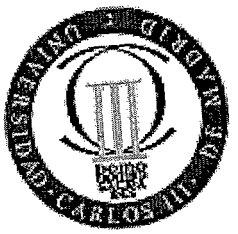


UNIVERSIDAD CARLOS III DE MADRID

Departamento de Ingeniería de Sistemas y Automática

TESIS DOCTORAL

Construcción de un mapa topo-geométrico del entorno y localización de manera simultánea

Autor:
Beatriz López Boada

Director:
Dr. Luis Moreno Lorente

Leganés, 2002

— A mi familia y amigos —

AGRADECIMIENTOS

En primer lugar quisiera dar las gracias a mi director de tesis, D. Luis Moreno, por su inapreciable ayuda y sus buenos consejos, que han permitido el desarrollo y finalización de esta tesis. También quisiera agradecerle su apoyo y confianza mostrados durante todos estos años. Esta tesis no hubiera sido posible sin él.

A D. Miguel Angel Salichs y a D. Carlos Balaguer, por darme la oportunidad de haber realizado mi tesis en el Departamento de Ingeniería de Sistemas y Automática. Al resto de compañeros del Área en el que he vivido momentos muy buenos: Jose Maria, Antonio, Arturo, Jose Manuel, Victor, Milagros, Salah, Alaa, Eladio y en especial a Paco Pepe, Santiago y Mohamed por su inestimable ayuda durante todos estos años. A Carlos Palazuelos que siempre ha estado disponible para lo que le pidiera. A los que han entrado nuevos y afrontan una nueva etapa, para ellos ¡ánimo!.

A mis compañeros de faenas, con los que he compartido risas, trabajo y, por que no, también momentos de decaimiento. Angela, Dolores, Cristina, Mario, Ramiro y Veronica muchas gracias por vuestro apoyo y ayuda. Nunca os olvidaré.

A los que ya no están aquí pero que también han sido importantes: Elena, Vicente, Eva, Susana y en especial a Samuel, la persona de la que he aprendido muchas cosas y que siempre ha tenido unas palabras de apoyo.

A los compañeros de otras áreas y que han hecho que todos estos años hayan tenido muchos más momentos alegres que tristes: Carolina, Manuela, Manuel, Michael, Ángel, ...

A Susana, Mari Mar y Mari Paz, secretarias del Departamento de Electricidad, Electrónica y Automática, por estar siempre disponibles.

A D. Vicente Díaz, Director del Departamento de Ingeniería Mecánica, por apostar por mí en esta nueva andadura profesional.

A los amigos, que siempre han estado a mi lado, aguantando mis problemas con los diagramas de Voronoi y mis charlas sobre temas que ni siquiera comprendían, por ayudarme en todo lo que han podido y darme ánimos cuando más los necesitaba. Y sobre todo por hacerme pasar muy buenos momentos: Ana, Angela, Angélica, Antonio, Blanca, Carolina, Carlos, Cristina, Damian, David, Diego, Elena, Fernando, Ivan, Jesús, Luz, Manuela, Manuel, Mariam, Mario, Mari Carmen, Mari Paz, Mari Mar, Michael, Pablo, Sonsoles, Susana, Veronica, Victor, Yolanda, ... Gracias por todo.

A mi familia y sobre todo a mi abuela Amalia, y a mis tías Eva y Tere que han creído siempre en mí.

A mis hermanos David y María Jesús (con la que he compartido todos los momentos anteriormente citados, además de un trabajo y la realización de una tesis) por aguantarme el mal humor y estar siempre a mi lado.

Finalmente quisiera dar las gracias a las personas más importantes de mi vida, que han dedicado todo su esfuerzo y sus vidas, que siempre me han animado, han creído en mí y han apoyado todo lo que hecho, y a los que debo parte de este trabajo: mis padres, Jesús y Amalia.

RESUMEN

Cuando un robot móvil autónomo se desplaza por un entorno desconocido, la planificación sólo se puede llevar a cabo si va acompañada de un modelado del entorno. Para construir un mapa del entorno, el robot debe conocer su posición relativa a localizaciones pasadas. Debido a los errores de odometría, que se incrementan a medida que el robot se desplaza, es necesario llevar a cabo un proceso de localización que permita estimar la posición del robot en relación a elementos conocidos. De este modo, el problema de generación de un mapa del entorno es referido frecuentemente como el problema de localización y construcción del mapa de manera simultánea, también conocido como problema SLAM (Simultaneous Localization And Mapping).

El método propuesto en esta tesis permite generar un mapa del entorno mientras que el robot se desplaza, al mismo tiempo que se localiza utilizando las características del mapa. Para resolver este problema, el robot, en cada desplazamiento, adquiere información del entorno, a través de un telémetro láser, y almacena la información odométrica.

El entorno es modelado como un grafo, donde los nodos representan lugares típicos de entornos de interiores estructurados, y los arcos representan el camino que unen estos nodos. Tanto los nodos como las ramas son enriquecidos con información geométrica, por lo que el mapa generado es un mapa topo-geométrico. Un Diagrama de Voronoi Local (DVL) es generado directamente a partir de las medidas proporcionadas por un telémetro láser. El diagrama de Voronoi representa el camino, o conjunto de puntos, que son equidistantes a varios objetos presentes en el entorno. De esta manera, el DVL representa el camino más seguro que puede seguir el robot.

El mapa topo-geométrico local es generado a partir del DVL. Este mapa no contiene solamente información topológica, sino que también contiene información geométrica, como es la posición de los nodos y los puntos que caracterizan las ramas.

El algoritmo SLAM presentado en esta tesis alterna dos pasos: 1) un paso de localización y 2) un paso de generación del mapa. El primer paso estima la mejor posición del robot, considerando que los mapas locales no poseen

errores, mientras que el segundo paso genera el mapa global del entorno utilizando las posiciones estimadas en el paso anterior.

El paso de localización utiliza algoritmos genéticos para estimar la posición del robot, contrastando los mapas locales obtenidos en cada nueva posición del robot. La información, tanto topo-geométrica del mapa como odométrica, son utilizadas para estimar la posición del robot y eliminar posibles ambigüedades presentes en el entorno.

El algoritmo de localización alterna dos pasos: 1) un paso *hacia adelante* y 2) un paso *hacia atrás*. El paso *hacia adelante* estima la posición actual del robot. El paso *hacia atrás* estima todas las posiciones del robot. El algoritmo *hacia adelante* se ejecuta en cada nuevo desplazamiento del robot, mientras que, el algoritmo *hacia atrás* solo se ejecuta cuando se cierra un ciclo, es decir, cuando el robot vuelve a pasar por una zona de la que ya tiene información. Este paso de corrección de las posiciones estimadas en instantes de tiempo anteriores hace que el algoritmo dé resultados buenos en entornos de grandes dimensiones con ciclos

El mapa topo-geométrico global es construido integrando la información de los mapas locales obtenidos en cada nueva posición del robot, utilizando las posiciones estimadas en el paso de localización. Este mapa contiene información tanto topológica (nodos y ramas que unen estos nodos) como geométrica (posición de los nodos, puntos que caracterizan las ramas, y distancias de estos puntos a los objetos equidistantes).

La ventaja de este mapa topo-geométrico es que puede ser utilizado para desarrollar varias tareas como son navegación, planificación y localización.

Los resultados experimentales, llevados a cabo en entornos de interiores de grandes dimensiones con ciclos, muestran la robustez y eficacia del algoritmo desarrollado para resolver el problema SLAM.

ABSTRACT

When an autonomous mobile robot moves in an unknown environment, the path planning can only be performed if there is a modeled environment. In order to build a map of the environment, a robot must know where it is relative to past locations. Due to errors in odometry, which are acumulative, it is necessary to localize the robot in relation to already known elements. Thus the problem of mapping is often referred to as Simultaneous Localization And Mapping (SLAM).

The proposed method allows to build a map of the environment and to use the map to self-localize simultaneously. In order to resolve this problem, in each displacement, the robot takes a scan with the laser telemeter and simultaneously stores the odometric information.

The environment is modeled as a graph, in a topo-geometric form, where each node corresponds to a distinctive place, and the arcs are paths which join the nodes. This graph is generated from a Local Voronoi Diagram (LVD), that is built from measurements of the laser telemeter. The Voronoi Diagram represents the way, or set of points which are equidistants to different objects in the environment. In this manner, the LVD is thought to be the safest way to move the robot.

The local topo-geometric map is generated from LVD directly. This map contains not only topological information but also contains geometric information, such as, the nodes position and the edges' points.

This thesis proposes a new SLAM method, which alternates two steps: 1) a localization step and 2) a mapping step. This process is executed every time the robot is displacing. The first step computes the best robot's position, based on the local maps (the best available). The second step generates the global map using the estimated locations in the previous step.

The localization step uses genetic algorithms to estimate the robot's position, matching local topo-geometric maps, which are obtained in each robot's displacement. The topological and geometric information, and the odometric measurements are used to estimate the robot's position and resolve posible ambiguities.

The localization algorithm combines two steps: 1) a forward step and 2) a backward step. The forward step estimates the current robot's position, while the backward step revises a full posterior over poses. The forward step is executed in each iteration, but the backward step is executed only when closing the environment scanning cycle.

The global topo-geometric map is built matching the local topo-geometric maps and making use of the estimated locations in the previous step. This map not only contains topological information, like representative places (doors, hallways, etc.), but also contains their position.

The topo-geometric map is used to execute different tasks such as navigation, path planning and localization.

Experimental results in large and cyclic indoor environments demostrate that our approach is well suited to resolve the SLAM problem.

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

LISTA DE SÍMBOLOS

d_{max}	Alcance del DVL	def. 4.2.1
S_{DVL}	Puntos generadores del DVL	def. 4.2.2
c_i	Grupo generador	def. 4.2.3
$\mathbb{CV}$	Campo de visión	def. 4.5
$\mathbb{CV}_{DVL}$	Campo de visión del DVL	def. 4.2.5
dc_i^j	Distancia entre *grupos generadores*	def. 4.2.6
$dim(c_i)$	Dimensión de un *grupo generador*	def. 4.2.9
PB	Punto bases	def. 4.2.10
d_{ck}^i	Distancia desde una celdilla a un *grupo generador*	def. 4.2.13
CB_i	*Grupo generador* base	def. 4.2.14
R_i	Rama del DVL	def. 4.2.15
N_i	Nodo del DVL	def. 4.2.16
E	Incertidumbre espacial	def. 6.2.2
E_x	Incertidumbre en la dirección x	def. 6.2.2
E_y	Incertidumbre en la dirección y	def. 6.2.2

E_θ	Incertidumbre en rotación	def. 6.2.2
$d(n_i, n_j)$	Distancia entre nodos	def. 6.2.5
l	Longitud cromosoma	sec. 6.2.3.3
f	Función de salud	sec. 6.2.5
$d(p, R)$	Distancia de un punto a una rama del mapa topo-geométrico.	def. 6.2.6
$Proj(p, R)$	Proyección de un punto p a una rama del mapa topo-geométrico	def. 6.2.7
p_{proj}	Punto proyector	def. 6.2.8
$d_{equi}(R_i, R_j)$	Distancia entre dos ramas equivalentes	def. 6.2.9
$factor_\theta(R_i, R_j)$	Factor que mide la orientación entre dos ramas R_i y R_j	ecu. 6.36
$d_\theta(R_i, R_j)$	Distancia entre dos ramas considerando sus orientaciones	ecu. 6.37
g_i	Mapa topo-geométrico local obtenido en el instante i	
$g_{(i,j)}$	Mapa topo-geométrico local obtenido en el instante i referido al sistema de coordenados situado en la posición j	
$\mathbb{CV}_i$	Campo visión del mapa local g_i	def. 7.3.1
$\mathbb{CV}_i^j$	Campo de Visión del mapa g_i referido al sistema de referencia j	def. 7.3.2
MS_i	Mapas con los que se solapa el mapa g_i	def. 7.3.3

ξ_i	Lectura odométrica que caracteriza la acción ejecutada por el robot entre el instante de tiempo $i-1$ y i	
M_i	Mapa topo-geométrico global calculado en el instante i	
X_i	Posición del robot en el instante i respecto al sistema de referencia global utilizando únicamente la información odométrica	
$\widehat{X}_i$	Posición corregida del robot en el instante i referida al sistema de referencia global	
$\widehat{X}_i^j$	Posición corregida del robot con respecto a la posición j referida al sistema de referencia global	
$X_L^{(j,i)}$	Posición i con respecto a la posición j considerando la información odométrica	
$\widehat{X}_L^{(j,i)}$	Posición corregida i con respecto a la posición j	
$\widehat{\mathbb{X}}_L^{(j,i)}$	Conjunto de mejores soluciones de la posición i con respecto a la posición j	def. 7.3.4

1. INTRODUCCIÓN

La característica principal que se requiere a un robot móvil autónomo es que conozca su posición en el entorno por donde se desplaza. Esta característica permite al robot planificar su trayectoria y seguir un camino.

Inicialmente se estimaba la posición del robot utilizando un sistema inercial (odometría). La odometría cuenta las vueltas que da una rueda para estimar el desplazamiento. Debido a los errores propios de los sensores y al deslizamiento de las ruedas se produce un error que se incrementa a medida que el robot se desplaza. Este método, por tanto, no es adecuado para desplazamientos grandes.

Un método alternativo es determinar la posición del robot conociendo la posición que ocupan diferentes marcas en el entorno. Estas marcas pueden ser artificiales o naturales. El inconveniente de este método es que se necesita proporcionar un mapa al robot donde se especifique la posición de estas marcas, de tal manera que, conociendo la posición relativa del robot con respecto a ellas y la posición que ocupan en el entorno, se puede estimar la posición del robot.

Hoy en día, han aparecido mucho trabajos que realizan de manera simultánea la construcción de un mapa y la localización del robot en el entorno por donde se desplaza. A este problema se le denomina SLAM (Simultaneous Localization And Mapping). El problema de construir un mapa del entorno es determinar la localización de lugares de interés (como son marcas u obstáculos), frecuentemente relativos a un sistema de coordenadas global. Para construir un mapa del entorno, el robot debe conocer donde está en relación con localizaciones pasadas. Como el movimiento del robot no es preciso, debido a los errores de los sensores internos y al deslizamiento, el robot debe resolver el problema de conocer su posición actual, cuya dificultad se incrementa a medida que aumenta el tamaño del entorno. La mayoría de los métodos propuestos para resolver el problema SLAM no son consistentes porque no proporcionan buenos resultados en entornos cíclicos de grandes dimensiones.

En esta tesis se presenta un algoritmo que es capaz de construir un modelo

consistente del entorno a medida que el robot se desplaza, y localizar el robot dentro de este mapa. El mapa generado es un mapa topo-geométrico obtenido a partir de la integración de los mapas topo-geométricos locales obtenidos en instantes de tiempo diferentes. Cada mapa topo-geométrico local se genera a partir de un diagrama de Voronoi generalizado utilizando la información procedente de un telémetro láser. El mapa topo-geométrico contiene nodos que representan lugares típicos de entornos de interiores, como son puertas e intersecciones, y ramas que unen estos nodos. Además este mapa es enriquecido con información geométrica, como son los puntos que definen tanto los nodos como las ramas, y las distancias de estos puntos a los objetos que son equidistantes. Esta información geométrica permite eliminar posibles ambigüedades presentes en el entorno y por tanto mejorar el proceso de localización.

Como en cada desplazamiento del robot se produce un error en los valores de odometría, que se incrementan a medida que el robot se desplaza, es necesario corregir la posición del robot. El método utilizado en esta tesis para corregir la posición del robot utiliza algoritmos genéticos, al que se le añade un paso de corrección de las posiciones estimadas en instantes de tiempo anteriores lo que hace que de resultados buenos en entornos de grandes dimensiones con ciclos.

La ventaja principal del mapa topo-geométrico generado en esta tesis es que puede ser empleado para distintas tareas como son navegación, planificación y localización del robot.

1.1 Objetivos de la tesis

El objetivo principal de esta tesis es la construcción de un mapa del entorno a medida que el robot se desplaza. El robot parte de una posición a priori desconocida en un entorno que no conoce. A partir de las medidas proporcionadas por un telémetro láser, el robot empieza a adquirir información del entorno por donde se desplaza construyendo de manera incremental el mapa del entorno. Los sensores internos (odometría) le proporcionan una primera estimación de lo que se ha desplazado, pero esta información tiene errores debido a los errores propios de todo elemento de medida y al deslizamiento de las ruedas. Este error se incrementa a medida que el robot se desplaza. Para construir un mapa del entorno de una manera consistente es necesario corregir la posición del robot, por lo que también se propone un método de localización.

El mapa construido puede ser utilizado para planificación, exploración y

localización del robot. Este mapa es un mapa topológico que contiene nodos que caracterizan lugares típicos de interiores como son puertas e intersecciones, y ramas que unen estos nodos. A este mapa topológico se le añade información geométrica como es las coordenadas de los puntos que definen los nodos y ramas, y las distancias de estos puntos a los objetos a los que son equidistantes. Esta información geométrica permite eliminar las posibles ambigüedades que se puedan encontrar en el entorno mejorando el proceso de localización.

1.2 Estructura de la tesis

El contenido de esta tesis se encuentra distribuido en 8 capítulos.

Inicialmente se presenta un capítulo dedicado a las distintas metodologías desarrolladas y relacionadas con el contenido de este trabajo. Primeramente, se indican métodos diferentes para construir un mapa del entorno, que será utilizado por el robot para navegar, explorar y/o localizarse. A continuación, se explican diferentes métodos que se han desarrollado para localizar un robot dentro de un entorno por donde se desplaza dado un mapa del entorno. Y por último, se mencionan los trabajos relacionados con la construcción de un mapa y localización de manera simultánea (SLAM).

En el capítulo 3 se definen las nociones del diagrama de Voronoi y las aplicaciones que ha tenido y tiene en el campo de la robótica. El diagrama de Voronoi ha sido empleado en esta tesis para la obtención de un mapa del entorno.

En el capítulo 4 se describe el algoritmo desarrollado para la construcción de un diagrama de Voronoi a partir de la información de un telémetro láser. A este diagrama de Voronoi se le llama Diagrama de Voronoi Local (DVL) y será utilizado para la construcción de un mapa topo-geométrico del entorno, para localización y para navegación. A este diagrama se le añade el término local porque solamente contiene información de una zona restringida del entorno por donde se desplaza el robot.

En el capítulo 5 se describe como se obtiene el mapa topo-geométrico local a partir del DVL obtenido según al algoritmo explicado en el capítulo 4. El mapa es un mapa topológico, donde los nodos representan lugares característicos de entornos de interiores (puertas, intersecciones, etc.) y las ramas unen estos nodos. Los nodos y las ramas del mapa son enriquecidos con información geométrica.

En el capítulo 6 se describe el algoritmo que se emplea para corregir la posición del robot, y poder así construir un mapa consistente del entorno. El

algoritmo empleado utiliza algoritmos genéticos restringiendo la búsqueda de la solución de la posición a la zona de incertidumbre espacial. Para estimar la posición se utilizan las características tanto topológicas como geométricas de los mapas construidos según el capítulo 5.

En el capítulo 7 se describe el algoritmo para la construcción de un mapa del entorno a medida que el robot se desplaza y localización de manera simultánea (SLAM). El algoritmo desarrollado permite construir un mapa del entorno de manera consistente, y funciona bien en entornos cíclicos de grandes dimensiones. Esto se debe a que el algoritmo de corrección de la posición añade un paso de corrección de las posiciones anteriores. El mapa se construye integrando la información de los mapas topo-geométricos locales (capítulo 5) obtenidos en cada desplazamiento del robot, realizando previamente una corrección de la posición (capítulo 6).

Por último, en el capítulo 8 se muestran los resultados experimentales obtenidos en entornos de interiores estructurados de grandes dimensiones con ciclos, aplicando el algoritmo desarrollado.

2. ESTADO DEL ARTE

2.1 Introducción

En el campo de la robótica móvil hay varios problemas a resolver, como son el de localización, generación del mapa del entorno por dónde el robot se desplaza, navegación y planificación [SalMor00].

Estos problemas no son independientes unos de otros, sino que están relacionados entre sí. Para que un robot pueda localizarse a medida que se desplaza es necesario que tenga un mapa del entorno; para que un robot pueda generar un mapa del entorno de una manera consistente tiene que conocer su posición; un robot para navegar por el espacio debe planificar su trayectoria y conocer por tanto el mapa por dónde se desplaza.

El trabajo presentado en esta tesis se centra en la generación de un mapa del entorno a medida que el robot se desplaza realizando de manera simultánea una localización. El mapa generado es un mapa topo-geométrico, es decir, es un mapa topológico que contiene información sobre la posición de lugares característicos típicos de entornos de interiores como son intersecciones, estrechamientos, puertas, etc.. Este mapa es generado a partir del DVL (Diagrama de Voronoi Local) [BlaBoaMor00] utilizando la información procedente de un telémetro láser. Aunque en esta tesis no se trata el tema de planificación, este mapa generado se puede emplear para planificar trayectorias.

En este capítulo se presenta un análisis de los trabajos realizados tanto en el campo de localización como de generación de mapas en un robot móvil autónomo de una manera tanto independiente como simultánea.

2.2 Aprendizaje de modelos y mapas de un entorno para un robot móvil autónomo

Mientras que muchos robots móviles pueden navegar por un entorno usando mapas que les han sido proporcionados previamente, muy pocos son capaces de generarlo a medida que se desplazan. Generalmente un humano

debe generar el mapa anteriormente, proporcionando la localización exacta de los obstáculos (mapas geométricos), o un grafo del entorno que representa la conectividad entre las diferentes regiones (mapas topológicos).

La exploración debe ser capaz de liberar al robot de esta limitación. Yamauchi [Yam97] define exploración como *el acto de moverse a través de un entorno, desconocido a priori, mientras que construye un mapa del entorno que puede ser usado luego para navegación.* Una estrategia de exploración buena es aquella que genera un mapa completo o casi completo del entorno en un tiempo razonable.

Una clasificación clásica de los mapas es distinguirlos en geométricos y topológicos. Algunas veces es necesario que un robot conozca exactamente su posición en términos de coordenadas métricas, en este caso, los mapas métricos son la mejor elección. En otros ambientes, como edificios de oficinas con pasillos y habitaciones, o redes de carreteras, los mapas que simplemente especifican la topología de localizaciones importantes y sus conexiones es suficiente. Estos mapas son menos complejos y soportan de una manera más eficiente la planificación que los mapas métricos. Los mapas topológicos son construidos en un nivel bajo de abstracción que permite que el robot se desplace a través de los arcos (quizá siguiendo una pared o una carretera) y reconocer propiedades de las localizaciones. Estos mapas son más flexibles que los geométricos, permitiendo una noción más general de estados e incluir otro tipo de información como es el estado de las baterías del robot.

Aprender mapas geométricos implica usar la información odométrica. De esta manera para espacios muy grandes y débiles, es muy difícil generar un mapa geométrico consistente. Aprender mapas topológicos implica abstraer tanto la información proporcionada por los sensores como las acciones desarrolladas por el robot. Algunos métodos aprenden mapas deterministas sin tener en cuenta el ruido debido tanto a los sensores como a las acciones.

