UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

Departamento de Electrónica e Computación

TESIS DOCTORAL

LOCALIZACIÓN Y MAPEADO SIMULTÁNEOS EN ROBÓTICA MEDIANTE VISIÓN OMNIDIRECCIONAL

Presentada por:

Cristina Gamallo Solórzano

Dirigida por:

Dr. Carlos Vázquez Regueiro

Dr. Manuel Mucientes Molina

Santiago de Compostela, julio de 2013

Dr. Manuel Mucientes Molina, Investigador Ramón y Cajal del Área de Ciencias de la Computación e Inteligencia Artificial de la Universidade de Santiago de Compostela

Dr. Carlos Vázquez Regueiro, Profesor Titular de Universidad del Área de Arquitectura y Tecnología de Computadores de la Universidade da Coruña

HACEN CONSTAR:

Que la memoria titulada **LOCALIZACIÓN Y MAPEADO SIMULTÁNEOS EN ROBÓTICA MEDIANTE VISIÓN OMNIDIRECCIONAL** ha sido realizada por Dña. **Cristina Gamallo Solórzano** bajo nuestra dirección en el marco del *programa de doctorado Interuniversitario en Tecnologías de la Información* del Departamento de Electrónica e Computación de la Universidade de Santiago de Compostela, y constituye la Tesis que presenta para optar al grado de Doctor.

Santiago de Compostela, julio de 2013

Fdo.: **Dr. Manuel Mucientes Molina**
Codirector de la tesis

Fdo.: **Dr. Carlos Vázquez Regueiro**
Codirector de la tesis

Fdo.: **Cristina Gamallo Solórzano**
Autora de la tesis

Miña nai, miña naiciña
coma miña nai nenghunha
que me quentou a cariña
co calorciño da súa.

Manuel foi o primeiro
que me entrou no corazón
ha de ser o derradeiro
después de la salvación.

María ramo de palma
María ramo de palma
lévote no corazón
atravesada na ialma.

Miña irmá, miña irmá, miña amiga
tan igual a min e tan distinta
dame a man e farémonos grandes
temos xuntas a forza dos mares.

A voz das miñas amigas
Dame folgos pra avanzar
Se non fose por vosoutras
como ía camiñar.

AGRADECIMIENTOS

En primer lugar, quiero expresar mi más sincero agradecimiento a mis directores de tesis, Dr. Carlos Vázquez Regueiro y Dr. Manuel Mucientes Molina, por la confianza y ayuda que me han brindado durante la elaboración de la tesis, sin las cuales la presente investigación no hubiera sido posible. Pero sobre todo, porque son en parte los responsables de mi crecimiento profesional y también personal a lo largo de toda esta etapa.

Al Departamento de Electrónica e Computación de la Universidade de Santiago de Compostela y al Departamento de Electrónica e Sistemas de la Universidade da Coruña, por proporcionar los recursos necesarios para la realización de esta tesis. A los miembros del Grupo de Sistemas Inteligentes, por su apoyo y colaboración en todo momento durante estos años, en particular a Roberto Iglesias, Alberto Bugarín, José L. Correa y Paulo Félix.

Y por supuesto, me gustaría agradecer la financiación que me ha permitido desarrollar esta investigación: proyecto INCITE08PXIB262202PR de la Xunta de Galicia, ayudas CN2011/058 y CN2012/151 cofinanciadas por el Fondo Europeo de Desarrollo Regional (FEDER) y la Xunta de Galicia, y proyectos TIN2011-22935, TIN2009-07737, PSS-310000-2009-003, TIN2008-04008/TSI, TIN2005-03844 y FCT-11-2489 del Gobierno de España.

Gracias también a todos mis compañeros del DEC y del CITIUS que me han hecho agradables estos años de intenso trabajo. En especial, a mis compis del 14 (Fabi, Carlos, Ronald, Bak, María, Miguel, Josito, J.C. y Adrián), a los robóticos (Pablo, Adri y Víctor), a los cafeteros (Óscar, Juan, Manolo, Roi, Pablo y Flora), a mis comensales favoritos (Bea, Julián, Noe, Xulio, Enrique, Juan Ángel y María L.), a los más nocturnos (Fer, David, Álex, Diego C., Lois, ...), y a los *mayores* (Diego, Álvaro, Natalia y Lebo), por todos los manjares compartidos: cafés, marrones, penas, comidas, dolores, copas, vinos, tapas, alegrías, ...

Mil gracias a mis amigos y amigas (Mery, Luz, Nati, Glo, San, Sara, Sarius, Helen, Chanin, Crisvi, Rous, Rebius, Yordi, Dani, Chusita, Vane, Ana, Yoya, Lara, Fer, Pili, Anita, Rul,

Pani, Jetxu, More, Chavalas y Chavales, Pasadizos consortes, los Gonsus, la gente de la FIC, chicos y chicas Celme, Erasmitus, Santiagueses, amigos del folklore y demás gentuza) por hacerme disfrutar de mis ratos libres.

A mi familia: *Gamallo's* y *Solórzano's* por soportarme y dejarme crecer con vosotros. Especialmente a María por su infinita generosidad, a Su por sus genialidades, a la Tita por ser una más, a mi tía Mercedes por su fuerza, a mis primos por ser como hermanos, a los más peques por su alegría y a mis abuelas por haber sido la base y energía de esta gran familia. Y por el último infinitas gracias a mis padres, Manolo y Loly, porque sin ellos yo no sería nadie.

Santiago de Compostela, julio de 2013

Querer es poder.
Popular

ÍNDICE GENERAL

PREFACIO

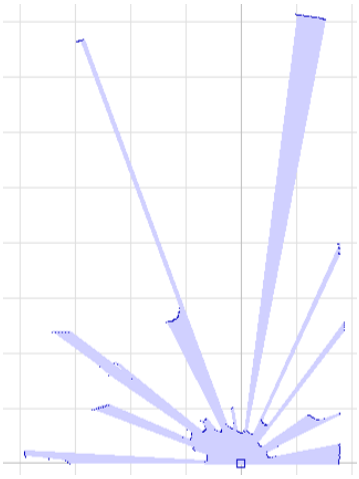
Motivación y objetivos

La operación de un robot autónomo en entornos reales requiere, habitualmente, conocer tanto el mapa del entorno como la ubicación del robot en dicho mapa. Si el mapa es conocido, los algoritmos de localización permiten a un robot estimar su pose (posición y orientación). Si por el contrario el mapa es desconocido, es necesario recurrir a algoritmos de localización y mapeado simultáneos (SLAM, del inglés *Simultaneous Localization And Mapping*), que permiten al mismo tiempo construir el mapa y mantener al robot localizado en el mismo.

La operación de un robot durante la localización y el mapeado en entornos interiores con un gran número de personas en movimiento introduce una serie de dificultades, tanto desde el punto de vista del sensor como de los propios algoritmos de localización o SLAM utilizados. En primer lugar, la pose del sensor juega un papel fundamental: todos los sensores que adquieren datos del plano en el que se mueve el robot no son apropiados, ya que no proporcionan información útil cuando el robot está completamente rodeado de gente (fig. 0.1). Algunas propuestas de SLAM en entornos dinámicos se basan en diferentes técnicas de *tracking* para poder descartar la información proveniente de los objetos móviles [Migliore y col., 2009; Solà, 2007]. Sin embargo, esta solución no es válida cuando la densidad de objetos es muy alta y los datos del sensor están totalmente afectados por la presencia de gente.

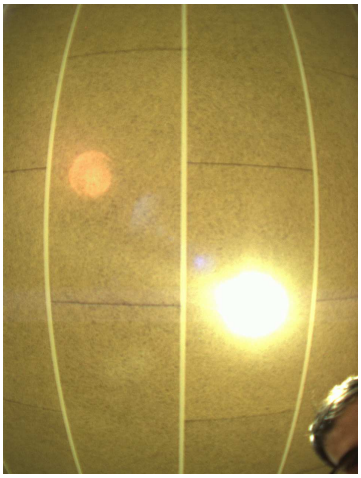
Diversos autores han utilizado cámaras orientadas hacia el techo [Jeong y Lee, 2005; Choi y col., 2010, 2012; Kang y col., 2012; Roda y col., 2007], ya que en entornos con elevada concentración de personas esta es la única zona en la que las medidas no se ven afectadas. En particular, la mayoría de estas aproximaciones usan las luces del techo como balizas, pues los techos suelen tener un número reducido de objetos significativos para mapear. También por este motivo, los descriptores de características tipo SIFT (*Scale-Invariant Feature Transform*), SURF (*Speeded Up Robust Features*), etc., son poco efectivos, pues el techo suele tener pocos

(a) Imagen real.



(b) Medidas del láser.

(c) Imagen de una cámara omnidireccional.



(d) Imagen de una cámara estándar.

Figura 0.1: Oclusión severa debido a la presencia de gente rodeando al robot.

objetos de los que extraer descriptores significativos, y además frecuentemente esos objetos relevantes suelen ser similares o iguales entre sí.

Las cámaras convencionales no son apropiadas para entornos con una baja densidad de balizas, ya que detectan pocas balizas en cada imagen. En cambio, las cámaras omnidireccio-

nales sí son sensores más adecuados en estas condiciones, ya que el número de características detectadas en cada imagen es mayor, y por tanto el robot tiene información más valiosa para corregir su pose y también la posición de las balizas. La combinación de visión omnidireccional y las luces del techo como balizas permite localizar al robot y mapear el entorno en entornos muy concurridos, aún cuando el sistema debe ser todavía capaz de operar bajo oclusiones severas cuando la gente se sitúe muy próxima al robot (fig. 0.1).

Como ya se ha mencionado, un algoritmo de localización o SLAM para entornos concurridos es bastante más complejo que uno convencional. El motivo es que, para la combinación de visión omnidireccional y luces en el techo, la asociación de datos es mucho más difícil por las siguientes razones:

– El uso de descriptores de características (SIFT, SURF, etc.) facilita la etapa de asociación de datos, ya que esta depende tanto de la similaridad entre los descriptores de medidas y balizas como de la distancia geométrica entre ellas. Sin embargo, cuando las balizas son objetos específicos y muy similares (p. ej. las luces), los descriptores de características no proporcionan ninguna información de interés (las balizas son indistinguibles entre sí) y la asociación de datos posee menos información.

– Los sensores que solo proporcionan orientaciones (*bearing-only sensors*), como las cámaras omnidireccionales, hacen la asociación de datos más compleja ya que:

  - Las balizas lejanas pueden generar, para ciertas poses del robot, medidas muy similares aunque sus posiciones en el mapa 3D sean bastante dispares.
  - En la mayoría de los algoritmos de SLAM con sensores que solo proporcionan orientaciones, la inicialización de las balizas se realiza de modo retardado y tiene en cuenta un conjunto de medidas asociadas a la baliza candidata. En consecuencia, la posición de las balizas recientemente inicializadas es ruidosa y la asociación de datos debe tratar con una mayor incertidumbre.
  - Durante la inicialización retardada la posición de las balizas candidatas experimenta cambios frecuentes y rápidos. Por ello, no es extraño que la probabilidad de asociación de una baliza candidata con respecto a una medida generada por una baliza del mapa sea mayor que la asociación correcta.

– Las medidas provenientes de las luces son más ruidosas debido a que: i) las balizas se encuentran en el techo y se detectan siempre a distancias de varios metros; ii) las

características aparecen con diferentes tamaños y formas en la imagen, y cambian dependiendo de la posición desde la que se observan y de las condiciones de iluminación.

– Cuando las oclusiones severas son frecuentes se dificulta la asociación de datos, y también se retarda la inicialización de las balizas candidatas.

El objetivo de esta tesis doctoral ha sido el desarrollo de algoritmos de localización y SLAM para cámaras omnidireccionales utilizando como balizas las luces del entorno. Además, los algoritmos deben operar en tiempo real y bajo oclusiones frecuentes y severas.

– **Localización**. El objetivo es que el algoritmo resuelva los problemas de seguimiento de la posición (*tracking* o localización local), localización global y *secuestro del robot*. Además, debe ser robusto frente al ruido, oclusiones y ante cambios en el entorno, así como escalable con el tamaño del mapa.

– **SLAM**. El algoritmo debe contar con un método de asociación de datos global y realizar la inicialización de las balizas (para sensores que solo proporcionan orientación) siguiendo un modelo probabilístico. La aproximación seguida debe gestionar oclusiones frecuentes y severas, así como la incertidumbre inherente a las luces, causada por distancias de detección elevadas, características en las imágenes de diferentes tamaños y formas, condiciones de iluminación variables, etc.

Contribuciones

Las principales contribuciones de esta memoria son las siguientes:

– OV-MCL es un algoritmo de localización para omnivisión que utiliza como mapa de características el formado por las luces existentes en el entorno. Está basado en la localización de Monte Carlo (MCL, del inglés *Monte Carlo Localization*), opera en tiempo real gracias al número de partículas adaptable, y es robusto a oclusiones y cambios en el entorno. Destacamos las siguientes características del algoritmo:

  - El modelo de medida. Este modelo incluye todos los pasos necesarios para transformar una imagen omnidireccional en un conjunto de características detectadas, así como la evaluación de la probabilidad de asociación entre medidas y balizas del mapa 3D.

  - Número variable de partículas. El método se basa en la medida de divergencia de Kullback-Leibler (KLD, del inglés *Kullback-Leibler Divergence*) y permite la operación de OV-MCL en tiempo real, variando el número de partículas de forma adaptativa de acuerdo con la calidad de la aproximación a la función de densidad de la probabilidad (PDF, del inglés *Probability Density Function*).
  - Inyección de partículas aleatorias. Mediante el análisis de la evolución de la distribución de probabilidad se realiza una inserción de nuevas partículas, lo que permite que OV-MCL sea más robusto y pueda recuperarse frente a cambios importantes en el entorno o en la posición del robot.

- OV-FastSLAM es un algoritmo de SLAM para cámaras omnidireccionales capaz de operar bajo oclusiones severas. Está basado en la propuesta de FastSLAM 2.0 [Montemerlo y col., 2002] y cuenta con un nuevo método de asociación de datos que es global y mantiene múltiples hipótesis por partícula. Las principales contribuciones de de OV-FastSLAM son:

  - Modelo inverso de la cámara. La cámara omnidireccional utilizada sigue un modelo de proyección no invertible [Bakstein y Pajdla, 2002]. Proponemos una aproximación basada en una tabla de referencia para transformar las coordenadas de los píxeles en la imagen en medidas utilizables por el algoritmo.
  - Modelado de la incertidumbre de las medidas adquiridas por cámaras omnidireccionales. La matriz de covarianza asociada a cada medida, se establece en función de la zona de la imagen en la que haya sido detectada la baliza.
  - Inicialización de las balizas. Presentamos un método retardado de inicialización de las balizas para sensores que solo proporcionan la orientación. El método parte de un grupo de medidas y genera un conjunto de posibles posiciones iniciales, seleccionando la más probable.
  - Asociación de datos jerárquica. Se ha desarrollado un nuevo algoritmo de asociación de datos que es capaz de tratar con toda la complejidad asociada a las cámaras omnidireccionales operando bajo oclusiones severas. El método gestiona múltiples hipótesis por partícula y es global, i. e., la asociación tiene en cuenta todas las medidas y balizas para calcular la probabilidad de cada posible asociación completa, en contraste con los métodos locales que iteran sobre todas las

medidas calculando la probabilidad de asociación de cada una de ellas de forma individual. Además, el método es jerárquico, dividiendo la asociación en dos etapas (balizas y balizas candidatas) para priorizar la asociación de las balizas, pues son más fiables que las balizas candidatas. La asociación de las balizas se basa en el algoritmo de Murty [Murty, 1986], que obtiene las n mejores asignaciones en tiempo polinómico. Por otra parte, la asociación de las balizas candidatas usa el método Húngaro [Kuhn, 1955], que permite generar la mejor asignación.

Las contribuciones de esta tesis doctoral se encuentran recogidas en las siguientes publicaciones:

- Gamallo, C.; Domenech, J. E.; Quintía, P.; Regueiro, C. V.; Correa, J. (2007). *Localización de un robot móvil en interiores mediante visión omnidireccional y balizas naturales.* En: Actas del Workshop de Agentes Físicos (WAF), pp. 119–126.
- Gamallo, C.; Regueiro, C. V.; Mucientes, M. (2008). *Global localization based on omnivision sensor for a guide mobile robot*. En: Actas del Workshop de Agentes Físicos (WAF), pp. 61–68.
- Gamallo, C.; Regueiro, C. V.; Mucientes, M.; Quintía, P. (2008). *Monte Carlo localization for a guide mobile robot in a crowded environment based on omnivision*. En: Proceedings of Towards Autonomous Robotics Systems (TAROS), pp. 1–8.
- Gamallo, C.; Regueiro, C. V.; Mucientes, M.; Quintía, P. (2009). *Localization through omnivision for a tour-guide robot*. Journal of Physical Agents, 2(3), pp. 1–10.
- Gamallo, C.; Mucientes, M.; Regueiro, C. V. (2009). *Visual FastSLAM through Omnivision*. En: Proceedings of Towards Autonomous Robotic Systems (TAROS), pp. 128–135.
- Gamallo, C.; Regueiro, C. V.; Quintía, P.; Mucientes, M. (2010).*Omnivision-based KLD-Monte Carlo Localization*. Robotics and Autonomous Systems, 58(3), pp. 295–305.
- Gamallo, C.; Quintía, P.; Iglesias-Rodríguez, R.; Lorenzo, J. V.; Regueiro, C. V. (2011). *Combination of a low cost GPS with visual localization based on a previous map for outdoor navigation*. En: Proceedings of the IEEE International Conference on Intelligent Systems Design and Applications (ISDA), pp. 1146-1151.

– Gamallo, C.; Mucientes, M.; Regueiro, C. V. (2012). *A FastSLAM Algorithm for Omnivision*. En: Actas del Workshop de Agentes Físicos (WAF), pp. 128–135.

– Gamallo, C.; Mucientes, M.; Regueiro, C. V. (2013). *A FastSLAM-based Algorithm for Omnidirectional Cameras*. Journal of Physical Agents, 7(1), pp. 13-22.

– Canedo-Rodríguez, A.; Álvarez-Santos, V.; Santos-Saavedra, D.; Gamallo, C.; Fernández-Delgado, M.; Iglesias-Rodríguez, R.; Regueiro, C. V. (2013). *Robust multi-sensor system for mobile robot localization*. En: Proceedings of the 5th International Work-Conference on the Interplay between Natural and Artificial Computation (IWINAC). Lecture Notes in Computer Science, 7931(2), pp 92-101.

– Gamallo, C.; Mucientes, M.; Regueiro, C. V. (2013). *Omnidirectional visual SLAM under severe occlusions*. Autonomous Robots, en proceso de revisión.

Estructura de la memoria

Esta memoria se estructura en los siguientes capítulos:

– En el capítulo 1 se presenta el problema de la localización en robótica y se realiza una revisión bibliográfica de los trabajos más relevantes en localización mediante visión.

– A continuación, en el capítulo 2 se realiza una introducción al problema del SLAM y se recogen los trabajos relacionados, centrándose fundamentalmente en aquellos basados en visión.

– En el capítulo 3 se realiza una descripción del sistema de visión artificial y los robots utilizados para resolver tanto el problema de localización como el de SLAM. También se presenta el proceso de detección de balizas realizado. Además, se definen los modelos de medida y de movimiento que serán necesarios para los algoritmos propuestos.

– El capítulo 4 presenta en detalle el algoritmo OV-MCL, como solución para el problema de localización en entornos interiores bajo oclusiones continuas y severas.

– El último capítulo (cap. 5) explica en detalle la solución que se propone para resolver el problema de SLAM con una cámara omnidireccional: el algoritmo OV-FastSLAM.

– La memoria finaliza con una exposición de las conclusiones del trabajo realizado y las posibles líneas de trabajo futuro.

Capítulo 1

Localización en robótica móvil

Los robots móviles autónomos que operan en entornos reales ejecutan habitualmente tareas que requieren el conocimiento de su posición. Dado un mapa del entorno, un sistema de localización debe estimar la pose (posición y ángulo) del robot en el mapa en base a las acciones de control sobre el propio robot y a la información que sus sensores obtienen del entorno. Un algoritmo de localización debe ser fiable, robusto, ejecutable en tiempo real y proporcionar en todo momento la posición más probable del robot con la menor incertidumbre posible.

En este capítulo se hace una breve introducción a la localización de robots móviles en entornos interiores. En primer lugar se realiza una discusión general del problema y se presentan diferentes enfoques para resolverlo. Posteriormente se recoge una revisión del estado del arte de las aproximaciones basadas en visión, poniendo especial énfasis en aquellas que emplean una cámara omnidireccional como sensor principal.

1.1. Descripción del problema de localización

El problema de localización consiste en responder a la pregunta *¿dónde estoy?* desde el punto de vista de un robot [Borenstein y col., 1996]. Es decir, el robot debe encontrar su posición relativa al entorno en el que se mueve. En general, dicha posición se define tanto por el desplazamiento x, y, z como por los ángulos α_x, α_y, α_z respecto a un sistema de referencia arbitrario. En la mayoría de las aplicaciones el robot móvil se desplaza sobre un plano, por lo que el problema de la localización se reduce a encontrar la terna (x, y, α) (fig. 1.1). En esta tesis usaremos indistintamente los términos localización, posición y pose para hacer referencia al resultado proporcionado por el algoritmo de localización.

La localización es clave en robótica móvil [Thrun y col., 2001], ya que si un robot no sabe dónde se encuentra no podrá sacar partido a la información que conozca previamente del entorno, o que vaya descubriendo durante su operación, y no podrá tomar las mejores decisiones en cada momento. Por tanto, será de gran utilidad para el robot conocer, aunque sea de forma aproximada, su posición en el entorno para poder operar y actuar con éxito [Cox, 1991].

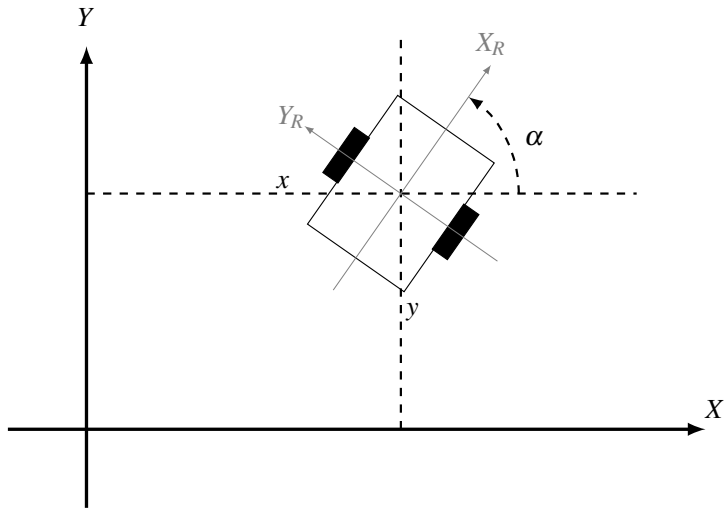


Figura 1.1: Sistemas de referencia asociados a un robot móvil que se mueve en un plano.

La estimación de la posición de un vehículo es un problema que ha recibido una considerable atención por parte de numerosos investigadores, y se han propuesto una gran variedad de técnicas para resolverlo. Estas aproximaciones varían significativamente en función del tipo de entorno en el cual el robot móvil ha de navegar, del conocimiento que se tenga de dicho entorno, de la tarea a realizar y del tipo de sensores con los que el robot está equipado.

A continuación se presenta una clasificación de las diferentes técnicas de localización según el tipo de problema que resuelven, el tipo de sensores que emplean y el modo de representar y almacenar la información sobre el entorno.

1.1.1. Clasificación según el tipo de problema

Existen diferentes enfoques a la hora de tratar el problema de la localización dependiendo del tipo de conocimiento del que se dispone inicialmente y durante el tiempo de ejecución. Se definen según la siguiente taxonomía en orden creciente de complejidad:

- **Localización local.** En este enfoque se asume que la posición inicial del robot es (aproximadamente) conocida. Por lo tanto, la localización del robot se centra en compensar el ruido asociado al propio movimiento del robot. También se denomina *tracking* de movimiento.

- **Localización global.** A diferencia del caso anterior, en la localización global se desconoce la posición inicial. Por lo tanto, es preciso que el robot maneje múltiples hipótesis para determinar su posición inicial, y en consecuencia su posición actual. En general, los errores en las estimaciones son mayores que en la localización local.

- **Secuestro.** El problema del secuestro es una variante del problema de la localización global, y consiste en que el robot se desplace (o sea desplazado) sin que sus sensores lo detecten. Su dificultad es máxima debido a que el robot parte de la creencia de que está en una pose que es completamente distinta a la real. Aunque es una situación poco frecuente, su importancia radica en que refleja la robustez del algoritmo de localización ante fallos importantes.

En esta tesis propondremos una técnica de localización capaz de resolver los tres tipos de problemas.

1.1.2. Clasificación según el tipo de sensor

Los distintos sistemas de localización también se pueden clasificar atendiendo al tipo de sensores que emplean para determinar la pose del robot. La primera división consiste en diferenciar si los sensores miden una magnitud física dentro del propio robot o del entorno en el que se mueve. Así, los sensores para localización pueden ser de dos tipos: internos o externos. Dentro de estos últimos, se distinguen a su vez dos variantes, según sean capaces de medir directa o indirectamente la posición del robot.

- **Sensores internos.** Son aquellos que miden alguna variable interna del robot. Destacan los codificadores (cuentan las vueltas que da una rueda), acelerómetros (miden la

aceleración que se aplica al robot), giróscopos (perciben los cambios de orientación y la velocidad de giro), etc. Integrando esta información a lo largo del tiempo es posible determinar la variación de la pose del robot mientras se mueve en su entorno. Si se conoce la posición inicial, entonces es posible (al menos teóricamente) calcular la pose del robot en todo momento. Las técnicas más conocidas son los sistemas odométricos (odometría) y los sistemas de navegación inercial (INS, del inglés *Inertial Navigation System*). Estos métodos son relativamente fáciles de implementar, pero presentan un grave inconveniente: los errores se van acumulando, y por tanto la incertidumbre crece proporcionalmente al espacio recorrido y al tiempo transcurrido. Por este motivo este tipo de sensores no se suelen emplear en solitario, sino que generalmente complementan algún otro sensor o mecanismo de localización que reduzca periódicamente la incertidumbre en la posición.

– **Sensores externos directos.** Son aquellos que miden directamente la posición del robot en su entorno. Están configurados en base a dos elementos: la unidad montada sobre el vehículo (el receptor), y la unidad o unidades externas que necesitan ser emplazadas en posiciones conocidas del entorno (los emisores). Dentro de este tipo de sistemas, el de mayor interés para robots móviles es el sistema de posicionamiento global (GPS, del inglés *Global Positioning System*), donde los emisores son una red de satélites y el receptor es una antena GPS. Tiene la ventaja de ser global y tener una precisión entre aceptable y buena (según la calidad del receptor GPS). Sin embargo, su uso está limitado a exteriores y con un horizonte relativamente despejado de obstáculos. Para interiores se suelen utilizar los sistemas Wifi o de identificación por radiofrecuencia (RFID, del inglés *Radio Frequency IDentification*), aunque requieren que cada entorno tenga que ser acondicionado de forma apropiada (incluyendo los emisores necesarios) y poseen una precisión y resolución relativamente pobres.

– **Sensores externos indirectos.** Son sensores que también se emplean para otros usos (p. ej. navegación) y con los cuales se puede inferir la pose del robot a través del emparejamiento de la información sensorial con los datos previamente conocidos del entorno (un mapa). Ejemplos de este tipo de sensores son las cámaras, los ultrasonidos, los infrarrojos y los láseres.

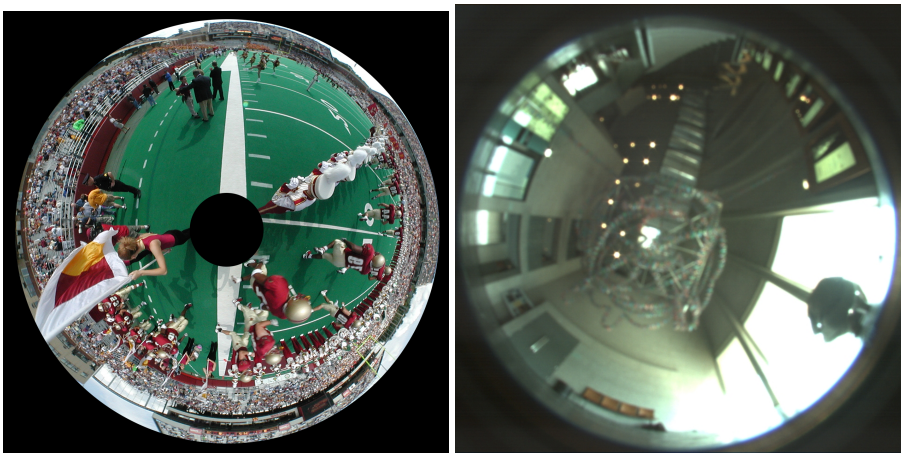
En entornos interiores, la combinación más habitual ha sido sensores internos y externos indirectos. Durante años los sensores más utilizados han sido los ultrasonidos [Leonard y

Durrant-Whyte, 1991; Tardós y col., 2002] y los láseres [Thrun y col., 2000; Fox y col., 2001], sobre todo por su simplicidad de procesado al proporcionar directamente medidas de distancia. La principal ventaja de los ultrasonidos es su relativamente bajo coste, pero por contra son muy vulnerables a reflexiones especulares (*crosstacking* y *multipath*), a las propiedades de reflexión de los materiales del medio y presentan una muy pobre resolución angular. Por otro lado, los láseres aunque presentan mayor precisión, fiabilidad, alcance y resolución angular, tienen un coste y un consumo energético mayores y son más voluminosos.

Las cámaras [Andreasson y col., 2005; Menegatti y col., 2006] siempre han estado presentes en todos los procesos de percepción en robótica móvil. Tienen numerosas ventajas, entre las que destacan la alta resolución angular, la gran velocidad de adquisición de la información, el bajo consumo energético, el coste moderado y el pequeño volumen. Como inconveniente principal hay que destacar que su funcionamiento es dependiente de las condiciones de iluminación y de la textura de las superficies del entorno. También se debe resaltar que con los datos de una única cámara no se puede estimar directamente información de distancia. Con dos o más cámaras previamente calibradas se pueden triangular distancias, pero su campo de visión es limitado y existe un fuerte compromiso entre precisión y alcance. Con el avance de la potencia de cálculo disponible hoy en día a un coste razonable y la aparición de técnicas más complejas, se ha extendido el uso de cámaras como sensor principal para los procesos de localización de robots móviles en interiores.

Sin embargo, las cámaras convencionales tienen un campo de visión (FOV, del inglés *Field Of View*) relativamente pequeño, por lo que detectan una parte muy restringida de su entorno en cada imagen. Para evitar esta limitación se pueden emplear cámaras omnidireccionales que poseen un campo de visión mucho mayor, lo que facilita la localización del robot en entornos muy concurridos e incluso en condiciones severas de oclusión (p. ej. debido a gente muy próxima al robot). Este será el sensor empleado en esta tesis.

Existen dos tipos de configuraciones posibles para visión omnidireccional: los sistemas *catadióptricos* (fig. 1.2(a)) donde las imágenes de la cámara se obtienen a través de un espejo convexo (cónico, esférico, parabólico o hiperbólico), y los sistemas *dióptricos* (fig. 1.2(b)) donde las imágenes se capturan a través de una lente gran angular de tipo ojo de pez (*fish eye*). La mayoría de los trabajos existentes optan por las cámaras catadióptricas. Sin embargo, el uso de espejos para obtener imágenes omnidireccionales tiene dos grandes inconvenientes. En primer lugar, parte de la imagen está ocluida por la propia cámara y/o el soporte del espejo. En segundo lugar, este tipo de cámaras es bastante más voluminoso y su calibración y mantenimiento son más complicados.



(a) Imagen de una cámara catadióptrica. (b) Imagen de una cámara dióptrica.

Figura 1.2: Comparación entre imágenes de una cámara catadióptrica (de espejo) y dióptrica (lente gran angular).

Algunos autores emplean un anillo de cámaras para poder percibir una imagen panorámica completa del entorno (fig. 1.3). Por ejemplo, en [Kaess y Dellaert, 2006] se emplean ocho cámaras y LadyBug integra seis en un único dispositivo. La principal ventaja del anillo de cámaras es utilizar equipamiento estándar y la posibilidad de obtener imágenes de una gran resolución. Otra ventaja teórica, que casi nunca se aprovecha, es que las cámaras del anillo se pueden disponer de modo que no interfieran con el resto de equipos que monta el robot, minimizando así las obstrucciones y oclusiones. Como desventajas, hay que destacar el mayor coste (por el mayor número de cámaras), y sobre todo la mayor complejidad tanto en la calibración del sistema como en la sincronización de las capturas.

1.1.3. Clasificación según el tipo de mapa

Como se ha visto en la sección anterior, empleando sensores externos indirectos para la localización es necesario disponer de un mapa previo del entorno en el que se almacena la información relevante según el tipo de sensor empleado. Esta información se puede estructurar de diversas maneras dando lugar a diferentes tipos de mapas, y por tanto de sistemas de localización.

En general, se define un mapa de un entorno como una lista de N objetos y sus localizaciones $m = m_1, m_2, ..., m_N$ [Thrun y col., 2003]. Se puede establecer la siguiente clasificación:

(a) (b)

Figura 1.3: Ejemplos de anillo de cámaras: a) LadyBug-2 de la compañía PointGrey, b) cámara de cuarta generación empleada en Google Street View.

– **Mapas basados en características.** Consisten en una lista de características (principalmente objetos) del entorno, en las que se almacena para cada una de ellas su posición y sus propiedades. Entre los mapas basados en características se distinguen dos tipos: los métricos y los geométricos [Siciliano y Khatib, 2008].

  - *Mapas métricos.* En los mapas métricos se mantiene una descripción precisa de la posición de los objetos relevantes del entorno utilizando un sistema de coordenadas arbitrario y fijo. Como la información almacenada en los mapas métricos es muy compacta, también son escalables y fáciles de crear.
  - *Mapas geométricos.* En ellos se definen una serie de primitivas de percepción (p. ej. esquinas, segmentos, imágenes, ...) que se relacionan o que se calculan directamente a partir de las medidas sensoriales. Ejemplos de este tipo de mapas son los basados en segmentos rectos.

– **Mapas basados en localizaciones.** Mantienen una lista de regiones específicas del entorno y de cada una se almacena una propiedad. Es decir, se guarda información sobre

el propio espacio y no tanto sobre los objetos del entorno. Dentro de los mapas basados en localizaciones se distinguen los mapas topológicos y los mapas de rejilla.

- *Mapas topológicos.* Están representados por grafos donde los nodos son lugares relevantes del entorno y los arcos son los posibles caminos entre nodos. Por lo tanto, son muy escalables ya que consumen pocos recursos. Permiten planificar trayectorias y determinar posiciones aproximadas, pero no inferir distancias o poses precisas, lo que limita su uso en interiores.
- *Mapas de rejilla.* Dividen el espacio del entorno en un mallado más o menos regular de celdas. En cada celda se almacenan las observaciones sensoriales relacionadas con esa región del espacio asociado. Un ejemplo típico y ampliamente utilizado de mapa de rejilla es el mapa de ocupación. Su principal ventaja es su simplicidad y capacidad para integrar diferentes fuentes de información (varios sensores, incluso de diferente tipo). Su principal inconveniente es que no escalan bien, pues existe un fuerte compromiso entre resolución y tamaño. Aún así existen técnicas para adaptar el tamaño de la celda a las medidas sensoriales asociadas (p. ej. *quadtrees*) y para operar en distintas resoluciones (jerarquía de mapas de rejilla).

Como hemos visto, los mapas basados en características tienen grandes ventajas para su uso en localización a nivel de precisión, fiabilidad y recursos computacionales moderados. Sin embargo, el principal inconveniente es que se deben procesar los datos sensoriales para calcular las primitivas de percepción y después asociar estas con los datos del mapa. Este proceso puede llegar a ser muy complejo en presencia de un alto nivel de ruido e imprecisiones en los sensores.

Dentro de la categoría de mapas métricos los más interesantes son los mapas de balizas, donde una baliza puede ser cualquier objeto característico y reconocible o identificable del entorno. La ventaja de usar balizas es que se simplifica el proceso de percepción y se facilita la detección individual de cada elemento. Por contra, la asociación de datos entre las medidas sensoriales y las balizas del mapa se torna más compleja y ambigua, más aún si todas las balizas son indistinguibles entre sí. Otra dificultad para la localización a partir de mapas de balizas consiste en la necesidad de garantizar un número de balizas suficiente que en todo momento esté libre de oclusiones y dentro del alcance de los sensores del robot. Es en estos dos

aspectos donde proponemos nuevas soluciones en esta tesis para algoritmos de localización basados en mapas de balizas.

1.2. Técnicas de localización

Existen muy diversas alternativas para solucionar el problema de la localización en robótica móvil en general [Borenstein y col., 1996; Gonzalez y Ollero, 1996; Cox, 1990], y de la localización basada en mapas de balizas en particular. La opción más inmediata consiste en emplear técnicas geométricas básicas para determinar la posición del robot a partir de las balizas detectadas [Sugihara, 1988; Krotkov, 1989; Manolakis, 1996; Thomas, 2005]. La posición se calcula por relaciones geométricas, bien a partir de medidas de distancias (trilateración), ángulos (triangulación) o una combinación de ambas. La desventaja de este tipo de técnicas reside en que son muy sensibles al ruido en la detección y a la ambigüedad del proceso de asociación de datos.

Un primer intento para subsanar estos problemas consistió en la aplicación de técnicas de minimización [Drumheller, 1987] para buscar la correspondencia óptima entre las medidas de los sensores y las balizas del mapa. Sin embargo, estas técnicas no son lo suficientemente eficientes para ser usadas de forma continua por un robot móvil, y además son muy sensibles a datos atípicos o muy lejanos de la media (*outliers*).

La familia de métodos que más éxito han tenido en la solución a todos estos problemas son los filtros probabilísticos [Thrun y col., 2003]. Con este tipo de métodos es posible modelar y procesar el ruido y la ambigüedad tanto del movimiento del robot como de las medidas de los sensores y de la asociación de datos. Los algoritmos probabilísticos de localización son variantes del filtro de Bayes, donde la pose del robot se representa por una función de densidad de probabilidad (PDF). La aplicación del filtro de Bayes al problema de localización se conoce como localización de Markov. Su origen es el trabajo de [Simmons, 1995] en el que se utiliza un mapa de rejilla para representar la probabilidad a posteriori. En [Fox y col., 1999b] se presenta en detalle el algoritmo de Markov para la localización.

En función del tipo de representación que se emplee, se pueden dividir los filtros en dos categorías: gaussianos y no paramétricos. En la tabla 1.2 se recoge un resumen de las características y el comportamiento de todos ellos. A continuación se realiza una breve explicación de cada una de las aproximaciones.

Tabla 1.1: Comparación de las principales técnicas de localización basadas en el filtro de Bayes.

Método	Problema	Mapa	PDF	Eficiencia
EKF/UKF	local	balizas	gaussiana	****
MHT	local, global, secuestro	balizas	mezcla gaussianas	***
GRID	local, global, secuestro	rejilla	histograma	*
Monte Carlo	local, global, secuestro	balizas/rejilla	partículas	**

1.2.1. Algoritmos basados en filtros gaussianos

Los filtros gaussianos constituyen una implementación del filtro de Bayes para espacios continuos en donde las distribuciones de probabilidad se representan por medio de distribuciones normales multivariable. Se pueden distinguir tres tipos principales: filtro de Kalman extendido (EKF, del inglés *Extended Kalman Filter*), filtro de Kalman *unscented* (UKF, del inglés *Unscented Kalman Filter*) y el seguimiento multi-hipótesis (MHT, del inglés *Multi-Hypothesis Tracking*).

– **Localización EKF.** El algoritmo de localización basado en EKF es una técnica muy popular para la solución del problema de localización local. En él se representa la pose del robot mediante una gaussiana con media μ_t y covarianza Σ_t. La pose del robot se estima en dos etapas: predicción y actualización. En la primera de ellas se calcula una predicción de la nueva pose aplicando el último control sobre el robot. En la segunda etapa se calculan los nuevos valores de la pose del robot incorporando la actualización propuesta por las medidas de los sensores.

  Los primeros trabajos de aplicación del EKF en localización requerían modificar el entorno situando balizas artificiales. Por ejemplo en [Leonard y Durrant-Whyte, 1991] se empleaban balizas geométricas para su detección con un sensor de ultrasonidos y en [Salichs y col., 1999] se usaban círculos negros sobre las paredes para ser detectados mediante visión. Una de las primeras aproximaciones de localización EKF en las que no se modificó el entorno fue presentada en [Cox, 1991] y se basaba en las medidas de distancia de un sensor de infrarrojo.

– **Localización UKF.** La localización UKF se basa en un filtro de Kalman que emplea la transformación *unscented* para linealizar los modelos de movimiento y medida, en lugar de hacerlo mediante la expansión en serie de Taylor utilizada en el EKF. En vez de calcular las matrices Jacobianas de los modelos, la transformación *unscented* representa

las gaussianas por puntos *sigma* que son pasados a los modelos iterativamente. En general, los puntos *sigma* se distribuyen simétricamente a lo largo de los ejes principales de la covarianza.

En la bibliografía existen pocos trabajos en los que se opte por esta aproximación para estimar la pose de robot. En [Li, 2003] se utiliza el algoritmo UKF para fusionar medidas de diferentes sensores para estimar la pose y en [Martinelli, 2008] se presenta una comparación entre una localización EKF y otra UKF empleando sensores RFID y láser.

– **Localización MHT.** En el seguimiento multi-hipótesis se representa la creencia mediante múltiples gaussianas, por lo que la probabilidad a posteriori se estima como una mezcla de gaussianas. Cada componente de la mezcla representa una secuencia de asociación de datos diferente, lo que permite mantener múltiples asociaciones a la vez en lugar de escoger únicamente la asociación más probable. El principal inconveniente del algoritmo MHT es que el número de gaussianas crece exponencialmente a lo largo del tiempo. Por ello es necesario un proceso de poda que elimine las gaussianas con una relevancia menor a un umbral, y que mantenga controlado el número de componentes que forman la mezcla de gaussianas. Además, MHT es capaz de resolver tanto la localización global como el problema del secuestro. En el primer caso, inicializando la creencia en la pose (inicial) del robot mediante múltiples hipótesis gaussianas, y en el segundo, inyectando hipótesis adicionales a la mezcla de gaussianas.

  Los primeros trabajos que presentaron la utilización de múltiples gaussianas para estimar la probabilidad fueron los de [Austin y Jensfelt, 2000] y [Jensfelt y Kristensen, 2001]. En [Fox y col., 2000] y [Howard y Sukhatme, 2003] se muestra la aplicación de la localización MHT para múltiples robots.

La diferencia entre la localización EKF y UKF es, en general, pequeña, y solamente cuando la incertidumbre aumenta o las no linealidades de los modelos son significativas, se aprecia un mejor rendimiento de la aproximación con UKF. Una limitación importante es que ambas técnicas de localización solo pueden ser aplicables al problema de localización local. En general todas las técnicas basadas en la linealización de los modelos de movimiento y medida funcionan bien si las incertidumbres son pequeñas y si la aproximación lineal realizada es buena. En caso de un aumento considerable de la incertidumbre se incrementan las posibilidades de que se produzcan asociaciones erróneas entre las medidas y los datos del mapa, lo que provocaría fallos en la estimación de la posición. MHT supera en parte estas limitaciones.

Otra desventaja de las aproximaciones EKF y UKF es que no tienen capacidad para procesar toda la información [Thrun y col., 2003] obtenida del sensor, ya que no tienen en cuenta la información negativa. La información negativa se corresponde con la ausencia de características cuando se esperaría que fuesen observadas por el sensor. MHT sí puede incorporar esta información negativa. A pesar de este conjunto de limitaciones, las tres técnicas son sorprendentemente robustas en su aplicación a la localización de robots móviles.

1.2.2. Algoritmos basados en filtros no paramétricos

Los filtros no paramétricos aproximan la PDF mediante un número finito de valores, cada uno de los cuales se corresponde con una región del espacio de entrada. Las principales ventajas de las técnicas basadas en filtros no paramétricos respecto a las técnicas gaussianas son dos: i) es posible trabajar con PDF multimodales (y por tanto resolver la localización global y el secuestro), frente a la unimodalidad de las distribuciones gaussianas (la mezcla de gaussianas sí es multimodal, aunque esa representación es mucho menos eficiente); ii) no es necesario linealizar los modelos de movimiento y medida. La mayor desventaja de estas técnicas es que las necesidades de almacenamiento de información y el coste computacional crece exponencialmente con el tamaño y la precisión del espacio de búsqueda (la pose del robot). Dentro de estas técnicas destacan la localización Grid y de Monte Carlo.

– **Localización Grid.** Esta aproximaxión usa un filtro basado en histogramas para representar la creencia a posteriori sobre una descomposición de tipo rejilla (*grid*) del espacio de estados (pose del robot). La creencia a posteriori es una colección de valores de probabilidad discretos, uno por cada celda de la rejilla. La resolución de la rejilla es un parámetro fundamental en esta aproximación, ya que representa el compromiso entre precisión y coste computacional y de almacenamiento.

  El primer trabajo de localización Grid fue realizado por Simmons y Koening [Simmons, 1995], y posteriormente ha sido utilizado por otros autores [Cassandra y Kaelbling, 1996; Hertzberg, 1996]. En [Burgard y col., 1999] se introduce una técnica de actualización selectiva de la rejilla, lo que permite trabajar con rejillas de mayor resolución.

– **Localización de Monte Carlo (MCL).** Es uno de los algoritmos de localización más utilizados. En él se representa la creencia o PDF de la pose del robot por un conjunto de M partículas, donde cada partícula es una muestra de la PDF y codifica una posible pose del robot. Las partículas se distribuyen de acuerdo con la PDF, i.e., la región de

la PDF con una probabilidad más alta tiene una mayor concentración de partículas. La precisión del filtro es tanto mayor cuanto más alto sea el número de partículas (a costa de aumentar la carga computacional).

Fox [Fox y col., 1999a] y Dellaert [Dellaert y col., 1999b] fueron los primeros en desarrollar un filtro de partículas para la localización de un robot móvil. En [Fox y col., 1999a] se incluyó además la idea de añadir partículas aleatorias, lo que permite al sistema recuperarse de fallos importantes en la localización. Siguiendo la misma idea, en [Lenser y Veloso, 2000] se presenta una solución al problema del secuestro basada en reiniciar el conjunto de partículas de acuerdo con la última medida del sensor.

Por otra parte, en [Gutmann y Fox, 2002] se introduce una modificación conocida como *Augmented MCL*, que permite determinar el número de partículas aleatorias que han de introducirse en el sistema para recuperarse ante un fallo. Diversos autores ha propuesto modificaciones al algoritmo básico con el fin de mejorar el rendimiento del mismo. Así, [Fox, 2003] introduce el KLD-sampling, que adapta el número de partículas según la calidad de la distribución en cada instante. En [Kwok y Fox, 2004] se presenta una implementación en tiempo real del algoritmo de Monte Carlo sin que sea necesario disminuir el número de partículas, permitiendo integrar varias medidas en una sola fase de actualización.

1.3. Trabajo relacionado

La localización mediante cámaras se ha abordado en numerosos trabajos que pueden ser clasificados atendiendo al tipo de cámara, de mapa, de entorno o del algoritmo utilizado (tabla 1.2). A lo largo de esta sección se comentarán los trabajos más representativos agrupados según el tipo del sensor utilizado (visión estéreo o monocular), haciendo énfasis en visión monocular, y especialmente en aquellos trabajos que usan cámaras omnidireccionales, pues son los más similares a la propuesta planteada en esta memoria.

1.3.1. Localización con cámara estéreo

Un tipo de cámara usada muy frecuentemente en localización es el par estéreo, que consiste en el empleo de dos cámaras correctamente calibradas que permiten obtener, además de las imágenes, la profundidad de las características detectadas. Así, en [Wolf y col., 2005] se aplica localización de Monte Carlo mediante visión estéreo. Además, también usan un método de

Tabla 1.2: Resumen de las principales aproximaciones de localización mediante visión.

VISIÓN MONOCULAR				
Basados en la apariencia global de la imagen				
Referencia	**Cámara**	**Mapa**	**Entorno**	**Algoritmo**
[Dellaert y col., 1999a]	Estándar	Mosaico (brillo)	Interior	Monte Carlo
[Thrun y col., 2000]	Estándar	Mosaico (brillo)	Interior	Monte Carlo
Basados en características locales de la imagen				
Referencia	**Cámara**	**Mapa**	**Entorno**	**Algoritmo**
[Atiya y Hager, 1993]	Estándar	Métrico (líneas)	Interior	Geométrico
[Chenavier y Crowley, 1992]	Estándar	Métrico (líneas)	Interior	Kalman
[Rofer y Jungel, 2003]	Estándar	Métrico (líneas)	Interior	Monte Carlo
[Moreira y col., 2007]	Estándar	Métrico (SIFT)	Exterior	Monte Carlo
VISIÓN OMNIDIRECCIONAL				
Basados en la apariencia global de la imagen				
Referencia	**Cámara**	**Mapa**	**Entorno**	**Algoritmo**
[Andreasson y col., 2005]	Omni/espejo	BD imágenes (SIFT)	Interior	Monte Carlo
[Gross y col., 2002]	Omni/ espejo	Topológico (color)	Interior	Monte Carlo
[Kröse y col., 2001]	Omni/ espejo	BD imágenes	Interior	Markov
[Jogan y Leonardis, 2003]	Omni/ espejo	BD imágenes	Interior	Minimización
[Menegatti y col., 2004]	Omni/ espejo	BD imágenes	Interior	Monte Carlo
Basados en características locales de la imagen				
Referencia	**Cámara**	**Mapa**	**Entorno**	**Algoritmo**
[Cao y col., 1986]	Omni/espejo	Métrico (balizas)	Interior	Geométrico
[Brassart y col., 2000]	Omni/ espejo	Métrico (líneas)	Interior	Minimización
[Shimizuhira y Maeda, 2003]	Estéreo (omni)	Métrico	Interior	Geométrico
[Sun y col., 2004]	Omni/ espejo	Métrico (líneas)	Interior	Geométrico
[Murillo y col., 2006]	Omni/ lente	Topológico/métrico (líneas)	Interior	Monte Carlo
[Cao y col., 2007]	Omni/ lente	Métrico	Interior	Geométrico
[Wu y Tsai, 2009]	Omni/ espejo	Métrico (círculos)	Interior	Minimización
[Ramalingam y col., 2009]	Omni/ lente	Métrico (líneas)	Exterior	Matching
[Philip y Ho, 2011]	Omni/espejo	Métrico (líneas)	Interior	Minimizacion

comparación de las imágenes actuales con las de una base de datos construida previamente, pero en este caso las marcas se basan en histogramas de características locales. En [Konolige y col., 2011] también se emplea un sistema estéreo, pero en este caso en entornos exteriores. La aproximación se basa en la estimación de la odometría visual, lo que permite seguir la pose del robot durante la trayectoria.

A pesar de la ventaja de obtener distancias asociadas a las imágenes, estos sensores presentan varios inconvenientes. En primer lugar, la limitación del FOV en el que se puede obtener

la distancia. En segundo lugar, el compromiso entre alcance y precisión. Además, se han de implementar algoritmos de asociación adicionales para poder establecer la correspondencia entre las dos imágenes, sin contar con que el proceso de calibración es más complejo. Por último, su coste es bastante superior al de una cámara convencional.

