo, pero la media de la distribución no cambia de forma sensible. En las instantáneas 10 y 11 el robot ha superado el obstáculo, vuelven a realizarse lecturas del pasillo real y la distribución se mueve otra vez hacia el pasillo real.

- **Experimento 2**

  En la figura 5.6 se comprueba el funcionamiento del algoritmo siguiendo finales de pasillo en un entorno complicado como el número 2. La dificultad de este entorno se debe a que el robot no obtiene ninguna lectura del pasillo cuando pasa frente a las puertas. El número de muestras de este experimento es el mismo que el anterior ($N = 300$). En la instantánea 1 se ven las muestras ya inicializadas. De la instantánea 1

a la 9 el robot pasa frente a diversas puertas, con lo que las muestras han evolucionado según el modelo dinámico del robot, sin ser reforzadas por lecturas del sonar. Sin embargo, cuando el robot vuelve a detectar el pasillo vemos como en las instantáneas 10 y 13 el algoritmo vuelve a reforzar las muestras correctas.

- **Experimento 3**

  Un último conjunto de pruebas (figuras 5.7 y 5.8) se ha realizado a partir de las lecturas de sonar tomadas por PIXIE evolucionando dentro de un pasillo. Al igual que en los experimentos simulados, el robot se mueve evitando obstáculos a una velocidad de 25 cm/s y realiza una lectura de sensores cada 0.25 segundos. El número de muestras también es de 300. Cada vez que se realiza una lectura se ejecuta un paso del filtro *bootstrap*. Las lecturas son especialmente ruidosas, debido a las columnas presentes en uno de los lados del pasillo.

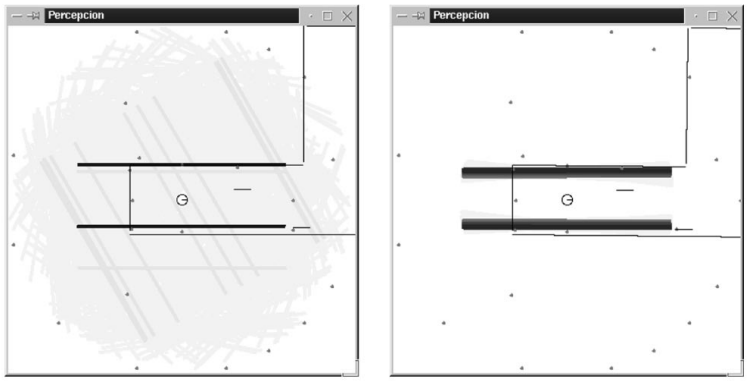


Figura 5.4: **Experimento 1.** Inicialización de la característica *pasillo* en el entorno 1.

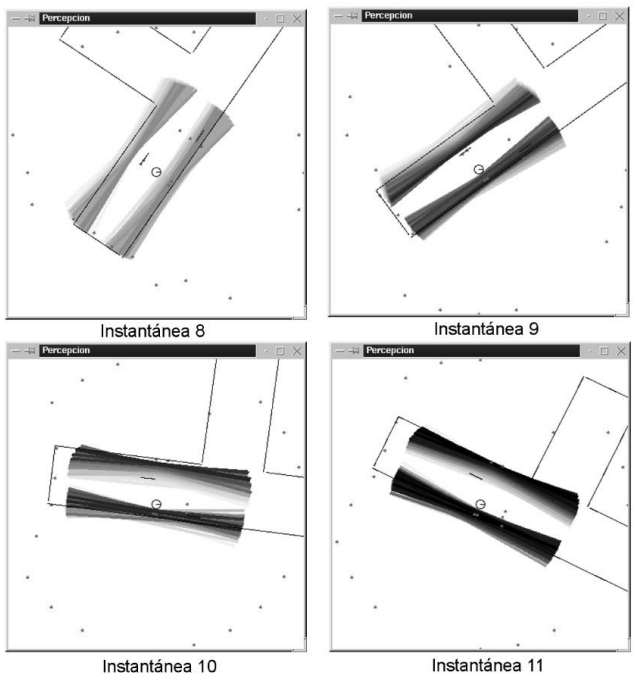


Figura 5.5: **Experimento 1.** Seguimiento del pasillo moviéndose el robot en el entorno 1. El obstáculo puede verse como un segmento recto paralelo al pasillo.

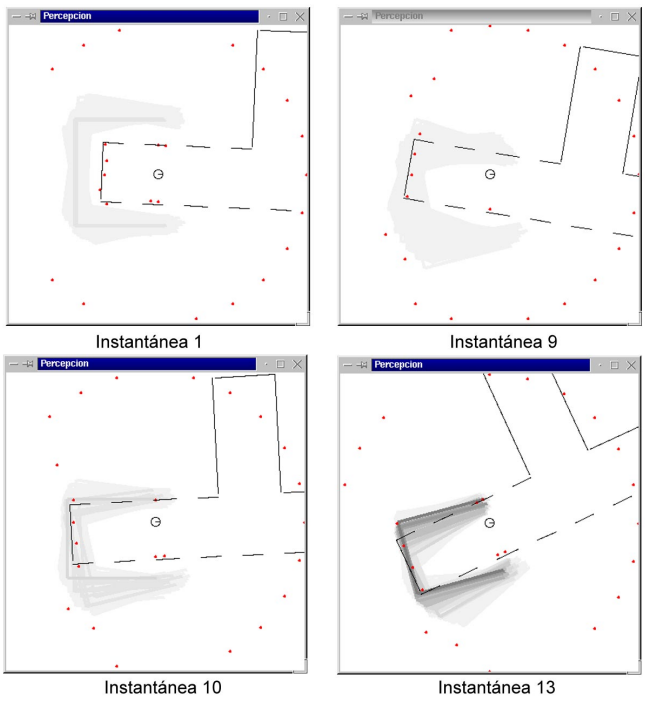


Figura 5.6: **Experimento 2.** Seguimiento de finales de pasillo moviéndose el robot en el entorno 2.

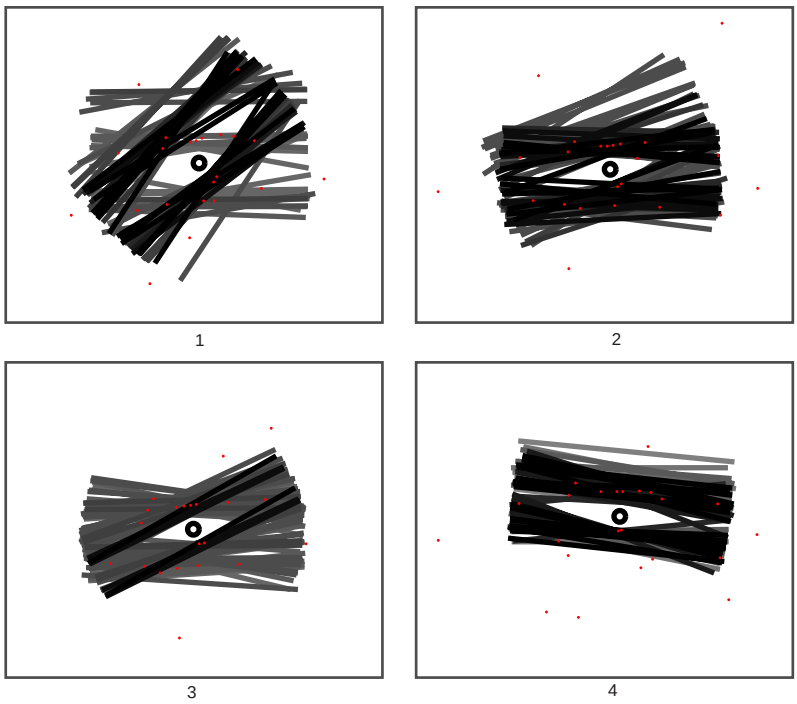


Figura 5.7: **Experimento 3.** Muestras generadas siguiendo un pasillo en datos reales.

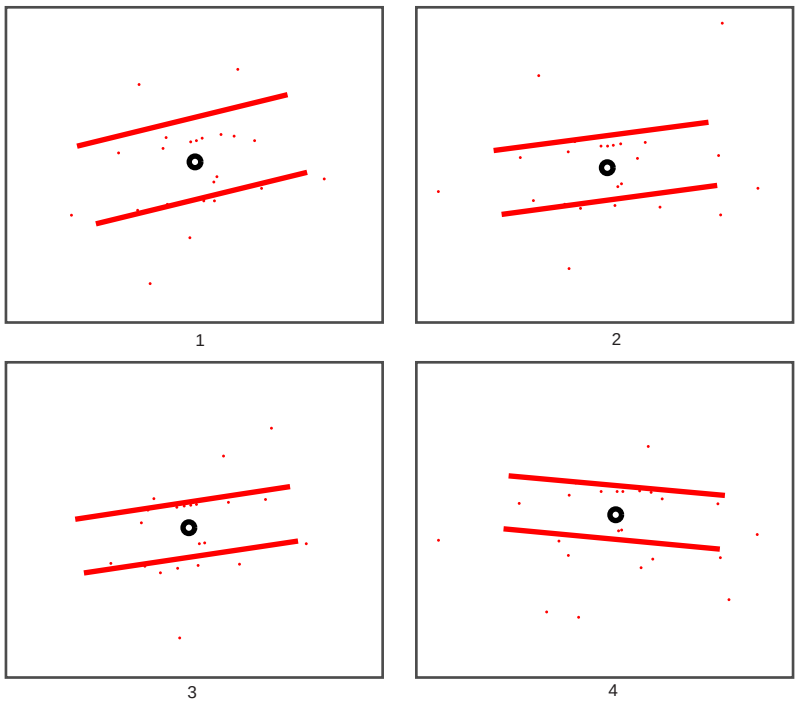


Figura 5.8: **Experimento 3.** Posiciones medias estimadas del pasillo. Los puntos representan las lecturas realizadas por el robot.

5.5 Aplicación a la localización

Un segundo grupo de experimentos aplica el filtro *bootstrap* al problema de localización propiamente dicho. Se trata de estimar (localización global) y realizar un seguimiento la posición global $\mathbf{x} = (x, y, \theta)$ del robot en un entorno conocido de antemano.

Cada muestra $\mathbf{m}_i$ del algoritmo *bootstrap* representa entonces una posible posición (x_i, y_i, θ_i) del robot en el entorno conocido. El algoritmo itera aplicando el modelo de movimiento a cada una de las muestras y actualizando su peso proporcionalmente a la verosimilitud de que las lecturas de sonar se hayan realizado desde la posición global definida por la muestra.

Se han realizado una gran variedad de experimentos, variando distintos elementos del filtro, para comprobar su funcionamiento.

Al igual que en los experimentos anteriores, el robot se mueve por el entorno evitando obstáculos a una velocidad de 25 cm/s y se obtienen lecturas de sus sonares y medidas de odometría cada 0.25 segundos (4 veces por segundo). En los experimentos 1 al 5 los datos se han tomado del simulador. En el experimento 6 los datos se han tomado de PIXIE. El modelo de entorno es el mismo para ambos casos.

- **Experimento 1**

  En un primer experimento (figuras 5.9 5.10,5.11,5.12) se realiza una localización global con el robot moviéndose a lo largo de un pasillo (zona de alta incertidumbre en la percepción) y entrando en una zona más abierta. Las distintas figuras representan el comportamiento del filtro *bootstrap* con 1000, 343 y 125 muestras respectivamente. En cada figura se muestran algunas instantáneas de la localización, representándose todas las muestras de la distribución con un tono de gris proporcional a su verosimilitud. En la parte inferior de las figuras se muestra una gráfica con la evolución del error absoluto medio de cada parámetro a estimar con respecto al tiempo (eje horizontal).

  Se puede observar que en una primera fase, mientras el robot se mueve por el pasillo, las muestras se distribuyen a lo largo del mismo y el error de estimación de la posición x del robot es muy elevado. Esto es debido a que la verosimilitud de las lecturas es alta en todas las posiciones centrales del pasillo, siempre que el robot esté orientado hacia la derecha. Por eso, se puede comprobar que el error de estimación de la posición y y de la orientación θ es bastante bajo desde los primeros instantes de tiempo.

  En el momento en que el robot entra en la zona abierta a su izquierda (alrededor del instante 18), la localización se realiza correctamente, siempre que el número de muestras sea suficiente (1000 y 343). En el caso del experimento con 125 muestras los resultados no son buenos, debido a que el número de muestras no es suficiente para ocupar toda la zona verosímil del espacio de estados.

Por último, en la figura 5.13 se muestra la evolución de la entropía de la distribución de probabilidad en cada instante de tiempo.

Hay que resaltar que en los casos en los que las muestras terminan concentrándose alrededor de una posición la entropía se reduce de forma acorde con esta concentración de la distribución. En el caso del experimento con 125 muestras, podemos ver que la entropía siempre es constante, ya que no se produce esta concentración de la distribución.

- **Experimento 2**

  En la segunda prueba se introduce una modificación en el filtro *bootstrap*, para intentar resolver el problema del número de muestras escaso. Consiste en modificar la fase de remuestreo para generar aleatoriamente un porcentaje de las muestras.

  En el caso de la figura 5.14, el 20 por ciento de las muestras se genera de nuevo aleatoriamente en cada iteración. De esta forma, si la distribución no está centrada en la posición real del robot, es posible que alguna de las muestras aleatorias caiga cerca de esta posición real del robot. Al calcular la verosimilitud, esta muestra vencerá al resto e inclinará la distribución hacia ella.

  Se puede observar que este comportamiento es el que sucede en la serie temporal, aunque hay que esperar al instante 38 para que ocurra.

- **Experimento 3**

  En este experimento (figura 5.15) se inicializa el robot en una esquina de la habitación, una zona de baja ambigüedad, ya que las lecturas de los sonares son muy características. Se puede comprobar que la localización es muy rápida, centrándose toda la distribución de muestras en tan sólo 4 instantes de tiempo.

- **Experimento 4**

  En esta prueba (figura 5.16) sucede lo contrario que en el experimento 1. Inicialmente el robot se encuentra en una zona muy característica, con lo que la localización es muy rápida, para pasar posteriormente a vagabundear por el pasillo.

  A partir de la entrada en el pasillo (instante 87) la distribución comienza a dispersarse debido a la ambigüedad de las lecturas en el pasillo.

  En la parte inferior de la figura se muestra la evolución de la entropía de la distribución, comprobándose de nuevo cómo la entropía puede proporcionar un buen estimador de la calidad de la localización.

- **Experimento 5**

  Un último experimento con datos simulados pretende evidenciar uno de los problemas más graves del filtro *bootstrap*. Se trata del problema del colapso, que sucede cuando las muestras están realizando un seguimiento de una distribución multimodal.

  En la figura 5.17 las muestras realizan la localización de un robot moviéndose en zig-zag en una habitación rectangular. La simetría del entorno hace que dos posiciones sean igual de verosímiles, y, de hecho, la distribución se centra pronto en esas dos posibilidades. Sin embargo, después de una evolución de unos 100 instantes de tiempo en donde se mantiene la multimodalidad, se produce una disparidad en el número de muestras de uno de los grupos y, al momento, el filtro colapsa en una de las modas.

  En la figura 5.18 se comprueba cómo la introducción de un porcentaje de muestras aleatorias empeora el problema del colapso.

- **Experimento 6**

  En este experimento se realiza el proceso de localización global sobre un conjunto de lecturas y posiciones leídas por PIXIE, evolucionando en un entorno idéntico al planteado en los experimentos 1 al 4. En la figura 5.19 se muestran las posiciones y las lecturas realizadas, que son mucho más ruidosas que las obtenidas por el simulador (figura 5.9). A pesar de ello, se puede comprobar en la figura 5.20 que el algoritmo de localización funciona perfectamente.

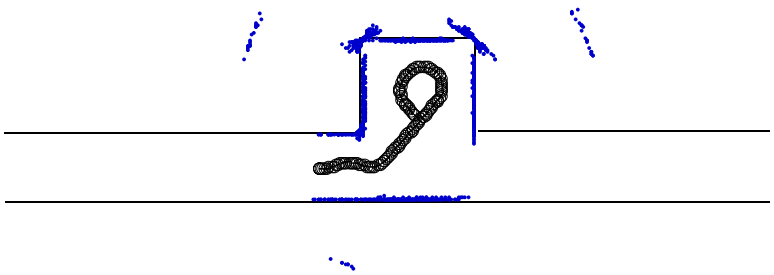
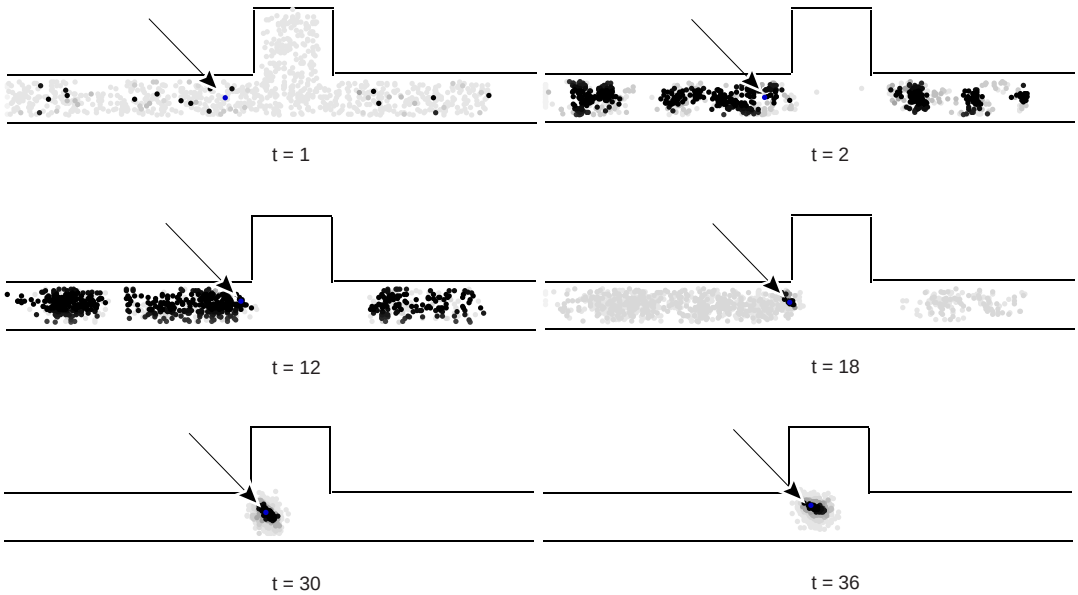


Figura 5.9: Lecturas y posiciones tomadas del simulador, con las que se han realizado los experimentos de localización 1 y 2.



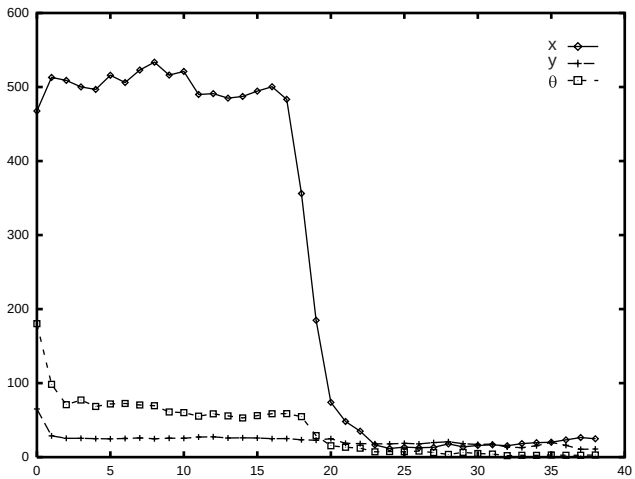


Figura 5.10: **Experimento 1.** Localización en un pasillo (zona de alta ambiguedad). Muestras con un nivel de gris más oscura indican mayores probabilidades de que el robot se encuentre en esa posición. Desviación absoluta media de las posiciones x, y y la orientación θ del robot. Número de muestras=1000.