En la literatura se pueden encontrar diferentes tipos de algoritmos que generan un mapa del entorno a partir de la información proporcionada por diferentes tipos de sensores: infrarrojos [AmaEstMán95], ultrasonidos [AntPet01] [SchAdaYam99], láser [MouCha91] [ThrGutFox98] y visión [LevHutBur99].

A continuación se explicarán más detalladamente los tipos de mapas que se pueden emplear.

2.2.1 Mapas geométricos

Los mapas geométricos describen el entorno como un conjunto de objetos o de posiciones ocupadas en el espacio, y las relaciones geométricas entre

ellos.

Cuando se genera un mapa geométrico, que representa un entorno físico, hay dos métodos básicos: los mapas basados en celdillas y los mapas basados en características geométricas.

2.2.1.1 Mapas basados en celdillas

Los mapas basados en celdillas típicamente representan el entorno usando una rejilla uniforme (ver figura 2.1). Cada celdilla tiene un valor que indica la probabilidad de que haya un obstáculo en esa posición del espacio. Los mapas basados en celdillas son muy usados para almacenar y mantener información de ocupación debido a que son muy fáciles de mantener y construir. Sin embargo, es difícil seleccionar la resolución de la celdilla que es adecuada para la representación, y que sirve como base para la representación del entorno. Una localización muy precisa requiere una resolución alta del mapa sobre el espacio entero. Esto implica unos requerimientos altos en cuanto a los datos, e induce a un detalle excesivo del modelado del entorno.

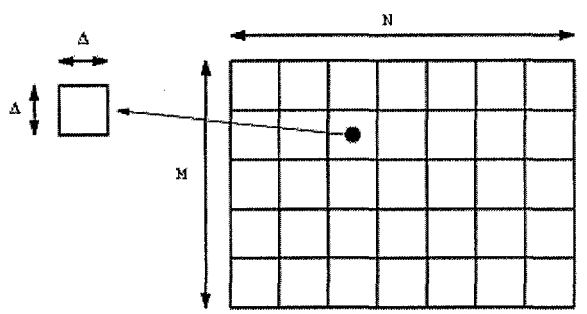
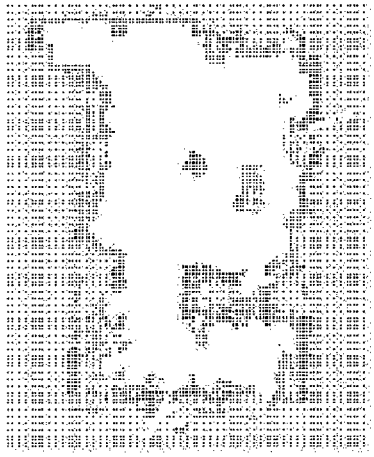


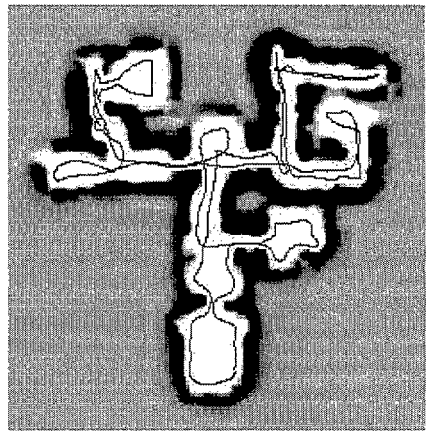
Fig. 2.1: Discretización del espacio en celdillas

Hay una gran cantidad de trabajos relacionados con el aprendizaje de este tipo de mapas. Yamauchi [Yam97] introduce un método para la exploración de entornos desconocidos basado en fronteras. A medida que el robot se mueve en un espacio no explorado añade información al mapa de ocupación. Cassandra et al. [CasKaeKur96] y Thrun [Thr98b] también generan un mapa de ocupación a medida que el robot se desplaza. Leven et al. [LevHutBur99] generan un modelo del entorno geométrico (un mapa basado en celdillas 2D) a partir de los datos aportados por una cámara estéreo. Este modelo almacena de una manera jerárquica la representación geométrica del entorno. Schultz et al. [SchAdaYam99] usan la representación en celdillas para la

representación espacial del entorno donde cada celda, correspondiente a una región espacial, expresa la probabilidad de ocupación. Estos crean dos tipos de mapas basados en celdillas: uno de percepción de corto alcance y otro métrico de gran alcance. Los primeros almacenan los nuevos datos de los sensores que no contienen errores odométricos importantes y que pueden ser usados para evitar obstáculos y para localización. Los segundos tienen una representación del entorno global y son usados para planificación. Roy et al. [RoyBurFox99] generan un mapa de probabilidad basado en celdillas que contiene cada posición del entorno. Esta representación contiene la probabilidad de que los datos de los sensores puedan ser afectados por la dinámica de los obstáculos.



(a) Mapa basado en celdillas generado por Yamauchi et al. [Yam97]



(b) Mapa basado en celdillas generado por Thrun [Thr98b]

Fig. 2.2: Ejemplos de mapas basados en celdillas

No sólo se han utilizado los mapas basados en celdillas para modelar entornos de interiores, sino que también se han empleado para modelar entornos de exteriores donde es necesario un modelo 3D. Trabajos realizados con mapas de celdillas en exteriores se pueden encontrar en: Olson et al. [OlsMat98], Nashashibi et al. [NasFilWri94] y Kweon et al. [KweKan90].

2.2.1.2 Mapas basados en extracción de características geométricas

Los métodos basados en características geométricas, a diferencia de los métodos basados en celdillas, tienden a identificar directamente las características del entorno. Las representaciones geométricas son difíciles de construir, pero son mucho más compactas. En dimensiones grandes los datos requeridos para los modelos geométricos llegan a ser exponencialmente más pequeños que el de los mapas basados en celdillas. Dependiendo del entorno, si es estructurado o no, los objetos se pueden representar mediante diferentes modelos como rectas [DevBul95] en el primer caso , B-splines o super-cuadricas [BreChaDev94] [Cha95] [DevPar98] en el segundo caso. Algunos modelos geométricos también contienen información sobre las relaciones existentes entre los diferentes objetos [BreChaDev95].

Hay autores que utilizan lógica fuzzy para determinar si dos segmentos pertenecen al mismo obstáculo o pared [AmaEstMán95], o lógica fuzzy con redes neuronales (Arquitectura Neuronal Fuzzy ART) [AraAlm98] para organizar los datos extraídos de los sensores.

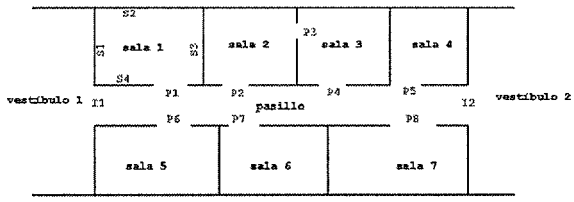
2.2.2 Mapas topológicos

Una alternativa a los mapas geométricos son los mapas topológicos, que son más abstractos. Los mapas topológicos modelan el entorno como un conjunto de estados y sus conexiones, es decir, que estados son alcanzados desde otros estados y en algunos casos que acciones hay que realizar para ir de un estado a otro.

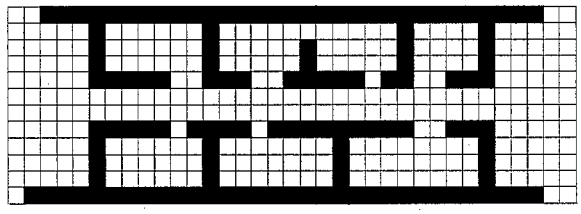
Los estados pueden representar lugares físicos o lugares característicos, como por ejemplo intersecciones, pasillos, puertas, etc. [BoaPalBla02], o pueden representar diferentes acciones que puede llevar a cabo el robot, por ejemplo, ir en línea recta o girar a la derecha [BarEgiSal01].

Kotenkamp et al. [KorWey94] denominan a los lugares característicos como *gateways* (puertas). Según Kotenkamp et al. *los gateways marcan la transición de un espacio a otro.* En entornos de interiores, los gateways son lugares como entradas a habitaciones, intersecciones, etc.. Para un robot móvil un gateway es importante por varias razones:

- Los gateways son lugares altamente visitados.
- Los gateways son lugares que abren nuevas vistas al robot, en las que se pueden extraer nuevas características que permiten distinguir un gateway de otro.



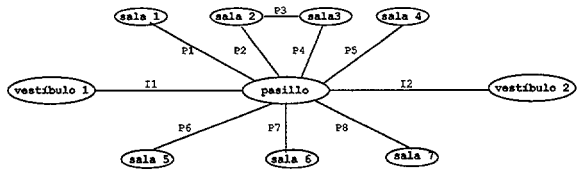
(a) Mapa de un entorno de interiores



(b) Representación basada en celdillas

```
sala1{                    Segmento{
      Segmento S1               Punto p1
      Segmento S2               Punto p2
      Segmento S3              };
      Segmento S4
      Puerta P1
     };
Puerta{                   Punto{
       Punto p1                 double x
       Punto p2                 double y
      };                       };
```

(c) Representación basada en características geométricas



(d) Representación topológica

Fig. 2.3: Distintas representaciones de un entorno

- El robot generalmente debe pasar a través de un gateway en un número pequeño de direcciones.

Típicamente los mapas topológicos son muchos menos complejos y soportan una planificación más eficiente que los mapas geométricos debido a que son mucho más compactos. La resolución de los mapas topológicos depende exclusivamente de la complejidad del entorno. Por otra parte, los modelos topológicos no requieren conocer la posición exacta del robot.

Hay dos tipos de estrategias para generar un mapa topológico: una es aprender el mapa topológico directamente, y otra es generarlo a partir de la información de un mapa geométrico.

Dentro del segundo método se puede encontrar los trabajos realizados por Thrun et al. [Thr98b] y Rekleitis et al. [RekDujDud99]. Thrun et al. [Thr98b] generan gradualmente un mapa de un entorno desconocido a priori. Con los datos proporcionados por los sensores construyen un mapa geométrico basado en celdillas. A partir de este mapa se genera uno topológico, obteniendo primeramente un grafo de Voronoi del que se obtiene diferentes regiones que forman los nodos del mapa topológico. Rekleitis et al. [RekDujDud99] construyen un mapa completo del entorno utilizando grafos planos. Un conjunto de puntos son creados en 2D, y su correspondiente triangulación de Delaunay. El problema de aprender previamente mapas geométricos es que implica conocer la odometría del robot, y si estos mapas además son pobres y de grandes dimensiones, es muy difícil obtener a partir de ellos mapas topológicos consistentes.

Otros trabajos se centran en aprender mapas topológicos directamente, asumiendo que la abstracción de la percepción del robot y las habilidades de acción ya han sido realizadas [Sha98]. Kuipers et al. [KuiByu91] proporcionan una estrategia de aprender mapas topológicos deterministas en un entorno desconocido a priori. Su modelo consiste en un conjunto de nodos y arcos, donde los nodos representan distintos lugares característicos del entorno, y los arcos representan el camino que une los nodos. Ellos suponen que cada estado (nodo) puede ser identificado basándose en la información local o en la distancia atravesada desde un estado previo bien identificado. Este método no funciona bien en aquellas situaciones en las que es necesario una gran secuencia de acciones y observaciones para identificar sin ninguna ambigüedad un estado. Simmons y Koening [SimKoe95] [KoeSim96] aprenden modelos POMDP (modelos topológicos estocásticos) en un entorno de oficinas. Ellos reconocen la dificultad de aprender modelos buenos sin una información inicial, y resuelven este problema proporcionando al robot un mapa topológico junto con algunas restricciones sobre la estructura del modelo.

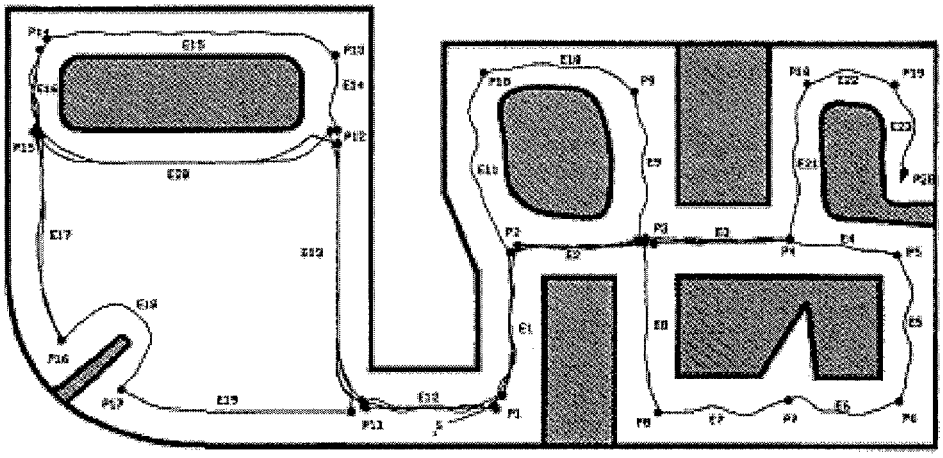


Fig. 2.4: Mapa topológico generado por Kuipers et al. [KuiByu91]

Para aprender los parámetros del modelo utilizan una versión incremental del algoritmo Baum-Welch [Rab89]. Este modelo contiene información métrica, representando los pasillos como un conjunto de cadenas de 1m y permitiendo que el algoritmo de aprendizaje seleccione la longitud de cadena más probable. Este método es bastante efectivo, pero hay que considerar que el tamaño del modelo es proporcional a la longitud del pasillo, y depende fuertemente del modelo inicial que se proporciona. Trabajos cercanos a éste se pueden encontrar en Shatkay et al. [ShaKae97] [Sha98]. Estos aprenden un modelo que combina un modelo del entorno y la interacción del robot con el mundo; esto permite una planificación robusta que tiene en cuenta la probabilidad del error tanto en los datos obtenidos de los sensores como en las acciones. También aprenden un modelo topológico directamente utilizando el algoritmo de Baum-Welch y la información odométrica para mejorar los resultados, pero sin proporcionar un modelo inicial al robot.

2.2.3 Mapas híbridos: mapas topo-geométricos

Hoy en día se tiende a generar mapas que combinen los dos paradigmas anteriores. Es decir, generar mapas topológicos que facilitan la planificación y que posean información geométrica que permita una localización mucho más precisa. A los mapas que contienen tanto información topológica como información geométrica se les denomina mapas topo-geométricos.

La generación del mapa topo-geométrico se basa en la extracción de características, donde cada nodo representa la posición de cada una de estas características, su localización relativa entre cada uno de ellos o incluso puede contener el comportamiento usado para navegar entre los nodos. Estas características se pueden extraer a partir de un mapa basado en celdillas [Thr98b], del diagrama de Voronoi [ZwySimAla00] o directamente a partir de la información proporcionada por los sensores [LevHutBur99] [KoeSim96] [BorBru95] [KunWilNou99].

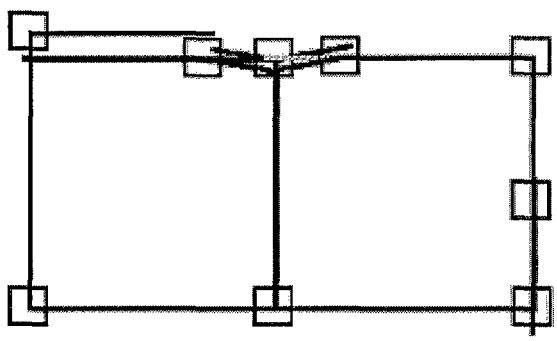


Fig. 2.5: Mapa topológico con información geométrica generado por Kunz et al. [KunWilNou99]

El mapa generado en esta tesis es un mapa topo-geométrico, que representa la topología típica de un entorno de interiores estructurado enriquecido con información geométrica. Este mapa es generado a partir de un diagrama de Voronoi generalizado utilizando la información procedente de un telémetro láser.

2.3 Localización de un robot móvil autónomo

El problema de estimar la posición de un robot móvil dentro de un entorno es un problema fundamental en el campo de la robótica. El conocimiento de su posición permite llevar a cabo las tareas impuestas con precisión y eficiencia. En general, el problema de la localización es estimar la posición del robot, es decir, conocer (x, y, θ) dentro del entorno por donde se desplaza dado un mapa y una información sensorial.

En los últimos años, ha habido una gran cantidad de algoritmos para estimar la posición del robot a partir de los datos proporcionados por los

sensores. En el contexto de robots móviles, el problema de localización se puede definir de la manera siguiente:

- Determinar el modelo del entorno a utilizar, si es geométrico o topológico.
- Estimar la posición del robot dentro del entorno basándose en las observaciones. Estas observaciones consisten, típicamente, en una mezcla de información odométrica acerca del desplazamiento del robot, y la información obtenida de los sensores tanto de proximidad como de cámaras.

Los métodos de localización tienen como objetivo compensar el error odométrico que ocurre durante la navegación utilizando la información sensorial.

Los métodos de localización se pueden clasificar siguiendo diferentes criterios. Fox [Fox98] agrupa los distintos métodos de localización en: locales/globales, entornos estáticos/dinámicos y métodos pasivos/activos. Hay otros autores que clasifican los métodos de localización en probabilísticos/no-probabilísticos, geométricos/topológicos, y basados en marcas/basados en correlación. Cada uno de estos grupos no es excluyente, es decir, pueden existir métodos de localización que se engloben en varios grupos.

A continuación se describen brevemente los diferentes métodos de localización.

2.3.1 Métodos de localización locales y globales

Los métodos de localización locales, *tracking*, son designados para compensar los errores odométricos utilizando los datos de los sensores. Ellos generalmente requieren que la localización inicial del robot sea conocida, y sólo son capaces de seguir la posición del robot. La mayoría de los métodos existentes caen dentro de esta categoría.

La tarea de las técnicas de seguimiento de la posición es corregir la posición del robot debido a los errores odométricos que se producen cuando se mueve de un punto a otro del espacio debido a los errores de los encoders y al deslizamiento. El seguimiento de la posición puede ser considerado como un caso especial de la estimación absoluta del robot, debido a que usa una búsqueda restringida del espacio, generalmente centrada alrededor de la posición estimada del robot, en vez de considerar cada punto del entorno como una posición posible.

Se han desarrollado diferentes técnicas para el seguimiento de la posición realizando un contraste (map matching) de las lecturas de los sensores

con mapas del entorno. Recientemente, más y más técnicas probabilísticas son aplicadas para estimar la posición del robot. Estas técnicas se pueden distinguir acorde al tipo de mapa empleado: métricas o topológicas.

Las técnicas métricas basadas generalmente en mapas de celdillas producen una distribución Gaussiana para representar la estimación de la posición del robot.

Otras técnicas utilizan métodos topológicos. Debido a la representación del entorno en nodos, los métodos topológicos, en contraste con los métodos métricos, permiten situaciones ambiguas en relación a la posición del robot. Estas ambigüedades son representadas por nodos diferentes que tienen una probabilidad alta. Sea como sea, estas técnicas proporcionan una precisión limitada debido a la baja granulación de la discretrización. Esta restricción en la precisión es una desventaja si el robot tiene que navegar rápidamente a través del entorno o incluso llevar a cabo tareas de manipulación.

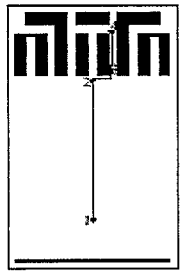
Los métodos de localización basados en celdillas proporcionan una discretización métrica del entorno y de este modo proporcionan una estimación métrica de la posición del robot. Otra ventaja de estos métodos es la posibilidad de explotar todas las lecturas de los sensores y no sólo aquellas que informan de ciertas características, como por ejemplo aquellas que informan de un pasillo, puerta, etc..

Una técnica muy conocida para el seguimiento de la localización del robot es el filtro de Kalman. El filtro de Kalman representa la probabilidad de la posición del robot por una distribución gaussiana unimodal sobre un espacio de estados tridimensional: (x, y, θ). Todos los métodos existentes que utilizan este filtro son prácticamente parecidos en cuanto al modelo de movimiento del robot, y difieren en el modelo sensorial: cómo actualizar la probabilidad gaussiana cuando una nueva lectura de los sensores es recibida. La ventaja de este método es que se obtiene una precisión alta en la estimación de la posición del robot. Sin embargo, la consideración de una distribución gaussiana unimodal implica que se tiene que conocer cual es la posición inicial del robot.

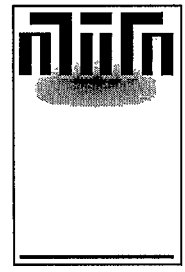
Los métodos de localización globales son más generales. Ellos permiten localizar al robot globalmente, es decir, pueden determinar su posición sin conocer su posición inicial. Recientemente, muchos métodos proponen un nuevo paradigma de localización, la localización de Markov, que permite que el robot se localice sobre una incertidumbre global. Los métodos globales tienen dos ventajas importantes sobre los locales: primero, la localización inicial del robot no tiene que ser conocida, y segundo, ellos proporcionan una mayor robustez, debido a su posibilidad de recuperar su localización a partir

de posiciones falsas.

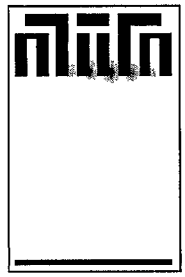
Burgard et al. [BurDerFox98] presentan un método de Localización de Markov Dinámica (DML) que combina la ventaja de ambos métodos, el de seguimiento de la posición del robot y el de estimación global de la posición del robot, consiguiendo: 1) una estimación global de la posición del robot, 2) un seguimiento efectivo de la posición del robot aunque la incertidumbre sea alta, y 3) detectar y recuperar la posición del robot cuando se produce un fallo de localización.



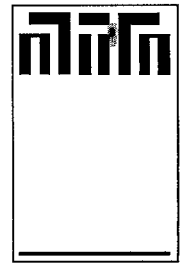
(a) Trayectoria del robot



(b) Distribución de probabilidad para la posición 2



(c) Distribución de probabilidad para la posición 3



(d) Distribución de probabilidad para la posición 4

Fig. 2.6: Localización Dinámica de Markov realizada por Burgard et al. [BurDerFox98]

Lin et al. [LinHanJud98] presentan un sistema basado en visión para el seguimiento de la posición del robot. El sistema obtiene la información sobre la localización del robot reconociendo marcas características típicas de carreteras con un sistema de visión. Las marcas son reconocidas utilizando

redes neuronales para adaptación y optimización. Las probabilidades de la localización del robot son modeladas por una mezcla de gaussianas y probabilidad anterior. Cada gaussiana indica la probabilidad de la localización del robot, con probabilidad y desviación típica. El sistema de localización es muy robusto, ya que, es capaz de recuperar la posición del robot incluso cuando se han producido errores. Cuando una nueva lectura de los sensores internos es recibida, el modelo de probabilidad se actualiza para reflejar el hecho de que el robot se ha desplazado a una nueva posición. Sea cuando sea una marca es reconocida, el modelo de probabilidad debe actualizarse para reflejar la llegada de nueva información para localización.