1.3.2. Localización con cámara estándar

Las principales aproximaciones de localización con visión se pueden clasificar en basadas en características y basadas en apariencia (píxeles), tal y como se recoge en la tabla 1.2. Como ya se ha comentado, las técnicas basadas en características [Sun y col., 2004; Krotkov, 1989] explotan las propiedades del entorno o cualquier objeto reconocible del mismo (balizas). Las aproximaciones basadas en apariencia [Andreasson y col., 2005; Thrun y col., 2000] computan la correlación entre imágenes para estimar la pose del robot. Para ello se necesita almacenar información previa sobre el entorno, utilizando normalmente una base de datos de imágenes con la que posteriormente se comparan las imágenes adquiridas para determinar la pose del robot.

La localización basada en apariencia tiene varias desventajas. En primer lugar, las imágenes son grabadas para una ruta, y si la ruta cambia o el robot tiene orientaciones muy diferentes en una posición, el error de la localización aumentará. En segundo lugar, el error de localización se incrementa enormemente en caso de oclusiones parciales de las imágenes, por ejemplo por la presencia de gente u objetos en movimiento. Finalmente, esta clase de algoritmos es muy sensible a posibles modificaciones en el entorno, cambios en la iluminación, etc.

Las primeras aproximaciones mediante visión monocular que se recogen en la bibliografía se basan en métodos clásicos, como la triangulación, para el cálculo de la posición del robot. Así, en [Sugihara, 1988] se establece la posición del robot geométricamente basándose en la correspondencia entre líneas de la imagen y un conjunto de postes verticales del entorno. En [Krotkov, 1989] se presenta una extensión del trabajo anterior, en el que se usan también las líneas verticales del entorno detectadas en las imágenes como referencias para establecer la posición, pero añade una componente de incertidumbre para tener en cuenta el ruido en las medidas.

Otra aproximación que sigue este mismo enfoque se presenta en [Atiya y Hager, 1993]. Además de la estimación de la pose basada en triangulación se emplea la teoría de conjuntos para seleccionar la posición consistente con todas las características (también líneas vertica-

les). Además se le añade un mecanismo a la cámara que permite deslizarla horizontalmente, lo que permite estimar la posición utilizando una sola baliza.

La mayoría de los algoritmos de localización con cámaras se basan en aproximaciones probabilísticas. En [Chenavier y Crowley, 1992] presentan una de las primeras implementaciones del filtro de Kalman para localización empleando visión. Se parte de una base de datos de balizas, que se corresponden con las líneas verticales del entorno (puertas, tuberías, esquinas, etc). En [Dellaert y col., 1999a] se describe una aproximación basada en Monte Carlo. El robot está equipado con una cámara monocular apuntando al techo y el mapa se construye como un mosaico de imágenes capturadas y alineadas globalmente. La pose del robot se estima usando la información del brillo de la imagen y se muestra su funcionamiento tanto en localización local como en localización global.

Una aproximación similar se sigue en [Thrun y col., 2000], donde se utiliza un mapa formado por imágenes del techo. La pose se estima en función de la medida del brillo de la imagen capturada por el sensor. Este sistema es sensible a oscilaciones de la cámara, y como consecuencia del pequeño FOV de la cámara, en algunos instantes no se observa ninguna de las luces. En [Rofer y Jungel, 2003] también se presenta un sistema de localización global tipo Monte Carlo, pero basado en las características del campo de la RoboCup: marcas de localización, porterías y líneas del campo.

Otro enfoque basado en características se presenta en [Moreira y col., 2007], pero en vez de usar características concretas reconocibles del entorno se emplean características SIFT extraídas de las imágenes. Finalmente en [Bangash y Ghafoor, 2011] se presenta una aproximación basada en balizas que determina la posición en función del cambio en la forma y el tamaño de los objetos en base a reglas trigonométricas.

1.3.3. Localización con cámara omnidireccional

Un tipo de cámaras particularmente interesante en localización son las omnidireccionales, puesto que cuentan con un amplio FOV ($>> 90°$). El primer trabajo que combina el uso de omnivisión y localización fue publicado en [Cao y col., 1986], aunque su uso no se ha extendido hasta hace pocos años.

En [Yagi y col., 1995] se emplea un espejo cónico para obtener las imágenes omnidireccionales. El sistema de localización que describe sigue un enfoque clásico en la línea de los presentados en [Sugihara, 1988] y [Krotkov, 1989], usando también las líneas verticales de los objetos del entorno como referencia. Para determinar la posición se realiza un ajuste por

mínimos cuadrados entre el azimut estimado a partir de las imágenes y el calculado a partir del mapa previo del entorno.

Un enfoque parecido se propone en [Brassart y col., 2000], que además simula una visión estereoscópica mediante el movimiento de la cámara sobre un raíl, lo que permite adquirir dos imágenes desde posiciones separadas $40\,cm$. De esta forma es posible obtener la distancia de cada línea vertical detectada (líneas, esquinas, puertas, etc.) al haber realizado previamente un proceso de calibración. Para asociar los puntos obtenidos de la imagen con los puntos almacenados en el modelo, los autores se basan en la distancia euclídea. La posición del robot se estima mediante un proceso que minimiza la distancia euclídea de todos los puntos detectados respecto a los del modelo.

En [Wu y Tsai, 2009] también se muestra una aproximación basada en características y en el uso de un espejo para obtener la imagen omnidireccional. Al contrario que el resto de trabajos presentados, en este caso se emplean balizas circulares artificiales que se sitúan en el techo del entorno y se orienta la cámara hacia el techo. La pose del robot se estima en base a los parámetros de las elipses que caracterizan las balizas en las imágenes. Un enfoque similar con la cámara apuntando hacia el techo se recoge en [Philip y Ho, 2011]. El trabajo se desarrolló en entornos exteriores urbanos y se estima la pose del robot comparando las proyecciones de las líneas horizontales segmentadas de la imagen del cielo con un mapa 2D generado a partir de un conjunto de imágenes aéreas.

Varias aproximaciones de omnivisión se basan en el algoritmo Monte Carlo usando diferentes técnicas para la estimación de los pesos de las partículas. Por ejemplo, en [Andreasson y col., 2005] el modelo de medida usa la correlación de la medida actual con un conjunto de imágenes que han sido grabadas previamente para una ruta específica. La comparación se lleva a cabo con una versión modificada del algoritmo SIFT. El sistema es capaz de funcionar con oclusiones parciales, pero incrementa su error en la localización. Otra aproximación es la presentada en [Menegatti y col., 2006], en la que se usan las distancias entre las transiciones de color más próximas de las imágenes para estimar la probabilidad de una medida. Para ello es necesario un mapa geométrico del entorno en el que se incluyan las transiciones cromáticas, en este caso del suelo.

En [Gross y col., 2002] se presenta otro algoritmo de localización basado en Monte Carlo donde se emplea el color del entorno para evaluar las diferentes partículas. Aunque en este caso se opta por usar la media de color para cada sección circular de las imágenes panorámicas rectificadas a partir de la imagen original (circular). El mapa, en lugar de ser geométrico es topológico, y cada nodo almacena el vector de medias de color para cada posición. El uso

de las imágenes panorámicas en lugar de las imágenes circulares obtenidas directamente de la cámara es una práctica muy común en las aproximaciones que emplean visión omnidireccional. Otro trabajo que utiliza esta misma estrategia es [Palettat y Simone, 2001], donde se recupera y utiliza el contexto visual, y mediante un razonamiento Bayesiano se estima la pose del robot comparando la información extraída de las imágenes con las múltiples vistas 3D de los objetos que forman el modelo o mapa del entorno.

En [Kröse y col., 2001] se propone usar la técnica de análisis de componentes principales (PCA) para representar el modelo del entorno. Así, en lugar de almacenar un conjunto de imágenes clave, las imágenes panorámicas son representadas por sus autovectores. La pose del robot se estima empleando un algoritmo de localización de Markov. Un esquema similar es seguido por [Jogan y Leonardis, 2003], donde la localización del robot se plantea como un proceso de reconocimiento de las vistas panorámicas en base a un modelo del entorno construido previamente a través de un conjunto de autovectores para cada imagen y sus poses asociadas. El sistema incluye un método de compresión que permite representar en un solo conjunto de autovectores todas las imágenes asociadas a una misma posición pero tomadas con distintas orientaciones del robot.

Otro trabajo que también utiliza un modelo basado en la apariencia de las imágenes panorámicas se recoge en [Menegatti y col., 2004]. Se utiliza el algoritmo de Monte Carlo para resolver la localización. Las imágenes de referencia se almacenan de una forma compacta usando los coeficientes de Fourier, lo que permite reducir las necesidades de almacenamiento y agilizar el proceso de correspondencia entre las imágenes de entrada y el modelo.

En [Sun y col., 2004] se presenta la aplicación de un sistema de localización sobre un entorno dinámico, el campo de la *RoboCup Middle Size League*. La aproximación usa las imágenes panorámicas para facilitar la extracción de las balizas del entorno (porterías y postes). La posición del robot se estima aplicando relaciones geométricas similares al método clásico de triangulación basado en balizas.

En [Murillo y col., 2006] los autores proponen un algoritmo de localización jerárquico que combina información topológica y métrica a partir de las imágenes de una cámara con una lente gran angular. La localización métrica se genera a partir de líneas verticales del entorno detectadas en las imágenes. Usando este mismo tipo de lente, en [Cao y col., 2007] eliminan la distorsión de las imágenes mediante un proceso de rectificación que reduce el ángulo de visión a 140°. La localización se resuelve de manera geométrica y el proceso de seguimiento de las balizas en las imágenes se realiza mediante un filtro de partículas. Al menos dos balizas han de ser vistas para obtener una posición.

En [Ramalingam y col., 2009] se presenta una aproximación para resolver la localización global en exteriores en base a modelos 3D de las ciudades mediante una cámara omnidireccional orientada hacia arriba. La propuesta estima la posición emparejando las imágenes del cielo con las líneas del horizonte de los edificios de un modelo 3D simplificado del entorno.

Finalmente, en [Shimizuhira y Maeda, 2003] se combinan tres cámaras omnidireccionales situadas cada una en el vértice de un triángulo equilátero para estimar la pose del robot en el campo de la RoboCup. Se obtiene la posición de los objetos en base al sistema de coordenadas del robot empleando las mismas técnicas que los sistemas estéreo. Mediante trilateración se obtiene la posición del robot en coordenadas absolutas a partir de las posiciones mapeadas de los postes de las esquinas.

Capítulo 2

Localización y mapeado simultáneos

Mapear un entorno requiere una correcta localización del robot pero, a su vez, obtener una pose (posición y orientación) precisa requiere la existencia de un mapa. Por tanto, el mapeado y la localización son problemas que deben resolverse de forma simultánea. La localización y el mapeado simultáneos (SLAM), consiste en construir un mapa de un entorno desconocido a la vez que el robot se mantiene localizado.

En los últimos años se ha dedicado un gran esfuerzo a resolver este problema con diferentes tipos de técnicas y sensores. Inicialmente, la mayoría de las propuestas trabajaba en entornos 2D y usando sensores de distancia como láser [Thrun y col., 1998; Dissanayake y col., 2001; Castellanos y Tardós, 1999] o ultrasonidos [Leonard y Durrant-White, 1991; Tardós y col., 2002]. Más recientemente, se ha extendido el uso de cámaras por sus ventajas (bajo coste, ligereza, consumo reducido y mayor información sobre el entorno), dando lugar al campo del SLAM visual.

Actualmente el SLAM es todavía un problema abierto del cual quedan por resolver importantes retos. Uno de los más destacados es operar en entornos donde se produzcan oclusiones frecuentes y severas, por ejemplo debido a la presencia de numerosas personas moviéndose alrededor del robot e interfiriendo en las medidas de los sensores. En este capítulo se presenta en primer lugar una introducción al SLAM, a continuación se describen las principales técnicas utilizadas, y por último se hace un repaso de los trabajos relacionados.

2.1. Descripción del problema de SLAM

En el escenario típico del problema de SLAM, un robot móvil se desplaza por un entorno desconocido ejecutando una secuencia de controles $u_{1:t} = \{u_1, \ldots, u_t\}$ y obteniendo observaciones del mundo $z_{1:t} = \{z_1, \ldots, z_t\}$. Tanto el control como las observaciones están típicamente afectadas por ruido. El objetivo final es estimar la pose del robot a lo largo de la trayectoria $x_{1:t} = \{x_1, \ldots, x_t\}$ y el mapa m del entorno. A lo largo de esta memoria se trabajará con un mapa de balizas (sec. 1.1.3). Si se limita el movimiento del robot a un plano, entonces la pose del robot x_t se define como una terna (x, y, α). Sin embargo, como la cámara percibe el mundo en tres dimensiones, el mapa de balizas se define como un vector de N posiciones en 3D, $m = \{B_1, \ldots, B_{N_t}\}$.

Los controles u_t almacenan información relativa al movimiento realizado por el robot. Pueden ser medidos usando diferentes sensores, aunque generalmente se basan en la información proporcionada por los codificadores incorporados en las ruedas del robot (posición odométrica). Las observaciones z_t proporcionan medidas de las posiciones de las balizas en el entorno[1]. La correspondencia entre las balizas y las observaciones está definida por la función de correspondencia o asociación de datos que proporciona como resultado la matriz de ambigüedad Ψ_t.

Desde un punto de vista probabilístico existen dos maneras de formular el problema del SLAM: en línea y completo.

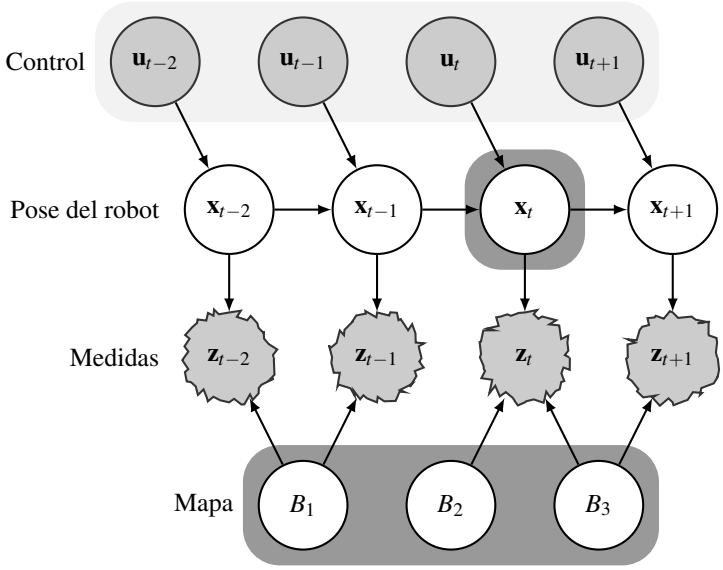
SLAM en línea: el objetivo (fig. 2.1(a)) es calcular la probabilidad en el instante actual de la pose y el mapa, $p(x_t, m, \Psi_t | z_{1:t}, u_{1:t})$.

SLAM completo: se busca (fig. 2.1(b)) estimar la probabilidad de la trayectoria completa y el mapa, $p(x_{1:t}, m, \Psi_{1:t} | z_{1:t}, u_{1:t})$.

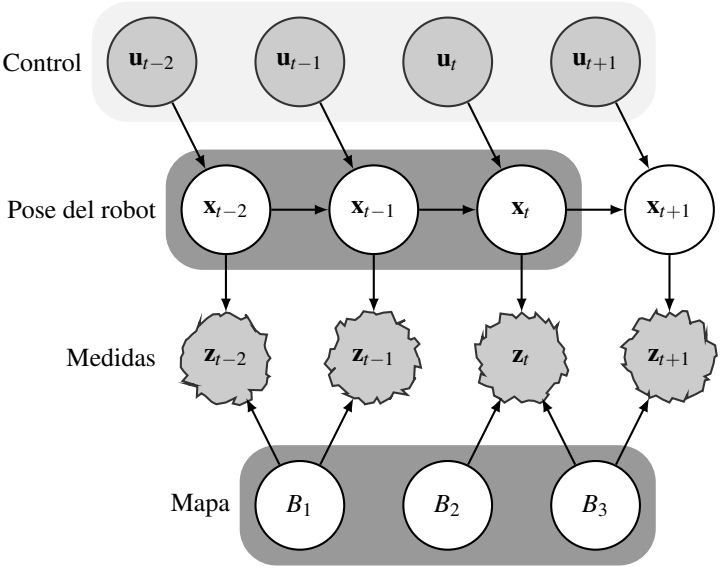
El SLAM posee una componente continua: la estimación de la pose del robot x_t y la localización de las balizas del mapa m, y también una componente discreta: estimar la correspondencia Ψ_t entre las medidas del sensor z_t y las balizas del mapa B_j (asociación de datos, sección 2.1.1).

La estimación de la pose del robot en un instante t, x_t, se puede representar como una función probabilística condicionada a la pose en el instante de tiempo anterior, x_{t-1}, y al

[1] En este capítulo se asume que en cada instante de tiempo se obtiene una única medida. En la sección 5.3 se mostrará el caso general de varias observaciones en cada instante de tiempo.



(a) SLAM en línea.



(b) SLAM completo.

Figura 2.1: Descripción gráfica de las dos formulaciones del problema de SLAM mediante redes Bayesianas dinámicas, donde los nodos representan las variables del sistema y los arcos indican las dependencias entre ellas. El sombreado oscuro representa la información que se estima en el instante t.

control ejecutado por el robot en ese instante, u_t. A esta función se la denomina **modelo de movimiento**, $p(x_t|x_{t-1},u_t)$, porque describe cómo los controles afectan al cambio en la pose del robot y modela el ruido en los controles para su tratamiento probabilístico.

Las observaciones del sensor también se rigen por una función probabilística conocida como **modelo de medida**, $p(z_t|x_t,m,\Psi_t)$. El modelo de medida permite estimar la probabilidad de una medida como función de la pose del robot y del mapa. Para ello se utiliza el modelo matemático que convierte las posiciones 3D de las balizas en medidas, así como el modelo de ruido del sensor.

Usando el modelo de medida y el modelo de movimiento, la probabilidad del SLAM puede ser calculada de modo recursivo como la función de densidad de probabilidad (PDF) en el instante $t-1$ [Montemerlo y Thrun, 2007]:

$$p\left(x_t,m,\Psi_t|z_{1:t},u_{1:t}\right)=\eta\, p(z_t|x_t,m,\Psi_t)\int p(x_t|x_{t-1},u_t)\sum_{\Psi_{t-1}} p\left(x_{t-1},m,\Psi_{t-1}|z_{1:\mathrm{t}-1},u_{1:\mathrm{t}-1}\right)dx_{t-1} \tag{2.1}$$

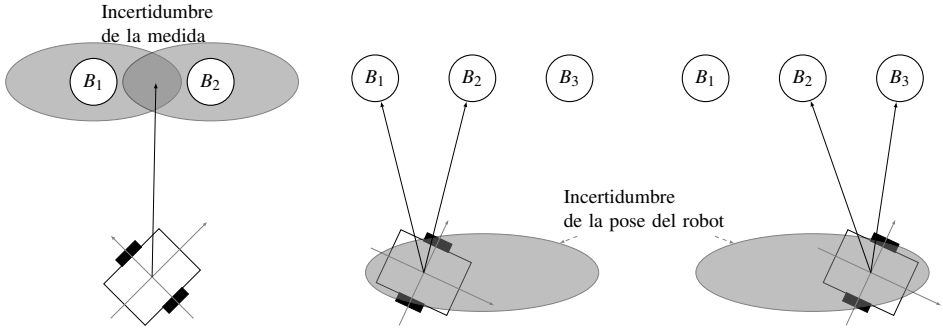
Esta actualización recursiva se conoce como filtro de Bayes para SLAM. En general la integral de la ecuación recursiva no puede ser calculada de forma cerrada debido a la alta dimensionalidad del espacio de estados y al elevado número de variables de la asociación de datos [Thrun y col., 2003]. Por ello, los diferentes algoritmos de SLAM existentes recurren a aproximaciones de dicha PDF.

2.1.1. Asociación de datos en SLAM

La asociación de datos determina la matriz de ambigüedad Ψ_t que define la correspondencia entre las medidas z_t y las balizas del mapa $m=\{B_1,\ldots,B_{N_t}\}$. La solución clásica al problema es utilizar el algoritmo de asociación de máxima probabilidad (ML, del inglés *Maximum Likelihood*) que asigna cada medida a la baliza que maximiza la probabilidad de asociación (sec. 4.1.1).

La asociación de datos está fuertemente influida por la incertidumbre existente en SLAM, que tiene como fuentes los ruidos asociados a los modelos de medida y movimiento. Si estos ruidos son suficientemente altos pueden provocar situaciones de ambigüedad en la asociación de datos (fig. 2.2):

1. Si el ruido en la medida es alto, entonces la incertidumbre asociada es grande y los valores medidos están más dispersos respecto a sus valores reales. Si dos balizas están muy próximas entre sí, entonces sus respectivas distribuciones de incertidumbre pueden



(a) Ruido en la medida. (b) Ruido en el movimiento.

Figura 2.2: Ambigüedad en la asociación de datos de SLAM. Las balizas se representan por triángulos y las medidas por flechas. Las elipses indican las incertidumbres en los valores.

solaparse sustancialmente (fig. 2.2(a)) y provocar ambigüedad en la asociación de datos. El resultado es que se puede atribuir una medida a una baliza incorrecta y por lo tanto se incrementa el error tanto en el mapa como en la pose del robot.

2. Si la ambigüedad está causada por un elevado ruido en el modelo de movimiento puede acarrear severas consecuencias sobre el sistema. Cuanto más alto sea el ruido del modelo de movimiento mayor será la incertidumbre en la pose estimada a partir del control. Un mayor error en la pose del robot conlleva variaciones drásticas sobre las hipótesis para las observaciones siguientes (fig. 2.2(b)), y por tanto implica una mayor probabilidad de realizar una incorrecta asociación de datos. Esto, a su vez puede provocar un aumento en la probabilidad de que la asociación de datos para observaciones posteriores también sea incorrecta. Todo este proceso daría lugar a un fallo irrecuperable en el sistema, sobre todo si la implementación se basa en una aproximación que solo mantiene una hipótesis en la asociación de datos, lo que hace al sistema mucho más vulnerable.

En el caso del algoritmo de SLAM desarrollado en esta tesis, las pruebas se han llevado a cabo en entornos con un suelo bastante irregular. El terreno desigual provoca que el robot vibre mientas se mueve, por lo que las medidas son mucho más ruidosas. A su vez, el cálculo de la posición odométrica se hace menos preciso, ya que se viola la hipótesis de que las ruedas se desplazan sobre un plano. Sin embargo hemos observado que una incorrecta calibración de la

Tabla 2.1: Características de los principales algoritmos de SLAM, donde n es el número de balizas del mapa, p el número de poses del robot y M el número de partículas.

Método	Tipo de SLAM	Coste computacional	Memoria
EKF-SLAM	En línea	$O(n^2)$	$O(n^2)$
Graph-SLAM	Completo	$O(n^2)$	$O(n+p)+O(p^2)$
SEIF-SLAM	En línea	$O(n^2)$	$O(n)$
FastSLAM	En línea/Completo	$O(Mn)$	$O(Mn)$

odometría del robot tiene en realidad un mayor impacto en el incremento en la incertidumbre del modelo de movimiento que un terreno desigual.

Por otra parte, una limitación del uso de cámaras omnidireccionales en los sistemas de SLAM es que conllevan un modelo de proyección más complejo. Estas imágenes poseen una importante deformación y una escala variable debido al cambio de resolución a lo largo de la dirección radial. De esta forma las condiciones locales en la imagen pueden cambiar drásticamente, especialmente para las características cercanas al eje óptico. Así, pequeñas traslaciones y rotaciones causan importantes cambios en las imágenes. Todo ello provoca que el proceso de asociación de datos sea aún más complejo, siendo más probables los emparejamientos incorrectos. En las cámaras estándar estas asignaciones pueden ser descartadas comprobando las restricciones epipolares, mientras que con las cámaras omnidireccionales estas restricciones son más difíciles de establecer.

2.2. Clasificación de los algoritmos de SLAM

El problema de la localización y el mapeado simultáneos ha sido abordado con multitud de técnicas [Thrun y col., 2003; Bailey y Durrant-Whyte, 2006; Durrant-Whyte, 2006; Piniés, 2009]. A lo largo de esta sección se presentará una breve descripción de las cuatro principales aproximaciones [Thrun y col., 2003]: EKF-SLAM, Graph-SLAM, SEIF-SLAM y FastSLAM (en la tabla 2.1 se resumen las características más relevantes de cada método).

2.2.1. SLAM con filtro de Kalman extendido (EKF-SLAM)

Las primeras soluciones, y también las más ampliamente utilizadas, se basan en el uso del filtro del Kalman extendido [Smith y Cheeseman, 1986]. El EKF representa la probabilidad del estado como una gaussiana multivariable de altas dimensiones, parametrizada por la media μ_t y la matriz de covarianza Σ_t. La media describe el estado que tiene la pose del robot y el

mapa del entorno más probable, y la matriz de covarianza codifica las correlaciones entre los pares de variables que conforman el vector de estados:

$$p(x_t, m, \Psi_t | z_{1:t}, u_{1:t}) = N(\mu_t, \Sigma_t), \tag{2.2}$$

donde $\mu_t = \begin{pmatrix} \mu_{x_t} & \mu_{B_1} & \cdots & \mu_{B_{N_t}} \end{pmatrix}^T$ y $\Sigma_t = \begin{bmatrix} \Sigma_{x_t} & \Sigma_{x_t,B_1} & \cdots & \Sigma_{x_t,B_{N_t}} \\ \Sigma_{B_1,x_t} & \Sigma_{B_1} & \cdots & \Sigma_{B_1,B_{N_t}} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{B_{N_t},x_t} & \Sigma_{B_{N_t},B_1} & \cdots & \Sigma_{B_{N_t}} \end{bmatrix}$

El EKF-SLAM tiene dos grandes desventajas. La primera es que su complejidad es cuadrática $O(n^2)$ en el número de balizas del mapa (tabla 2.1), por lo que no se puede aplicar a mapas grandes. La segunda, y más importante, es que solamente se mantiene una hipótesis de asociación de datos, lo que hace al sistema más sensible a posibles errores en la asociación, provocando a menudo errores de asociación que se tornan irrecuperables.

Para solucionar el problema de la complejidad, varios autores [Leonard y Feder, 1999; Guivant y col., 2001] y [Williams, 2001] han desarrollado métodos que descomponen el mapa global en mapas más pequeños (submapas), explotando el hecho de que en un periodo de tiempo el robot solo puede estar en una pequeña zona del mapa. Además, también tienen en cuenta que las observaciones de una baliza tienen un efecto débil sobre las posiciones de las otras balizas, en parte debido al alcance limitado de los sensores y a las oclusiones (percepción parcial del entorno).

El método propuesto en [Leonard y Feder, 1999], *Decoupled Stochastic Mapping (DSM)*, descompone el mapa en una colección de submapas pequeños. Esta aproximación es computacionalmente eficiente, pero no genera ningún mecanismo para propagar información entre los submapas, lo que introduce inconsistencias en el mapa global, ya que en realidad los diferentes submapas raramente son estadísticamente independientes. La gran ventaja de este método es que cada vez que se crea un submapa se toma la pose en ese instante como referencia, y por tanto el submapa y la nueva pose del robot se inicializan con incertidumbre nula. Sin embargo, es complejo establecer los criterios para crear los diferentes submapas de modo que sean lo más independientes entre sí.

La aproximación *Compressed EKF filter (CEKF)* [Guivant y col., 2001] reduce el coste computacional realizando una factorización aproximada de la covarianza, haciendo las actualizaciones en áreas locales alrededor del robot y retrasando la actualización del mapa global hasta que el robot se mueve a otra área. Sin embargo, aunque la solución es óptima, solo re-

duce la complejidad en un factor constante y la actualización del mapa global sigue siendo de complejidad $O(n^2)$.

En [Williams y col., 2002] se propone el método *Constrained Local Submap Filter (CLSF)*, para crear submapas independientes en la vecindad próxima al vehículo. El mismo enfoque se sigue en *Sequential Map Joining Technique* [Tardós y col., 2002]. De nuevo se parte de la suposición de que no se comparte ninguna o muy poca información entre los submapas, por lo que se pueden combinar al final para obtener un único y definitivo mapa global. Recientemente en [Paz y col., 2008a] y [Paz y col., 2008b] se presenta el algoritmo *Divide and Conquer SLAM*(*D&C*) para crear mapas locales a la hora de construir mapas de espacios grandes y evitar así que aumente el coste computacional. Para ello emplean un par estéreo y trabajan en 6 grados de libertad (DOF, del inglés *Degree Of Freedom*). Han obtenido resultados en entornos de tamaño medio, tanto en exteriores como en interiores.

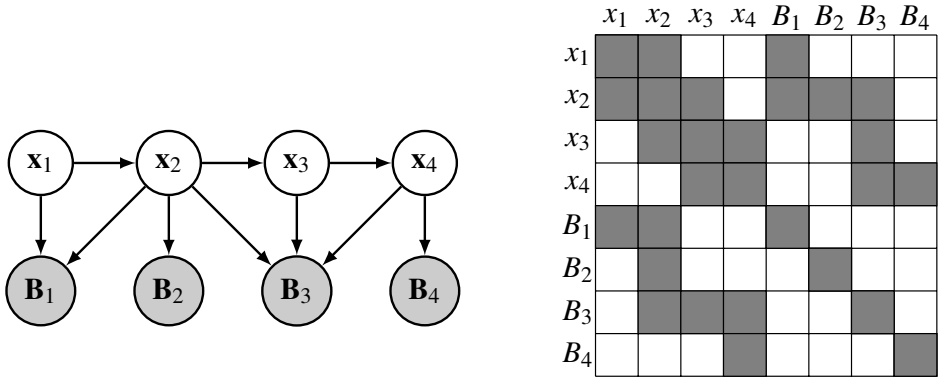
Por último, y en la misma línea que los trabajos anteriores, existen aproximaciones basadas en algoritmos jerárquicos en dos niveles que permiten obtener mapas métricos para grandes entornos en tiempo real, como *Consistent, Convergent, and Constant-Time SLAM* [Leonard y Newman, 2003], *Atlas* [Bosse y col., 2004] y *Hierarchical SLAM* [Estrada y col., 2005]. El nivel más bajo se encarga de construir submapas locales estadísticamente independientes usando la aproximación clásica de EKF-SLAM. En el nivel superior se representa la topología del entorno usando un grafo en el cual los nodos se corresponden con submapas y los arcos representan las relaciones topológicas entre ellos.

2.2.2. Graph-SLAM

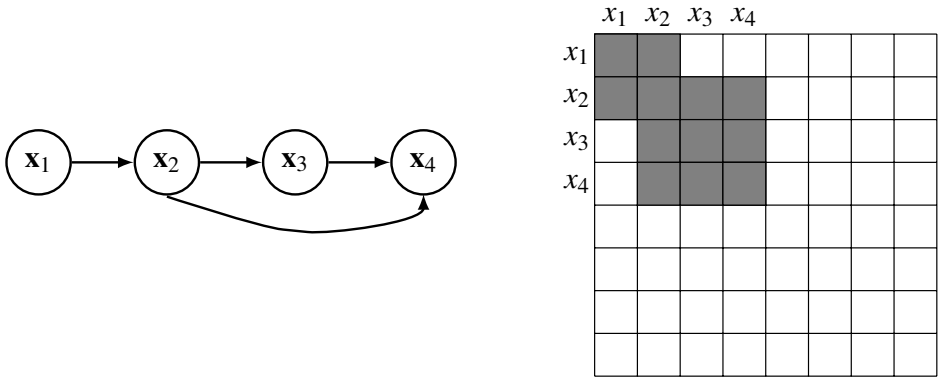
Graph-SLAM [Lu y col., 1997] resuelve el SLAM completo representando el problema como un grafo disperso que conduce a una suma de restricciones cuadráticas no lineales. Optimizando estas restricciones se genera un mapa en base al criterio de máxima similitud y un conjunto de poses correspondientes.

Los nodos del grafo (fig. 2.3) son las poses del robot y las balizas, mientras que los arcos son de dos tipos: movimiento y medida. Los primeros unen dos poses consecutivas del robot y los segundos unen las poses con las balizas que fueron observadas desde cada posición. Cada línea del grafo se corresponde con una restricción no lineal. El resultado de la suma de todas las restricciones es un problema no lineal de mínimos cuadrados.

Para calcular la probabilidad a posteriori del mapa, Graph-SLAM linealiza el conjunto de las restricciones. El resultado de la linealización es una matriz y un vector de información



(a) Dependencias completas.



(b) Dependencias reducidas.

Figura 2.3: Correspondencia entre el grafo de dependencias y la matriz de información utilizada por el algoritmo Graph-SLAM. Los arcos de dependencias se reducen a dependencias entre las poses, por lo que para contemplar que las poses x_2 y x_4 observan B_3 se establece un arco de dependencia entre ellas.

(fig. 2.3(a)). La matriz de información es dispersa por la baja densidad del grafo construido, lo que permite que se pueda aplicar un algoritmo de eliminación de variables. De este modo, se transforma el grafo en uno mucho más pequeño donde solo se definen las poses del robot (fig. 2.3(b)). La probabilidad a posteriori de la trayectoria sobre el mapa se calcula usando técnicas estadísticas de inferencia.

La principal diferencia con los algoritmos EKF-SLAM es en la representación de la información. EKF usa gaussianas caracterizadas por una covarianza y una media, mientras que

Graph-SLAM usa un grafo con restricciones. Por lo tanto, actualizar un EKF es computacionalmente costoso, mientras que añadir restricciones al grafo no tiene prácticamente coste computacional. Gracias a ello, Graph-SLAM puede adquirir mapas de un orden de magnitud superior a EKF-SLAM.

Otra importante diferencia (tabla 2.1) es que Graph-SLAM requiere de una fase de inferencia al final para recuperar el mapa y las poses (SLAM completo), mientras que EKF-SLAM mantiene en todo momento la mejor estimación tanto de la pose del robot como del mapa (SLAM en línea). Como contrapartida, Graph-SLAM tiende a generar mapas mucho más fiables y precisos que EKF-SLAM.

Una limitación de Graph-SLAM es que el espacio de almacenamiento (el grafo) crece con el tiempo (el número de poses del robot), mientras que en EKF-SLAM es independiente del tiempo, ya que solo depende del número de balizas del entorno (tabla 2.1).

En los últimos años han aparecido numerosas aproximaciones que siguen la filosofía de Graph-SLAM, pero introduciendo las restricciones entre poses por medio de odometría visual (VO, del inglés *Visual Odometry*), reduciendo el grafo a sus nodos más significativos (denominados *keyframes*), e implementando métodos de resolución de mínimos cuadrados no lineales muy eficientes. Así, FrameSLAM [Konolige y Agrawal, 2008] realiza una reducción del número de restricciones del grafo mediante la marginalización de variables y la transformación de las restricciones que permanecen, eliminando la mayor parte de los nodos correspondientes a características del entorno y a poses del robot. Las características del entorno se identifican mediante un par estéreo con el detector de características CenSure, y se realiza el cierre de lazos utilizando la transformación entre dos imágenes por medio de RANSAC. En [Konolige y col., 2010] se mejora FrameSLAM incorporando un cierre de lazos basado en una lista de palabras visuales (*Bag-of-Words*, BoW) junto con una comprobación de la consistencia geométrica de la transformación.

Otro trabajo destacado en esta línea es [Newman y col., 2009], donde se presenta un sistema que utiliza cámaras estéreo, sensores láser y cámara omnidireccional. Por una parte, el par estéreo se utiliza para proporcionar la VO y construir el grafo mediante la técnica *sliding window filter*. El cierre de lazos se realiza con el algoritmo *Fast Appearance Based Mapping* (FAB-MAP), basado en BoW y en imágenes omnidireccionales. El grafo construido se optimiza con técnicas clásicas de optimización no lineal que aprovechan las propiedades de dispersión del grafo.

2.2.3. Sparse Extended Information Filter (SEIF-SLAM)

El *Sparse Extended Information Filter (SEIF-SLAM)* [Thrun y col., 2004] presenta una solución al problema del SLAM en línea (tabla 2.1) basada en el filtro de información extendida (EIF). Este filtro representa la distribución gaussiana usando la parametrización canónica: una matriz de información $\Omega = \Sigma^{-1}$ y un vector de información $\xi = \Sigma^{-1}\mu$ (fig. 2.4).

Esta representación de la distribución es la misma que usa Graph-SLAM, pero solo se mantiene la probabilidad a posteriori sobre la pose actual y el mapa (al igual que en el EKF-SLAM). La pose del robot solo está enlazada con un pequeño subconjunto del total de las balizas, a las que se denominan balizas activas (fig. 2.4). Aunque en el grafo se mantiene un nodo para cada baliza y conexiones entre ellas, el peso de cada enlace está relacionado con la distancia entre las balizas. Así, los enlaces con más peso estarán entre las balizas más próximas entre sí.

La matriz de información se puede considerar dispersa, ya que la mayoría de los elementos tienen valores muy pequeños y se pueden aproximar a cero, por lo que la necesidad de almacenamiento es lineal con el número de balizas (tabla 2.1). El resultado final es que la ejecución será más eficiente aunque, por contra, los resultados serán en general menos robustos que los obtenidos por Graph-SLAM o EKF-SLAM.

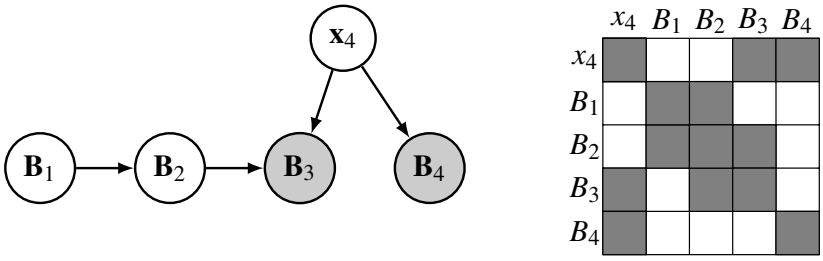


Figura 2.4: Correspondencia entre el grafo de dependencias y la matriz de información utilizada por el algoritmo SEIF-SLAM. Se resaltan las balizas cuyos elementos de la matriz de información son distintos de cero para la posición actual.

Se han presentado varias modificaciones de SEIF-SLAM que mejoran la consistencia del mapa. La primera de ellas, *Exactly Sparse Delayed Filter (ESDF)* [Eustice y col., 2006], elimina la información referente a las balizas del vector de estado y almacena únicamente el histórico de las poses del robot donde se han hecho las observaciones. De esta manera evita que se destruya la dispersión de la matriz de información y utiliza las observaciones como restricciones entre los pares de poses. Por otra parte, *Exactly Sparse Information Filter*

(ESIF) [Walter y col., 2007] mantiene la dispersión de la matriz de información, pero en el proceso de inicialización evita la creación de aquellos enlaces que rompan la dispersión.

2.2.4. FastSLAM

El algoritmo FastSLAM [Montemerlo y col., 2002; Montemerlo y Thrun, 2007] se basa en la propiedad del problema de SLAM por la cual, conocida la pose del robot, las posiciones de los objetos del mapa (balizas) pueden estimarse de forma totalmente independiente unas de las otras, teniendo como única dependencia la incertidumbre de la pose del robot (fig. 2.1). En FastSLAM la PDF resultante se factoriza como:

$$p(x_{1:t}, m, \Psi_{1:t} \,|\, Z_{1:t}, u_{1:t}) = p(x_{1:t} | Z_{1:t}, u_{1:t}, \Psi_{1:t}) \prod_{j=1}^{N} p(B_j \,|\, x_{1:t}, Z_{1:t}, \Psi^n_{1:t}), \tag{2.3}$$

siendo N el número de balizas del mapa y B_j cada una de las balizas.

Esta factorización plantea que el cálculo de la probabilidad sobre la trayectoria y el mapa puede descomponerse en $N+1$ probabilidades. Para ello se utiliza un filtro de partículas de tipo Rao-Blackwell (RBPF, del inglés *Rao-Blackwellized Particle Filter*), donde parte de las variables son estimadas por un filtro de partículas y otro grupo mediante una PDF parametrizada (generalmente una gaussiana). En concreto, en FastSLAM las variables que representan las posiciones de las balizas del mapa se modelan mediante gaussianas independientes aplicando un EKF de baja dimensionalidad para cada una de ellas. Por otra parte, la probabilidad a posteriori de la pose del robot se representa mediante las partículas.

El número total de filtros del sistema para M partículas será de $1+MN$, siendo N el número de balizas en el mapa para cada partícula. Esto permite que el tiempo de ejecución del algoritmo sea logarítmico respecto al número de balizas, en lugar de exponencial como sucede en el caso del EKF-SLAM. En consecuencia, tanto su coste computacional como de almacenamiento son linealmente dependientes (tabla 2.1) respecto al número de partículas (que es constante a lo largo del tiempo) y al número de balizas del mapa (que también es fijo: hipótesis de entorno estático).

FastSLAM permite solucionar tanto el problema de SLAM completo como el problema de SLAM en línea. Otra ventaja es que permite el uso de modelos de movimiento del robot no lineales, mientras que el resto de aproximaciones tienen que realizar linealizaciones de los mismos. Esta característica se vuelve especialmente importante cuando el movimiento del robot es altamente no lineal o la incertidumbre de la pose es relativamente alta.

Sin embargo, la principal ventaja de FastSLAM es que la asociación de datos puede ser diferente para cada una de las partículas, lo que permite al filtro mantener información sobre múltiples asociaciones de datos a lo largo del tiempo. Esto hace al algoritmo muy robusto frente a posibles fallos en la asociación de datos.

Por todos estos motivos FastSLAM constituye la base del algoritmo de SLAM presentado en esta memoria. En el apéndice A se describen en detalle las versiones 1 y 2 de FastSLAM.

2.3. Trabajo relacionado de SLAM visual

En esta sección se hace una revisión de los trabajos más relevantes que se pueden encontrar en la bibliografía relativos al SLAM visual [Fuentes-Pacheco y col., 2012]. Previamente se hace una revisión de la selección y detección de las balizas, así como una breve comparación con aquellas líneas de investigación de visión por computador más relacionadas con el SLAM visual. En el análisis de los diferentes trabajos se ha puesto especial énfasis en el tipo de algoritmo de SLAM y de asociación empleados, en las características utilizadas y si se tienen en cuenta las oclusiones. Las tablas 2.2 y 2.3 muestran un resumen de los trabajos más relevantes, agrupados según el tipo de algoritmo, de cámara y de balizas.

Para una completa comparación de los algoritmos propuestos sería necesaria una valoración cuantitativa. Sin embargo, existen muy pocos conjuntos de datos de test (*benchmarks*) disponibles públicamente y aceptados por la comunidad [Fuentes-Pacheco y col., 2012]. Actualmente existen varios proyectos internacionales tratando de construir tales conjuntos de test: *Rawseeds project*, *The cheddar gorge data set*, *The new college vision and laser data set*, *RGB-D dataset and benchmark*, *Radish*. Desgraciadamente ninguno de ellos incluye el tipo de imágenes que necesita nuestro sistema.

2.3.1. Balizas

Un elemento clave de los algoritmos SLAM es la selección de las balizas del entorno que se emplean para el mapeado. Las características visuales más empleadas en la bibliografía son las esquinas de Harris, Shi-Tomasi o SIFT. No obstante, casi todas estas características son detectadas localmente en las imágenes y dependen en gran medida de las condiciones de luz, texturas que las rodean, de la posición y orientación de la cámara, etc.

Algunos autores proponen representar las balizas del entorno de una forma genérica. Así, en [Folkesson y col., 2007] se propone el espacio-M (*M-Space Feature Representation*), ca-

Tabla 2.2: Resumen del trabajo relacionado de SLAM visual con cámaras estándar.

VISIÓN MONOCULAR					
Basados en la apariencia global de la imagen					
Referencia	**Algoritmo**	**Asociación**	**Cámara**	**Mapa**	**Ocl.**
[Silveira y col., 2008]	Optimización de 2^o orden	ML	Estándar	Planos	—
[Glover y col., 2010]	RatSLAM + FAB-MAP	Bayes + Chow Liu	Estándar	Topológico (SURF)	SI
Basados en características locales de la imagen					
Referencia	**Algoritmo**	**Asociación**	**Cámara**	**Mapa**	**Ocl.**
[Kang y col., 2012]	FastSLAM + NeoSLAM	Vecino más próximo	Estándar	Luces techo	—
[Choi y col., 2010]	EKF	ML	Estándar	Líneas	—
[Choi y col., 2012]	EKF	KLT	Estándar	Harris + Líneas	—
[Klein y Murray, 2007]	PTAM (VO+BA)	ML	Estándar	FAST-10	–
[Pirker y col., 2011]	BA + FAB-MAP	ML	Estándar	SIFT + HoC	SI

paz de representar las balizas con un número variable de dimensiones, lo que permite tratar simultáneamente con diferentes tipos de objetos (puntos, líneas, planos, etc.). Otra ventaja del espacio-M es que la representación de una baliza puede cambiar con el tiempo. Así, la primera medida (imagen) de un punto solo proporciona información sobre su orientación respecto a la cámara. Pero a medida que se obtiene más información (imágenes) se podrá precisar la posición de la baliza en el entorno.

Todas estas limitaciones en la detección de balizas en imágenes también tienen un alto impacto en el cierre de lazos, es decir, cuando una zona del entorno está siendo revisitada. El limitado ángulo de visión de las cámaras estándar hace que esta identificación sea difícil y propensa a la ambigüedad, ya que el nuevo punto de vista puede ser muy diferente al de la visita previa. Por este motivo varios autores introducen métodos específicos para el cierre de lazos, por ejemplo el empleo de imágenes clave o de referencia [Kim y Kweon, 2007]. El uso de cámaras omnidireccionales cuyo eje óptico sea perpendicular al suelo facilita el cierre de lazos, ya que las imágenes son invariantes a la orientación de la cámara.

Por otra parte, conviene recordar que las cámaras son sensores que solamente proporcionan información sobre la orientación de los objetos que son detectados en sus imágenes. Al no proporcionar información de distancia o profundidad no se pueden inicializar las posiciones en 3D de los objetos con una sola captura (imagen). Normalmente se utiliza una inicialización retrasada usando varias imágenes obtenidas desde distintas posiciones de la cámara en

Tabla 2.3: Resumen del trabajo relacionado de SLAM visual con cámaras omnidireccionales.

VISIÓN OMNIDIRECCIONAL					
Basados en la apariencia global de la imagen					
Referencia	**Algoritmo**	**Asociación**	**Cámara**	**Mapa**	**Ocl.**
[Roda y col., 2007]	Teoría información	Información mutua	Lente (100°)	Mosaicos del techo	—
[Scaramuzza y col., 2010]	VO + BoW	ML + RANSAC	Espejo	SIFT	—
[Valiente García y col., 2012]	VO	Búsqueda global similaridad	Espejo	Firma Fourier + SURF	—
[Kawewong y col., 2011]	BoW	ML	Espejo	PIRF	SI
Basados en características locales de la imagen					
Referencia	**Algoritmo**	**Asociación**	**Cámara**	**Mapa**	**Ocl.**
[Lemaire y Lacroix, 2007]	EKF	Seguimiento	Espejo	Harris	—
[Andreasson y col., 2007, 2008]	Graph optimization	ML	Espejo	SIFT	—
[Rybsk y col., 2008]	IEKF + ML linealizada	KLT	Espejo	Características de la imagen	—
[Kim y Oh, 2008]	EKF	ML	Espejo + laser 2D	Líneas	—
[Wongphati y col., 2009]	FastSLAM	ML	Espejo	Líneas	—
[Schlegel y Hochdorfer, 2008]	EKF	Mahalanobis + geometría	Espejo	SIFT	—
[Lui y Jarvis, 2012]	Marco probabilístico	Distancia Euclídea	Estéreo (espejo)	Coeficientes Haar	—
[Jeong y Lee, 2005]	EKF	ML	Lente (150°)	Harris	—
[Saedan y col., 2007]	Filtro partículas + BD imágenes	ML	Lente	Descomposición Wavelet	—

el entorno. De esta forma es posible generar una posición inicial de cada baliza a partir de la fusión de las diferentes medidas adquiridas desde distintas posiciones.

En [Solà y col., 2005a] se hace un breve estudio de los diversos métodos de inicialización existentes ([Bailey, 2003; Davison y col., 2004]) y los clasifican en dos tipos: inicialización retrasada e inicialización instantánea. También proponen sendos métodos, uno en cada categoría. El primero consiste en un mecanismo de inicialización retrasada donde las balizas se crean como una suma ponderada de gaussianas cuyo estado inicial se expresa en función de la pose actual del robot y no del sistema de referencia global. El segundo método propuesto

es una inicialización instantánea a la que denominan *Federated Information Sharing (FIS)* en el que la posición de cada baliza se modela con una función de probabilidad cónica y la representan por una serie geométrica de gaussianas [Solà y col., 2005b].

La ventaja de la inicialización instantánea es que permite incluir las balizas en el mapa desde el mismo momento en el que son detectadas. De ese modo, aunque no proporcionan información de distancia, si proporcionan información sobre ángulo y orientación. La principal desventaja es que se incrementa el coste computacional al aumentar el número de variables en el vector de estado. Por su parte, la mayor ventaja de los métodos de inicialización retrasada es que permiten realizar el seguimiento de un mayor número de balizas no inicializadas debido a su bajo coste computacional, y también pueden hacer una selección previa de las mejores balizas. Esta última característica es muy importante en aquellos entornos donde el número de balizas sea elevado o el error en las medidas sea considerable.

En [Bailey, 2003] se presenta una extensión del EKF-SLAM para cámaras basada en inicializar las balizas usando observaciones desde dos poses diferentes. Por otra parte, Davison y col. [Davison y col., 2004] proponen representar e inicializar las balizas con un filtro de partículas. De esta forma mejoran la robustez del algoritmo EKF-SLAM con una cámara con un FOV de 90 grados. Por último, en [Civera y col., 2007] se estima la pose de las balizas a partir de la *Inverse-Depth Parametrization* (IDP), incorporando la inversa de la distancia al vector de estados del EKF.

Por último, algunos autores emplean sensores adicionales para poder inicializar las balizas con una sola adquisición. Por ejemplo, en [Kim y Oh, 2008] se emplean como balizas las líneas verticales del entorno. Estas se pueden extraer fácilmente a partir de una imagen panorámica: son aquellas que pasan por el centro óptico de la cámara cuando su eje óptico es perpendicular al suelo. Por otra parte utilizan los datos de un láser 2D cuyo plano de medición está paralelo al suelo para calcular la posición de cada línea vertical en el entorno, y por tanto en el mapa.

2.3.2. Aproximaciones desde visión por computador

El SLAM con una única cámara se puede comparar con el problema de estimar la estructura del entorno a partir del movimiento de la cámara (SFM, del inglés *Structure From Motion*). La diferencia entre SFM y SLAM no se limita exclusivamente a los métodos empleados sino, sobre todo, a los objetivos que se persiguen. Se puede considerar que ambos problemas son en realidad el mismo pero tratados con prioridades diferentes [Tomasi y Kanade, 1992; Dellaert

y col., 2003]. En ambos casos el objetivo es determinar, a partir de una colección o una secuencia de imágenes, la estructura tridimensional del entorno y los parámetros que definen la pose de la cámara (posición y orientación) desde donde las imágenes han sido capturadas. Sin embargo la prioridad en el SFM es obtener la mayor exactitud posible, sin importar el tiempo de ejecución, mientras que en el SLAM el objetivo principal es estimar la pose del robot y el mapa del entorno, preferiblemente en tiempo real.