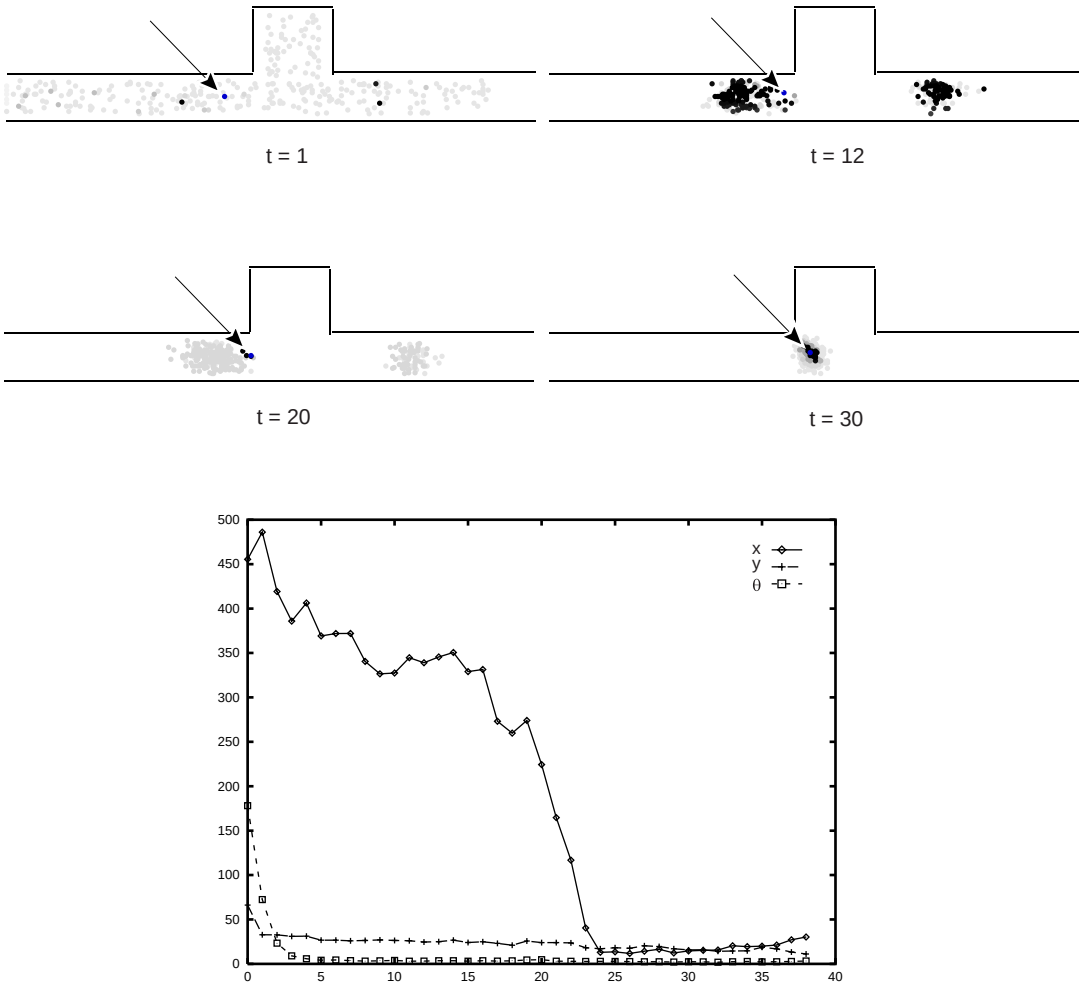


Figura 5.11: **Experimento 1.** Localización en un pasillo (zona de alta ambiguedad). Desviación absoluta media de las posiciones x, y y la orientación θ del robot. Número de muestras=343.

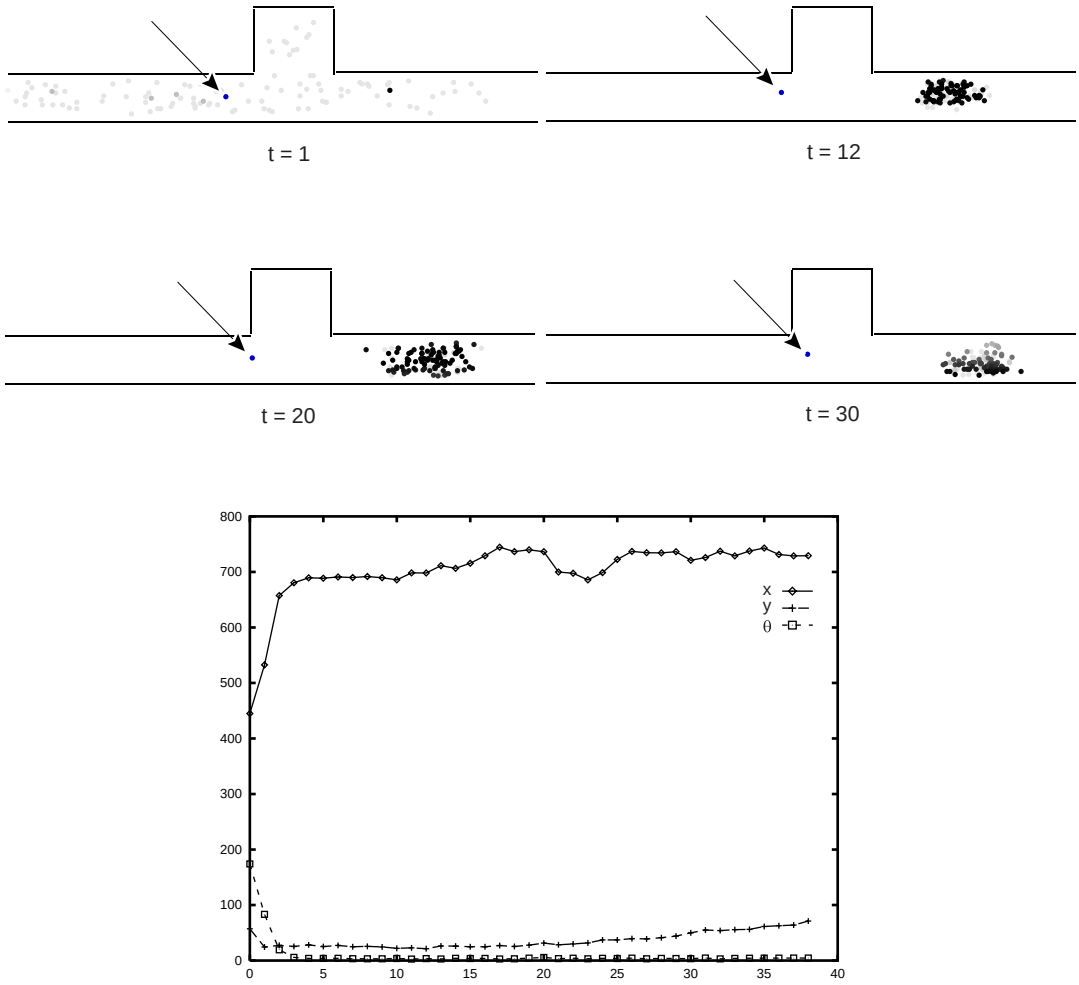


Figura 5.12: **Experimento 1.** Localización en un pasillo (zona de alta ambiguedad). Desviación absoluta media de las posiciones x, y y la orientación θ del robot. Número de muestras=125.

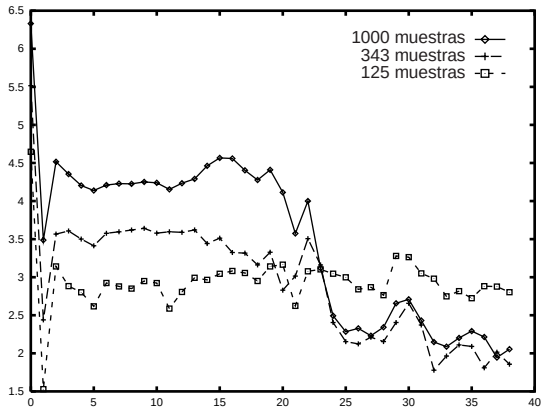
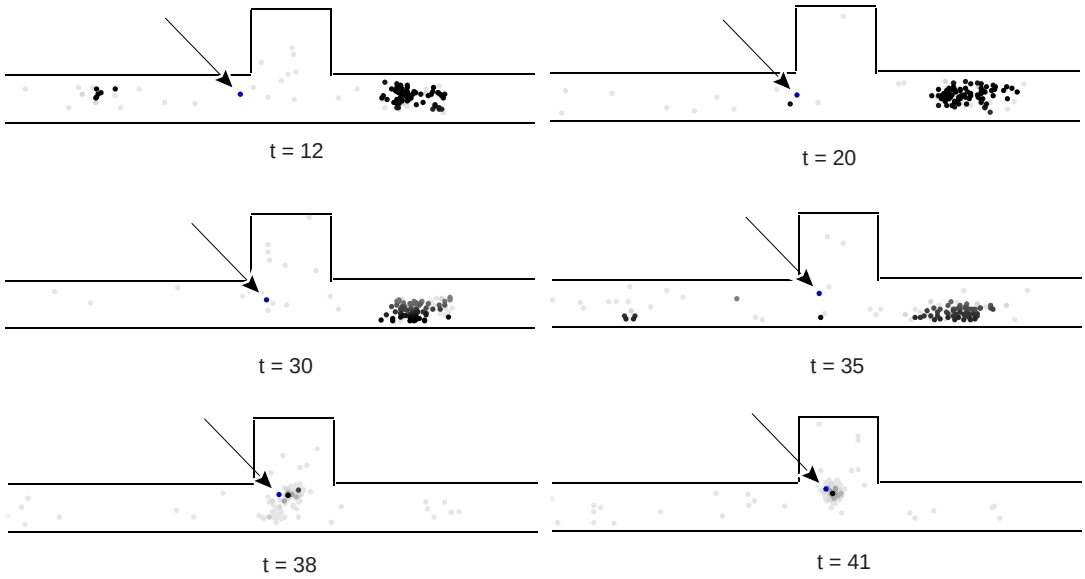


Figura 5.13: **Experimento 1.** Entropía de las distribuciones de muestras en cada instante de tiempo en algunos de los experimentos de localización 5.15, 5.10 y 5.11, en los que se utilizan 1000, 343 y 125 muestras respectivamente.



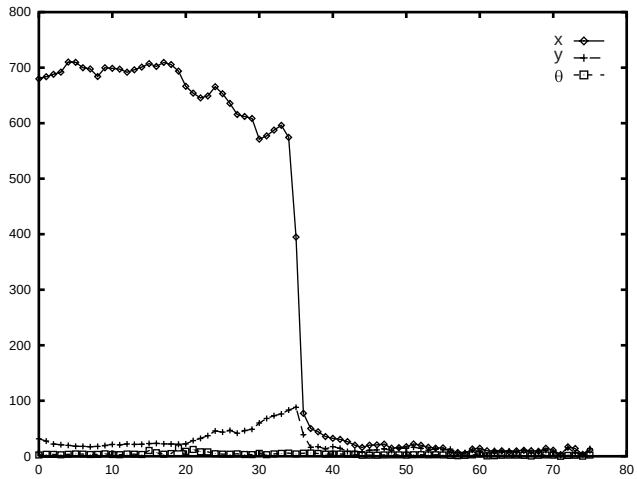
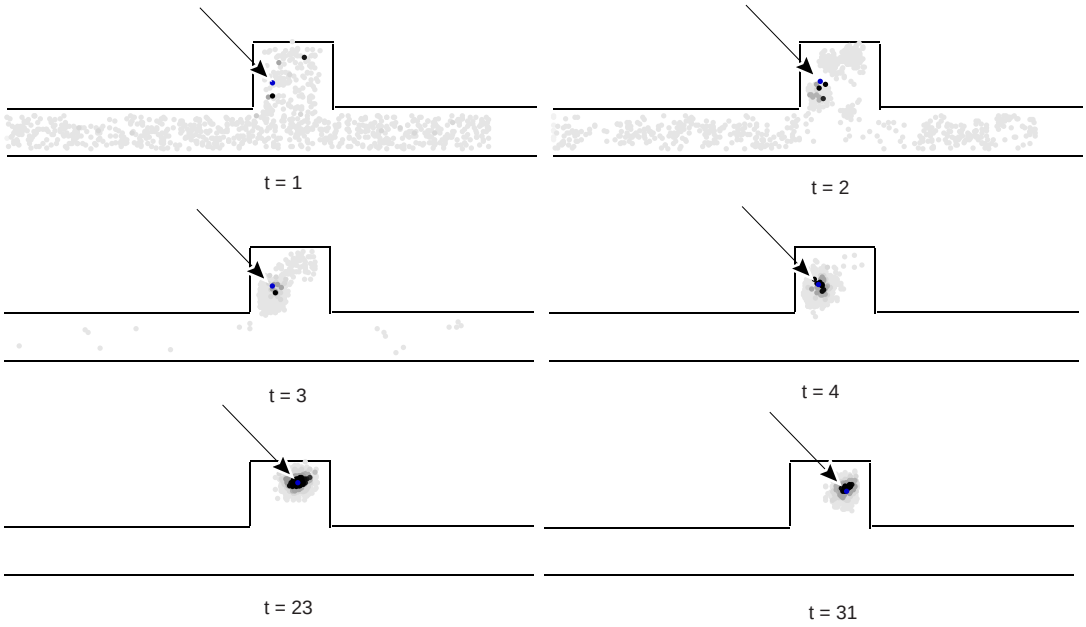


Figura 5.14: **Experimento 2.** Localización en un pasillo (zona de alta ambigüedad). Desviación absoluta media de las posiciones x, y y la orientación θ del robot. Número de muestras=125. Muestras escogidas aleatoriamente = 20 por ciento



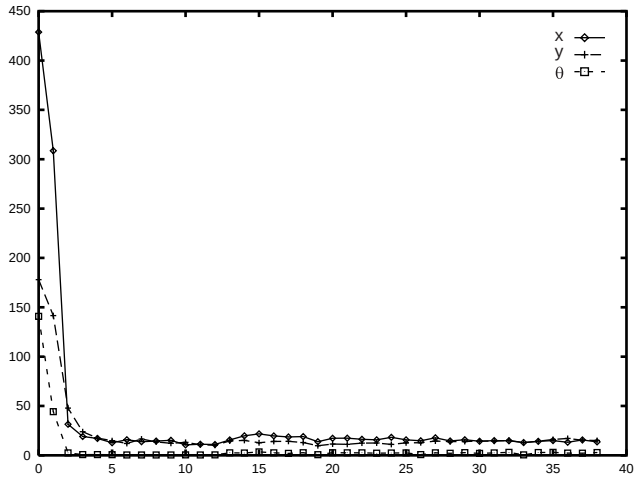


Figura 5.15: **Experimento 3.** Localización en una zona de alta distinguibilidad. Desviación absoluta media de las posiciones x, y y la orientación θ del robot. Número de muestras=1000.

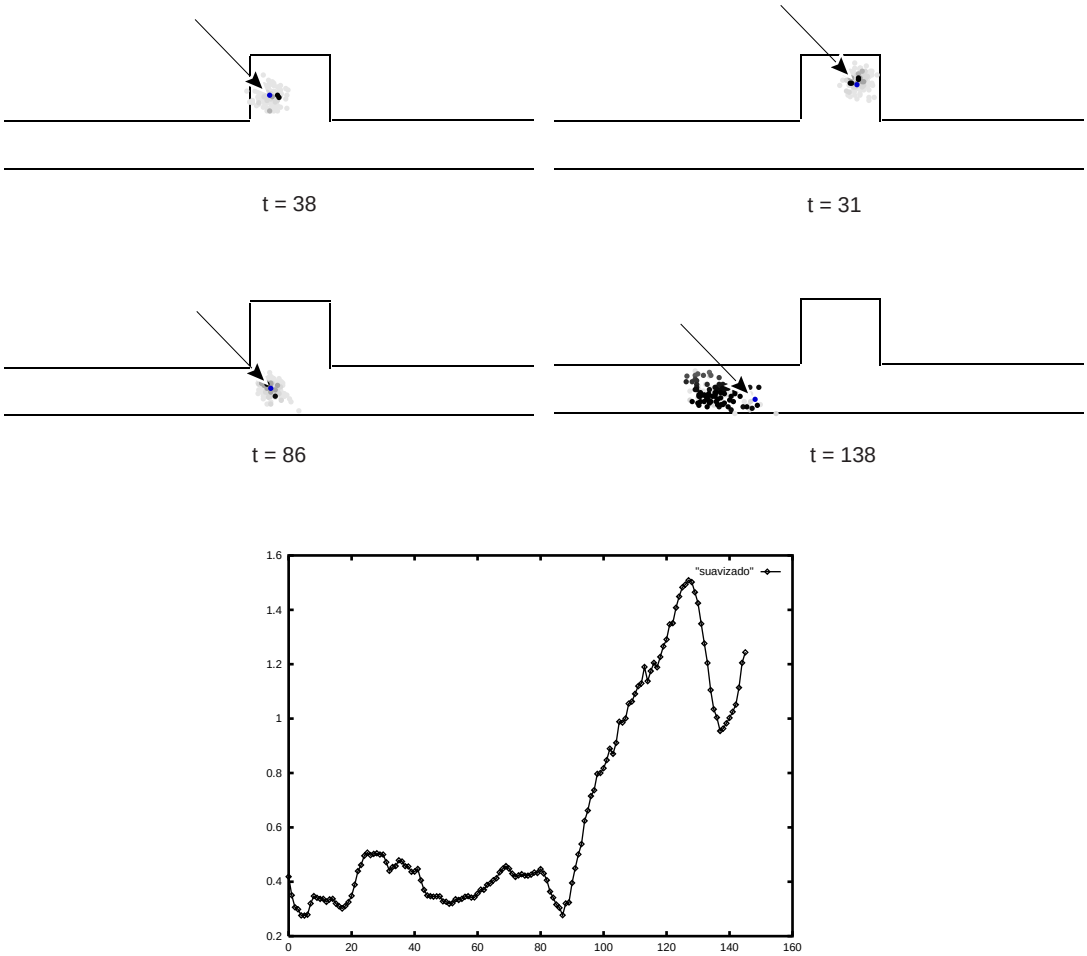


Figura 5.16: **Experimento 4.** Ejemplo de serie temporal que termina en una mala localización. Número de muestras=125. Abajo: representación de la entropía de la distribución de muestras en cada instante de tiempo (suavizada con una ventana de 3 instantes de tiempo) .

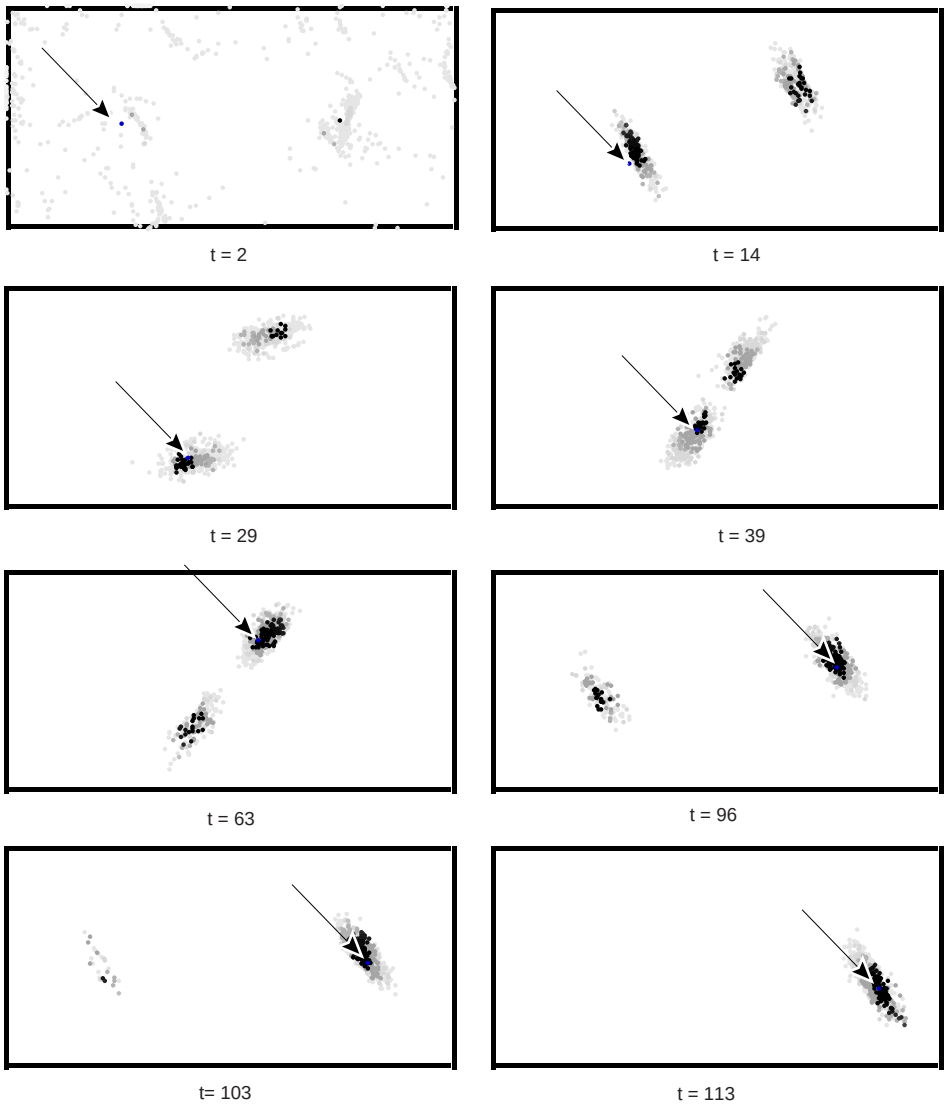


Figura 5.17: **Experimento 5.** Problemas en situaciones simétricas, en donde se produce multimodalidad. Colapso del filtro de bootstrap.