2.3.2 Métodos de localización en entornos estáticos y dinámicos

Una segunda dimensión referente a los métodos de localización puede ser agruparlos teniendo en cuenta la naturaleza del entorno. La mayoría de los métodos existentes están basados considerando que, acorde a los sensores del robot, solamente lo que hace que sus lecturas cambien es la propia localización del robot. Esto ocurre si el entorno es estático o si los cambios dinámicos que se producen en el entorno no se pueden percibir a través de los sensores del robot. Desafortunadamente, esto no es verdadero para la mayoría de los entornos de interiores, debido a que su apariencia puede cambiar significativamente, porque los muebles cambian de lugar, las puertas pueden estar abiertas o cerradas, o porque hay personas u otros robots que se están desplazando por este entorno.

La mayoría de los métodos sólo funcionan en entornos estáticos, es decir, en entornos donde de acuerdo a las lecturas de los sensores, el único aspecto que puede cambiar con el tiempo es la localización del robot. Sea como sea, estos métodos son "frágiles" en entornos donde el dinamismo es percibido a través de las lecturas de los sensores.

En [IocMasNar01] se describe un método de localización basado en el filtro de Kalman y en el que el entorno es representado por segmentos. Este método es bastante fiable en entornos dinámicos.

2.3.3 Métodos de localización pasivos y activos

Una tercera clasificación, es considerar si el método de localización permite controlar al robot para que este lleve a cabo una mejor localización.

La mayoría de los métodos de localización desarrollados son métodos *pasivos*. La localización pasiva estima la localización del robot teniendo en cuenta exclusivamente los datos proporcionados por los sensores. El resto de consideraciones como son el movimiento del robot, o la dirección del punto de los sensores del robot no pueden ser controlados. La localización *activa* asume que durante la localización, la rutina que lleva a cabo esta misión tiene un control parcial o total sobre el robot, proporcionando la oportunidad de incrementar la eficiencia y la robustez del proceso de localización. Las claves para la localización activa son ¿dónde moverse? y ¿dónde mirar? para que el robot se pueda localizar mejor.

Fox et al. [FoxBurThr98] utilizan un método de localización activa basado en la localización de Markov. La localización de Markov es un método de localización pasivo que ha sido utilizado recientemente. En cualquier instante, la localización de Markov mantiene una densidad de probabilidad sobre una configuración completa del espacio del robot. Sin embargo, no proporciona una respuesta de como controlar los actuadores del robot. Fox *et al.* controlan los actuadores para disminuir las incertidumbres esperadas en el futuro. La incertidumbre es medida por la entropía de distribuciones de probabilidad futuras. El robot es capaz de realizar una localización activa eligiendo las acciones que minimicen la incertidumbre esperada. Inicialmente al robot se le proporciona un mapa métrico del entorno, pero no sabe dónde está.

En [KaeLitCas98] las acciones en el entorno son modeladas como Procesos de Markov Parcialmente Observables (POMDP). El conjunto de acciones y observaciones realizadas por el robot son empleadas para estimar su posición.

2.3.4 Métodos de localización basados en marcas y basados en correlación

Virtualmente todos los algoritmos de localización existentes extraen un pequeño conjunto de características a partir de la información proporcionada por los sensores. El rango de características que se pueden extraer es muy amplio. Se pueden utilizar marcas artificiales como códigos de barras o marcas naturales como son luces, paredes o esquinas.

Una configuración típica para un robot móvil es tener un sistema de posicionamiento relativo como son los encoders, que proporcionan una estimación del desplazamiento del robot respecto a la posición anterior, y unas medidas del sensor (ultrasonidos, láser, sistema de visión) que proporcionan un conjunto de puntos 2D en las coordenadas locales del robot, correspondientes a

la caras visibles de los objetos cercanos a ellos. Esta configuración es adecuada para métodos de localización que están basados en la superposición de modelos o mapas.

Entre muchos de los métodos existentes para localización, el contraste de mapas o *map matching* ha sido estudiado con profundidad en los años pasados y los métodos propuestos pueden agruparse en dos grupos dependiendo de la representación del mapa:

1. Conjunto de puntos.
2. Conjunto de características geométricas.

El primer grupo incluye a los algoritmos que realizan un contraste de mapas usando directamente los datos obtenidos de los sensores. Una característica común de estos métodos es que el proceso de contraste tiende a ser costoso computacionalmente, y en algunos casos los métodos propuestos requieren una optimización dura para implementar una tarea de localización en tiempo-real.

En [FoxBurThr99] hay una representación explícita de la distribución de probabilidad de la posición del robot en el entorno y cada punto del sensor es usado para actualizar la distribución referente al mapa.

El segundo grupo hace uso de características geométricas, lo que requiere un preprocesamiento de los datos de los sensores para extraer las características (o marcas naturales) de los sensores. Muchos de estos métodos extraen líneas, segmentos, esquinas, y de esta manera el mapa es representado por un conjunto de estas características.

Otro de los métodos es el basado en marcas. Los métodos basados en marcas, analizan las lecturas de los sensores para determinar la presencia o ausencia de marcas y estimar así la posición del robot [Thr98a] [ArmMorEsc98].

Fox et al. [FoxBurThr98] proponen una extensión de la localización de Markov que permite a los robots localizarse incluso en entornos altamente poblados. Utilizan dos filtros diferentes para determinar la probabilidad de las lecturas de los sensores. Estos filtros se utilizan para detectar lecturas de los sensores que están "corrompidas" debido a la presencia de personas o a cambios en el entorno.

2.3.5 Métodos probabilísticos y métodos no probabilísticos

Los métodos no probabilísticos estiman la posición del robot sin tener en cuenta la incertidumbre de movimiento ni las características estadísticas de

las medidas. Los primeros trabajos desarrollados en localización se encuentran dentro de este grupo.

En [DubSie98] se presenta un método de seguimiento de la posición y orientación del robot a partir de la información proporcionada por un láser en un entorno parcialmente estructurado donde se presenta un extensión de la transformada de Hough. Este método busca líneas que son clasificadas según su dirección utilizando una red neuronal para acelerar el proceso. Compara dos lecturas consecutivas para determinar la orientación y posición del robot.

En [KnoNouMor98] se aplica técnicas de IA (Inteligencia Artificial) para localizar al robot en un entorno de interiores. A partir de la información proporcionada por los sensores de ultrasonidos se extraen marcas del entorno, como esquinas y paredes, que son comparadas con las características almacenadas en un mapa que ha sido previamente almacenado en el robot. En este método se realizan dos tipos de localización: una localización longitudinal, y otra transversal.

En [AntPet01] se presenta un método basado en la detección de características, como son líneas y esquinas, a partir de la información proporcionada por los sensores de ultrasonidos, y no requiere unas características estadísticas de las medidas obtenidas. Las características detectadas son contrastadas con las almacenadas en el mapa global. Tiene un coste computacional bajo, ya que, sólo se utiliza la información de los sensores obtenida en el instante de tiempo actual, y no en instantes anteriores.

Recientemente han aparecido gran cantidad de algoritmos que tienen en cuenta tanto los errores producidos por los sensores externos como internos, son los llamados métodos probabilísticos. Los métodos probabilísticos están basados en estimar la posición más probable dada toda la información del entorno procedente de los sensores. Esta tarea es generalmente llevada a cabo contrastando la información procedente de los sensores con un mapa de referencia dado, en orden a determinar la posición absoluta del robot en este mapa. La información procedente de los sensores es integrada generalmente con la información odométrica para incrementar la precisión de la localización.

Los métodos probabilísticos combinan dos pasos:

1. **Observación**: A intervalos regulares, el robot requiere las lecturas proporcionadas por los sensores. Estas medidas son usadas para refinar la probabilidad de donde se encuentra el robot en el entorno. Este proceso generalmente reduce la incertidumbre del robot.

2. **Acción**: Cuando el robot ejecuta un comando de acción, la probabili-

dad de su localización es actualizada. Como el movimiento del robot es inexacto debido al deslizamiento y deriva, la incertidumbre se incrementa.

Los métodos probabilísticos más conocidos son el filtro de Kalman, la localización de Markov y más recientemente la localización de MonteCarlo. Sin embargo, hay otros autores que no utilizan ninguno de ellos. Por ejemplo: en [OlsMat98] se lleva a cabo una localización en terrenos naturales contrastando mapas. Un mapa de ocupación del terreno es primeramente generado usando visión estéreo. Este mapa local es comparado con un mapa global usando una medida derivada de la distancia de Haussdorff. Una versión probabilística de esta medida ha sido formulada usando el principio de Máxima Probabilidad. Utiliza los datos odométricos como una primera estimación de la posición del robot.

En [BurFoxHen96a] [BurFoxHen96b] se describe un método probabilístico de localización para estimar la posición absoluta del robot (método global) utilizando un mapa basado en celdillas. Este método se puede implementar con diferentes tipos de sensores y no requiere el conocimiento de la posición inicial del robot. Es un método bayesiano basado en la incertidumbre de las celdillas. Cada celda almacena la probabilidad de ocupación de la celda. Estas probabilidades son obtenidas integrando la probabilidad de las lecturas de los sensores en cada instante de tiempo. Para cada mapa local, este valor de incertidumbre es obtenido activando los sensores varias veces y acumulando en x la probabilidad de los valores obtenidos de los sensores supuesto que el centro de esta incertidumbre es la posición del robot en el entorno modelado por m. En cada disparo de los sensores, dos pasos se llevan a cabo:

1. **Integración de las lecturas de los sensores**: Para estimar la posición del robot es necesario integrar las lecturas de los sensores. Supuesto el modelo m del entorno, y $p(x|s_1 \wedge \cdots \wedge s_{n-1} \wedge m)$ la probabilidad (posterior) de que el robot esté en x, dado m y las medidas $s_1, \cdots, s_{n-1}$, entonces, para actualizar la probabilidad de x dada la nueva entrada de los sensores s_n, se utiliza la fórmula de actualización siguiente:

$$p(x|s_1 \wedge \cdots \wedge s_{n-1} \wedge s_n \wedge m) = \alpha p(x|s_1 \wedge \cdots \wedge s_{n-1} \wedge m) \cdot p(s_n|x \wedge m) \tag{2.1}$$

donde $p(s_n|x \wedge m)$ es la probabilidad de observar s_n dado el modelo del mundo m y considerando que x es la posición actual del robot. Para inicializar las celdillas se usa la probabilidad inicial $p(x|m)$ de x

referida a la posición actual del robot. El cálculo de $p(x|m)$ y $p(s_n|x \wedge m)$ depende del modelo del mundo dado y del tipo de sensores utilizado.

2. **Integración de los movimientos del robot**: Para actualizar la probabilidad de las celdillas con respecto al movimiento del robot e indicar los errores odométricos que se pueden cometer, se usa la fórmula procedente de las cadenas de Markov. Se considera cada celda en P como un estado posible del robot, y se determina la probabilidad de transición de un estado a otro $p(x|\widetilde{x} \wedge \tau \wedge t)$ para cada par $(x, \widetilde{x})$ de las celdas en P, que depende de la trayectoria τ realizada por el robot en ese instante de tiempo t. La nueva probabilidad del mapa de celdillas después del movimiento del robot es:

$$P[x] := \alpha \cdot \sum_{\widetilde{x} \in P} P[\widetilde{x}] \cdot p(x|\widetilde{x} \wedge \tau \wedge t) \tag{2.2}$$

donde α es un factor de normalización. En cualquier instante de tiempo el valor de la celda con un valor más alto de P representa la posición actual del robot.

A continuación se explicarán brevemente algunos de los métodos probabilísticos:

2.3.5.1 Filtro de Kalman

El filtro de Kalman es una técnica de actualización recursiva empleada para determinar los parámetros más verosímiles del modelo de un proceso. Dada una estimación inicial, el filtro de Kalman permite predecir los parámetros del modelo y ajustarlos en cada medida, proporcionando una estimación del error en cada actualización. Permite incorporar los efectos del ruido (tanto de las medidas como del modelo).

El filtro de Kalman se basa en dos modelos:

- **Modelo de planta**. El modelo de planta describe como varía la posición del robot teniendo en cuenta las acciones que ha realizado. Las ecuaciones que definen la nueva posición del robot son:

$$x(k+1) = F(k)x(k) + G(k)u(k) \tag{2.3}$$

donde $x(k)$ es el vector de estado del robot en el instante de tiempo $t = k$ y $u(k)$ es el ruido de entrada.

- **Modelo de medida.** El modelo de medida describe como el sensor transforma las variables espaciales del vector de estado en variables sensoriales. Las ecuaciones del modelo de medida son:

$$z(k) = H(k)x(k) + v(k) \tag{2.4}$$

donde $z(k)$ es el vector medidas y $v(k)$ es el ruido de medida.

El algoritmo de estimación comienza con la estimación inicial de $\widehat{x}(0|0) = x(0)$ y de la covarianza inicial $P(0|0)$ que se supone conocida.

Las ecuaciones matemáticas del filtro de Kalman son:

- Predicción

$$\begin{aligned} \widehat{x}(k+1|k) &= F(k) \cdot \widehat{x}(k|k) \\ P(k+1|k) &= F(k) \cdot P(k|k) \cdot F(k)^T + G(k) \cdot Q(k) \cdot G(k)^T \end{aligned} \tag{2.5}$$

- Estimación

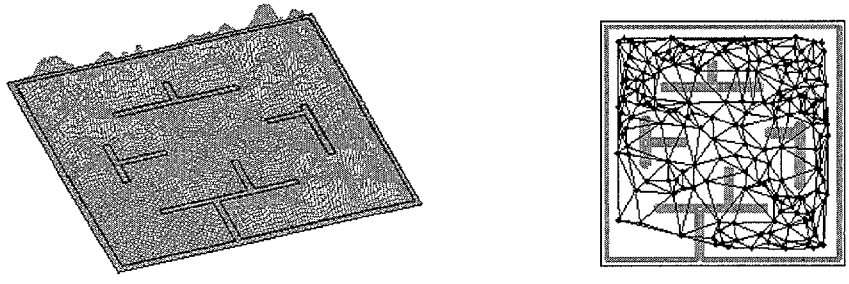
$$\begin{aligned} K(k+1) = \;& P(k+1|k) \cdot H(k+1) \cdot (H(k+1) \cdot P(k+1|k) \cdot \\ & H(k+1)^T + R(k+1))^{-1} \\ \widehat{x}(k+1|k+1) = \;& \widehat{x}(k+1|k) + K(k+1) \cdot (z(k+1) - H(k+1) \cdot \\ & \widehat{x}(k+1|k)) \\ P(k+1|k+1) = \;& (I - K(k+1) \cdot H(k+1)) \cdot P(k+1|k) \end{aligned} \tag{2.6}$$

En [NeiHorTar96] se presenta una versión 2D del Modelo de Simetrías y Perturbación (SP), en el que se integra un EKF (Filtro de Kalman Extendido) y un modelo de representación probabilístico para representar la incertidumbre, es decir, un mecanismo apropiado para la integración de la información procedente de varios sensores. Dos tipos de informaciones sensoriales permiten al robot localizarse en un entorno conocido: el rango y la intensidad de las imágenes procedentes de un sensor láser.

En [FilDev93] se describe un método que estima la mejor posición del robot en el entorno, a partir de la información proporcionada un telémetro láser. Estiman la posición del robot contrastando las características de los elementos geométricos percibidos en muchas imágenes. Para re-estimar la posición del robot y las características percibidas del entorno la agregación de modelos sucesivos se lleva a cabo utilizando el Filtro de Kalman Generalizado (GKF). Este método de localización se basa en la detección de características naturales:

- Bordes verticales 3D, que corresponden a postes o columnas.
- Bordes horizontales 3D, que corresponden a paredes.

En [VlaTsa98] se presenta un método para la localización probabilística de un robot móvil. Utilizan el modelo SUF (Sensory Uncertainty Field) para modelar la incertidumbre del robot. Extienden el concepto de SUF para entornos dinámicos y desconocidos. Proponen un modelo de red neuronal auto-organizativa para construir y mantener una estimación del SUF, mientras el robot se mueve alrededor del espacio, basado en información dinámica de localización, por ejemplo, el Filtro de Kalman. La configuración del robot y su incertidumbre son introducidas a una red neuronal auto-organizativa que lleva a cabo un agrupamiento (clustering) adaptativo y probabilístico de la configuración del espacio libre. Las redes actualizan los parámetros de acuerdo a las entradas de las observaciones, aproximando una Función de Densidad de Probabilidad (PDF) sobre el espacio de entrada. Para dimensiones pequeñas, 2D y 3D, el resultado del PDF es aproximado mediante polígonos usando la triangulación de Delanay, un método geométrico para la construcción de celdillas irregulares que optimizan algún criterio. Esta triangulación sobre el centro de la agrupación (cluster) proporciona adicionalmente un grafo de conectividad del espacio libre del robot sobre el que se puede realizar una planificación de trayectorias.



(a) PDF estimado (b) Triangulación de Delanay

Fig. 2.7: Estimación del SUF del robot por Vlassis et al. [VlaTsa98]

2.3.5.2 Localización de Markov

La idea principal de la localización de Markov es calcular la distribución de probabilidad sobre todas las localizaciones posibles del entorno. $Bel(L_t = l)$

denota la probabilidad del robot de estar en la posición l en el instante t. Donde l es una localización en el espacio (x, y, θ), x e y son las coordenadas cartesianas y α es la orientación del robot. Inicialmente, $Bel(L_o)$ refleja el estado inicial de conocimiento: si el robot sabe cual es su posición de inicio, $Bel(L_o)$ está centrada en la posición correcta; si la posición inicial del robot no es conocida $Bel(L_o)$ es una distribución uniforme que representa la incertidumbre global del robot.

La probabilidad Bel es actualizada como sigue:

- **El robot se mueve.** El movimiento del robot es modelado por la probabilidad condicional $p_a(l|l')$. $p_a(l|l')$ indica la probabilidad de que la acción de movimiento a, cuando es ejecutada en l', lleve al robot a la posición l. $p_a(l|l')$ es usada para actualizar la probabilidad sobre el movimiento del robot, donde $\widehat{Bel}(L_t = l)$ denota la probabilidad resultante en el instante t:

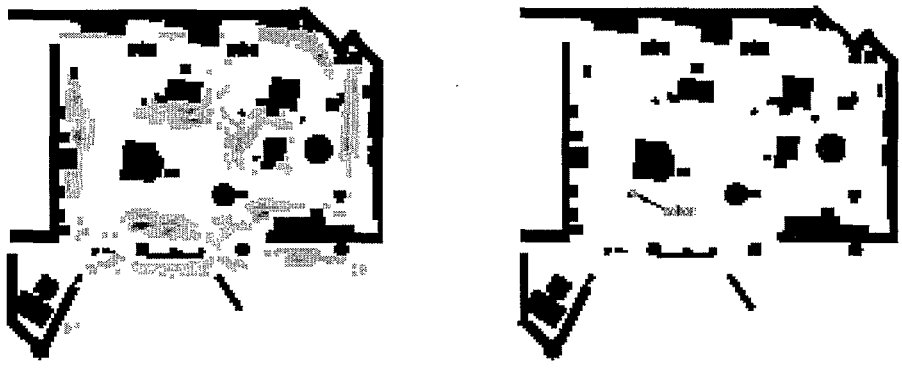
$$\widehat{Bel}(L_t = l) \longleftarrow \sum_{l'} p_a(l|l') Bel(L_{t-1} = l') \tag{2.7}$$

- **El robot percibe con los sensores.** s indica una lectura de los sensores y $p(s|l)$ la probabilidad de percibir s en l. La probabilidad $p(s|l)$ es generalmente referida como a un mapa del entorno, ya que, especifica la probabilidad de observaciones en localizaciones diferentes del entorno. Cuando se percibe s, la probabilidad es actualizada acorde a la regla siguiente:

$$Bel(L_t = l) \longleftarrow \frac{p(s|l)\widehat{Bel}(L_t = l)}{p(s)} \tag{2.8}$$

En [Thr98a] [Thr98b] se presenta un algoritmo llamado BaLL (Bayesian Landmark Learning). BaLL posibilita que el robot aprenda marcas por sí mismo, y que aprenda rutinas para su reconocimiento usando redes neuronales artificiales. El resultado de esta red es utilizado luego para estimar la posición del robot usando la localización de Markov. El robot inicialmente no conoce su posición inicial.

En [MarThr98] se utiliza un método de localización similar a la localización de Markov. Es un método que estima probabilísticamente la posición del robot en un entorno conocido. La representación que se hace del espacio es



(a) Distribución de probabilidad de la posición del robot al incorporar una sola información sensorial

(b) Distribución de probabilidad de la posición del robot al incorporar varias informaciones sensoriales

Fig. 2.8: Localización global utilizando la localización de Markov según Fox et al. [FoxBurThr99]

un mapa basado en celdillas. El método presentado aquí permite entrenar a un robot para reconocer un objeto específico utilizando un árbol de decisión.

Fox et al. [FoxBurThr99] presentan una técnica de localización llamada localización de Markov. Esta técnica es una técnica global que estima la posición del robot dado un modelo del entorno. Este método utiliza un espacio de estados métrico discretizado. Las ventajas que presenta este método son: 1) proporciona una estimación bastante precisa sobre la posición del robot lo que le permite llevar a cabo las tareas de una manera eficiente, 2) utiliza directamente la información procedente de los sensores. Este método puede incluso utilizarse en entornos dinámicos, en el que las lecturas de los sensores no corresponden con las medidas esperadas.

2.3.5.3 Localización de MonteCarlo

La idea principal de la localización de MonteCarlo (MCL) es representar la incertidumbre posterior $Bel(l)$ por un conjunto de N muestras aleatorias o partículas con un peso, $S = \{s_i | i = 1, \cdots, N\}$. Un conjunto muestra constituye una aproximación discreta de la distribución de probabilidad. Las muestras en MCL son del tipo:

$$\langle \langle x, y, \theta \rangle, p \rangle \tag{2.9}$$

donde $\langle x, y, \theta \rangle$ indica la posición del robot, y $p \geq 0$ es un factor de peso numérico, análogo a la probabilidad discreta. Para consistencia se asume $\sum_{n=1}^{N} p_n = 1$.

De la misma manera que la localización de Markov, el método de localización de MonteCarlo se desarrolla en 2 fases:

- **El robot se mueve.** Cuando el robot se mueve, el MCL genera N nuevas muestras que se aproximan a la posición del robot después del comando de movimiento. Cada muestra es generada aleatoriamente dibujando una muestra del conjunto calculado anteriormente, con una probabilidad que viene dada por sus p valores. l' indica la posición de esta muestra. La nueva muestra de l es luego generada creando una muestra simple y aleatoria a partir de $P(l|l', a)$ y usando la acción observada a. El valor p de la nueva muestra es N^{-1}. El conjunto de muestras se aproxima a distribuciones en las que la incertidumbre se incrementa, representando la pérdida gradual de información de la posición del robot debido al deslizamiento y deriva.