Otro ámbito directamente relacionado es el del cálculo de la VO, donde el movimiento (básicamente local) del robot se obtiene a partir de una secuencia de dos o más pares de imágenes. Sin embargo, la VO se limita a recorridos relativamente cortos (en espacio y en tiempo) y no se pretende que las trayectorias y las características finales sean coherentes globalmente. La VO debe calcularse en tiempo real porque la posición ha de estar disponible en línea. Existen soluciones avanzadas con un nivel de deriva bajo a pesar de haber recorrido largas distancias [Solà y col., 2008; Konolige y col., 2011].

Una restricción importante de los sistemas con una única cámara es que solo se puede estimar el entorno y el movimiento de la cámara hasta un factor de escala desconocido. Por este motivo, se debe incluir información adicional para calcular dicho factor de escala. Así en [Murillo y col., 2012] se demuestra que se consigue una resolución suficiente en el mapa generado con solo medir la oscilación vertical de una persona al caminar de forma regular. Con esta información se puede determinar heurísticamente la frecuencia de los pasos y estimar la velocidad. Otros autores emplean distintos sensores con este fin: GPS [Civera y col., 2010; Dusha y Mejias, 2012], sistemas inerciales [Nützi y col., 2011; Jones y Soatto, 2011], etc.

En cualquier caso, la principal limitación de todos estos algoritmos (tanto VO como SFM) es que necesitan que el entorno tenga suficiente densidad de características relevantes para que funcionen correctamente. Es decir, que en cada imagen adquirida exista un conjunto relativamente alto de balizas y que estas estén bien distribuidas sobre toda la imagen. Además, se necesita bastante solape entre cada par de imágenes consideradas para establecer de forma segura y precisa la relación entre ambas imágenes. Por todo ello, no está clara la aplicación de estos algoritmos en entornos donde existan pocas características (el techo), estén relativamente separadas entre sí (las luces) o se puedan ocluir frecuentemente (p. ej. debido a las personas en movimiento que rodean al robot).

2.3.3. SLAM con múltiples cámaras

La configuración de múltiples cámaras más común en la bibliografía es el par estéreo. Como las cámaras estéreo permiten obtener directamente la posición y orientación de las balizas detectadas, todos los algoritmos de SLAM pueden ser aplicados directamente y sin ningún tipo de adaptación. Las principales limitaciones vienen dadas por el propio sensor: compromiso entre alcance y precisión, reducido ángulo de visión, etc.

El primer trabajo con visión estéreo fue publicado en el año 1998 por Davison y Murray [Davison y Murray, 1998]. Presenta un modelo de visión activa en el que las cámaras se montan sobre una cabeza mecánica que permite la selección de las mejores características en base al detector de esquinas Harris. El SLAM se soluciona aplicando un filtro EKF estándar.

Un sistema más completo, basado en características SIFT, se describe en [Sim y col., 2007]. La principal contribución de este trabajo es que en lugar de usar la aproximación clásica del EKF, emplea una aproximación basada en un filtro de partículas de tipo Rao-Blackwell y una VO basada en el algoritmo de optimización no lineal de Levenberg-Marquardt. Además, presenta un mecanismo de optimización basado en árboles *KD-tree* que, a pesar del número elevado de balizas, permite su ejecución en tiempo real. Otros trabajos [Marks y col., 2007; Pupilli y Calway, 2006] introducen un filtro de partículas fusionado con el EKF-SLAM.

Sáez y col. [Sáez y Escolano, 2005] también calculan la odometría de manera visual. Pero su aproximación presenta un enfoque diferente, ya que se basa en el uso de un algoritmo *off-line* de rectificación global del mapa que trata de minimizar la energía estimada a partir de la entropía. Además añade un algoritmo de exploración y extiende la solución a 6-DOF [Sáez y col., 2005] y a entornos submarinos [Sáez y col., 2006].

Solà y col. presentan un SLAM multicámara [Solà y col., 2008] que permite fusionar información de una o varias cámaras: sistemas estéreo, cámaras moviéndose de forma independiente y cámaras de diferentes características. A pesar de tener más de un sensor, por lo que teóricamente podrían inicializar las balizas del mapa con una sola adquisición, apuestan por una inicialización retrasada (IDP [Civera y col., 2007]) argumentando que de esta forma consiguen un algoritmo más genérico.

Otra configuración también presente en la bibliografía es el empleo de tres cámaras (*triclops*). En [Se y col., 2002] se utiliza un algoritmo EKF-SLAM con características tipo SIFT para la creación del mapa y la localización del robot. Sin embargo, los experimentos solo se han llevado a cabo en un pequeño entorno de laboratorio. Por otra parte en [Marzorati y col., 2007] se emplea un esquema muy similar pero usando segmentos como balizas.

Finalmente, existen algunos casos de múltiples cámaras omnidireccionales. Así, la propuesta de Lui y Jarvis [Lui y Jarvis, 2012] se basa en una cámara estéreo catadióptrica con una línea base variable. De esta forma es posible calcular la posición 3D de cada característica en las imágenes en una única adquisición. Los autores generan tanto un mapa topológico como un mapa de rejilla 2D y han realizado experimentos tanto en interiores como en exteriores.

2.3.4. SLAM con cámaras estándar

Klein y Murray [Klein y Murray, 2007] introducen el algoritmo PTAM (*Parallel Tracking And Mapping*), donde un mapa de características dispersas es actualizado empleando el método *Bundle Adjustment* (BA) al mismo tiempo que la cámara se mantiene localizada continuamente en dicho mapa. En la formulación original PTAM se limitaba a entornos pequeños, principalmente debido al coste computacional del algoritmo BA. Castle y col. [Castle y col., 2011] han extendido el algoritmo a entornos más grandes empleando varios mapas locales.

Recientemente se han realizado numerosos avances en este área. Así, McDonald y col. [McDonald y col., 2012] presentan un SLAM 6D visual y multi-sesión para una cámara estéreo Bumblebee a partir del algoritmo *windowed bundle adjustment* y utilizando para la identificación y cierre de lazos un método basado en la apariencia; concretamente, emplean BoW. Mei y col. [Mei y col., 2011] se basan en un algoritmo tipo BA sobre imágenes estéreo capaz de ejecutarse en tiempo real ($30-45Hz$), en largas trayectorias (del orden de km) y con una precisión de unos pocos metros. Un trabajo reciente [Strasdat y col., 2012] afirma que los algoritmos de SLAM basados en imágenes de referencia (*keyframe*) mejoran el rendimiento de los basados en filtros, específicamente el EKF-SLAM. Sin embargo no está claro que se pueda mantener esta afirmación sobre técnicas de filtrado más avanzadas o cuando se requiere una precisión inferior al metro.

Algunos trabajos emplean una aproximación directa del SLAM visual [Silveira y col., 2008] y usan la intensidad de las imágenes como observaciones en lugar de características. Estos autores consideran el SLAM como un problema de estimación de los parámetros que optimicen el alineamiento de las imágenes sucesivas de una secuencia de vídeo a una imagen de referencia. Su principal limitación son los cambios en las condiciones lumínicas, el alto coste computacional y que solamente pueden modelar superficies planas.

Un punto de vista muy diferente es el aportado por [Choi y Oh, 2007]. En este caso su objetivo es crear un mapa de rejilla (y no un mapa de balizas) pero empleando un sensor pasivo (una cámara estándar) en vez de un sensor activo (láser o ultrasonidos). El método propuesto

se basa en detectar los puntos de unión entre los obstáculos y el suelo, de tal manera que la unión de esos puntos con el centro de la cámara constituyen los rayos virtuales de los ultrasonidos o el láser.

Existen muy pocos trabajos de SLAM que utilicen las características del techo como balizas y la mayoría de ellos se basan en cámaras estándar. La primera propuesta presentada [Jeong y Lee, 2005] fue un EKF que podía ser ejecutado en tiempo real en un área muy pequeña. Por su parte [Choi y col., 2010, 2012] aplican un algoritmo de Monte Carlo modificado para la tarea de localización y un algoritmo EKF estándar para SLAM. Emplean como balizas los bordes, las luces y los círculos del techo, pero solo ha sido probado en entornos muy pequeños.

Finalmente, en [Kang y col., 2012] se comparan tres algoritmos de SLAM diferentes (EKF, FastSLAM y el nuevo NeoSLAM) en varios entornos de interiores y sobre trayectorias muy cortas de únicamente un lazo. Para funcionar correctamente, los tres algoritmos necesitan una densidad de balizas relativamente alta (más de 140 en un área de $21x21m^2$). De hecho, los autores tuvieron que añadir balizas artificiales en entornos con pocas balizas o con balizas distribuidas irregularmente.

2.3.5. SLAM con cámaras omnidireccionales

Hasta donde llega nuestro conocimiento, el primer trabajo que usa una cámara omnidireccional para resolver el problema de SLAM fue desarrollado por Deans [Deans, 2000]. La aproximación se basa en el uso del *invariant filter* para eliminar la correlación entre el error de la estimación de la pose y el error en la estimación de las posiciones de las balizas. Las balizas se inicializan de modo retrasado usando dos poses en las que han sido observadas y se triangula su posición respecto a las balizas que ya están incluidas en el mapa. Estas relaciones entre balizas se optimizan mediante el algoritmo de Levenberg-Marquardt, y se usan para estimar la pose del robot y la odometría. Se considera un trabajo preliminar, ya que tanto la extracción de las balizas como el proceso de asociación de datos se realiza de manera manual. Se utilizan balizas artificiales y el entorno de los experimentos es de tan solo $5x3{,}5m^2$, y la estimación de la pose de las balizas tiene una precisión de medio metro, aunque no se indica nada acerca de la pose del robot.

Los autores Lemaire y Lacroix [Lemaire y Lacroix, 2007] presentan una adaptación del EKF-SLAM clásico para sensores que solo miden orientación. Su principal aportación es el método para la inicialización retrasada de las balizas. Este método se basa en aproximar la

profundidad a la que se encuentra la baliza a través de una gaussiana consistente, i. e., su media debe estar dentro de un rango permitido. Para ello se inicializa cada baliza como la suma de las posibles gaussianas, y estas se mantienen fuera del vector de estado y se actualizan en cada paso, eliminando de la suma las gaussianas correspondientes a hipótesis erróneas hasta que queda una sola gaussiana. Si esta se considera consistente pasa al mapa, y si no será eliminada. Este mecanismo consigue que solo las mejores características se añadan al mapa, por lo que no se genera un elevado número de balizas, y por lo tanto la actualización del EKF será más eficiente computacionalmente. Además incorpora un mecanismo de cierre de lazos a través de consultas a una base de datos de imágenes adquiridas previamente que están próximas a la posición estimada del robot en el instante actual.

En [Andreasson y col., 2007] se presenta un método minimalista basado en grafos capaz de obtener mapas topológicos correctos y métricos fiables para entornos de tamaño medio y grande con un bajo coste computacional. En el grafo los nodos representan las poses desde donde se captura cada imagen y los arcos representan las relaciones entre dichas poses. Estas relaciones se basan en la odometría del robot y en la similitud entre las imágenes. Para calcular el mapa global, se aplica el algoritmo *MultiLevel Relaxation* (MLR) [Frese y col., 2005] sobre el grafo. El criterio a minimizar es la distancia obtenida a partir de la similitud entre las imágenes del grafo. El algoritmo no calcula la pose de las balizas, por lo que es escalable. En [Andreasson y col., 2008] se extiende el trabajo para la estimación de SLAM multirobot.

Rybsk y col. [Rybsk y col., 2008] presentan dos algoritmos para resolver el problema del SLAM. El primero está basado en el EKF Iterativo (IEKF, del inglés *Iterated Extended Kalman Filter*) y se puede ejecutar en tiempo real. El segundo es un método *off-line* basado en el estimador de máxima verosimilitud que construye un mapa topológico a partir de las imágenes panorámicas (firmas) capturadas por el robot. Por otra parte, para cerrar lazos y reconocer lugares ya visitados, se almacena una imagen cuando esta es suficientemente dispar a las ya almacenadas según el algoritmo Lucas-Kanade-Tomasi (KLT).

En [Wongphati y col., 2009] se encuentra la aproximación más parecida a la propuesta en esta memoria en cuanto al tipo de algoritmo de SLAM implementado. Se presenta una solución que emplea un filtro de partículas FastSLAM 1.0, en la que como modelo de medida se utilizan las líneas verticales del entorno extraídas de las imágenes. La inicialización de las posiciones de las líneas se realiza mediante triangulación, empleando dos imágenes consecutivas y la distancia entre ellas. Para adaptar el algoritmo y utilizar múltiples medidas simultáneamente se escoge el resultado con máxima probabilidad.

Otra aproximación basada en un filtro de partículas es la presentada en [Saedan y col., 2007]. En ella se crea un mapa híbrido topológico y métrico a partir de un mecanismo basado en la similitud entre las características de las imágenes. Cada nodo del mapa representa una imagen y su pose, mientras que los enlaces son las relaciones entre las poses. La pose de un nodo nuevo se estima a partir de la odometría y de las poses de los otros nodos. También se permite la eliminación y fusión de nodos, y no todos los nodos están enlazados entre sí (es decir, existen submapas). El filtro de partículas se encarga solo de la parte de localización y se estima la pose del robot a partir de la pose del nodo más cercano. La evaluación de las partículas en el filtro se basa en la similaridad entre las imágenes. Además se añade un mecanismo de cierre de lazos basado en un proceso de búsqueda global sobre todos los nodos según la similitud entre imágenes, que consigue concentrar las partículas alrededor de los nodos candidatos cuando el valor de similitud es muy alto.

En [Schlegel y Hochdorfer, 2008] se sigue la aproximación clásica EKF-SLAM presentada por [Bailey, 2003] para sensores que solo miden orientación. Como modelo de medida utiliza el ángulo azimut de las características obtenido sobre las imágenes, por lo que se evita el proceso de calibración de la cámara. La pose de las balizas se restringe al plano del suelo y su inicialización se estima como el punto de cruce de dos rectas en un plano 2D definidas por un par de medidas tomadas en dos instantes de tiempo. Su principal aportación es el método de selección de balizas basado en características SIFT. Para reducir el coste computacional introducen una medida de calidad basada en la distancia de Kullback-Leibler para evitar inicializaciones mal condicionadas que puedan desestabilizar el EKF.

En [Roda y col., 2007] optan por orientar la cámara hacia el techo y construyen un mapa en forma de mosaico del cielo fusionando un conjunto de imágenes rectificadas. La estimación del movimiento del robot se realiza maximizando la información mutua entre dos imágenes consecutivas. La técnica empleada para la construcción del mosaico de imágenes se basa en alinear las imágenes (rectificadas) aplicando un algoritmo de minimización de la energía (*fast energy*), en el que se evalúan las acciones candidatas midiendo la información mutua entre las imágenes mediante la fórmula de la entropía de Shanon. Para evitar la deriva del mapa, cada k iteraciones se aplica una fase de rectificación global en la que se corrige la desviación de la trayectoria global minimizando la entropía total del mapa.

Por otra parte, Scaramuzza y col. [Scaramuzza y col., 2010] emplean una cámara omnidireccional para crear una BoW que puede ser empleada para identificar y cerrar lazos. Para estimar el movimiento de la cámara emplean VO basada en características SIFT. En la misma línea, Valiente y col. [Valiente García y col., 2012] emplean una cámara catadióptrica y odo-

metría visual. Sin embargo, su principal contribución es simplificar el emparejamiento entre las características de dos imágenes consecutivas empleando restricciones en sus posiciones.

2.3.6. SLAM en entornos dinámicos

La presencia de objetos en movimiento (entornos dinámicos) es la principal causa de oclusiones en SLAM. Varios autores han abordado el problema del SLAM con la detección y seguimiento de objetos en movimiento (DATMO, del inglés *Detection And Tracking of Moving Objects*) [Wang y col., 2007; Vu y col., 2007], aunque tanto la densidad de objetos en movimiento como el grado de oclusiones es bajo. De todos modos nuestro interés no es tanto la integración de SLAM con DATMO sino el SLAM bajo condiciones severas de oclusión, independientemente de la fuente de dicha oclusión. Así, nos interesan tanto aquellas propuestas que explícitamente representan y manejan oclusiones como aquellas otras que únicamente evalúen su rendimiento bajo condiciones variables de oclusión, aunque no lleguen a implementar ninguna técnica específica para su gestión. El trabajo presentado en esta memoria pertenece a este segundo grupo de métodos.

Hemos encontrado un único método en esta línea: el algoritmo PIRF-Nav 2.0 [Tongprasit y col., 2011; Kawewong y col., 2011]. Este método emplea el detector de características PIRF (*Position-Invariant Robust Features*), que es extremadamente robusto a los cambios dinámicos, ya que se basa en modelar objetos lejanos. La propuesta presentada se ha probado en varios entornos, incluido uno que constituye un reto relevante: un comedor universitario durante la hora del almuerzo. Los autores encontraron que más del 50 % de los descriptores (características) provenían de los objetos en movimiento, lo que afecta negativamente tanto a la robustez como al rendimiento. Al mismo tiempo, demostraron que las características PIRF se pueden emplear incluso en entornos de interiores, en donde la probabilidad de detectar objetos lejanos es baja. Aún así, el porcentaje de recuerdo o recuperación en las trayectorias de pruebas en un entorno tan masificado como el comerdor universitario llega al 88 %.

Sí existe algún ejemplo más de autores que tratan de crear un mapa del entorno en días diferentes, por lo que tienen que tener en cuenta los posibles cambios (normalmente poco importantes). Pirker y col. [Pirker y col., 2011] proponen el algoritmo CD-SLAM para el SLAM continuo basado en la selección de imágenes de referencia y en un método de *Sliding Window Bundle Adjustment*. Para cerrar lazos organizan las poses de la cámara en un grafo no dirigido y sin ponderar, y se detectan las imágenes revisitadas por medio de FAB-MAP [Cummins y Newman, 2008]. Después de detectar el cierre de un lazo, aplican un procedimiento de opti-

mización que minimiza el error de las restricciones de las posiciones relativas del subconjunto de imágenes de referencia que forman dicho lazo (un subgrafo).

Otra novedad importante del trabajo de Pirker y col. es el descriptor HoC (*Histogram of Oriented Cameras*) asociado a cada una de las imágenes de referencia. Este descriptor almacena para cada baliza información sobre su importancia (peso) para el proceso de localización y sobre su visibilidad (básicamente la posición y orientación de la cámara cuando la imagen ha sido capturada). El empleo de HoC mejora la asociación de datos e implícitamente ayuda a filtrar regiones desactualizadas del mapa, ya que más del 80 % de las características de cada imagen son rechazadas. El algoritmo ha sido probado sobre trayectorias largas (1.200 m) en entornos cotidianos tanto interiores como exteriores y en diferentes días. El error cometido es menor al medio metro, pero únicamente se ha medido en 10 posiciones de referencia. Por último, los autores demuestran que revisitar un área varias veces solo incrementa marginalmente el tamaño del mapa.

Otro ejemplo similar es el de Glover y col. [Glover y col., 2010], que presentan un método basado exclusivamente en la apariencia y que realiza el SLAM de entornos exteriores (calles) en días diferentes. Para ello han desarrollado un sistema híbrido que combina RatSLAM [Milford y Wyeth, 2008] y FAB-MAP [Cummins y Newman, 2008] para ser invariante respecto a las variaciones de las condiciones lumínicas.

CAPÍTULO 3

SISTEMA DE VISIÓN Y ROBOT MÓVIL

En este capítulo se describen los dos componentes necesarios para implementar los algoritmos de localización y SLAM presentados en esta memoria: la cámara omnidireccional y el robot móvil. En primer lugar se presenta el sistema de visión que engloba, además de la propia cámara omnidireccional, los siguientes procesos y modelos:

- El proceso de detección de las balizas a partir de una imagen.
- El modelo de la cámara, que permite obtener las medidas trasformando la posición de las balizas del sistema de referencia de la imagen (píxeles) al sistema de referencia 3D del mundo (ángulos).
- El modelo de medida u observación, que realiza una estimación de la medida en función de la posición de la baliza y la pose del robot.

En segundo lugar, se presentan los robots móviles utilizados, se describe el modelo de movimiento odométrico asociado a los mismos, y finalmente se definen las matrices de covarianza que representan el ruido del modelo de movimiento tanto en el espacio de controles como en el espacio de estados.

3.1. Descripción del sistema de visión

El sistema de visión consta de una cámara omnidireccional (fig. 3.1) basada en una lente gran angular de tipo ojo de pez (*fish-eye*). La cámara seleccionada (tabla 3.1) es muy común en tareas de visión artificial por su alta sensibilidad: permite capturar a altas velocidades,

(a) Cámara MDCS2. (b) Lente FE185CO46HA-1. (c) Imagen obtenida (sin filtro IRP).

Figura 3.1: Cámara omnidireccional utilizada.

con una gran tolerancia al ruido y en condiciones lumínicas extremas . Por su parte la lente omnidireccional es de alta resolución y con un ángulo de visión muy amplio (tabla 3.2), lo que facilita la percepción de las balizas en entornos con un baja densidad de balizas y en situaciones con oclusiones frecuentes y/o severas.

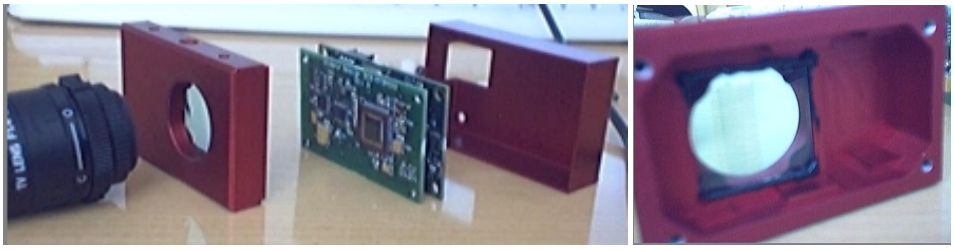
Tabla 3.1: Características de la cámara utilizada.

Marca	Modelo	Resolución	FPS	Lente	Formato	Imagen	Ganancia
Videre Design	MDCS2	640x480	60	C/CS	1/2	Bayer	0 – 18 dB

Tabla 3.2: Características de la lente gran angular [Fujifilm Europe, 2013].

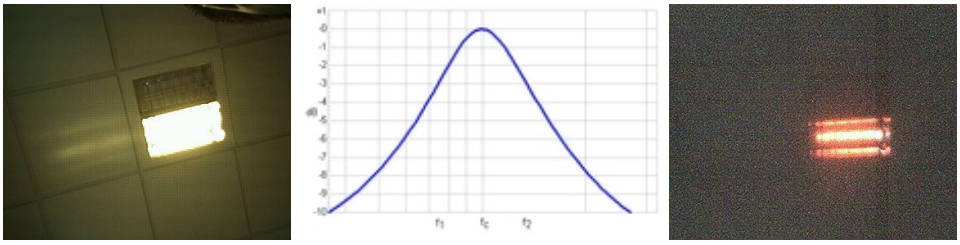
Marca	Modelo	Formato	Distancia Focal	FOV	Peso
Fujinon	FE185CO46HA-1	1/2” 1/4”	1,4 *mm*	185°	140 g

El mapa de balizas utilizado en los algoritmos de localización y SLAM descritos en esta memoria es un mapa de las luces del techo, lo que aporta una serie de ventajas. En primer lugar, todos los edificios suelen tener fuentes de iluminación, por lo que no es preciso alterar el entorno. En segundo lugar, en entornos muy concurridos el techo es la zona en la que las medidas sensoriales se ven menos afectadas por las personas en movimiento. En tercer lugar, las luces presentan características fácilmente reconocibles y detectables, a diferencia de otro tipo de características visuales. No obstante, el principal inconveniente es la complejidad que resulta de la fase de asignación o asociación de datos (entre las medidas y las balizas), ya que las luces suelen ser indistinguibles entre sí.



(a) Despiece de la cámara. (b) Montaje del filtro.

Figura 3.2: Montaje del filtro pasa banda infrarrojo en la cámara MDCS2.



(a) Imagen sin filtro. (b) Respuesta en frecuencia. (c) Imagen con filtro.

Figura 3.3: Efecto del filtro pasa banda infrarrojo sobre las imágenes.

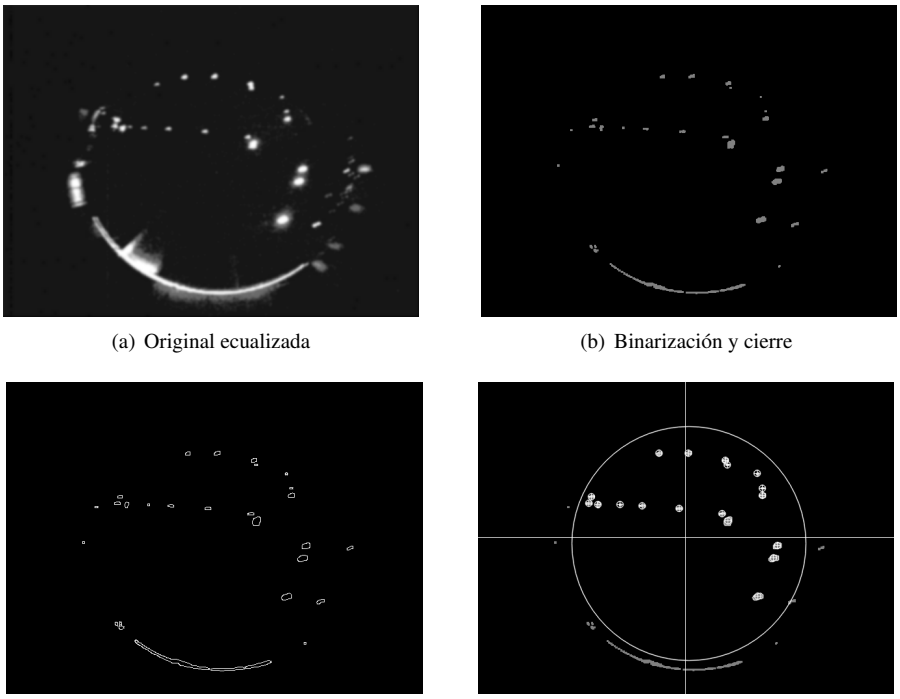
Para facilitar la detección de las luces en las imágenes se ha añadido un tercer elemento al sistema de visión omnidireccional: un filtro pasa banda infrarrojo (IRP, del inglés *InfraRed bandPass filter*), que atenúa las componentes visibles de la luz y solo deja pasar el rango cercano al infrarrojo. El filtro (fig. 3.2) posee características similares a los filtros comerciales RT830 o IR-85 [Hoya Optics, 2013; Edmund Optics, 2013]. Su efecto sobre las imágenes es la atenuación de los elementos de la escena a excepción de las luces, que resaltan sobre el resto porque, además del espectro de luz visible, también tienen una fuerte componente infrarroja (fig. 3.3).

Aunque el empleo del filtro pasa banda IR facilita el procesamento, mejora la robustez y reduce el ruido en la percepción, sin su uso (es decir, utilizando únicamente el espectro visible) también se han obtenido resultados aceptables. A continuación se presenta el proceso de detección de las balizas (luces) a partir de las imágenes adquiridas.

3.1.1. Detección de balizas

El proceso de detección de balizas determina las propiedades y, sobre todo, la posición en la imagen de las características relevantes del entorno, en este caso, las luces. La detección consta de tres etapas: segmentación, extracción de características y reconocimiento.

3.1.1.1. Segmentación

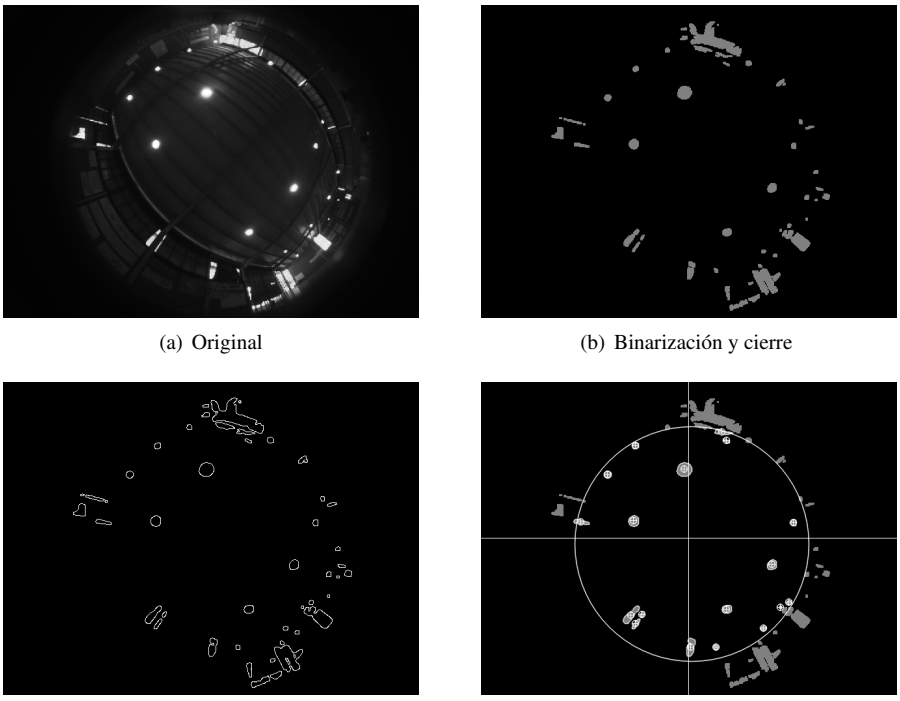


(a) Original ecualizada (b) Binarización y cierre

(c) Segmentación (d) Selección

Figura 3.4: Detección de balizas en el entorno Domus.

La segmentación de una imagen consiste en dividirla en regiones con significado. En este caso, el objetivo es distinguir las balizas del resto de objetos del entorno. Las imágenes son adquiridas por la cámara en escala de grises (figs. 3.4(a) y 3.5(a)). La primera operación que se realiza sobre ellas es la de binarización, a partir de la cual se obtiene una imagen en la que se realzan las regiones de interés para el proceso de detección (figs. 3.4(b) y 3.5(b)). A



(a) Original (b) Binarización y cierre

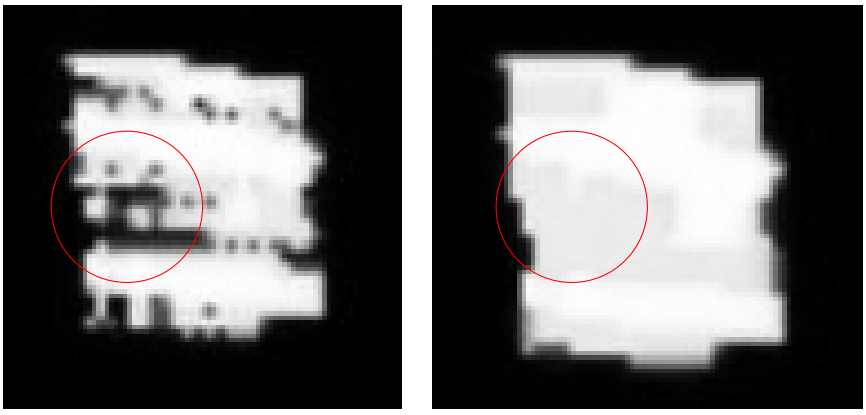
(c) Segmentación (d) Selección

Figura 3.5: Detección de balizas en el entorno Pío XII.

continuación se realiza la detección de bordes, en este caso mediante el filtro Canny, seguida de la extracción de los contornos [Suzuki y Be, 1985] (figs. 3.4(c) y 3.5(c)). La segmentación es una etapa crítica, ya que las balizas filtradas en esta fase no podrán ser recuperadas a posteriori.

Para binarizar la imagen se ha aplicado primero una *umbralización* (con un umbral que es función del nivel medio de la luminancia de la imagen) y después un filtro morfológico de cierre. El filtro de *cierre* (fig. 3.6) permite eliminar puntos aislados y rellenar los huecos de las regiones de interés para evitar así que puedan ser interpretados como más de un objeto. Para ello, el filtro de cierre aplica una operación de *dilatación* seguida de una operación de *erosión*.

En determinados casos la imagen adquirida puede estar muy saturada (fig. 3.7(a)), por ejemplo cuando una luz próxima o muy potente apunta directamente a la cámara. En estas cir-



(a) (b)

Figura 3.6: Aplicación de un filtro de cierre. El círculo destaca como los huecos de la imagen original (a) desaparecen después de aplicar el operador cierre (b).

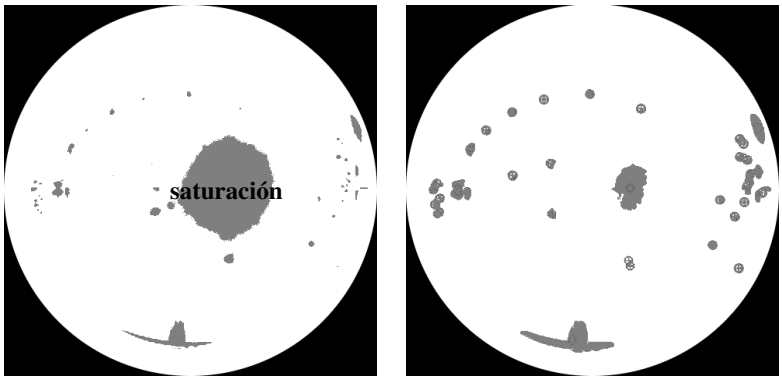
cunstancias el umbral de binarización es muy alto y se elimina demasiada información, por lo que en muchos casos solo se detectaría la luz que ha provocado la saturación. Estas situaciones se detectan por la existencia de contornos de gran tamaño en la imagen (fig. 3.7(a)), y se resuelven aumentando el umbral de binarización en la zona de la saturación y disminuyendo el umbral en el resto de la imagen (fig. 3.7(b)).

3.1.1.2. Extracción de características

La segunda etapa del proceso de detección de balizas consiste en determinar las características que van a definir e identificar cada una de las regiones segmentadas. Se pueden extraer características de diferentes tipos, aunque aquí nos centraremos en aquellas que permiten decidir si una región es una baliza o no (perímetro, radio y elipse), y en las que establecen la posición de las posibles balizas (centroide y azimut).

Centroide. En visión por computador, el centroide de una región se calcula como su centro de gravedad. Se utilizará para representar la posición de la baliza en la imagen (C, fig. 3.8).

Azimut. Es el ángulo formado entre el vector que une el centroide de la baliza y el centro de la imagen, y el eje X de la cámara (A, fig. 3.8).



(a) (b)

Figura 3.7: Aumento del umbral de binarización en una imagen saturada: a) binarización inicial y b) binarización final.

Perímetro. Se define como el número de píxeles del borde de dicha región. Se utilizará este parámetro como un indicador de confianza o fiabilidad de las posibles balizas.

Radio. Se calcula como la distancia del centroide de la región al centro de la imagen (R, fig. 3.8). Esta característica proporciona una indicación aproximada de la distancia de la baliza a la cámara.

Elipse. Se obtiene la elipse asociada a cada región segmentada, es decir, sus ejes mayor y menor.

3.1.1.3. Reconocimiento

En esta etapa se discrimina, en base al vector de características extraídas en la fase anterior, qué regiones segmentadas son balizas (luces) y cuáles no. El principal objetivo es reducir al máximo la detección de falsos positivos (p. ej. reflejos, luz natural, etc.) y eliminar aquellas balizas consideradas como poco fiables (p. ej. muy lejanas o próximas al horizonte). Se han implementado cuatro tipos de filtrado:

Visibilidad. Se descartan todos los objetos con un ángulo $\theta > \theta_{max}$, ya que están muy lejos de la cámara y no son suficientemente fiables. La zona de la imagen que cumple este criterio se delimita en las figuras 3.4(d), 3.5(d) y 3.8 por una circunferencia.

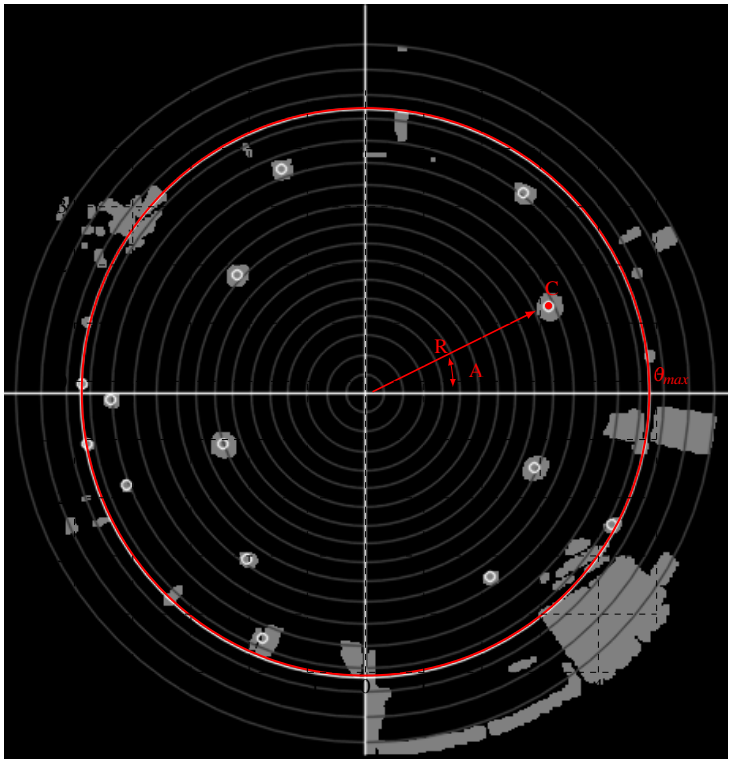


Figura 3.8: Representación gráfica de las características extraídas para cada región de interés (azimut (A), radio (R) y centroide (C)) y de la circunferencia θ_{max} que delimita el criterio de visibilidad.

Tamaño. Solo se consideran balizas aquellas regiones con un cierto tamaño y con una forma regular. Por ello se eliminan aquellas regiones cuyo diámetro es pequeño, por ejemplo debido a reflejos o defectos del proceso de segmentación. Y también aquellas con un perímetro grande, pues suelen corresponderse con zonas de luz natural (ventanas, etc.).

Forma. Se asume que las luces son aproximadamente circulares, por lo que se utiliza la forma de las regiones segmentadas para eliminar falsos positivos. Para ello se comprueba la diferencia entre los ejes menor y mayor de la elipse calculada.

Fusión. Si dos balizas están muy próximas entre sí, se considera que una de ellas puede ser el reflejo de la otra o el resultado de una mala segmentación y se fusionan. El centroide de la baliza resultante será el punto medio de ambas.

3.1.2. Modelo de medida

El modelo de medida u observación (h) permite estimar la medida que obtendría el sensor para una baliza B_j del mapa y una pose del robot x_t. Como cualquier cámara estándar, la cámara omnidireccional utilizada no puede proporcionar distancias, sino únicamente información sobre orientaciones (ángulos). Así, la medida correspondiente a la baliza será $z_j = (\varphi_j \; \theta_j)^T$, donde los ángulos de azimut y elevación definen la recta $3D$ o rayo de proyección en el que se encuentra la baliza (fig. 3.9). El modelo de medida se define como:

$$\overline{z}_j = h(B_j^C, x_t) = \begin{pmatrix} \overline{\varphi}_j \\ \overline{\theta}_j \end{pmatrix} = \begin{pmatrix} atan\left(\frac{y_{B_j}^C}{x_{B_j}^C}\right) \\ acos\left(\frac{z_{B_j}^C}{\sqrt{{z_{B_j}^C}^2 + {y_{B_j}^C}^2 + {x_{B_j}^C}^2}}\right) \end{pmatrix} \tag{3.1}$$

donde $(x_{B_j}^C \; y_{B_j}^C \; z_{B_j}^C)$ son las coordenadas de la baliza en el sistema de referencia de la cámara, y $\overline{z}_j = (\overline{\varphi}_j \; \overline{\theta}_j)^T$ es la medida estimada por el modelo de observación, que en un caso ideal coincidirá con la medida real $z_j = (\varphi_j \; \theta_j)^T$.

En el caso general, los sistemas de referencia de la cámara y del mundo no coincidirán, por lo que será necesario aplicar la siguiente transformación:

$$B_j^C = R^{CW} B_j^W + T^{CW} \tag{3.2}$$

donde R^{CW} y T^{CW} son las matrices de rotación y traslación del sistema de referencia del mundo (W) al de la cámara (C), y B_j^W es la posición de la baliza en el sistema de referencia del mundo.

El modelo de medida no es lineal, y por tanto su utilización en los EKFs de FastSLAM (cap. 5) requiere de la linealización del modelo mediante expansión en serie de Taylor de primer orden. Para ello es necesario calcular las matrices Jacobianas de h con respecto a las variables de estado del robot (H_x) y las variables de la medida (H_m). Esta derivación se detalla en el apéndice B.

3.1.3. Modelo de la cámara

El modelo de la cámara describe cómo una escena $3D$ se transforma en una imagen 2D [De la Escalera y Armingol, 2001]. Los modelos que describen esta transformación difieren dependiendo del tipo de lente, siendo el modelo estándar el *Pin-Hole* (fig. 3.10), que proyecta

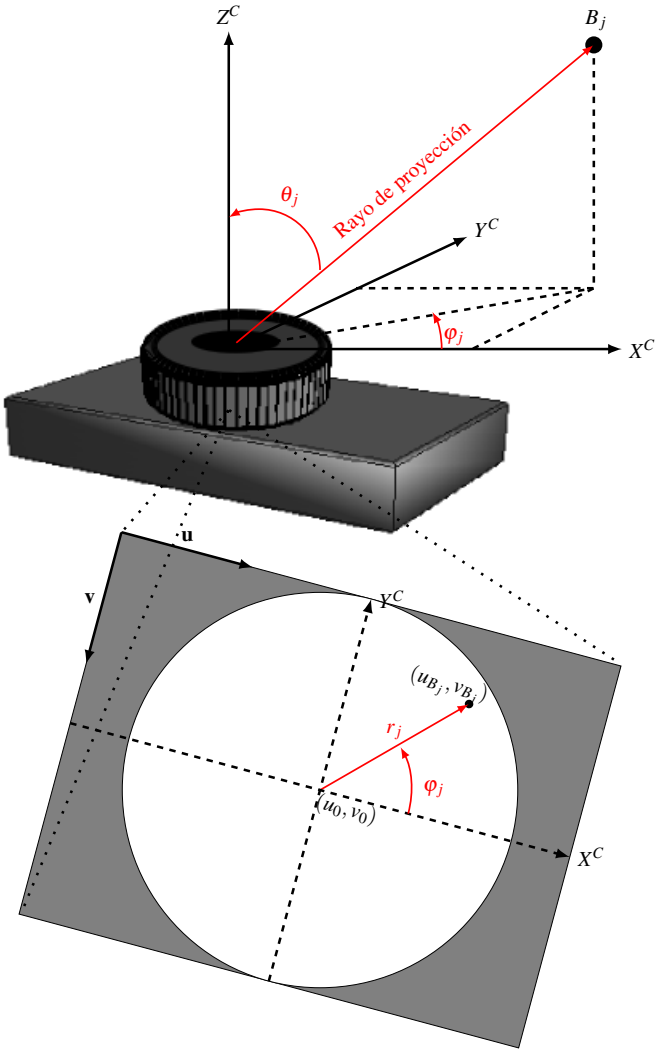


Figura 3.9: Proyección de un punto B_j sobre el plano de la imagen de una cámara. El rayo de proyección de B_j se define por la elevación (θ_j) y el azimut (φ_j) respecto al sistema de coordenadas de la cámara ($X^C\, Y^C\, Z^C$). r_j y φ_j son las coordenadas polares del punto proyectado (u_{B_j}, v_{B_j}), y (u_0, v_0) indica el centro de la imagen.

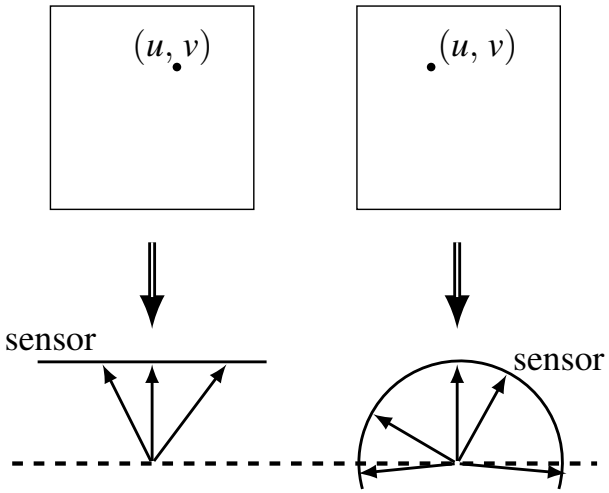


Figura 3.10: Comparación del modelo de cámara *Pin-Hole* (izquierda) frente al modelo de cámara omnidireccional o de retina esférica (derecha).

la escena sobre una retina plana pero está limitado a cámaras con FOV $<<$ 180°. El modelo que mejor se ajusta a la cámara omnidireccional utilizada es el desarrollado por Bakstein y Pajdla [Bakstein y Pajdla, 2002], basado en la proyección de la escena sobre una retina esférica (fig. 3.10). En este modelo, la proyección de un punto B_j sobre la imagen (u_{B_j}, v_{B_j}) se obtiene en función de los ángulos θ_j y φ_j (fig. 3.9):

$$\left.\begin{aligned} r_j &= a \cdot \tan\frac{\theta_j}{b} + c \cdot \sin\frac{\theta_j}{d} \\ u_{B_j} &= u_0 + r_j \cdot \cos\varphi_j \\ v_{B_j} &= \beta \cdot (v_0 + r_j \cdot \sin\varphi_j) \end{aligned}\right\} \tag{3.3}$$

donde β es la relación entre anchura y altura de un píxel, (u_0, v_0) las coordenadas del centro de la imagen, r_j la distancia entre el centro de la imagen y la proyección del punto B_j (u_{B_j}, v_{B_j}), y *a, b, c, d* son los parámetros de ajuste del modelo para una cámara concreta, cuya estimación se realiza mediante un proceso de calibración (apéndice C).

Estas ecuaciones no son invertibles, por lo que no se pueden obtener de forma directa (a partir de las coordenadas de la imagen) los ángulos de la medida correspondiente a una baliza. En el capítulo 5 se presentará la solución propuesta para resolver esta dificultad.

3.2. Robot móvil

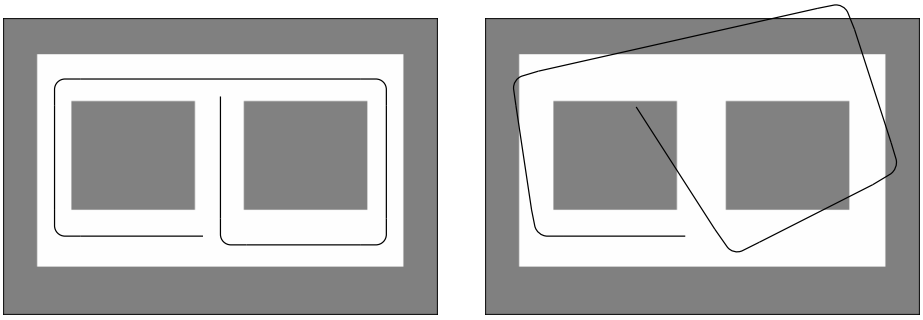
En esta tesis se han utilizado dos tipos de robots móviles, ambos de la empresa Mobile Robots (anteriormente ActivMedia): el P2AT y el P3DX (fig. 3.11). Ambos robots disponen de un anillo de 16 sensores de ultrasonidos y han sido equipados con un sensor láser de la empresa Sick (modelos LMS200 y LMS100, según el experimento). El sistema motriz de ambos robots es de tipo diferencial, aunque en el caso del P2AT (al tener 4 ruedas) los giros se realizan por deslizamiento de las ruedas sobre el suelo, causando que la odometría sea más imprecisa.

(a) P3DX (b) P2AT

Figura 3.11: Robots móviles utilizados.

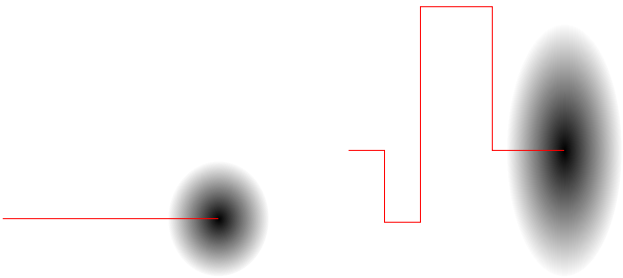
El modelo de movimiento predice el estado actual del robot (x_t) a partir del estado previo (x_{t-1}) y la acción de control (u_t). El movimiento del robot es inherentemente incierto [Thrun y col., 2003] debido tanto al ruido del control como a causas externas no modeladas (fig. 3.12). En la figura 3.13 se muestran dos ejemplos de funciones de densidad de la probabilidad (PDF)

del estado del robot tras aplicar una acción de control. En la parte izquierda se visualiza un movimiento rectilíneo que genera incertidumbre tanto traslacional como rotacional. En la parte derecha se muestra (fig. 3.13(b)) una trayectoria más compleja, por lo que la incertidumbre resultante es mayor.



(a) Pose real (b) Pose odométrica

Figura 3.12: Acumulación de los errores de odometría en una trayectoria.



(a) (b)

Figura 3.13: PDFs del estado del robot tras aplicar una acción de control (más oscuro indica más probable). La PDF ha sido proyectada sobre el plano XY, aunque la PDF original tiene una dimensión más debida a la orientación.

En la bibliografía [Thrun y col., 2003] se describen dos modelos de movimiento probabilísticos específicos: odométrico y de velocidad. El modelo de movimiento odométrico asume que los comandos de movimiento (u_t) se proporcionan en base a la información odométrica del robot sobre la distancia recorrida y el ángulo girado. Por otra parte, en el modelo de movimiento de velocidad los comandos de control son las velocidades lineal y angular enviadas

al robot. En la práctica, el modelo odométrico es más preciso que el modelo de velocidad, y por ello es el que se utilizará a lo largo de esta memoria.

3.2.1. Modelo de movimiento odométrico

El modelo de movimiento odométrico se basa en la diferencia relativa entre la odometría del instante previo, $\overline{x}_{t-1} = (\overline{x}\ \overline{y}\ \overline{\alpha}\ )^T$ y la del instante actual $\overline{x}_t = \left(\overline{x}'\ \overline{y}'\ \overline{\alpha}'\ \right)^T$, donde x e y representan la posición del robot y α su ángulo. El comando de control se define como $u_t = (\overline{x}_{t-1}\, \overline{x}_t)^T$.