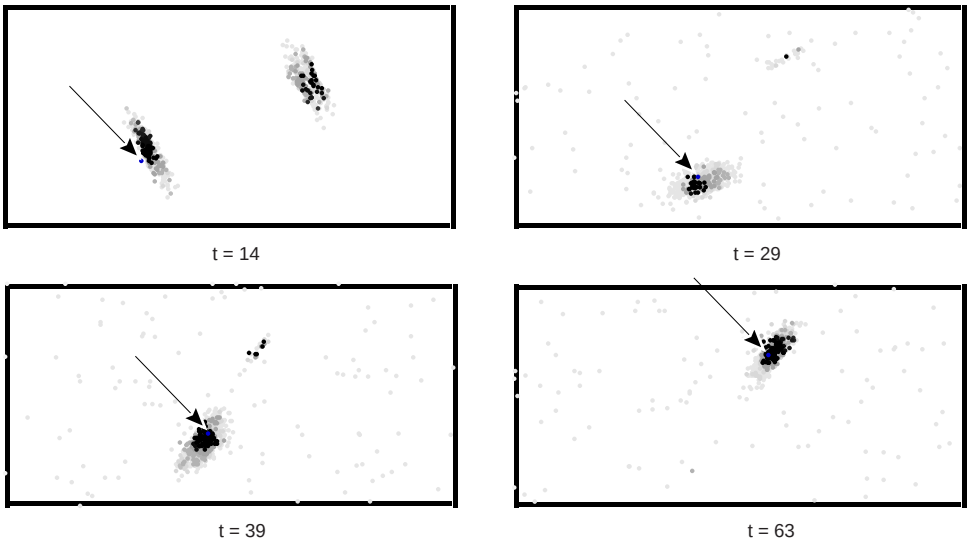


Figura 5.18: **Experimento 5.** La selección de muestras aleatorias no soluciona el problema del colapso.

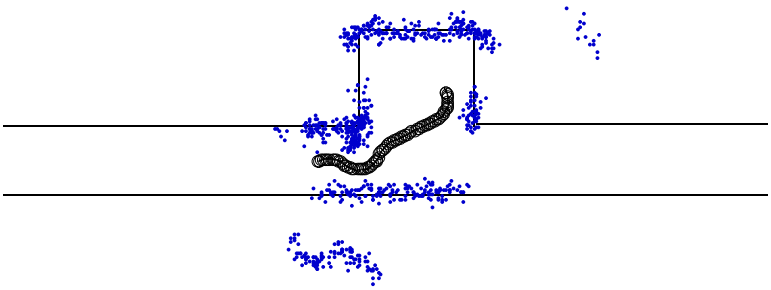


Figura 5.19: **Experimento 6.** Lecturas y posiciones reales, con las que se ha realizado el experimento de localización 6.

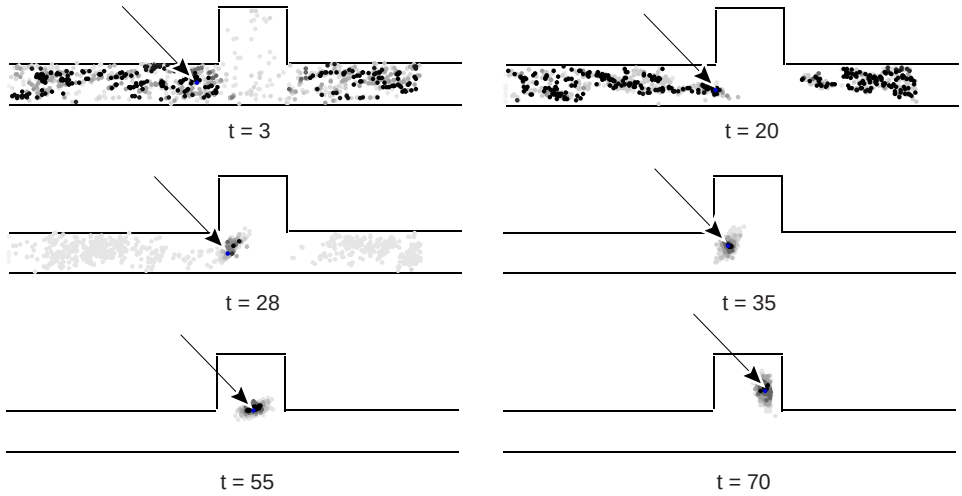


Figura 5.20: **Experimento 6.** Localización global con datos reales de PIXIE. Número de muestras = 1000.

5.6 Discusión

En este capítulo se ha presentado el filtro *bootstrap*, adaptándolo al problema de la localización global de robots móviles. Este filtro se basa en representar la función de probabilidad a posteriori mediante un conjunto de muestras extraídas de la distribución y en realizar un proceso de selección y actualización de las muestras acorde con la estimación bayesiana.

Hemos presentado múltiples ejemplos de los resultados de aplicar este algoritmo al problema de la localización, en sus distintos aspectos de localización de características topológicas en el entorno, localización global del robot y seguimiento de la posición.

Se ha demostrado que el algoritmo es efectivo y robusto, salvo en el caso de que la localización tenga una ambigüedad duradera, como sucede en entornos simétricos o en problemas de localización global en el que el robot no encuentra pronto zonas distinguibles. En estos casos se produce el problema más importante del filtro, el problema del colapso de la distribución.

Se ha probado una estrategia de generación aleatoria de muestras, demostrándose que alivia el problema del colapso en el caso de la localización global, pero no en el de los entornos simétricos.

Capítulo 6

Mapeado basado en el algoritmo EM adaptativo

Para localizar un robot es necesario un mapa del entorno en el que se mueve. Un problema interesante es la construcción automática de dicho mapa.

Esta construcción automática proporciona una mayor fiabilidad al proceso de localización, ya que los modelos bayesianos que se utilizan en este último problema son los mismos que los que se utilizan en el problema del mapeado. Un modelo de entorno construido con el mismo modelo con el que después se va a localizar el robot será más robusto que un modelo construido a mano.

En capítulos anteriores se han definido los mapas paramétricos. Veremos en este capítulo que es posible determinar los parámetros que mejor adaptan un determinado modelo a las lecturas realizadas por el robot en el entorno, *aún sin conocer las posiciones absolutas desde las que se han realizado estas lecturas*.

6.1 Introducción

El algoritmo EM, introducido por Dempster *et al.* (Dempster, Laird, y Rubin 1977), proporciona una técnica iterativa para realizar una estimación de máxima verosimilitud de un conjunto de parámetros en problemas en los que existen *datos ocultos* que dependen estadísticamente de los parámetros a estimar y de los datos observados.

La estimación de mapas del entorno es un ejemplo del tipo de problemas en los que se puede aplicar este algoritmo. Tal y como se detalla en el capítulo 2, el problema se puede formalizar de la siguiente manera. Un robot móvil ejecuta una secuencia acciones de movimiento $(\mathbf{a}_1, \ldots, \mathbf{a}_{N-1})$ por un entorno dado, tomando una serie de observaciones del mismo (lecturas de alcance, por ejemplo) $(\mathbf{z}_1, \ldots, \mathbf{z}_N)$. Estas acciones y mediciones

constituyen los datos observados. Las posiciones del robot $(\mathbf{x}_1, \ldots, \mathbf{x}_N)$ desde las que se han realizado las mediciones no son conocidas, y constituyen los *datos ocultos* del problema. Se trata de estimar los parámetros $\phi = \{d_1, \ldots, d_n\}$ que definen el mapa del entorno de máxima verosimilitud, dadas las acciones y observaciones y considerando los datos ocultos. Para resolver el problema se cuenta con un modelo que calcula la verosimilitud de unas lecturas dadas una posición y unos parámetros del modelo del entorno. También se dispone de un modelo de movimiento que predice la siguiente posición del robot, dada la posición actual y la acción ejecutada.

En el apéndice B se detalla el algoritmo EM. Un resumen de su funcionamiento, aplicado al problema del mapeado, es el siguiente. Sean $X = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$ la secuencia de posiciones del robot (datos ocultos), $Z = (\mathbf{z}_1, \ldots, \mathbf{z}_T)$ la secuencia de observaciones del entorno y $A = (\mathbf{a}_1, \ldots, \mathbf{a}_{T-1})$ la secuencia de acciones realizadas por el robot. Los conjuntos Z y A constituyen los datos observados, y se denota por Y al conjunto total de datos $Y = X \cup Z \cup A$. Los parámetros a estimar son unos parámetros ϕ que determinan el mapa del entorno.

El algoritmo EM busca el mapa del entorno ϕ que maximiza el logaritmo de la función de verosimilitud marginal de las lecturas y acciones observadas sobre todas las posibles posiciones del entorno

$$\ln p(Z, A \mid \phi) = \int_X \ln p(Z, A, X \mid \phi).$$

Tal y como se describe en el apéndice B, se comienza por un mapa inicial ϕ^0 y se van obteniendo nuevas soluciones de forma iterativa. En cada iteración se parte del mapa obtenido en la iteración anterior, ϕ^k, para obtener el siguiente valor de los parámetros del mapa que maximizan la *verosimilitud esperada* de los datos observados y los datos ocultos, dado la estimación anterior del mapa y los datos observados. Formalmente, se buscan los parámetros del mapa que cumplen

$$\begin{aligned} &\arg\max_{\phi} E[\ln p(Z, A, X \mid \phi) \mid Z, A, \phi^k)] = \\ &\quad = \arg\max_{\phi} \int_X \ln[p(X, Z, A \mid \phi)] p(X \mid Z, A, \phi^k). \end{aligned}$$

La última ecuación no se utiliza realmente en el algoritmo, se introduce para clarificar el término condicional de la primera expresión.

La iteración del algoritmo EM consiste en un paso de estimación (paso E), en el que se obtienen los valores esperados de las posiciones del entorno, seguido de un paso de maximización (paso M). En cada iteración la verosimilitud aumenta de forma monótona. En la tabla 6.1 se formula el algoritmo EM (ver apéndice B para mayor detalle).

Algoritmo EM

1. Sea ϕ^0 la solución inicial
2. Repetir hasta la convergencia
   (a) Paso E:
   Calcular el valor esperado de la verosimilitud de los datos completos, condicionados por los datos observados y por la solución actual ϕ^k

$$Q(\phi \mid \phi^k) = E[\ln p(Z, A, X \mid \phi) \mid Z, A, \phi^k].$$

   (b) Paso M:

$$\phi^{k+1} \leftarrow \arg\max_{\phi} Q(\phi \mid \phi^k).$$

Tabla 6.1: Formulación del algoritmo *estimación-maximización*.

6.2 Enfoque muestral del algoritmo EM

En un gran número de problemas no es posible calcular analíticamente los pasos de estimación y maximización, por lo que hay que utilizar otro tipo de enfoques.

Se han propuesto algunos enfoques que implementan el paso de estimación utilizando técnicas de muestreo (ver (Cappe, Doucet, Lavielle, y Moulines 1999) para un resumen de las distintas propuestas genéricas). En resumen, todos estos trabajos utilizan el valor estimado de los parámetros en el instante anterior ϕ^k y los datos observados $z_1, \ldots, z_T$ para obtener un conjunto de N muestras de los datos ocultos. Los valores ocultos esperados y la función Q se pueden estimar entonces a partir de las N muestras.

En el marco de la estimación bayesiana temporal, algunos trabajos muy recientes utilizan los algoritmos de estimación muestral como estimadores del paso de estimación. Por ejemplo, North y Blake (North y Blake 1998) utilizan el algoritmo EM para estimar los parámetros que determinan el modelo de movimiento de objetos en secuencias de aprendizaje.

En nuestro caso, dado que la localización es frecuentemente multimodal, utilizar como estimador la posición esperada, calculando la media de las muestras, introduciría errores

muy importantes. Proponemos, en su lugar, utilizar un estimador de posición robusto como es la posición de máxima probabilidad.

En cuanto al paso de maximización, no conocemos propuestas genéricas en la literatura relacionada con el algoritmo EM que permitan realizar la maximización cuando no se dispone de una fórmula analítica. En el campo de la visión artificial, sin embargo, algunos trabajos proponen utilizar técnicas adaptativas para encontrar parámetros de máxima verosimilitud, dados unos datos (ver (Hornegger y Niemann 1997) para una introducción a estos métodos). Es posible aplicar estas técnicas al algoritmo EM, si se considera que en la fase de maximización se han obtenido los valores ocultos esperados.

Para aplicar el algoritmo EM al problema de estimación del mapa del entorno es necesario introducir estas adaptaciones. Un resumen de la propuesta que se plantea en este capítulo se muestra en la tabla 6.2.

Algoritmo EM muestral

A partir de un valor anterior de los parámetros a estimar ϕ^k, de los datos observados $Z = \{\mathbf{z}_1, \ldots, \mathbf{z}_T\}$ y de las acciones del robot $A = \{\mathbf{a}_1, \ldots, \mathbf{a}_{T-1}\}$ se obtienen las posiciones de máxima verosimilitud del robot $\hat{X} = \{\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_T\}$ y se refina la estimación de los parámetros, obteniéndose ϕ^{k+1}.

1. **Fase de estimación**:

   Utilizar el filtro *bootstrap* suavizado con el algoritmo de Kitagawa para obtener las posiciones de máxima verosimilitud del robot $\hat{X} = \{\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_T\}$, dada la función de verosimilitud de las observaciones $p(\mathbf{z}_i \mid \mathbf{x}_i, \phi^k)$ y el modelo de movimiento $p(\mathbf{x}_i \mid \mathbf{x}_{i-1}, \mathbf{a}_{i-1})$.

2. **Fase de maximización**:

   Estimar con una búsqueda aleatoria adaptativa los parámetros del mapa del entorno que maximizan la verosimilitud de las lecturas y las posiciones esperadas

$$\phi^{k+1} \leftarrow \arg\max_{\phi} \ln p(Z, A, \hat{X} \mid \phi)$$

Tabla 6.2: Versión muestral del algoritmo EM para estimación de mapas del entorno.

6.2.1 Estimación de posiciones esperadas

El filtro *bootstrap* realiza la estimación muestral de la densidad a posteriori $p(\mathbf{x}_t \mid Z^t, A^{t-1})$, representando la función por un conjunto de muestras $\{(\mathbf{m}_t^i, \pi_t^i),\ i = 1 \ldots N\}$. Esta función de densidad representa toda la información conocida acerca de las posiciones del robot, $\mathbf{x}_t$, dada la historia de sus observaciones $Z^T = \{\mathbf{z}_1, \ldots, \mathbf{z}_t\}$ y de sus acciones $A^{T-1} = \{\mathbf{a}_1, \ldots, \mathbf{a}_{t-1}\}$. A partir de cada conjunto de muestras es posible estimar la esperanza de la distribución a posteriori para ese instante de tiempo utilizando las muestras y sus pesos asociados

$$E[\mathbf{x}_t \mid Z^t, A^{t-1}] \approx \hat{\mathbf{x}}_t = \sum_{i=1}^{N} \pi_t^i \mathbf{m}_t^i \tag{6.1}$$

Estos valores esperados, sin embargo, no son representativos en el caso de una distribución multimodal. Por ello proponemos utilizar en su lugar la muestra de mayor verosimilitud de cada instantánea

$$\hat{\mathbf{x}}_t = \mathbf{m}_t^i \mid \pi_t^i \geq \pi_t^j \ \forall j = 1 \ldots N \tag{6.2}$$

como el conjunto de soluciones de la fase de estimación del algoritmo EM.

Es posible, sin embargo, ajustar aun más estas estimaciones. Las estimaciones $\hat{\mathbf{x}}_t$ recogen únicamente la información de las observaciones del robot hasta el instante t. Estas estimaciones suelen tener un alta variabilidad, ya que durante el seguimiento en línea es normal que se generen varias hipótesis de localización para un mismo instante de tiempo (recordemos otra vez que se estamos tratando con distribuciones multimodales). Pero también sucede con frecuencia que todas las hipótesis menos una desaparecen cuando se hace evidente que corresponden a distracciones o errores de estimación. Por ello, cuando se dispone de la secuencia completa de observaciones (hasta el instante T), es posible utilizar toda esta información para mejorar estos estimadores, ya que se puede eliminar variabilidad producidas por distractores temporales. Se trata entonces de estimar la densidad de probabilidad $p(\mathbf{x}_t | Z^T, A^{T-1})$.

Isard (Isard y Blake 1998b), recogiendo la formulación inicial de Kitagawa (Kitawa 1996), formula un algoritmo para muestrear estas distribuciones. A continuación presentamos una adaptación de este algoritmo que considera la secuencia de acciones $(\mathbf{a}_1, \ldots, \mathbf{a}_{T-1})$ realizadas por el robot.

El algoritmo consiste en un barrido hacia adelante (equivalente al cálculo de los valores α en el algoritmo Baum-Welch), en el que se generan las muestras $\{(\mathbf{m}_t^i, \pi_t^i)\}$ para $t = 1, \ldots, T$, utilizando el filtro *bootstrap*. Una vez generadas las muestras se reajustan los pesos π_t^i, para que representen la evidencia proporcionada por las observaciones posteriores $(\mathbf{z}_t, \ldots, \mathbf{z}_T, \mathbf{a}_t, \ldots, \mathbf{a}_{T-1})$.

Para hacer más compacta la notación, introducimos la siguiente modificación. Sean $Z_j^k = (\mathbf{z}_j, \ldots, \mathbf{z}_k)$ y $A_j^k = (\mathbf{a}_j, \ldots, \mathbf{a}_k)$ las secuencias de observaciones y acciones desde el instante j hasta el instante k (con $j \geq k$). Con esta notación podemos expresar la probabilidad de una posición $\mathbf{x}_t$ dado todo el conjunto de observaciones y acciones como

$$\begin{aligned} & p(\mathbf{x}_t \mid Z_1^T, A_1^{T-1})\, \alpha \\ & \quad \alpha\, p(\mathbf{x}_t, Z_t^T, A_t^{T-1} \mid Z_1^{t-1}, A_1^{t-1}) = \\ & \quad p(Z_t^T, A_t^{T-1} \mid \mathbf{x}_t)\, p(\mathbf{x}_t \mid Z_1^{t-1}, A_1^{t-1}) \end{aligned} \tag{6.3}$$

Esta reordenación permite que las posiciones muestreadas $\mathbf{m}_t^i$ permanezcan fijas después del paso de suavizado. Recordemos que el conjunto $\{\mathbf{m}_t^i\}$ es, aproximadamente, una muestra correcta de la distribución $p(\mathbf{x}_t | Z_1^{t-1}, A_1^{t-1})$, por lo que, al reemplazar los pesos originales de las muestras π_t^i por los pesos suavizados

$$\psi_t^i = p(Z_t^T, A_t^{T-1} \mid \mathbf{x}_t = \mathbf{m}_t^l) \tag{6.4}$$

El conjunto $\{(\mathbf{m}_t^i, \psi_t^i)\}$, una vez normalizado, aproximará la distribución requerida $p(\mathbf{x}_t \mid Z_1^T, A_1^{T-1})$.

Un algoritmo recursivo para calcular las funciones de densidad $p(Z_t^T, A_t^{T-1} \mid \mathbf{x}_t)$ se puede formular matemáticamente como sigue

$$p(Z_t^T, A_t^{T-1} \mid \mathbf{x}_t) = p(\mathbf{z}_t \mid \mathbf{x}_t)\, p(Z_{t+1}^T, A_t^{T-1} \mid \mathbf{x}_t) \tag{6.5}$$

$$p(Z_{t+1}^T, A_t^{T-1} \mid \mathbf{x}_t) = \int_{\mathbf{x}_{t+1}} p(Z_{t+1}^T, A_{t+1}^{T-1})\, p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \mathbf{a}_t) \tag{6.6}$$

Una implementación concreta requiere la derivación de una aproximación δ_t^i a la función $p(Z_{t+1}^T, A_t^{T-1} \mid \mathbf{x}_t = \mathbf{m}_t^i)$. La integral se puede aproximar mediante un sumatorio

$$p(Z_{t+1}^T, A_t^{T-1} \mid \mathbf{x}_t = \mathbf{m}_t^i) \approx \delta_t^i = \sum_{l=1}^{N} \psi_{t+1}^i \frac{p(\mathbf{x}_{t+1} = \mathbf{m}_{t+1}^l \mid \mathbf{x}_t = \mathbf{m}_t^i, \mathbf{a}_t)}{\gamma_t^i}, \tag{6.7}$$

donde

$$\psi_{t+1}^i = p(Z_{t+1}^T, A_{t+1}^{T-1} \mid \mathbf{x}_{t+1} = \mathbf{m}_{t+1}^l) \tag{6.8}$$

$$\gamma_t^i = \sum_{k=1}^{N} \pi_t^k\, p(\mathbf{x}_{t+1} = \mathbf{x}_{t+1}^l \mid \mathbf{x}_t = \mathbf{m}_t^k, \mathbf{a}_t) \tag{6.9}$$

representan, respectivamente, los pesos calculados en el paso anterior y una corrección de las probabilidades que tiene en cuenta los pesos π_t^i.