- **El robot percibe con los sensores.** Las lecturas de los sensores son incorporadas al conjunto de muestra, implementando la regla de Bayes en la localización de Markov. Más especificamente, se supone que $\langle l, p \rangle$ es una muestra. Entonces:

$$p \longleftarrow \alpha P(s|l) \tag{2.10}$$

  donde s es la medida del sensor, y α es la constante de normalización que fuerza $\sum_{n=1}^{N} p_N = 1$. La incorporación de las lecturas de los sensores es típicamente realizada en dos fases, una en la que p es multiplicado por $P(s|l)$, y otra en la que los valores de p son normalizados.

En [FoxBurDel99] se presenta un algoritmo de localización llamado Localización de MonteCarlo (MCL), que es una versión de la localización de Markov. El MCL difiere de otros métodos en que usa una muestra aleatoria para representar la probabilidad. El algoritmo MCL reduce el tiempo de cálculo con respecto a otros métodos en el que se estudian todo el espacio de soluciones posibles y es mucho más fácil de implementar que otros métodos.

2.3.6 Métodos de localización geométricos y topológicos

Otra clasificación de los métodos de localización es distinguirlos en geométricos, topológicos o híbridos, dependiendo del tipo de mapa que se emplee.

Los métodos geométricos usan generalmente un mapa basado en celdillas bidimensional. Ellos intentan seguir la posición del robot con respecto a un sistema de coordenadas del mapa.

Los métodos topológicos usan un grafo como representación del entorno. Ellos tienden a determinar el nodo del grafo que corresponde con la localización del robot.

Los métodos híbridos combinan los métodos tanto geométricos como topológicos.

La mayoría de los métodos llevan a cabo una localización geométrica. En general estos métodos están basados en el contraste de mapas (map matching) o en la detección de marcas. Muchos de los sistemas basados en el contraste de mapas utilizan el Filtro de Kalman junto con la información procedente de los sensores internos. Aquí es necesario proporcionar un modelo bueno de los sensores y de la incertidumbre debido al movimiento del robot.

La localización basada en marcas se puede realizar utilizando marcas naturales o artificiales. Las marcas artificiales son más fáciles de detectar que las marcas naturales. Sea como sea, las marcas artificiales requieren modificaciones del entorno, por lo que los métodos basados en marcas naturales son preferidos. Varias características pueden ser utilizadas como marcas naturales: esquinas, puertas, luces de los techos, difusores de aire, etc.. Debido a que los métodos de localización basados en marcas son adaptados a entornos específicos, ellos raramente son aplicados a entornos diferentes.

Los métodos métricos, donde la posición del robot es representada por (x, y, θ), basados en el filtro de Kalman permiten una precisión alta y complejidad baja cuando se usan características basadas en líneas a partir de la información procedente de ultrasonidos, cámaras CCD o láser. Sin embargo, el estimador de Kalman es unimodal, de esta manera no se puede detectar y recuperar la posición del robot desde una posición de pérdida (por ejemplo, una situación donde el robot no puede contrastar características correctamente debido a que la diferencia entre la posición estimada y real es demasiado grande). Otros métodos que no sufren estas limitaciones son los métodos basados en celdillas y/o topológicos. Dentro de este grupo se encuentran los Procesos de Markov Parcialmente Observables (POMDP), la localización de Markov y recientemente la localización de Montecarlo.

En [Fox98] y [BurDerFox98], entre otros, se presenta un método de localización basado en un mapa de celdillas.

Autores que utilizan un método de localización basado en marcas son Margaritis et al. [MarThr98]. Ellos usan un árbol de decisión para reconocer características específicas del entorno, como por ejemplo, sillas.

En [UlrNou00] se presenta un método de localización topológico que funciona bien tanto en entornos de exteriores como de interiores estructurados. Este método usa un sistema de visión de color que trabaja en entornos reales y que es fácilmente entrenable para nuevos entornos. Se usa un grafo adyacente donde los nodos representan localizaciones y los arcos relacionan estos nodos. Este sistema de localización debe ser entrenado previamente. Durante el entrenamiento, las imágenes de entrada son comparadas con las imágenes de referencia del mapa. La localización cuya referencia de imagen contraste mejor es considerada como la nueva localización. De este modo, el objetivo del entrenamiento es asignar a un conjunto de imágenes representativas una localización.

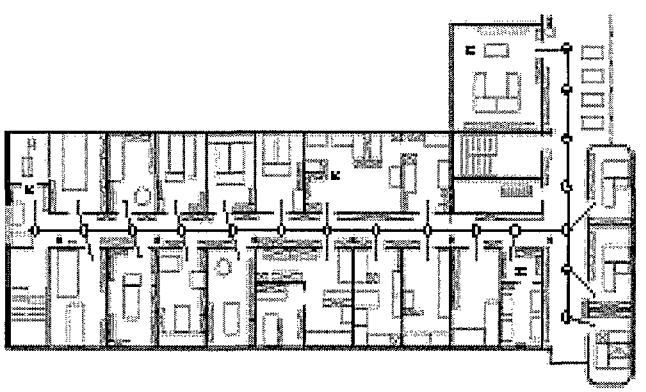


Fig. 2.9: Mapa del entorno y su grafo correspondiente del trabajo de Tomatis et al. [TomNouArr01]

En [TomNouArr01] se combina los dos paradigmas, localización geométrica y topológica. El modelo geométrico consiste en un conjunto finito de líneas que pertenecen al mismo lugar. Estos lugares son relacionados por medio de un mapa topológico que está compuesto de nodos y ramas. La localización métrica es realizada mediante un Filtro de Kalman Extendido. Este método garantiza una alta precisión con baja complejidad y pocos requerimientos de memoria. La navegación topológica usa POMDP. Esto permite una planificación eficiente en entornos grandes, tiene la ventaja de una representación simbólica para la interacción hombre-máquina y una robustez frente a la pérdida de la localización gracias a la multi-modalidad. Las características que se usan para la representación topológica son: 1) discontinuidades perpendiculares a la dirección de movimiento en el pasillo, 2) las puertas son usadas para la estimación del estado y para el modelo de transición y 3) los pasillos

perpendiculares a la dirección de movimiento creando una intersección.

En [NagCho99] [NagChoThr98] se propone un método de localización usando un mapa topológico parcialmente construido a partir del Diagrama de Voronoi Generalizado (GVG). Proponen una técnica de contraste topológica para la localización. El GVG es el conjunto de puntos equidistantes al menos a dos objetos. Introducen el concepto de punto de encuentro (meet point) que es un punto del espacio que es equidistante a tres o más objetos. Este método se realiza contrastando los puntos de encuentro.

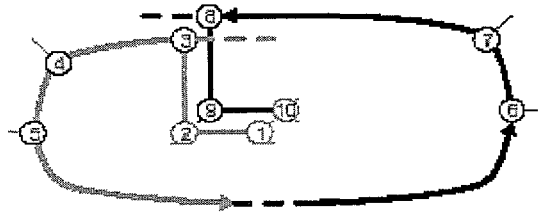


Fig. 2.10: Localización basada en el contraste de 2 mapas topológicos según Nagatani et al. [NagCho99]

En Blanco et al. [BlaBoaMor01] utilizan un método de localización basado en el contraste de características del diagrama de Voronoi obtenido a partir de la información de un telémetro láser. Estas características son comparadas con las características del diagrama de Voronoi del entorno que le ha sido dado inicialmente al robot.

2.4 Localización y construcción del mapa del entorno de manera simultánea (SLAM)

El problema de generar un mapa del entorno a medida que el robot se desplaza ha recibido una atención considerable en los últimos años. El problema de generar un mapa es el problema de generar modelos del entorno por donde se desplaza el robot a partir de la información proporcionada por los sensores. La importancia de este problema es debido al hecho de que muchos robots móviles para llevar a cabo una tarea requieren mapas para operar. Algunos métodos buscan una descripción topológica del entorno, mientras que otros buscan más detalle, generando mapas geométricos. Csorba [Cso97] y Dissanayake et al. [DisNewCla01] resuelven el problema SLAM utilizando un mapa basado en características geométricas. Yamauchi et al.

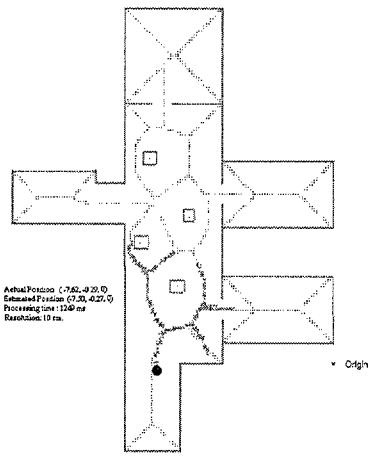


Fig. 2.11: Localización basada en el contraste de diagramas de Voronoi según Blanco et al. [BlaBoaMor01]

[YamSchAda98] resuelven el problema SLAM utilizando un mapa basado en celdillas y Zwynsvoorde [Zwy00] resuelve el problema utilizando un mapa topo-geométrico del entorno extraído a partir del diagrama de Voronoi.

Para generar un mapa del entorno se tiene que tener en cuenta dos tipos de ruidos: ruido de percepción, debido a los sensores, y ruido de movimiento, debido a los errores en la odometría. Debido al último de ellos, el problema de generar un mapa conlleva inherentemente un problema de localización, que es el problema de determinar la posición del robot relativa a su propio mapa. El problema de generación del mapa del entorno es de esta manera referido como el **CML** (Concurrent Mapping and Localization) [ThrBurFox98a] [ThrBurFox98b] o el problema **SLAM** (Simultaneous Localization and Mapping) [LeoWhi91] [CasMarNei98]. Para lograr el SLAM se pueden emplear diferentes tipos de sensores: ultrasonidos, láser o visión [SeLowLit01].

Hay varios errores como el ruido asociado tanto a la información obtenida de los sensores como al error cometido en el desplazamiento del robot. En los mapas estocásticos o probabilísticos, un filtro es utilizado para mantener una estimación de la posición tanto de las marcas como de la posición del robot. Considerar esta incertidumbre aumenta la complejidad computacional. Se pueden clasificar los métodos SLAM en deterministas o no probabilísticos, y estocásticos o probabilísticos, dependiendo si consideran o no la incertidumbre.

2.4.1 Métodos probabilísticos

Los trabajos relacionados con la resolución del problema de SLAM de manera probabilística se pueden clasificar en dos grandes grupos, cada uno de los cuales tiene sus ventajas e inconvenientes.

El primer grupo, por razones históricas, está formado por aquellos métodos que utilizan el Filtro de Kalman. Estos métodos permiten una construcción on-line de los mapas. Los filtros de Kalman representan la probabilidad por Gaussianas unimodales; de este modo se requiere $O(N^2)$ parámetros para representar el instante de tiempo posterior sobre un espacio N-dimensional. Esto limita el número de características que pueden ser modeladas en el mapa. Otras de las limitaciones, que se puede decir que es la más importante, es que debe existir una correcta asociación entre las medidas y las características del mapa. Esto es, las características en el entorno deben poseer una única posición que permita al robot eliminar posibles ambigüedades. En la práctica, esto raramente se cumple. De este modo, muchas implementaciones llevan a cabo métodos de asociación de datos utilizando máxima probabilidad, que tienden a trabajar bien cuando los errores en la posición son pequeños pero son frágiles en el caso de que los errores sean grandes.

El segundo grupo está basado en el algoritmo de Máxima Expectación de Dempster (EM). Los métodos EM [DemLaiRub77] resuelven el problema de asociación de datos realizando una búsqueda *hill-climbing* en el espacio de todos los mapas que constantemente refina la asociación de datos. Estos métodos han sido aplicados con éxito en entornos cíclicos de grandes dimensiones con una gran cantidad de características ambiguas. Sea como sea, ellos son inherentes a los algoritmos de lotes, requiriendo pasos múltiples a través del conjunto entero de datos. Como consecuencia, ellos no son aplicables a problemas de construcción de mapas on-line.

2.4.1.1 Trabajos basados en el filtro de Kalman

En esta sección se describen algunos trabajos que han resuelto el problema SLAM utilizando el filtro de Kalman.

Leonard et al. [LeoWhi91] utilizan el Filtro Extendido de Kalman para la localización del robot, basado en el contraste (map matching) de características (planos, esquinas, y cilindros) obtenidas a partir de la información proporcionada por los ultrasonidos.

Bulata et al. [BulDev96] presentan un algoritmo en el que el robot construye incrementalmente mapas sucesivos (snapshots) a partir de la información proporcionada por un láser, y son fusionados en un modelo global para

poder localizar al robot con respecto a un sistema de referencia pertinente. A partir de la información del láser se extraen marcas que corresponden a características locales como esquinas, pasillos, intersecciones y puertas. Cada marca tiene su propio sistema de referencia y su propio modelo geométrico. Luego las marcas correlacionadas son mantenidas juntas en un nivel de área, finalmente el modelo global es construido a partir de las relaciones entre estas áreas, proporcionando una descripción topológica del mundo. Para cada nivel (marca, área, entorno) el modelo es representado por un vector y una matriz de covarianzas, que es actualizado usando el Filtro Extendido de Kalman.

Castellanos et al. [CasTarSch97] presentan un algoritmo para la relocalización y generación del mapa del entorno de interiores por donde se desplaza el robot. Usan el Filtro Extendido de Kalman para procesar la información proporcionada por un láser y los datos odométricos. El modelo SPM (Symmetries and Perturbations Model) es utilizado para modelar la incertidumbre de la información geométrica. Las características geométricas obtenidas de los sensores son segmentos. En [CasMarNei98] se emplea el mismo método utilizando además un láser y un sistema de visión para disminuir las incertidumbres. Las características geométricas que obtienen son esquinas y semiplanos.

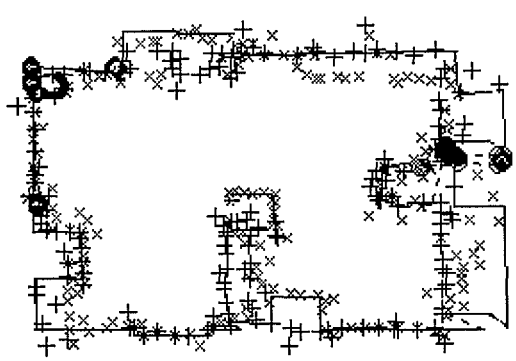


Fig. 2.12: Algoritmo presentado por Zhang et al. [ZhaGho00] que resuelve el problema SLAM utilizando EKF y características geométricas

Delahoche et al. [DelPégMou98] presentan un sistema de navegación que permite a un robot móvil localizarse en un entorno parcialmente conocido. Estos integran un módulo de actualización del mapa permitiendo al robot estimar la posición de nuevas marcas verticales a lo largo del camino. Utilizan el Filtro Extendido de Kalman para estimar la posición del robot y para

extraer observaciones que luego serán usadas para determinar la posición de marcas no listadas. La integración de nuevas marcas en el mapa global se realiza mediante una matriz de covarianzas asociada con cada marca no listada.

Zhang et al. [ZhaGho00] utilizan el Filtro Extendido de Kalman basado en características geométricas como segmentos, arcos circulares y conjuntos de puntos para resolver el problema SLAM. La construcción del mapa tiene dos partes: 1) la construcción de un mapa local basada en medidas locales, y 2) la construcción de un mapa global actualizado basada en el mapa local y en el resultado de la localización.

Victorino et al. [VicRivBor00] han desarrollado un método aplicado a interiores utilizando la información proporcionada por un telémetro láser que resuelve el problema SLAM. Un mapa de carreteras (roadmap), basado en el Diagrama de Voronoi Generalizado (DVG), es generado para construir incrementalmente un mapa del entorno. De esta manera, el robot puede explorar un entorno desconocido a priori. El proceso de localización y generación del mapa es llevado a cabo probabilísticamente fusionando la información del láser y los datos odométricos gracias al Filtro de Kalman Extendido.

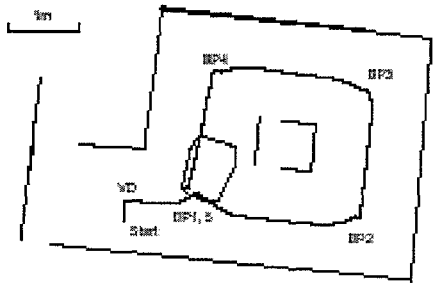


Fig. 2.13: Algoritmo presentado por Victorino et al. [VicRivBor00] que resuelve el problema SLAM utilizando EKF y el diagrama de Voronoi generalizado

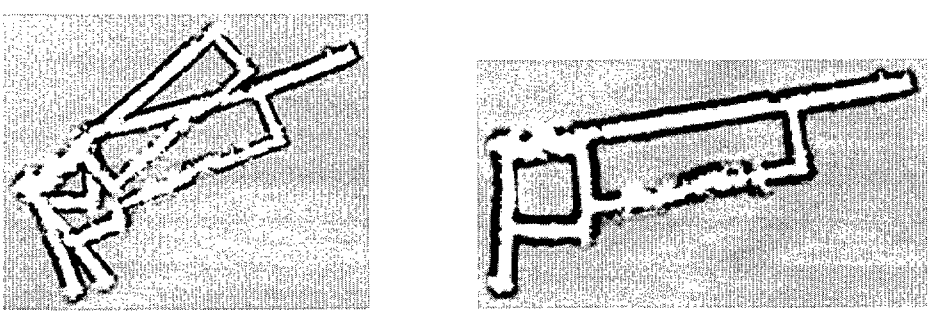
Guivant et al. [GuiNeb01] usan el Filtro Extendido de Kalman para resolver el problema SLAM. El robot comienza en una posición desconocida con una incertidumbre dada y obteniendo unas medidas del entorno relativas a su localización. Esta información es usada para construir incrementalmente y mantener un mapa, y para localizar al robot con respeto a este mapa. El sistema detectará una nueva característica en el comienzo de la misión y cada

vez que el robot explore un nuevo área. Una vez que estas características ya son fiables y estables son incorporadas al mapa formando parte del vector de estado. Cada región tiene una lista de características que son conocidas dentro de sus límites. Cada vez que una nueva marca es detectada en la región en la que se encuentra, pone un índice de definición de marca y la adiciona a la lista. Las marcas que aquí se detectan son árboles utilizando la información proporcionada por un telémetro láser.

Dissanayake et al. [DisNewCla01] utilizan el filtro Extendido de Kalman para resolver el problema de SLAM. Emplean un radar que permite detectar tanto marcas artificiales (reflectores) como marcas naturales.

2.4.1.2 Trabajos basados en EM

En esta sección se describen algunos trabajos que han utilizado el algoritmo EM para resolver el problema SLAM. Thrun et al. [ThrBurFox98a] [ThrFoxBur98] utilizan el método probabilístico EM para construir un mapa de entornos de interiores de grandes dimensiones. Este algoritmo alterna localización y generación del mapa, mejorando tanto la estimación del mapa como la localización del robot. En el paso de localización, el robot se localiza utilizando el mapa anterior, y en el paso de generación de mapa, calcula el mapa más probable utilizando las posiciones estimadas anteriormente. Los mapas que construyen para resolver el problema de SLAM son mapas de ocupación generados a partir de la información proporcionada por los sensores de ultrasonidos.



(a) Mapa del entorno sin utilizar EM (b) Mapa del entorno utilizando EM

Fig. 2.14: Método SLAM presentado por Thrun et al. [ThrBurFox98a]

Burgard et al. [BurFoxJan99] presentan un método para aprender mapas.

Este método usa EM para estimar el mapa más probable usando dos niveles de representación de los mapas durante el algoritmo EM. De esta manera, es capaz de operar utilizando los datos proporcionados por los sensores de ultrasonidos y no requiere marcas predefinidas. Los mapas son aprendidos en dos pasos: 1) paso E donde los mapas locales son aprendidos sobre la consideración que la odometría es localmente correcta y, 2) el paso M es entonces aplicado para integrar estos mapas locales, usando la localización de Markov (paso E) y la representación de mapas en niveles.

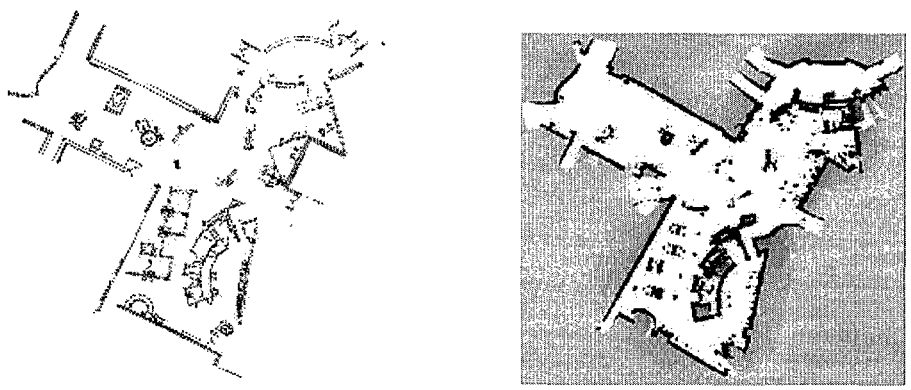
En [ThrBurFox00] se presenta una método para generar on-line un mapa robusto y localización en un entorno de interiores. El método combina la idea de construcción del mapa de manera incremental (máxima probabilidad), con la idea de los métodos no-incrementales que son más potentes computacionalmente (estimación posterior, corrección backward). El resultado es un algoritmo robusto y rápido para la generación de mapas en tiempo real en entornos de interiores. Este método usa el potencial del algoritmo EM usando las estimaciones posteriores y un mecanismo rápido para la revisión hacia atrás (backward).

2.4.1.3 Métodos híbridos

Thrun [Thr01] presenta un método que combina las mejores características de ambos métodos. Propone un algoritmo on-line que construye un mapa a partir de la información de los sensores, sin la necesidad de una asociación de datos exacta. Thrun presenta un método que combina el estimador de máxima probabilidad con un estimador de la posición posterior. Los mapas son generados on-line encontrando el mapa posterior más probable sobre los datos de los sensores recientes. Sea como sea, esta metodología on-line falla cuando se genera un mapa cíclico de grandes dimensiones. De esta manera cuando se detecta un error en la estimación del error suficientemente grande, una corrección hacia atrás (backward) de la estimación de las posiciones posteriores es realizada, utilizando la localización de MonteCarlo.