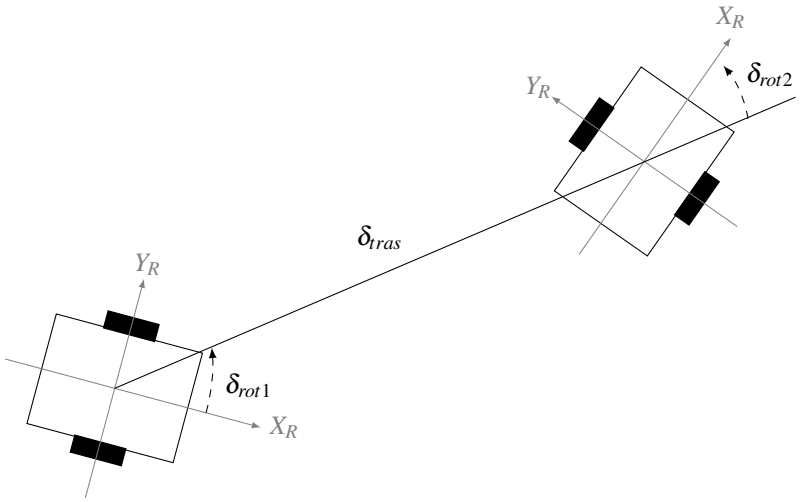


Figura 3.14: Modelo de movimiento odométrico: el movimiento del robot en el intervalo $(t-1,t]$ se aproxima por una rotación δ_{rot1}, seguida de una traslación δ_{tras} y de una segunda rotación δ_{rot2}.

Para extraer la odometría relativa entre las dos posiciones, u_t se transforma en una secuencia de tres pasos (fig. 3.14): una rotación inicial (δ_{rot1}), seguida por un desplazamiento en línea recta (δ_{tras}) y una segunda rotación (δ_{rot2}). El algoritmo 3.1 [Thrun y col., 2003] describe el cálculo del modelo de movimiento a partir de la odometría. En primer lugar, se estiman las variables del control $(\delta_{rot1}\ \delta_{tras}\ \delta_{rot2})^T$ a partir de la odometría (alg. 3.1, líneas 1 - 3). En segundo lugar, se determina la pose actual (x_t) a partir de la pose previa (x_{t-1}) y las variables del control (alg. 3.1, líneas 4 - 6).

Los algoritmos presentados en esta memoria están basados en filtros de partículas, por lo que es necesario definir un modelo de movimiento que contemple el ruido asociado al

Algoritmo 3.1 Modelo de movimiento odométrico: $g(u_t, x_{t-1})$.

1: $\delta_{rot1} = atan2(\overline{y}' - \overline{y}, \overline{x}' - \overline{x}) - \overline{\alpha}'$
2: $\delta_{tras} = \sqrt{(\overline{x}' - \overline{x})^2 + (\overline{y}' - \overline{y})^2}$
3: $\delta_{rot2} = \overline{\alpha}' - \overline{\alpha} - \delta_{rot1}$

4: $x' = x + \delta_{tras}\, cos(\alpha + \delta_{rot1})$
5: $y' = y + \delta_{tras}\, sin(\alpha + \delta_{rot1})$
6: $\alpha' = \alpha + \delta_{rot1} + \delta_{rot2}$

7: **return** $x_t = (x'\, y'\, \alpha')^T$

Algoritmo 3.2 Muestreo del modelo de movimiento odométrico: $x_t \sim p(x_t | x_{t-1}, u_t)$.

1: $\delta_{rot1} = atan2(\overline{y}' - \overline{y}, \overline{x}' - \overline{x}) - \overline{\alpha}'$
2: $\delta_{tras} = \sqrt{(\overline{x}' - \overline{x})^2 + (\overline{y}' - \overline{y})^2}$
3: $\delta_{rot2} = \overline{\alpha}' - \overline{\alpha} - \delta_{rot1}$

4: $\widehat{\delta}_{rot1} = \delta_{rot1} - \text{muestreo}\left(\kappa_1 \delta_{rot1}^2 + \kappa_2 \delta_{tras}^2\right)$
5: $\widehat{\delta}_{tras} = \delta_{tras} - \text{muestreo}\left(\kappa_3 \delta_{tras}^2 + \kappa_4 \delta_{rot_1}^2 + \kappa_4 \delta_{rot_2}^2\right)$
6: $\widehat{\delta}_{rot2} = \delta_{rot2} - \text{muestreo}\left(\kappa_1 \delta_{rot2}^2 + \kappa_2 \delta_{tras}^2\right)$

7: $x' = x + \widehat{\delta}_{tras}\, cos(\alpha + \widehat{\delta}_{rot1})$
8: $y' = y + \widehat{\delta}_{tras}\, sin(\alpha + \widehat{\delta}_{rot1})$
9: $\alpha' = \alpha + \widehat{\delta}_{rot1} + \widehat{\delta}_{rot2}$
10: **return** $x_t = (x'\, y'\, \alpha')^T$

desplazamiento del robot. Esta función se conoce como muestreo del modelo de movimiento y se define en el algoritmo 3.2. La única diferencia con el modelo de movimiento básico es la incorporación del ruido (alg. 3.2, líneas 4 - 6); para ello se calculan las variables del control con ruido $\left(\widehat{\delta}_{rot1}\, \widehat{\delta}_{tras}\, \widehat{\delta}_{rot2}\right)^T$ a partir de sus versiones sin ruido $(\delta_{rot1}\, \delta_{tras}\, \delta_{rot2})^T$. El ruido se genera a partir del muestreo de una distribución normal con media cero y covarianza proporcional a los cuadrados de las variables del control, de acuerdo con la siguiente matriz de covarianza:

$$M_t = \begin{pmatrix} \kappa_1 \delta_{rot1}^2 + \kappa_2 \delta_{tras}^2 & 0{,}0 & 0{,}0 \\ 0{,}0 & \kappa_3 \delta_{tras}^2 + \kappa_4 \delta_{rot1}^2 + \kappa_4 \delta_{rot2}^2 & 0{,}0 \\ 0{,}0 & 0{,}0 & \kappa_1 \delta_{rot2}^2 + \kappa_2 \delta_{tras}^2 \end{pmatrix} \tag{3.4}$$

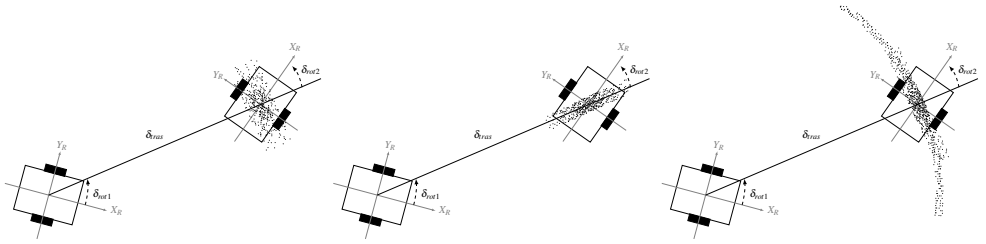


Figura 3.15: Muestreo del modelo de movimiento odométrico para diferentes valores de los parámetros del ruido. Cada PDF está representada por 500 muestras: (a) ruido moderado, (b) ruido de rotación pequeño y de traslación grande y (c) ruido de rotación grande y de traslación pequeño.

De acuerdo con esta definición, el ruido de las rotaciones (y de forma equivalente el ruido de la traslación) depende tanto de la magnitud de la propia rotación, como de la traslación, pudiéndose modular esa dependencia por medio de los parámetros del ruido: κ_1, κ_2, κ_3, y κ_4. Estos parámetros se definen como el porcentaje de error que comete la odometría al estimar el movimiento del robot cuando se realiza un giro o un desplazamiento:

κ_1 cuantifica el error en rotación cuando el robot realiza un giro.

κ_2 cuantifica el error en rotación cuando el robot realiza un desplazamiento.

κ_3 cuantifica el error en traslación cuando el robot realiza un desplazamiento.

κ_4 cuantifica el error en traslación cuando el robot realiza un giro.

La figura 3.15 muestra varias PDFs del estado del robot para diferentes niveles de ruido, definidos por distintos valores de κ_1, κ_2, κ_3 y κ_4.

Cuando se trabaja con EKF, es necesario definir la matriz de covarianza del ruido del modelo de movimiento en el espacio de estados (R_t) y no en el espacio de controles (M_t): $R_t = V_t M_t V_t^T$, donde V_t es la matriz Jacobiana del modelo de movimiento (g) respecto a las variables del control $(\delta_{rot1}\ \delta_{tras}\ \delta_{rot2})^T$:

$$V_t = \begin{pmatrix} -\delta_{tras}\, sin\,(\alpha_{t-1}+\delta_{rot1}) & cos\,(\alpha_{t-1}+\delta_{rot1}) & 0 \\ \delta_{tras}\, cos\,(\alpha_{t-1}+\delta_{rot1}) & sin\,(\alpha_{t-1}+\delta_{rot1}) & 0 \\ 1 & 0 & 1 \end{pmatrix} \tag{3.5}$$

Por tanto, la matriz de covarianza del ruido del modelo de movimiento en el espacio de estados es:

$$R_t = \begin{pmatrix} R(1,1) & R(1,2) & R(1,3) \\ R(2,1) & R(2,2) & R(2,3) \\ R(3,1) & R(3,2) & R(3,3) \end{pmatrix} \tag{3.6}$$

donde

$$\begin{aligned}
R(1,1) &= \delta_{tras}^2 M(1,1)\, sin\,(\alpha_{t-1}+\delta_{rot1})^2 + M(2,2)\, cos\,(\alpha_{t-1}+\delta_{rot1})^2 \\
R(1,2) &= M(2,2)\, cos\,(\alpha_{t-1}+\delta_{rot1})\, sin\,(\alpha_{t-1}+\delta_{rot1}) - \delta_{tras}^2 M(1,1)\, cos\,(\alpha_{t-1}+\delta_{rot1}) \\
&\quad\; sin\,(\alpha_{t-1}+\delta_{rot1}) \\
R(1,3) &= -\delta_{tras} M(1,1)\, sin\,(\alpha_{t-1}+\delta_{rot1}) \\
R(2,1) &= M(2,2)\, cos\,(\alpha_{t-1}+\delta_{rot1})\, sin\,(\alpha_{t-1}+\delta_{rot1}) - \delta_{tras}^2 M(1,1)\, cos\,(\alpha_{t-1}+\delta_{rot1}) \\
&\quad\; sin\,(\alpha_{t-1}+\delta_{rot1}) \\
R(2,2) &= M(2,2)\, sin\,(\alpha_{t-1}+\delta_{rot1})^2 + \delta_{tras}^2 M(1,1)\, cos\,(\alpha_{t-1}+\delta_{rot1})^2 \\
R(2,3) &= \delta_{tras} M(1,1)\, cos\,(\alpha_{t-1}+\delta_{rot1}) \\
R(3,1) &= -\delta_{tras} M(1,1)\, sin\,(\alpha_{t-1}+\delta_{rot1}) \\
R(3,2) &= \delta_{tras} M(1,1)\, cos\,(\alpha_{t-1}+\delta_{rot1}) \\
R(3,3) &= M(3,3) + M(1,1)
\end{aligned} \tag{3.7}$$

Capítulo 4

OV-MCL: localización de Monte Carlo con visión omnidireccional

En este capítulo se presenta el algoritmo de localización OV-MCL. La aproximación se basa en la localización de Monte Carlo (MCL), adaptada para la utilización de visión omnidireccional y un mapa de las luces del techo, lo que permite al robot localizarse en entornos donde las oclusiones son frecuentes y severas. OV-MCL cuenta con dos propiedades que lo hacen especialmente robusto frente al ruido, cambios en el entorno y oclusiones:

– El número de partículas es variable, y depende de lo dispersa que sea la distribución de probabilidad de la posición del robot. Esta dispersión se evalúa mediante KLD. Esto permite trabajar con un número muy bajo de partículas cuando las hipótesis de localización del robot se encuentran concentradas, y por el contrario utilizar un número superior de partículas en situaciones de localización global. La variabilidad del número de partículas contribuye a la ejecución en tiempo real del algoritmo y aumenta la robustez del mismo, puesto que se mantiene siempre elevada la calidad con la que el filtro de partículas aproxima la PDF real, y únicamente se utilizan los recursos computacionales que son estrictamente necesarios.

– Se realizan inserciones de partículas aleatorias en función de la evolución de la PDF a corto y largo plazo, lo que permite corregir divergencias en la aproximación a la PDF real. Esta característica hace que OV-MCL sea muy robusto frente a cambios en el entorno y le permite resolver el problema del *secuestro* del robot.

La estructura del capítulo es la siguiente: en primer lugar se describe en detalle el algoritmo OV-MCL (sec. 4.1); a continuación, se muestran los resultados obtenidos para diferentes experimentos en condiciones reales (sec. 4.2), incluyendo un análisis de la influencia de las oclusiones en la precisión de la localización; finalmente se exponen las conclusiones más relevantes.

4.1. Algoritmo OV-MCL

El algoritmo de localización OV-MCL se basa en la localización de Monte Carlo, i. e., la PDF se representa mediante un conjunto de partículas y cada una de ellas codifica una posible pose del robot. OV-MCL (alg. 4.1) recibe como entradas el conjunto de partículas (Y_{t-1}) que representan la PDF en el instante anterior, el comando de control en el instante actual (u_t), el conjunto de medidas (Z_t) y el mapa (m). Los pasos del algoritmo se pueden agrupar en 4 etapas (fig. 4.1):

- **Monte Carlo (MCL)**. Es el núcleo del algoritmo (alg. 4.1, líneas 3-6). Realiza la actualización de las partículas usando el modelo de movimiento y estima el peso de cada partícula a través del modelo de medida.
- **Número de partículas adaptativo (KLD)**. Esta etapa realiza el cálculo del número de partículas que son necesarias para representar de forma adecuada la PDF de la pose del robot (alg. 4.1, líneas 8-17).
- **Inserción de partículas aleatorias (*Augmented*)**. Realiza la inserción de partículas aleatorias cuando las medidas actuales no se corresponden con las esperadas (alg. 4.1, líneas 7 y 18-25). Esta etapa permite que el algoritmo se recupere ante fallos de localización, como por ejemplo el problema del *secuestro*.
- **Remuestreo**. Se muestrea el conjunto de partículas obtenido en función del peso de las mismas (alg. 4.1, línea 26).

4.1.1. Localización de Monte Carlo

El núcleo de OV-MCL (alg. 4.1, líneas 3-6) repite para cada partícula en Y_{t-1} los siguientes pasos (fig. 4.1):

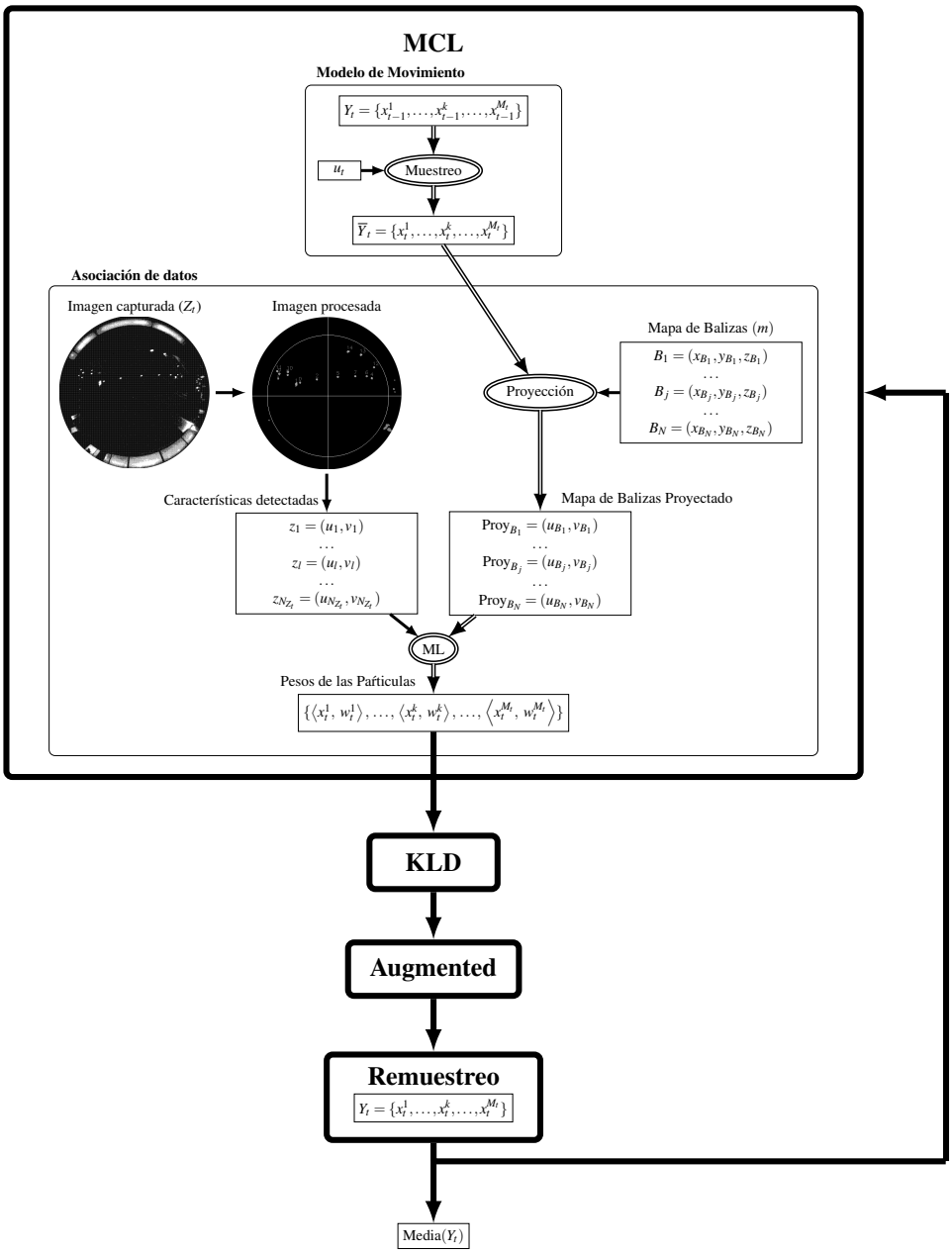


Figura 4.1: Esquema del algoritmo OV-MCL. Se señalan con doble línea las operaciones que se realizan una vez por partícula.

Algoritmo 4.1 OV-MCL $(Y_{t-1},\, u_t,\, Z_t,\, m)$

static: ω_{slow}, ω_{fast}, M_{kld}
$Y_t = \overline{Y}_t = \emptyset$, $M_t = |Y_{t-1}|$, $M_r = 0$, $N_b = 0$, $\omega_{avg} = 0$, $celda = 0$ $(\forall celda \in G)$

for $k = 1$ to N_t **do** ▷ MCL
  $x_t^k = modeloMovimiento\left(u_t,\, x_{t-1}^k\right)$
  $\omega_t^k = modeloMedida\left(x_t^k,\, Z_t,\, m\right)$
  $\overline{Y}_t = \overline{Y}_t \cup \left\langle x_t^k,\, \omega_t^k \right\rangle$

  $\omega_{avg} = \omega_{avg} + \frac{\omega_t^k}{M_t}$ ▷ Augmented

  if $celda\left(x_t^k\right) = 0$ **then** ▷ KLD
    $celda\left(x_t^k\right) = 1$
    $N_b = N_b + 1$
  end if
end for
if $N_b > 1$ **then**
  $M_{kld} = \frac{N_b - 1}{2\varepsilon}\left\{1 - \frac{2}{9(N_b-1)} + \sqrt{\frac{2}{9(N_b-1)}}\, z_{1-\delta}\right\}^3$
else
  $M_{kld} = 1$
end if

$\omega_{slow} = \omega_{slow} + \alpha_{slow}\left(\omega_{avg} - \omega_{slow}\right)$ ▷ Augmented
$\omega_{fast} = \omega_{fast} + \alpha_{fast}\left(\omega_{avg} - \omega_{fast}\right)$
for $i = 1$ to M_{kld} **do**
  if $rand\,() < max\left\{0,\, 1 - \omega_{fast}/\omega_{slow}\right\}$ **then**
    añadir pose aleatoria a Y_t
    $M_r = M_r + 1$
  end if
end for

$Y_t = remuestreoBajaVarianza\left(\overline{Y}_t,\, max\left\{0,\, M_{kld} - M_r\right\}\right) \cup Y_t$ ▷ Remuestreo
return Y_t

1. Se calcula una nueva pose (x_t^k) muestreando el modelo de movimiento (sec. 3.2.1), y teniendo en cuenta la pose previa (x_{t-1}^k) y el comando de movimiento (u_t).

2. Dada la nueva pose, el conjunto de medidas (Z_t) y el mapa (m), la asociación de datos calcula la probabilidad de que el conjunto de medidas se correspondan con esa pose y mapa (asociación de datos, fig. 4.1). Esta probabilidad es el peso de cada partícula (ω_t^k).

3. Finalmente, se añade la partícula al nuevo conjunto de partículas ($\overline{Y}_t$).

Asociación de datos

OV-MCL utiliza un mapa (m) compuesto de un conjunto de balizas (B_j) que se corresponden con las luces del entorno, y de las que se conoce su posición 3D en un sistema de referencia cartesiano (fig. 4.2). La asociación de datos estima la probabilidad de que dada una pose del robot (x_t^k) y un mapa (m), Z_t sea el conjunto de medidas en el instante t. Asumiendo independencia condicional entre las medidas, la probabilidad del conjunto de medidas puede calcularse como:

$$P\left(Z_t \,|\, x_t^k,\, m\right) = \prod_l P\left(z_{t,l} \,|\, x_t^k,\, m\right) \tag{4.1}$$

de tal forma que la probabilidad de cada medida puede estimarse de forma independiente.

El cálculo de la probabilidad del conjunto de medidas exige establecer una correspondencia (asociación) entre cada medida ($z_{t,l}$) y una baliza (B_j). OV-MCL resuelve la asociación de datos, y por tanto la probabilidad de un conjunto de medidas, mediante un algoritmo de máxima probabilidad (ML) (alg. 4.2).

La estimación de $P\left(Z_t \,|\, x_t^k,\, m\right)$ requiere, en primer lugar, de la inicialización de λ_j para todas las balizas (alg. 4.2, línea 3). Esta variable indica si una baliza (B_j) ha sido asociada a alguna medida y es necesaria para evitar la asignación de varias medidas a una baliza. A continuación, para cada medida se inicializa la probabilidad a $P_{falsePos}$, que es la probabilidad de que una medida no se corresponda con ninguna baliza, i.e., que sea un falso positivo (alg. 4.2, línea 6).

Para cada baliza (B_j) no asociada se calcula la probabilidad de asociación con cada medida ($z_{t,l}$) (alg. 4.2, línea 8). Para ello es necesario tener las balizas y las medidas en el mismo sistema de referencia. Puesto que las ecuaciones de la cámara (ec. 3.3) no son invertibles, se ha optado por transformar todas las balizas del mapa proyectándolas sobre la imagen (fig. 4.2). La proyección de una baliza B_j son las coordenadas del píxel en la imagen (u_{B_j}, v_{B_j}) estimadas a partir del modelo de medida (sec. 3.1.2) y las ecuaciones que modelan la cámara (ec. 3.3).

Algoritmo 4.2 *asociacionDatos* $\left(x_t^k,\, Z_t,\, m\right)$.

1: $P\left(Z_t \,|\, x_t^k,\, m\right) = 1$
2: **for** $B_j \in m$ **do**
3: $\quad \lambda_j = 0$
4: **end for**
5: **for** $z_{t,l} \in Z_t$ **do**
6: $\quad P\left(z_{t,l} \,|\, x_t^k,\, m\right) = P_{falsePos}$
7: $\quad$ **for** $B_j \in M$ con $\lambda_j = 0$ **do**
8: $\quad\quad P\left(z_{t,l} \,|\, x_t^k,\, B_j\right) = \text{máx}\left\{1 - \frac{\sqrt{\left(u_l - u_{B_j}\right)^2 + \left(v_l - v_{B_j}\right)^2}}{max_{dist}},\, 0\right\}$
9: $\quad\quad$ **if** $P\left(z_{t,l} \,|\, x_t^k,\, B_j\right) > P\left(z_{t,l} \,|\, x_t^k,\, m\right)$ **then**
10: $\quad\quad\quad P\left(z_{t,l} \,|\, x_t^k,\, m\right) = P\left(z_{t,l} \,|\, x_t^k,\, B_j\right)$
11: $\quad\quad$ **end if**
12: $\quad$ **end for**
13: $\quad P\left(Z_t \,|\, x_t^k,\, m\right) = P\left(Z_t \,|\, x_t^k,\, m\right) \cdot P\left(z_{t,l} \,|\, x_t^k,\, m\right)$
14: $\quad \eta = argmax_j P\left(z_{t,l} \,|\, x_t^k,\, B_j\right)$
15: $\quad \lambda_\eta = 1$
16: **end for**
17: **return** $P\left(Z_t \,|\, x_t^k,\, m\right)$

A partir de $h(B_j, x_t)$ (ec. 3.1) se obtienen θ_j y φ_j , a los que se aplican las ecuaciones de la cámara (ec. 3.3) para obtener la proyección de las balizas (u_{B_j}, v_{B_j}). El resto de componentes de la ecuación (alg. 4.2, línea 8) son las coordenadas en la imagen de la medida $z_{t,l}$ y la constante max_{dist}.

El algoritmo finaliza actualizando la probabilidad de asociación del conjunto de medidas con la máxima probabilidad de asociación de la medida $z_{t,l}$ ($P\left(z_{t,l} \,|\, x_t^k,\, m\right)$) (alg. 4.2, línea 13), y marcando la baliza ganadora (η) para que no pueda ser asociada de nuevo (alg. 4.2, líneas 14-15).

En la parte superior de la figura 4.3(c), se muestra un ejemplo gráfico del proceso de asociación entre las medidas obtenidas en una imagen ($z_{t,l}$) (fig. 4.3(a)) y las proyecciones de las balizas del mapa (B_j) (fig. 4.3(b)). La imagen inferior (fig. 4.3(d)) representa el proceso sobre una imagen real.

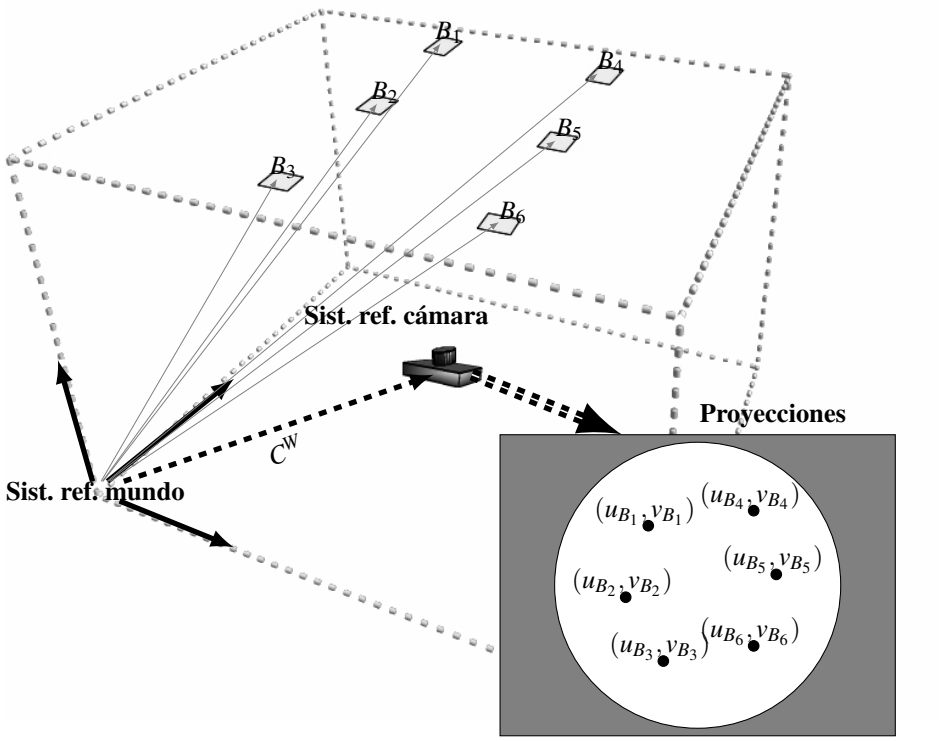
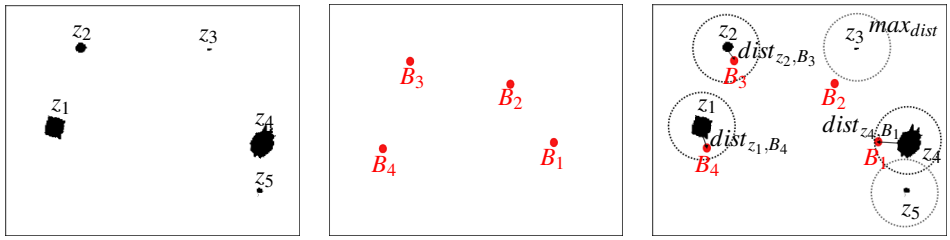


Figura 4.2: Representación gráfica del proceso de generación del mapa proyectado de las balizas.

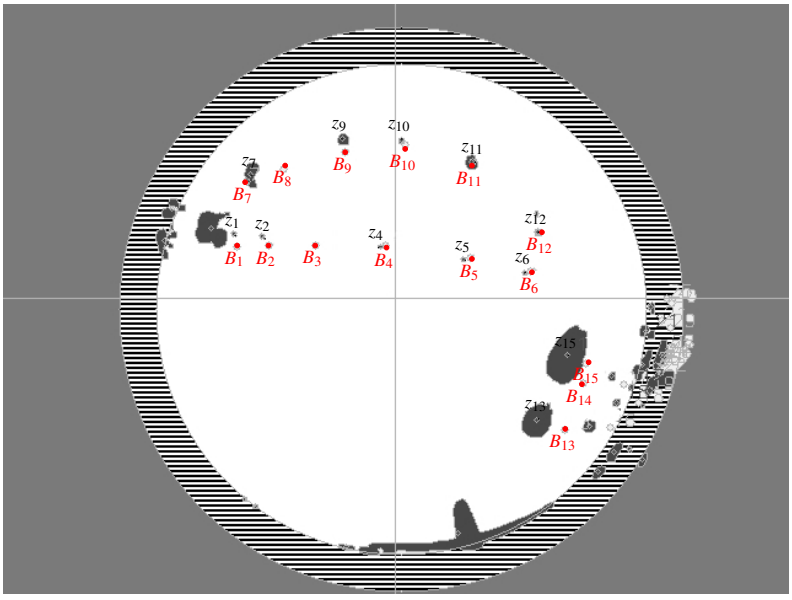
4.1.2. Número de partículas adaptativo

El objetivo de esta etapa es determinar el número de partículas necesario para representar la PDF, utilizando para ello la calidad de la aproximación medida mediante KLD. De esta forma es posible trabajar con un número reducido de partículas cuando la PDF se aproxima a una gaussiana, como en el problema de localización local, o con un número elevado de partículas cuando la PDF es más compleja, como en las primeras etapas de la localización global. Esta aproximación favorece la ejecución del algoritmo en tiempo real e incrementa su robustez en comparación con el algoritmo MCL básico.

Para estimar el límite estadístico del número de partículas en base a KLD se tiene en cuenta el volumen que cubren las partículas en el entorno, para lo que se divide el mapa en celdas. El número de celdas vacías se calcula durante el bucle principal del algoritmo (alg. 4.1,



(a) (b) (c)



(d)

Figura 4.3: Ejemplo de asociación entre medidas y balizas: (a) medidas, (b) balizas proyectadas, (c) asociación de datos y (d) imagen omnidireccional con las balizas proyectadas (rojo) y las medidas (negro).

líneas 8-11): si la celda en la que se encuentra la partícula está vacía, se marca como no vacía y se incrementa el contador de celdas no vacías (N_b).

El número de partículas (M_{kld}) se calcula de manera proporcional al número de celdas no vacías (N_b) y a los límites estadísticos de error ε y δ (alg. 4.1, línea 14) [Fox, 2003], de tal forma que, con probabilidad $1-\delta$, el error entre la distribución real y la muestreada sea menor que ε. $z_{1-\delta}$ representa el cuantil superior $1-\delta$ de la distribución normal. Un valor

elevado de N_b (y por tanto de M_{kld}) indica que las partículas están distribuidas a lo largo del espacio de estados, i. e., hay una alta incertidumbre en la pose del robot. Para representar esta situación y mantener el seguimiento de todas las posibles poses, se necesita un mayor número de partículas. Por otro lado, cuando N_b es bajo las partículas se concentran alrededor de unas pocas regiones del espacio de estados, y por tanto la PDF se puede representar con un número pequeño de partículas.

Uno de los factores que puede incrementar la incertidumbre durante la localización es la existencia de simetrías en el entorno. Con un número fijo de partículas el conjunto se divide entre las diferentes hipótesis de localización. La concentración de partículas en cada grupo podría ser baja, y en consecuencia se producirían fallos en la localización al eliminar partículas que representen poses correctas. Por el contrario, al utilizar un número adaptativo de partículas su número se incrementa dinámicamente al aumentar la incertidumbre en la pose del robot, i. e., existen varias buenas poses candidatas debido a las simetrías del entorno. Una vez que el robot se mueve y las ambigüedades desaparecen se reduce la incertidumbre, de tal forma que el número de partículas que son necesarias es también menor.

4.1.3. Inserción de partículas aleatorias

La inserción de partículas aleatorias tiene como objetivo que OV-MCL se pueda recuperar de fallos importantes en la localización, siendo un ejemplo extremo de esta situación el problema del *secuestro* del robot. La inserción de las partículas está basada en el algoritmo Augmented-MCL [Gutmann y Fox, 2002], que ha sido modificado para adaptarlo a OV-MCL.

La idea básica es que el conjunto de partículas está representando correctamente la PDF real cuando el peso medio de las partículas es alto. Para ello, en primer lugar se determina el peso medio de las partículas en el instante actual, ω_{avg} (alg. 4.1, línea 7). ω_{fast} y ω_{slow} se corresponden respectivamente con las medias de corto y largo plazo de los pesos de las partículas, y son actualizadas a partir de su diferencia con la media de los pesos actuales (ω_{avg}) y los parámetros α_{fast} y α_{slow} (alg. 4.1, líneas 18 -19) . Estos parámetros pueden ser considerados como velocidades de actualización de las medias y deben cumplir que $0 \leq \alpha_{slow} \ll \alpha_{fast}$. El número de partículas generado aleatoriamente (M_r) se corresponde con un porcentaje aproximado de $1 - \omega_{fast}/\omega_{slow}$ del total de partículas (alg. 4.1, líneas 20 -25).

Si ω_{fast} y ω_{slow} son muy similares, entonces no se añaden partículas aleatorias. Esto indica que las medidas son las esperadas para las poses representadas por las partículas. Por otra parte, si $\omega_{fast} < \omega_{slow}$ la PDF no se corresponde con las medidas y se deberían añadir partí-

culas aleatorias. Cuanto menor sea el cociente entre la media a corto plazo y la media a largo plazo, mayor será el número de partículas aleatorias que se añadan. Por ejemplo, cuando se produce un *secuestro* del robot, en las primeras iteraciones ω_{fast} decrece drásticamente mientras que ω_{slow} decrece suavemente, por lo que el número de partículas que serán generadas aleatoriamente en las primeras iteraciones será alto.

4.1.4. Remuestreo de baja varianza

La última fase de OV-MCL es el remuestreo del conjunto de partículas ($\overline{Y}_t$), obteniendo $M_{kld} - M_r$ nuevas partículas. Los algoritmos de remuestreo convencionales seleccionan las partículas de manera aleatoria y proporcional a los pesos de las mismas. Sin embargo, y debido a la total aleatoriedad en la selección, puntualmente se pueden producir dos problemas: i) que haya diferencias significativas entre la distribución real y la muestra; ii) pérdida de diversidad de la muestra. Para solucionar estos problemas se ha optado por utilizar el algoritmo de remuestreo de baja varianza (alg. 4.3) [Thrun y col., 2003].

La ventaja del muestreo de baja varianza es doble. En primer lugar, cubre el espacio de muestras de un modo sistemático, realizando siempre un ciclo a través de todas las partículas en lugar de elegirlas aleatoriamente de modo independiente (fig. 4.4). En segundo lugar, si todas las muestras tienen el mismo factor de importancia (peso) el resultado de muestrear el conjunto Y_t es equivalente a Y_t, por lo que no se pierde ninguna muestra si todas las partículas son equiprobables.

Algoritmo 4.3 *RemuestreoBajaVarianza*(Y_t, ω_t).

1: $\overline{Y}_t = \emptyset$
2: $r = rand(0; M_t^{-1})$
3: $c = \omega_t^1$
4: $i = 1$
5: **for** $k = 1$ to M_t **do**
6: $\quad U = r + (k-1)M_t^{-1}$
7: $\quad$ **while** $U > c$ **do**
8: $\quad\quad i = i + 1$
9: $\quad\quad c = c + \omega_t^i$
10: $\quad$ **end while**
11: $\quad$ Añadir x_t^i
12: **end for**
13: **return** $\overline{Y}_t$

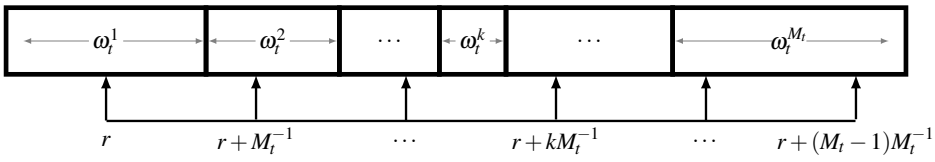


Figura 4.4: Esquema de selección de partículas del remuestreo de baja varianza.

4.2. Resultados experimentales

El algoritmo de localización OV-MCL ha sido testeado con un robot Pioneer 2-AT (fig. 3.11(b)) en la planta baja del Museo Domus de A Coruña (fig. 4.5 y 4.6). El entorno tiene unas dimensiones de $27 \times 7\,m^2$ y las imágenes han sido obtenidas a una frecuencia de $2\,Hz$.

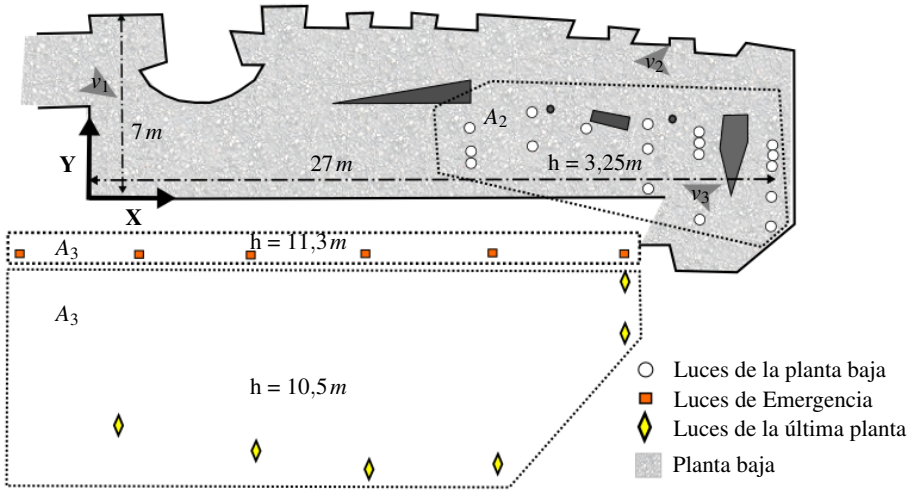
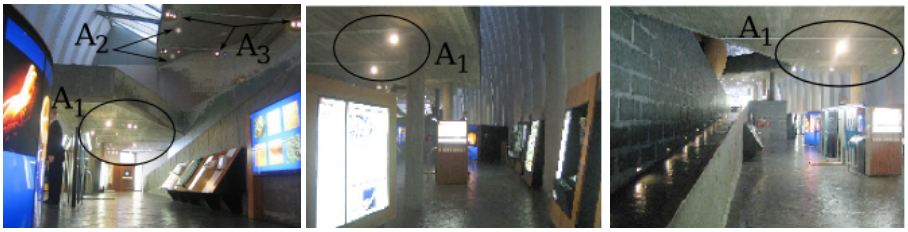


Figura 4.5: Mapa de la sala de exposiciones de la planta baja del Museo Domus de A Coruña.

Para comparar nuestros resultados se ha hecho una estimación de la posición real del robot en cada instante con un sensor láser situado sobre el robot. Estas medidas fueron obtenidas sin presencia de gente en el entorno, ya que bajo oclusiones severas las medidas del láser no permiten realizar una localización precisa. Esta situación es muy típica en el Museo Domus, donde el robot está normalmente rodeado por gente incluso cuando se desplaza.



(a) v_1 (fig 4.5). (b) v_2 (fig 4.5). (c) v_3 (fig 4.5).

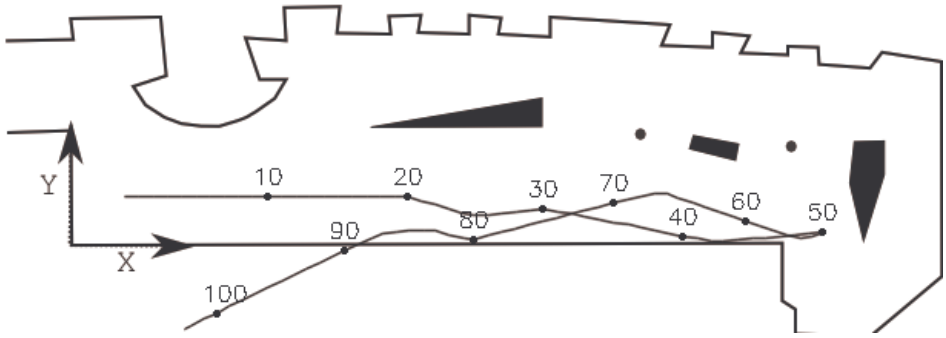
Figura 4.6: Fotos de diferentes localizaciones de la planta baja de la sala de exposiciones del Museo Domus.

El mapa de balizas usado en estos experimentos consiste en un conjunto de las luces del museo. La figura 4.5 muestra el plano del suelo de la planta baja del museo (donde se han realizado los experimentos), y los tres tipos de luces que se han usado para la localización: luces de la planta baja (A_1), luces de emergencia (A_2) y luces del último piso del museo (A_3). En la misma figura también se muestra la media de la altura (h) a la que se encuentra cada subconjunto de luces.

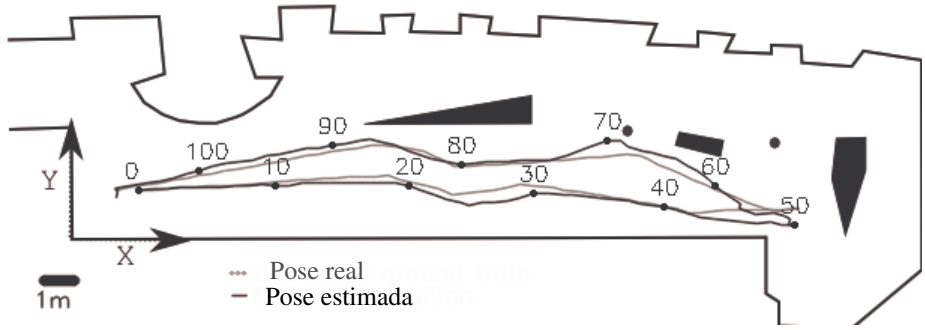
La figura 4.6 recoge diferentes fotografías del entorno donde se observan algunas de las luces pertenecientes a los subconjuntos usados. Como se puede apreciar en las imágenes tomadas desde la planta baja, es posible ver varios techos de otras salas a distintas alturas. También se puede observar como algunas luces no son visibles desde ciertas áreas de la sala debido a las oclusiones de los techos a diferentes alturas. Además, hay que destacar que existen numerosas luces en el entorno que no son consideradas balizas y que, por lo tanto, deben ser ignoradas durante el proceso de asociación de datos. Por último, otra característica a destacar del entorno es el suelo irregular que provoca que las ruedas del robot estén a diferentes alturas, produciendo alabeos y cabeceos que se ven amplificados en la cámara situada en la parte superior de la estructura (señalada con un círculo en la figura 3.11(b)).

Los experimentos han sido diseñados para probar que el algoritmo de localización es capaz de estimar la posición del robot móvil de una forma fiable en diferentes situaciones (localización local, localización global y *secuestro*). Además, se ha realizado un análisis del comportamiento del algoritmo bajo oclusiones severas y continuas. Todos los experimentos han sido ejecutados con los siguientes valores de los parámetros del algoritmo: $\varepsilon = 0,20$, $z_{1-\delta} = 0,99$, $\alpha_{slow} = 0,1$, $\alpha_{fast} = 0,5$ y $max_{dist} = 30$.

4.2.1. Localización local



(a) Odometría.



(b) Poses real y estimada por OV-MCL.

Figura 4.7: Trayectoria del robot para el experimento de localización local.

El primer conjunto de experimentos se ha realizado para mostrar la capacidad del algoritmo para seguir la posición del robot a partir de una posición inicial conocida con poca incertidumbre mientras se mueve a una velocidad de $40 cm/s$. En la figura 4.7[1] se muestra la trayectoria estimada por OV-MCL y la trayectoria real estimada por el láser para uno de los experimentos realizados. Los círculos negros indican la posición cada diez iteraciones del

[1]El vídeo asociado está disponible en `http://persoal.citius.usc.es/cristina.gamallo/Videos/OV-MCL_local.mp4`.

algoritmo[2]. La distancia recorrida fue superior a $45\,m$ y se procesaron 110 imágenes, siendo la distancia media entre dos imágenes consecutivas de $0,4\,m$. Por otra parte, la variación en ángulo puede ser mucho más brusca, llegando a cambios de 180° en tres imágenes.

El error en posición (fig. 4.8) ha sido calculado como la distancia euclídea entre la pose estimada por OV-MCL y la estimada por el láser en condiciones ideales. El máximo error en posición en el experimento fue de $0,75\,m$ y la media de $0,29\,m$. Aunque el error máximo en orientación fue de 23°, la media tan solo alcanzó 3° (fig. 4.8). Esta precisión angular es una de las principales ventajas del uso de una cámara omnidireccional para la localización. El suelo irregular del museo afecta muy negativamente a los errores de posición. A pesar de ello, se puede concluir que el sistema de localización estima la pose del robot en un entorno complejo y durante trayectorias largas con un alto grado de precisión para tareas de navegación.

La figura 4.8 muestra también como se adapta el tamaño del conjunto de partículas gracias a la fase KLD de OV-MCL. Inicialmente se parte de un conjunto de 3.000 partículas generadas a partir de una distribución gaussiana centrada en la posición inicial del robot. En la etapa inicial, el tamaño de la muestra decrece rápidamente hasta alcanzar un mínimo de 100 partículas debido a que el error de localización es bajo. Sin embargo, cuando la calidad de la distribución de las muestras decrece, el número de partículas se incrementa hasta llegar a 1.000 muestras. Así ocurre por ejemplo cuando el robot realiza un giro rápido. Esto refleja la eficiencia de OV-MCL, ya que el tamaño de la muestra (y por lo tanto el tiempo de ejecución) depende solo de la calidad de la PDF. El tiempo de ejecución de OV-MCL se muestra en la figura 4.8; estos tiempos han sido medidos sobre un procesador Intel Pentium 4 CPU $3{,}06\,GHz$. El tiempo de procesado para cada imagen es prácticamente constante, excepto cuando se trata de una imagen saturada, en cuyo caso se ejecuta un segundo proceso de detección con un umbral más alto (sec. 3.1.1.1).

El tiempo medio de ejecución de cada iteración ha sido de $100\,ms$, mientras que el tiempo máximo ha sido de $250\,ms$. Por lo tanto, el algoritmo OV-MCL se puede ejecutar en tiempo real.

4.2.2. Localización global

El siguiente conjunto de experimentos se realizó para comprobar el comportamiento de OV-MCL en localización global. El conjunto de muestras se inicializó con 3.000 partículas

[2]Cada iteración del algoritmo se corresponde con la adquisición de una nueva imagen. La frecuencia de ejecución es de $2\,Hz$.

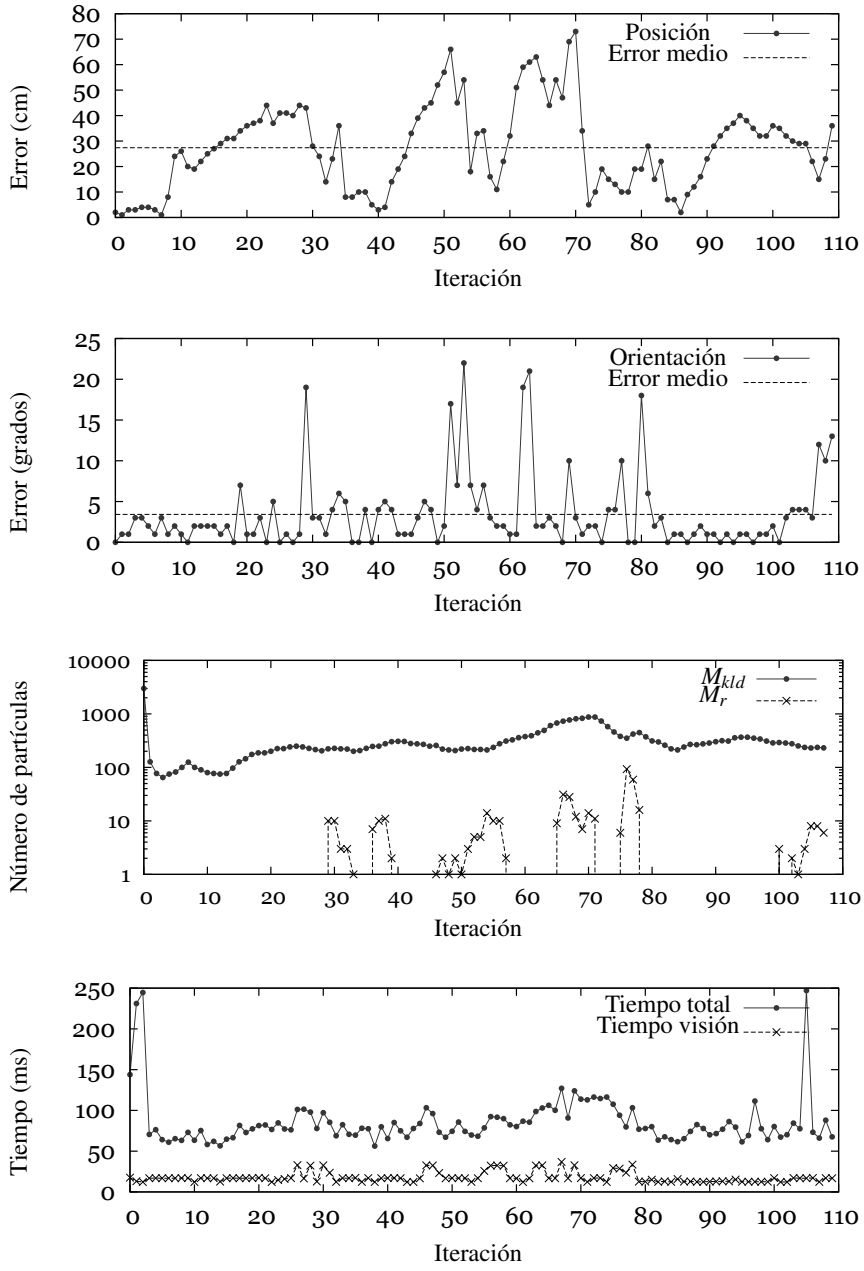


Figura 4.8: Resultados de un experimento típico de localización local.

distribuidas uniformemente en todo el entorno. La figura 4.9[3] muestra la trayectoria estimada por OV-MCL. Como OV-MCL determina la pose del robot con una media pesada del conjunto de la muestras, al principio la posición estimada va a estar siempre próxima al centro del mapa. Tras las primeras iteraciones el sistema consigue determinar la posición del robot y a partir de ahí seguirla de manera fiable.