En la tabla 6.3 (pag. 133) se presenta un algoritmo que realiza estos cálculos.

6.2.2 Mapas de máxima verosimilitud

Una vez calculados los parámetros ocultos esperados (posiciones del robot) $\hat{X} = (\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_T)$ a partir de la estimación del mapa anterior ϕ^k y de los datos observados, el algoritmo EM realiza un paso de maximización. En este paso se debe obtener la nueva estimación de los parámetros del mapa, θ^{k+1}, buscando los parámetros de maxima verosimilitud. Éstos son los que maximizan el logaritmo de la verosimilitud de las posiciones esperadas, las lecturas del entorno y las acciones,

$$\phi^{k+1} \leftarrow \arg\max_{\phi} \ln p(Z, A, \hat{X} \mid \phi). \tag{6.10}$$

En la sección 2.6 se formulaba la función de verosimilitud $p(Z, A, \mathbf{x}_1, \ldots, \mathbf{x}_T \mid \phi)$ como

$$p(Z \mid \mathbf{x}_1, \ldots, \mathbf{x}_T, \phi) p(\mathbf{x}_1, \ldots, \mathbf{x}_T \mid Z, A),$$

y, tras desarrollar esta expresión, se llegaba a la ecuación

$$p(Z, A, \mathbf{x}_1, \ldots, \mathbf{x}_T \mid \phi) = p(\mathbf{x}_1) \prod_{t=1}^{T} p(\mathbf{z}_t \mid \mathbf{x}_t, \phi) \prod_{t=2}^{T} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{a}_{t-1}). \tag{6.11}$$

Sustituyendo esta ecuación en la ecuación 6.10 se llega a la formulación final del mapa de máxima verosimilitud

$$\begin{aligned} \phi^{k+1} &\leftarrow \arg\max_{\phi} \left[\ln p(\mathbf{x}_1) \prod_{t=1}^{T} p(\mathbf{z}_t \mid \mathbf{x}_t, \phi) \prod_{t=2}^{T} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{a}_{t-1}) \right] = \\ &= \arg\max_{\phi} \left[\ln p(\mathbf{x}_1) + \sum_{t=1}^{T} \ln p(\mathbf{z}_t \mid \mathbf{x}_t, \phi) + \sum_{t=2}^{T} \ln p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{a}_{t-1}) \right]. \end{aligned}$$

Los términos $p(\mathbf{x}_1)$ y $p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{a}_{t-1})$ son constantes e independientes de los parámetros ϕ, por lo que la expresión final queda como

$$\phi^{k+1} \leftarrow \arg\max_{\phi} \sum_{t=1}^{T} \ln p(\mathbf{z}_t \mid \mathbf{x}_t = \hat{\mathbf{x}}_t, \phi). \tag{6.12}$$

El paso de maximización del algoritmo EM depende, entonces, únicamente del modelo de observación definido (ver sección 4.2). Debido a que la densidad $p(\mathbf{z}_t \mid \mathbf{x}_t, \phi)$ no tiene una fórmula cerrada no es posible derivar este máximo de forma analítica. Por ello, utilizamos un sencillo algoritmo de muestreo aleatorio adaptativo, generando un número grande de muestras de parámetros y realizando una maximización local para cada muestra.

Esta técnica de muestreo aleatorio ha sido utilizada con éxito en gran número de problemas de optimización de visión artificial (Hornegger y Niemann 1997).

Algoritmo de Kitagawa

Dado un conjunto de muestras de posiciones estimadas del robot $\{(\mathbf{m}_t^i, \pi_t^i)\}$ para cada instante de tiempo $t = 1, \ldots, T$, sustituye los pesos π_t^i de las muestras por los pesos ψ_t^i que representan la probabilidad de la muestra $\mathbf{m}_i$ asociada dada *toda* la secuencia de observaciones $p(\mathbf{x}_t = \mathbf{m}_t^i \mid Z_1^T, A_1^{t-1})$.

1. **Inicializar** los pesos ψ_T^i:

$$\psi_T^i = \pi_T^i \quad \text{para } i = 1, \ldots, N$$

2. **Iterar** hacia atrás para toda la secuencia $t = T - 1, \ldots, 1$

   (a) **Calcular** el modelo de movimiento:

$$\alpha_t^{i,j} = p(\mathbf{x}_{t+1} = \mathbf{m}_{t+1}^i \mid \mathbf{x}_t = \mathbf{m}_t^j, \mathbf{a}_t) \quad \text{para } i, j = 1, \ldots, N.$$

   (b) **Calcular** los factores de corrección

$$\gamma_t^i = \sum_{j=1}^{N} \pi_t^j \alpha_t^{i,j} \quad \text{para } i = 1, \ldots, N.$$

   (c) **Aproximar** las variables δ

$$\delta_t^j = \sum_{i=1}^{N} \psi_{t+1}^j \frac{\alpha_t^{i,j}}{\gamma_t^j} \quad \text{para } i = 1, \ldots, N.$$

   (d) **Evaluar** los pesos suavizados

$$\psi_t^i = \pi_t^i \delta_t^i,$$

   normalizar para que $\sum \psi_t^i = 1$ y almacenarlos con la muestra correspondiente

$$\{(\mathbf{m}_t^i, \psi_t^i),\ i = 1, \ldots, N\}.$$

Tabla 6.3: Algoritmo de suavizado de las probabilidades asociadas a las muestras del filtro *bootstrap*.

6.3 Experimentos

Se presenta en esta sección un conjunto de experimentos sobre el algoritmo de Kitagawa para el suavizado de la distribución de muestras y un experimento en el que se muestra el funcionamiento del enfoque EM para el mapeado.

- **Experimento 1.**

  En este experimento (figuras 6.1, 6.2, 6.3, 6.4) se aplica el algoritmo de Kitagawa a la serie temporal de muestras producidas en el experimento 1 de localización. Recordemos que en ese experimento el robot circula por un pasillo hasta que llega a una zona con un espacio abierto a su izquierda. En este momento la distribución de muestras localiza correctamente al robot y durante el resto de instantáneas de la serie la localización se realiza correctamente.

  En la figura 6.1 aparecen las muestras de localización en las instantáneas 1, 30, 67 y 107, junto con una gráfica de la evolución en el tiempo de la entropía de la distribución. En esta gráfica se puede comprobar que, después de una primera fase en la que la distribución tiene un alto grado de dispersión, debido a la ambigüedad de las lecturas del pasillo, en una segunda fase la distribución termina centrándose. Esto sucede alrededor del instante 30, y continua así hasta el final de la serie.

  Es de esperar que el algoritmo de Kitagawa propague hacia atrás la información precisa en la localización de la segunda fase, centrando también la primera fase.

  En la figura 6.2 vemos el resultado de aplicar el algoritmo de Kitagawa. Se muestran dos instantáneas de la serie temporal (la 16 y la 19) antes y después de aplicar Kitagawa. En la parte superior de la figura se comprueba que la distribución está muy dispersa a lo largo del pasillo, indicando que todas esas posiciones tienen igual verosimilitud. Sin embargo, después de aplicar Kitagawa (abajo) vemos cómo obtienen la mayor verosimilitud las muestras cercanas a la posición real del robot, disminuyendo drásticamente la verosimilitud del resto de muestras.

  En la figura 6.3, para resaltar este efecto, se representan las 30 mejores muestras de ambas instantáneas antes (arriba) y después (abajo) de aplicar Kitagawa.

  Por último, la gráfica de la figura 6.4 muestra el error absoluto medio en la posición x y la entropía de la serie temporal, también antes y después de aplicar Kitagawa. Se comprueba que el suavizado reduce drásticamente el error absoluto medio y también reduce la entropía de la distribución.

- **Experimento 2.**

  En un segundo experimento (figura 6.5) se aplica el suavizado a la serie temporal resultante del experimento 5 de localización (pag. 121). Recordemos que esta serie

temporal resultaba de la localización del robot moviéndose en una habitación cerrada. La simetría del entorno resultaba en una distribución multimodal, que persistía bastante tiempo hasta que se hacía evidente el problema del colapso del filtro.

La aplicación del suavizado de Kitagawa a esta serie, como podemos ver en la figura 6.5, no elimina la multimodalidad de la distribución. Este comportamiento es correcto, ya que esta multimodalidad es producto de una ambigüedad de localización, y no de ruido que desaparece al poco tiempo.

- **Experimento 3.**

  El último experimento muestra un ejemplo de aplicación del algoritmo EM para realizar un mapeado. Los datos han sido tomados del simulador con el robot evolucionando en una habitación en forma de L invertida. En la figura 6.7 se muestra la evolución del robot, junto con las lecturas de los sonares.

  El modelo de habitación a estimar se puede observar en la figura 6.6. Se define mediante tres parámetros, d_1, d_2 y d_3, que miden, respectivamente, la anchura del pasillo vertical, la profundidad del pasillo horizontal y la anchura del pasillo horizontal.

  El objetivo del experimento es comprobar si es posible estimar correctamente los parámetros d_1, d_2 y d_3 que determinan la forma de la habitación a partir de los datos obtenidos, utilizando el algoritmo EM.

  En la figura 6.8 se muestra la evolución del algoritmo EM. Recordemos que en cada iteración del algoritmo consiste en dos fases:

  1. Aplicación del filtro *bootstrap* para estimar la localización del robot en toda la secuencia a partir de los datos y del mejor mapa obtenido en la iteración anterior, seguida de un suavizado de la distribución mediante el algoritmo de Kitagawa.
  2. Búsqueda de los parámetros de máxima verosimilitud para la mejor posición obtenida para cada instantánea de la secuencia temporal, después de aplicada la fase anterior.

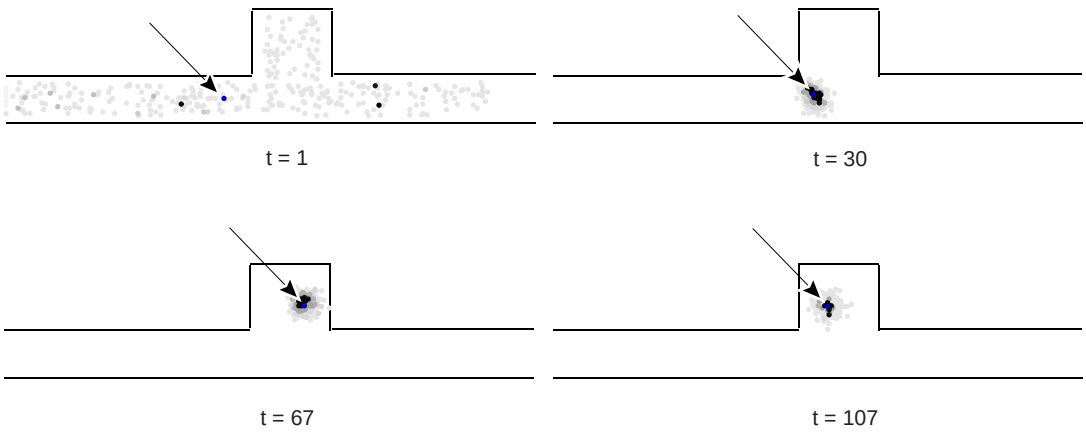
  En esta figura se muestra, en la columna de la izquierda, las mejores posiciones resultantes de la aplicación del filtro *bootstrap*, seguido del algoritmo de suavizado, al mapa resultante de la iteración anterior. La columna de la derecha muestra el mapa de máxima verosimilitud obtenido con el algoritmo de búsqueda adaptativa planteado en este capítulo.

  Los valores iniciales de los parámetros son ($d_1 = 500, d_2 = 0, d_3 = 500$). Se puede comprobar cómo el algoritmo converge rápidamente al mapa correcto ($d_1 = 250, d_2 = 400, d_3 = 400$).

Iteración	Parámetros	Verosimilitud
1	(200,450,400)	-71.743
2	(250,400,400)	-20.193
3	(250,400,400)	-9.408

Tabla 6.4: Evolución del algoritmo EM. Parámetros del mejor mapa obtenido en cada iteración y suma de los logaritmos de las verosimilitudes de las posiciones para ese mapa. Parámetros del mapa correcto: $d_1 = 250$, $d_2 = 400$, $d_3 = 400$.

En la tabla 6.4 se muestra el resultado del valor a maximizar (suma de los logaritmos de las verosimilitudes de las posiciones, dado el mapa) en cada iteración, junto con los parámetros que proporcionan dicho resultado.



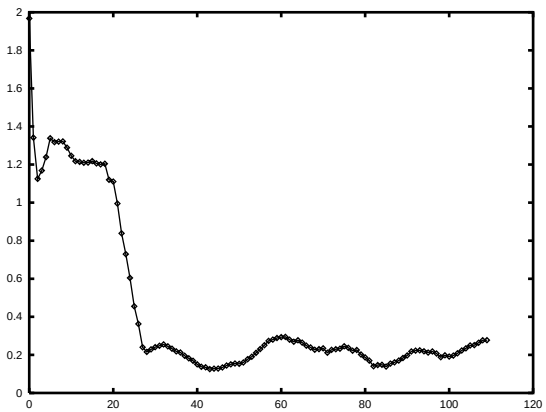


Figura 6.1: **Experimento 1.** Ejemplo de una serie temporal a la que se le aplicará el algoritmo de suavizado. Número de muestras=125. Abajo: representación de la entropía de la distribución de muestras en cada instante de tiempo (suavizada con una ventana de 3 instantes de tiempo) .

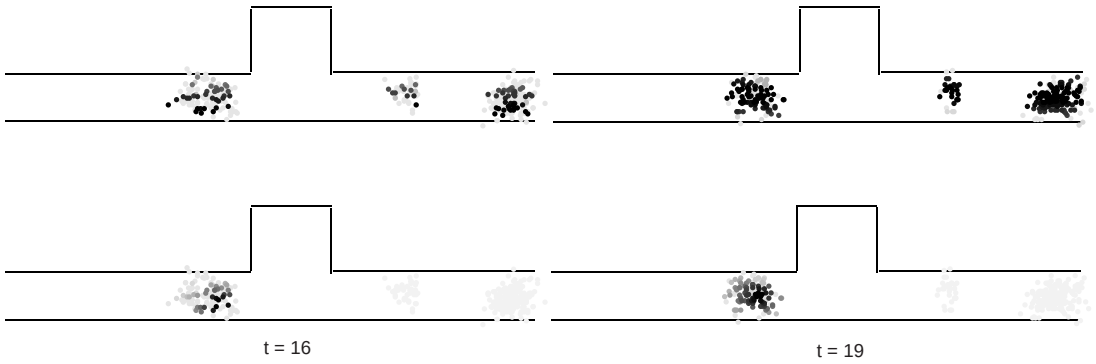


Figura 6.2: **Experimento 1.** Dos instantáneas de la serie temporal antes (arriba) y después (abajo) de aplicar el algoritmo de Kitagawa.

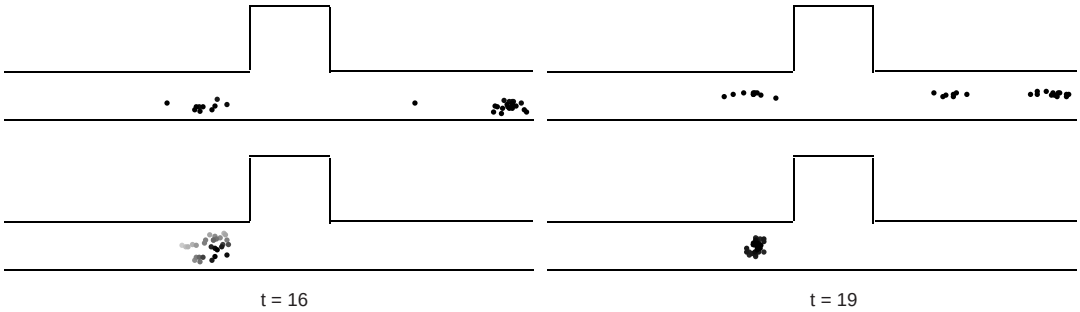


Figura 6.3: **Experimento 1.** Selección de las 30 mejores muestras de cada instantánea de la figura anterior.

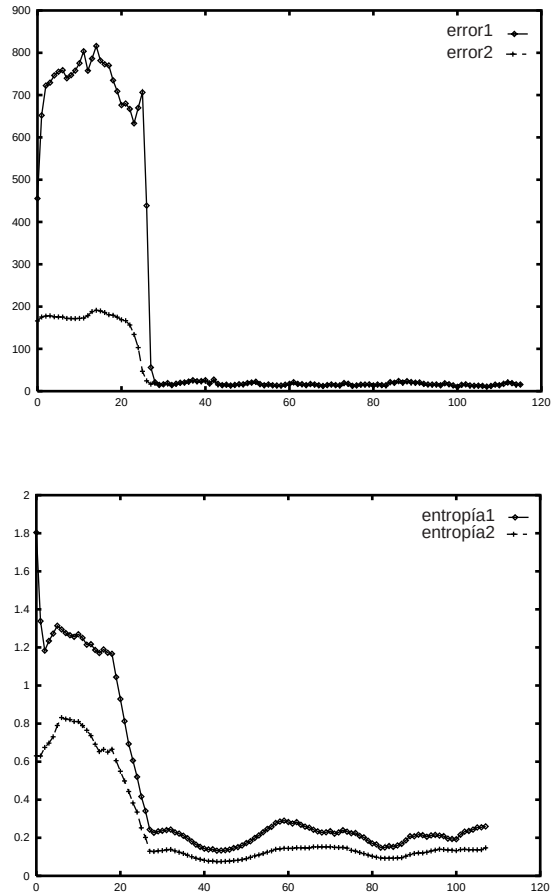


Figura 6.4: **Experimento 1.** Arriba: Error absoluto medio en la posición x del robot de la serie de la figura 6.1 antes (error1) y después (error2) de aplicar el algoritmo de Kitagawa. Abajo: entropía de la misma serie temporal antes (entropía1) y después (entropía2) de aplicar el algoritmo de Kitagawa. Número de muestras=125.

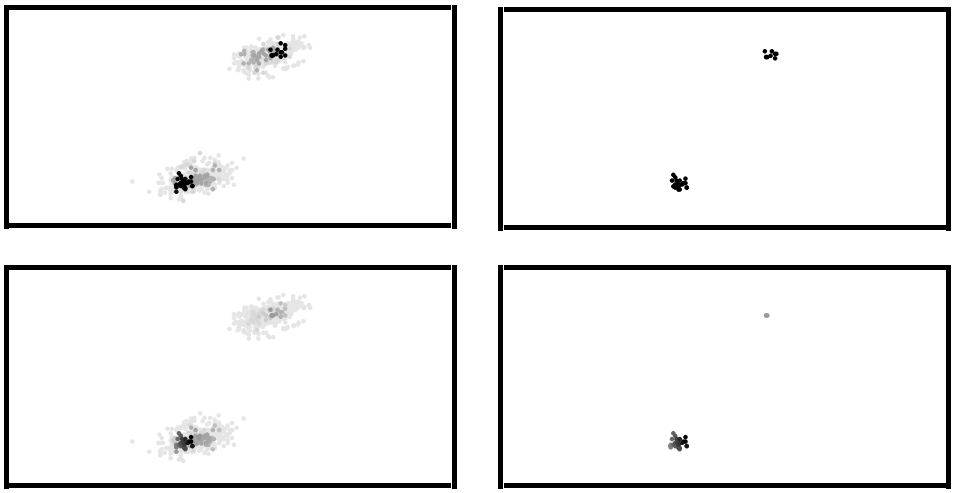


Figura 6.5: **Experimento 2.** Instantánea 29 de la serie temporal mostrada en la figura 5.17 antes (arriba) y después (abajo) de aplicar el algoritmo de Kitagawa. A la derecha de cada una las 30 muestras con mayor verosimilitud. El algoritmo de Kitagawa no elimina la multimodalidad de una distribución ambigua.

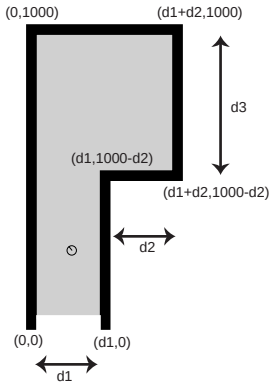


Figura 6.6: **Experimento 3.** Modelo de habitación del experimento 3, definida mediante los parámetros d_1, d_2 y d_3.