2.4.2 Métodos no probabilísticos

En la sección anterior se han comentado algunos de los trabajos más importantes relacionados con la construcción de un mapa del entorno y localización (SLAM) de manera probabilística. La ventaja que tienen estos métodos es que trabajan bastante bien en entornos cíclicos pero, sin embargo, son más costosos computacionalmente. Por esta razón, hay autores que



(a) Mapa real del entorno

(b) Mapa construido a partir de la información sensorial

Fig. 2.15: Mapa generado utilizando el algoritmo propuesto en [Thr01]

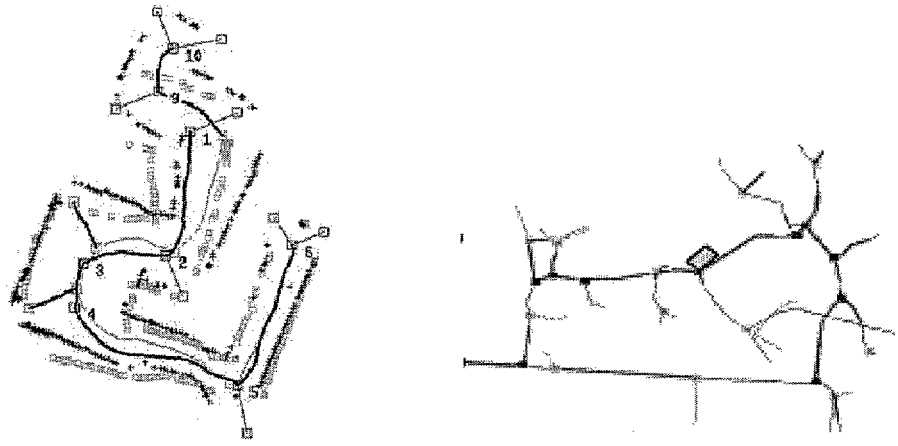
han preferido resolver el problema SLAM utilizando métodos deterministas. A continuación se muestran algunos de los trabajos de SLAM deterministas.

Thrun et al. [ThrBüc96] combinan los mapas basados en celdillas y los mapas topológicos generados a partir de ultrasonidos. Los mapas basados en celdillas son aprendidos usando redes neuronales y la regla de Bayes. El mapa basado en celdillas es dividido en regiones críticas que definen entradas a puertas, pasillos, etc.. A partir de aquí se genera el diagrama de Voronoi para construir luego el mapa topológico. Para localizar al robot utilizan un método de localización basado en el contraste entre el mapa local basado en celdillas y el global, también tiene en cuenta la orientación de las paredes.

Schultz et al. [SchAdaYam99] han desarrollado e integrado técnicas para la exploración autónoma, generación de mapas y localización continúa. Este sistema incluye métodos que adaptan los mapas y que permiten una navegación robusta en entornos dinámicos. Como resultado el robot puede entrar en un entorno desconocido a priori y generar un mapa mientras que se localiza. Utilizan una representación basada en celdillas donde cada celdilla almacena la probabilidad de que esté ocupada. La ventaja de estos tipos de mapas es que permiten fusionar la información de sensores diferentes. Crean dos tipos de representaciones: mapas locales que contienen la información de los sensores obtenida en un instante de tiempo, y los mapas globales que se obtienen por fusión de los mapas locales. El primer tipo de mapa se utiliza para localización ya que contiene poco error odométrico mientras que el

segundo se utiliza para navegación. El proceso de localización se hace contrastando los mapas locales. Con la información del proceso de localización se actualiza el mapa global.

Se et al. [SeLowLit01] presentan un algoritmo para localización y generación del mapa del entorno a partir de la información proporcionada por una Triclops, sistema de visión triocular. Las características 3D son localizadas y la posición del robot es estimada realizando un contraste entre imágenes sucesivas. El algoritmo lleva a cabo el SLAM realizando un seguimiento de las características SIFT (Scale Invariant Feature Transform). La estimación de la posición 3D de las marcas puede ser obtenida realizando un contraste de las 3 imágenes obtenidas por la cámara, de esta manera, se puede generar un mapa 3D y el robot se puede localizar. El mapa 3D, representado como una base de datos de marcas SIFT, es incrementalmente actualizado en el tiempo y adaptado a las variaciones que se producen en el entorno.



(a) Mapa construido por Choset et al. [ChoNag01]

(b) Mapa construido por Zwynsvoorde et al. [ZwySimAla00]

Fig. 2.16: Ejemplos de algoritmos que resuelven el problema SLAM utilizando el diagrama de Voronoi

Trabajos próximos al algoritmo desarrollado en esta tesis son los presentados por Choset et al. [ChoNag01] y Zwynsvoorde et al. [ZwySimAla00]. Ambos métodos abordan el problema de SLAM construyendo un diagrama de Voronoi, utilizando sus características tanto topológicas como geométricas para localizar el robot. Sin embargo, estos métodos fallan en entornos cíclicos de grandes dimensiones. El algoritmo desarrollado en esta tesis es

capaz de localizar al robot en entornos de grandes dimensiones con ciclos, introduciendo en el proceso de localización un paso de estimación posterior de las posiciones del robot. A diferencia del método presentado en [ChoNag01], el algoritmo propuesto no construye un punto único del diagrama de Voronoi cada vez que el robot toma una observación del entorno, lo que permite al robot planificar acciones futuras como son seguir recto, girar a la derecha, etc.. Por otra parte, el algoritmo presentado en esta tesis genera el diagrama de Voronoi directamente a partir de las medidas del sensor, en este caso un telémetro láser, reduciendo el tiempo de cálculo a diferencia del algoritmo presentado por Zwynsvoorde et al. [ZwySimAla00], que construye el diagrama de Voronoi a partir de segmentos obtenidos de las medidas del sensor.

El método propuesto en esta tesis construye un mapa topo-geométrico del entorno a partir de la información obtenida del diagrama de Voronoi. Este mapa puede ser utilizado para planificación, para localización y para navegación.

3. APLICACIÓN DEL DIAGRAMA DE VORONOI EN ROBÓTICA

3.1 Introducción

El objetivo de la planificación de trayectorias (path planning) [Lat91] es generar automáticamente un camino para que el robot se desplace desde un punto origen, $\mathcal{C}_{init}$, a un punto destino, $\mathcal{C}_{goal}$, sin colisionar con ningún objeto presente en el entorno $\mathcal{W}$ (ver figura 3.1). Los objetos en movimiento pueden tener una gran variedad de formas y tamaños, complicando el desarrollo de soluciones generalizadas para resolver el problema de la planificación de trayectorias. El espacio de configuración $\mathcal{C}$ es el conjunto de todas las posibles configuraciones de un objeto y está formado por la transformación del problema del espacio-objeto en el de un punto en un espacio dimensional apropiado, $\mathcal{C}$, del objeto. Una configuración del objeto es un conjunto de parámetros independientes que especifican completamente la posición de cada punto del objeto. El objeto en movimiento es de esta manera representado como un punto en $\mathcal{C}$ mientras que los obstáculos y los límites del espacio físico son transformados en regiones prohibidas llamadas obstáculos en $\mathcal{C}$.

A diferencia de la planificación clásica, la planificación de movimientos basada en sensores incorpora la información sensorial, reflejando el estado actual del entorno, en el proceso de planificación del robot. La planificación clásica considera un conocimiento completo a priori de la geometría del entorno para llevar a cabo la planificación.

Según Choset [Cho96] el problema de la planificación basada en sensores es encontrar el *camino libre* (libre de colisiones) para un robot móvil usando la información proporcionada por los sensores. El problema de generar automáticamente un mapa del entorno proporciona una solución al problema de planificación.

La planificación basada en los sensores es importante porque tiene en cuenta las consideraciones reales de los robots:

- El robot frecuentemente no tiene un conocimiento a priori del entorno.

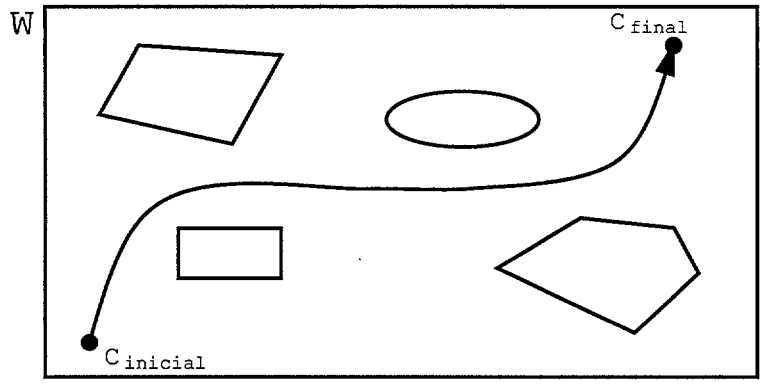


Fig. 3.1: Posible trayectoria desde una posición inicial $C_{inicial}$ a una posición destino C_{final} evitando obstáculos

- El robot puede tener un conocimiento limitado del entorno debido a requerimientos de memoria.
- El modelo del entorno es definido para contener imprecisiones que puede complicar o hacer fallar el proceso de planificación.
- El entorno está sujeto a situaciones no esperadas o a cambios bruscos.

Los dos problemas principales en planificación basada en sensores en entornos estáticos son:

1. Cuando a un robot se le da un punto destino, se supone que siempre conoce su posición actual (lleva a bordo un sistema inercial, *dead reckoning*), y se considera que no tiene un conocimiento del entorno a priori, el robot debe encontrar el camino libre de colisiones hasta el punto destino utilizando la información proporcionada por los sensores.
2. El robot se sitúa en un punto del entorno sin un conocimiento a priori del entorno. Usando solamente los sensores que lleva a bordo y el sistema inercial, el robot debe construir un mapa de caminos (roadmap) completo del entorno para posteriormente poder planificar el camino a seguir.

Se han desarrollado una gran variedad de métodos para resolver el problema de planificación de trayectorias. Estos métodos se pueden clasificar en locales y globales.

Los métodos globales usan toda la información del entorno para desarrollar una representación concisa de la conectividad del espacio libre, que es procesada posteriormente para encontrar un camino global óptimo. Los métodos globales pueden ser clasificados en *exactos y heurísticos.* Los métodos globales exactos usan modelos algebraicos de los obstáculos del espacio permitiendo extraer soluciones exactas para el problema de planificación de trayectorias, si es que la solución existe. Estos algoritmos generalmente tienen una gran complejidad y, por tanto, un alto coste computacional. Los métodos heurísticos tienden a reducir la complejidad en una variedad de caminos, muchos de ellos debiéndose a restricciones de posición de las curvas de los obstáculos. Los grafos de visibilidad, los métodos de retracción, los métodos de mapas de caminos son algunos métodos globales heurísticos. Los métodos globales envuelven la construcción de grafos de conectividad.

Un método local es el método de campo de potencial [KasKum99] porque la selección de configuraciones posteriores en el camino son basadas en los potenciales de las configuraciones vecinas de $\mathcal{C}$.

3.2 Mapas de caminos

Muchos autores han utilizado un mapa de caminos (roadmap) [BooOveSta99] [VicRivBor00] para resolver el problema de planificación de trayectorias. Un mapa de caminos es un conjunto de curvas unidimensionales que capturan la topología y las propiedades geométricas del entorno del robot. Los mapas de caminos tienen las propiedades siguientes:

Accesibilidad El planificador puede construir un camino entre dos puntos conectando los componentes del espacio libre del robot, encontrando un camino libre de colisiones sobre el mapa.

Alcanzabilidad Se puede construir un camino libre de colisiones desde un punto cualquiera del mapa hasta el punto destino.

Conectividad El robot puede ir desde un punto origen a un punto destino a través de los puntos que definen el mapa.

Un tipo de mapa de caminos es el diagrama de Voronoi [OkaBooSug92]. Numerosos autores han utilizado el diagrama de Voronoi para construir una representación topológica del entorno empleando distintos tipos de sensores: ultrasonidos [ChoBur95], láser [ZwySimAla01] y cámaras [KanHayLi99].

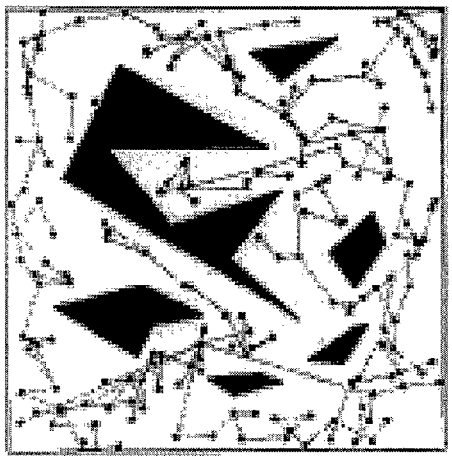


Fig. 3.2: PRM generado por Boor et al. [BooOveSta99]

3.3 Teoría sobre los grafos de Voronoi

El concepto de diagrama de Voronoi es muy simple: dado un conjunto finito de puntos en un espacio continuo, se asocian todas las localizaciones del espacio con el miembro más cercano del conjunto de puntos. El resultado es un espacio dividido en un conjunto de regiones.

El diagrama de Voronoi es utilizado en una gran variedad de disciplinas:

- Astronomía para estudiar la estructura del Universo.
- Arqueología para identificar las partes de una región que han tenido una influencia de civilizaciones distintas.
- Meteorología para estimar las precipitaciones cuando ha ocurrido un fallo en el sensor.
- Física para estudiar el comportamiento de un compuesto.
- Robótica para planificar el camino que ha de seguir un robot móvil autónomo.
- etc.

La ventaja principal del diagrama de Voronoi es que existen una gran variedad de generalizaciones: considerando regiones con subconjuntos de

puntos mejor que con un único punto, incluyendo obstáculos, considerando regiones con características geométricas, etc..

En las secciones siguientes se definirá diagrama de Voronoi y se mostrarán algunas generalizaciones.

3.3.1 Definiciones de un diagrama de Voronoi ordinario

Se considera un conjunto de puntos en un plano Euclídeo (ver figura 3.3). El número de puntos puede ser dos o más, pero un número finito y todos ellos distintos en el sentido de que no coinciden en el plano. Dado este conjunto de puntos, se asigna cada posición en el plano al miembro más cercano en el conjunto de puntos. Si una posición es igual a dos o más miembros del conjunto de puntos, se asigna esta posición a estos miembros. Como resultado, el conjunto de posiciones asignadas a cada miembro en el conjunto de puntos forma su propia región (región sombreada de la figura 3.3). Las regiones resultantes son agrupadas exhaustivamente en el plano porque cada localización es asignada al menos a un miembro en el conjunto de puntos. El conjunto de localizaciones asignadas a dos o más miembros en el conjunto de puntos forman los límites de las regiones (las líneas en la figura 3.3). Las regiones adyacentes sólo se solapan en sus límites. De este modo el conjunto de regiones son agrupadas exhaustivamente y son excluyentes mutuamente excepto en los límites. El conjunto de estas regiones forma un *mosaico.* Se denominará a este mosaico *diagrama plano de Voronoi ordinario* y a las regiones que se forman *polígonos de Voronoi ordinarios.*

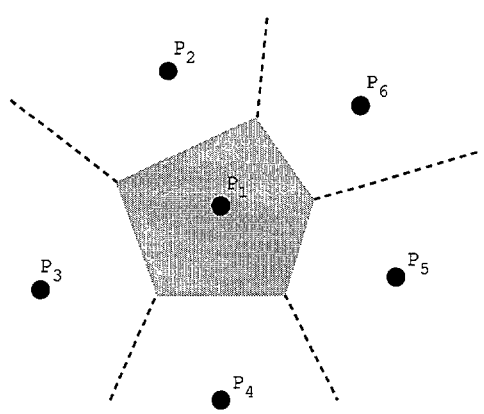


Fig. 3.3: Diagrama de Voronoi ordinario

Definición 3.3.1: **Diagrama de Voronoi ordinario** Dado un conjunto de dos o más puntos pero un número finito de puntos distintos en el plano Euclídeo, se asocia cada posición en este espacio con el miembro/s más cercano del conjunto de puntos con respecto a la distancia Euclídea. El resultado es un mosaico en el que cada región está asociado con miembros del conjunto de puntos. A este mosaico se le denomina *diagrama de Voronoi ordinario.*

Se considera un número finito, n, de puntos en el plano Euclídeo, tal que, $2 \leq n \leq \infty$. Los n puntos se denominan $p_1, \cdots, p_n$ con coordenadas $(x_{11}, x_{12}), \cdots, (x_{n1}, x_{n2})$ o vectores de posición $x_1, \cdots, x_n$. Los n puntos son distintos en el sentido que $x_i \neq x_j$ para $i \neq j, i, j \in I_n = \{1, \cdots, n\}$. Sea p un punto arbitrario en el plano Euclídeo con coordenadas (x_1, x_2) o vector de localización x. La distancia Euclídea desde p a p_i es dada por $d(p, p_i) = \|x - x_i\| = \sqrt{[(x_1 - x_{i1})^2 + (x_2 - x_{i2})^2]}$. Si p_i es el punto más cercano a p, o p_i es uno de los puntos más cercanos a p, se tiene la relación $\|x - x_i\| \leq \|x - x_j\|$ para $j \neq i, j \in I_n$. En este caso p se asigna a p_i.

Definición 3.3.2: **Diagrama de Voronoi ordinario** Se supone $P = \{p_1, \cdots, p_n\}$, donde $2 \leq \infty$ y $x_i \neq x_j$ para $i \neq j, i, j \in I_n$. Se llamará a la región dada:

$$V(p_i) = \{x \mid \|x - x_i\| \leq \|x - x_j\| \; para \; j \neq i, j \in I_n\} \tag{3.1}$$

el *polígono de Voronoi (ordinario)* asociado a p_i (o polígono de Voronoi de p_i), y al conjunto dado por:

$$\mathcal{V} = \{V(p_1), \cdots, V(p_n)\} \tag{3.2}$$

el *diagrama de Voronoi (ordinario)* generado por P (o diagrama de Voronoi de p_i)

3.3.2 Diagrama de Voronoi de series puntos

Sea $G = \{g_1, g_2, \ldots, g_n\}$ una colección de n series de puntos en el plano que no se solapan $g_i \subset \Re^2, i = 1, 2, \ldots, n$ y $g_i \cap g_j = 0, i \neq j$. Para cualquier punto $p \in \Re^2$ se denomina $d_E(p, g_i)$ a la mínima distancia euclídea desde p a cualquier punto de g_i. La región de Voronoi se define como:

$$V(g_i) = \left\{p | p \in \Re^2, d_E(p, g_i) \leq d_E(p, g_j), j \neq i\right\} \tag{3.3}$$

Y la serie dada por:

$$V = \{V(g_1), \ldots, V(g_n)\} \tag{3.4}$$

el *diagrama de Voronoi generalizado* generado por G. Se utiliza la denominación de diagrama de Voronoi generalizado cuando los elementos generadores son series de puntos en lugar de puntos aislados.

3.3.3 Grafos de Voronoi para líneas

Se considera un conjunto $L = \{L_1, \ldots, L_n\} \subset \mathbb{R}^2 (1 \leq n \leq \infty)$, donde L_i es un punto, un segmento de línea (que puede ser curva) o un elemento geométrico formado por un conjunto de segmentos de líneas conectados. Se supone que los elementos en L no están conectados, es decir, $L_i \bigcap L_j = \emptyset, i \neq j, i, j \in I_n$. Sobre estas consideraciones, se define la distancia de p a L_i por la distancia más corta entre p y un punto p_i sobre L_i, es decir:

$$d_s(p, L_i) = \min_{x_i} \{ \| x - x_i \| \mid x_i \in L_i \} \tag{3.5}$$

donde x y x_i son los vectores de localización de p y p_i respectivamente. Con esta distancia se define

$$V(L_i) = \{ p \mid d_s(p, L_i) \leq d_s(p, L_j), j \neq i, j \in I_n \} \tag{3.6}$$

Alternativamente, se considera

$$\begin{array}{rcl} Dom(L_i, L_j) & = & \{ p \mid d_s(p, L_i) \leq d_s(p, L_j) \}, \; j \neq i, \\ b(L_i, L_j) & = & \{ p \mid d_s(p, L_i) = d_s(p, L_j) \}, \; j \neq i. \end{array} \tag{3.7}$$

Entonces el conjunto $V(L_i)$ se escribe como

$$V(L_i) = \bigcap_{j \in I_n \setminus \{i\}} Dom(L_i, L_j) \tag{3.8}$$

Como $Dom(L_i, L_j) \cup Dom(L_j, L_i) = \mathbb{R}^2$, y el conjunto $b(L_i, L_j)$ es el límite de $Dom(L_i, L_j)$, el dominio de la region $Dom(L_i, L_j)$ tiene un buen comportamiento y es regular. De este modo el conjunto resultante $\mathbb{V}(L, D_s, \mathbb{R}^2) = \mathbb{V}(L) = \{V(L_1), \ldots, V(L_m)\}$ proporciona un diagrama generalizado de Voronoi. Se llamará a este diagrama generalizado de Voronoi el *diagrama de Voronoi de líneas*, y el conjunto $V(L_i)$ la *región de Voronoi de líneas* asociado con L_i. Si L_i degenera en un punto para todo i, el diagrama de Voronoi de líneas se reduce al diagrama de Voronoi ordinario. De este modo el diagrama de Voronoi de líneas es una generalización del diagrama de Voronoi ordinario.

Se llama a P_i de $V(p_i)$ el punto generador o generador del i-ésimo polígono de Voronoi, y el conjunto $P = \{p_1, \ldots, p_n\}$ el *conjunto generador* del diagrama de Voronoi $\mathcal{V}$.

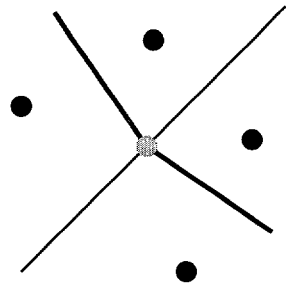


Fig. 3.4: Diagrama de Voronoi degenerativo

Los límites de un polígono de Voronoi pueden ser un segmento de línea, medias líneas o líneas infinitas, a los que se denomina *bordes de Voronoi.* Se denota a estos bordes por e_i. El punto final de un borde de Voronoi se llama *vértice de Voronoi.* Alternativamente se puede definir vértice de Voronoi como un punto de división de 3 o mas polígonos de Voronoi. Se denota a un vértice de Voronoi por q_i. Cuando existe al menos un vértice de Voronoi en el que 4 o más bordes de Voronoi se encuentran en el diagrama de Voronoi $\mathcal{V}$, se dice que $\mathcal{V}$ es degenerativo (grafo de la figura 3.4); en caso contrario se dice que no es degenerativo (grafo de la figura 3.3).

3.3.4 Diagramas de Voronoi para un conjunto de puntos y segmentos lineales

Se considera que un generador L_i, es un punto o un segmento lineal o una cadena de segmentos lineales. También se considera que un segmento lineal L_i contiene dos puntos extremos. Sobre estas consideraciones la distancia mínima de la ecuación (3.5) es:

$$d_s(p, L_i) = \begin{cases} \|x - x_{i1}\| & si\ p \in \mathbb{R}_{i1}, \\ \|x - x_{i2}\|, & si\ p \in \mathbb{R}_{i2} \\ \left\| \|(x - x_{i1})\| - \frac{(x-x_{i1})^T(x_{i2}-x_{i1})}{\|(x_{i2}-x_{i1})^2\|}(x_{i2} - x_{i1}) \right| & si\ p \in \mathbb{R}_{i3} = \mathbb{R}^2 \backslash [R_{i1} \bigcup R_{i2}] \end{cases} \tag{3.9}$$

donde x_{i1} y x_{i2} son los puntos finales de L_i y $R_{i1} = \{x \backslash (x_{i2} - x_{i1})^T (x - x_{i1}) < 0\}, R_{i2} = \{x \backslash (x_{i1} - x_{i2})^T (x - x_{i2}) < 0\}$ (ver figura 3.5).