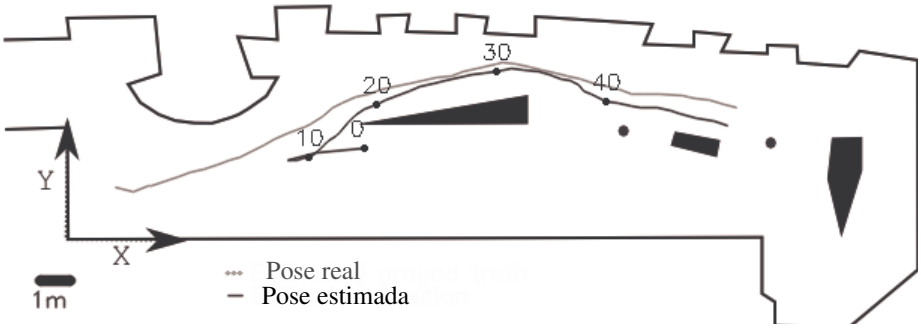
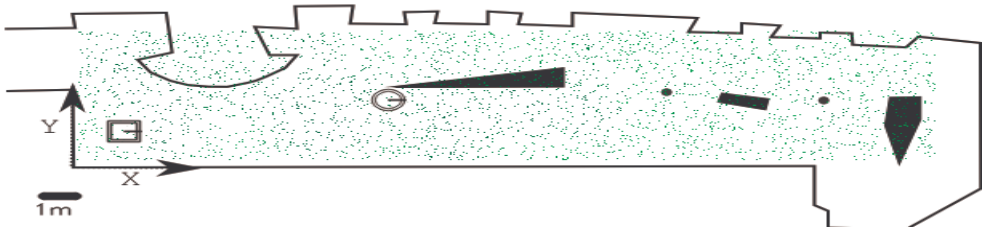


Figura 4.9: Trayectoria estimada para la localización global.

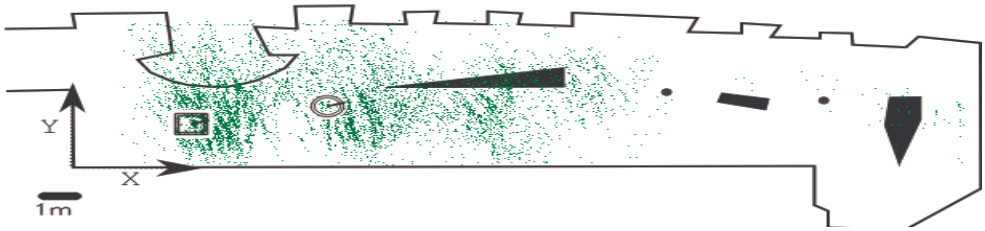
La secuencia de la figura 4.10 muestra como converge el conjunto de muestras durante el proceso de localización global. Inicialmente las partículas están distribuidas uniformemente sobre el entorno, y a medida que se realizan iteraciones el conjunto de muestras converge progresivamente hacia la pose real del robot. Hay que destacar que el mapa (de luces) utilizado tiene simetrías, lo que provoca que el conjunto de partículas represente una PDF con aspecto multimodal, como se aprecia en las figuras 4.10(b) y 4.10(c). Por esta razón, OV-MCL requiere de aproximadamente 12-15 iteraciones para converger (fig. 4.10(d)) a la posición correcta del robot, aun cuando la orientación correcta se alcanza en únicamente 2-5 iteraciones.

La figura 4.11 muestra las gráficas de error en posición y ángulo, el tiempo de ejecución y el número de partículas para cada iteración. En ellas se puede observar que, al principio, el error en posición es alto y por tanto el número de partículas también es elevado, lo que incrementa el tiempo de ejecución (alcanzando picos de hasta 400 *ms* cuando OV-MCL opera con casi 10000 partículas). Cuando el robot se localiza correctamente, todos los valores se

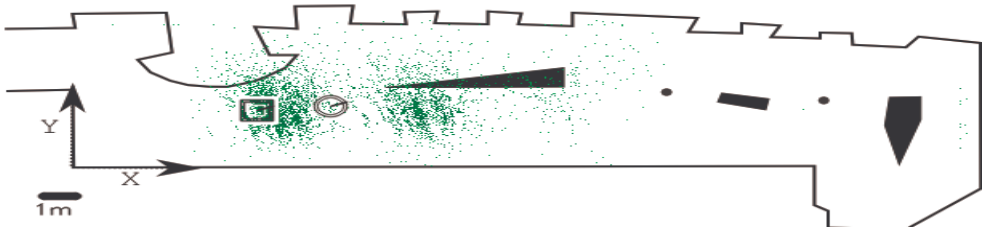
[3]El vídeo asociado está disponible en `http://persoal.citius.usc.es/cristina.gamallo/Videos/OV-MCL_global.mp4`.



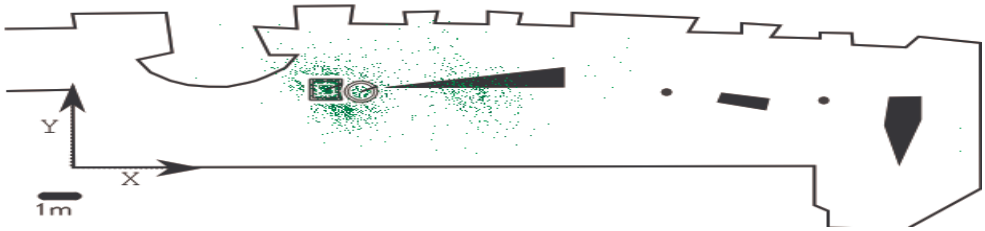
(a) Iteración inicial.



(b) Iteración 5.



(c) Iteración 10.



(d) Iteración 15

Figura 4.10: Evolución del conjunto de muestras durante la localización global. La pose real se representa con un cuadrado, la estimada por OV-MCL con un círculo y las partículas con puntos (cuanto más oscuros, mayor peso).

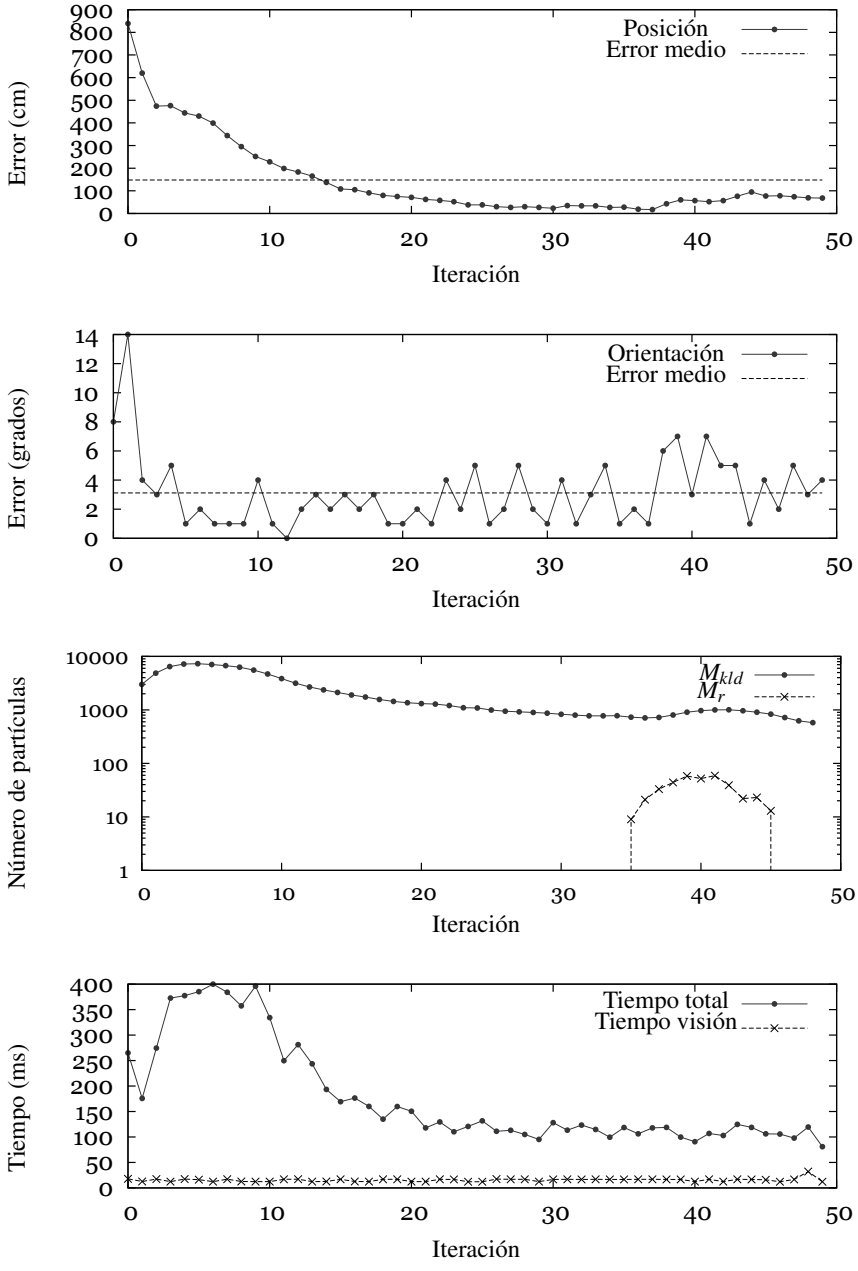


Figura 4.11: Resultados de un experimento típico de localización global.

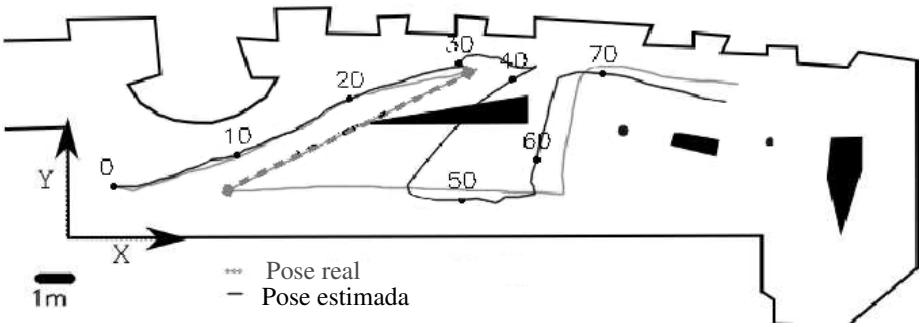


Figura 4.12: Trayectoria seguida por el robot durante el experimento del *secuestro*.

reducen y el problema se convierte en una situación de localización local. En todas las pruebas de localización global ejecutadas se han obtenido resultados similares.

4.2.3. *Secuestro* del robot

El tercer grupo de experimentos se ha llevado a cabo para demostrar que OV-MCL es capaz de recuperarse ante fallos importantes de localización. El conjunto de muestras se inicializó igual que en el experimento de localización local, pero tras haber procesado 30 imágenes se desplazó el robot a una posición totalmente distinta sin realizar ninguna modificación de la odometría (*secuestro* del robot).

La trayectoria estimada y el error para cada una de las poses se muestran en las figuras 4.12 y 4.13. En las gráficas se puede observar claramente el momento en el que se produce el *secuestro*: en la iteración 30 hay un incremento brusco en el error tanto en posición como en orientación. Al igual que en los experimento de localización global, el sistema requiere de 12-15 iteraciones tras el *secuestro* para recuperar la posición correcta y solamente 2-5 para establecer la orientación real.

El tamaño del conjunto de partículas, calculado en la etapa KLD, tiene un comportamiento similar al del experimento de localización local: decrece rápidamente en las primeras iteraciones, y a continuación crece lentamente debido a las simetrías del entorno. Cuando se produce el *secuestro*, el peso de las partículas se reduce al no corresponderse las medidas con las esperadas, y en consecuencia el número de partículas aleatorias (M_r) se incrementa (fig. 4.13). En ese instante, la evolución del tamaño del conjunto de partículas es similar al del experimento

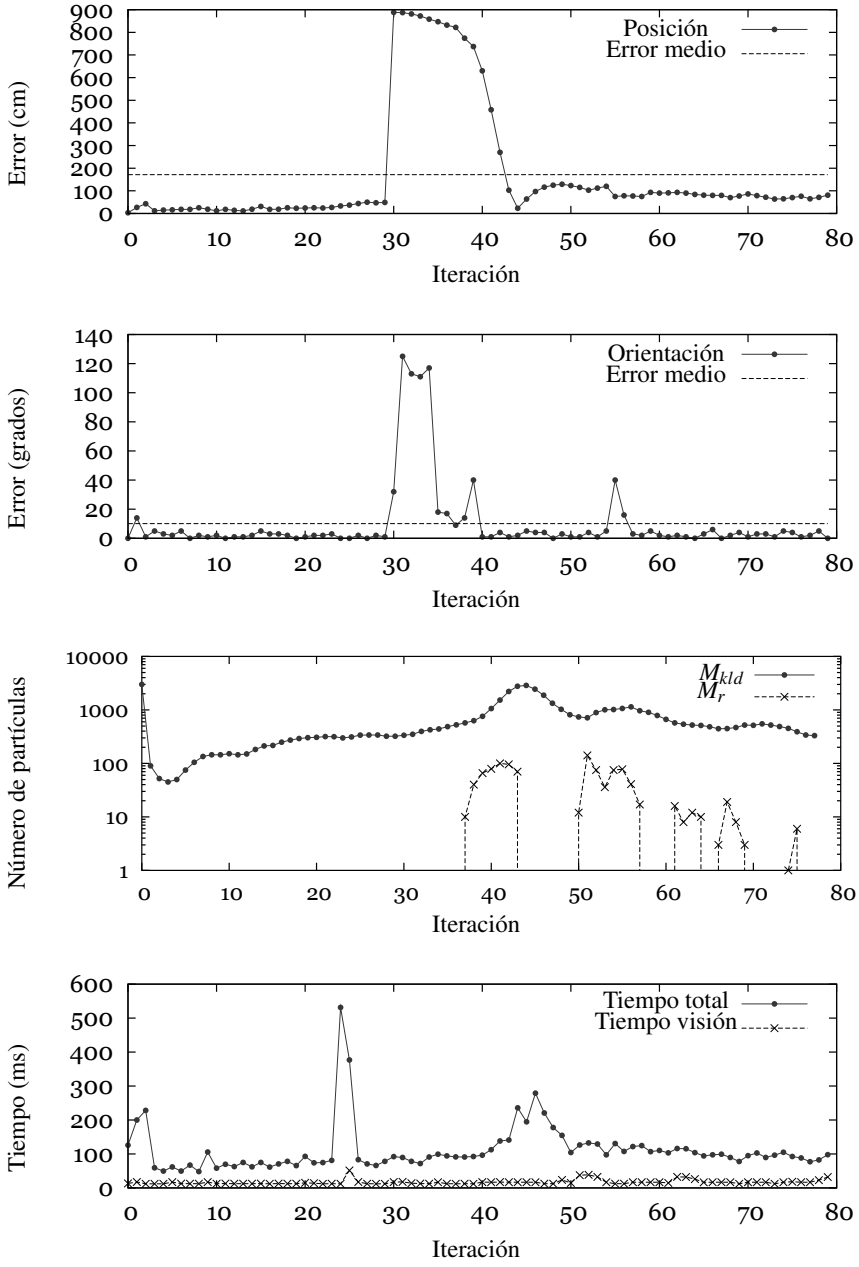
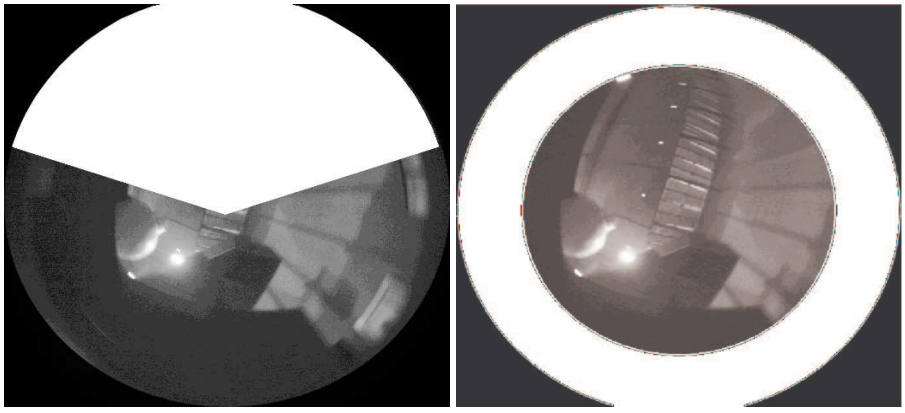


Figura 4.13: Resultados de un experimento típico de *secuestro*.

de localización global, ya que las situaciones son muy parecidas. Inicialmente, el tamaño se incrementa hasta que se establece la posición y orientación correcta del robot, y a partir de ese momento el conjunto de partículas vuelve a reducir su tamaño. Se ha repetido esta clase de experimento con diferentes tipos de trayectorias y en todos los casos el sistema ha sido capaz de recuperarse de los fallos importantes de localización.

4.2.4. Oclusiones

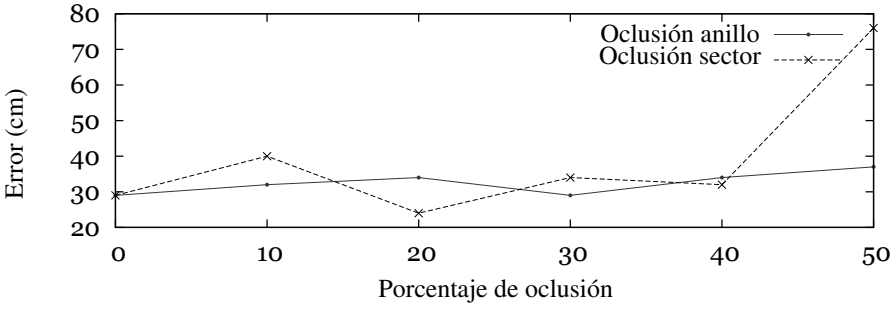
OV-MCL ha sido diseñado para operar bajo oclusiones continuas y severas. Una situación típica en la que se produce este tipo de oclusiones es cuando el robot se mueve completamente rodeado de gente durante un largo periodo de tiempo; en estas condiciones, los métodos de localización tradicionales suelen fallar. Aún cuando OV-MCL utiliza una cámara omnidireccional y las luces del entorno, todavía se pueden producir oclusiones cuando las personas que rodean al robot están muy cercanas (fig. 0.1).



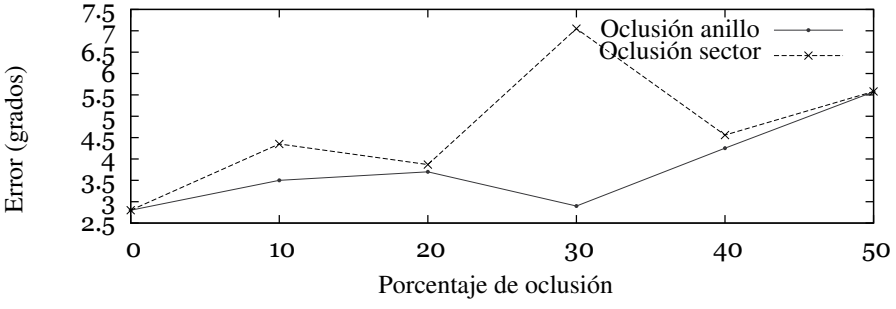
(a) Sector circular del 40 % (b) Anillo exterior del 40 %

Figura 4.14: Diferentes tipos de máscaras utilizados en los experimentos con oclusiones continuas.

Para analizar el comportamiento de OV-MCL bajo estas condiciones se ha diseñado un conjunto de experimentos con diferentes tipos y grados de oclusión. Las oclusiones se han generado mediante la superposición de máscaras sobre las imágenes de forma continua (durante todo el experimento). Se han usado dos tipos de máscaras: en forma de sector circular (fig. 4.14(a)) y en forma de anillo (fig. 4.14(b)).



(a)



(b)

Figura 4.15: Error con oclusiones continuas de diferentes grados.

La figura 4.15 muestra la media del error en posición y ángulo a lo largo de la trayectoria de la figura 4.7(b). Para la máscara de anillo, OV-MCL puede trabajar con oclusiones de hasta el 50 % sin que se vea afectado de forma significativa (en comparación con una situación sin oclusión) el error en posición o en ángulo. Para la máscara de sector ocurre lo mismo pero hasta un porcentaje de oclusión del 40 %. Ambos tipos de oclusiones, pero particularmente las de sector, son un reto importante para la asociación de datos, especialmente cuando estas oclusiones son continuas.

4.3. Conclusiones

En este capítulo se ha presentado el algoritmo de localización OV-MCL. La estructura del mismo se basa en la localización de Monte Carlo, pero modificada y extendida para trabajar con cámaras omnidireccionales, utilizar un mapa de luces, y hacer la localización robusta frente a oclusiones severas y continuas y a cambios en el entorno. Las principales contribuciones de OV-MCL son:

– El modelo de medida que permite, a partir de una imagen de una cámara omnidireccional, determinar la probabilidad de que el conjunto de medidas obtenidas de la imagen se corresponda con el mapa de luces para una pose del robot específica. Para ello se ha realizado una proyección del mapa de balizas y se han asociado las medidas a dichas proyecciones utilizando un algoritmo de asociación de datos basado en máxima probabilidad.

– El tamaño adaptativo del conjunto de partículas, que depende de la diferencia entre la PDF real y la aproximada por las partículas, medida a través de KLD. El efecto sobre el número de partículas es que cuando estas se encuentran muy dispersas el tamaño del conjunto aumenta para mejorar la calidad de la aproximación. Por el contrario, cuando las partículas están muy concentradas el tamaño del conjunto disminuye, ya que la PDF real puede ser aproximada con pocas partículas. La ventaja del número de partículas variable es doble:

  - Se optimiza el uso de recursos computacionales, mientras que si el tamaño fuese fijo en algunos instantes se estarían desaprovechando partículas. Además, no es necesario definir el número de partículas, que es un parámetro crítico de MCL por lo que, en general, se suele sobrestimar para que el sistema no falle.
  - Se aumenta la robustez de la localización, pues en situaciones con múltiples hipótesis con probabilidades similares se incrementa el número de partículas, permitiendo retrasar la eliminación de partículas prometedoras hasta que lleguen nuevas medidas que corroboren o descarten las hipótesis.

– La inserción de partículas aleatorias mediante el análisis de la evolución a corto y largo plazo de los pesos medios de las partículas. Un peso medio elevado indica que un buen número de partículas representan poses desde las que el mapa está en concordancia con

las medidas recibidas. Por el contrario, pesos medios bajos son un indicio de que las medidas recibidas no coinciden con el mapa para las poses codificadas en las partículas. Cuando la media de pesos de corto plazo cae de forma más brusca que la de largo plazo se produce la inyección de partículas aleatorias en un porcentaje proporcional a dicha caída. El resultado es que OV-MCL es más robusto frente a cambios en el entorno y oclusiones severas y continuas, siendo capaz de recuperarse incluso ante situaciones de *secuestro* del robot.

OV-MCL ha sido testeado en un robot *Pioneer 2-AT* para los distintos problemas de localización (local, global y *secuestro*), así como bajo oclusiones severas y continuas. Todas las pruebas presentadas han sido realizadas en el Museo Domus de A Coruña, un entorno de alta complejidad por la irregularidad del suelo y las oclusiones propias del entorno (objetos expuestos y techos a varios niveles), lo que provoca que tanto la odometría como las imágenes presenten un alto nivel de ruido.

Se ha realizado un amplio análisis de estos experimentos, incluyendo el error en posición y ángulo respecto al estimado por medio de un sensor láser, el tiempo de ejecución y la evolución del número de partículas. En todos los casos los resultados reflejan una gran robustez y precisión del algoritmo (error medio de $0,29\,m$ en localización local), destacando que el robot solo necesita unos pocos segundos (imágenes) para recuperarse de un *secuestro* a pesar de que el mapa de luces presenta numerosas simetrías. Por otra parte, se ha hecho un estudio en profundidad de la capacidad de OV-MCL para operar bajo oclusiones severas y continuas; el sistema opera con normalidad frente a niveles de oclusión de hasta el $50\,\%$. En definitiva, se puede concluir que OV-MCL es un algoritmo preciso y robusto para la localización en entornos reales complejos, incluso en condiciones de oclusiones severas y continuas que afectan negativamente a la asociación de datos.

Capítulo 5

OV-FastSLAM: SLAM con omnivisión bajo oclusiones severas

En este capítulo se describe OV-FastSLAM, un algoritmo de SLAM para cámaras omnidireccionales operando en condiciones de oclusiones severas. La propuesta se basa en el algoritmo FastSLAM 2.0 [Montemerlo y col., 2002] e incluye un nuevo método de asociación de datos que es global, jerárquico y mantiene múltiples hipótesis por partícula. OV-FastSLAM posee dos características que lo hacen especialmente preciso y robusto con imágenes omnidireccionales en entornos con oclusiones severas:

- La asociación de datos jerárquica. Se han definido dos tipos de balizas: las estándar (o simplemente balizas) y las candidatas (es decir, las que aún no se han podido inicializar). En una primera etapa se realiza la asociación de las medidas con las balizas mediante el algoritmo de Murty [Murty, 1986], que obtiene las n mejores asociaciones en tiempo polinómico. En una segunda fase se lleva a cabo la asociación de las restantes medidas con las balizas candidatas a través del método Húngaro [Kuhn, 1955]. De esta forma se prioriza la asociación de balizas al ser más fiables que las balizas candidatas. Como resultado de esta asociación jerárquica cada partícula mantiene varias hipótesis, permitiendo retardar la elección de la mejor asociación hasta que se han incorporado suficientes medidas y es posible discernir entre aquellas hipótesis con probabilidades iniciales muy similares.

- La inicialización de las balizas. Se realiza con un método retardado, puesto que la cámara omnidireccional (como cualquier cámara monocular) solo proporciona información

de la recta 3D sobre la que se encuentra la baliza, pero no su posición 3D. El método parte del conjunto de medidas asignadas a lo largo del tiempo por la asociación de datos jerárquica y evalúa la probabilidad de los distintos puntos de cruce, permitiendo eliminar balizas candidatas cuando el mejor punto de cruce no tiene una probabilidad alta con el conjunto de medidas asociadas.

Este capítulo se organiza de la siguiente forma: en primer lugar se presentan el modelo inverso de la cámara y el modelo de medida utilizados, y a continuación se describe el tipo de mapa que genera OV-FastSLAM. En la sección 5.3 se realiza una descripción detallada del algoritmo OV-FastSLAM, y posteriormente se muestran los resultados experimentales (sec. 5.4). Finalmente, en la sección 5.5 se recogen las conclusiones más destacadas de OV-FastSLAM.

5.1. Modelo inverso de la cámara y modelo de medida

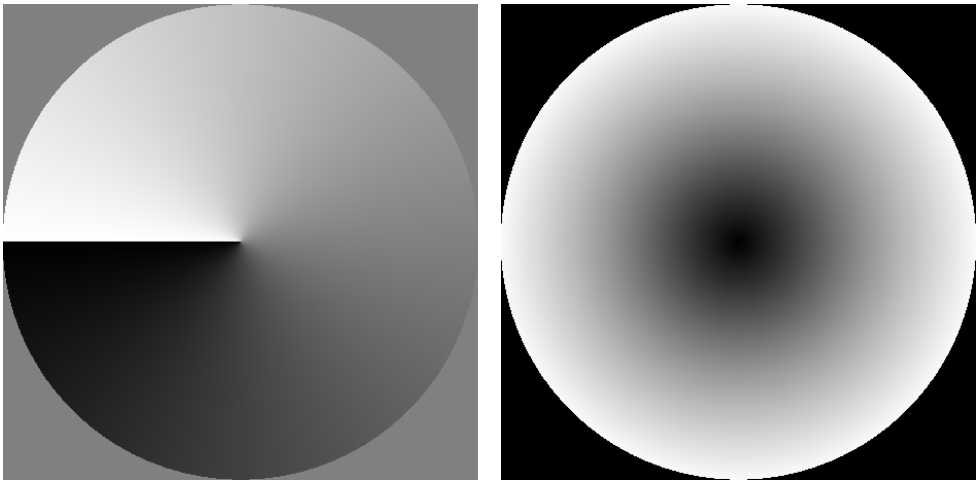
5.1.1. Modelo inverso de la cámara

Las características —luces del entorno— se extraen de las imágenes omnidireccionales siguiendo el proceso descrito en la sección 3.1.1. La salida del proceso de extracción de características es una lista de coordenadas de píxeles (u_l, v_l) que representan el centroide de cada característica l. Esta lista debe ser transformada en un conjunto de medidas (Z_t), donde cada medida $(z_{t,l})$ viene dada por los ángulos de azimut y elevación $(\varphi_{t,l},\, \theta_{t,l})$.

El modelo de la cámara (sec. 3.1.3) describe cómo se transforma un punto 3D en un píxel de la imagen 2D. Sin embargo, la obtención de las medidas requiere el modelo inverso de la cámara, i. e., dado un píxel el modelo inverso devuelve las coordenadas del punto en 3D en el sistema de referencia absoluto —una línea en el espacio tridimensional para sensores que solo proporcionan orientaciones—.

Desafortunadamente, las ecuaciones que definen el modelo de la cámara no son invertibles, por lo que ha sido necesario definir una tabla de referencia: dadas las coordenadas de un píxel, la tabla de referencia proporciona los valores de φ y θ. La figura 5.1 muestra la representación gráfica de dicha tabla de referencia. El proceso de construcción de la tabla es el siguiente:

1. Se muestrean los valores de φ y θ con precisiones δ_φ y δ_θ. Los valores de las coordenadas del píxel se obtienen mediante las ecuaciones 3.3.



(a) Valores de φ (blanco= π, negro= $-\pi$) (b) Valores de θ (blanco= $\pi/2$, negro= 0)

Figura 5.1: Representación gráfica de los valores de φ y θ proporcionados por la tabla de referencia para cada píxel de la imagen omnidireccional.

2. Para cada píxel se obtienen los valores máximos y mínimos de φ y θ, ya que un rango de valores podría corresponderse con el mismo píxel. Se almacena el valor medio de φ y θ para cada píxel.

5.1.2. Modelo de medida

El modelo de medida utilizado se corresponde con el descrito en la sección 3.1.2, que transforma una posición 3D de una baliza en los ángulos $(\varphi_{t,l}, \theta_{t,l})$. Este modelo es genérico para cualquier tipo de sensor, sin embargo, el modelo de ruido de la medida sí que es específico para cada sensor. El modelo de ruido del sensor es necesario para aplicar el algoritmo FastSLAM (ver apéndice A).

En el caso de la cámara omnidireccional utilizada en esta tesis, la matriz de covarianza del ruido para la l-ésima medida se define como:

$$Q_l = \begin{pmatrix} Q_l^{\varphi} & 0 \\ 0 & Q_l^{\theta} \end{pmatrix} = \begin{pmatrix} \left(\frac{\Delta_{np}}{r}\right)^2 & 0 \\ 0 & \left(\frac{\pi\Delta_{np}}{2r_{max}}\right)^2 \end{pmatrix} \tag{5.1}$$

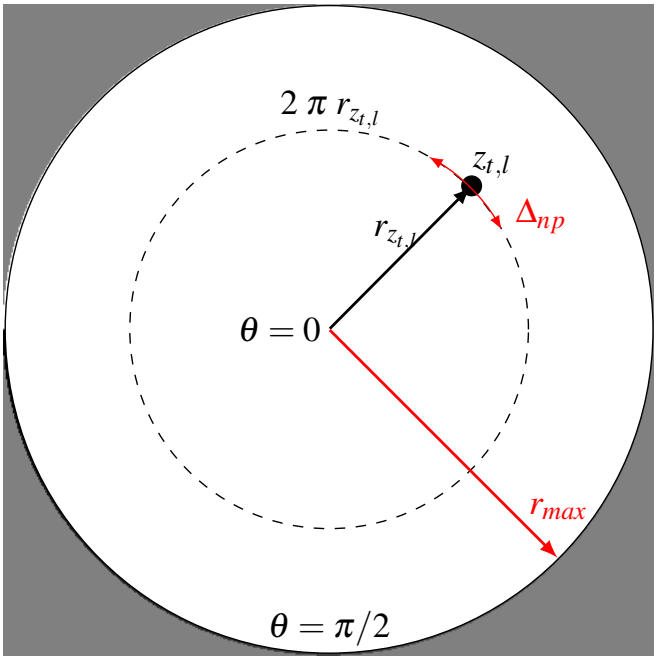
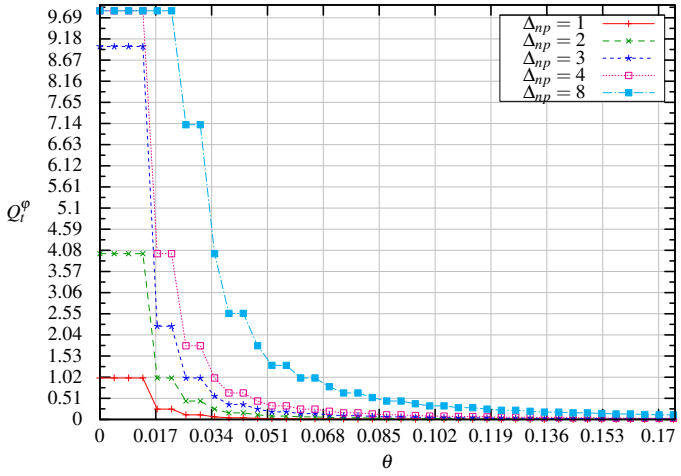


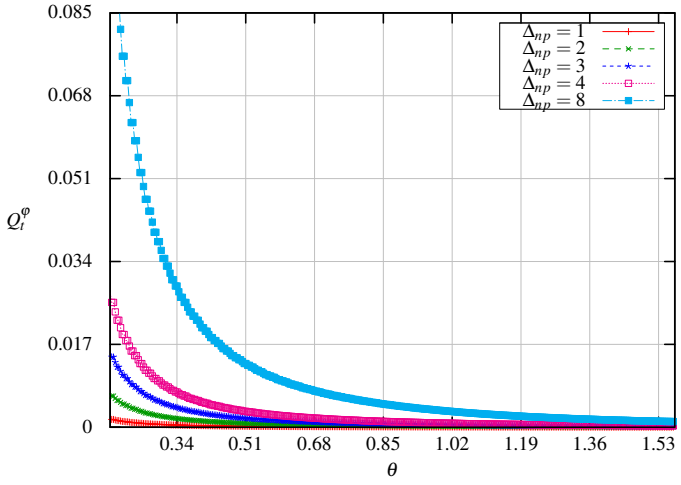
Figura 5.2: Estimación de Q_l

donde Δ_{np} representa la incertidumbre (en píxeles) del proceso de extracción de las características y r indica la distancia (en píxeles) desde el centro de la imagen a la característica, siendo r_{max} su valor máximo (fig. 5.2).

Q_l^{φ} se obtiene como el cuadrado del ángulo de un arco con longitud Δ_{np} y radio r. Por tanto, el valor de Q_l^{φ} depende de θ. La figura 5.3 muestra esta dependencia: el mayor error se produce en el centro de la imagen ($\theta = 0$), ya que en ese punto no hay información sobre el valor de φ (cualquier valor de φ se corresponde con ese punto). Por otra parte, en el extremo de la imagen ($\theta = \frac{\pi}{2}$) los errores en la posición de las características tienen poca influencia en el valor de φ. Por último, Q_l^{θ} se define como el cuadrado del ángulo correspondiente a un segmento de longitud Δ_{np} a lo largo de un radio de la imagen.



(a) $\theta \in [0{,}0 - 0{,}175)$



(b) $\theta \in [0{,}175 - 1{,}57)$

Figura 5.3: Representación de Q_l^φ en función de θ para diferentes valores de Δ_{np}.

5.2. Mapa

El mapa de características generado por OV-FastSLAM se compone de dos tipos de balizas: balizas estándar —en adelante denominadas simplemente balizas— y balizas candidatas:

– Una baliza $B_{t,j}^{k,s}$ —j es el índice de la baliza, k es la partícula, s la hipótesis de asociación y t el instante de tiempo— contiene la información de su posición 3D, representada mediante una distribución gaussiana de media $\mu_{t,j}^{k,s}$ y covarianza $\Sigma_{t,j}^{k,s}$, y el número de veces que ha sido detectada, $i_{t,j}^{k,s}$. Debido a las características especiales de los entornos en los que OV-FastSLAM debe operar habitualmente, y también por la utilización de una cámara omnidireccional, muchas balizas son inicializadas con medidas desde posiciones muy alejadas del robot. Estas balizas poseen en ocasiones posiciones 3D con bastante error, que es corregido con el paso del tiempo y a medida que son detectadas desde posiciones más cercanas. Con el fin de reducir la influencia de estas balizas incorrectamente inicializadas, se han clasificado las balizas en dos grupos:

  - Las balizas de tipo I poseen posiciones fiables, pues han sido detectadas por el robot desde poses cercanas (como máximo a una distancia d_{min} medida en el plano XY).
  - Las balizas de tipo II tienen posiciones iniciales imprecisas, pues han sido detectadas por el robot desde poses lejanas. En nuestro sistema estas balizas no contribuyen ni a la actualización de la pose del robot ni al peso de la partícula. Sin embargo es necesario tenerlas en cuenta ya que permiten mejorar el proceso de asociación de datos y, además, cuando se transforman en balizas de tipo I su posición 3D es mucho más precisa.

– Las balizas candidatas son aquellas que aún no se han podido inicializar. Cada baliza candidata $C_{t,j}^{k,s}$ contiene un conjunto de medidas asociadas $Z_{t,j}^{k,s} = \{z_{t-\tau,l_{t-\tau}}, \ldots, z_{t,l_t}\}$ que son utilizadas para chequear las condiciones de inicialización y para calcular la posición inicial de la baliza.

5.3. Algoritmo

OV-FastSLAM se basa en el algoritmo FastSLAM 2.0 [Montemerlo y col., 2002] (ver apéndice A), que utiliza un filtro de partículas de tipo Rao-Blackwell [Thrun y col., 2003] pa-

Algoritmo 5.1 OV-FastSLAM (Z_t, u_t, Y_{t-1})

1: **for** $k = 1$ to M **do**
2: **for** $s = 1$ to N_Ψ **do**
3: Obtener partícula k con asociación de datos s de Y_{t-1}:
$$x_{t-1}^{k,s}, \{B_{t-1,1}^{k,s}, \ldots, B_{t-1,N_{t-1}^{k,s}}^{k,s}\}, \{C_{t-1,1}^{k,s}, \ldots, C_{t-1,\eta_{t-1}^{k,s}}^{k,s}\}$$
4: $\widehat{x}_t = g\left(x_{t-1}^{k,s}, u_t\right)$
5: $\Phi_{t,1}^{k,s} = probabilidadMedidas\_balizas\,()$
6: **end for**
7: $\{\Psi_{t,1}^{k,1}, \ldots, \Psi_{t,1}^{k,N_\Psi}\} = asociaciónDatos\_balizas\,()$
8: **for** $s = 1$ to N_Ψ **do**
9: $\Phi_{t,2}^{k,s} = probabilidadMedidas\_balizasCandidatas\,()$
10: $\Psi_{t,2}^{k,s} = asociaciónDatos\_balizasCandidatas\,()$
11: $actualizaciónPoseRobot\,()$
12: $actualizaciónMapa\,()$
13: $inicializaciónBalizas\,()$
14: **end for**
15: **end for**
16: $Y_t = remuestreo\,()$

ra representar la distribución de probabilidad. La trayectoria del robot se estima con un filtro de partículas y cada partícula contiene una pose del robot y un mapa basado en características. Las balizas del mapa están representadas por distribuciones gaussianas y se actualizan mediante EKFs.

OV-FastSLAM (alg. 5.1) recibe como entradas el conjunto de medidas en el momento t (Z_t), el control (u_t) y el conjunto de partículas en el instante anterior (Y_{t-1}). Cada partícula k contiene N_Ψ asociaciones de datos para hacer frente a toda la complejidad de las imágenes omnidireccionales con oclusiones severas. Por cada asociación s hay una pose del robot estimada ($x_{t-1}^{k,s}$), un mapa de $N_{t-1}^{k,s}$ balizas $\{B_{t-1,1}^{k,s}, \ldots, B_{t-1,N_{t-1}^{k,s}}^{k,s}\}$ y un conjunto de $\eta_{t-1}^{k,s}$ balizas candidatas $\{C_{t-1,1}^{k,s}, \ldots, C_{t-1,\eta_{t-1}^{k,s}}^{k,s}\}$.

OV-FastSLAM itera sobre las M partículas y las N_Ψ asociaciones para obtener los pesos ($w^{k,s}$). La asociación de datos es jerárquica, i. e., se divide en dos etapas — balizas y balizas candidatas— para priorizar la asociación de las balizas, pues son más fiables que las balizas candidatas. Inicialmente OV-FastSLAM realiza la predicción de la pose del robot, y a continuación (Alg. 5.1, líneas 2 - 6) se estiman las probabilidades de las medidas para el primer

nivel de la asociación de datos, que se resuelve en la línea 7. El segundo bucle sobre las N_Ψ asociaciones (Alg. 5.1, líneas 8 - 14) calcula la probabilidades de las medidas y la asociación de datos del segundo nivel (balizas candidatas), realiza las actualizaciones de la pose del robot y del mapa, y termina con la posible inicialización de las nuevas balizas.

Finalmente, OV-FastSLAM remuestrea el conjunto de partículas mediante el algoritmo de baja varianza (sec. 4.1.4) y utilizando el criterio del *tamaño de la muestra efectivo* [Liu, 1996], calculado como [Doucet y col., 2001]:

$$M_{eff} = \frac{1}{\sum_{i=1}^{M} (\widetilde{w}^k)^2} \tag{5.2}$$

donde $\widetilde{w}^k$ es el peso normalizado de la partícula. Cuando todas las partículas tienen pesos muy similares M_{eff} toma su valor máximo, indicando que la PDF real está siendo correctamente aproximada. Sin embargo, cuando la varianza de los pesos de las partículas se incrementa, el valor de M_{eff} disminuye, reflejando una aproximación pobre de la PDF. OV-FastSLAM remuestrea siempre que $M_{eff} < M/2$ [Doucet y col., 2001]. Solo la mejor asociación de cada partícula toma parte de la etapa de remuestreado (sec. 5.3.2).

A continuación se explican en detalle las distintas fases de OV-FastSLAM, haciendo especial énfasis en aquellas partes que presentan novedades respecto al estado del arte.

5.3.1. Probabilidad de asociación de las medidas

La probabilidad de asociación ($\phi_{j,l}$) entre una baliza j y una medida l se realiza siguiendo el algoritmo 5.2. Se mantiene el mismo enfoque que en el algoritmo FastSLAM 2.0 (alg. A.5), que modela la probabilidad de asociación por una gaussiana con media igual a la predicción de la medida, y con una covarianza que depende de la covarianza de la baliza ($\Sigma^{k,s}_{t-1,\,j}$) y del ruido de la medida (Q_l). La principal novedad respecto al proceso seguido en FastSLAM 2.0 es que se añade el bucle de las líneas 5 - 12 para procesar múltiples medidas en cada iteración, y además la pose del robot se obtiene (a partir del muestreo de la gaussiana asociada) en una etapa posterior con el fin de tener en cuenta la asociación de datos.

La probabilidad de que una medida no asignada en el primer nivel de asociación se corresponda a una baliza candidata j se calcula a partir del algoritmo 5.3. El proceso se basa también en una distribución de probabilidad gaussiana, pero como las balizas candidatas no tienen una posición definida — son solo un conjunto de medidas previas —, el proceso es más complejo. El cálculo de $\phi_{j,l'}$ tiene los siguientes pasos:

Algoritmo 5.2 OV-FastSLAM: *probabilidadMedidas_balizas*().

1: **for** $j = 1$ to $N_{t-1}^{k,s}$ **do**
2: $\quad \bar{z}_j = h\left(\mu_{t-1,j}^{k,s}, \widehat{x}_t\right)$
3: $\quad H_{x,j} = \nabla_{x_t} h\left(\mu_{t-1,j}^{k,s}, \widehat{x}_t\right)$
4: $\quad H_m = \nabla_m h\left(\mu_{t-1,j}^{k,s}, \widehat{x}_t\right)$
5: $\quad$ **for** $l = 1$ to N_{Z_t} **do**
6: $\quad\quad Q_{j,l} = Q_l + H_m \Sigma_{t-1,j}^{k,s} H_m^T$
7: $\quad\quad \Sigma_x = [H_{x,j}^T Q_{j,l}^{-1} H_{x,j} + R_t^{-1}]^{-1}$
8:
9: $\quad\quad \mu_x = \Sigma_x H_{x,j}^T Q_{j,l}^{-1} \left(z_{t,l} - \bar{z}_j\right) + \widehat{x}_t$
10: $\quad\quad \widehat{z} = h\left(\mu_{t-1,j}^{k,s}, \mu_x\right)$
11: $\quad\quad \phi_{j,l} = (2\pi)^{-\frac{Dim\left(Q_{j,l}\right)}{2}} \left|Q_{j,l}\right|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}\left(z_{t,l} - \widehat{z}\right)^T Q_{j,l}^{-1} \left(z_{t,l} - \widehat{z}\right)\right\}$
12: $\quad$ **end for**
13: **end for**
14: **return** $\Phi_{t,1}^{k,s} = \left[\phi_{j,l}\right]$

1. Para cada medida $z_i \in Z_{t-1,j}^{k,s}$ — el conjunto de medidas que pertenece a la baliza candidata $C_{t-1,j}^{k,s}$ — se calcula el punto de cruce entre z_i y $z_{t,l'}$ (alg. 5.3, línea 6). La probabilidad asignada al punto de cruce (ϕ_c) es la probabilidad conjunta de todas las medidas de $Z_{t-1,j}^{k,s}$ para la posición 3D (x_c) del punto de cruce (alg. 5.3, líneas 9 - 18). La probabilidad de que una medida z_q se haya generado a partir de x_c (ϕ_q) es modelada por una distribución gaussiana con media igual a la predicción de la medida ($\bar{z}$). Esta predicción se estima a partir de la posición del punto de cruce (x_c) y la predicción de la pose del robot ($\widehat{x}_{t(q)}$) en $t(q)$, que es el instante en que se obtuvo la medida q. Por otra parte, la covarianza depende de la covarianza del punto de cruce (Σ_0, la covarianza inicial de una baliza) y del ruido de la medida (Q_q).

2. Se selecciona el punto de cruce que maximiza la probabilidad sobre todas las medidas (ϕ_c) en $Z_{t-1,j}^{k,s}$. El ángulo entre las medidas que definen el punto de cruce debe ser mayor de un umbral γ_{min} (alg. 5.3, línea 19) para tener una confianza elevada en su posición 3D.

3. $\phi_{j,l'}$ será la probabilidad mínima de todas las medidas para el punto de cruce seleccionado (alg. 5.3, línea 21). Esta condición es muy restrictiva, y ayuda a eliminar las

Algoritmo 5.3 OV-FastSLAM: *probabilidadMedidas_balizasCandidatas*()

for $j = 1$ to $\eta_{t-1}^{k,s}$ **do**
  for $l' = 1$ to $N_{Z_t'}^{k,s}$ **do**
    $\phi_{max} = 0$
    $validCross = false$
    for $i = 1$ to $|Z_{t-1,j}^{k,s}|$ **do**
      $x_c = crossPoint(z_{t,l'}, z_i)$
      $\phi_c = 1$
      $\phi_{min} = 1$
      for $q = 1$ to $N_{Z_{t-1,j}^{k,s}}$ **do**
        $\overline{z} = h\left(x_c, \widehat{x}_{t(q)}\right)$
        $H_m = \nabla_m h\left(x_c, \widehat{x}_{t(q)}\right)$
        $Q_{j,l'} = Q_q + H_m \Sigma_0 H_m^T$
        $\phi_q = (2\pi)^{-\frac{Dim\left(Q_{j,l'}\right)}{2}} \left|Q_{j,l'}\right|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(z_q - \overline{z})^T Q_{j,l'}^{-1}(z_q - \overline{z})\right\}$
        $\phi_c := \phi_c \phi_q$
        if $\phi_q < \phi_{min}$ **then**
          $\phi_{min} = \phi_q$
        end if
      end for ▷ For $q = 1$ to $|Z_{t-1,j}^{k,s}|$
      if $\phi_c > \phi_{max} \wedge \left(\widehat{z_{t,l'}, z_i} > \gamma_{min} \vee !validCross\right)$ **then**
        $\phi_{max} = \phi_c$
        $\phi_{j,l'} = \phi_{min}$
        if $\widehat{z_i, z_k} > \gamma_{min} \wedge !validCross$ **then**
          $validCross = true$
        end if
      end if
    end for ▷ For $i = 1$ to $N_{Z_{t-1,j}^{k,s}}$
  end for ▷ For $l' = 1$ to $N_{Z_t'}^{k,s}$
end for ▷ For $j = 1$ to $\eta_{t-1}^{k,s}$
return $\Phi_{t,2}^{k,s} = \left[\phi_{j,l'}\right]^T$

balizas candidatas con medidas asociadas que no tienen una buena correspondencia con el punto de cruce más probable.

5.3.2. Asociación de datos jerárquica

La asociación de datos de OV-FastSLAM consta de dos niveles. En el primer nivel, se realiza la asociación entre balizas y medidas mediante el algoritmo de Murty [Cox y Miller, 1995], que obtiene las N_Ψ mejores asignaciones. Todas aquellas medidas que no han sido asociadas en esta primera fase pasan al segundo nivel de la jerarquía, donde se realiza la asociación de las mismas con las balizas candidatas mediante el algoritmo Húngaro [Kuhn, 1955] que obtiene la mejor asignación.

A continuación se describen en primer lugar los algoritmos Húngaro y de Murty para, posteriormente, explicar en detalle las diferentes etapas de la asociación de datos jerárquica.

Método Húngaro

El método Húngaro [Kuhn, 1955] es un algoritmo de optimización combinatoria que resuelve el problema de asociación de datos entre los elementos de dos conjuntos (A y B) en tiempo polinómico. El algoritmo modela el problema de asignación por medio de una matriz de coste (Φ) de dimensiones $N_A \times N_B$, donde cada elemento ($\phi_{a,b}$) representa el coste de asignar el elemento $a \in A$ al elemento $b \in B$ [1].

Como resultado, el método genera una matriz de hipótesis o ambigüedad (Ψ) en la que cada elemento ($\psi_{a,b}$) puede tener un valor de 1 ó 0, indicando que el elemento $a \in A$ ha sido asociado o no, respectivamente, al elemento $b \in B$. La matriz de ambigüedad cumple las siguientes condiciones:

$$\sum_b \psi_{a,b} = 1, \ \ \forall a \qquad y \qquad \sum_a \psi_{a,b} \in \{0, 1\}, \ \ \forall b \tag{5.3}$$

La primera condición indica que cada elemento de A debe ser asignado a un único elemento de B, mientras que la segunda condición refleja que cada elemento de B puede ser, o bien asociado a un único elemento de A, o bien no ser asignado a ninguno.

Algoritmo de Murty

El algoritmo de Murty determina el conjunto de las N_Ψ mejores asociaciones en tiempo polinómico. Fue diseñado originalmente por Murty [Murty, 1986], y posteriormente modificado por Cox y Miller [Cox y Miller, 1995] con el fin de poder resolver múltiples problemas

[1] Al ser una matriz de coste, en realidad cada elemento es $-\log \phi_{a,b}$, pero con el fin de simplificar la notación se utilizará $\phi_{a,b}$.