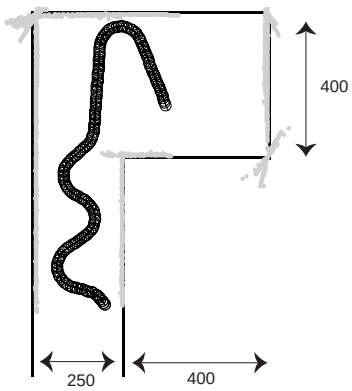


Figura 6.7: **Experimento 3.** Mapa de la habitación y lecturas y movimientos del robot.

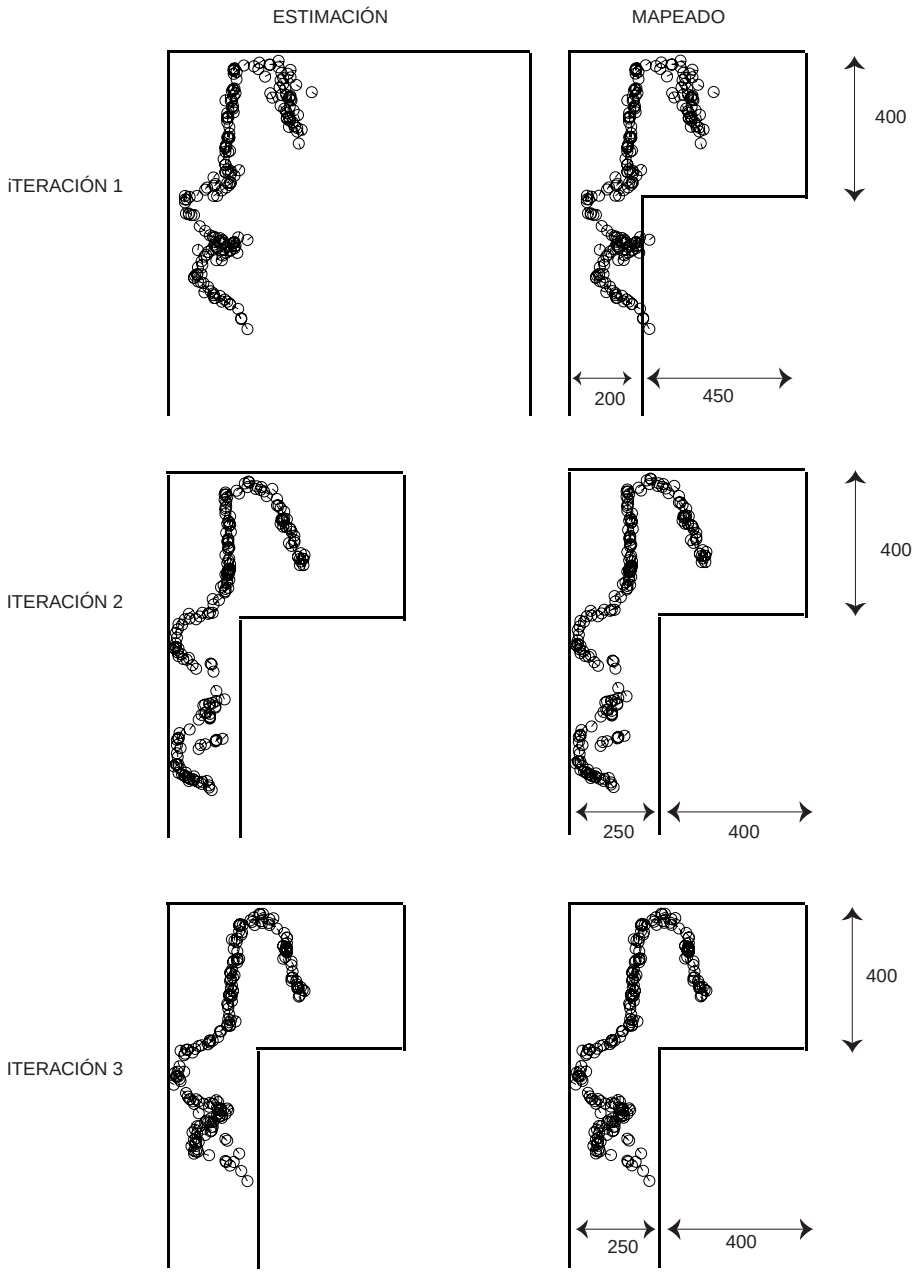


Figura 6.8: **Experimento 3.** Evolución del algoritmo EM para estimar los mejores parámetros d_1, d_2 y d_3 que definen el mapa de la habitación.

6.4 Discusión

Se ha presentado en este capítulo un algoritmo de *estimación-maximización* (EM) que resuelve el problema del mapeado. En este problema, el robot evoluciona por el entorno recogiendo lecturas de los sensores e incrementos de posición obtenidos por odometría. Una vez realizada la exploración, se busca *off-line* el mapa que mejor explica las mediciones obtenidas, aún desconociéndose las posiciones absolutas por las que el robot ha pasado.

Para ello, el algoritmo EM itera una primera fase de estimación y una segunda de mapeado.

La fase de estimación se basa en la utilización del filtro *bootstrap* para obtener la serie completa de posiciones del robot en todos los instantes de tiempo. A continuación se corrigen las probabilidades de las muestras de esta serie temporal utilizando el algoritmo de Kitagawa, y se escogen las muestras de más probabilidad en cada instante de tiempo.

La fase de maximización busca los parámetros del mapa que maximizan la verosimilitud de las lecturas en las posiciones obtenidas en la fase anterior.

Se ha comprobado el buen funcionamiento del algoritmo de Kitagawa para realizar la corrección de las probabilidades de las muestras y el correcto funcionamiento del algoritmo EM para ajustar correctamente los parámetros, obteniendo el mapa del entorno.

Capítulo 7

Conclusiones

En esta tesis se ha realizado un estudio en profundidad de los problemas de localización y mapeado de robots móviles en entornos de oficina, basándonos en el enfoque bayesiano. Este enfoque ha demostrado ser de gran utilidad para formalizar correctamente distintos aspectos de estos problemas, como son los conceptos de *posición más probable*, *mapa que mejor se adapta a unas observaciones* o *trayectoria más probable*. Todos estos conceptos se manejaban habitualmente en la literatura sin tener una contrapartida formal.

El enfoque bayesiano ha demostrado también ser muy fértil en cuanto a las técnicas que se pueden derivar de él. En concreto, en los últimos años se ha utilizado para resolver el problema de la localización mediante técnicas basadas en el filtro de Kalman, en las redes bayesianas o en las rejillas de probabiliad. A estas técnicas hay que añadir la que hemos propuesto en esta tesis: los *filtros de partículas*.

El enfoque de los filtros de partículas, consistente en representar una distribución de probabilidad mediante un conjunto de muestras, es totalmente novedoso en el campo de la robótica móvil y es previsible la aparición de múltiples aplicaciones y algoritmos basados en el mismo. Un ejemplo es la propuesta de algoritmo de *estimación-maximización* que se realiza en la tesis para estimar el mejor mapa del entorno.

Esta tesis presenta, en concreto, las siguientes aportaciones, desde el punto de vista de modelos.

1. *Formulación bayesiana que integra los problemas de localización y mapeado*: se presenta una formulación bayesiana de los problemas de localización y mapeado que generaliza muchas propuestas y que presenta un marco desde el que derivar interesantes modelos, técnicas e implementaciones.

2. *Modelo del sensor de ultrasonidos:* se propone un modelo estocástico que define una función de verosimilitud multimodal de las lecturas de sonar. Este modelo es muy

realista, funciona correctamente y contempla tipos de lecturas de los sonares que hasta el momento no se habían modelado (*dobles rebotes*).

3. *Modelo de observación:* presentamos un modelo de observación para la localización del robot que contempla tanto la posibilidad de ruido aleatorio como la de obstáculos no modelados. Esta formulación lo hace especialmente robusto y eficiente, en contraposición con modelos más sencillos utilizados en la literatura.

4. *Mapas paramétricos:* los modelos de mapas de entorno que proponemos son modelos métricos parametrizables mediante las posiciones de vertices de polígonos. Este tipo de parametrización permite definir de una forma sencilla y con pocos parámetros modelos con restricciones geométricas elaboradas. Esto hará que los algoritmos de mapeado sean más rápidos y exactos.

En cuanto a técnicas y algoritmos, en la tesis hemos planteado de forma novedosa la aplicación de una serie de algoritmos a problemas de localización y mapeado:

1. *Filtro bootstrap:* se aporta la aplicación del filtro *bootstrap* al problema de la detección de características topológicas, de la localización global y del seguimiento de posiciones. El algoritmo ha localizado correctamente las características topológicas y la posición del robot, utilizando tanto datos simulados como datos reales.

2. *Algoritmo de Kitagawa:* proponemos la aplicación del algoritmo de Kitagawa a la corrección de la localización de una serie temporal completa. El algoritmo funciona correctamente con datos del simulador y datos reales.

3. *Algoritmo EM:* presentamos la formulación de un algoritmo EM basada en partículas y en técnicas de búsqueda adaptativa, así como la aplicación del algoritmo EM al problema del mapeado. Se ha comprobado la corrección del algoritmo en datos obtenidos con el simulador.

Como líneas futuras de desarrollo de esta investigación proponemos las siguientes:

1. *Problemas prácticos:* se deben solucionar problemas prácticos, como es el tiempo de computo de los algoritmos de simulación y de *bootstrap*. Se debe mejorar su eficiencia para que sean aplicables en tiempo real.

2. *Colapso del filtro bootstrap:* uno de los problemas fundamentales del filtro *bootstrap* es su colapso en situaciones de multimodalidad. Es necesario la aportación de soluciones teóricas y prácticas de este problema.

3. *Distinguibilidad de las posiciones:* profundización en los problemas teóricos de distinguibilidad de las posiciones, dado un modelo de entorno y un modelos de observación. Existen posiciones de los entornos que tienen una alta distinguibilidad. Sería interesante utilizar estas posiciones como *landmarks* naturales en los que el robot se va a relocalizar correctamente. Esta línea lleva a una teoría de la localización activa que pasa por un uso obligado de la teoría de la información.

4. *Extensión a localización por visión:* la extensión de la propuesta a visión pasa por formular la verosimilitud utilizando como datos percibidos por el robot las imágenes obtenidas por el robot. Se debería comparar esta imagen con la que el robot vería en las posiciones candidatas. Para ello es necesario formular un modelo del entorno que permita generar vistas desde posiciones arbitrarias.

Apéndice A

Muestreo por rechazo

Supongamos una distribución de densidad de probabilidad

$$p(x) = cg(x)h(x) \tag{A.1}$$

donde $g(x)$ puede ser calculada para cualquier valor x y es posible generar muestras de la distribución $h(x)$.

Para generar una población de N muestras $(m_1, \ldots, m_N)$ de la función $p(x)$ se puede utilizar el algoritmo de aceptación y rechazo (ver tabla A).

Algoritmo MUESTREO POR RECHAZO
Salida: Valores $(m_1, m_2, \ldots, m_N)$ muestreados de la densidad de probabilidad $p(x) = cg(x)h(x)$
1. $i \leftarrow 1$
2. $u \leftarrow$ valor aleatorio de la distribución $U(0, 1)$
3. $y \leftarrow$ valor aleatorio de la distribución $h(x)$
4. Si $u \leq g(y)$ hacer $m_i = y$. En otro caso ir a 2.
5. $i \leftarrow i + 1$ y saltar a 2 hasta que $i = N$

Tabla A.1: Algoritmo de rechazo y su versión modificada para mejorar su eficiencia.

En la primera versión del algoritmo se generan muestras y de la distribución $h(x)$. El valor devuelto por $g(y)$, para cada muestra de $h(x)$ determina la probabilidad de aceptar dicha muestra. Para hacer efectiva esa probabilidad se utiliza el número u generado por la distribución uniforme, de manera que la muestra se acepta cuando este número es menor que el valor $g(y)$.

Algoritmo MUESTREO POR RECHAZO MODIFICADO
Salida: Valores $(m_1, m_2, \ldots, m_N)$ y probabilidades asocidas $(\pi_1, \pi_2, \ldots, \pi_N)$ muestreados de la densidad de probabilidad $p(x) = cg(x)h(x)$

1. $i \leftarrow 1$
2. $m_i \leftarrow$ valor aleatorio de la distribución $h(x)$
3. $\pi_i \leftarrow g(m_i)$
4. $i \leftarrow i + 1$ y saltar a 2 hasta que $i = N$

Tabla A.2: Algoritmo de rechazo y su versión modificada para mejorar su eficiencia.

Por ejemplo, supongamos que se genera el número $y = 1.5$ y $g(1.5) = 0.75$. Esto significa que la probabilidad de aceptar $y = 1.5$ como un valor aleatorio generado por $p(x)$ es 0.75. Se utiliza el valor aleatorio u para realizar esta aceptación.

Un problema muy importante del algoritmo de rechazo es que se necesitan demasiadas muestras cuando la función $g(x)$ nos da valores de probabilidad muy pequeños. La mejora planteada por la versión modificada del algoritmo (ver tabla A) consiste en ir almacenando las probabilidades obtenidas por $g(x)$ junto con las muestras. De esta forma, como salida del algoritmo se obtienen N muestras $(m_1, \ldots, m_N)$ y sus probabilidades asociadas $(\pi_1, \ldots, \pi_N)$.

Apéndice B

Algoritmo EM

El algoritmo EM, inicialmente propuesto por Dempster (Dempster, Laird, y Rubin 1977), presenta un técnica iterativa general para realizar una estimación de máxima verosimilitud de parámetros de problemas en los que existen ciertos *datos ocultos*. Presentamos a continuación el algoritmo, y un ejemplo de su aplicación a la resolución de un problema concreto, tomando (Mitchel 1997) como referencia.

B.1 Descripción del algoritmo EM

El algoritmo EM puede aplicarse en muchas situaciones en las que se desea estimar un conjunto de parámetros θ que describen una distribución de probabilidad subyacente, dada únicamente una parte observada de los datos completos producidos por la distribución. En general, supongamos que en cada realización del experimento aleatorio se observa un parámetro z_i y existe un parámetro oculto x_i. Denotamos entonces por $Z = \{z_1, \ldots, z_m\}$ al conjunto de datos observados en m realizaciones del experimento, por $X = \{x_1, \ldots, x_m\}$ al conjunto de datos no observados y por $Y = Z \cup X$ al conjunto completo de datos. Los datos X pueden considerarse una variable aleatoria cuya distribución de probabilidad depende de los parámetros a estimar θ y de los datos observados Z. De la misma forma, Y es una variable aleatoria ya que está definida en términos de la variable aleatoria X. Llamemos h a la hipótesis actual de los valores de los parámetros θ, y denotemos por h' la hipótesis revisada que se estima en cada iteración del algoritmo EM.

El algoritmo EM busca la hipótesis h' que maximiza la esperanza $E[\ln p(Y \mid h')]$, siendo $p(Y \mid \theta)$ la distribución de probabilidad que define Y y que depende de los parámetros desconocidos θ. Esta distribución de probabilidad define la verosimilitud de los datos completos Y dada una hipótesis h' de los parámetros ocultos. Al maximizar el logaritmo de la distribución se está maximizando la verosimilitud. Se introduce el valor esperado $E[\ln p(Y \mid h')]$

debido a que el conjunto completo de datos Y es una variable aleatoria. Dado que el conjunto completo de datos Y contiene datos X no observados, se deben considerar todos los posibles valores de X, ponderándolos según su probabilidad. En otras palabras, se calcula el valor esperado $E[\ln p(Y \mid h')]$ sobre la distribución de probabilidad que gobierna la variable aleatoria Y. Esta distribución está determinada por los valores observados Z más por la distribución de los valores no observados X.

En general, se desconoce la distribución de Y, porque está determinada por los parámetros θ que se intenta estimar. Por ello, el algoritmo EM usa la hipótesis actual h para estimar la distribución de Y. Se define entonces una función $Q(h \mid h')$ que proporciona $E[\ln p(Y \mid h')]$ como una función de h', bajo la suposición de que $\theta = h$ y dada el conjunto de observaciones Z del conjunto completo de datos Y

$$Q(h' \mid h) = E[\ln p(Y \mid h') \mid h, Z].$$

En la función $Q(h' \mid h)$ se supone que la hipótesis h y los datos observados Z tienen unos valores fijos y que éstos definen la distribución de probabilidad de las variables ocultas X (y, por tanto, sus valores esperados). La distribución de probabilidad de Y definida por Z y h es, entonces, la que se utiliza para calcular $E[\ln p(Y \mid h')]$ para una hipótesis cualquiera h'. En su forma general, el algoritmo EM repite la siguiente pareja de pasos hasta que converge.

Paso 1: *Paso de estimación (E)*: Calcular $Q(h' \mid h)$ utilizando la hipótesis actual h y los datos observados Z para estimar la distribución de probabilidad de Y

$$Q(h' \mid h) \leftarrow E[\ln p(Y \mid h') \mid h, Z]. \tag{B.1}$$

Paso 2: *Paso de maximización (M)*: Sustituir h por la hipótesis h' que maximiza la función Q

$$h \leftarrow \arg\max_{h'} Q(h' \mid h). \tag{B.2}$$

B.2 Aplicación a la estimación de k medias

Para ilustrar el funcionamiento del algoritmo EM, vamos a utilizarlo para derivar un algoritmo que estime las medias de una mezcla de k distribuciones normales $\theta = (\mu_1, \ldots, \mu_k)$ con igual desviación típica σ, que se supone conocida. Los datos observados $Z = \{z_j\}$ son los datos producidos por la distribución. Los datos no observados

$$X = \{(x_{1j}, \ldots, x_{kj})\}, \qquad x_{ij} = (0, 1), \qquad \sum_{i=1}^{k} x_{ij} = 1$$

indican cuál de las k distribuciones normales se ha utilizado para obtener el dato z_j.

Para aplicar EM primero se necesita derivar una expresión de $Q(h \mid h')$ para el problema. Derivemos primero la formulación de $\ln p(Y \mid h')$. Para un único conjunto de datos $y_j = (z_j, x_{1j}, \ldots, x_{kj})$, la verosimilitud de que estos datos hayan sido obtenidos con una hipótesis $h' = (\mu'_1, \ldots, \mu'_k)$ se puede escribir como

$$p(y_j \mid h') = p(z_j, x_{1j}, \ldots, x_{kj} \mid h') = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{k} x_{ji}(z_j - \mu'_i)^2\right). \quad \text{(B.3)}$$

Esta expresión proporciona la probabilidad de que el valor z_j haya sido generado por la distribución normal seleccionada por los datos ocultos. La probabilidad para todos las instancias m de los datos es

$$\begin{aligned} \ln p(Y \mid h') &= \ln \prod_{i=j}^{m} p(y_j \mid h') = \\ &= \sum_{j=1}^{m} \ln p(y_j \mid h') = \\ &= \sum_{j=1}^{m} \left(\ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2} \sum_{i=1}^{k} x_{ij}(z_j - \mu'_i)^2\right) \end{aligned} \quad \text{(B.4)}$$

Por último, se debe calcular el valor esperado de esta expresión $\ln p(Y \mid h')$ sobre toda la distribución de probabilidad que gobierna Y o, de forma equivalente, sobre la distribución de los datos ocultos de Y, x_{ij}. Al ser la expresión anterior una expresión linear en función de estos datos, es posible derivar la siguiente expresión

$$\begin{aligned} E[\ln p(Y \mid h')] &= E\left[\sum_{j=1}^{m} \left(\ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2} \sum_{i=1}^{k} x_{ij}(z_j - \mu'_i)^2\right)\right] \\ &= \sum_{j=1}^{m} \left(\ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2} \sum_{i=1}^{k} E[x_{ij}](z_j - \mu'_i)^2\right) \end{aligned} \quad \text{(B.5)}$$

Para resumir, la función $Q(h \mid h')$ del problema de las k medias es

$$Q(h' \mid h) = \sum_{j=1}^{m} \left(\ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2} \sum_{i=1}^{k} E[x_{ij}](z_j - \mu'_i)^2\right), \quad \text{(B.6)}$$

donde $h' = (\mu'_1, \ldots, \mu'_k)$ y donde los valores esperados de los datos ocultos $E[x_{ij}]$ se calculan a partir de la hipótesis actual y a los datos observados Z. Este valor $E[x_{ij}]$ es simplemente la probabilidad de que la muestra z_j haya sido generada por la distribución normal i

$$\begin{aligned} E[x_{ij}] &= \frac{p(x = z_j \mid \mu = \mu_i)}{\sum_{n=1}^{k} p(x = z_j \mid \mu = \mu_n)} = \\ &= \frac{\exp\left(-\frac{1}{2\sigma^2}(z_j - \mu_i)^2\right)}{\sum_{n=1}^{k} \exp\left(-\frac{1}{2\sigma^2}(z_j - \mu_n)^2\right)} \end{aligned} \tag{B.7}$$

Esta ecuación completa el primer paso del algoritmo EM, en el que se define la función Q a partir de los datos ocultos esperados. El segundo paso (maximización) consiste en encontrar los valores $(\mu'_1, \ldots, \mu'_k)$ que maximizan la función Q así definida. En este caso,

$$\begin{aligned} &\arg\max_{h'} Q(h' \mid h) = \\ &\quad = \arg\max_{h'} \sum_{j=1}^{m} (\ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2} \sum_{i=1}^{k} E[x_{ij}](z_j - \mu'_i)^2) = \\ &\quad = \arg\min_{h'} \sum_{i=1}^{k} \sum_{j=1}^{m} E[x_{ij}](z_j - \mu'_i)^2 \end{aligned} \tag{B.8}$$

Esto es, la hipótesis de máxima verosimilitud es la que minimiza la suma ponderada de los errores al cuadrado, donde la contribución de cada instancia z_j al error que define μ'_i está ponderada por $E[x_{ij}]$. Esta hipótesis se puede calcular de forma analítica con la siguiente expresión

$$\mu_j \leftarrow \frac{1}{m} \sum_{j=1}^{m} E[x_{ij}] z_j. \tag{B.9}$$

Apéndice C

Generación de trayectorias robustas mediante algoritmos genéticos

Este apéndice y el siguiente presentan trabajos previos al desarrollo del cuerpo central de la tesis, relacionados con técnicas de navegación local. Estos trabajos dieron lugar a publicaciones (Gallardo, Colomina, Flórez, Arques, Company, y Rizo 1997; Gallardo, Colomina, Flórez, y Rizo 1998) y sirvieron para plantear algunas preguntas de las que ha surgido la tesis.