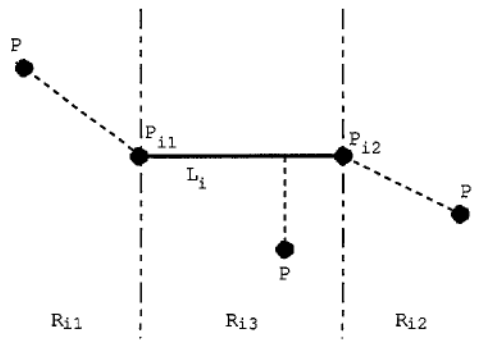


Fig. 3.5: Distancia entre un punto y un segmento recto

3.3.5 Diagramas de Voronoi para un conjunto de puntos, segmentos lineales rectos y arcos circulares

Para facilitar los cálculos computacionales se considera que un arco circular es menor o igual a un semicírculo. Cuando el arco circular es mayor que un semicírculo se consideran dos arcos circulares conectados en un punto, y un círculo completo como dos semicírculos conectados en dos puntos. Para un arco circular con radio r_i centrado en x_{ci}, la distancia menor de la ecuación (3.5) se escribe como:

$$d_s(p, L_i) = \begin{cases} \|x - x_{i1}\| & si\ x \in \mathbb{R}_{i1}, \\ \|x - x_{i2}\|, & si\ x \in \mathbb{R}_{i2} \\ |\|(x - x_{ci})\| - r_i| & si\ x \in \mathbb{R}_{i3} = \mathbb{R}^2 \backslash [R_{i1} \bigcup R_{i2}] \end{cases} \tag{3.10}$$

donde x, x_{i1} y x_{i2} son los vectores de posición de p y el punto final de L_i respectivamente, y $\mathbb{R}_{i1}$, $\mathbb{R}_{i2}$ y $\mathbb{R}_{i3}$ son las regiones definidas en la figura 3.6.

3.3.6 Diagramas de Voronoi para un conjunto de círculos

La distancia menor de la ecuación (3.5) se escribe como:

$$d_{c1}(p, L_i) = |\|x - x_{ci}\| - r_i|, \tag{3.11}$$

donde L_i es el círculo centrado en p_{ci} con radio r_i (ver figura 3.7).

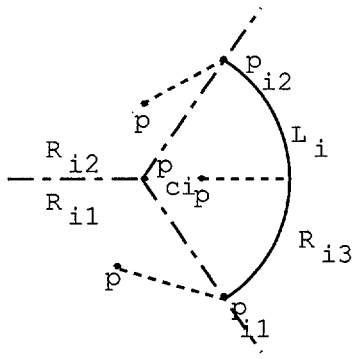


Fig. 3.6: Distancia desde un punto a un arco circular

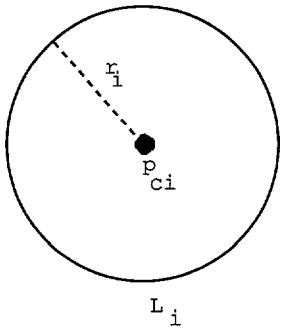


Fig. 3.7: Círculo centrado en p_{ci} con radio r_i

3.3.7 Eje medio

Un eje medio puede ser definido para una figura C geométrica conectada (regiones no conectadas pueden ser consideradas por separado). El *eje medio* de una figura geométrica es un conjunto de puntos $x \in C$ para el que existe $x_i, x_j \in \partial C$ para todo $x_i \neq x_j$ y $\|x - x_i\| = \|x - x_j\| = min\{\|x - y\| | y \in \partial C\}$, esto es, el eje medio de C, denominado M(C), es dado por:

$$M(C) = \{x \mid \|x - x_i\| = \|x - x_j\| = \min_{y} \{\|x - y\| | y \in \partial C\}, x_i \neq x_j, x_i, x_j \in \partial C, x \in C\} \tag{3.12}$$

Un ejemplo se muestra en la figura 3.8.
Se puede obtener un eje medio usando el diagrama de Voronoi lineal. Se

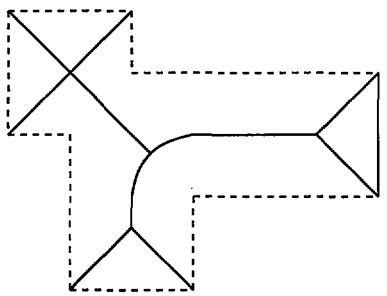


Fig. 3.8: Un eje medio

supone una figura geométrica C representada por un polígono cuyos límites son definidos por una cadena de segmentos rectos L (línea de puntos en la figura 3.8). Se descompone L en $L^{(d)}$, para construir a continuación el diagrama de Voronoi lineal $V(L^{(d)})$. Considerando la propiedad de que los puntos sobre las ramas de las regiones de Voronoi lineales son equidistantes desde las líneas generadoras (los límites de la figura geométrica), entonces el eje medio está incluido en las ramas de $V(L^{(d)})$. Borrando las ramas no necesarias en $V(L^{(d)})$, se obtiene el eje medio de C (ver figura 3.8)

3.4 Trabajos relacionados

El problema básico de planificación de trayectorias implica encontrar el camino libre de obstáculos para un robot desde su posición inicial a su posición final con la presencia de obstáculos. Un método para resolver este problema es usar el diagrama de Voronoi. Como ya se ha indicado en secciones anteriores, el diagrama de Voronoi es un conjunto de objetos definidos por puntos en el espacio que son equidistantes a dos o más objetos del entorno. El diagrama divide el espacio es varias regiones, llamadas *regiones de Voronoi.* La región de Voronoi de un objeto A corresponde a la parte del espacio formados por los puntos que son equidistantes a A y a otro/s objeto/s.

Los métodos tradicionales para la planificación de trayectorias usando diagramas de Voronoi, reducen al robot a un punto y expanden apropiadamente todos los obstáculos. En [SudNanSri99] se considera que un modelo de los obstáculos y del robot son disponibles a priori.

El diagrama de Voronoi no es sólo usado en el campo de la robótica para navegación o planificación de caminos [ThrBüc96] [KanHayLi99] [KeeOngHua99] [LeeChoRiz01] [CasMorSal01], sino que también ha sido y es

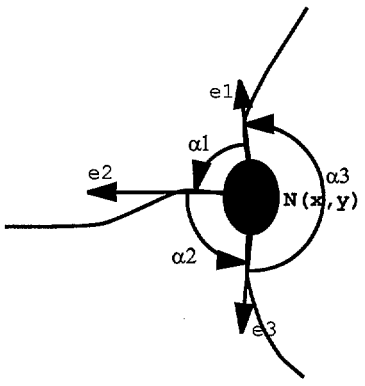


Fig. 3.9: Características geométricas que se utilizan en [ZwySimAla00] para estimar la posición del robot

empleado para localización, es decir, para estimar la posición del robot en el entorno por donde se desplaza. Piaggio et al. [PiaZac98] identifican ciertos lugares característicos, como son puntos de giro, estrechamientos y centros de habitaciones, a partir de la información proporcionada por el diagrama de Voronoi que luego emplean para estimar la posición del robot. Zwynsvoorde et al. [ZwySimAla00] utilizan tanto las características topológicas como geométricas (posición de los nodos y orientación de las ramas) del diagrama de Voronoi para localizar al robot. Choset [Cho01] localiza el robot en el entorno por donde se desplaza utilizando los puntos de encuentro o meet points, que son los vértices de Voronoi (ver sección 3.3.3). Blanco et al. [BlaBoaMor01] contrastan el diagrama de Voronoi construido a partir de la información de un telémetro láser con el diagrama de Voronoi que tiene almacenado el robot del entorno.

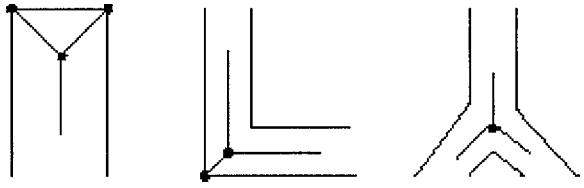


Fig. 3.10: Diferentes tipos de meet points que utiliza Choset [Cho01] para estimar la posición del robot

Numerosos algoritmos robustos, que generan el diagrama de Voronoi ordi-

nario, se han aplicado para generar diferentes tipos de diagramas de Voronoi generalizados.

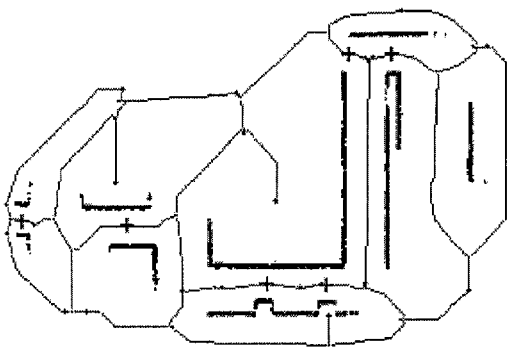


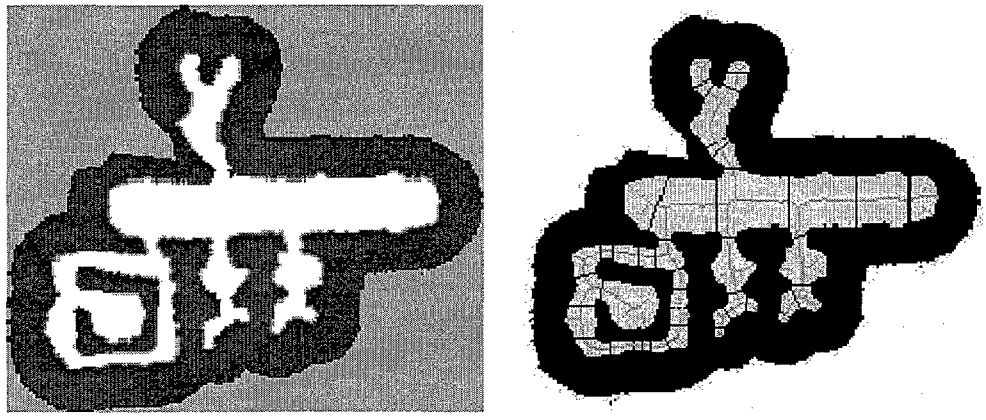
Fig. 3.11: Diagrama de Voronoi construido por el algoritmo de Piaggio et al. [PiaZac98]

Choset et al. [ChoBur95] construyen un grafo generalizado de Voronoi de manera incremental a partir de la información proporcionada por los sensores de ultrasonidos. En cada disparo de los sensores se obtiene un único punto del diagrama de Voronoi.

Piaggio et al. [PiaZac98] combinan el uso de representaciones basadas en celdillas con un método de extracción del diagrama de Voronoi para obtener una descripción topológica del entorno. Es decir, primero construyen un mapa de probabilidad y a partir de él construyen el diagrama de Voronoi. Este procedimiento de obtención del diagrama de Voronoi también es utlizado por Thrun [Thr98b]. Al igual que el algoritmo de construcción del diagrama de Voronoi propuesto en esta tesis, Piaggio et al. [PiaZac98] no consideran las zonas libres que no pueden ser atravesadas por el robot para generar el diagrama de Voronoi.

Kanbara et al. [KanHayLi99] construyen un diagrama de Voronoi a partir de la información proporcionada por una cámara. De la imagen se extraen los objetos que son modelados como pilares poligonales.

Zwynsvoorde et al. [ZwySimAla00] construyen un diagrama de Voronoi a partir de la información proporcionada por un telémetro láser generando un polígono de visibilidad y los segmentos artificiales (líneas de escape) obtenidos a partir de los datos del sensor. El inconveniente de este algoritmo es que necesita tratar previamente los datos del telémetro láser para obtener segmentos y generar a partir de ellos el diagrama de Voronoi.



(a) Mapa basado en celdillas (b) Diagrama de Voronoi

Fig. 3.12: Construcción del diagrama de Voronoi a partir del mapa basado en celdillas mediante el algoritmo propuesto por Thrun [Thr98b]

3.5 Algoritmos de construcción del diagrama de Voronoi

En esta sección se describen algunos algoritmos para la construcción del diagrama de Voronoi.

3.5.1 Algoritmo de construcción del diagrama de Voronoi para una imagen aproximada por un conjunto de puntos

Sugihara [Sug93] presenta un algoritmo para la construcción de un diagrama de Voronoi generalizado para figuras.

Se supone que $G = \{g_1, g_2, \cdots, g_n\}$ es una colección de n conjuntos de puntos cercanos que no se solapan en el plano: $g_i \subset \mathbb{R}^2$ para $i = 1, 2, \cdots, n$ y $g_i \bigcap g_j = \emptyset$ para $i \neq j$. Para cada punto $p \in \mathbb{R}^2$, se denota a $d_E(p, g_i)$ como la distancia Euclídea mínima desde p hasta el punto en g_i, y se define:

$$V_R = \{p | p \in \mathbb{R}^2, d_E(p, g_i) < d_E(p, g_j) \text{ para todo } j \neq i\} \tag{3.13}$$

La colección $\mathcal{V}(G, d_E) = \{V_E(g_1), V_E(g_2), \cdots, V_E(g_n)\}$ pertenece a $\mathbb{R}^2$. Se llama $\mathcal{V}(G, d_E)$ al *diagrama de Voronoi generado* por G con respecto a

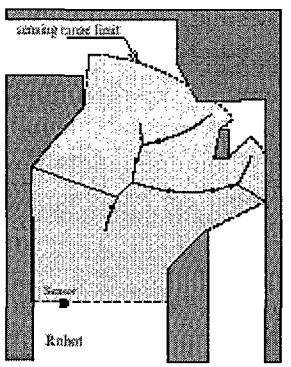


Fig. 3.13: Grafo de Voronoi local construido por el algoritmo de Zwynsvoorde et al. [ZwySimAla00]

la distancia Euclídea d_E o el *diagrama de Voronoi Euclídeo generado* por G. Los elementos de G son denominados *generadores* y $V_E(g_i)$ se llama *región de Voronoi* para g_i. Las ramas de Voronoi y los puntos de Voronoi son definidos de la misma manera que para el diagrama de Voronoi ordinario.

El diagrama de Voronoi para G puede ser construido aproximadamente reemplazando primeramente cada generador por un conjunto de puntos, para luego construir el diagrama de Voronoi de estos puntos, eliminando finalmente las ramas y puntos del Voronoi superfluos. El algoritmo es el siguiente:

Entrada: Conjunto $G = \{g_1, g_2, \cdots, g_n\}$ de n figuras que no se solapan.

Salida: Aproximación del diagrama de Voronoi $\mathcal{V}(G, d_E)$

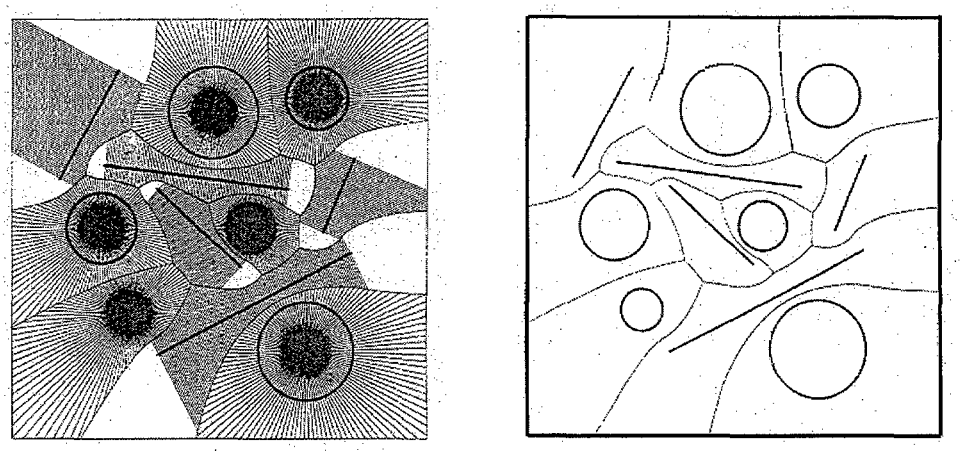
Procedimiento:

Paso 1: Para cada $i = 1, 2, \cdots, n$ crear un conjunto finito P_i de puntos que aproximan a g_i.

Paso 2: Construir el diagrama de Voronoi Ordinario $\mathcal{V}$ generado por un conjunto de puntos $P_1 \bigcup P_2 \bigcup \cdots \bigcup p_n$.

Paso 3: Eliminar de $\mathcal{V}$ aquellas ramas para las que dos puntos generados pertenecen a la misma figura. Borrar los puntos de Voronoi aislados si lo hubiese de $\mathcal{V}$.

Paso 4: Devolver $\mathcal{V}$.



(a) Diagrama de Voronoi ordinario

(b) Aproximación del diagrama de Voronoi generado para segmentos y círculos

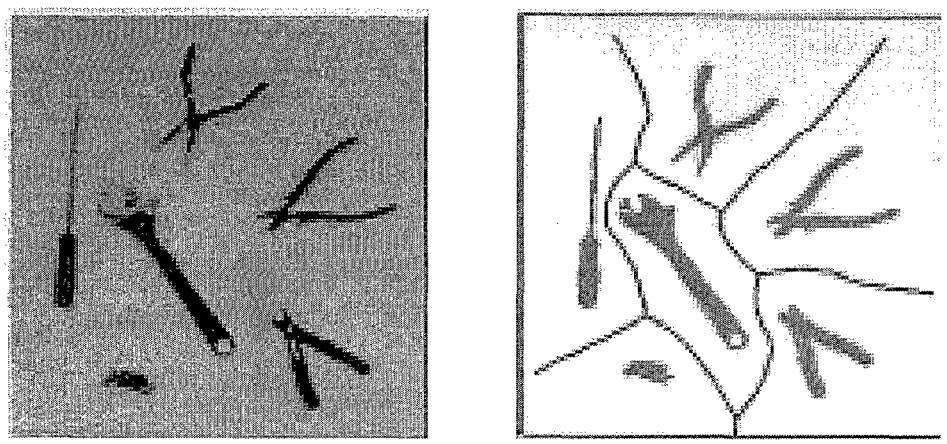
Fig. 3.14: Construcción del diagrama de Voronoi mediante el algoritmo propuesto por Sugihara [Sug93]

En el algoritmo propuesto en esta tesis para la construcción del diagrama de Voronoi, a diferencia de este algoritmo, no se generan ramas que pertenecen al mismo objeto, ya que los datos que pertenecen al mismo objeto son agrupados previamente.

3.5.2 Algoritmo de construcción del diagrama de Voronoi para una imagen utilizando un mapa basado en celdillas

Sudha et al. [SudNanSri99] presentan un algoritmo para la construcción del diagrama de Voronoi basado en la distancia Euclídea. Obtienen una imagen del espacio de trabajo usando una cámara digital, representándola como un mapa de celdillas 2D y binarizando la imagen.

Se construye el diagrama de Voronoi de los obstáculos de la imagen binarizada formada por obstáculos (O) y del libre espacio (F). Cada obstáculo es considerado como un componente conectado. La construcción envuelve la dilatación de cada componente conectado de la imagen binarizada uniformemente y manteniendo la conectividad entre los pixels vecinos que pertenecen



(a) Imagen real (b) Diagrama de Voronoi

Fig. 3.15: Construcción del Diagrama de Voronoi mediante el algoritmo propuesto por Sudha et al. [SudNanSri99]

	y-1	**y**	**y+1**
x-1	p_{NW}	p_N	p_{NE}
x	p_W	p_C	p_E
x+1	p_{SW}	p_S	p_{SE}

Tab. 3.1: Vecindad del pixel

al mismo componente conectado. El límite de cada obstáculo es dilatado hasta que los pixeles límite encuentran el límite de otro obstáculo dilatado. El proceso de dilatación se para cuando la imagen total es ocupada por el objeto dilatado, llamándose *dilatación total.* La unión de los límites de los objetos dilatados sobre la dilatación total proporciona el diagrama de Voronoi discretizado.

Se denomina p_c al pixel de la fila x y columna y de la imagen. La vecindad de p_c, $N_{vd}(p_c)$, se representa en la tabla 3.1.

Se denomina:

$d_e(p_c)$: a la distancia Euclídea entre el pixel p_c y el obstáculo más cercano a p_c:

$$d_e(p_c) = \sqrt{\triangle x(p_c)^2 + \triangle y(p_c)^2} \tag{3.14}$$

Donde:

$$\Delta x_i = \begin{cases} \Delta x^{(k)}(p_i) & i = C, E, W \\ \Delta x^{(k)}(p_i) + 1 & i = N, S, NW, NE, SW, SE \end{cases}$$
$$\Delta y_i = \begin{cases} \Delta y^{(k)}(p_i) & i = C, N, S \\ \Delta y^{(k)}(p_i) + 1 & i = W, E, NW, NE, SW, SE \end{cases} \quad (3.15)$$

$d(p_c)$: a la aproximación entera de $d_e(p_c)$ que satisface la siguiente desigualdad $d(p_c) - 0.5 < d_e(p_c) \leq d(p_c) + 0.5$.

$D(p_c)$: al cuadrado de la distancia euclídea $d_e(p_c)$.

$(\triangle x(p_c), \triangle y(p_c))$: a las dos componentes del vector desplazamiento que son más cercanos al pixel del obstáculo.

$b(p_c)$: al valor binario del pixel p_c.

El algoritmo es:

Entrada: Una imagen binaria, formada por pixeles que forman parte de objetos (O) y del espacio libre (E).

Salida: El diagrama de Voronoi.

Paso 1: Inicialización. $\triangle x$, $\triangle y$ y d_f son inicializadas a 0 y el valor b de todos los pixeles son inicializados de acuerdo a la imagen binaria:

$$b(p_c) = \begin{cases} 1 & p_c \in O \\ 0 & p_c \in F \end{cases} \quad (3.16)$$

Los flags de conexión son inicializados como siguen: $N(p_c) = b(p_N), \cdots,$ $NE(P_c) = b(p_{NE})$.

Paso 2: Proceso iterativo de dilatación. En cada iteración $k \geq 0$, se calcula el vector desplazamiento $(\triangle x(p_c), \triangle y(p_c))$ y d_f de los pixeles cuya distancia Euclídea aproximada a un entero es igual a k.

$$d_f^{(k+1)}(p_c) = 2(k+1) + \max_{p_i \in N_{vd}(p_c)} [d_f^{(k)}(p_i) - (\Delta X_i + \Delta Y_i)] \quad (3.17)$$

Donde:

$$\Delta X_i = \begin{cases} 0 & i = C, E, W \\ 2\Delta x^{(k)}(p_i) + 1 & i = N, S, NW, NE, SW, SE \end{cases}$$
$$\Delta Y_i = \begin{cases} 0 & i = C, N, S \\ 2\Delta y^{(k)}(p_i) + 1 & i = W, E, NW, NE, SW, SE \end{cases} \quad (3.18)$$

Paso 3: Identificación de los pixels de Voronoi. Se supone que $vor(p_c) = 1$ entonces p_c pertenece al diagrama de Voronoi.