Algoritmo 5.4 Algoritmo de Murty

$L = \{< \Phi_0, \Psi_0 = Hungarian(\Phi_0) >, .. < \Phi_N, \Psi_N = Hungarian(\Phi_N) >\}$
$L_\Psi = \emptyset$
for $i = 1$ to N_Ψ **do**
    Obtener el mejor par, $< \Phi_n, \Psi_n > \in L$
    $L = L - < \Phi_n, \Psi_n >$
    $L_\Psi = L_\Psi \cup \Psi_n$
    for $\forall < a, b, c > \in \Psi_n$ **do**
        $\Phi'_n = \Phi_n - < a, b, c >$
        $\Psi'_n = Hungarian(\Phi'_n)$
        if $\exists \Psi'_n$ **then**
            $L = L \cup < \Phi'_n, \Psi'_n >$
        end if
        Eliminar de Φ_n todas las tripletas que incluyan a o b excepto $< a, b, c >$
    end for
end for

de asociación al mismo tiempo y además cambiar la condición de terminación, pues hipótesis con una probabilidad menor que un cierto porcentaje de la mejor hipótesis pueden ser descartadas.

El algoritmo de Murty más general (alg. 5.4) parte de un conjunto de pares problema/solución (matriz de coste/matriz de ambigüedad) que han sido resueltos con el método Húngaro. De forma iterativa se repite el siguiente proceso: se selecciona la mejor solución, eliminándola de la lista L y añadiéndola a la lista de soluciones que serán devueltas (L_Ψ). Para cada una de las tripletas $< a, b, c >$ de la solución elegida, donde a y b son los elementos que se están asignando y c el coste de dicha asignación, se crea un nuevo problema (Φ'_n) eliminando la posibilidad de esa asociación. De esta forma es posible encontrar múltiples soluciones a un mismo problema, y en orden creciente de coste. El proceso concluye cuando se han obtenido las N_Ψ mejores soluciones del conjunto de problemas.

Primer nivel de asociación

En el primer nivel de asociación se obtienen las N_Ψ mejores asignaciones para cada partícula entre las medidas actuales y las balizas de tipos I y II. Se resuelve utilizando el algoritmo de Murty con la matriz de coste $\Phi^{k,s}_{t,1}$ (fig. 5.4) de tamaño $N^{k,s}_{t-1} \times \left(N_{Z_t} + N^{k,s}_{t-1}\right)$. El lado iz-

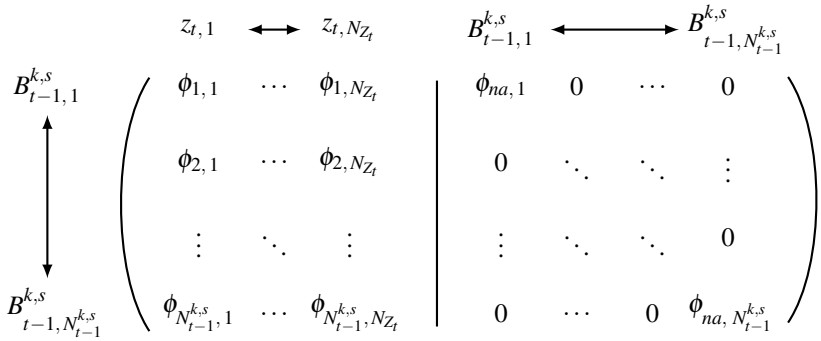


Figura 5.4: Matriz de coste del primer nivel de asociación, $\Phi^{k,s}_{t,1}$.

quierdo de la matriz se genera a partir del algoritmo 5.2, en el que las filas representan las balizas y las columnas las medidas en cada instante.

El lado derecho es una submatriz diagonal con una columna por baliza. Los elementos de la diagonal se inicializan al valor de $\phi_{na,j}$ (ec. 5.4) y representan la probabilidad de que una baliza no sea asociada con ninguna medida. Los demás elementos se inicializan a 0 para evitar que distintas combinaciones de estas últimas columnas puedan producir asociaciones diferentes. Con esta definición para $\Phi^{k,s}_{t,1}$ se debe seleccionar una asignación para cada baliza; esto incluye la posibilidad de que una baliza no sea asociada. Por otra parte, algunas de las medidas puede que no sean asignadas, y por lo tanto pasarán al segundo nivel de la asociación.

La probabilidad de no asociación se define como:

$$\phi_{na,j} = \phi_{new,j}\ \phi_{out,j} \tag{5.4}$$

El primer término se refiere a la probabilidad de que ninguna de las medidas haya sido originada por la baliza. La probabilidad de que una medida provenga de una nueva baliza es:

$$\phi_{new,l} = (2\pi)^{-\frac{Dim(Q_l)}{2}} |Q_l|^{-\frac{1}{2}} \exp\{-\frac{1}{2}\xi^2_{new}\} \tag{5.5}$$

donde Q_l es el ruido de la medida (ec. 5.1) y ξ_{new} es un parámetro que representa la diferencia (error), en número de desviaciones estándar, entre la medida y la predicción de la misma. Valores superiores a ξ_{new} indican que la asociación más probable es que la medida l no provenga de una baliza. Sin embargo, nuestro objetivo es modelar que ninguna de las medidas provenga de la baliza j ($\phi_{new,j}$), por lo que para estimar esta probabilidad se sustituye en la ecuación 5.5

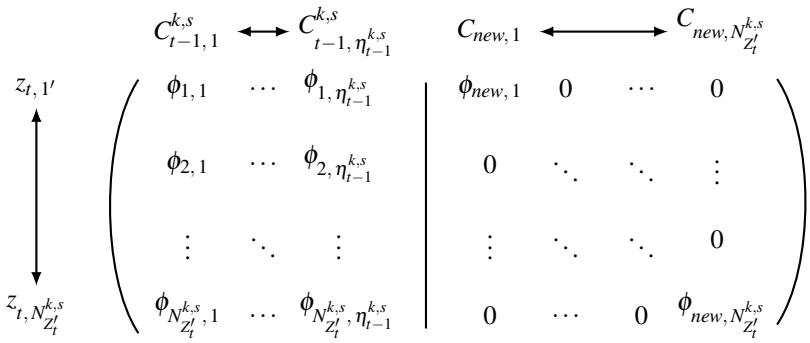


Figura 5.5: Matriz de coste del segundo nivel de asociación, $\Phi^{k,s}_{t,2}$.

el ruido de una medida específica Q_l por el ruido de la predicción de la medida para la baliza j (Q_j), que se estima en base a la predicción de la medida $\bar{z}_j$.

El segundo elemento de la ecuación 5.4 modela la probabilidad de que la baliza esté fuera del campo de visión de la cámara:

$$\phi_{out,j} = \begin{cases} \text{si} \quad \theta <= \theta_{max} & 0 \\ \text{si} \quad \theta > \theta_{max} & 1-(2\pi)^{-\frac{1}{2}} \left|Q^{\theta}_{j}\right|^{-\frac{1}{2}} \exp\{-\frac{1}{2}\frac{\left(\theta_{\bar{z}_j}-\theta_{max}\right)^2}{Q^{\theta}_{j}}\} \end{cases} \tag{5.6}$$

donde $\theta_{\bar{z}_j}$ es el ángulo de elevación de la baliza j, θ_{max} es un parámetro y Q^{θ}_{j} es Q^{θ}_{l} (ec. 5.1) para $\bar{z}_j$. Esta probabilidad modula $\phi_{new,j}$: si la baliza está fuera del campo de visión ($\phi_{out,j} = 1$), la probabilidad de que no se asocie será $\phi_{new,j}$, mientras que si la baliza está dentro del campo de visión su probabilidad de no asociación es muy baja.

Segundo nivel de asociación

El segundo nivel de asociación establece la correspondencia entre las medidas que no han sido asociadas en el primer nivel y las balizas candidatas, utilizando el método Húngaro y la matriz de coste $\Phi^{k,s}_{t,2}$ (fig. 5.5). El tamaño de $\Phi^{k,s}_{t,2}$ es $N^{k,s}_{Z'_t} \times \left(\eta^{k,s}_{t-1} + N^{k,s}_{Z'_t}\right)$, de forma que existe una fila para cada medida ($z_{t,l'}$) que no fue asociada en el primer nivel y que debe ser asignada obligatoriamente en esta segunda fase.

El lado izquierdo de la matriz se genera mediante el algoritmo 5.3. Por tanto, existe una columna por cada baliza candidata y cada elemento $\phi_{j,l'}$ representa la probabilidad de asocia-

ción de la baliza candidata j con la medida l' —es necesario trasponer la matriz con elementos $\phi_{j,l'}$ (alg. 5.3, línea 29) para generar $\Phi^{k,s}_{t,2}$—.

Por otra parte, el lado derecho es una matriz diagonal con una columna por cada medida que pasa al segundo nivel. Los elementos de la diagonal ($\phi_{new,l'}$) representan la probabilidad de que la medida sea originada desde una nueva baliza candidata (ec. 5.5). En este nivel de asociación no se incluye la posibilidad de que la baliza candidata se encuentre fuera del campo de visión, ya que las posiciones estimadas de las balizas candidatas experimentan cambios rápidos.

Proceso completo de asociación

La figura 5.6 muestra el proceso completo de asociación de datos para una partícula. Un ciclo de asociación comprende varias iteraciones de OV-FastSLAM, empezando tras el último remuestreo en $t-1$ y finalizando antes del siguiente remuestreo en $t+n$. Al comienzo de la iteración t existe una única matriz de coste por partícula ($\Phi^{k,best}_{t,1}$) en el primer nivel de asociación de datos ($DA_{t,1}$), generando las mejores N_Ψ asociaciones mediante el algoritmo de Murty. De cada una de estas asociaciones ($\Psi^{k,s}_{t,1}$), el algoritmo crea una matriz de coste para el segundo nivel ($\Phi^{k,s}_{t,2}$) y el método húngaro devuelve la mejor asociación ($\Psi^{k,s}_{t,2}$) para cada una de ellas. Por lo tanto, al final de la primera iteración cada partícula tiene N_Ψ asociaciones distintas, y en consecuencia N_Ψ poses del robot y mapas.

En la segunda iteración ($t+1$), la asociación de primer nivel recibe N_Ψ matrices de coste —cada una representa un problema de asociación diferente— y genera las N_Ψ mejores asociaciones en conjunto. Esto significa que todas las matrices de coste de una misma partícula compiten entre sí para generar las N_Ψ mejores asociaciones mediante el algoritmo de Murty, y por tanto podría ocurrir que parte de las matrices de coste no contribuyan a las N_Ψ mejores asociaciones. A continuación, las mejores asociaciones se utilizan de nuevo para generar las matrices de coste para el segundo nivel ($\Phi^{k,s}_{t+1,2}$), y para cada una de ellas el método húngaro devuelve la mejor correspondencia. El proceso se repite hasta que tiene lugar el remuestreo. En la última iteración se selecciona la mejor asociación por partícula — $\Psi^{k,best}_{t+n}$ representa los dos niveles de asociación— basándose en el mejor valor de $\Psi^{k,s}_{t+n,1}$ y su correspondiente $\Psi^{k,s}_{t+n,2}$. Por lo tanto, al final del último ciclo del proceso de asociación completo y antes de la etapa de remuestreo, cada partícula tiene una única asociación de datos —y en consecuencia, una sola pose del robot y un mapa—.

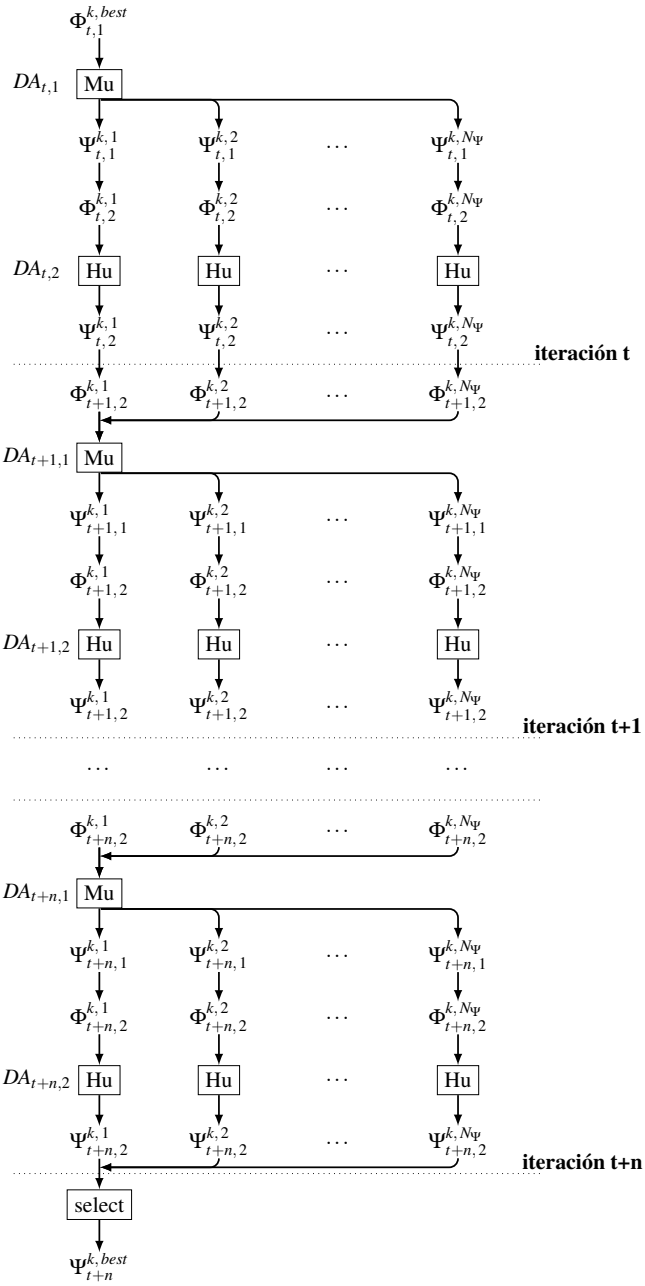


Figura 5.6: Ciclo completo de asociación de datos para una partícula en OV-FastSLAM.

5.3.3. Actualización de la pose del robot

El algoritmo 5.5 describe la actualización de la pose del robot para múltiples medidas simultáneas. Si ninguna medida ha sido asociada con una baliza, la pose se muestrea a partir de la distribución de probabilidad a priori estimada teniendo en cuenta solo el modelo de movimiento (alg. 5.5, línea 2). En otro caso, OV-FastSLAM solo tiene en cuenta las balizas del tipo I ($\Upsilon_j = 1$) a las cuales se haya asociado una medida, y construye a partir de ellas una PDF gaussiana de la que se muestreará la pose del robot. La PDF se modela inicialmente con una distribución gaussiana con media igual a la pose estimada por el modelo de movimiento ($\widehat{x_t}$) y cuya covarianza es la del modelo de movimiento (R_t). Esta distribución se va modificando en el bucle (alg. 5.5, líneas 6 - 11), incorporando las medidas asociadas a las balizas siguiendo un proceso equivalente al de actualización de un EKF.

Cada vez que se incorpora una observación la covarianza de la distribución propuesta se reduce, por lo que el orden en el que se procesan las balizas es importante: al principio la covarianza de la PDF es mayor, y por lo tanto las correcciones propuestas por las primeras observaciones procesadas tendrán una mayor influencia en la PDF final. Por este motivo, OV-FastSLAM procesa las balizas en orden creciente de $Tr\left(Q_{j,\psi_j}\right)$ —la traza de la matriz de covarianza para la medida asociada a la baliza j (ψ_j)—, i. e., covarianzas menores se procesan

Algoritmo 5.5 OV-FastSLAM: *actualizaciónPoseRobot*()

1: **if** $\sum_{j=1}^{N_{t-1}^{k,s}} \psi_j = 0$ **then**
2: $\quad x_t^{k,s} \sim p\left(x_t | x_{t-1}^{k,s}, u_t\right)$
3: **else**
4: $\quad \Sigma_x = R_t$
5: $\quad \mu_x = \widehat{x_t}$
6: $\quad$ **for** $j = 1$ to $N_{t-1}^{k,s}$ **do**
7: $\quad\quad$ **if** $\psi_j > 0 \wedge \Upsilon_j = 1$ **then**
8: $\quad\quad\quad \Sigma_x := [H_{x,j}^T Q_{j,\psi_j}^{-1} H_{x,j} + \Sigma_x^{-1}]^{-1}$
9: $\quad\quad\quad \mu_x := \mu_x + \Sigma_x H_{x,j}^T Q_{j,\psi_j}^{-1} \left(z_{t,\psi_j} - \overline{z}_j\right)$
10: $\quad\quad$ **end if**
11: $\quad$ **end for**
12: $\quad x_t^{k,s} \sim N(\mu_x, \Sigma_x)$
13: **end if**

en primer lugar. De esta manera, tanto las covarianzas de las balizas como las de las medidas se tienen en cuenta, y aquellas con una confianza mayor se procesarán primero.

5.3.4. Actualización del mapa

La actualización del mapa tiene lugar en dos fases: balizas y balizas candidatas.

Actualización de las balizas

El algoritmo 5.6 describe el proceso de actualización de las balizas y el cálculo del peso de las partículas. La actualización de la posición de las balizas requiere recalcular la predicción de la medida (y las matrices que de ella dependen), puesto que la pose del robot ha sido modificada en la etapa de actualización de la pose del robot (alg. 5.6, líneas 5 - 7). El resto del proceso de actualización de la posición sigue los mismos pasos que FastSLAM 2.0 (apéndice A), i. e., la actualización EKF estándar. Además, $i_{t,j}^{k,s}$ cuenta el número de observaciones para la baliza j, y disminuye su valor cada vez que esta no se detecta pero se encuentra dentro del campo de visión de la cámara (alg. 5.6, líneas 26 - 28).

El peso de la partícula-asociación ($\omega^{k,s}$) es el producto de los pesos parciales correspondientes a cada baliza ($\widehat{w}$). Aunque se realiza la actualización tanto de las balizas de tipo I como de las balizas de tipo II, OV-FastSLAM solo tiene en cuenta las balizas de tipo I ($\Upsilon_j = 1$) para el cálculo del peso. Si la baliza ha sido observada en el instante t, el peso parcial sigue una distribución gaussiana con media igual a la predicción de la medida y la covarianza es una combinación del ruido de la medida, la covarianza de la balizas y el ruido del movimiento. Sin embargo, si la baliza no se observa OV-FastSLAM también incorpora esta evidencia negativa, asignándole al peso parcial la probabilidad de que la baliza esté fuera del campo de visión de la cámara (alg. 5.6, línea 22).

Finalmente, tras la actualización de las balizas se comprueba si alguna de las de tipo II que haya sido asociada está a una distancia inferior a d_{min} del robot, en cuyo caso es transformada a baliza de tipo I y reinicializada.

Actualización de las balizas candidatas

La actualización de las balizas candidatas ($C_{t,j}^{k,s}$) se realiza teniendo en cuenta el resultado de la asociación de datos del segundo nivel (sec. 5.3.2):

Algoritmo 5.6 OV-FastSLAM: *actualizaciónBalizas*(). Se muestran en negro las diferencias respecto al algoritmo A.7

1: $\omega^{k,s} = 1$
2: **for** $j = 1$ to $N_{t-1}^{k,s}$ **do**
3: **if** $\psi_j > 0$ **then**
4: $i_{t,j}^{k,s} = i_{t-1,j}^{k,s} + 1$
5: $\widetilde{z} = h\left(\mu_{t-1,j}^{k,s}, x_t^{k,s}\right)$
6: $\widetilde{H}_m = \nabla_m h\left(\mu_{t-1,j}^{k,s}, x_t^{k,s}\right)$
7: $\widetilde{Q}_{j,\psi_j} = Q_{\psi_j} + \widetilde{H}_m \Sigma_{t-1,j}^{k,s} \widetilde{H}_m^T$
8: $K = \Sigma_{t-1,j}^{k,s} \widetilde{H}_m^T \widetilde{Q}_{j,\psi_j}^{-1}$
9: $\mu_{t,j}^{k,s} = \mu_{t-1,j}^{k,s} + K\left(z_{t,\psi_j} - \widetilde{z}\right)$
10: $\Sigma_{t,j}^{k,s} = \left(I - K\widetilde{H}_m\right)\Sigma_{t-1,j}^{k,s}$
11: **if** $\Upsilon_j = 1$ **then**
12: $\widetilde{H}_x = \nabla_{x_t} h\left(\mu_{t-1,j}^{k,s}, x_t^{k,s}\right)$
13: $L = \widetilde{H}_x R_t \widetilde{H}_x^T + \widetilde{Q}_{j,\psi_j}$
14: $\widehat{w} = (2\pi)^{-\frac{Dim(L)}{2}} |L|^{-\frac{1}{2}}$ $\exp\left\{-\frac{1}{2}\left(z_{t,\psi_j} - \widetilde{z}\right)^T L^{-1}\left(z_{t,\psi_j} - \widetilde{z}\right)\right\}$
15: **else** ▷ If $\Upsilon_j = 1$
16: $\widehat{w} = 1$
17: **end if** ▷ If $\Upsilon_j = 1$
18: **else** ▷ If $\psi_j > 0$
19: $\mu_{t,j}^{k,s} = \mu_{t-1,j}^{k,s}$
20: $\Sigma_{t,j}^{k,s} = \Sigma_{t-1,j}^{k,s}$
21: **if** $\Upsilon_j = 1$ **then**
22: $\widehat{w} = \phi_{out,j}$
23: **else** ▷ If $\Upsilon_j = 1$
24: $\widehat{w} = 1$
25: **end if** ▷ If $\Upsilon_j = 1$
26: **if** $\mu_{t-1,j}^{k,s}$ está dentro del rango perceptual de $x_t^{k,s}$ **then**
27: $i_{t,j}^{k,s} = i_{t-1,j}^{k,s} - 1$
28: **end if** ▷ If $\mu_{t-1,j}^{k,s}$ está dentro
29: **end if** ▷ If $\psi_j > 0$
30: $\omega^{k,s} = \omega^{k,s} \cdot \widehat{w}$
31: **end for**

1. **Baliza candidata con medida asignada**. Se añade la medida a $Z_{t,j}^{k,s}$, se incrementa el contador de observaciones ($i_{t,j}^{k,s}$) y se comprueba si se dan las condiciones de inicialización (alg. 5.7).

2. **Baliza candidata sin medida asociada**. Si la baliza se encuentra dentro del campo de observación de la cámara, se decrementa el contador de observaciones: $i_{t,j}^{k} = i_{t-1,j}^{k,s} - I_{not}$, siendo I_{not} el número de iteraciones consecutivas en las que la baliza candidata no se observó. Si $i_{t,j}^{k,s} < 0$, la baliza candidata será eliminada.

3. **Nueva baliza candidata**. Se realiza la inicialización de su conjunto de medidas y de su contador de observaciones.

5.3.5. Inicialización de balizas

El proceso de inicialización de las balizas transforma balizas candidatas en balizas. El proceso (alg. 5.7) se basa en el algoritmo para el cálculo de probabilidades de las medidas para el conjunto de balizas candidatas (alg. 5.3), y devuelve como posición inicial de la baliza (si esta es inicializada) el punto de cruce (x_c) con la mayor probabilidad (ϕ_c). Las principales diferencias entre los algoritmos 5.7 y 5.3 son:

– En lugar de generar los puntos de cruce entre la medida actual ($z_{t,l'}$) y todas las medidas de la baliza candidata ($Z_{t-1,j}^{k,s}$), la inicialización tiene en cuenta todos los puntos de cruce entre cualesquiera dos medidas en el conjunto $Z_{t-1,j}^{k,s} \cup \left\{z_{t,l'}\right\}$. Esto implica la modificación de los límites de los bucles en l' e i.

– La posición inicial de la baliza es el punto de cruce con el mayor valor de ϕ_c, pero además este punto debe cumplir que el ángulo entre las medidas que lo generan está por encima de un umbral (γ_{min}), con el fin de evitar la inicialización desde dos posiciones muy cercanas (alg. 5.7, línea 6).

Algoritmo 5.7 OV-FastSLAM: *inicializaciónBalizas*()

$\phi_{max} = 0$
$\#validCross = 0$
$validCross = false$
for $l' = 1$ to $|Z_{t,j}^{k,s}|$ **do**
  for $i = l' + 1$ to $|Z_{t,j}^{k,s}|$ **do**
    if $\widehat{z_{l'}, z_i} > \gamma_{min}$ **then**
      $x_c = crossPoint(z_{l'}, z_i)$
      $\#validCross = \#validCross + 1$
      if $!validCross \wedge (l' = |Z_{t,j}^{k,s}| \vee i = |Z_{t,j}^{k,s}|)$ **then**
        $validCross = true$
      end if
      $\phi_c = 1$
      for $q = 1$ to $|Z_{t,j}^{k,s}|$ **do**
        $\overline{z} = h\left(x_c, \widehat{x}_{t(q)}\right)$
        $H_m = \nabla_m h\left(x_c, \widehat{x}_{t(q)}\right)$
        $Q_{l'} = Q_q + H_m \Sigma_0 H_m^T$
        $\phi_q = (2\pi)^{-\frac{Dim\left(Q_{l'}\right)}{2}} |Q_{l'}|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(z_q - \overline{z})^T Q_{l'}^{-1} (z_q - \overline{z})\right\}$
        $\phi_c := \phi_c \, \phi_q$
      end for ▷ For $q = 1$ to $|Z_{t,j}^{k,s}|$
      if $\phi_c > \phi_{max}$ **then**
        $x_{C_{t,j}^{k,s}} = x_c$
        $\phi_{max} = \phi_c$
      end if
    end if
  end for ▷ For $i = l' + 1$ to $|Z_{t,j}^{k,s}|$
end for ▷ For $l' = 1$ to $|Z_{t,j}^{k,s}|$
if $\phi_{max} > \varepsilon \wedge \#validCross > min_{validCross} \wedge validCross$ **then**
  $B_{t,j}^{k,s} = N(x_{C_{t,j}^k}, \Sigma_0)$
  $C_t^{k,s} = C_t^{k,s} - C_{t,j}^{k,s}$
  $B_t^{k,s} = B_t^{k,s} \cup B_{t,j}^{k,s}$
end if

5.4. Resultados experimentales

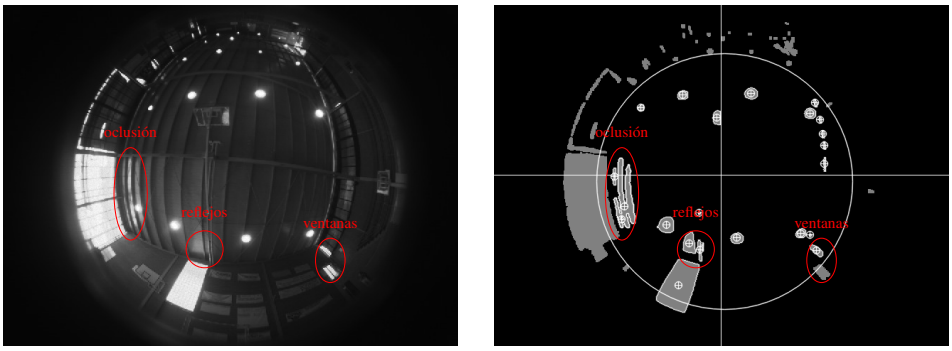
OV-FastSLAM ha sido validado con un robot *Pioneer 3-DX* equipado con una cámara omnidireccional (cap. 3) con la misma configuración que en los experimentos presentados en la sección 4.2. Se han utilizado dos entornos interiores: una sala de exposiciones del Museo Domus (A Coruña) y el pabellón de deportes del colegio Pío XII (Santiago de Compostela) (fig. 5.7). Los entornos son muy diferentes entre sí, lo que permite validar con mayores garantías la robustez de OV-FastSLAM.

El entorno Domus está iluminado en su totalidad con luces artificiales y presenta un suelo irregular. Por otra parte, Pío XII tiene un suelo totalmente plano pero combina iluminación de luz natural con luz artificial, aumentando la dificultad en la detección de las balizas por la presencia de reflejos, saturación de la imagen, etc. (fig. 5.8). Otra diferencia importante es el tipo y la situación de luces: Domus tiene dos tipos de focos (dirigidos y luces de emergencia) que se encuentran situados a diferentes alturas y colocados de forma irregular (figs. 4.5 y 4.6). En cambio, Pío XII tiene todos los focos del mismo tipo y tamaño y están colocados a la misma altura, formando una malla regular (fig. 5.7).

Figura 5.7: Entorno Pío XII.

Tabla 5.1: Características de los entornos y de los recorridos seleccionados para la validación experimental.

Entorno	Tamaño	Traj.	Lazos	Imágenes		v_{max}	ω_{max}	Balizas	
				#img.	Freq.			#B	altura
Domus	$27x7\,m^2$	$72\,m$	4	548	$2\,Hz$	$0{,}36\,m/s$	$0{,}58\,rad/s$	36	$11{,}30\,m$, $3{,}25\,m$
Pío XII	$24x24\,m^2$	$174\,m$	6	1180	$2\,Hz$	$0{,}36\,m/s$	$0{,}58\,rad/s$	20	$6{,}5\,m$



(a) Imagen original (b) Características detectadas

Figura 5.8: Influencia de la luz natural en la detección de las balizas.

La tabla 5.1 recoge las principales características de los dos entornos. De entre todos los recorridos realizados se han seleccionado uno de ejemplo en cada entorno (figs. 5.9 y 5.10). Con el fin de llevar a cabo una validación completa de OV-FastSLAM, para ambos recorridos se ha obtenido la pose real del robot en cada instante a partir de los datos de un sensor láser en condiciones ideales, así como las posiciones reales de las balizas (figs. 5.9 y 5.10).

En todos los experimentos de OV-FastSLAM los parámetros tomaron los siguientes valores: $a = 406{,}1510$, $b = 2{,}9951$, $c = 2{,}0066$, $d = 0{,}2079$, $\beta = 1{,}0$ (parámetros del modelo de la cámara, ec. 3.3), $\Delta_{np} = 2$ (*Pío XII*), 4 (*Domus*) (ruido del modelo de medida, ec. 5.1), $\gamma_{min} = 1{,}22\,rad\,(7°)$ (Alg. 5.3), $\xi_{new} = 8$ (ec. 5.5), $\theta_{max} = 1{,}26\,rad\,(72°)$ (ec. 5.6), $d_{min} = 8\,m$ (*Pío XII*), $6\,m$ (*Domus*) (actualización de balizas), $\Sigma_0 = [0{,}0025\,0{,}0; 0{,}0\,0{,}0025]$.

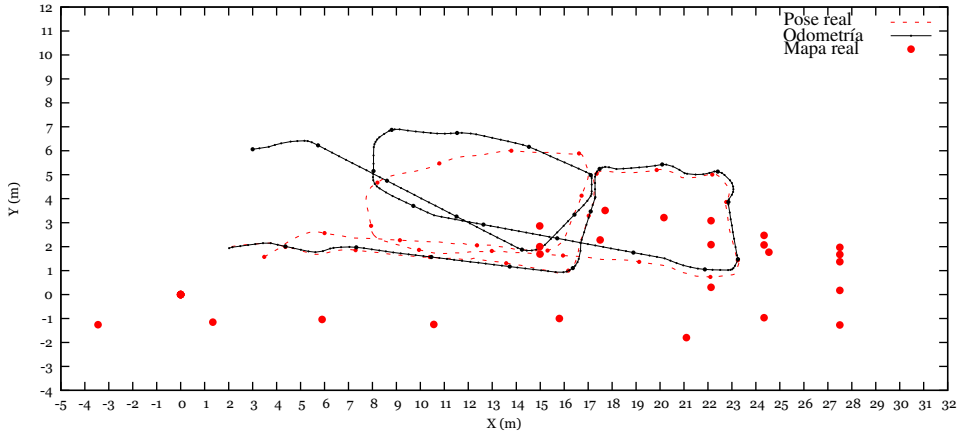


Figura 5.9: Mapa, pose real y odometría del robot en el recorrido seleccionado en el entorno Domus.

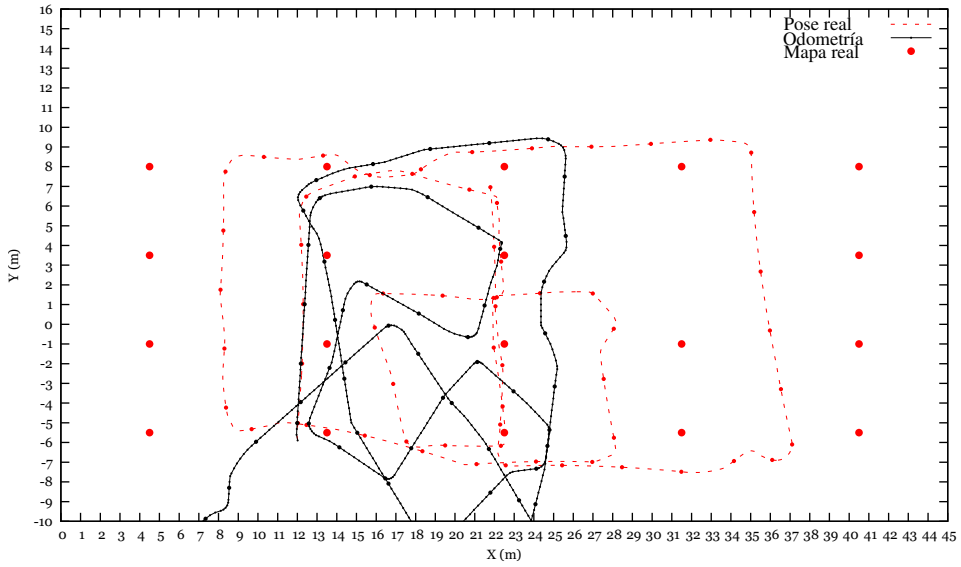


Figura 5.10: Mapa, pose real y odometría del robot en el recorrido seleccionado en el entorno Pío XII.

5.4.1. Comparativa de algoritmos

Con el fin de evaluar OV-FastSLAM hemos utilizado variantes del mismo con diferentes métodos de asociación de datos:

– OV-FastSLAM (M, N_Ψ). OV-FastSLAM con M partículas y la asociación de datos jerárquica descrita en esta memoria: el primer nivel usa el algoritmo de Murty con N_Ψ asociaciones y el segundo nivel se basa en el método Húngaro.

– H-H(M). OV-FastSLAM con M partículas y asociación de datos jerárquica pero en la que los dos niveles utilizan el método húngaro.

– También se han probado diversas modificaciones de OV-FastSLAM con asociaciones de datos no jerárquicas (asociación de datos con un único nivel). En concreto, las asociaciones de datos que se han examinado son basadas en el algoritmo de Murty, en el método Húngaro y en el algoritmo de máxima probabilidad (ML). En todos estos casos, los algoritmos de SLAM fueron incapaces de cerrar lazos y los errores tanto en la pose del robot como en el mapa eran inasumibles.

La comparación entre los diferentes algoritmos se ha realizado en base al error medio en la posición del robot a lo largo del recorrido y para diferentes valores de $M \cdot N_\Psi$, pues el producto de estos dos parámetros es el número de hipótesis que utiliza OV-FastSLAM(M, N_Ψ) en cada iteración. Los algoritmos empleados en la comparación son OV-FastSLAM con diferentes valores en el número de asociaciones por partícula (N_Ψ), así como H-H(M). Puesto que el algoritmo de SLAM es estocástico, se ha utilizado como valor para la comparativa la media de los dos entornos del valor medio de 10 ejecuciones con diferentes semillas (para cada entorno); de esta forma el estudio tiene en cuenta la fiabilidad del algoritmo independientemente de la aleatoriedad de los procesos de muestreo.

La figura 5.11 muestra los resultados de la comparativa, para la que se han realizado un total de 600 ejecuciones con distintos algoritmos, valores de M y N_Ψ y semillas. Se ha comenzado en un valor de $M \cdot N_\Psi = 10$ para que OV-FastSLAM$(M, 5)$ tenga al menos dos partículas. Como se puede observar, ambas implementaciones de OV-FastSLAM(M, N_Ψ) superan sistemáticamente a H-H(M), excepto en dos circunstancias:

– cuando el número de partículas es muy bajo (OV-FastSLAM$(2, 5)$ vs. H-H(10)), puesto que 2 partículas son claramente insuficientes para los entornos de prueba.

– cuando el número de partículas es suficientemente elevado (aproximadamente 100 partículas para los entornos utilizados) el comportamiento de todos los algoritmos es muy similar ya que, como sucede con cualquier filtro de partículas, la distribución propuesta aproxima muy fielmente la PDF real.

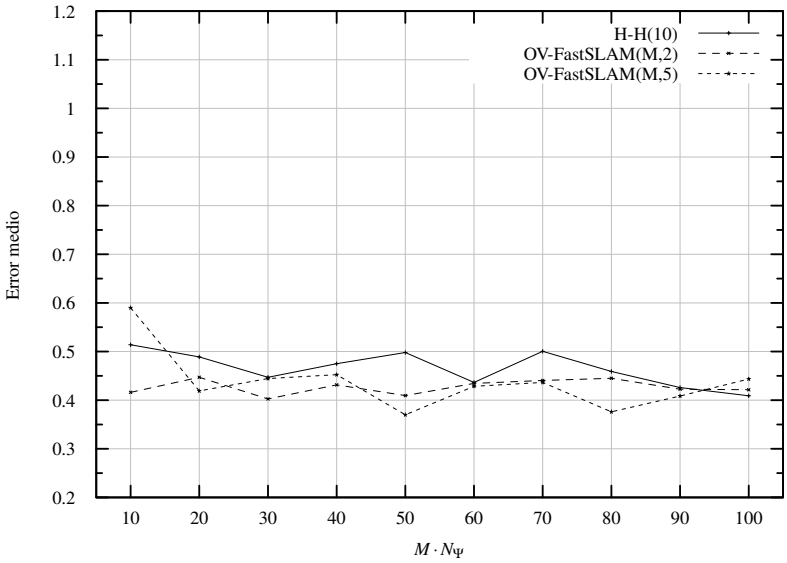


Figura 5.11: Rendimiento medio de los algoritmos en los dos entornos de pruebas con diferentes valores de $M \cdot N_\Psi$.

Se ha realizado la comparación de los diferentes métodos de asociación de datos utilizando tests estadísticos no paramétricos [Demsar, 2006; García y Herrera, 2008] . En primer lugar se aplicó el test de Friedman que calcula el ranking de los resultados de los algoritmos, y rechaza la hipótesis nula —que establece que los resultados de todos los algoritmos son equivalentes— con un nivel de confianza dado —nivel de significancia (α)—. En segundo lugar, se aplicó el test *post-hoc* de Holm [Holm, 1979] para detectar diferencias significativas entre los resultados. Los test se aplicaron sobre el error en la posición del robot para diferentes valores de $M \cdot N_\Psi$, por tanto, siempre se han comparado algoritmos que utilizan un número de hipótesis igual.

La tabla 5.2 resume los resultados de los test usando OV-FastSLAM$(M, 5)$ como algoritmo de control. La columna de ranking se generó usando el test de Friedman y, teniendo en cuenta estos valores, se asignó el valor de i. z representa el valor calculado por el test de Holm y p es su correspondiente p-valor. Todas las hipótesis con $p < \alpha/i$ son rechazadas, lo que significa que los algoritmos son diferentes con un nivel de confianza de $1 - \alpha$. El test muestra que la diferencia de comportamiento entre OV-FastSLAM(M, N_Ψ) y H-H(M) es estadísticamente

Tabla 5.2: Test no paramétrico del comportamiento de los algoritmos de SLAM para diferentes valores de $M \cdot N_\Psi$ (fig. 5.11) con $\alpha = 0{,}05$.

i	Alg.	Ranking	z	p	$\alpha/\mathbf{i}$	Hipótesis
—	OV-FastSLAM(M, 5)	1,6	—	—	—	—
2	H-H(M)	2,7	2,46	0,01	0,025	Rechazada
1	OV-FastSLAM(M, 2)	1,7	0,22	0,82	0,050	Aceptada

significativa. Por otro lado, aunque OV-FastSLAM(M, 5) es mejor que OV-FastSLAM(M, 2), el test no puede rechazar la hipótesis nula.

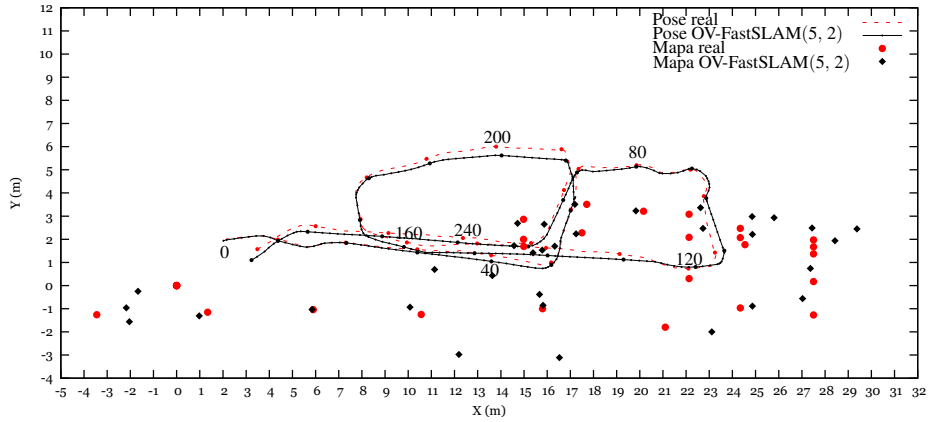
5.4.2. Trayectorias y mapas

Las figuras 5.12 y 5.13 muestran el mapa y la trayectoria reales, así como los resultados obtenidos por OV-FastSLAM(5, 2) en los entornos Domus y Pío XII, respectivamente[2]. También se ha incluido la evolución del error en posición y ángulo para cada una de las pruebas. Se ha escogido como combinación de parámetros para OV-FastSLAM la versión OV-FastSLAM(5, 2) pues ha demostrado en los test previos un buen balance entre precisión y complejidad. La ejecución seleccionada para cada una de las figuras se corresponde con aquella semilla cuyo error medio a lo largo de la trayectoria es el más cercano a la media de todas las ejecuciones (por lo tanto se corresponde con la ejecución más probable).

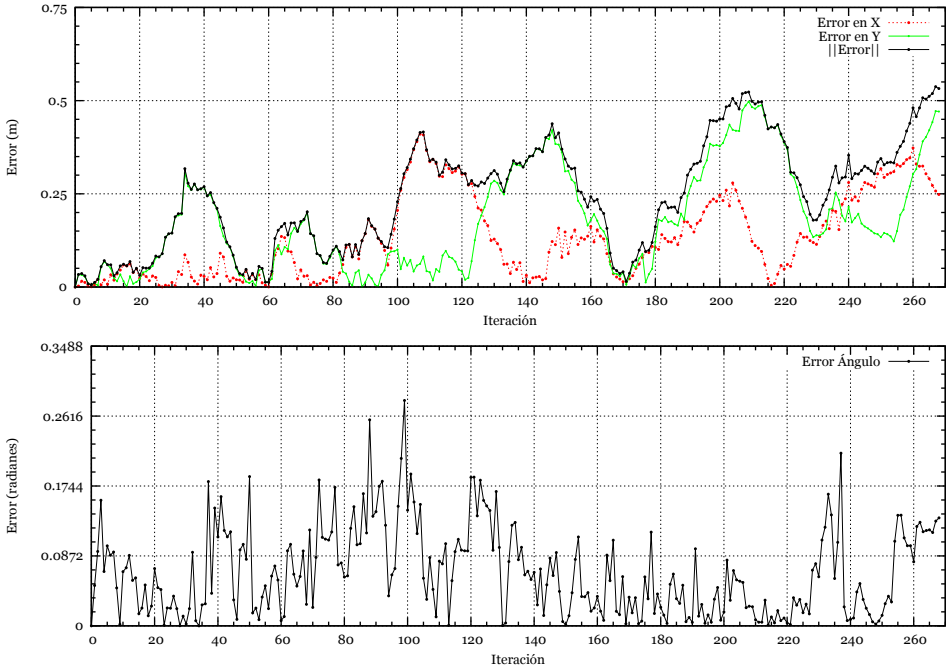
Para el entorno Domus (fig. 5.12)[3], durante la primera parte del recorrido la estimación de la pose es muy fiable, con un error inferior a $0{,}25\,m$. A partir de la iteración 100 el error aumenta, alcanzando un máximo de $0{,}5\,m$ debido a que el robot entra en una zona ($x = 23\,m$, fig. 5.12(a)) de techo bajo. En este caso, la gran mayoría de las balizas que se estaban utilizando dejan de verse, y se comienzan a observar balizas de muy reciente creación y cuya posición aún no es tan fiable. Una vez se vuelven a visualizar las balizas previamente observadas ($x = 22\,m$, fig. 5.12(a)) el error se recupera (iteración 120). El aumento de error más significativo se produce en las iteraciones previas a la 220, y es debido a la presencia de balizas situadas a $11{,}3\,m$ de altura ($x \in [10\,m, 16\,m]$, fig. 5.12(a)) y que solo son visibles desde esa zona del mapa. En consecuencia, sus posiciones son poco fiables, ya que apenas han sido modificadas desde su valor de inicialización. Una vez que el robot abandona esa zona, el error

[2]El algoritmo se ha ejecutado sobre un procesador $Intel(R)Core(TM)i5-2500CPU@3{,}30GHz$. El tiempo medio de ejecución ha sido de $0{,}38$ segundos por iteración.

[3]El vídeo asociado está disponible en `http://persoal.citius.usc.es/cristina.gamallo/Videos/OV-FastSLAM_domus.mp4`.



(a) Mapa y trayectorias reales y estimadas. Los puntos en las trayectorias muestran la pose del robot en iteraciones múltiplos de 10.



(b) Error de la trayectoria en posición y ángulo

Figura 5.12: Resultados de OV-FastSLAM(5, 2) en el entorno Domus.

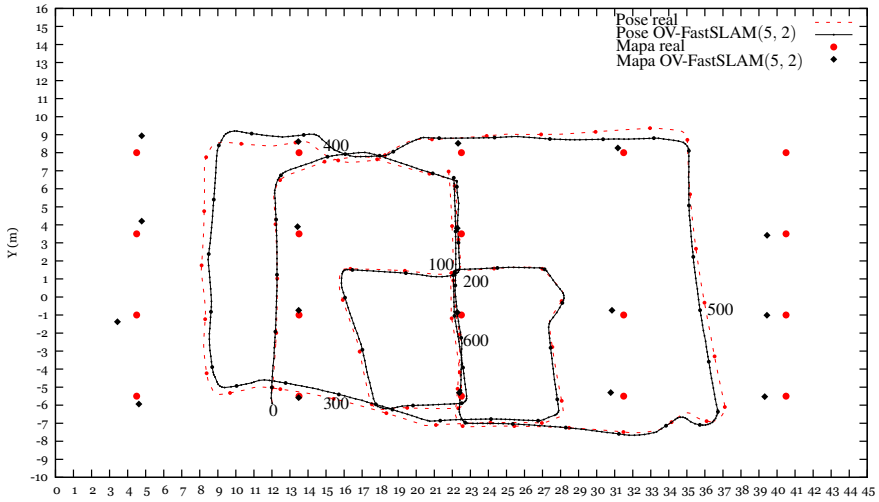
desciende de nuevo hasta valores en torno a $0,25\,m$. Por otra parte, el error en ángulo se mantiene bastante bajo a lo largo del recorrido y, aun cuando hay picos de error, estos se corrigen en dos o tres iteraciones.

En cuanto a los resultados del entorno Pío XII (fig. 5.13)[4] en general, y al igual que sucedía en el entorno Domus, los aumentos del error se deben a la influencia de balizas con una posición inicial incorrecta y, por tanto, una vez que estas balizas dejan de influir significativamente en la corrección de la pose de robot (porque dejan de ser visibles o porque se detectan balizas correctamente posicionadas) el error desciende . El caso más significativo se produce entre las iteraciones 260 y 375, cuando el robot se mueve por la zona situada en el lado izquierdo del entorno ($x \sim= 8\,m$, fig. 5.13(a)) y las balizas situadas a la izquierda ($x \sim= 4{,}5\,m$, fig. 5.13(a)) comienzan a influir en la corrección de la pose del robot. Estas balizas han sido inicializadas recientemente y, además, desde distancias relativamente lejanas, por lo que sus posiciones son imprecisas.

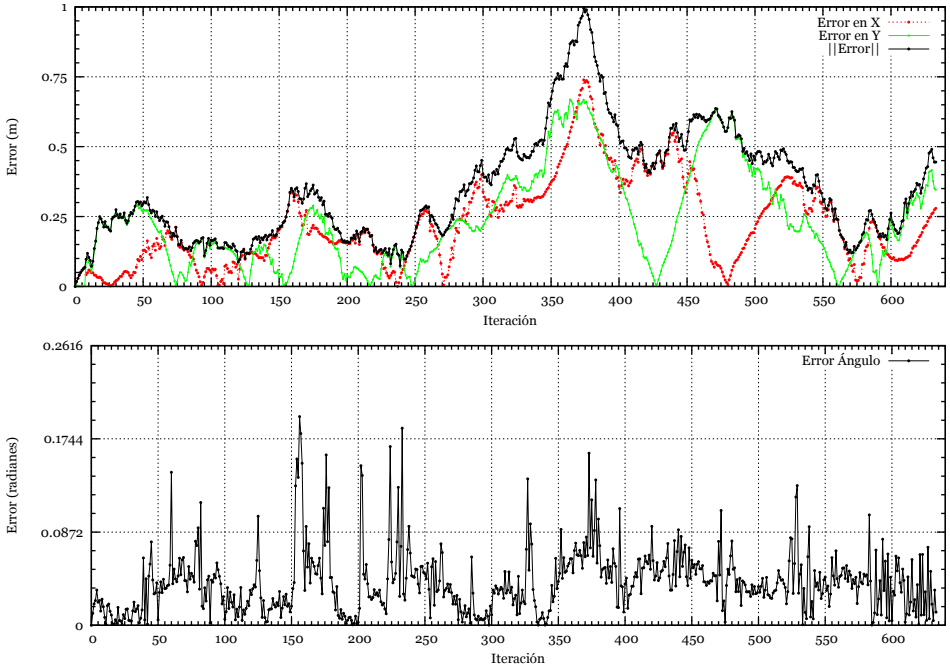
El error comienza a descender cuando se vuelven a observar balizas bien posicionadas que suelen tener una covarianza menor y, por tanto, más influencia en la corrección de la pose del robot. En particular, el error en y se cancela al llegar a la iteración 430 ($x \in [20\,m,\, 23\,m]$ e $y \sim= 9\,m$, fig. 5.13(a)). A partir de ese instante, el robot entra en la zona de la esquina superior derecha ($x \in [28\,m,\, 36\,m]$ e $y \in [7m,\, 9\,m]$, fig. 5.13(a)), donde no cuenta con ninguna baliza inicializada debido a la incidencia de luz natural. Este efecto provoca que el error aumente de nuevo y que todas las balizas de la zona derecha se inicialicen con posiciones con bastante error (como se puede apreciar fácilmente en el vídeo). Una vez que el robot regresa a una zona donde existen balizas bien posicionadas ($x \in [26\,m, 30\,m]$ e $y \in [-6\,m,\, -8\,m]$, fig. 5.13(a)) el sistema se recupera (cierre de lazo), llegando a errores en posición inferiores a $0,15\,m$ (iteración 575).