C.1 Introducción

El problema de la generación de un camino a seguir por un robot móvil para llegar de una posición inicial a otra final evitando los obstáculos del entorno ha sido tratado ampliamente en la literatura (Latombe 1991), encontrándose soluciones eficientes tanto mediante técnicas algorítmicas como mediante técnicas evolutivas (Xiao, Michalewicz, Zhang, y Trojanowski 1997; Doyle 1995; Rendas y W.Tetenoire 1997; Ahuactzin, Talbi, Bessiere, y Mazer 1992). La mayor parte de estos enfoques resuelven el problema en el espacio de configuraciones del robot (definido normalmente por su posición x, y y su orientación θ). En ese caso la solución es una secuencia continua de configuraciones espaciales, esto es, una trayectoria en el plano que no colisiona con los obstáculos y que conecta el punto inicial con el punto final. Sin embargo, una trayectoria espacial no resuelve directamente el problema de mover el robot, ya que le falta una ley temporal asociada a la misma (distintas velocidades lineales y angulares pueden generar la misma trayectoria espacial).

Para resolver completamente el problema hay que planificar una secuencia de velocidades, realizando entonces la búsqueda no en el espacio de configuraciones sino en el espacio de velocidades (en el caso de robots *synchro-drive* como el que se utiliza en este trabajo,

estas velocidades son la velocidad lineal v y la velocidad angular ω). En este nuevo espacio, los obstáculos ya no son definibles geométricamente, por lo que se limita mucho el tipo de técnicas algorítmicas que pueden aplicarse. Adicionalmente, en muchos casos es necesario imponer restricciones que reducen localmente la dimensionalidad del espacio de velocidades del robot. Por ejemplo, un robot que se mueve con las características de un automóvil, como es el presente caso, no puede moverse lateralmente y tiene limitado el ángulo de giro. A este problema se le denomina planificación de trayectorias con restricciones cinemáticas no holonómicas y su solución es materia de investigación actual en el campo de la robótica y el control (Latombe 1991; Divelbiss y Wen 1994; Chatila, Khatib, Jaouni, y Laumond 1997). Todas estas soluciones plantean dos importantes problemas:

1. Soluciones subóptimas: la mayor parte de las técnicas algorítmicas que resuelven el problema de la búsqueda de trayectorias no holonómicas encuentran *una* trayectoria, pero no contemplan factores prácticos de eficiencia como el tiempo de la misma, su regularidad, etc.

2. Robustez de la trayectoria: una vez calculada una trayectoria, ésta deberá ser ejecutada por un robot real, lo que nos lleva directamente al problema de la incertidumbre en su ejecución. Esta incertidumbre se debe a la inexactitud en la ejecución de las órdenes de velocidad y a la inexactitud en la localización del robot por la incertidumbre asociada a la odometría. Esta incertidumbre es acumulativa con lo que cuanto más larga sea la trayectoria, más se sufrirá.

   Debido a ello, las trayectorias que se obtengan como solución deben ser robustas, en el sentido de que la introducción en ellas de pequeños cambios no debe variar mucho el resultado final. Esta característica de robustez no es garantizada por ninguna de las aproximaciones que se utilizan para resolver el problema.

En este trabajo se presenta una solución a la planificación de trayectorias no holonómicas mediante computación evolutiva que contempla los dos problemas arriba planteados. La búsqueda de la mejor trayectoria para unos criterios dados se realizará de forma natural introduciendo esos criterios en la función de bondad, y la robustez de la trayectoria se garantizará introduciendo en esta función de bondad un término de ruido gausiano acumulativo y adaptando el método de búsqueda genética para que esto sea contemplado.

C.2 Planteamiento del problema

Esta sección describe las ecuaciones de movimiento fundamentales de un robot *synchro-drive*, definiendo la componente dinámica del control del mismo. Las trayectorias a seguir por el robot y las funciones de evaluación asociadas a los mismos se basan en estas ecuaciones.

Las variables de estado del robot móvil son su posición (x, y), su orientación (θ) y sus velocidades angular y lineal (ω y v, respectivamente). Consideraremos constantes las aceleraciones angulares y lineales (a_ω y a_v). Estas aceleraciones tienden a no ser demasiado altas en implementaciones reales de robots móviles, para no someter a la estructura mecánica del robot a tensiones excesivas y para conseguir movimientos suaves. Debido a ello estas aceleraciones no deben ser despreciadas si se quieren conseguir resultados aplicables a la realidad.

El control del robot lo modelamos con las variables c_v y c_ω que toman valores discretos $\{-1, 0, 1\}$ y definen si, para un instante de tiempo determinado, las velocidades deben ser decrementadas, no modificadas o incrementadas. El incremento de estas velocidades vendrá dado por la constante de aceleración.

$$\omega(t_n) = \omega(t_0) + \int_{t_0}^{t_n} a_\omega c_\omega(t)dt \tag{C.1}$$

$$v(t_n) = v(t_0) + \int_{t_0}^{t_n} a_v c_v(t)dt \tag{C.2}$$

La dinámica de las variables de posición y dirección angular del robot se modela con las siguientes ecuaciones.

$$\theta(t_n) = \theta(t_0) + \int_{t_0}^{t_n} \omega(t)dt \tag{C.3}$$

$$x(t_n) = x(t_0) + \int_{t_0}^{t_n} v(t)\cos\theta(t)dt \tag{C.4}$$

$$y(t_n) = y(t_0) + \int_{t_0}^{t_n} v(t)\sin\theta(t)dt \tag{C.5}$$

Las ecuaciones anteriores pueden simplificarse si se asume que el robot debe controlarse de forma discreta, con intervalos de control de tiempo $\triangle t$. Suponemos, pues, constantes las variables de control c_v y c_ω para estos intervalos de tiempo. De esta forma se simplifican las anteriores ecuaciones.

$$x(t_n) = x(t_0) + \sum_{i=0}^{n-1} \int_{t_i}^{t_i+\triangle t} (v(t_i) + \triangle_{t_i}^t v)\cos(\theta(t_i) + \triangle_{t_i}^t \theta)dt \tag{C.6}$$

donde

$$\triangle_{t_i}^{t} v = a_v c_v(t)(t - t_i) \quad \text{(C.7)}$$

$$\triangle_{t_i}^{t} \omega = \omega(t_i) c_\omega(t_i)(t - t_i) + \frac{1}{2} a_\omega(t_i) c_\omega(t_i)(t - t_i)^2 \quad \text{(C.8)}$$

Por ello el problema de encontrar una trayectoria en un espacio de velocidades puede formularse como la búsqueda de una secuencia temporal de valores para las órdenes c_v y c_ω

$$\langle (c_{v_{t_0}}, c_{\omega_{t_0}}), (c_{v_{t_1}}, c_{\omega_{t_1}}), \ldots, (c_{v_{t_n}}, c_{\omega t_n}) \rangle$$

que haga moverse el robot desde la configuración inicial ($x = x_0$, $y = y_0$, $\theta = \theta_0$, $v = 0$, $\omega = 0$, $t = t_0$) hasta la posición objetivo ($x = x_{\text{obj}}$, $y = y_{\text{obj}}$, $v = 0$, $\omega = 0$, $t = t_n$). Hay que hacer notar que el robot debe terminar en una configuración estática y que no imponemos ninguna restricción a la orientación final.

Además, como restricciones adicionales, buscamos que t_n sea el mínimo posible y que la trayectoria resultante sea robusta, en el sentido comentado anteriormente.

C.3 Generación de trayectorias mediante algoritmos genéticos

En el problema de la generación de trayectorias óptimas, la elección del uso de algoritmos genéticos (Back, Fogel, y Michalewicz 1997) se fundamenta en dos razones principales: en primer lugar, es una técnica adecuada para realizar búsquedas en espacios de dimensión elevada, como en este caso. Por otro lado, el método impone pocas restricciones de tipo matemático en la forma de la función a optimizar, de tal manera que es aplicable a la generación de trayectorias para cualquier tipo de comportamiento (evitar obstáculos, seguir paredes, etc.).

C.3.1 Representación de las soluciones

Cada individuo de la población representará una trayectoria que parte del punto origen e intenta llegar al destino. Como el objetivo no es únicamente obtener una secuencia de coordenadas espaciales que conecte el punto origen con el punto de destino, sino además encontrar cuál es la velocidad lineal y angular que el robot debe tomar en cada punto del camino, un cromosoma estará formado por una serie de velocidades lineales y angulares de referencia para el robot en instantes de tiempo sucesivos. Así, una trayectoria en el espacio de velocidades se codificará como un vector de pares de valores para v y w:

$$\langle (v_{t_1}, w_{t_1}), (v_{t_2}, w_{t_2}), \ldots, (v_{t_n}, w_{t_n}) \rangle$$

Esta secuencia de velocidades de referencia se transformará para su evaluación en una secuencia de órdenes de cambio de velocidad como los descritos en la sección C.2.

C.3.2 Función de evaluación

La función de evaluación mide la optimalidad de cada trayectoria en términos de la distancia de la posición final del robot a la posición objetivo y el tiempo invertido en llegar hasta ella:

$$f = \alpha d_{\text{obj}}(t_n, \sigma) + \beta t_n$$

donde:

- $d_{\text{obj}}(t, \sigma) = \sqrt{(\chi(t, \sigma) - x_{\text{obj}})^2 + (\psi(t, \sigma) - y_{\text{obj}})^2 + \nu(t, \sigma)^2 + \varphi(t, \sigma)^2}$ es la distancia euclídea en el espacio (x, y, v, w) entre el objetivo y el punto final de la trayectoria.
- t_n es el tiempo invertido en recorrer la misma.
- α, β son coeficientes de ponderación que miden la importancia relativa de cada uno de los factores.

Para asegurar que los individuos que chocan siempre tienen una adecuación peor (mayor) que los que llegan al final sin chocar, se le suma al valor de su función de adecuación el valor de la adecuación del peor de los individuos que no ha chocado en esa generación.

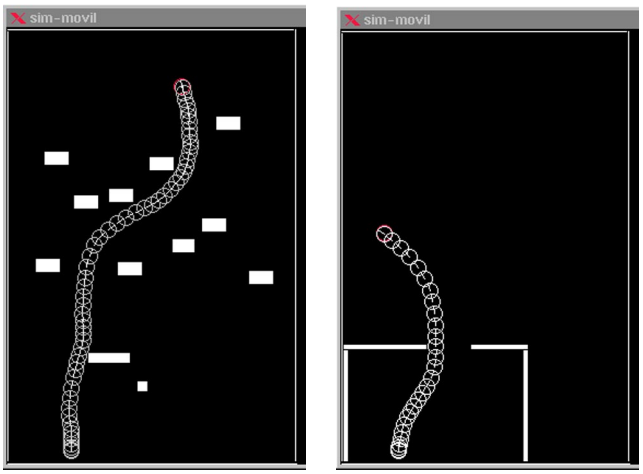
Las funciones $\nu(t, \sigma)$ y $\varphi(t, \sigma)$ representan las velocidades lineal y angular obtenidas introduciendo un ruido gausiano de desviación típica σ en las ecuaciones (C.1) y (C.2), quedando las mismas como:

$$\varphi(t_n, \sigma) = \omega(t_0) + \int_{t_0}^{t_n} (a_\omega c_\omega(t) + \rho(t, \sigma))dt \tag{C.9}$$

$$\nu(t_n, \sigma) = v(t_0) + \int_{t_0}^{t_n} (a_v c_v(t) + \rho(t, \sigma))dt \tag{C.10}$$

Las funciones $\chi(t, \sigma)$ y $\psi(t, \sigma)$ son los valores de x e y que se obtienen con estas velocidades. El término de ruido $\rho(t, \sigma)$ modela la incertidumbre en las velocidades y en las posiciones que se produciría al ejecutar la secuencia de órdenes codificados en la trayectoria en un robot real.

Desde el punto de vista de la función de evaluación esta incertidumbre se traduce en que un cromosoma no tiene asociado un único valor de adecuación. Por ello, la estrategia adoptada consiste en evaluar N veces cada individuo y tomar como su valor de adecuación el máximo (el peor) de los valores obtenidos, para asegurar que se promueven las trayectorias robustas. Para impedir que la introducción de ruido mejore accidentalmente trayectorias que de otro modo obtendrían peor valoración, cada individuo se evalúa además suponiendo la inexistencia de ruido.



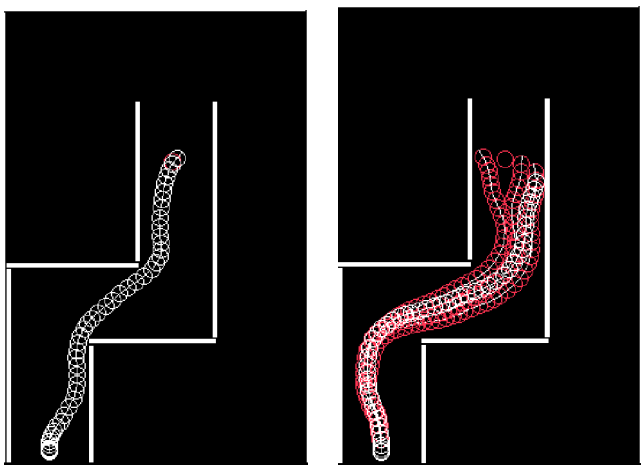
(a) (b)

Figura C.1: Resultados sin ruido para distintos entornos.

C.3.3 Operadores genéticos

- **Cruce:** este operador combina las velocidades de referencia de dos trayectorias. Para ello se calcula de manera aleatoria un punto de cruce independiente para cada uno de los cromosomas y se intercambia el material genético que queda a la derecha de los respectivos puntos de cruce. Al ser el punto de cruce distinto para cada uno de los "padres" este método da lugar a individuos de longitud variable. Esto permite que la longitud de la trayectoria vaya creciendo en las sucesivas generaciones y el robot se vaya aproximando al punto de destino.

- **Mutación:** cada mutación consiste en sumar un valor aleatorio procedente de una distribución normal de media 0 y varianza 5 metros por segundo (o 5 grados por segundo en el caso de velocidades angulares) a una de las velocidades de referencia que componen el cromosoma.



(a) (b)

Figura C.2: Resultados para el entorno *pasillo*.

C.4 Resultados

C.4.1 Resultados en distintos entornos sin ruido

En la figura C.1 pueden verse dos trayectorias generadas por el algoritmo genético en distintos entornos, ambas sin aplicar ningún término de ruido a la función de evaluación. Para obtenerlas se han utilizado los siguientes parámetros: número de individuos $m = 100$, número de generaciones $n = 100$, probabilidad de cruce $p_c = 0.75$ y probabilidad de mutacion $p_m = 0.01$.

Hay que hacer notar que las trayectorias resultantes alcanzan velocidades considerables que pasan demasiado cerca de los obstáculos. Una ejecución de alguna de ellas en un robot real terminaría, probablemente, en una situación de colisión con alguno de ellos.

C.4.2 Comparación de resultados con y sin ruido

Con el objeto de comprobar los efectos que produce la introducción de ruido en las características de las trayectorias obtenidas mediante el algoritmo evolutivo, se han realizado distintas pruebas en un mismo entorno (figura C.2), manteniendo constantes los parámetros del algoritmo (los mismos que en el apartado anterior) y variando únicamente la desviación estándar σ del ruido gausiano generado. El promedio de los resultados obtenidos tras 20

ejecuciones del algoritmo con cada nivel de ruido se muestra en la tabla C.1. El parámetro D_{min} representa la distancia mínima del robot a los objetos del entorno a lo largo de toda la trayectoria, mientras que V_{med} es la velocidad lineal media del robot.

Como puede observarse, la presencia de ruido fuerza a que las trayectorias se alejen a una mayor distancia de los objetos. Las trayectorias sin ruido consiguen una mejor velocidad promedio al no verse sometidas a restricciones de distancia (figura C.2), aunque este aspecto se ve ampliamente compensado al utilizar ruido, ya que éstas son más seguras de cara a su ejecución en un robot real.

Parámetro	Nivel de ruido (σ)		
	0.0	0.04	0.1
$D_{min}(cm)$	38.2	39.8	43.8
$V_{med}(cm/s)$	51.31	50.92	46.4

Tabla C.1: Resultados obtenidos para el entorno *pasillo*

C.5 Conclusiones

Se ha presentado una solución al problema de generación de trayectorias robustas en el espacio de velocidades de un robot móvil mediante algoritmos genéticos. Los resultados obtenidos son correctos y se mejora la robustez de las trayectorias resultantes con respecto a las de otros métodos. Para ello se ha utilizado en el algoritmo genético una función de evaluación que incorpora un término de ruido acumulativo y se ha adaptado este algoritmo para que sea capaz de encontrar una solución subóptima y robusta que satisfaga esta función de evaluación.

Como trabajos futuros nos planteamos modificar la función de evaluación para generar diferentes trayectorias que correspondan a esquemas de comportamiento local, como es el seguimiento de paredes, o la navegación por el centro del pasillo, con objeto de utilizar estas trayectorias obtenidas como muestras de aprendizaje para la generación de un controlador local que responda a estos esquemas. También estamos estudiando otras alternativas de codificación de las soluciones que permita obtener mejores resultados en entornos con mínimos locales pronunciados.

Apéndice D

Aprendizaje de conductas locales de navegación

D.1 Introducción

Existen dos tipos básicos de enfoques para controlar la navegación de un robot móvil: técnicas globales y locales. En las técnicas globales, como son los métodos geométricos, la programación dinámica o los métodos de campo de potencial (ver en (Latombe 1991) resumen y referencias complementarias) se asume totalmente conocida la descripción geométrica del entorno en el que se va a mover el robot. Se trata de métodos potentes y eficaces para generar tanto trayectorias a seguir como secuencias de comandos a ejecutar. Son métodos usados para el control de robots que trabajan en entornos sin ninguna variabilidad y que realizan tareas repetitivas en las que se conocen en todo momento el valor de todas sus variables de estado.

Los métodos locales o reactivos, por el contrario, consideran que el robot va a moverse en un entorno no conocido a priori y proporcionan unas conductas estándar para reaccionar ante lecturas de los sensores del robot (evitar obstáculo, seguir pared, entrar en puerta, alinearse con objeto, etc.). Estas conductas (o *esquemas* en la terminología de Arkin (Arkin 1990)) son aplicables en gran número de entornos distintos y se suelen usar junto con un control de alto nivel que se encarga de secuenciarlas. Este tipo de control es el que vamos a utilizar en el presente trabajo.

D.1.1 Técnicas previas para el control local

Entre las propuestas de control local, cabe destacar el enfoque del *histograma de campo vectorial* (Borenstein y Korem 1991), el método de velocidad-curvatura (Simmons 1996), el

Método de control local	**Esquemas**

Histograma de campo de potencial	`avanzar-evitando-obstáculos`
	`ir-a-objetivo`
	`entrar-por-puerta`

Velocidad-Curvatura	`avanzar-evitando-obstáculos` `ir-a-objetivo`
Ventana Dinámica	`avanzar-evitando-obstáculos` `ir-a-objetivo`
Esquemas motores	esquemas configurables

Tabla D.1: Esquemas de actuación susceptibles de ser implementados con cada uno de los métodos de control local. Ver en el texto principal las referencias correspondientes a cada uno de los métodos.

método de ventana dinámica (D.Fox, Burgard, y Thrun 1997) y el método del propio Arkin de esquemas motores (Arkin 1989).