$vor(p_c) = 1$ si $\neg N(p_c) \bigsqcup \neg S(p_c) \bigsqcup \neg E(p_c) \bigsqcup \neg W(p_c)$.

Tal que $W(P_c) = 1$ si $E(p_W) \vee [N(p_C) \wedge SW(p_N)] \vee [NW(p_C) \wedge S(p_{NW})] \vee [S(p_C) \wedge NW(p_S)] \vee [SW(p_C) \wedge N(p_{SW})]$ (igual para p_E, p_N y p_S), y $NW(p_c) = 1$ si $SE(p_{NW}) \vee [N(p_C) \wedge W(p_N)] \vee [W(p_C) \wedge N(p_W)]$ (igual para p_{NE}, p_{SW} y p_{SE}).

3.5.3 Algoritmo de construcción del diagrama de Voronoi a partir de los puntos proporcionados por un telémetro láser

Mahkovic et al. [MahSli98] presentan un método para la construcción local de un estructura unidimensional similar al diagrama de Voronoi generalizado. Llaman a esta estructura GLVD (Generalized Local Voronoi Diagram). El GLVD es construido a partir de la información de un telémetro láser.

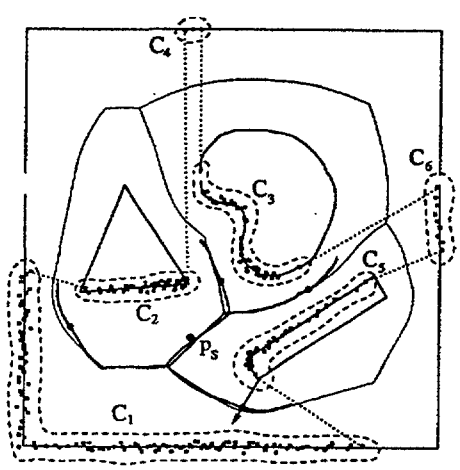


Fig. 3.16: Construcción del GLVD según el algoritmo presentado por Mahkovic et al. [MahSli98]

El sensor proporciona un conjunto de puntos, $P_s = p_m$, donde p_m es cada una de las medidas proporcionadas por el sensor. P_s representa de manera aproximada los límites de la región visible. Se considera una función, Clusterize(p_i) : $P_s \Longrightarrow C$ donde $C = \{C_k\}, k = 1, 2, \cdots, l$ es un conjunto

de colecciones de puntos del sensor tal que cada colección está formada por puntos que pertenece al mismo obstáculo. El número de conjuntos de agrupamientos de datos (clusters) l no se conoce a priori, siendo su límite superior igual al número de medidas del sensor. El algoritmo es:

1. Crear las colecciones de C_k, clusters, llamando a la función Clusterize(p_i).
2. Construir el diagrama de Voronoi ordinario generado por los puntos que forman P_s.
3. Eliminar del diagrama aquellas ramas cuyos dos puntos generados pertenecen al mismo C_k. Eliminar las ramas fuera de P_s. Eliminar las ramas no conectadas (con respecto a la proyección de p_s). Eliminar del diagrama los puntos aislados, si es que los hay.
4. Retornar GLVD.

El algoritmo de construcción del diagrama de Voronoi propuesto en esta tesis también realiza un proceso previo de agrupación de los datos del sensor que pertenecen al mismo objeto, pero a diferencia de [MahSli98], se utilizan los agrupamientos para no generar las ramas que pertenecen al mismo objeto, reduciendo el número de operaciones a realizar.

4. CONSTRUCCIÓN DEL DIAGRAMA DE VORONOI LOCAL (DVL)

4.1 Introducción

En el capítulo anterior se han comentado las aplicaciones del diagrama de Voronoi dentro del campo de la robótica. El diagrama de Voronoi es utilizado para estimar la posición del robot en el entorno por donde se desplaza utilizando sus características tanto topológicas como geométricas, y para planificar la trayectoria que el robot ha de seguir para ir desde la posición donde se encuentra a otra posición final.

Para poder conocer tanto la topología como las características geométricas del entorno por donde el robot se desplaza, y generar así un modelo del entorno que permita tanto localizar al robot como navegar por él, en esta tesis se presenta un algoritmo que construye un mapa del entorno de manera incremental a partir del diagrama de Voronoi. El diagrama de Voronoi es por tanto utilizado para construir un modelo consistente del entorno, para localizar al robot utilizando sus características topológicas y geométricas, y para planificar las trayectorias del robot tanto globalmente como localmente.

La planificación global se realiza utilizando el mapa global que se ha generado, integrando la información de los distintos mapas locales obtenidos en instantes de tiempo diferentes. Una planificación global sería planificar el camino para ir de una habitación A a una habitación B.

La planificación local se realiza utilizando el mapa que se obtiene en cada instante de tiempo y que permite planificar las trayectorias locales del robot. Esta planificación local es importante si el entorno es dinámico. Por ejemplo, en un entorno de interiores puede haber puertas que se abren, puertas que se cierran, personas moviéndose, nuevos obstáculos, etc..

En este capítulo se presenta el algoritmo de construcción del diagrama de Voronoi a partir de la información proporcionada por un telémetro láser.

Como el diagrama de Voronoi construido sólo contiene información de una zona reducida del espacio se le denomina Diagrama de Voronoi Local (DVL).

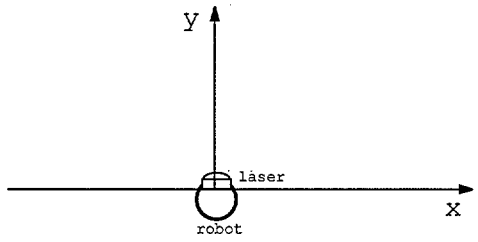


Fig. 4.1: Sistema de coordenadas local cuyo origen se encuentra en el telémetro láser

4.2 Construcción del Diagrama de Voronoi Local (DVL)

Se denota por $S = \{r_1, \cdots, r_i\}$ al conjunto de medidas obtenidas directamente del telémetro láser para un ángulo determinado y ordenadas en orden creciente según el ángulo para el que se obtienen. Se utilizará la generalización del diagrama de Voronoi de series de puntos (ver sección 3.3.2) para la construcción del Diagrama de Voronoi Local (DVL). El problema consiste en encontrar los puntos del espacio que son equidistantes a 2 o más elementos generadores formados por conjuntos de puntos.

Al diagrama de Voronoi construido se le añade el término local porque sólo se tiene una visión de un área localmente restringida del espacio.

Los pasos para calcular el DVL son(ver figura 4.2):

1. **Obtención de los datos del telémetro láser.** El telémetro láser proporciona un vector de medidas de distancias cada 0.5° en un campo de visión de 180° (ver apéndice B). Cada dato representa la distancia del láser a un objeto para cada ángulo.

   Para reducir el número de datos, y por tanto, el número de operaciones a realizar, sólo se utilizan los datos que se encuentran a una distancia menor de d_{max} del sensor láser, en nuestro caso práctico $d_{max} = 4metros$.

   Definición 4.2.1: **Alcance del DVL (d_{max}).** Se define por alcance del DVL, y se denota como d_{max}, a la distancia máxima a la que se han de

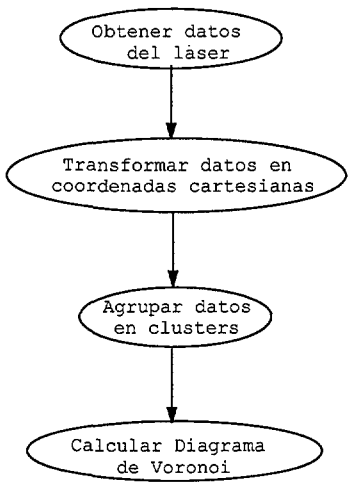


Fig. 4.2: Algoritmo del DVL

encontrar los datos del sensor para ser considerados en la construcción del DVL. Esta distancia tiene como valor máximo el alcance del sensor utilizado:

$$d_{max} \leq rango_{sensor} \tag{4.1}$$

Cuanto mayor es d_{max} mayor es el número de operaciones a realizar, debido a que el número de medidas del sensor a considerar en la construcción del DVL aumenta. Al aumentar el número de medias aumenta por tanto el tiempo de cálculo. Sin embargo, si se elige una d_{max} pequeña puede ocurrir que no exista ninguna medida dentro de esta distancia y no se genere ningún DVL. Debido a estas consideraciones, hay que llegar a un compromiso para la elección de d_{max}.

Definición 4.2.2: **Puntos generadores del DVL (S_{DVL}).** Se define como puntos generadores del DVL, y se denota por S_{DVL}, al conjunto de puntos $\in S$ tales que la distancia de estos puntos al sensor sea menor o igual a d_{max}:

$$S_{DVL} = \{r_1, \cdots, r_n\} \in S \mid r_i \leq d_{max} \tag{4.2}$$

Al reducir el número de puntos a considerar en la construcción del DVL, eliminando los puntos más alejados del sensor, se consiguen dos ventajas importantes: 1.- se reduce el número de operaciones y 2.- se obtiene

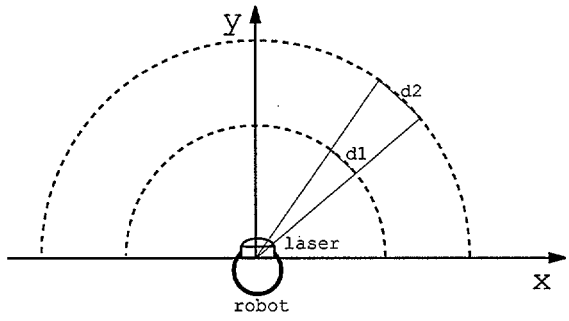


Fig. 4.3: Distancias entre 2 medidas sucesivas para distintos rangos del sensor

un DVL más preciso, ya que, a medida que nos alejamos del sensor la distancia entre dos medidas sucesivas aumenta (ver figura 4.3).

2. **Transformación de coordenadas.** Los datos del telémetro láser, proporcionados en coordenadas polares y que $\in S_{DVL}$, son transformados a coordenadas cartesianas según la expresión 4.3.

$$\begin{aligned} x_i &= dist \cdot cos(\theta_i) \\ y_i &= dist \cdot sin(\theta_i) \end{aligned} \tag{4.3}$$

donde *dist* es el valor de la distancia desde el sensor al objeto para cada ángulo θ. Cada medida del telémetro láser tiene su correspondiente coordenada cartesiana, $r_i = \{x_i, y_i\} \in S_{DVL}$.

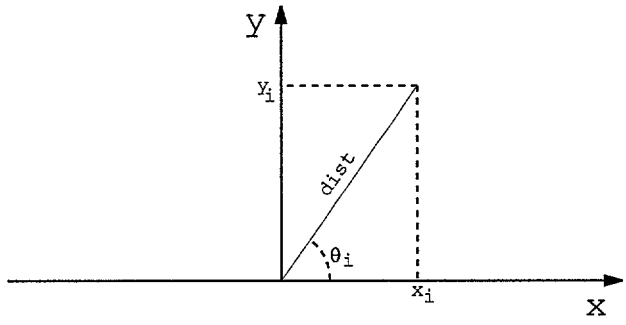


Fig. 4.4: Coordenadas polares y sus correspondientes coordenadas cartesianas

3. **Agrupación de datos.** A continuación se agrupan los datos que pertenecen a un mismo objeto. Este proceso es importante, ya que, influye en la precisión del DVL. A cada conjunto de puntos que pertenecen al mismo objeto se le denomina *grupo generador*.

4. **Cálculo del DVL.** Una vez agrupados los datos, se calcula el DVL hallando los puntos del mapa que son equidistantes al menos a 2 objetos o *grupos generadores*.

En las secciones siguientes se describirán con más detalle cada uno de los pasos mencionados.

4.2.1 Agrupación de los datos que constituyen un mismo objeto

El paso de agrupar los datos que pertenecen al mismo objeto es fundamental en la obtención del DVL. Si se considera que un punto pertenece a un objeto o a un *grupo generador* al que realmente no pertenece se puede obtener un DVL distorsionado, es decir, se pueden obtener ramas o nodos falsos, o ramas con direcciones equivocadas.

Definición 4.2.3: **Grupo generador (c_i).** Se define como grupo generador, y se denota por c_i, al conjunto de puntos del espacio que pertenecen al mismo objeto u entidad, de tal manera que:

$$c_i = \{r_j, \cdots, r_n\} \in S_{DVL} \tag{4.4}$$

Cada *grupo generador* está formado por las medidas del telémetro láser que se encuentran dentro del campo de visión del DVL, $\mathbb{CV}_{DVL}$, y que pertenecen al mismo objeto o entidad, y sus correspondientes coordenadas cartesianas considerando que el origen de coordenadas se encuentra en el sensor:

$$\begin{aligned} c_i &= \{r_j, r_{j+1}, \cdots, r_n\} \\ &= \{(x_j, y_j), (x_{j+1}, y_{j+1}), \cdots, (x_n, y_n)\}, c_i \in S_{DVL} \end{aligned} \tag{4.5}$$

Definición 4.2.4: **Campo de visión ($\mathbb{CV}$).** Se define como campo de visión, y se denota por $\mathbb{CV}$, a la zona del espacio comprendida entre el robot y los puntos que pertenecen a S.

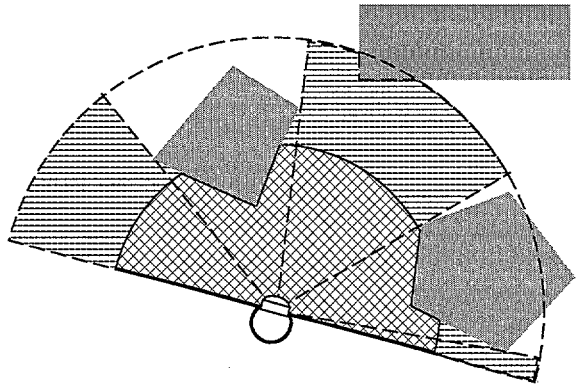


Fig. 4.5: Representación del campo de visión del robot. La zona rayada más la zona con cuadros representa $\mathbb{CV}$. La zona con cuadros representa $\mathbb{CV}_{DVL}$

Definición 4.2.5: **Campo de visión del DVL ($\mathbb{CV}_{DVL}$).** Se define como campo de visión del DVL, y se denota por $\mathbb{CV}_{DVL}$, a la zona del espacio comprendida entre el robot y los puntos que pertenecen a S_{DVL}. $\mathbb{CV}_{DVL} \subseteq \mathbb{CV}$.

Al conjunto de *grupos generadores* se denota por $C = \{c_1, \cdots, c_n\} \in S_{DVL}, (0 \leq n < \infty)$. Los *grupos generadores* no están conectados entre sí, es decir, no tienen ningún punto en común, $c_i \bigcap c_j = \emptyset, i \neq j$, ya que, no puede existir un punto que pertenezca a varios objetos a la vez.

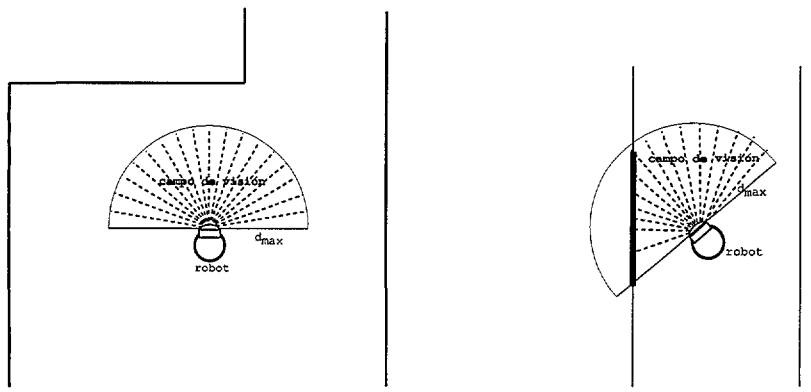
El requisito fundamental para que se genere el DVL es que el número de *grupos generadores* tiene que ser mayor o igual a 2, es decir,

$$dim(C) \geq 2 \tag{4.6}$$

Las dos situaciones por las que no se genera el DVL son: 1.- los objetos se encuentran fuera del alcance del campo de visión (ver figura 4.6(a)) y 2.- el robot sólo ve un objeto, por ejemplo una de las dos paredes de un pasillo (ver figura 4.6(b)).

En el caso de que no se genere un DVL, se puede construir un DVL virtual que garantice la continuidad del diagrama de Voronoi.

Si sólo se forma un *grupo generador* c_1 se construye otro *grupo generador* c_2 que tiene como puntos los puntos extremos del campo de visión $\mathbb{CV}_{DVL}$, tal que, el espacio comprendido entre estos dos grupos $\in \mathbb{CV}_{DVL}$ y la distancia entre ellos, dc_1^2, sea tal que el robot pueda moverse entre ellos (ver figura 4.7(b)).



(a) Ningún objeto dentro del campo de visión

(b) Un sólo objeto dentro del campo de visión

Fig. 4.6: Casos en los que no se genera el Diagrama de Voronoi

Definición 4.2.6: **Distancia entre *grupos generadores* (dc_i^j).** Se define como distancia entre el *grupo generador* i y el *grupo generador* j, y se denota por dc_i^j, a la distancia mínima euclídea que existe entre los puntos que forman dichos *grupos generadores.*

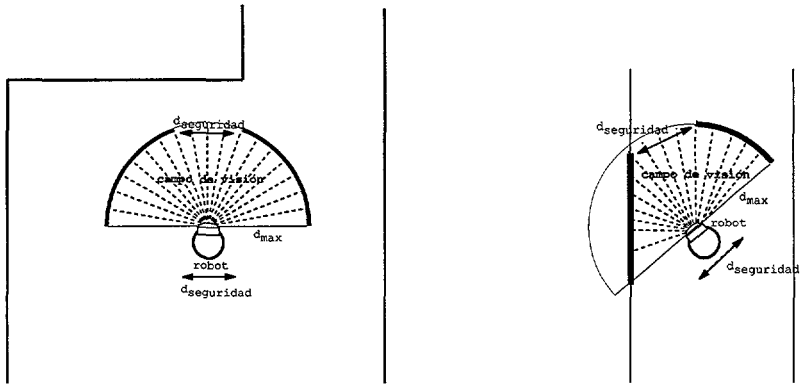
Si no se genera ningún *grupo generador*, el DVL virtual se puede construir considerando que se forman dos *grupos generadores* virtuales que tienen como puntos los puntos extremos del campo de visión ($\mathbb{CV}_{DVL}$), paralelos ambos al desplazamiento del robot y separados una distancia, tal que, permita al robot moverse entre ellos (ver figura 4.7(a)).

Los puntos del DVL que se han obtenido a partir de *grupos generadores* virtuales no se consideran a la hora de localizar al robot, únicamente se utilizan para navegación. Esto es debido, a que estos puntos no representan la topología real del entorno.

Los pasos para agrupar los datos que pertenecen al mismo objeto son:

1. Se agrupan los datos del telémetro láser $\in S_{DVL}$ que no presentan discontinuidades en sus medidas generando los distintos *grupos generadores*, $C = \{c_1, \cdots, c_i\}$.

2. Se agrupan los *grupos generadores* contiguos si las distancias entre sus elementos es menor que un determinado umbral.



(a) Caso en el que se generan dos *grupos generadores* virtuales

(b) Caso en el que se genera un *grupo generador* virtual

Fig. 4.7: *Grupos generadores* virtuales

3. Se agrupan los *grupos generadores* alternos si las distancias entre los elementos de cada uno de ellos es menor que un determinado umbral.

4. Se eliminan los *grupos generadores* que contienen un número de datos menor o igual que un umbral.

5. Se agrupan los *grupos generadores* que ocultan zonas.

A continuación se describe con detalle cada uno de los pasos anteriores.

4.2.1.1 Separación de datos con discontinuidades

Si se estudia los puntos del telémetro láser en coordenadas polares que $\in S_{DVL}$ se comprueba que los objetos son claramente diferenciables porque hay grandes discontinuidades entre los valores de dos medidas sucesivas.

Se considera que $S = \{r_1, \cdots, r_{361}\}$ es el conjunto de puntos proporcionados por el telémetro láser, y $S_{DVL} = \{r_1^{DVL}, \cdots, r_i^{DVL}\} \subseteq S$ es el conjunto de medidas que se encuentran dentro del campo de visión del DVL, $\mathbb{CV}_{\mathbb{DVL}}$. El algoritmo para generar los *grupos generadores* es:

Entrada: S y S_{DVL}.

Salida: $C = \{c_1, \cdots, c_n\}$.

Paso 1: Generar el primer *grupo generador*, c_1, para la primera medida de S que pertenece a S_{DVL}: $r_k = r_1^{DVL} \in c_1$.

Paso 2: Para $i = k+1, \cdots, Dim(S)$, comprobar que $r_i \in S_{DVL}$

Paso 2.1: Si $r_i \in S_{DVL}$ se genera un nuevo *grupo generador* si se cumple uno de los criterios siguientes:

- r_{i-1} no $\in S_{DVL}$.
- Si el valor de la medida r_i es mayor que la distancia $r_{i-1} \in S_{DVL}$ un determinado umbral :

$$|r_i - r_{i-1}| > umbral_{discontinuidad} \tag{4.7}$$

Volver al paso 2.

Paso 2.2: Si $r_i \in S_{DVL}$ pero no cumple cualquiera de los requisitos anteriores entonces $r_i \in$ al último *grupo generador* generado. Volver al paso 2.

Paso 2.3: Si r_i no $\in S_{DVL}$ volver al paso 2.

Paso 3: Devolver C.

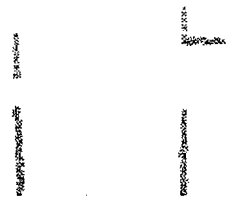
Al final de proceso se obtiene un conjunto de *grupos generadores* que contienen las medidas del sensor que pertenecen al mismo objeto, $C = \{c_1, c_2, \cdots, c_i\} \in \mathbb{CV}_{DVL}$. Este proceso es rápido y fácil de implementar.

En la figura 4.8(a) se representan los datos del telémetro láser en coordenadas polares para un pasillo con puertas a ambos lados. En este caso el número de objetos es 4 siendo fácilmente identificables, ya que, existe una clara discontinuidad entre las medidas de los extremos que forman cada *grupo generador*. En la figura 4.8(b) se representan los datos en coordenadas cartesianas agrupados en *grupos generadores*. El diagrama de Voronoi, considerando sólo el paso de separar datos con discontinuidades, queda como se muestra en la figura 4.8(c). Los puntos en azul representan las celdas equidistantes a 3 o más objetos y se denominan nodos. Las celdas representadas en verde pertenecen a las ramas, es decir, celdas que son equidistantes a 2 objetos.