Una muestra clara de la robustez de OV-FastSLAM(M, N) es su capacidad de reconocer cuando se vuelve a una región mapeada previamente — cierre de lazos — tal y como se ha mostrado en los experimentos presentados. En la trayectoria presentada en el entorno Domus (fig. 5.12(a)), existen dos zonas por las que el robot pasa más de una vez: la zona definida por $x \in [15\,m, 17\,m]$ e $y \in [1\,m, 5\,m]$ y la zona $x \in [3\,m, 17\,m]$ e $y \in [1\,m, 3\,m]$ (fig. 5.12(a)). La trayectoria realizada en el pabellón de deportes Pío XII (fig. 5.13(a)), incluye seis instantes en los que se vuelve al mismo sitio. Todas estas situaciones se resuelven correctamente, reconociendo las balizas ya mapeadas a pesar de que se llega a la zona conocida con un cierto

[4] El vídeo asociado está disponible en `http://persoal.citius.usc.es/cristina.gamallo/Videos/OV-FastSLAM_pio.mp4`.



(a) Mapa y trayectorias reales y estimadas. Los puntos en las trayectorias muestran la pose del robot en iteraciones múltiplos de 10.



(b) Error de la trayectoria en posición y ángulo.

Figura 5.13: Resultados de OV-FastSLAM$(5, 2)$ en el entorno Pío XII.

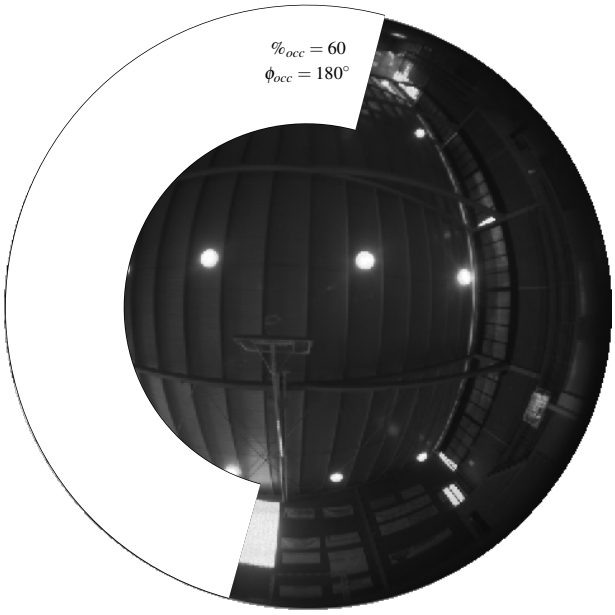


Figura 5.14: Máscara de oclusión con $\phi_{occ} = 60^{\circ}$ y $\%_{occ} = 180^{\circ}$.

error acumulado en la pose. Es de destacar que no ha sido necesario incluir ningún mecanismo específico para detectar los cierres de lazos, como es habitual en la bibliografía.

5.4.3. Oclusiones

OV-FastSLAM ha sido diseñado para operar bajo oclusiones severas. Con el fin de comprobar la influencia de las oclusiones en el rendimiento del algoritmo, se ha evaluado OV-FastSLAM bajo oclusiones continuas con diferentes grados de severidad. Las oclusiones han sido generadas artificialmente mediante la superposición de una máscara en las imágenes, con el objetivo de poder medir para diferentes grados de oclusión la pérdida en el rendimiento del algoritmo. Las máscaras se han construido como la intersección de un sector circular con un anillo (fig. 5.14). Por tanto, una máscara se define con dos parámetros: $\phi_{occ} \in [0^{\circ}, 360^{\circ}]$ y $\%_{occ} \in [0, 100]$. ϕ_{occ} representa el ángulo del sector circular y $\%_{occ}$ el porcentaje del área de la imagen correspondiente al anillo circular.

Se han realizado experimentos en los dos entornos de prueba, para valores de ϕ_{occ} cada 30° y para ocho valores de $\%_{occ}$: $\{10-60, 75, 100\}$. OV-FastSLAM se ejecutó con $M = 5$ y $N_\Psi = 2$, y se comparó con H-H(10). La tabla 5.3 resume el test no paramétrico de Friedman —el test de Holm no es necesario ya que solo se comparan dos algoritmos—. El test indica que ambos algoritmos son diferentes con un nivel de confianza de $\alpha = 0{,}06$ y, dado que el ranking de OV-FastSLAM$(5, 2)$ es mejor, se puede concluir que OV-FastSLAM$(5, 2)$ es superior a H-H(10) en su rendimiento frente a oclusiones severas.

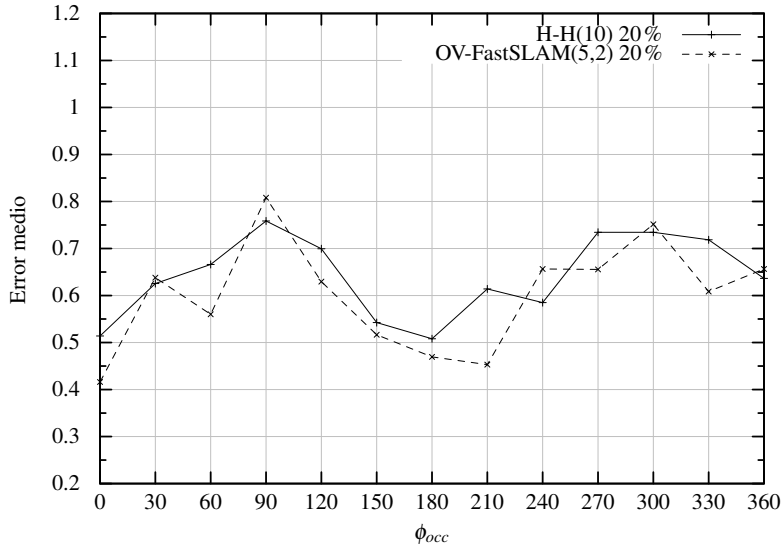
Tabla 5.3: Test no paramétrico del comportamiento de los algoritmos de SLAM bajo oclusiones.

Alg.	**Ranking**
OV-FastSLAM$(5, 2)$	1.3841
H-H(10)	1.6159
Friedman p-value = 0,054083	

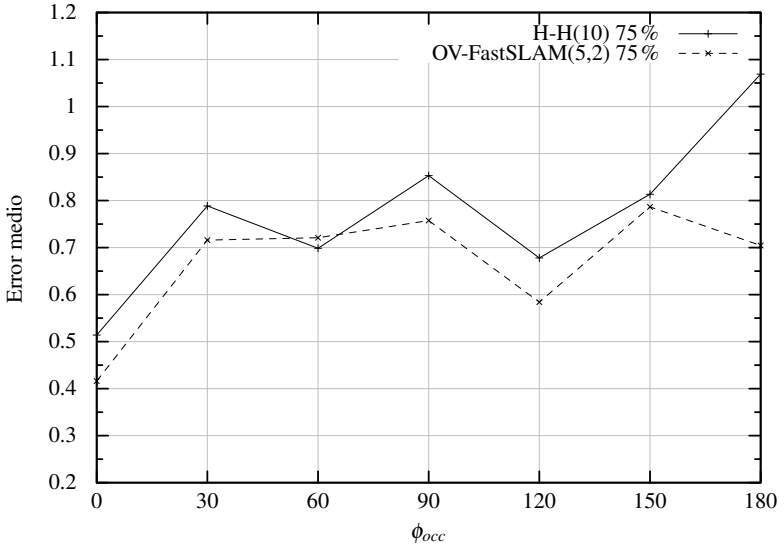
La figura 5.15 muestra el comportamiento de ambos algoritmos para dos valores de $\%_{occ}$ y todo el rango de valores de ϕ_{occ}. Aquellos experimentos en los que el error en ambos algoritmos está por encima de un umbral (dos veces el error mínimo sin oclusión) han sido descartados, ya que se considera que ambos algoritmos fallan.

Para $\%_{occ} = 20$ (fig. 5.15(a)) ambos algoritmos son capaces de operar bajo las oclusiones (hasta $\phi_{occ} = 360$) con un funcionamiento aceptable, aunque OV-FastSLAM$(5, 2)$ es en la mayoría de los casos mejor. Aunque a priori se podría esperar que el error aumentase con el grado de oclusión, la situación es bastante más compleja y es necesario realizar un análisis de resultados bastante más sutil. Por ejemplo, cuando hay dos balizas próximas entre sí, la oclusión de una de ellas puede generar un fallo en la asociación de datos —que la baliza ocluida sea asociada con la medida correspondiente a la baliza visible—. Esto sucede, por ejemplo, con $\phi_{occ} = 90$ (fig. 5.15(a)), pero cuando aumenta el grado de oclusión (y ambas balizas están ocluidas) el error disminuye de nuevo pues no hay error de asociación entre las mismas. Estos errores en la asociación de datos afectan en el comportamiento de los algoritmos debido a su influencia en la inicialización de las balizas, la eliminación de balizas o la corrección de la posición del robot con la información de las balizas detectadas. Para $\%_{occ} = 75$ (oclusión severa) (fig. 5.15(b)), el patrón es similar, pero para $\phi_{occ} \geq 180$ ambos algoritmos fallan ya que la mayoría de las balizas están ocluidas de forma permanente.

La figura 5.16 muestra los valores máximos de ϕ_{occ} para los que el rendimiento de OV-FastSLAM no se degrada significativamente. Como era de esperar, ϕ_{occ} disminuye con el



(a) $\%_{occ} = 20$.



(b) $\%_{occ} = 75$.

Figura 5.15: Comportamiento de los algoritmos de SLAM para diferentes grados de oclusión.

aumento de $\%_{occ}$, pero incluso para $\%_{occ} = 100$, OV-FastSLAM es capaz de obtener un error razonable a pesar de que un tercio de la imagen está ocluida.

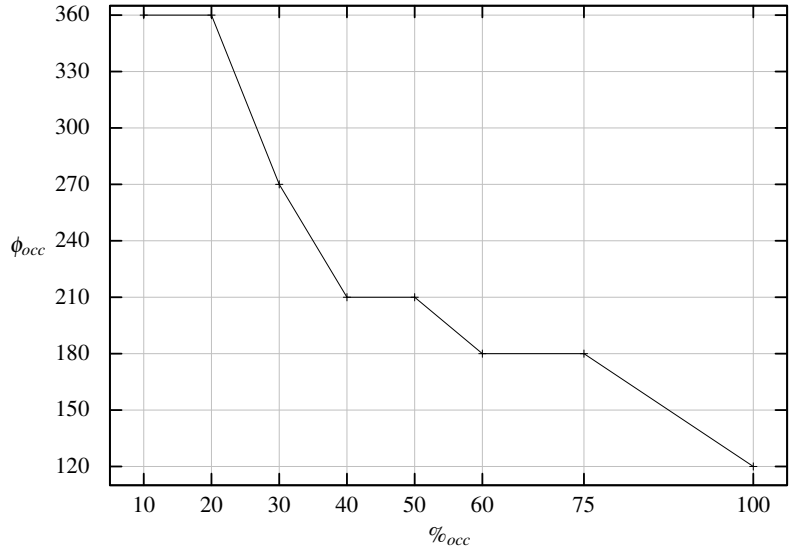


Figura 5.16: Valor máximo de ϕ_{occ} hasta el cual el funcionamiento de OV-FastSLAM no se degrada significativamente.

5.5. Conclusiones

En este capítulo se ha descrito OV-FastSLAM, un algoritmo de SLAM para cámaras omnidireccionales que utiliza las luces del entorno como balizas, que es capaz de operar en entornos con oclusiones severas y continuas, y que puede ejecutarse en tiempo real. Las principales contribuciones de OV-FastSLAM son:

– El modelo inverso de la cámara omnidireccional no es invertible, por lo que ha sido necesario aproximarlo mediante una tabla de referencia que permite obtener las medidas a partir de los píxeles de la imagen.

– Covarianza de las medidas. Se ha definido la incertidumbre asociada a cada medida por medio de una matriz de covarianza específica para cámaras omnidireccionales con lente

tipo *fish-eye*, de tal forma que la incertidumbre depende de la zona de la imagen en la que se encuentre el píxel asociado a la baliza.

– Inicialización de las balizas. Puesto que la cámara omnidireccional, al igual que cualquier otra cámara monocular, es un sensor que solo proporciona información de la orientación en la que se encuentran las balizas, se ha recurrido a un método de inicialización retardada de las mismas. La inicialización utiliza las medidas asociadas a la baliza candidata y selecciona la posición inicial de la baliza que maximiza la probabilidad conjunta de todas esas medidas. El método es bastante restrictivo y ayuda a eliminar balizas candidatas que contienen medidas con un emparejamiento pobre con la posición inicial más probable.

– Asociación de datos jerárquica. El método de asociación de datos ha sido diseñado para operar con cámaras omnidireccionales y oclusiones severas y frecuentes. Sus principales características son:

  - Es global, i. e., es capaz de obtener la asociación de mayor probabilidad conjunta para las medidas y balizas en cada instante. Además, dentro de la propia asociación se integra la creación de nuevas balizas.
  - Es jerárquico, puesto que es necesario asignar las medidas a dos tipos de balizas: las existentes en el mapa y las candidatas. La asociación de primer nivel (balizas) se ha realizado por medio del algoritmo de Murty que permite obtener las n mejores asociaciones en tiempo polinómico. Por otra parte, la asociación de segundo nivel (balizas candidatas) utiliza el método Húngaro que genera la mejor asociación.
  - Permite mantener varias hipótesis de asociación por partícula. Esta característica es especialmente relevante en situaciones en las que existen múltiples asociaciones con probabilidades similares debido, por ejemplo, a balizas relativamente próximas en el espacio de medidas, oclusiones, balizas fuera del campo de detección del sensor, balizas muy alejadas, etc. En todas estas situaciones conviene retrasar la eliminación de hipótesis a la espera de que nuevas medidas clarifiquen cuáles son las hipótesis válidas. En este sentido, mantener varias hipótesis de asociación por partícula permite retrasar la selección de la asignación correcta hasta el momento del remuestreo —que se produce cuando la calidad de la muestra de partículas empeora—.

Se ha validado OV-FastSLAM sobre un robot Pioneer 3-DX en dos entornos interiores (Domus y Pío XII), complejos, con pocas balizas y con características muy diferentes. En el primer caso, el suelo es muy irregular y existen techos a diferentes alturas, lo que provoca que algunas balizas solo se detecten desde zonas muy específicas, bien por estar ocluídas por techos a alturas inferiores, bien porque están situadas en zonas de techo bajo; además, algunas balizas están situadas muy lejos, a más de $13\,m$ de altura. En el segundo entorno (Pío XII) la mayor dificultad es la presencia de luz natural en algunas zonas del entorno, lo que provoca fallos en la detección de las balizas.

De entre todas las pruebas realizas, se ha presentado un análisis muy pormenorizado de dos trayectorias típicas de $72\,m$ (Domus) y $174\,m$ (Pío XII), mostrando que OV-FastSLAM es siempre capaz de cerrar lazos basándose en el método de asociación de datos jeráquico, y consiguiendo errores medios en la posición a lo largo de la trayectoria de $0,24\,m$ (Domus) y $0,34\,m$ (Pío XII) respectivamente. El error medio cometido por OV-FastSLAM$(5,\,2)$ en el conjunto de todas las trayectorias y entornos probados ha sido de $0,42\,m$.

Además, se ha comparado OV-FastSLAM con otras versiones del algoritmo de SLAM que incluyen diferentes métodos de asociación de datos. Todos aquellos que no son jerárquicos o que se basan en asociaciones de máxima probabilidad han conseguido resultados inaceptables, y solo el método H-H (la variante del algoritmo propuesto con asociación con el método Húngaro) ha logrado unos resultados comparables a OV-FastSLAM. Para extraer conclusiones de la comparativa se han aplicado test estadísticos no paramétricos a OV-FastSLAM y a H-H, demostrándose que OV-FastSLAM es mejor, y que las diferencias entre ambos son estadísticamente significativas para $\alpha = 0,05$.

Por último, se ha llevado a cabo un estudio de la robustez de OV-FastSLAM frente a oclusiones severas y continuas. De nuevo, los test estadísticos han confirmado la superioridad de OV-FastSLAM frente a H-H en experimentos con diferentes grados de oclusión. En casos extremos OV-FastSLAM es capaz de operar con un tercio de la imagen permanentemente ocluida sin empeorar significativamente su rendimiento.

CONCLUSIONES

En esta memoria se han presentado dos algoritmos (OV-MCL y OV-FastSLAM) para resolver los problemas de localización y SLAM en robótica móvil con una cámara omnidireccional, utilizando como balizas las luces del entorno, y bajo oclusiones severas y continuas. Estas condiciones de operación son típicas de entornos con gran cantidad de gente que se desplaza próxima al robot, como museos, aeropuertos, estaciones, etc.

Desde el punto de vista de la localización de robots autónomos se ha propuesto OV-MCL, un algoritmo cuya estructura está basada en la localización de Monte Carlo, y cuyas principales contribuciones son:

- El modelo de medida, que permite obtener la probabilidad de que una imagen omnidireccional se corresponda con el mapa de luces para una pose determinada del robot.
- El tamaño adaptativo del conjunto de partículas, calculado por medio de la divergencia de Kullback-Leibler, y que optimiza el uso de recursos computacionales, aumentando la robustez de la localización.
- La inserción de partículas aleatorias, para conseguir mayor fiabilidad y robustez frente a cambios en el entorno y oclusiones severas y continuas.

OV-MCL ha sido validado en un entorno real complejo para los tres problemas de localización (local, global y *secuestro*), así como bajo oclusiones severas y continuas. En todos los casos OV-MCL ha mostrado una gran robustez y precisión (error medio $= 0,29m$), destacando que: i) solo necesita unos pocos segundos para recuperarse de un *secuestro*; ii) con oclusiones severas y continuas el sistema opera con normalidad hasta un grado de oclusión del $50\,\%$.

La propuesta presentada para resolver el problema de SLAM, OV-FastSLAM, se basa en el algoritmo FastSLAM 2.0. Las principales aportaciones de OV-FastSLAM son:

– El modelo inverso de la cámara omnidireccional, que permite obtener las medidas a partir de los píxeles de la imagen.

– El modelado de la incertidumbre asociada a las medidas, específico para cámaras omnidireccionales de tipo *fish-eye*, que establece diferentes matrices de covarianza en función de la zona de la imagen en la que se encuentre la baliza.

– El algoritmo de inicialización de las balizas, que establece como posición inicial aquella que maximiza la probabilidad conjunta de las medidas asociadas a la baliza candidata.

– La asociación de datos jerárquica, global y que mantiene varias hipótesis de asociación por partícula. Está basada en el algoritmo de Murty y el método Húngaro, y permite retrasar la eliminación de hipótesis en aquellas situaciones en las que varias asociaciones tienen probabilidades similares y hasta que nuevas medidas clarifiquen cuáles son las hipótesis más probables. Esta situación es relativamente frecuente en entornos en los que las medidas de las diferentes balizas son muy similares y cuando las oclusiones son habituales.

OV-FastSLAM ha sido validado en dos entornos complejos y con características diferentes, mostrando una gran precisión (error medio $= 0{,}42\,m$), robustez y capacidad de cierre de lazos. Además, se han realizado test estadísticos no paramétricos que demuestran la superioridad de OV-FastSLAM frente a otras variantes del algoritmo con diferentes métodos de asociación. Finalmente, también se ha hecho un estudio experimental de la robustez de OV-FastSLAM frente a oclusiones de diferente grado. OV-FastSLAM ha sido capaz de trabajar con oclusiones permanentes de un tercio de la imagen sin empeorar significativamente su rendimiento, aun cuando el número de balizas detectadas en cada imagen es bajo.

Futuras líneas de investigación

Para dar continuidad al trabajo presentado en esta memoria se han identificado una serie de líneas de investigación que sería interesante explorar en el futuro:

– Extender los modelos para realizar la estimación de la pose del robot en 3D (6 grados de libertad), lo que permitiría la aplicación de los algoritmos en otros ámbitos como la robótica aérea o submarina.

– Realizar SLAM multi-robot, de tal forma que se facilite la exploración de áreas grandes.

– Analizar la detección y el reconocimiento de otros objetos significativos del entorno, de manera que los algoritmos puedan trabajar con distintos tipos de balizas. En esta línea, también se podrían incorporar las propiedades de cada una de las balizas en el proceso jerárquico de asociación de datos.

– Incorporar otros sensores (láser, cámara estéreo) que complementen la información proporcionada por la cámara omnidireccional. Esto implicará no solo modificar los algoritmos, sino también realizar la fusión de información sensorial siguiendo un marco Bayesiano.

APÉNDICE A

FastSLAM

El algoritmo FastSLAM utiliza un filtro de partículas de tipo Rao-Blackwell para resolver el problema de SLAM. En este apéndice se describen las dos versiones básicas de FastSLAM: FastSLAM 1.0 y FastSLAM 2.0. La principal diferencia entre ellas se encuentra en la estimación de la pose del robot, aunque esta fase también influye en las etapas de actualización de las medidas y en el cálculo del peso de las partículas. En los algoritmos descritos en este apéndice, y por simplicidad, se asume que en cada instante de tiempo solo se recibe una medida.

A.1. FastSLAM 1.0

El algoritmo FastSLAM 1.0 (alg. A.1) recibe como entradas la medida actual (z_t), el control (u_t) y el conjunto previo de partículas (Y_{t-1}). Cada partícula k (fig. A.1) contiene información de la pose del robot (x_t^k) y el mapa, definido como un conjunto de balizas ($B_{t-1,1}^k, B_{t-1,2}^k, \ldots, B_{t-1,j}^k, \ldots, B_{t-1,N_t^k}^k$) de las que se conoce su posición —definida por medio de una gaussiana de media $\mu_{t-1,j}^k$ y covarianza $\Sigma_{t-1,j}^k$— y el número de veces que han sido detectadas ($i_{t-1,j}^k$). FastSLAM 1.0 consta de las siguientes etapas:

- **Actualización de la pose del robot** (alg. A.1, línea 3). Se muestrea la nueva pose del robot (x_t^k) para cada partícula en Y_{t-1} en función de la pose del instante anterior x_{t-1}^k y teniendo en cuenta la PDF del modelo de movimiento.

- **Probabilidad de asociación de la medida** (alg. A.1, línea 4). Se calcula la probabilidad de que la medida sea asociada a cada una de las balizas del mapa (Φ_t^k).

Algoritmo A.1 FastSLAM 1.0 (z_t, u_t, Y_{t-1}) [Thrun y col., 2003]

1: **for** $k = 1$ to M **do**
2: Seleccionar la partícula k de Y_{t-1}: $x^k_{t-1}, N^k_{t-1}, \{B^k_{t-1,1}, \ldots, B^k_{t-1,N^k_t}\}$
3: $x^k_t \sim p\left(x_t | x^k_{t-1}, u_t\right)$ ▷ Actualización de la pose
4: $\Phi^k_t = probabilidadMedidas\,()$
5: $\Psi^k_t = argmax\,\Phi^k_t$ ▷ Asociación de datos
6: $actualizaciónMapa\,()$
7: Añadir $x^k_t, N^k_t, \{B^k_{t,1}, \ldots, B^k_{t,N^k_t}\}$ a $\overline{Y}_t$
8: **end for**
9: $Y_t = remuestreo\left(\overline{Y}_t\right)$
10: **return** Y_t

Partícula	Pose del robot	$B_{t,1}$	$\cdots$	$B_{t,j}$	$\cdots$	B_{t,N_t}
1	$x^1_t = (x\ y\ \alpha)^{T\,1}_t$	$\mu^1_{t,1}, \Sigma^1_{t,1}, i^1_{t,1}$	$\cdots$	$\mu^1_{t,j}, \Sigma^1_{t,j}, i^1_{t,j}$	$\cdots$	$\mu^1_{t,N_t}, \Sigma^1_{t,N_t}, i^1_{t,N_t}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
k	$x^k_t = (x\ y\ \alpha)^{T\,k}_t$	$\mu^k_{t,1}, \Sigma^k_{t,1}, i^k_{t,1}$	$\cdots$	$\mu^k_{t,j}, \Sigma^k_{t,j}, i^k_{t,j}$	$\cdots$	$\mu^k_{t,N_t}, \Sigma^k_{t,N_t}, i^k_{t,N_t}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
M	$x^M_t = (x\ y\ \alpha)^{T\,M}_t$	$\mu^M_{t,1}, \Sigma^M_{t,1}, i^M_{t,1}$	$\cdots$	$\mu^M_{t,j}, \Sigma^M_{t,j}, i^M_{t,j}$	$\cdots$	$\mu^M_{t,N_t}, \Sigma^M_{t,N_t}, i^M_{t,N_t}$

Figura A.1: Estructura de las partículas en FastSLAM [Thrun y col., 2003].

– **Asociación de datos** (alg. A.1, línea 5). En base a la probabilidad de asociación estimada para cada baliza (Φ^k_t), se resuelve la asociación de datos Ψ^k_t para cada partícula. La medida será asignada a la baliza con mayor probabilidad de asociación.

– **Actualización del mapa** (alg. A.1, línea 6). Se estima el factor de importancia o peso de cada partícula (ω^k) y se actualizan las balizas del mapa para incorporar la información de la medida actual z_t.

– **Remuestreo**. Se genera el nuevo conjunto de partículas (Y_t) a partir del conjunto de partículas actualizado ($\overline{Y}_t$), y en base a probabilidades proporcionales a los pesos de las partículas (ω^k).

Algoritmo A.2 FastSLAM 1.0: *probabilidadMedidas* ()

1: **for** $j = 1$ to N_{t-1}^k **do**
2: $\quad \bar{z}_j = h\left(\mu_{t-1,\,j}^k, x_t^k\right)$
3: $\quad H_{m,j} = \nabla_m h\left(\mu_{t-1,\,j}^k, x_t^k\right)$
4: $\quad Q_j = Q_t + H_{m,j}\Sigma_{t-1,\,j}^k {H_{m,j}}^T$
5: $\quad \phi_j = (2\pi)^{-\frac{Dim\left(Q_j\right)}{2}} \left|Q_j\right|^{-\frac{1}{2}} \exp\{-\frac{1}{2}(z_t - \bar{z}_j)^T Q_j^{-1}(z_t - \bar{z}_j)\}$
6: **end for**
7: $\phi_{N_{t-1}^k+1} = \phi_{new}$

A continuación se explican en detalle aquellas etapas para las que FastSLAM presenta una solución propia.

Probabilidad de asociación de la medida

La probabilidad de asociación de la medida (Φ_t^k) es un vector que contiene las probabilidades (ϕ_j) de que la medida z_t haya sido generada por cada una de las balizas del mapa $B_{t,\,j}^k$ (alg. A.2).

Cada probabilidad se calcula en base a una distribución gaussiana de media $\bar{z}_j$ y covarianza Q_j. $\bar{z}_j$ es la predicción de la medida, determinada a partir del modelo de medida y la pose estimada (x_t^k). Q_j es la matriz de covarianza de la medida (alg. A.2, línea 4), y se calcula teniendo en cuenta el ruido en la medida (Q_t), la covarianza previa de la baliza ($\Sigma_{t-1,\,j}^k$) y la matriz Jacobiana de h con respecto a las variables del vector de medida (alg. A.2, línea 3). Por último, en la línea 7 se añade la probabilidad de que la medida haya sido originada por una nueva baliza, no incluida en el mapa hasta el instante actual.

Actualización del mapa

La fase de actualización del mapa (alg. A.3) consta de tres partes, correspondiéndose con los distintos resultados de la etapa de asociación de datos:

1. **Medida asignada a una baliza ya existente** (alg. A.3,líneas 2 - 7). En este caso la actualización de las balizas sigue el proceso estándar de actualización de un EKF. Primero se obtiene la ganancia de Kalman (K) usando la covarianza previa de la baliza y

Algoritmo A.3 FastSLAM 1.0: *actualizaciónMapa*()

for $j = 1$ to N_{t-1}^k **do**
  if $\psi_j > 0$ **then** ▷ Baliza asociada a una medida
    $K = \Sigma_{t-1,\,j}^k H_{m,j}^T Q_j^{-1}$
    $\mu_{t,\,j}^k = \mu_{t-1,\,j}^k + K\left(z_t - \bar{z}_j\right)$
    $\Sigma_{t,\,j}^k = (I - KH_{m,j})\,\Sigma_{t-1,\,j}^k$
    $i_{t,\,j}^k = i_{t-1,\,j}^k + 1$
    $N_t^k = N_{t-1}^k$
  else ▷ Baliza no observada
    $\mu_{t,\,j}^k = \mu_{t-1,\,j}^k$
    $\Sigma_{t,\,j}^k = \Sigma_{t-1,\,j}^k$
    if $\mu_{t-1,\,j}^k$ está fuera del rango perceptual de x_t^k **then**
      $i_{t,\,j}^k = i_{t-1,\,j}^k$
    else
      $i_{t,\,j}^k = i_{t-1,\,j}^k - 1$
      if $i_{t,\,j}^k < 0$ **then**
        eliminar baliza j
      end if
    end if
  end if
end for
if $\Psi_t^k > N_{t-1}^k$ **then** ▷ Nueva baliza
  $\mu_{t,\,\Psi_t^k}^k = h^{-1}\left(z_t, x_t^k\right)$
  $H_m = \nabla_m h\left(\mu_{t,\,\Psi_t^k}^k, x_t^k\right)$
  $\Sigma_{t,\,\Psi_t^k}^k = H_m^{-1} Q_t (H_m)^T$
  $i_{t,\,\Psi_t^k}^k = 1$
  $N_t^k = N_{t-1}^k + 1$
end if
$\omega^k = max\,\Phi_t^k$

la matriz de covarianza de la medida (Q_j). La posición de la baliza se actualiza proporcionalmente a la ganancia y a la diferencia entre la medida y su predicción. Si la ganancia de Kalman es elevada indica que la confianza en la actualización es alta. Esto puede ocurrir si la covarianza previa de la baliza era alta (baja confianza en la posición actual), por lo que en la actualización la medida actual tiene más peso. También sucede

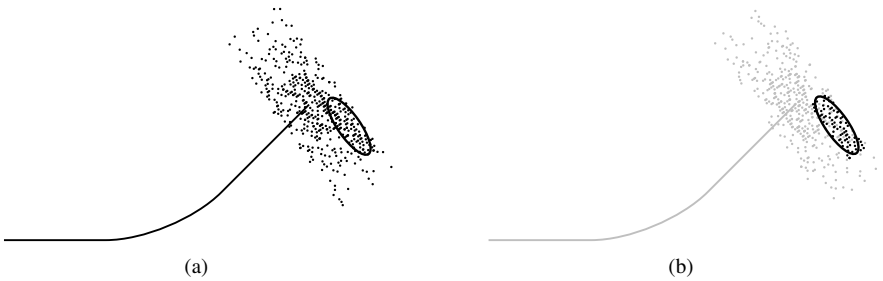


Figura A.2: Diferencias entre la distribución a priori y la distribución a posteriori.

si la matriz de covarianza de la medida es alta (alta confianza en la medida). Además, $i^k_{t,j}$ se incrementa, ya que la baliza ha sido detectada.

2. **Baliza sin medida asignada** (alg. A.3, líneas 8 - 19). La posición y la covarianza de la baliza se mantienen invariables. Para actualizar el número de observaciones hay dos posibilidades:

   a) Si la baliza está fuera del rango de percepción del sensor no se modifica el contador $i^k_{t,j}$ (alg. A.3, línea 12).

   b) Si la baliza está dentro del rango de percepción del sensor, el contador $i^k_{t,j}$ se decrementa (alg. A.3, línea 14). Si el contador alcanza un valor por debajo de 0, la baliza se elimina, lo que permite tratar falsos positivos y mantener en el mapa un número razonable de balizas.

3. **Medida asociada a una nueva baliza** (alg. A.3, líneas 21 - 27). Si la probabilidad de asociar la medida con cada una de las balizas del mapa es inferior a un umbral (ϕ_{new}), se crea una baliza nueva. Para ello se genera un nuevo EKF cuya media se inicializa a partir de la medida y la pose del robot, y su covarianza es proporcional a la covarianza del ruido de la medida (Q_t).

Finalmente, se determina el peso de la partícula (ω^k) (alg. A.3, línea 28), que se corresponderá con el valor máximo del vector de probabilidad de asociación de la medida.

Limitaciones de FastSLAM 1.0

La principal limitación de FastSLAM 1.0 es que la pose del robot se muestrea teniendo en cuenta solo la acción de control, y se utiliza la información de la medida únicamente para calcular el peso de la partícula. Este aspecto es especialmente problemático cuando la fiabilidad del sensor es más alta que la fiabilidad del control, dando como resultado que la PDF a posteriori, generada mediante el muestreo de la PDF a priori, desaprovecha muchas partículas y puede dar lugar a una aproximación pobre de la PDF real. En la figura A.2 se presenta un ejemplo en el que la distribución de probabilidad a priori genera un largo espectro de muestras, pero solamente un pequeño subconjunto de las mismas (las incluidas en la elipse) tienen un peso elevado y sobreviven al remuestreo.

Esta limitación se resuelve con FastSLAM 2.0, puesto que las poses del robot se muestrean en base al movimiento corregido por las observaciones. En general, FastSLAM 2.0 genera conjuntos de partículas con pesos medios más elevados, lo que produce una mayor diversidad tras el remuestreo.

A.2. FastSLAM 2.0

FastSLAM 2.0 [Montemerlo y col., 2002] obtiene la pose del robot a partir del muestreo de una PDF que ha sido generada teniendo en cuenta tanto el movimiento del robot como las observaciones. Para ello es necesario introducir cambios en algunas de las etapas principales del algoritmo (alg. A.4):

1. **Probabilidad de asociación de la medida** (alg. A.4, línea 4). Se determina la probabilidad de asociación de cada baliza con la medida, pero generando la predicción de la medida a partir de una pose del robot corregida.

2. **Actualización de la pose del robot** (alg. A.4, línea 6). Se estima la pose teniendo en cuenta la incorporación de la medida.

3. **Actualización del mapa** (alg. A.4, línea 7). Se integran las etapas de cálculo del factor de importancia y actualización de las balizas, ya que la estimación del peso de la partícula ha de reflejar los cambios realizados en la pose del robot.

Algoritmo A.4 FastSLAM 2.0 (z_t, u_t, Y_{t-1}) [Thrun y col., 2003]. En texto negro se marcan las etapas que presentan diferencias con FastSLAM 1.0. (alg. A.1)

for $k = 1$ to M **do**
  Seleccionar la partícula k de Y_{t-1}: $x_{t-1}^k, N_{t-1}^k, \{B_{t-1,1}^k, \ldots, B_{t-1,N_t^k}^k\}$
  $\widehat{x}_t = g\left(x_{t-1}^k, u_t\right)$ ▷ Predicción de la pose
  $\Phi_t^k = probabilidadMedidas\,()$
  $\Psi_t^k = argmax\,\Phi_t^k$ ▷ Asociación de datos
  $actualizaciónPoseRobot\,()$
  $actualizaciónMapa\,()$
  Añadir $x_t^k, N_t^k, \{B_{t,1}^k, \ldots, B_{t,N_t^k}^k\}$ a $\overline{Y}_t$
end for
$Y_t = remuestreo\left(\overline{Y}_t\right)$
return Y_t

Algoritmo A.5 FastSLAM 2.0: *probabilidadMedidas*(). En negro se destacan las diferencias con el algoritmo A.2

for $j = 1$ to N_{t-1}^k **do**
  $\overline{z}_j = h\left(\mu_{t-1,j}^k, x_t^k\right)$
  $H_{x,j} = \nabla_{x_t} h\left(\mu_{t-1,j}^k, x_t^k\right)$
  $H_{m,j} = \nabla_m h\left(\mu_{t-1,j}^k, x_t^k\right)$
  $Q_j = Q_t + H_{m,j}\Sigma_{t-1,j}^k {H_{m,j}}^T$
  $\Sigma_{x,j} = [{H_{x,j}}^T Q_j^{-1} H_{x,j} + R_t^{-1}]^{-1}$
  $\mu_{x_t,j} = \Sigma_{x,j} {H_{x,j}}^T Q_j^{-1}\,(z_t - \overline{z}_j) + \widehat{x}_{t,j}$
  $x_{t,j}^k \sim N\,(\mu_{x_t,j}, \Sigma_{x,j})$
  $\widehat{z}_j = h\left(\mu_{t-1,j}, x_{t,j}^k\right)$
  $\phi_j = (2\pi)^{-\frac{Dim\left(Q_j\right)}{2}} \left|Q_j\right|^{-\frac{1}{2}} \exp\{-\frac{1}{2}(z_t - \widehat{z}_j)^T Q_j^{-1}\,(z_t - \widehat{z}_j)\}$
end for
$\phi_{1+N_{t-1}^k} = \phi_{new}$

Probabilidad de asociación de la medida

Para obtener la probabilidad de asociación entre la medida y cada una de las balizas es necesario conocer la pose del robot. Sin embargo, en FastSLAM 2.0 esta pose se obtiene a partir de una PDF que incorpora la corrección propuesta por la medida y, por tanto, no puede

Algoritmo A.6 FastSLAM 2.0: *actualizaciónPoseRobot*()

1: **if** $\Psi_t^k <= N_{t-1}^k$ **then**
2: $\quad x_t^k = x_{t,\Psi_t^k}^k$
3: **else**
4: $\quad x_t^k \sim p\left(x_t | x_{t-1}^k, u_t\right)$
5: **end if**

ser generada hasta que no se resuelve la asociación de datos. Por este motivo, el cálculo de la probabilidad de asociación para cada baliza requiere crear una pose del robot muestreada ($x_{t,j}^k$, alg. A.5, línea 8) por cada posible asociación.

La PDF de la que se muestrea la pose del robot es una gaussiana cuya covarianza ($\Sigma_{x,j}$) depende de dos términos: el ruido del modelo de movimiento (R_t) y la matriz de covarianza de la medida (Q_j). Por otra parte, la media de la gaussiana ($\mu_{x_t,j}$) se determina a partir del modelo de movimiento ($\widehat{x}_{t,j}$) corregido por la diferencia entre la medida (z_t) y la estimación de la medida respecto a la baliza j ($\overline{z}_j$). Esta corrección es proporcional a la matriz $\Sigma_{x,j} {H_{x,j}}^T Q_j^{-1}$, que puede ser interpretada como una ganancia, en el mismo sentido que la ganancia de Kalman.

El cálculo de la probabilidad (ϕ_j) de que la medida z_t se corresponda con la baliza j (alg. A.5, línea 10) depende de la pose muestreada y de la predicción actualizada de la medida ($\widehat{z}_j$), puesto que la predicción de la medida ($\overline{z}_j$) se determinó utilizando una pose del robot que tenía únicamente en cuenta el modelo de movimiento.

Actualización de la pose del robot

La pose del robot (x_t^k) se corresponderá con la calculada previamente a partir de la baliza con la que ha sido asociada (alg. A.6, línea 2). Si la medida no se asocia con ninguna de las balizas del mapa, la pose se muestrea teniendo en cuenta solo el modelo de movimiento (alg. A.6, línea 4), tal y como se hace en FastSLAM 1.0.

Actualización del mapa

El cálculo de los pesos (ω^k) en FastSLAM 2.0 (alg. A.7) utiliza una matriz de covarianza (L) que tiene en cuenta, además del ruido de la medida (Q_t) y la covarianza previa de la baliza ($\Sigma_{t-1,j}^k$), el ruido del modelo de movimiento (R_t).

Algoritmo A.7 FastSLAM 2.0: *actualizaciónMapa*(). En negro se destacan las diferencias con el algoritmo A.3.

for $j = 1$ to N_t^k **do**
  if $\psi_j > 0$ **then** ▷ Baliza asociada a una medida
    $i_{t,j}^k = i_{t-1,j}^k + 1$
    $K = \Sigma_{t-1,j}^k H_{m,j}^T Q_j^{-1}$
    $\mu_{t,j}^k = \mu_{t-1,j}^k + K(z_t - \widehat{z}_j)$
    $\Sigma_{t,j}^k = (I - KH_{m,j})\Sigma_{t-1,j}^k$
    $L = H_{x,j} R_t H_{x,j}^T + H_{m,j} \Sigma_{t-1,j}^k H_{m,j}^T + Q_t$
    $\omega^k = (2\pi)^{-\frac{Dim(L)}{2}} |L|^{-\frac{1}{2}} \exp\{-\frac{1}{2}(z_t - \widehat{z}_j)^T L^{-1} (z_t - \widehat{z}_j)\}$
    $N_t^k = N_{t-1}^k$
  else ▷ Baliza no observada
    $\mu_{t,j}^k = \mu_{t-1,j}^k$
    $\Sigma_{t,j}^k = \Sigma_{t-1,j}^k$
    if $\mu_{t-1,j}^k$ está fuera del rango perceptual de x_t^k **then**
      $i_{t,j}^k = i_{t-1,j}^k$
    else
      $i_{t,j}^k = i_{t-1,j}^k - 1$
      if $i_{t,j}^k < 0$ **then**
        eliminar baliza j
      end if
    end if
  end if
end for
if $\Psi_t^k > N_{t-1}^k$ **then** ▷ Nueva baliza
  $i_{t,j}^k = 1$
  $\mu_{t,j}^k = h^{-1}(z_t, x_t^k)$
  $H_m = \nabla_m h\left(\mu_{t,j}^k, x_t^k\right)$
  $\Sigma_{t,j}^k = \left(H_{m,j}^{-1}\right)^T Q_t H_m^{-1}$
  $N_t^k = N_{t-1}^k + 1$
  $\omega^k = \phi_{new}$
end if

APÉNDICE B

MATRICES JACOBIANAS DEL MODELO DE MEDIDA

En este apéndice se presentan las matrices Jacobianas del modelo de medida respecto a las variables de estado del robot (H_x) y las variables de la medida (H_m). Puesto que el modelo de medida ($h(B_j^C, x_t)$, ec. 3.1, sec. 3.1.2) está definido como función de la posición de las balizas en el sistema de referencia de la cámara ($B_j^C = (x_{B_j}^C \, y_{B_j}^C \, z_{B_j}^C)$), las matrices Jacobianas se calculan como:

$$H_x = \frac{\partial h}{\partial B_j^C} \cdot \frac{\partial B_j^C}{\partial x_t} \tag{B.1}$$

$$H_m = \frac{\partial h}{\partial B_j^C} \cdot \frac{\partial B_j^C}{\partial B_j^W} \tag{B.2}$$

El primer término en ambas ecuaciones es la derivada parcial de h con respecto a la posición de la baliza en el sistema de referencia de la cámara, mientras que el segundo término se estima derivando B_j^C respecto a la pose del robot ($x_t = (x \, y \, \alpha)$) para H_x, y respecto a la posición de la baliza en el sistema de referencia del mundo ($B_j^W = (x_{B_j} \, y_{B_j} \, z_{B_j})$) para H_m.

La posición de una baliza en el sistema de referencia de la cámara se obtiene mediante una simple transformación de coordenadas:

$$
\begin{aligned}
x_{B_j}^C &= \cos\gamma_C^R \left[\cos\alpha_C^R \left((y_{B_j}-y)\sin\alpha + (x_{B_j}-x)\cos\alpha - x_C^R\right)\right. \\
&\quad \left. + \sin\alpha_C^R \left((y_{B_j}-y)\cos\alpha - (x_{B_j}-x)\sin\alpha - y_C^R\right)\right] \\
&\quad - \sin\gamma_C^R \left\{(z_{B_j}-z_C^R)\cos\beta_C^R - \sin\beta_C^R \right. \\
&\quad \left[\cos\alpha_C^R \left((y_{B_j}-y)\cos\alpha - (x_{B_j}-x)\sin\alpha - y_C^R\right)\right. \\
&\quad \left. \left. - \sin\alpha_C^R \left((y_{B_j}-y)\sin\alpha + (x_{B_j}-x)\cos\alpha - x_C^R\right)\right]\right\} \\
y_{B_j}^C &= (z_{B_j}-z_C^R)\sin\beta_C^R + \cos\beta_C^R \left[\cos\alpha_C^R \left((y_{B_j}-y)\cos\alpha - (x_{B_j}-x)\sin\alpha - y_C^R\right)\right. \\
&\quad \left. - \sin\alpha_C^R \left((y_{B_j}-y)\sin\alpha + (x_{B_j}-x)\cos\alpha - x_C^R\right)\right] \\
z_{B_j}^C &= \cos\gamma_C^R \left\{(z_{B_j}-z_C^R)\cos\beta_C^R - \sin\beta_C^R \left[\cos\alpha_C^R \left((y_{B_j}-y)\cos\alpha \right.\right.\right. - (x_{B_j}-x)\sin\alpha - y_C^R\right) \\
&\quad \left. \left. - \sin\alpha_C^R \left((y_{B_j}-y)\sin\alpha + (x_{B_j}-x)\cos\alpha - x_C^R\right)\right]\right\} \\
&\quad + \sin\gamma_C^R \left[\cos\alpha_C^R \left((y_{B_j}-y)\sin\alpha + (x_{B_j}-x)\cos\alpha - x_C^R\right)\right. \\
&\quad \left. + \sin\alpha_C^R \left((y_{B_j}-y)\cos\alpha - (x_{B_j}-x)\sin\alpha - y_C^R\right)\right]
\end{aligned}
\tag{B.3}
$$

donde $\left(x_C^R\, y_C^R\, z_C^R\, \alpha_C^R\, \gamma_C^R\, \beta_C^R\right)$ representa la traslación y la rotación de la cámara respecto al sistema de referencia del robot.

La derivada de h con respecto a la posición de la baliza en el sistema de referencia de la cámara es una matriz de dimensión 2×3 definida como:

$$
\begin{aligned}
\partial h/\partial B_{j\,1,1}^C &= -\frac{y_{B_j}^C}{{x_{B_j}^C}^2\left(\frac{{y_{B_j}^C}^2}{{x_{B_j}^C}^2}+1\right)} \\
\partial h/\partial B_{j\,1,2}^C &= \frac{1}{x_{B_j}^C\left(\frac{{y_{B_j}^C}^2}{{x_{B_j}^C}^2}+1\right)} \\
\partial h/\partial B_{j\,1,3}^C &= 0 \\
\partial h/\partial B_{j\,2,1}^C &= \frac{x_{B_j}^C z_{B_j}^C}{\left({z_{B_j}^C}^2+{y_{B_j}^C}^2+{x_{B_j}^C}^2\right)^{\frac{3}{2}}\sqrt{1-\frac{{z_{B_j}^C}^2}{{z_{B_j}^C}^2+{y_{B_j}^C}^2+{x_{B_j}^C}^2}}} \\
\partial h/\partial B_{j\,2,2}^C &= \frac{y_{B_j}^C z_{B_j}^C}{\left({z_{B_j}^C}^2+{y_{B_j}^C}^2+{x_{B_j}^C}^2\right)^{\frac{3}{2}}\sqrt{1-\frac{{z_{B_j}^C}^2}{{z_{B_j}^C}^2+{y_{B_j}^C}^2+{x_{B_j}^C}^2}}}
\end{aligned}
\tag{B.4}
$$

$$\partial h/\partial B^C_{j\,2,3} = -\frac{\frac{1}{\sqrt{{z^C_{B_j}}^2+{y^C_{B_j}}^2+{x^C_{B_j}}^2}} - \frac{{z^C_{B_j}}^2}{\left({z^C_{B_j}}^2+{y^C_{B_j}}^2+{x^C_{B_j}}^2\right)^{\frac{3}{2}}}}{\sqrt{1-\frac{{z^C_{B_j}}^2}{{z^C_{B_j}}^2+{y^C_{B_j}}^2+{x^C_{B_j}}^2}}}$$

Por otra parte, la derivada de B^C_j respecto a la pose del robot es una matriz de dimensión 3×3:

$$\begin{aligned}
\partial B^C_j/\partial {x_t}_{1,1} &= \cos\gamma^R_C\left(\sin\alpha^R_C\sin\alpha-\cos\alpha^R_C\cos\alpha\right)\\
&\quad+\sin\gamma^R_C\sin\beta^R_C\left(\cos\alpha^R_C\sin\alpha+\sin\alpha^R_C\cos\alpha\right)\\
\partial B^C_j/\partial {x_t}_{1,2} &= \sin\gamma^R_C\sin\beta^R_C\left(\sin\alpha^R_C\sin\alpha-\cos\alpha^R_C\cos\alpha\right)\\
&\quad+\cos\gamma^R_C\left(-\cos\alpha^R_C\sin\alpha-\sin\alpha^R_C\cos\alpha\right)\\
\partial B^C_j/\partial {x_t}_{1,3} &= \cos\gamma^R_C\left[\sin\alpha^R_C\left(-(y_{B_j}-y)\sin\alpha-(x_{B_j}-x)\cos\alpha\right)\right.\\
&\quad\left.+\cos\alpha^R_C\left((y_{B_j}-y)\cos\alpha-(x_{B_j}-x)\sin\alpha\right)\right]\\
&\quad+\sin\gamma^R_C\sin\beta^R_C\left[\cos\alpha^R_C\left(-(y_{B_j}-y)\sin\alpha-(x_{B_j}-x)\cos\alpha\right)\right.\\
&\quad\left.-\sin\alpha^R_C\left((y_{B_j}-y)\cos\alpha-(x_{B_j}-x)\sin\alpha\right)\right]\\
\partial B^C_j/\partial {x_t}_{2,1} &= \cos\beta^R_C\left(\cos\alpha^R_C\sin\alpha+\sin\alpha^R_C\cos\alpha\right)\\
\partial B^C_j/\partial {x_t}_{2,2} &= \cos\beta^R_C\left(\sin\alpha^R_C\sin\alpha-\cos\alpha^R_C\cos\alpha\right)\\
\partial B^C_j/\partial {x_t}_{2,3} &= \cos\beta^R_C\left[\cos\alpha^R_C\left(-(y_{B_j}-y)\sin\alpha-(x_{B_j}-x)\cos\alpha\right)\right.\\
&\quad\left.-\sin\alpha^R_C\left((y_{B_j}-y)\cos\alpha-(x_{B_j}-x)\sin\alpha\right)\right]\\
\partial B^C_j/\partial {x_t}_{3,1} &= \sin\gamma^R_C\left(\sin\alpha^R_C\sin\alpha-\cos\alpha^R_C\cos\alpha\right)\\
&\quad-\cos\gamma^R_C\sin\beta^R_C\left(\cos\alpha^R_C\sin\alpha+\sin\alpha^R_C\cos\alpha\right)\\
\partial B^C_j/\partial {x_t}_{3,2} &= \sin\gamma^R_C\left(-\cos\alpha^R_C\sin\alpha-\sin\alpha^R_C\cos\alpha\right)\\
&\quad-\cos\gamma^R_C\sin\beta^R_C\left(\sin\alpha^R_C\sin\alpha-\cos\alpha^R_C\cos\alpha\right)\\
\partial B^C_j/\partial {x_t}_{3,3} &= \sin\gamma^R_C\left[\sin\alpha^R_C\left(-(y_{B_j}-y)\sin\alpha-(x_{B_j}-x)\cos\alpha\right)\right.\\
&\quad\left.+\cos\alpha^R_C\left((y_{B_j}-y)\cos\alpha-(x_{B_j}-x)\sin\alpha\right)\right]\\
&\quad-\cos\gamma^R_C\sin\beta^R_C\left[\cos\alpha^R_C\left(-(y_{B_j}-y)\sin\alpha-(x_{B_j}-x)\cos\alpha\right)\right.\\
&\quad\left.-\sin\alpha^R_C\left((y_{B_j}-y)\cos\alpha-(x_{B_j}-x)\sin\alpha\right)\right]
\end{aligned}\tag{B.5}$$

Finalmente, la derivada de B_j^C respecto a respecto a B_j^W es otra matriz de dimensión 3×3:

$$
\begin{aligned}
\partial B_j^C / \partial B_j^W{}_{1,1} &= \cos\gamma_C^R \left(\cos\alpha_C^R \cos\alpha - \sin\alpha_C^R \sin\alpha\right) \\
&\quad + \sin\gamma_C^R \sin\beta_C^R \left(-\cos\alpha_C^R \sin\alpha - \sin\alpha_C^R \cos\alpha\right) \\
\partial B_j^C / \partial B_j^W{}_{1,2} &= \sin\gamma_C^R \sin\beta_C^R \left(\cos\alpha_C^R \cos\alpha - \sin\alpha_C^R \sin\alpha\right) \\
&\quad + \cos\gamma_C^R \left(\cos\alpha_C^R \sin\alpha + \sin\alpha_C^R \cos\alpha\right) \\
\partial B_j^C / \partial B_j^W{}_{1,3} &= -\sin\gamma_C^R \cos\beta_C^R \\
\partial B_j^C / \partial B_j^W{}_{2,1} &= \cos\beta_C^R \left(-\cos\alpha_C^R \sin\alpha - \sin\alpha_C^R \cos\alpha\right) \\
\partial B_j^C / \partial B_j^W{}_{2,2} &= \cos\beta_C^R \left(\cos\alpha_C^R \cos\alpha - \sin\alpha_C^R \sin\alpha\right) \\
\partial B_j^C / \partial B_j^W{}_{2,3} &= \sin\beta_C^R \\
\partial B_j^C / \partial B_j^W{}_{3,1} &= \sin\gamma_C^R \left(\cos\alpha_C^R \cos\alpha - \sin\alpha_C^R \sin\alpha\right) \\
&\quad - \cos\gamma_C^R \sin\beta_C^R \left(-\cos\alpha_C^R \sin\alpha - \sin\alpha_C^R \cos\alpha\right) \\
\partial B_j^C / \partial B_j^W{}_{3,2} &= \sin\gamma_C^R \left(\cos\alpha_C^R \sin\alpha + \sin\alpha_C^R \cos\alpha\right) \\
&\quad - \cos\gamma_C^R \sin\beta_C^R \left(\cos\alpha_C^R \cos\alpha - \sin\alpha_C^R \sin\alpha\right) \\
\partial B_j^C / \partial B_j^W{}_{3,3} &= \cos\gamma_C^R \cos\beta_C^R
\end{aligned}
\tag{B.6}
$$

Apéndice C

Calibración experimental de la cámara

El proceso de calibración permite determinar los valores de los parámetros de una cámara de manera que minimicen el error de las ecuaciones que la modelan. Para ello se utiliza un conjunto de puntos en el espacio tridimensional con coordenadas $3D$ conocidas y sus correspondientes proyecciones sobre la imagen. Este conjunto de puntos se selecciona a partir de un *patrón* o *rejilla de calibración*.