En la tabla D.1 se comparan el tipo de esquemas de conducta susceptibles de ser implementados usando cada uno de estos enfoques. El enfoque de esquemas motores, pese a ser el más genérico, es muy complejo de llevar a la práctica por el gran número de parámetros que deben ser ajustados en las ecuaciones de control del robot. El mismo problema plantea el enfoque del histograma de campo de potencial. Un problema añadido de ambos métodos es que obtienen los comandos del robot en dos fases separadas. En la primera fase se obtiene la dirección objetivo en la que debe moverse el robot. En la segunda fase se generan los comandos de modificación de las velocidades lineales y angulares necesarios para conducir al robot en la dirección deseada. Este enfoque sólo es factible si consideramos que las aceleraciones aplicables al robot son infinitas y el robot puede realizar de forma instantánea los incrementos de velocidades. Sin embargo, la realidad es que las aceleraciones usadas en la navegación de robots móviles debe ser baja para obtener trayectorias suaves y no forzar a la estructura mecánica del robot a tensiones excesivas.

Por otra parte, los enfoques de Simmons y Fox han demostrado ser capaces de controlar con éxito robots móviles que actuan en entornos de oficina con gran cantidad de obstáculos y personas en movimiento (una universidad en el primer caso, y un museo en el segundo). Sin embargo, se trata de propuestas difícilmente generalizables a otro tipo de esquemas de conducta distintos de la evitación de obstáculos, como puede ser el seguimiento de una pared, la localización de esquinas o la localización y entrada en puertas abiertas.

Hay que hacer notar también que en todos estos enfoques se trabaja con información proporcionada por sensores de rango de baja densidad y limitado alcance, como son los ultrasonidos. Esto provoca, entre otros, el problema de ambigüedad en la percepción denominado *perceptual aliasing*, en el que situaciones del robot distintas, en las que se deberían tomar acciones también distintas, se solapan en una misma percepción.

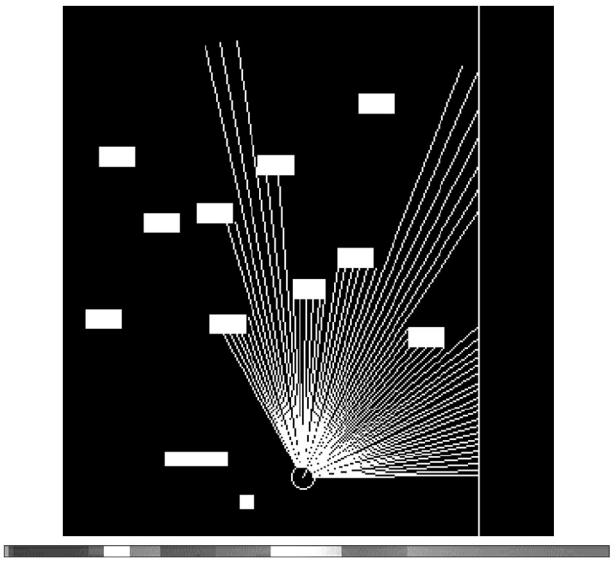


Figura D.1: Una muestra del simulador sobre el que se ha realizado la experimentación del artículo. En la parte inferior de la figura se incluye el mapa de profundidad captado por el robot. Tonos de gris oscuros corresponden con lecturas de profundidad cercanas.

D.1.2 Control local basado en reconocimiento estadístico de situaciones

En línea con las propuestas de Arkin y Chapman (Chapman 1991), la acción a ejecutar vendrá indexada por el esquema de conducta activo en ese momento y por la situación del entorno percibida por el robot. En la propuesta que presentamos en este trabajo formulamos el problema del control local de un robot móvil como un problema de reconocimiento de situaciones. Asociado a cada uno de los esquemas de actuación definimos un conjunto de acciones aplicables y aprendemos las situaciones de percepción en las que esas deben aplicarse.

Para obtener muestras de aprendizaje de un funcionamiento correcto del esquema optimizamos una función que considera la trayectoria seguida por el robot, tanto en el espacio cartesiano como en el espacio de velocidades lineales y angulares, y que premia trayectorias consistentes con el esquema. Por ejemplo, para un esquema `seguir-pared` consideramos que la distancia cartesiana a la pared debe ser pequeña y uniforme, premiando aquellas trayectorias con mayores velocidades lineales. Dado que el espacio de búsqueda es enorme, y que la función a optimizar no es susceptible de ser diferenciada, utilizamos para su resolución la técnica de algoritmos genéticos, diseñando una codificación de las trayectorias del

robot y un método de cruzamiento que han demostrado tener una alta efectividad.

A partir de las trayectorias obtenidas se generan todas las parejas de percepción y acción que el robot ha ido encontrando en la misma, y se agrupan en situaciones percibidas para un mismo tipo de acción.

La dimensionalidad del espacio de percepción debe ser alta para poder establecer diferencias entre las pautas de percepción asociadas a distintos esquemas de conducta. Por ello utilizamos como entrada percivida el campo denso de profundidad existente frente al robot móvil. Aunque hemos desarrollado el trabajo sobre un simulador (ver figura D.1), existen técnicas que permiten obtener este campo de profundidad en tiempo real mediante técnicas actuales de visión artificial (ver (Kanade, Kano, Kimura, Yoshida, y Oda 1995) como ejemplo de utilización de visión estéreo).

Una forma de caracterizar estas situaciones es utilizar un técnica estadística estándar como es el Análisis de Componentes Principales (Fukunaga 1990) para reducir la dimensionalidad de las muestras de aprendizaje correspondientes a cada situación. Veremos que el análisis de componentes principales permite reducir los mapas densos de profundidad a unos pocos parámetros en los que se mantienen la identidad propia de cada situación y que pueden utilizarse de forma efectiva para el reconocimiento.

D.2 Aprendizaje y clasificación de situaciones

Una vez se han generado un conjunto de trayectorias correctas para el esquema que se está aprendiendo, utilizando el método propuesto en el apéndice anterior, se trata de caracterizar las situaciones en las que el robot se va a encontrar cuando evolucione siguiendo ese esquema.

Modelamos el estado en el que puede encontrarse un robot móvil evolucionando por un entorno mediante: 1) un mapa de denso de profundidad observado por el robot $d(\theta)$ y 2) las velocidades lineales y angulares del mismo (v y ω). Definimos un mapa denso de profundidad como el vector $d(\theta)$, que nos indica la distancia a la que se encuentra el obstáculo más cercano en la orientación θ (considerando como 0 la orientación frontal). La obtención de estos mapas de profundidad es inmediata en el simulador. Los límites del ángulo θ vienen dados por las características de la cámara. Para el presente artículo hemos variado θ entre -45 y 45 grados.

A partir de las trayectorias generadas por algoritmo genéticos se generan las estados que ha encontrado el robot en su evolución siguiendo dichas trayectorias. Estos estados se agrupan en conjuntos de situaciones prototipo que son aprendidos y reconocidos de la forma que se explica a continuación. Un ejemplo de campos de profundidad asociado a una situación determinada (en concreto, velocidad lineal del robot entre 60 y 70 cm/s) se presenta en la figura D.2. Cada fila de la figura corresponde al mapa de profundidad en un instante de tiempo.

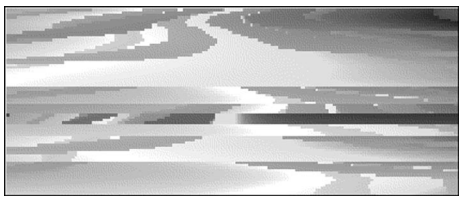


Figura D.2: Ejemplo del conjunto de muestras de entrenamiento de la situación correspondiente a v_{60} ($v \in [60cm/s, 70cm/s]$). Cada fila de la figura corresponde al mapa de profundidad en un instante de tiempo.

D.2.1 Análisis de Componentes Principales

El análisis de componentes principales ha sido utilizado con éxito recientemente en la comunidad de visión artificial para representar imágenes de caras humanas (Sirovich y Kirby 1997) y para reconocer imágenes de caras (Turk y Pentland 1991). Con esta técnica se calculan los autovectores del conjunto de muestras de alta dimensionalidad y son usados como base ortogonal para representar cada una de las muestras individuales. Estos autovectores constituyen la dimensión de un subespacio de muestras, denominado el *autoespacio*, en el que las muestras se pueden representar de forma compacta. Utilizaremos este enfoque aplicándolo al conjunto de mapas de profundidad de cada una de las situaciones.

Representamos, pues, esta distribución de mapas de profundidad mediante una función de perturbación alrededor de un mapa de profundidad medio correspondiente a la situación:

$$d(\theta) = d_0(\theta) + \rho(\theta) \tag{D.1}$$

donde $\rho(\theta)$ son pequeñas fluctuaciones $\rho/d_0 < 1$ que capturan la identidad de cada una de las muestras. El mapa de profundidad medio se calcula a partir de las muestras de entrenamiento correspondientes a la situación

$$d_0(\theta) = \frac{1}{N}\sum_{i=1}^{n} d_i(\theta) \tag{D.2}$$

Para representar $\rho(\theta)$ de cada situación realizamos un análisis de las componentes principales y se expanden las fluctuaciones en térmimos de un conjunto de autovectores Ψ_l extraidos a partir de cada distribución siguiendo el procedimiento estándar (Fukunaga 1990).

$$d(\theta) = d_0(\theta) + \sum_{l} a_l \Psi_l(\theta) \tag{D.3}$$

De esta forma, un mapa denso de profundidad correspondiente a una situación determinada pasa a representarse como una combinación lineal de los modos de variación más

importantes (Ψ_l) sumado al mapa de profundidad medio. Por ello, considerando el nuevo espacio paramétrico definido por los modos de variación principales, un mapa de profundidad pasa a representarse por el vector $b = (b_1, \ldots, b_l)$ correspondiente a las coordenadas en el nuevo espacio paramétrico, reduciéndose considerablemente la dimensionalidad de la distribución. Los autovalores λ_l asociados a cada uno de los autovectores representan la varianza de cada uno de los modos de variación principales.

D.2.2 Clasificación

La proyección b de un mapa de profundidad d en un autoespacio P_l se obtiene mediante la ecuación

$$b = P_l^T (d - d_0), \tag{D.4}$$

siendo $P_l = [\Psi_1, \ldots, \Psi_l]$ la matriz con los l primeros autovectores.

Una métrica muy usada para cuantificar la pertenencia de una muestra a una distribución es la distancia de Mahalanobis, que mide la distancia de la muestra al centro de una distribución, ponderada por la varianza en cada uno de las dimensiones del espacio: D_{Mah}:

$$D_{\text{Mah}}(b, \lambda) = \sum_{k=1}^{l} \left(\frac{b_k^2}{\lambda_k} \right) \tag{D.5}$$

El criterio usado para medir a que situación pertenece un mapa de profundidad percibido es escoger aquella situación cuyo autoespacio minimiza la distancia de Mahalanobis con el mapa de profundidad percibido:

$$\text{situación actual} = \min_{i=0}^{t} D_{\text{Mah}}(b_i, \lambda_i), \tag{D.6}$$

siendo t el numero de situaciones aprendidas, b_i la proyección del mapa de profundidad actual en el autoespacio correspondiente a la situación i y λ_i los autovalores de la situación i.

D.3 Resultados

Se ha realizado una implementación del esquema `avanzar-evitando-obstáculos` siguiendo la propuesta del trabajo. Para ello se han generado comportamientos correspondientes a ese esquema en distintos entornos aleatorios.

A partir de estas trayectorias hemos considerado 8 situaciones correspondientes a velocidades lineales (v_{10}, v_{20}, $\ldots$, v_{80}), con v_i agrupando las velocidades lineales en el rango (i cm/s,$i + 10$ cm/s), y 3 situaciones correspondientes a velocidades angulares (ω_{-10},ω_0,ω_{10}),

Figura D.3: Ejemplo del conjunto de muestras de entrenamiento de las situaciones correspondiente a v_{50}, v_{60} y v_{70}.

con ω_i agrupando las velocidades angulares en el rango ($i - 5$ grados/s, $i + 5$ grados/s). Se han obtenido muestras de los mapas de profundidad percibidos por el robot en cada una de estas situaciones (en la figura D.2 se puede observar un ejemplo de los mapas de profundidad asociados las situaciones v_{50},v_{60} y v_{70}).

Se ha realizado un análisis de componentes principales de las muestras correspondientes a cada una de las situaciones, obteniéndose el autoespacio asociado a cada una de ellas. En la figura D.4 se muestran los mapas de profundidad medios correspondientes a las situaciones $v_{10}, \ldots, v_{80}$. En ellos se representa la distancia media (en centímetros) a la que se encuentran obstáculos (desde -45 grados hasta 45 grados en dirección frontal) cuando el robot se movía a la velocidad correspondiente a cada una de las situaciones. Se puede ver que es coherente con lo esperado: a velocidades más altas el robot se encuentra los obstáculos a mayor distancia.

Por último, la figura D.5 muestra un ejemplo de evolución del robot utilizando el reconocimiento de situaciones propuesto anteriormente. Cada 0,2 segundos se realiza una lectura del mapa de profundidad del entorno, se proyecta esa lectura sobre los autoespacios correspondientes a cada una de las situaciones aprendidas y se obtiene la velocidad lineal y angular a la que debería estar moviéndose el móvil (aquellas con las que se minimiza su distancia de Mahalanobis), modificándose las velocidades actuales consecuentemente.

D.4 Conclusiones

Se ha presentado un enfoque con el que es posible aprender automáticamente esquemas locales de conducta que guían la navegación de un robot móvil, aplicándose al ejemplo concreto de avanzar evitando obstáculos. El método se puede resumir en: 1) generación off-line de trayectorias consistentes con el esquema de navegación, 2) aprendizaje de las situaciones de percepción (mapas densos de profundidad) que el robot se ha encontrado cuando estaba

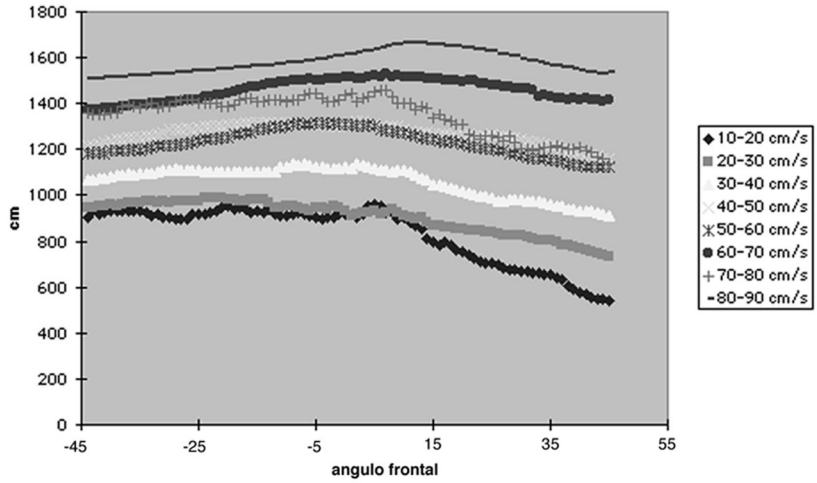


Figura D.4: Mapas de profundidad medios correspondientes a cada una de las distintas situaciones de velocidad lineal.

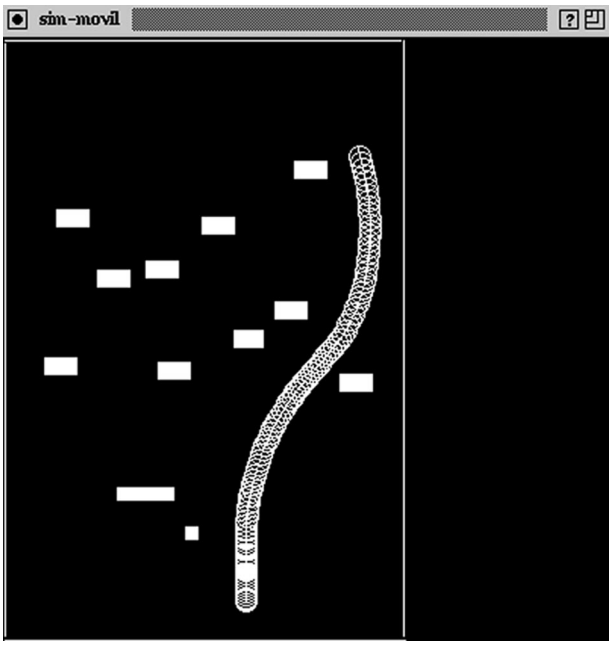


Figura D.5: Ejemplo de trayectoria seguida por el robot aplicando el algoritmo de control basado en reconocimiento estadístico de situaciones.

evolucionando en el entorno, agrupados por situaciones discretas (robot moviéndose entre 10 y 20 cm/s, robot moviéndose entre 20 y 30 cm/s, etc.) y 3) control del robot basado en el reconocimiento de situaciones.

El primer aspecto se ha llevado a cabo utilizando técnicas de algoritmos genéticos, el segundo con un análisis de componentes principales y el tercero utilizando las distancias a las distribuciones aprendidas.

Como trabajo futuro, estamos comenzando a caracterizar otros esquemas utilizando estas técnicas (como `seguir-pared` o `entrar-en-puerta`) al tiempo que prentendemos comprobar la validez del planteamiento en un robot real.

Referencias

Ahuactzin, J., E. Talbi, P. Bessiere, y E. Mazer (1992). Using genetic algorithms for robot motion planning. En *European Conference on Artificial Intelligence (ECAI92)*.

Arkin, R. (1989). Motor schema based mobile robot navigation. *The International Journal of Robotics Research 8*(4), 92–112.

Arkin, R. (1990). Integrating behavioral, perceptual, and world knowlege in reactive navigation. *Robotics and Autonomous Systems 6*, 105–122.

Ayache, N. y D. Faugeras (1989, December). Maintaining representations of the environment of a mobile robot. *IEEE Transactions on Robotics and Automation 5*(6), 804–819.

Ayrulu, B., B. Barshan, y S. Utete (1997). Target identification with multiple logical sonars using evidential reasoning and simple majority voting. En *Proceedings of the 1997 IEEE International Conference Robotics and Automation*.

Back, T., D. Fogel, y Z. Michalewicz (1997). *Handbook of Evolutionary Computation*. Oxford University Press.

Bar-Shalom, Y. y T. Fortmann (1988). *Tracking and Data Association*. Academic Press.

Barker, A., D. Brown, y W. Martin (1994). Bayesian estimation and the kalman filter. Technical Report IPC-TR-94-002, Institute for Parallel Computation, University of Virginia.

Barshan, B. y R. Kuc (1990, June). Differentiating sonar reflections from corners and planes by employing an intelligent sensor. *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-12*(6), 560–569.

Bauer, R. (1995). Active manoeuvres for supporting the localisation process of an autonomous mobile robot. *Robotics and Autonomous Systems 16*, 39–46.

Bonasso, R. y T. Dean (1996). Robots with ai: A retrospective on the aaai robot competitions and exhibitions. En *Proceedings of the 13th National Conference on Artificial Intelligence*.

Borenstein, J. y Y. Korem (1991). The vector field histogram – fast obstacle avoidance for mobile robots. *IEEE Transctions on Robotics and Automation 7*(3), 278–288.

Borenstein, J. y Y. Korem (1995). Error eliminating rapid ultrasonic firing for mobile robot obstacle avoidance. *IEEE Transctions on Robotics and Automation 11*(1), 132–138.

Brooks, R. (1986). A robust layered control system for a mobile robot. *IEEE Journal on Robotics and Automation 2*(1), 14–23.

Burgard, W., A. Cremers, D. Fox, D. Hahnel, G. Lakemeyer, D. Schulz, W. Steiner, y S. Thrun (1998). The interactive museum tour-guide robot. En *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98)*, Madison, Wisconsin. AAAI Press.

Burgard, W., D. Fox, D. Henning, y T. Schmidt (1996). Estimating the absolute position of a mobile robot using position probability grids. En *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI-96)*, Menlo Park. AAAI Press.

Burgard, W., D. Fox, y S. Thrun (1997). Active mobile robot localization. En *Proceedings of the IJCAI-97*. IJCAI.

Cappe, O., A. Doucet, M. Lavielle, y E. Moulines (1999). Simulation-based methods for blind maximum-likelihood filter identification. *Signal processing 73*, 3–25.

Carpenter, J., P. Clifford, y P. Fernhead (1997). An improved particle filter for non-linear problems. Technical report, Dept. of Statistics, University of Oxford.

Cassandra, A., L. Kaelbling, y M. Littman (1994). Acting optimally in partialy observable stochastic domains. En *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI-94)*, Seattle, WA.

Chapman, D. (1991). *Vision, instruction and action*. MIT Press.

Chatila, R., M. Khatib, H. Jaouni, y J.-P. Laumond (1997). Dynamic path modification for car-like non-holonomic mobile robots. En *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*.

Chatila, R. y J. Laumond (1985). Position referencing and consistent world modeling for mobile robots. En *Proceedings of the 1985 IEEE International Conference on Robotics and Automation*, pp. 138–145.

Chong, K. y L. Kleeman (1997). Sonar based map building for a mobile robot. En *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, pp. 1700–1705.

Courtney, J. y A. Jain (1994). Mobile robot localization via classification of multisensor maps. En *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pp. 1672–1678.

Cox, I. y J. Leonard (1994). Modeling a dynamic environment using a bayesian multiple hypothesis approach. *Artificial Intelligence 66*, 311–344.

Cox, I. J. (April, 1991). Blanche – an experiment in guidance and navigation of an autonomous robot vehicle. *IEEE Transactions on Robotics and Automation 7*(2), 193–204.

Daniel Pagac, Eduardo M.Ñebot, H. D.-W. (1996, May). An evidential approach to probabilistic map-building. En *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, Los Alamitos, CA, USA. IEEE Computer Society Press.

Dean, T., L. Kaelbling, J. Kirkman, y A.Ñicholson (1993). Planning with deadlines in stochastic domains. En *Proceedings of the 11th National Conference on Artificial Intelligence (AAAI'93)*, pp. 574–579. AAAI Press.

Dean, T. L. y M. P. Wellman (1991). *Planning and Control*. San Mateo, California: Morgan Kaufmann.

Dempster, A., A. Laird, y D. Rubin (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B 39*, 1 38.

D.Fox, W. Burgard, y S. Thrun (1997). The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine march*, 23–33.

Divelbiss, A. y J. Wen (1994). Nonholonomic motion planning with inequality constraints. En *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*.

Doyle, A. (1995). *Algorithms and computational techniques for robot path planning*. Ph. D. thesis, School of Electronic Engineering and Computer Systems, University of Wales.

Drumheller, M. (1987). Mobile robot localization using sonar. *IEEE Transactions on Pattern Analysis and Machine Intelligence 9*(2), 325–332. See also: S. B. Thesis, Dept. of Mechanical Engineering, MIT, 1984 and MIT AI Lab Memo BZ6, Mobile Robot Localization Using Sonar.

Dudek, G., Jenkin, Milos, y Wilkes (1992). Reflections on sonar range sensing. Technical Report CIM-92-09, McGill Centre for Intelligent Machines.

Edingler, T., E. Puttkamer, y R. Trieb (1991). Accurate postion estimation for an autonomous mobile robot fusing shaft encoder values and laser range data. En *Proceedings of the International Advanced Robotics Program (IARP-91)*.

Elfes, A. (1987, June). Sonar-based real-world mapping and navigation. *IEEE Journal of Robotics and Automation RA-3*(3), 249–265.

Elfes, A. (1989, June). Using occupancy grids for mobile robot perception and navigation. *Computer 22*(6), 46–57.

Engelson, S. y D. McDermott (1992). Error correction in mobile robot map learning. En *Proceedings of the 1992 IEEE International Conference on Robotics and Automation*, pp. 2555–2560.

Escolano, F. (1997). *Plantillas Deformables Extendidas*. Ph. D. thesis, Departamento de Tecnología Informática y Computación.

Escolano, F., M. Cazorla, D. Gallardo, F. Llorens, R. Satorre, y R. Rizo (1997). Plantillas espacio-temporales para el tracking y reconocimiento gestual. En *Actas de CAEPIA '97. Conferencia de la Asociación Española para la Inteligencia Artificial*.

Escolano, F., M. A. Cazorla, D. Gallardo, F. Llorens, R. Satorre, y R. Rizo (1998). A combined probabilistic framework for learning gestures and actions. En *Proceedings of the 11th IEA-IAE International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*.

Escolano, F., M. A. Cazorla, D. Gallardo, y R. Rizo (1997). Deformable templates for tracking and analysis of intravascular ultrasound sequences. En M. Pelillo y E. R. Hancock (Eds.), *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pp. 521–534. Springer.

Fleeman, L. y R. Kuc (1994). An optimal sonar array for target localization and classification. En *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pp. 3130–3135.

Fox, D., W. Burgard, S. Thrun, y A. Cremers (1998a). A hybrid collision avoidance method for mobile robots. En *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*.

Fox, D., W. Burgard, S. Thrun, y A. Cremers (1998b). Position estimation for mobile robots in dynamic environments. En *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI'98)*. AAAI Press.

Fukunaga, K. (1990). *Introduction to Statistical Pattern Recognition*. Academic Press.

Gallardo, D., O. Colomina, F. Flórez, P. Arques, P. Company, y R. Rizo (1997). Control local de robots móviles basado en métodos estadísticos y algoritmos genéticos. En *Actas de CAEPIA '97. Conferencia de la Asociación Española para la Inteligencia Artificial*.

Gallardo, D., O. Colomina, F. Flórez, y R. Rizo (1998). A robust algorithm for robust motion planning. En *Proceedings of the 11th IEA-IAE International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*.

Gallardo, D., F. Escolano, R. Rizo, O. Colomina, y M. Cazorla (1998). Estimación bayesiana de características en robots móviles mediante muestreo de la densidad a posteriori. En *Actas del Primer Congrés Català dÍntelligència Artificial*.

Geman, S. y D. Geman (1984). Stochastic relaxation, gibbs distributions and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence 6*(6), 721–741.

Glassner, A. (1989). *An introduction to Ray Tracing*. Academic Press.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley.

Gordon, N., D. Salmond, y A. Smith (1993). Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE procedings F 140*, 107–113.

Grenander, U. (1993). *General Pattern Theory: A Mathematical Study of Regular Structures*. Oxford University Press.

Grimson, W. (1990). *Objecto Recognition by Computer: The Role of Geometric Constraints*. MIT Press.

Gutmann, J. y C. Schlegel (1996). Amos: Comparison of scan matching apporaches for self-localization in indoor environments. En *Proceedings of EUROBOT'96*, pp. 61–67.

Henderson, T., B. Bruderlin, M. Dekhil, L. Schenkat, y L. Veigel (1996). Sonar sensing strategies. En *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*.

Hendler, J., A. Tate, y M. Drummond (1990). Ai planning: Systems and techniques. *AI Magazine 11*(2).

Hinkle, D., D. Kortenkamp, y D. Miller (1996). 1995 robot competition and exhibition. *AAAI Magazine 17*(1), 31–45.

Hornegger, J. y H.Ñiemann (1997). Optimization problems in statistical object recognition. En M. Pelillo y E. R. Hancock (Eds.), *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pp. 311–326. Springer.

Hwang, Y. (1992). A potential field approach to path planning. *IEEE Transactions on Robotics and Automation 8*(1), 23–32.

Isard, M. y A. Blake (1996). Contour tracking by stochastic propagation of conditional density. En *Proc. European Conf. Computer Vision 1996 (ECCV'96)*, pp. 343–356. Springer Verlag.

Isard, M. y A. Blake (1998a). Condensation — conditional density propagation for visual tracking. *International Journal on Computer Vision*.

Isard, M. y A. Blake (1998b). A smoothing filter for condensation. En *Proc 5th European Conf. Computer Vision*, Volumen 1, pp. 767–781.

J. Borenstein, B. Everett, L. F. (1996). *Navigating Mobile Robots: Systems and Techniques*. A.K. Peters, Ltd.

Kaelbling, L., A. Cassandra, y J. Kurien (1996). Acting under uncertainty: discrete bayesian models for mobile-robot navigation. En *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'96)*, pp. 963–972.

Kaelbling, L., M. Littman, y A. Cassandra (1995). Planning and acting in partially observable stochastic domains. Technical report, Department of Computer Science, Brown University.

Kalman, R. E. (1960). A new approach to lineal filtering and prediction problems. *Transactions of the ASME – Journal of basic engineering March*, 35–45.

Kanade, T., H. Kano, S. Kimura, A. Yoshida, y K. Oda (1995). Development of a video-rate stereo machine. En *Proc. of International Robotics and Systems Conference (IROS-95)*.

Khatib, O. (1985). Real-time obstacle avoidance for manipulators and mobile robots. En *Proceedings of the 1985 IEEE International Conference on Robotics and Automation*, pp. 500–505.

Kim, J. y P. Khosla (1991). Real-time obstacle avoidance using harmonic potential functions. En *Proceedings of the 1991 IEEE International Conference on Robotics and Automation*, pp. 790–796.

Kitawa, G. (1987). Non-gaussian state-space modelling of non-stationary time series. *Journal of the American Statistical Association 82*, 1032–1063.

Kitawa, G. (1996). Monte carlo filter and smoother for non-gaussian nonlinear state space models. *Journal of Computational and Graphical Statisitcs 5*, 1–25.

Ko, J., S. Kim, y M. Chung (1996). A method of indoor mobile robot navigation using acoustic landmarks. En *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, pp. 1726–1731.

Koenig, S. y R. Simmons (1996). Passive distance learning for robot navigation. En *Proceedings of the 13th International Conference on Machine Learning (ICML)*, pp. 266–274. Morgan Kaufmann Publishers.

Koenig, S. y R. Simmons (1998). Xavier: a robot navigation architecture based on partially observable markov decision process models. En D. Kortenkamp, P. Bonasso, y R. Murphi (Eds.), *Artificial Intelligence and Mobile Robots*, pp. 91–122. AAAI Press/MIT Press.

Konolige, K. (1994). Designing the 1993 robot competition. *AAAI Magazine 15*(1), 57–62.

Korba, L. (1994). Variable aperture sonar for mobile robots. En *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pp. 3136–3141.

Kortenkamp, D., P. Bonasso, y R. Murphi (Eds.) (1998). *Artificial Intelligence and Mobile Robots*. AAAI Press/MIT Press.

Kortenkamp, D., I.Ñourbakhsh, y D. Hinkle (1997). The 1996 aaai mobile robot competition and exhibition. *AAAI Magazine 18*(1), 25–32.

Kortenkamp, D. y T. Weymouth (1994). Topological mapping for mobile robots using a combination of sonar and vision sensing. En *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI-94)*, Seattle, WA, pp. 979–984.

Krogh, B. y C. Thorpe (1986). Integrating path planning and dynamic steering control for autonomous vehicles. En *Proceedings of the 1986 IEEE International Conference on Robotics and Automation*, pp. 1664–1669.

Krose, B. J., K. M. Compagner, y F. C. Groen (1993). Accurate estimation of environment parameters from ultrasonic data. *Robotics and Autonomous Systems 11*, 221–230.

Kuc, R. (1990, July). A spatial sampling criterion for sonar obstacle detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-12*(7), 686–690.

Kuc, R. y M. W. Siegel (1987). Physically based simulation model for acoustic sensor based robot navigation. *IEEE Trans. Pattern Analysis and Machine Intelligence 9*(6), 766–768.

Kuipers, B. y Y. Byun (1988). A robust, qualitative method for robot spatial learning. En *Proceedings of the 8th National Conference on Artificial Intelligence (AAAI-88)*, pp. 774–779. AAAI Press.

Kuipers, B. y Y. Byun (1991). A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Robotics and Autonomous Systems 8*, 47–63.

Kuipers, B. y T. Levitt (1988, Summer). Navigation and mapping in large-scale space. *AI Magazine*, 25–43.

Latombe, J. (1991). *Robot Motion Planning*. Kluwer Academic Publishers.

Leonard, J., A. Bradley, I. Cox, y M. Miller (1995). Underwater sonar data fusion using an efficient multiple hypothesis algorithm. En *Proceedings of the 1995 IEEE International Conference on Robotics and Automation*, pp. 2995–3002.

Leonard, J. y H. Durrant-Whyte (1992). *Directed sonar sensing for mobile robot navigation*. Dordrecht, The Netherlands: Kluwer.

Leonard, J., H. Durrant-Whyte, y I. Cox (1992). Dynamic map building for an autonomous mobile robot. *International Journal of Robotics Research 11*(4), 89–96.

Lu, F. y E. Milios (1994). Robot pose estimation in unknown environments by matching 2d range scans. En *Proceedings of the 1994 IEEE Conference on Robotics and Automation*, pp. 935–938.

Lu, F. y E. Milios (1997). Globally consistent range scan alignment for environment mapping. *Autonomous Robots 4*, 333–349.

Manyika, J. M. y H. F. Durrant-Whyte (1993, May). A tracking sonar sensor for vehicle guidance. En *Proceedings of the 1993 IEEE International Conference on Robotics and Automation*, Atlanta, GE, pp. 424–429. IEEE Computer Society Press.

Mataric, M. (1992, June). Integration of representation into goal-driven behavior-based robots. *IEEE Transactions on Robotics and Automation 8*(3), 304–312.

Matthies, L. y A. Elfes (1988). Integration of sonar and stereo range data using a grid-based representation. En *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, pp. 727–733.

McKerrow, P. J. (1993). Echolocation – from range to outline segments. *Robotics and Autonomous Systems 11*, 205–211.

Mitchel, T. (1997). *Machine Learning*. Mc Graw Hill.

Moravec, H. (1998, Summer). Sensor fusion in certainty grids for mobile robots. *AI Magazine 9*(2), 61–74.

Moravec, H. P. y A. Elfes (1985). High resolution maps from wide angle sonar. En *Proceedings of the 1985 IEEE International Conference on Robotics and Automation*, pp. 116–121.

Neira, J., J. Horn, J. Tardos, y G. Schmidt (1997). Multisensor mobile robot localization. En *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, pp. 673–679.

North, B. y A. Blake (1998). Learning dynamical models by expectation maximisation. En *Proceedings of the 6th International Conference on Computer Vision*.

Nourbakhsh, I., R. Powers, y S. Birchfeild (1995, Summer). Dervish, an office-navigating robot. *AI Magazine 16*(2), 53–60.

Ohya, A., Y.Ñagashima, y S. Yuta (1994). Exploring unknown environment and map construction using ultrasonic sensing of normal direction of walls. En *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pp. 485–492.

Ondet, A. y J. Babry (1989). Modeling of sound propagation in fitted workshops using ray tracing. *J. Acousti. Soc. Am. 85*(2), 787–796.

Oriolo, G., M. Vendittelli, y G. Ulivi (1995). On-line map building and navigation for autonomous mobile robots. En *Proceedings of the 1995 IEEE International Conference on Robotics and Automation*, pp. 2900–2906.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, Inc.

Pierce, D. y B. Kuipers (1994). Learning to explore and build maps. En *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI-94)*, pp. 1264–1271. AAAI Press.

Pitt, M. y N. Shephard (1997). Filtering via simulation: auxiliary particle filters. Technical report, Dept. of Mathematics, Imperial College.

Polaroid Corp. (1984). *Ultrasonic Ranging System*. Cambridge,MA.

Prescott, T. y J. Mayhew (1992). Adaptive local navigation. En A. Yuille y P. Hallinan (Eds.), *Active Vision*, pp. 203–215. MIT Press.

Press, W., S. Teukolsky, W. Vetterling, y B. Flannery (1988). *Numerical Recipes in C*. Cambridge University Press.

Rabiner, L. y B. Juang (1986, January). An introduction to hidden markov models. *IEEE ASSP Magazine 3*(1), 4–16.

Rao, B. (1992). Data association methods for tracking systems. En A. Yuille y P. Hallinan (Eds.), *Active Vision*, pp. 91–105. MIT Press.

Rencken, W. (1993). Concurrent localisation and map building for mobile robots using ultrasonic sensors. En *Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'93)*, pp. 2192–2197.

Rendas, M. y W.Tetenoire (1997). Definition of exploratory trajectories in robotics using genetic algorithms. En *Tenth International Conference in Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*.

Ryu, B. y H. Yang (1988). An enhaced topological map for efficient and reliable mobile robot navigation with imprecise sensors. *Robotics and Computer-Integrated Manufacturing 14*, 185–197.

Sabatini, A. M. y O. D. Benedetto (1994). Towards a robust methodology for mobile robot localization using sonar. En *Proceedings of the 1994 IEEE International Conference on Robotics and Automation.*, pp. 3142–3148.

Schiele, B. y J. Crowley (1994). A comparison of position estimation techniques using occupancy grids. En *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pp. 1628–1634. IEEE Computer Society Press.

Schneider, J. (1993). High dimension action spaces in robot skill learning. Technical report, Department of Computer Science, University of Rochester.

Shatkay, H. y L. Kaelbling (1997). Learning hidden markov models with geometric information. Technical Report CS-97-04, Department of Computer Science, Brown University.

Simmons, R. (1995). The aaai mobile robot competition and exhibition. *AAAI Magazine 16*(2), 19–30.

Simmons, R. (1996). The curvature-velocity method for local obstacle avoidance. En *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*.

Simmons, R. y S. Koenig (1995). Probabilistic navigation in partially observable environments. En *Proceedings of the IJCAI-95*, Montreal, Canada, pp. 1080–1087.

Singh, S. (1995). *Synthesis of Tactical Plans for Robotic Excavation*. Ph. D. thesis, The Robotics Institute, Carnegie Mellon University.

Sirovich, L. y M. Kirby (1997). Low dimensional procedure for the characterization of human faces. *Journal of Optical Society of America 4*, 519–524.

Slack, M. G. (1993). Fixed computation real-time sonar fusion for local navigation. En *Proceedings of the 1993 IEEE International Conference on Robotics and Automation*, pp. 123–129.

Sorenson, H. (1970). Least-squares estimation: from gauss to kalman. *IEEE Spectrum 7*(7), 63–68.

Stephenson, U. (1990). Comparison of the mirror image source method and the sound particle method. *Applied Acoustics 29*, 35–72.

Stevens, A., M. Stevens, y H. Durrant-Whyte (1995). Oxnav: Reliable autonomous navigation. En *Proceedings of the 1995 International Conference on Robotics and Automation*, pp. 2607–2612.

Thrun, S. (1997). To know or not to know: On the utility of models in mobile robotics. *AI Magazine 18*(1), 47 54.

Thrun, S. (1998). Learnig metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence 99*(1), 21–72.

Thrun, S., A. Bucken, W. Burgard, D. Fox, T. Frohlinghaus, D. Hennig, T. Hofmann, M. Krell, y T. Schmidt (1998). Map learning and high-speed navigation in rhino. En D. Kortenkamp, P. Bonasso, y R. Murphi (Eds.), *Artificial Intelligence and Mobile Robots*, pp. 21–52. AAAI Press/MIT Press.

Thrun, S., W. Burgard, y D. Fox (1998). A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning and Autonomous Robots (joint issue)*.

Turk, M. A. y A. P. Pentland (1991). Face recognition using eigenfaces. En *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*.

Vian, J. y D. van Maercke (1986). Calculation of the room impulse response using a ray-tracing method. En *12th ICA, Proceedings of the Vancouver Symposium*.

Vlieger, J. H. d. y R. H. J. G. Meyling (1992). Maximum likelihood estimation for long-range target tracking using passive sonar measurements. *IEEE Transactions on Signal Processing 40*(5).

Watt, A. y M. Watt (1992). *Advanced Animation and Rendering Techniques*. ACM Press.

Weib, G., C. Wetzler, y E. Puttkamer (1994). Keeping track of position and orientation of moving indoor systems by correlation of range-finder scans. En *Proceedings of the International Conference on Intelligent Robots and Systems*, pp. 595–601.

Weigl, M., B. Siemiatkowska, K. Sikorski, y A. Borkowski (1993). Grid-based mapping for autonomous mobile robot. *Robotics and Autonomous Systems 11*, 13–21.

West, M. (1992). Modeling with mixtures. *Bayesian Statistics 4*, 503–524.

Xiao, J., Z. Michalewicz, L. Zhang, y K. Trojanowski (1997). Adaptive evolutionary planner/navigator for robots. *IEEE Trans. on Evolutionary Computation 1*.

Yamauchi, B. (1996, May). Mobile robot localization in dynamic environments using dead reckoning and evidence grids. En *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, Los Alamitos, CA, USA. IEEE Computer Society Press.

Yamauchi, B. y P. Langley (1997). Place recognition in dynamic environments. *Journal of Robotic Systems, Special Issue on Mobile Robots 14*, 107–120.

Zelinsky, A. (1991, October). Mobile robot map making using sonar. *Journal of Robotic Systems 8*(5), 557–577.

Zhang, Y. y R. Webber (1992). On combining the hough transform and occupancy grid methods for detection of moving objects. En *Proceedings of the 1992 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'92)*, pp. 2155–2160.