La precisión en la estimación de los parámetros de calibración depende, en gran medida, de las medidas del patrón de calibración y de la precisión con la que se extraigan las características proyectadas en la imagen. En nuestro caso se han seguido las pautas indicadas en [Bakstein y Pajdla, 2002]:

1. **Patrón de calibración**
   Se utiliza como patrón de calibración un cilindro en cuya pared interior se adhiere el patrón de calibración de la figura C.1, formándose un conjunto de círculos paralelos y líneas verticales (cuerdas) equidistantes.

   Cada uno de los círculos se corresponde con un ángulo θ, que representa el ángulo que forman los rayos de luz con el eje óptico de la cámara, como se muestra en la derecha de la figura C.2(b). La distancia de muestreo de los ángulos no es uniforme: se utiliza un incremento de 5° para ángulos entre 90° y 70°, que se corresponderán con los círculos de la parte superior del cilindro, y de 10° para los que formarán la parte inferior del cilindro.

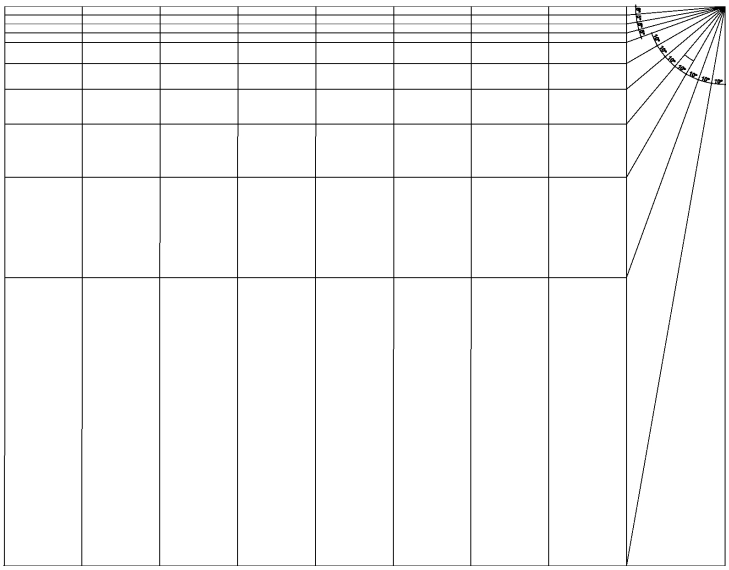


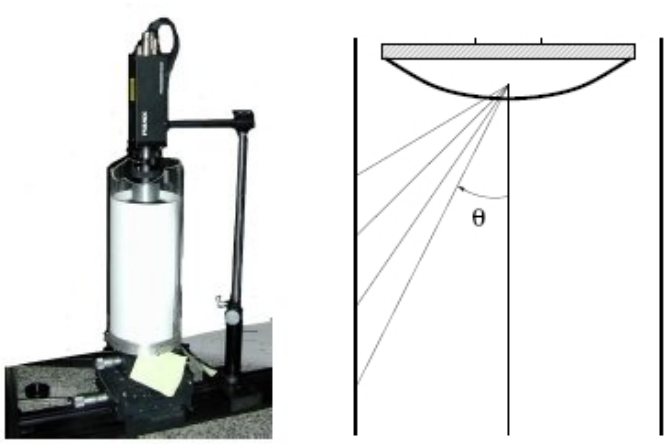
Figura C.1: Patrón de calibración empleado para la cámara omnidireccional utilizada (fig. 3.1). En la parte derecha se muestra como se estiman las distancias entre las líneas horizontales para los distintos valores de θ

Las cuerdas representan los distintos valores del ángulo φ en el cilindro. El muestreo se realiza cada 45°, lo que hace un total de ocho líneas equidistantes entre sí: la primera y la última línea del patrón de calibración (fig. C.1) deben coincidir exactamente al enrollarse el plano para formar el cilindro de calibración. Las intersecciones entre las líneas verticales (cuerdas) y las líneas horizontales (círculos) del patrón de calibración determinan los puntos que se utilizarán para el resto del proceso.

2. **Obtención de las imágenes**

   El siguiente paso consiste en alinear la cámara con el patrón de calibración (fig. C.2(a)) para tomar la imagen de calibración. En primer lugar, el eje óptico de la cámara (z) debe ser paralelo al eje del cilindro (fig. C.2(b)) y la cámara debe estar a la altura del círculo $\theta = 90^\circ$ (fig. C.2(b)). De esta manera, la primera recta horizontal del patrón de calibración (fig. C.1) se verá como un círculo en el borde exterior de la imagen (fig. C.3). El resto de rectas horizontales del patrón aparecen como círculos concéntricos.

   Por último, se gira la cámara respecto al eje óptico para que los dos ejes del plano de la imagen apunten a sendas cuerdas del cilindro de calibración (las líneas verticales



(a) Montaje experimental. (b) Corte transversal del cilindro.

Figura C.2: Montaje y alineamiento de la cámara respecto al patrón de calibración (en la cara interior del cilindro).

del patrón de calibración en la figura C.1). En la imagen las cuerdas del cilindro de calibración aparecen como radios dispuestos cada 45° (fig. C.3).

3. **Extracción de características**
   El tercer paso en el proceso de calibración consiste en extraer una serie de características fáciles de detectar con precisión en la imagen de calibración. En nuestro caso se utilizan los puntos de corte entre los círculos y los radios. El resultado es un conjunto coordenadas (u_i, v_i).

4. **Obtención de los parámetros del modelo**
   Gracias al montaje experimental, se puede asociar fácilmente cada punto de la imagen con un punto en la cara interior del cilindro de calibración. Como se conoce el modelo de la cámara (ec. 3.3), se puede estimar de forma sencilla la proyección de cada punto en la imagen $(\tilde{u}_i, \tilde{v}_i)$ y determinar el error cometido en cada caso.

   El último paso será minimizar el error global (J) para todos los puntos del patrón de calibración variando los parámetros del modelo de la cámara: $a, b, c, d, \beta, u_0, v_0$. El ajuste se ha realizado mediante el algoritmo de minimización de Levenberg-Marquardt[Levenberg, 1944].

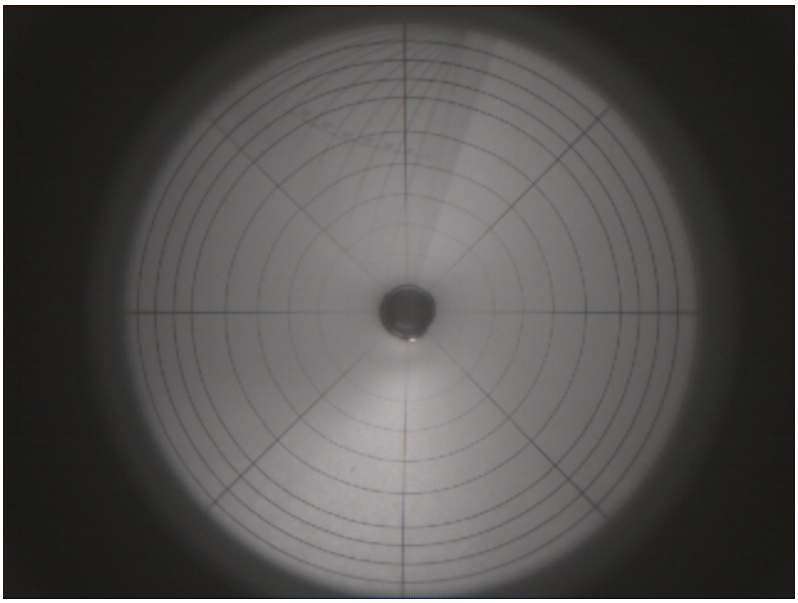


Figura C.3: Imagen de calibración.

$$J(a,b,c,d,\beta,u_0,v_0) = \Sigma_{i=1}^{N} \| r(u_i,v_i) - r(\tilde{u}_i,\tilde{v}_i) \| \tag{C.1}$$

$$r(u',v') = \sqrt{(u'-u_0)^2 + (v'-v_0)^2} \tag{C.2}$$

donde $\| \dots \|$ denota la norma euclídea, N es el número de puntos y la función r calcula la distancia del píxel central (u_0, v_0) con respecto a las coordenadas de los puntos (u', v'). Estas puntos, o bien son medidas directamente en la imagen (u_i, v_i), o bien calculadas mediante la proyección del punto 3D por medio del modelo de la cámara $(\tilde{u}_i, \tilde{v}_i)$.

Los valores finales de los parámetros obtenidos después de la calibración se recogen en la tabla C.1. El error medio cometido en la calibración, comparando los valores teóricos de las coordenadas en la imagen con los valores estimados por su proyección con el modelo de la cámara, es de tan solo 1 píxel.

Tabla C.1: Paramétros de la cámara omnidireccional utilizada.

a	b	c	d	β	u_0	v_0
406,1510,	2,9951	2,0066	0,21	1	248	324,5

BIBLIOGRAFÍA

ANDREASSON, H.; DUCKETT, T. y LILIENTHAL, A. J. (2007). «Mini-SLAM: Minimalistic Visual SLAM in Large-Scale Environments Based on a New Interpretation of Image Similarity». En: *IEEE International Conference on Robotics and Automation (ICRA)*, págs. 4096–4101.

ANDREASSON, H.; DUCKETT, T. y LILIENTHAL, A. J. (2008). «A Minimalistic Approach to Appearance-Based Visual SLAM». *IEEE Transactions on Robotics*, **24(5)**, págs. 991–1001.

ANDREASSON, H.; TREPTOW, A. y DUCKETT, T. (2005). «Localization for Mobile Robots using Panoramic Vision, Local Features and Particle Filter». En: *IEEE International Conference on Robotics and Automation (ICRA)*, págs. 3348–3353.

ATIYA, S. y HAGER, G. D. (1993). «Real-Time Vision-Based Robot Localization». *IEEE Transactions on Robotics and Automation*, **9(6)**, págs. 785 – 800.

AUSTIN, D. J. y JENSFELT, P. (2000). «Using multiple Gaussian hypotheses to represent probability distributions for mobile robot localization». En: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, págs. 1036–1041.

BAILEY, T. (2003). «Constrained initialisation for bearing-only SLAM». En: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, págs. 1966 – 1971.

BAILEY, T. y DURRANT-WHYTE, H. F. (2006). «Simultaneous localization and mapping (SLAM): Part II». *IEEE Robotics & Automation Magazine*, **13(6)**, págs. 108–117.

BAKSTEIN, H. y PAJDLA, T. (2002). «Panoramic mosaicing with a 180º field of view lens». En: *Proceedings of the Workshop on Omnidirectional Vision (OMNIVIS)*, págs. 60–67.

BANGASH, S. A. y GHAFOOR, A. (2011). «Vision based mobile node localization using a landmark». En: *Proceedings of the International Conference on Automation, Robotics and Applications (ICARA)*, págs. 255–259.

BORENSTEIN, J.; EVERETT, H. R. y FENG, L. (1996). *Where am I? Sensors and Methods for Mobile Robot Positioning*. A. K. Peters, Ltd..

BOSSE, M.; NEWMAN, P. M.; LEONARD, J. J. y TELLER, S. (2004). «Simultaneous Localization and Map Building in Large-Scale Cyclic Environments Using the Atlas Framework». *The International Journal of Robotics Research (IJRR)*, **23(12)**, págs. 1113–1139.

BRASSART, E.; DELAHOCHE, L.; CAUCHOIS, C.; DROCOURT, C.; PEGARD, C. y MOUADDIB, E. M. (2000). «Experimental Results got with the Omnidirectional Vision Sensor: Syclop». En: *Proceedings of the Workshop on Omnidirectional Vision (OMNIVIS)*, págs. 145–152.

BURGARD, W.; CREMERS, A. B.; FOX, D.; LAKEMEYER, G.; SCHULZ, D.; STEINER, W. y THRUN, S. (1999). «Experiences with an interactive museum tour-guide robot». *Artificial Intelligence*, **114(1)**, págs. 3–55.

CAO, Z.; LIU, S. y RONING, J. (2007). «Omni-directional Vision Localization Based on Particle Filter». En: *Proceedings of the International Conference on Image and Graphics (ICIG)*, págs. 478–483.

CAO, Z. L; OH, S. J y HALL, E. L. (1986). «Dynamic omnidirectional vision for mobile robots», **3(1)**, págs. 5–17.

CASSANDRA, A. R. y KAELBLING, L. P. (1996). «Acting under uncertainty: Discrete Bayesian models for mobile-robot navigation». En: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, págs. 963–972.

CASTELLANOS, J. A. y TARDÓS, J. D. (1999). «Mobile Robot Localization and Map Building: A Multisensor Fusion Approach». En: *Proceedings of the International Symposium on Experimental Robotics*, págs. 173–178.

CASTLE, R. O.; KLEIN, G. y MURRAY, D. W. (2011). «Wide-area augmented reality using camera tracking and mapping in multiple regions». *Computer Vision and Image Understanding*, **115(6)**, págs. 854–867.

CHENAVIER, F. y CROWLEY, J. L. (1992). «Position estimation for a mobile robot using vision and odometry». En: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, págs. 2588–2593.

CHOI, H.; JO, S. y KIM, E. (2012). «CV-SLAM using line and point features». En: *Proceedings of the International Conference on Control, Automation and Systems (ICCAS)*, págs. 1465–1468.

CHOI, H.; KIM, D. Y.; HWANG, J. P.; KIM, E. y KIM, Y. O. (2010). «CV-SLAM using ceiling boundary». En: *Proceedings of the IEEE Conference on Industrial Electronics and Applications (ICIEA)*, págs. 228–233.

CHOI, Y. H. y OH, S. Y. (2007). «Map building through pseudo dense scan matching using visual sonar data». *Autonomous Robots*, **23(4)**, págs. 293–304.

CIVERA, J.; DAVISON, A. J. y MONTIEL, J. M. M. (2007). «Inverse Depth to Depth Conversion for Monocular SLAM». En: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, págs. 2778–2783.

CIVERA, J.; GRASA, O. G.; DAVISON, A. J. y MONTIEL, J. M. M. (2010). «1-Point RANSAC for extended Kalman filtering: Application to real-time structure from motion and visual odometry». *Journal of Field Robotics*, **27(5)**, págs. 609–631.

COX, I. J. (1990). *Autonomous robot vehicles*. Springer-Verlag Berlin.

COX, I. J. (1991). «Blanche-an experiment in guidance and navigation of an autonomous robot vehicle». *IEEE Transactions on Robotics*, **7(2)**, págs. 193–204.

COX, I. J. y MILLER, M. L. (1995). «On finding ranked assignments with application to multitarget tracking and motion correspondence». *IEEE Transactions on Aerospace and Electronic Systems*, **31(1)**, págs. 1–4.

CUMMINS, M. y NEWMAN, P. (2008). «FAB-MAP: Probabilistic localization and mapping in the space of appearance». *The International Journal of Robotics Research*, **27(6)**, págs. 647–665.

DAVISON, A. J.; CID, Y. G. y KITA, N. (2004). «Real-time 3D SLAM with wide-angle vision». En: *Proceedings of the IFAC/EURON Symposium on Intelligent Autonomous Vehicle (IAV)*, .

DAVISON, A. J. y MURRAY, D. W. (1998). «Mobile Robot Localisation using Active Visual Sensing». En: *Proceedings of the Fifth European Conference on Computer Vision*, págs. 809–825.

DE LA ESCALERA, A. y ARMINGOL, J. M. (2001). *Visión por computador. Fundamentos y métodos.* Pearson Educacion.

DEANS, M. C. (2000). «Invariant filtering for simultaneous localization and mapping». En: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, págs. 1042–1047.

DELLAERT, F.; BURGARD, W.; FOX, D. y THRUN, S. (1999a). «Using the CONDENSATION Algorithm for Robust, Vision-based Mobile Robot Localization». En: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, págs. 588–590.

DELLAERT, F.; FOX, D. y BURGARD, W. (1999b). «Monte Carlo localization for mobile robots». En: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, págs. 1322–1328.

DELLAERT, F.; SEITZ, S. M.; THORPE, C. E. y THRUN, S. (2003). «EM, MCMC, and chain flipping for structure from motion with unknown correspondence». *Machine learning*, **50(1-2)**, págs. 45–71.

DEMSAR, J. (2006). «Statistical comparisons of classifiers over multiple data sets». *The Journal of Machine Learning Research*, **7**, págs. 1–30.

DISSANAYAKE, M. W. M.; NEWMAN, P.; CLARK, S. y DURRANT-WHITE, H. F. (2001). «A solution to the simultaneous localization and map building (SLAM) problem». *IEEE Transactions on Robotics and Automation*, **17(3)**, págs. 229–241.

DOUCET, A.; DE FREITAS, N. y GORDON, N. (2001). «An Introduction to Sequential Monte Carlo Methods». En: *Sequential Monte Carlo Methods in Practice*, capítulo 1, págs. 3–14. Springer New York.

DRUMHELLER, M. (1987). «Mobile robot localization using sonar.» *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, **9(2)**, págs. 325–32.

DURRANT-WHYTE, H. F. (2006). «Simultaneous localization and mapping: part I». *IEEE Robotics & Automation Magazine*, **13(2)**, págs. 99–110.

DUSHA, D. y MEJIAS, L. (2012). «Error analysis and attitude observability of a monocular GPS/visual odometry integrated navigation filter». *The International Journal of Robotics Research*, **31(6)**, págs. 714–737.

EDMUND OPTICS (2013). «Colored Glass Bandpass Filters». `http://www.edmundoptics.com/optics/optical-filters/bandpass-filters/colored-glass-bandpass-filters/1924?showall#products`

ESTRADA, C.; NEIRA, J. y TARDÓS, J. D. (2005). «Hierarchical SLAM: real-time accurate mapping of large environments». *IEEE Transactions on Robotics*, **21(4)**, págs. 588–596.

EUSTICE, R. M.; SINGH, H. y LEONARD, J. J. (2006). «Exactly Sparse Delayed-State Filters for View-Based SLAM». *IEEE Transactions on Robotics*, **22(6)**, págs. 1100–1114.

FOLKESSON, J.; JENSFELT, P. y CHRISTENSEN, H. I. (2007). «The M-Space Feature Representation for SLAM». *IEEE Transactions on Robotics*, **23(5)**, págs. 1024–1035.

FOX, D. (2003). «Adapting the Sample Size in Particle Filters Through KLD-Sampling». *The International Journal of Robotics Research*, **22(12)**, págs. 985–1003.

FOX, D.; BURGARD, W.; DELLAERT, F. y THRUN, S. (1999a). «Monte Carlo Localization: Efficient Position Estimation for Mobile Robots». En: *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, págs. 343–349.

FOX, D.; BURGARD, W. y KRUPPA, H. (2000). «A probabilistic approach to collaborative multi-robot localization». *Autonomous Robots*, **8(3)**, págs. 325–344.

FOX, D.; BURGARD, W. y THRUN, S. (1999b). «Markov Localization for Mobile Robots in Dynamic Environments». *Journal of Artificial Intelligence Research*, **11**, págs. 391–427.

FOX, D.; THRUN, S.; BURGARD, W. y DELLAERT, F. (2001). «Particle Filters for Mobile Robot Localization.» En: *Sequential Monte Carlo Methods in Practice*, capítulo 19, págs. 401–428. Springer New York.

FRESE, U.; LARSSON, P. y DUCKETT, T. (2005). «A multilevel relaxation algorithm for simultaneous localization and mapping». *IEEE Transactions on Robotics*, **21(2)**, págs. 196–207.

FUENTES-PACHECO, J.; RUIZ-ASCENCIO, J. y RENDÓN-MANCHA, J. (2012). «Visual simultaneous localization and mapping: a survey». *Artificial Intelligence Review*, págs. 1–27.

FUJIFILM EUROPE (2013). «FE185C046HA-1».
`http://www.fujifilm.eu/eu/products/optical-devices/cctv-and-machine-vision/p/fe185c046ha-1/`

GARCÍA, S. y HERRERA, F. (2008). «An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons». *Journal of Machine Learning Research*, **9**, págs. 2677–2694.

GLOVER, A. J.; MADDERN, W. P.; MILFORD, M. J. y WYETH, G. F. (2010). «FAB-MAP + RatSLAM: Appearance-based SLAM for Multiple Times of Day». En: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, págs. 3507–3512.

GONZALEZ, J. y OLLERO, A. (1996). «Estimación de la Posición de un Robot Móvil». En: *Proceedings of Automática*, págs. 3–18.

GROSS, H. M.; KOENIG, A.; BOEHME, H. J. y SCHROETER, CH. (2002). «Vision-based Monte Carlo self-localization for a mobile service robot acting as shopping assistant in a home store». En: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, págs. 256–262.

GUIVANT, J. E.; NEBOT, E. M. y MEMBER, S. (2001). «Optimization of the Simultaneous Localization and Map-Building Algorithm for Real-Time Implementation». *IEEE Robotics and Automation*, **17(3)**, págs. 242–257.

GUTMANN, J. S. y FOX, D. (2002). «An experimental comparison of localization methods continued». En: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, págs. 454–459.

HERTZBERG, J. (1996). «Landmark-based autonomous navigation in sewerage pipes». En: *Proceedings of the First Euromicro Workshop on Advanced Mobile Robot*, págs. 68–73.

HOLM, S. (1979). «A simple sequentially rejective multiple test procedure». *Scandinavian Journal of Statistics*, **6(2)**, págs. 65–70.

HOWARD, A. y SUKHATME, G. S. (2003). «Cooperative relative localization for mobile robot teams: an ego-centric approach». En: *Proceedings of the International Workshop on Multi-Robot Systems*, págs. 65–76.

HOYA OPTICS (2013). «Infrared Transmitting Filters».
`http://www.hoyaoptics.com/color_filter/ir_transmitting.htm`

JENSFELT, P. y KRISTENSEN, S. (2001). «Active Global Localization for a Mobile Robot Using Multiple Hypothesis Tracking». *IEEE Transactions on Robotics and Automation*, **17(5)**, págs. 748–760.

JEONG, W. Y. y LEE, K. M. (2005). «CV-SLAM: A new ceiling vision-based SLAM technique». En: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, págs. 3195–3200.

JOGAN, M. y LEONARDIS, A. (2003). «Robust localization using an omnidirectional appearance-based subspace model of environment». *Robotics and Autonomous Systems*, **45(1)**, págs. 51–72.

JONES, E. S. y SOATTO, S. (2011). «Visual-inertial navigation, mapping and localization: A scalable real-time causal approach». *The International Journal of Robotics Research*, **30(4)**, págs. 407–430.

KAESS, M. y DELLAERT, F. (2006). «Visual SLAM with a Multi-Camera Rig». *Informe técnico*, Georgia Institute of Technology.

KANG, J. G.; KIM, S.; AN, S. Y. y OH, S. Y. (2012). «A new approach to simultaneous localization and map building with implicit model learning using neuro evolutionary optimization». *Applied Intelligence*, **36(1)**, págs. 242–269.

KAWEWONG, A.; TONGPRASIT, N. y HASEGAWA, O. (2011). «PIRF-Nav 2.0: Fast and online incremental appearance-based loop-closure detection in an indoor environment». *Robotics and Autonomous Systems*, **59(10)**, págs. 727–739.

KIM, J. y KWEON, I. (2007). «Robust feature matching for loop closing and localization». En: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, págs. 3905–3910.

KIM, S. y OH, S. Y. (2008). «SLAM in Indoor Environments using Omni-directional Vertical and Horizontal Line Features». *Journal of Intelligent and Robotic Systems*, **51(1)**, págs. 31–43.

KLEIN, G. y MURRAY, D. (2007). «Parallel tracking and mapping for small AR workspaces». En: *Proceedings of the 6th IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, págs. 225–234.

KONOLIGE, K. y AGRAWAL, M. (2008). «FrameSLAM: From Bundle Adjustment to Real-Time Visual Mapping». *IEEE Transactions on Robotics*, **24(5)**, págs. 1066–1077.

KONOLIGE, K.; AGRAWAL, M. y SOLÀ, J. (2011). «Large-Scale Visual Odometry for Rough Terrain». En: *Robotics Research*, págs. 201–212. Springer Berlin Heidelberg.

KONOLIGE, K.; BOWMAN, J.; CHEN, J. D.; MIHELICH, P.; CALONDER, M.; LEPETIT, V. y FUA, P. (2010). «View-based maps». *The International Journal of Robotics Research*, **29(8)**, págs. 941–957.

KRÖSE, B. J. A.; VLASSIS, N.; BUNSCHOTEN, R. y MOTOMURA, Y. (2001). «A probabilistic model for appearance-based robot localization». *Image and Vision Computing*, **19(6)**, págs. 381–391.

KROTKOV, E. (1989). «Mobile robot localization using a single image». En: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, págs. 978–983.

KUHN, H. W. (1955). «The Hungarian method for the assignment problem». *Naval Research Logistics Quarterly*, **2(1-2)**, págs. 83–97.

KWOK, C. y FOX, D. (2004). «Real-time particle filters». *Proceedings of the IEEE*, **92(3)**, págs. 469– 484.

LEMAIRE, T. y LACROIX, S. (2007). «SLAM with Panoramic Vision». *Journal of Field Robotics*, **24(1-2)**, págs. 91–111.

LENSER, S. y VELOSO, M. (2000). «Sensor resetting localization for poorly modelled mobile robots». En: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, págs. 1225–1232.

LEONARD, J. J. y DURRANT-WHITE, H. F. (1991). «Simultaneous map building and localization for an autonomous mobile robot». En: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, págs. 1442–1447.

LEONARD, J. J. y DURRANT-WHYTE, H. F. (1991). «Mobile Robot Localization by Tracking Geometric Beacons». *IEEE Transaction on Robotics and Automation*, **7(3)**, págs. 376–382.

LEONARD, J. J. y FEDER, H. J. S. (1999). «A computationally efficient method for large-scale concurrent mapping and localization». En: *Proceedings of the International Symposium Robotics Research*, págs. 169–176.

LEONARD, J. J. y NEWMAN, P. M. (2003). «Consistent, convergent, and constant-time SLAM». En: *Proceedings of the 18th international joint conference on Artificial intelligence (IJCAI)*, págs. 1143–1150.

LEVENBERG, K. (1944). «A method for the solution of certain problems in least squares». *Quarterly of Applied Mathematics*, **2**, págs. 164–168.

LI, W. (2003). «Constrained unscented Kalman filter based fusion of GPS/INS/digital map for vehicle localization». En: *Proccedings of the IEEE Intelligent Transportation Systems*, págs. 1362–1367.

LIU, J. S. (1996). «Metropolized independent sampling with comparisons to rejection sampling and importance sampling». *Statistics and Computing*, **6(2)**, págs. 113–119.

LU, F.; MILIOS, E. y YORK, N. (1997). «Globally Consistent Range Scan Alignment for Environment Mapping». *Autonomous Robots*, **4(4)**, págs. 333–349.

LUI, W. L. D. y JARVIS, R. (2012). «A pure vision-based topological SLAM system». *The International Journal of Robotics Research*, **31(4)**, págs. 403–428.

MANOLAKIS, D. E. (1996). «Efficient solution and performance analysis of 3-D position estimation by trilateration». *IEEE Transactions on Aerospace and Electronic Systems*, **32(4)**, págs. 1239 – 1248.

MARKS, T. K.; HOWARD, A.; BAJRACHARYA, M.; COTTRELL, G. W. y MATTHIES, L. (2007). «Gamma-SLAM: Stereo Visual SLAM in Unstructured Environments Using Variance Grid Maps». En: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, págs. 3717 – 3724.

MARTINELLI, F. (2008). «Robot localization: comparable performance of EKF and UKF in some interesting indoor settings». *Proceedings of the 16th Mediterranean Conference on Control and Automation*, págs. 499–504.

MARZORATI, D.; MATTEUCCI, M. y SORRENTI, D. G. (2007). «Particle-based Sensor Modeling for 3D-Vision SLAM». En: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, págs. 4801–4806.

MCDONALD, J.; KAESS, M.; CADENA, C.; NEIRA, J. y LEONARD, J. J. (2012). «Real-time 6-DOF multi-session visual SLAM over large scale environments». *Robotics and Autonomous Systems*.

MEI, C.; SIBLEY, G.; CUMMINS, M.; NEWMAN, P. y REID, I. (2011). «RSLAM: A system for large-scale mapping in constant-time using stereo». *International Journal of Computer Vision*, **94(2)**, págs. 198–214.

MENEGATTI, E.; PRETTO, A.; SCARPA, A. y PAGELLO, E. (2006). «Omnidirectional Vision Scan Matching for Robot Localization in Dynamic Environments.» *IEEE transactions on robotics*, **22(3)**, págs. 523–535.

MENEGATTI, E.; ZOCCARATO, M.; PAGELLO, E. y ISHIGURO, H. (2004). «Image-based Monte-Carlo localisation with omnidirectional images». *Robotics and Autonomous Systems*, **48(1)**, pág. 31.

MIGLIORE, D.; RIGAMONTI, R.; MARZORATI, D.; MATTEUCCI, M. y SORRENTI, D. G. (2009). «Use a single camera for simultaneous localization and mapping with mobile object tracking in dynamic environments». En: *Proceedings of the Workshop on Safe Navigation in Open and Dynamic Environments: Application to Autonomous Vehicles held at the International Conference on Intelligent Robots and Systems (ICRA)*, .

MILFORD, M. J. y WYETH, G. F. (2008). «Mapping a Suburb With a Single Camera Using a Biologically Inspired SLAM System». *IEEE Transactions on Robotics*, **24(5)**, págs. 1038–1053.

MONTEMERLO, M. y THRUN, S. (2007). *FastSLAM: A Scalable Method for the Simultaneous Localization and Mapping Problem in Robotics*. Springer Verlag Berlin.

MONTEMERLO, M.; THRUN, S.; KOLLER, D. y WEGBREIT, B. (2002). «FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem». En: *Proceedings of the AAAI National Conference on Artificial Intelligence*, págs. 593–598.

MOREIRA, M. G.; MACHADO, H. N.; MENDONCA, C. F. C. y PEREIRA, G. S. (2007). «Mobile robot outdoor localization using planar beacons and visual improved odometry». En: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, págs. 2468–2473.

MURILLO, A. C.; GUTIÉRREZ-GÓMEZ, D.; RITUERTO, A.; PUIG, L. y GUERRERO, J. J. (2012). «Wearable omnidirectional vision system for personal localization and guidance». En: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, págs. 8–14.

MURILLO, A. C.; SAGÜÉS, C.; GUERRERO, J. J.; GOEDEMÉ, T.; TUYTELAARS, T. y GOOL, L. V. (2006). «Hierarchical localization by matching vertical lines in omnidirectional images». En: *Proceedings of the Workshop From Sensors to Human Spatial Concepts held at the International Conference on Intelligent Robots and Systems (IROS)*, págs. 13–19.

MURTY, K. G. (1986). «An Algorithm for Ranking all the Assignments in Order of Increasing Cost.» *Operations research*, **16(3)**, págs. 682–687.

NEWMAN, P.; SIBLEY, G. y SMITH, M. (2009). «Navigating, recognizing and describing urban spaces with vision and lasers». *The International Journal of Robotics Research*, **28(11-12)**, págs. 1406–1433.

NÜTZI, G.; WEISS, S.; SCARAMUZZA, D. y SIEGWART, R. (2011). «Fusion of IMU and vision for absolute scale estimation in monocular SLAM». *Journal of Intelligent and Robotic Systems*, **61(1)**, págs. 287–299.

PALETTAT, L. y SIMONE, F. (2001). «Robust Localization Using Context in Omnidirectional Imaging». En: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, págs. 2072–2077.

PAZ, L. M.; PINIÉS, P.; TARDÓS, J. D. y NEIRA, J. (2008a). «Large-Scale 6-DOF SLAM With Stereo-in-Hand». *IEEE Transactions on Robotics*, **24(5)**, págs. 946–957.

PAZ, L. M.; TARDÓS, J. D. y NEIRA, J. (2008b). «Divide and Conquer: EKF SLAM in O(n)». En: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, págs. 1107–1120.

PHILIP, D. y HO, S. (2011). «Orientation descriptors for localization in urban environments». En: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, págs. 494–501.

PINIÉS, P. (2009). *SLAM in Large Environments with Wearable Sensors*. Tesis doctoral, Universidad de Zaragoza.

PIRKER, K.; RUTHER, M. y BISCHOF, H. (2011). «CD SLAM-continuous localization and mapping in a dynamic world». En: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, págs. 3990–3997.

PUPILLI, M. y CALWAY, A. (2006). «Real-Time Visual SLAM with Resilience to Erratic Motion». En: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, págs. 1244–1249.

RAMALINGAM, S.; BOUAZIZ, S.; STURM, P. y BRAND, M. (2009). «Geolocalization using skylines from omni-images». En: *Procceedings of the IEEE International Conference on Computer Vision Workshop (ICCV)*, págs. 23–30.

RODA, J. P.; SÁEZ, J. M. y ESCOLANO, F. (2007). «Ceiling mosaics through information-based SLAM». En: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, págs. 3898–3904.

ROFER, T. y JUNGEL, M. (2003). «Vision-based fast and reactive monte-carlo localization». En: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, págs. 856–861.

RYBSK, P. E.; ROUMELIOTIS, S.; GINI, M. y PAPANIKOPOULOS, N. (2008). «Appearance-based mapping using minimalistic sensor models». *Autonomous Robots*, **24(3)**, págs. 229–246.

SAEDAN, M.; LIM, C. W. y ANG, M. H. (2007). «Appearance-based SLAM with map loop closing using an omnidirectional camera». En: *Proccedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, págs. 1–6.

SÁEZ, J. M. y ESCOLANO, F. (2005). «Entropy minimization SLAM using stereo vision». En: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, págs. 36–43.

SÁEZ, J. M.; ESCOLANO, F. y PENALVER, A. (2005). «First Steps towards Stereo-based 6DOF SLAM for the Visually Impaired». En: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pág. 23.

SÁEZ, J. M.; HOGUE, A.; ESCOLANO, F. y JENKIN, M. (2006). «Underwater 3D SLAM through entropy minimization». *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, págs. 3562–3567.

SALICHS, M. A.; ARMINGOL, J. M.; MORENO, L. E. y DE LA ESCALERA, A. (1999). «Localization System for Mobile Robots in Indoor Environments». *Integrated Computer-Aided Engineering*, **6(4)**, págs. 303—-318.

SCARAMUZZA, D.; FRAUNDORFER, F. y POLLEFEYS, M. (2010). «Closing the loop in appearance-guided omnidirectional visual odometry by using vocabulary trees». *Robotics and Autonomous Systems*, **58(6)**, págs. 820–827.

SCHLEGEL, C. y HOCHDORFER, S. (2008). «Localization and mapping for service robots: Bearing-only slam with an omnicam». En: *Advances in Service Robotics*, capítulo 15, págs. 253–278. InTech.

SE, S.; LOWE, D. G. y LITTLE, J. J. (2002). «Mobile Robot Localization and Mapping with Uncertainty using Scale-Invariant Visual Landmarks». *The International Journal of Robotics Research*, **21(8)**, págs. 735–758.

SHIMIZUHIRA, W. y MAEDA, Y. (2003). «Self-Localization Method UsedMultiple Omnidirectional Vision System.» En: *Proccedings of the SICE Annual Conference*, págs. 324 – 327.

SICILIANO, B. y KHATIB, O. (2008). *Handbook of Robotics*. Springer-Verlag Berlin.

SILVEIRA, G.; MALIS, E. y RIVES, P. (2008). «An Efficient Direct Approach to Visual SLAM». *IEEE Transactions on Robotics,*, **24(5)**, págs. 969–979.

SIM, R.; ELINAS, P. y LITTLE, J. J. (2007). «A Study of the Rao-Blackwellised Particle Filter for Efficient and Accurate Vision-Based SLAM». *International Journal of Computer Vision*, **74(3)**, págs. 303–318.

SIMMONS, R. (1995). «Probabilistic robot navigation in partially observable environments». En: *International Joint Conference on Artificial Intelligence*, págs. 1080–1087.

SMITH, R. C. y CHEESEMAN, P. (1986). «On the Representation and Estimation of Spatial Uncertainty». *The International Journal of Robotics Research*, **5(4)**, págs. 56–68.

SOLÀ, J. (2007). *Towards visual localization, mapping and moving objects tracking by a mobile robot: a geometric and probabilistic approach*. Tesis doctoral, Institut National Polytechnique de Toulouse.

SOLÀ, J.; LEMAIRE, T.; DEVY, M.; LACROIX, S. y MONIN, A. (2005a). «Delayed vs undelayed landmark initialization for bearing-only SLAM». En: *Proceedings of the Workshop on SLAM held at the IEEE International Conference on Robotics and Automation (ICRA)*, págs. 2499 – 2504.

SOLÀ, J.; MONIN, A.; DEVY, M. y LEMAIRE, T. (2005b). «Undelayed initialization in bearing only SLAM». En: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, págs. 2499–2504.

SOLÀ, J.; MONIN, A.; DEVY, M. y VIDAL-CALLEJA, T. (2008). «Fusing Monocular Information in Multicamera SLAM». *IEEE Transactions on Robotics*, **24(5)**, págs. 958–968.

STRASDAT, H.; MONTIEL, J. M. M. y DAVISON, A. J. (2012). «Visual SLAM: Why filter?» *Image and Vision Computing*, **30(2)**, págs. 65–77.

SUGIHARA, K. (1988). «Some location problems for robot navigation using a single camera». *Computer Vision, Graphics, and Image Processing*, **42(1)**, págs. 112 – 129.

SUN, Y.; CAO, Q. y CHEN, W. (2004). «An Object Tracking and Global Localization Method using Omnidirectional Vision System.» En: *Proccedings of the 5th World Congress on Intelligent Control and Automation*, págs. 4730 – 4735.

SUZUKI, S. y BE, K. (1985). «Topological structural analysis of digitized binary images by border following». *Computer Vision, Graphics, and Image Processing*, **30(1)**, págs. 32–46.

TARDÓS, J. D.; NEIRA, J.; NEWMAN, P. M. y LEONARD, J. J. (2002). «Robust Mapping and Localization in Indoor Environments Using Sonar Data». *The International Journal of Robotics Research*, **21(4)**, págs. 311–330.

THOMAS, F. (2005). «Revisiting Trilateration for Robot Localization». *IEEE Transactions on Robotics*, **21(1)**, págs. 93–101.

THRUN, S.; BEETZ, M.; BENNEWITZ, M.; CREMERS, A.; DELLAERT, F.; FOX, D.; HÄHNEL, D.; ROSENBERG, C.; ROY, N.; SCHULTE, J. y SCHULZ, D. (2000). «Probabilistic Algorithms and the Interactive Museum Tour-Guide Robot Minerva». *The International Journal of Robotics Research*, **19(11)**, págs. 972–999.

THRUN, S.; BURGARD, W. y FOX, D. (1998). «A Probabilistic Approach to Concurrent Mapping and Localization for Mobile Robots». *Autonomous Robots*, **5(3-4)**, págs. 253–271.

THRUN, S.; BURGARD, W. y FOX, D. (2003). *Probabilistic Robotics*. MIT Press.

THRUN, S.; FOX, D.; BURGARD, W. y DELLAERT, F. (2001). «Robust Monte Carlo localization for mobile robots». *Artificial Intelligence*, **128(1-2)**, págs. 99–141.

THRUN, S.; KOLLER, D.; GHAHRAMANI, Z.; DURRANT-WHYTE, H. y NG, A. Y (2004). «Simultaneous Mapping and Localization With Sparse Extended Information Filters». *The International Journal of Robotics Research*, **23**, págs. 693–716.

TOMASI, C. y KANADE, T. (1992). «Shape and motion from image streams under orthography: a factorization method». *International Journal of Computer Vision*, **9(2)**, págs. 137–154.

TONGPRASIT, N.; KAWEWONG, A. y HASEGAWA, O. (2011). «PIRF-Nav 2: Speeded-up online and incremental appearance-based SLAM in an indoor environment». En: *Proccedings of the IEEE Workshop on Applications of Computer Vision (WACV)*, págs. 145–152.

VALIENTE GARCÍA, D.; FERNÁNDEZ ROJO, L.; GIL APARICIO, A.; PAYÁ CASTELLÓ, L. y REINOSO GARCÍA, O. (2012). «Visual Odometry through Appearance-and Feature-Based Method with Omnidirectional Images». *Journal of Robotics*.

VU, T. D.; AYCARD, O. y APPENRODT, N. (2007). «Online localization and mapping with moving object tracking in dynamic outdoor environments». En: *Proccedings of the IEEE Intelligent Vehicles Symposium*, págs. 190–195.

WALTER, M. R.; EUSTICE, R. M. y LEONARD, J. J. (2007). «Exactly Sparse Extended Information Filters for Feature-based SLAM». *The International Journal of Robotics Research*, **26(4)**, págs. 335–359.

WANG, C. C.; THORPE, C.; THRUN, S.; HEBERT, M. y DURRANT-WHYTE, H. F. (2007). «Simultaneous localization, mapping and moving object tracking». *The International Journal of Robotics Research*, **26(9)**, págs. 889–916.

WILLIAMS, S. B. (2001). *Efficient solutions to autonomous mapping and navigation problems*. Tesis doctoral, The University of Sydney.

WILLIAMS, S. B.; DISSANAYAKE, G. y DURRANT-WHYTE, H. F. (2002). «An Efficient Approach to the Simultaneous Localisation and Mapping Problem». En: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, págs. 406 – 411.

WOLF, J.; BURGARD, W. y BURKHARDT, H. (2005). «Robust vision-based localization by combining an image-retrieval system with Monte Carlo localization.» *IEEE Transactions on Robotics*, **21(2)**, págs. 208–216.

WONGPHATI, M.; NIPARNAN, N. y SUDSANG, A. (2009). «Bearing only FastSLAM using vertical line information from an omnidirectional camera». En: *Proccedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO)*, págs. 1188–1193.

WU, C. J. y TSAI, W. H. (2009). «Location estimation for indoor autonomous vehicle navigation by omni-directional vision using circular landmarks on ceilings». *Robotics and Autonomous Systems*, **57(5)**, págs. 546–555.

YAGI, Y.; NISHIZAWA, Y. y YACHIDA, M. (1995). «Map-based navigation for a mobile robot with omnidirectional image sensor COPIS». *IEEE Transactions on Robotics and Automation*, **11(5)**, págs. 634–648.

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE ALGORITMOS

LISTADO DE ACRÓNIMOS

BA ***B****undle* ***A****djustment*
BD ***B****ase de* ***D****atos*
BoW *Lista de palabras visuales, del inglés* ***B****ag-****o****f-****W****ords*

DOF *Grados de libertad , del inglés* ***D****egrees* ***O****f* ***F****reedom*

EKF *Filtro de Kalman extendido, del inglés* ***E****xtended* ***K****alman* ***F****ilter*

FAB-MAP ***F****ast* ***A****ppearance* ***B****ased-****MAP****ping*
FOV *Campo de visión, del inglés* ***F****ield* ***O****f* ***V****iew*

GPS *Sistema de posicionamiento global, del inglés* ***G****lobal* ***P****ositioning* ***S****ystem*

H-H *Algoritmo de SLAM propuesto, pero con asociación de datos basada en el método Húngaro en los dos niveles de la jerarquía*
HoC ***H****istogram of* ***O****riented* ***C****ameras*

IDP ***I****nverse-****D****epth* ***P****arametrization*
IEKF *Filtro de Kalman extendido iterativo, del inglés* ***I****terated* ***E****xtended* ***K****alman* ***F****ilter*
IRP *Filtro pasa banda infrarrojo, del inglés* ***I****nfra****R****ed band****P****ass filter*

KLD *Divergencia de Kullback-Leibler, del inglés* ***K****ullback-****L****eibler* ***D****ivergence*
KLT ***K****anade-****L****ucas-****T****omasi*

MCL *Localización de Monte Carlo, del inglés* ***M****onte* ***C****arlo* ***L****ocalization*
MHT *Seguimiento multi-hipótesis, del inglés* ***M****ulti-****H****ypothesis* ***T****racking*

ML *Máxima probabilidad, del inglés* ***M****aximum* ***L****ikelihood*

PDF *Función de densidad de la probabilidad, del inglés* ***P****robability* ***D****ensity* ***F****unction*
PIRF ***P****osition-****I****nvariant* ***R****obust* ***F****eatures*
PTAM ***P****arallel* ***T****racking* ***A****nd* ***M****apping*

RFID *Identificación por radiofrecuencia, del inglés* ***R****adio* ***F****requency* ***ID****entification*

SEIF ***S****parse* ***E****xtended* ***I****nformation* ***F****ilter*
SFM ***S****tructure* ***F****rom* ***M****otion*
SIFT ***S****cale-****I****nvariant* ***F****eature* ***T****ransform*
SLAM *Localización y Mapeado Simultáneos, del inglés* ***S****imultaneous* ***L****ocalization* ***A****nd* ***M****apping*
SURF ***S****peeded* ***U****p* ***R****obust* ***F****eatures*

UKF *Filtro de Kalman* unscented*, del inglés* ***U****nscented* ***K****alman* ***F****ilter*

VO *Odometría visual, del inglés* ***V****isual* ***O****dometry*

NOTACIÓN

Índices

$\mathbf{t}$ *Índice asociado al instante de tiempo*

$\mathbf{k}$ *Índice asociado a la partícula*

$\mathbf{j}$ *Índice asociado a la baliza*

$\mathbf{l}$ *Índice asociado a la medida*

$\mathbf{s}$ *Índice asociado a la hipótesis de asociación*

Partículas

$\mathbf{Y_t}$ *Conjunto de partículas*

$\mathbf{M_t}$ *Número total de partículas*

$\mathbf{x_t^k} = (\mathbf{x}\ \mathbf{y}\ \alpha)^{\mathbf{T}}_{\mathbf{t}}{}^{\mathbf{k}}$ *Pose del robot asociada a la partícula k*

$\omega_{\mathbf{t}}^{\mathbf{k,s}}$ *Factor de importancia o peso de la partícula k*

Balizas

$\mathbf{m}$ *Mapa*

$\mathbf{N_t}$ *Número total de balizas del mapa en el instante t*

$\mathbf{B_{t,j}^{k,s}}$ *Baliza j*

$(\mathbf{x_{B_j}}, \mathbf{y_{B_j}}, \mathbf{z_{B_j}})$ *Posición 3D de la baliza j en el mundo*

$\Upsilon_{\mathbf{j}}$ *Tipo de la baliza j*

$\mu_{\mathbf{t,j}}^{\mathbf{k,s}}, \Sigma_{\mathbf{t,j}}^{\mathbf{k,s}}$ *Media y covarianza de la distribución gaussiana que representa la posición 3D de la baliza j*

$\mathbf{i_{t,j}^{k,s}}$ *Número de veces que una baliza j ha sido detectada*

$\mathbf{C_{t,j}^{k,s}}$ *Baliza candidata j*

$\eta_{\mathbf{t}}^{\mathbf{k,s}}$ *Número total de balizas candidatas en el instante t*

Medidas

$\mathbf{Z_t}$ *Conjunto de medidas en el instante t*

$\mathbf{N_{Z_t}}$ *Número total de medidas en el instante t*

$\mathbf{z_{t,l}}$ *Medida l en el instante t*

$(\mathbf{u_{t,l}}, \mathbf{v_{t,l}})$ *Coordenadas en la imagen de la medida l*

$(\varphi_{\mathbf{t,l}}, \theta_{\mathbf{t,l}})$ *Ángulos de azimut y elevación de la medida l*

$\mathbf{Q_l}$ *Matriz de covarianza del ruido para la medida l*

$\mathbf{h}(\mathbf{B_j}, \mathbf{x_t})$ *Modelo de medida*

$\mathbf{H_x}$ *Matriz Jacobiana del modelo de medida respecto a las variables de estado del robot*

$\mathbf{H_m}$ *Matriz Jacobiana del modelo de medida respecto a las variables de la medida*

Robot

$\mathbf{u_t}$ *Comando de control en el instante t*

$\mathbf{g}(\mathbf{u_t}, \mathbf{x_{t-1}})$ *Modelo de movimiento del robot*

$\mathbf{R_t}$ *Matriz de covarianza del ruido del modelo de movimiento*

$\mu_{\mathbf{x,j}}$,$\Sigma_{\mathbf{x,j}}$ *Media y covarianza de la distribución gaussiana que representa la pose del robot estimada a partir del modelo de movimiento corregida*

Asociaciones

$\mathbf{N_\Phi}$ *Número total de asociaciones*

$\phi_{\mathbf{j,l}}$ *Probabilidad de asociación*

$\Phi^{\mathbf{k,s}}_{\mathbf{t,1}}$ *Matriz de probabilidad de asociación de las medidas con las balizas*

$\Psi^{\mathbf{k,s}}_{\mathbf{t,1}}$ *Asociación de datos*

$DA_{\mathbf{t,1}}$ *Nivel 1 de la asociación de datos*