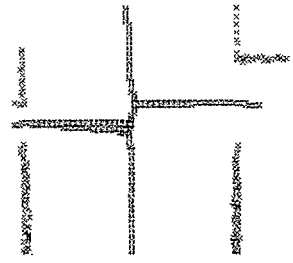
4.2.1.2 Agrupación de *grupos generadores* contiguos

En este segundo paso se agrupan los *grupos generadores* contiguos cuando la distancia entre ellos es inferior a un cierto umbral. De esta manera se

(a) Medidas del telémetro láser en coordenadas polares



(b) Medidas del telémetro láser en coordenadas cartesianas



(c) DVL

Fig. 4.8: DVL para una intersección

simplifica el DVL, ya que, se evita que se generen ramas que el robot no puede seguir debido a que la distancia entre los objetos que las forman es inferior a su anchura.

Definición 4.2.7: ***Grupos generadores*** **contiguos.** Sea $S_{DVL} = \{r_1^{DVL}, \cdots, r_n^{DVL}\} \subseteq S$ el conjunto de medidas del sensor ordenadas desde el ángulo 0^o hasta el ángulo 180^o y que están dentro del campo de visión de $\mathbb{CV}_{DVL}$, donde r_j representa a la medida j del vector. Aplicando el criterio de discontinuidad se generan una serie de *grupos generadores* $C = \{c_1, \cdots, c_k\}$ ordenados según el índice de las medidas que los forman, de tal manera que, el *grupo generador* c_i contiene medidas con índices menor que las medidas que forman c_{i+1}. Los *grupos generadores* contiguos son aquellos que tienen índices sucesivos.

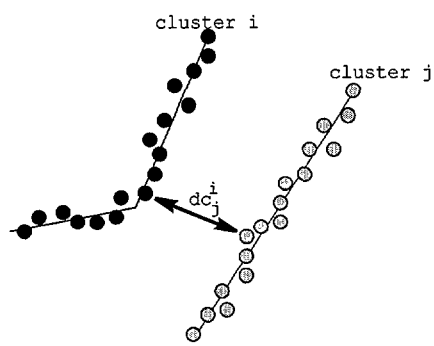


Fig. 4.9: Distancia entre el *grupo generador* i y el *grupo generador* j, dc_i^j

Se consideran dos *grupos generadores* contiguos obtenidos a partir del criterio de discontinuidad (ver sección 4.2.1.1) $c_j = \{r_i, \cdots, r_k\}$ y $c_{j+1} = \{r_{k+1}, \cdots, r_n\}$, entonces las medidas de estos dos *grupos generadores* pertenecerán al mismo objeto si:

$$dc_j^{j+1} < umbral_{\text{clusters contiguos}} \tag{4.8}$$

Si el criterio anterior se cumple, entonces se agrupan los *grupos generadores* c_j y c_{j+1} en uno sólo. El *grupo generador* creado c_i está formado por las medidas que componen estos dos grupos, es decir, $c_i = c_j \bigcup c_{j+1}$.

El valor del umbral puede ser elegido como un valor mayor a la anchura del robot móvil. Si el robot móvil no puede pasar entre dos objetos no se calcula la rama que los atraviesa reduciendo el tiempo de cálculo.

Se considera que las medidas en coordenadas polares dadas por el sensor son las que muestran en la figura 4.10(a). Si se aplica sólo el paso de discontinuidad el número de *grupos generadores* que se detectan es 6 (ver figura 4.10(b)), quedando el DVL como se muestra en la figura 4.10(c).

Se puede observar que aparece una rama que atraviesa dos objetos que están muy próximos entre sí, por donde es imposible que el robot pase. En la figura 4.10(d) se representan los datos en coordenadas cartesianas y los *grupos generadores* creados a partir del nuevo criterio. El número de *grupos generadores* se reduce de 6 a 5, eliminándose la rama que el robot no puede seguir y quedando el DVL como se observa en la figura 4.10(e).

4.2.1.3 Agrupación de *grupos generadores* alternos

La agrupación de *grupos generadores* alternos tiene como objetivo agrupar los *grupos generadores* que son alternos y cuya distancia entre ellos es inferior a la anchura del robot.

Definición 4.2.8: ***Grupos generadores* alternos.** Sea $S_{DVL} = \{r_1^{DVL}, \cdots, r_n^{DVL}\} \subseteq S$ el conjunto de medidas del sensor ordenadas desde el ángulo 0^o hasta el ángulo 180^o y que están dentro del campo de visión de $\mathbb{CV}_{DVL}$, donde r_j representa a la medida j del vector. Aplicando los criterios de discontinuidad y agrupamiento de *grupos generadores* contiguos se generan una serie de *grupos generadores* $C = \{c_1, \cdots, c_k\}$ ordenados según el índice de las medidas que los forman, de tal manera que, el *grupo generador* c_i contiene medidas con índices menor que las medidas que forman c_{i+1}. Los *grupos generadores* alternos son aquellos entre los que existe otro *grupo generador*.

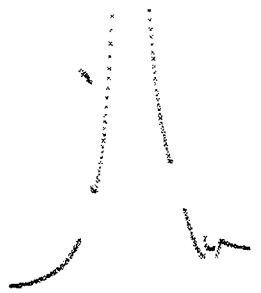
Se supone un conjunto de *grupos generadores* $C = \{c_1, \cdots, c_n\}$. Dos *grupos generadores* alternos $c_k = \{r_j, \cdots, r_p\}$ y $c_{k+2} = \{r_m, \cdots, r_l\}$ están formados por medidas que pertenecen al mismo objeto si:

$$dc_k^{k+2} < umbral_{\text{clusters alternos}} \tag{4.9}$$

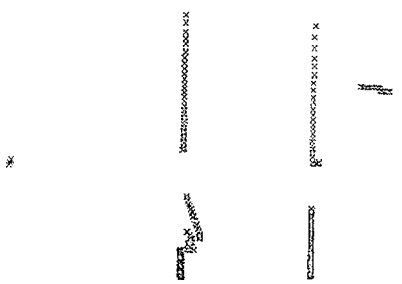
El criterio de selección del umbral puede ser el mismo que en el caso anterior, se elige un valor de umbral mayor que la anchura del robot evitando que se formen ramas que el robot no puede seguir.

El nuevo *grupo generador* tiene como medidas la de estos dos *grupos generadores*, tal que, $c_j = c_k \bigcup c_{k+2}$. Las medidas del *grupo generador* intermedio c_{k+1} se eliminan, ya que, no aportan información al DVL.

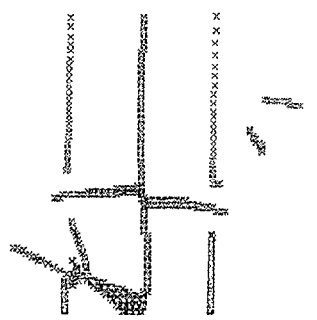
En la figura 4.11(a) se muestran los datos en coordenadas polares obtenidos del sensor y en la figura 4.11(b) los *grupos generadores* creados aplicando los



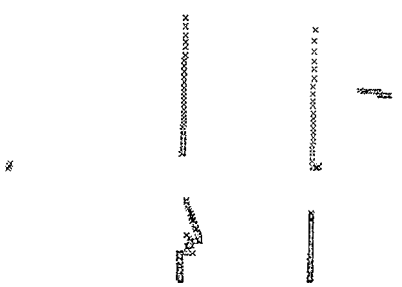
(a) Medidas del telémetro láser en coordenadas polares



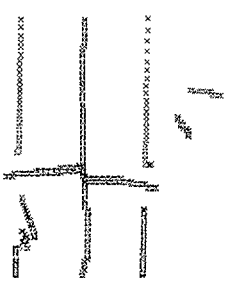
(b) Medidas del telémetro láser en coordenadas cartesianas sin agrupar *grupos generadores* contiguos



(c) DVL sin agrupar *grupos generadores* contiguos



(d) Medidas del telémetro láser en coordenadas cartesianas agrupando *grupos generadores* contiguos



(e) DVL agrupando *grupos generadores* contiguos

Fig. 4.10: DVL para una intersección y una puerta abierta a la izquierda

dos criterios anteriores. El número de *grupos generadores* es igual a 6. El DVL generado se muestra en la figura 4.11(c) donde aparece una rama que se puede eliminar porque la distancia entre los *grupos generadores* que la forman es inferior a la anchura del robot.

Si se unen los *grupos generadores* alternos cuya distancia es inferior a un umbral los resultados que se obtienen se muestran en la figura 4.11(d), donde se representa los *grupos generadores* creados, y en 4.11(e), donde se representa el DVL construido. El número de *grupos generadores* se reduce y el DVL se simplifica.

4.2.1.4 Eliminación de *grupos generadores* con pocas medidas

Para evitar un DVL distorsionado, es decir, un DVL con ramas o puntos falsos, se eliminan los *grupos generadores* que tienen un número de medidas igual o inferior a un umbral.

Se considera un conjunto de *grupos generadores* $C = \{c_1, c_2, \cdots, c_n\}$, se eliminan de dicho conjunto los *grupos generadores* en los que se cumple:

$$dim(c_i) < umbral_{medidas} \tag{4.10}$$

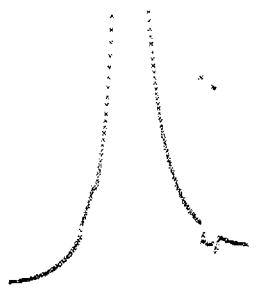
Definición 4.2.9: **Dimensión de un *grupo generador* ($dim(c_i)$).** Se define como dimensión de un *grupo generador*, y se denota por $dim(c_i)$, al número de puntos que forman el *grupo generador* c_i.

En la figuras 4.12(a) y 4.12(b) se muestran las medidas del sensor en coordenadas polares y las medidas en coordenadas cartesianas agrupadas en *grupos generadores* respectivamente. Se observa que hay un *grupo generador* que tiene una única medida y que genera un DVL bastante distorsionado (ver figura 4.12(c)). Si se elimina este *grupo generador* (figura 4.12(d)) el nuevo DVL queda según la figura 4.12(e).

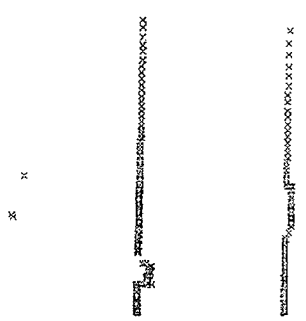
4.2.1.5 Agrupación de los *grupos generadores* de zonas ocultas

Como ya se ha indicado anteriormente, el campo de visión de los sensores es limitado, esto hace que aparezcan zonas ocultas, es decir, zonas del espacio que no son visibles al robot debido a la presencia de objetos en el entorno que se interponen entre el sensor y esta zona.

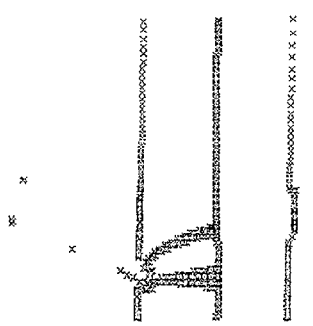
La presencia de zonas ocultas en el entorno puede provocar que se genere un DVL con información falsa, por ejemplo, que se generen ramas o nodos que no existen realmente. Esto influye en las tareas de planificación y localización que puede llevar a cabo el robot utilizando el DVL.



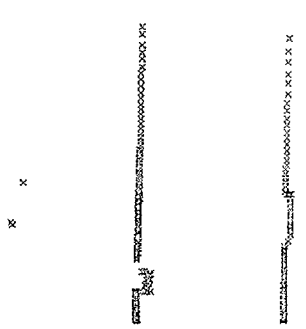
(a) Medidas del telémetro láser en coordenadas polares



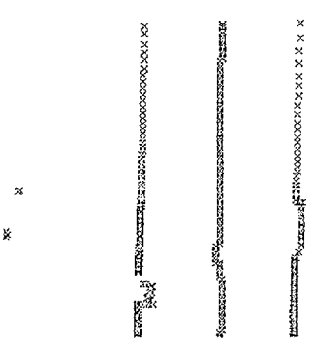
(b) Medidas del telémetro láser en coordenadas cartesianas sin agrupar *grupos generadores* alternos



(c) DVL sin agrupar *grupos generadores* alternos



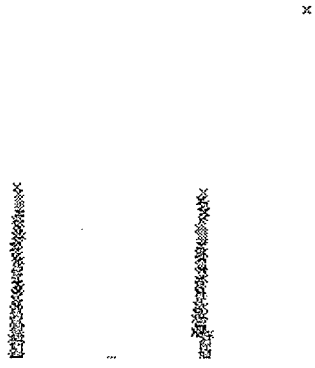
(d) Medidas del telémetro láser en coordenadas cartesianas agrupando *grupos generadores* alternos



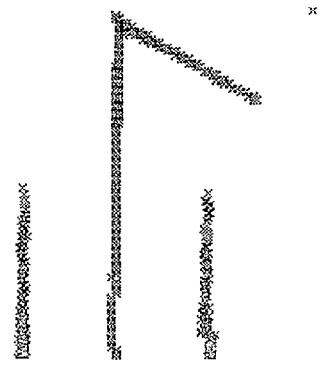
(e) DVL agrupando *grupos generadores* alternos

Fig. 4.11: DVL para un pasillo con una caja para extintores

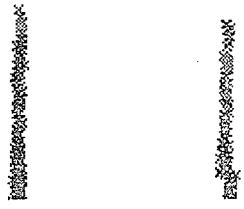
(a) Medidas del telémetro láser en coordenadas polares



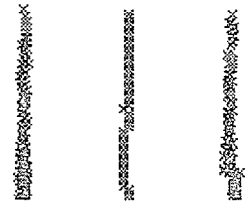
(b) Medidas del telémetro láser en coordenadas cartesianas sin eliminar los *grupos generadores* con un número de puntos bajo



(c) DVL sin eliminar los *grupos generadores* con un número de puntos bajo



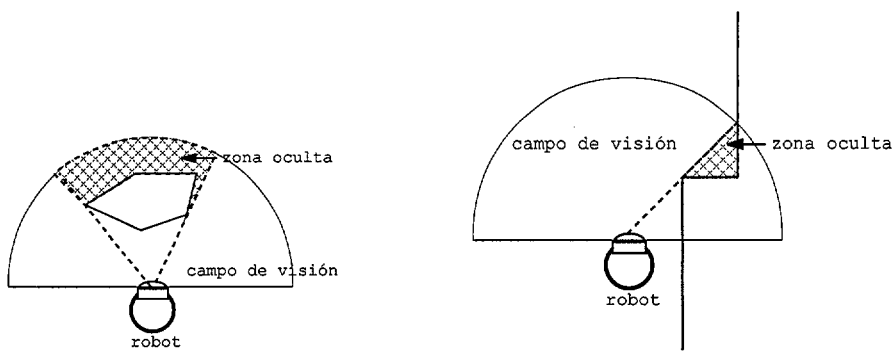
(d) Medidas del telémetro láser en coordenadas cartesianas eliminando *grupos generadores* con un número de puntos bajo



(e) DVL eliminando los *grupos generadores* con un número de puntos bajo

Fig. 4.12: DVL para el paso de un pasillo a un vestíbulo

En un entorno de interiores estructurado, las esquinas juegan un papel fundamental en la aparición de zonas ocultas. Una esquina puede generar una zona oculta al sensor, de tal manera, que exista una discontinuidad entre 2 valores de medidas sucesivas del sensor, considerando como objetos u entidades distintas medidas que realmente pertenecen a la misma entidad. Esto provoca la aparición de una rama que no existe realmente.



(a) Presencia de un obstáculo (b) Presencia de una esquina

Fig. 4.13: Zonas ocultas

Se consideran dos *grupos generadores* $c_j = \{r_k, \cdots, r_{i-1}\}$ y $c_{j+1} = \{r_i, \cdots, r_n\}$, con dos medidas sucesivas r_{i-1} y r_i. Si estas dos medidas están en el primer cuadrante, considerando el centro del sistema de coordenadas en el sensor, entonces existe una zona oculta si se cumple que (ver figura 4.14(a)):

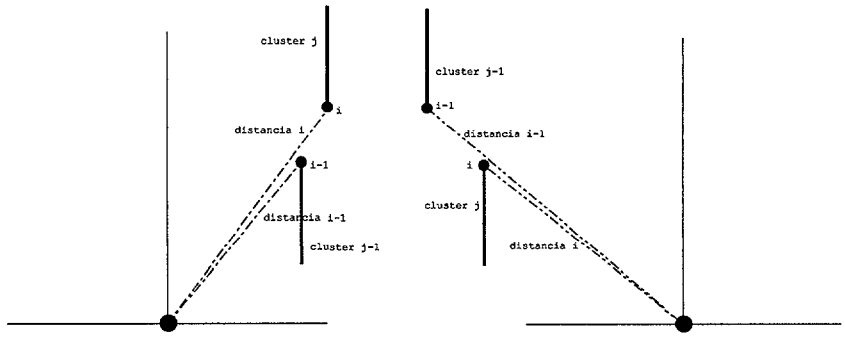
$$r_{i-1} < r_i \tag{4.11}$$

Si dos medidas sucesivas se encuentran en el segundo cuadrante, entonces se ha de cumplir que (ver figura 4.14(b)):

$$r_{i-1} > r_i \tag{4.12}$$

El paso de un pasillo a un vestíbulo se muestra en las figuras 4.15. En este caso la esquina queda oculta. Si no se aplica el criterio de zona oculta, la pared del pasillo y del vestíbulo, que pertenecen al mismo objeto, se considera como objetos distintos (ver figura 4.15(b)) generando una rama en el DVL que no existe realmente (ver figura 4.15(c)).

Si se aplica el criterio de zonas ocultas los resultados obtenidos se muestran en las figuras 4.15(d), donde se representa los *grupos generadores*, y



(a) Caso 1 (b) Caso 2

Fig. 4.14: Zonas ocultas según el cuadrante

en 4.15(e), donde se representa el DVL. La rama que atraviesa la esquina inferior derecha se elimina.

Al aplicar el criterio de zonas ocultas, puede ocurrir que se dejen de ver puertas (ver figura 4.16). Pero esto sólo sucede cuando el robot se encuentra alejado de ella. Al avanzar el robot y aproximarse a la puerta esta ya será detectada por los sensores, generándose la rama del DVL que la atraviesa.

4.2.2 Construcción del DVL

Una vez que se han agrupado las medidas del sensor que pertenecen al mismo objeto se procede a la construcción del DVL.

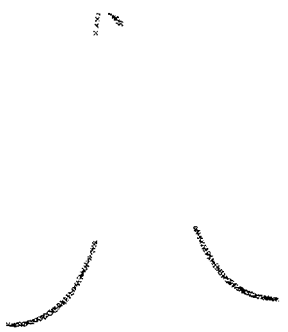
Para cada punto del espacio libre, $p_{li} = (x_i, y_i) \in L$, existe al menos un punto del espacio ocupado $p_{oj} = (x_j, y_j) \in O$, para el que la distancia es menor. A todo este conjunto de puntos se denomina *puntos base*, PB.

Definición 4.2.10: **Punto bases (PB).** Se define como puntos base, y se denota por PB, al conjunto de puntos del espacio ocupado tal que:

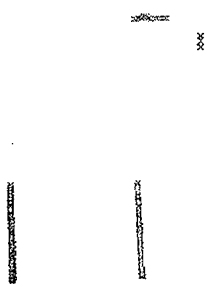
$$PB = \{\{p_{oi}, \cdots, p_{oj}\} \ : \ \forall p_{lh} \in L \ \exists \ p_{ok} \mid d(p_{ok}, p_{lh}) = \min_{p_{on} \in O} \|p_{on} - p_{lh}\|\} \tag{4.13}$$

El DVL estará formado por todos aquellos puntos que pertenecen al espacio libre y que son equidistantes al menos a 2 puntos base.

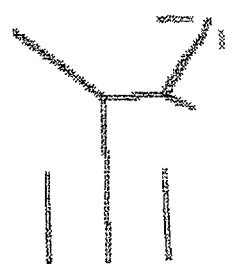
El conjunto de puntos del espacio ocupado, $O = \{(x_{o1}, y_{o1}), \cdots, (x_{on}, y_{on})\}$, se pueden agrupar en subconjuntos o *grupos generadores*, de tal manera que,



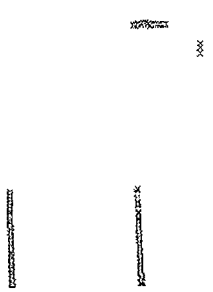
(a) Medidas del telémetro láser en coordenadas polares



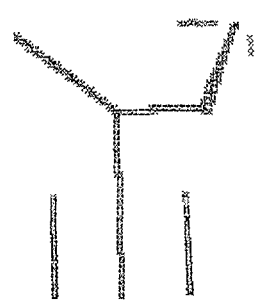
(b) Medidas del telémetro láser en coordenadas cartesianas sin agrupar los *grupos generadores* de zonas ocultas



(c) DVL sin agrupar los *grupos generadores* de zonas ocultas

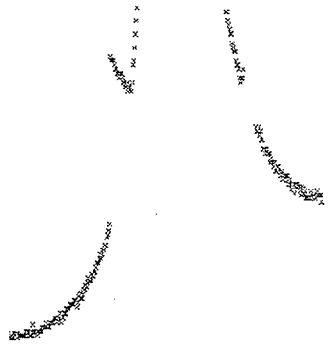


(d) Medidas del telémetro láser en coordenadas cartesianas agrupando los *grupos generadores* de zonas ocultas

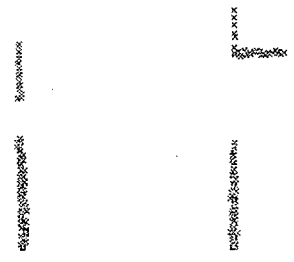


(e) DVL agrupando los *grupos generadores* de zonas ocultas

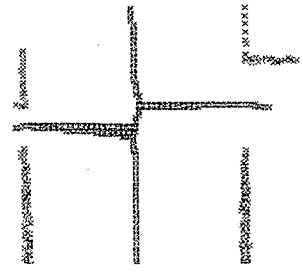
Fig. 4.15: DVL para el paso de un pasillo a un vestíbulo con presencia de zonas ocultas



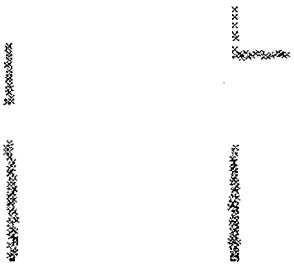
(a) Medidas del telémetro láser en coordenadas polares



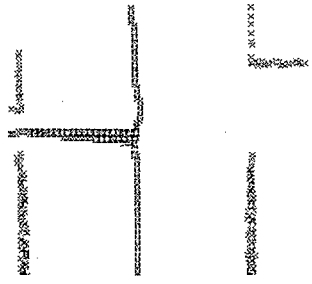
(b) Medidas del telémetro láser en coordenadas cartesianas sin agrupar el *grupo generador* de zonas ocultas



(c) DVL sin agrupar los *grupos generadores* de zonas ocultas



(d) Medidas del telémetro láser en coordenadas cartesianas agrupando los *grupos generadores* de zonas ocultas



(e) DVL agrupando los *grupos generadores* de zonas ocultas

Fig. 4.16: DVL para una intersección con presencia de zonas ocultas

a cada subconjunto pertenecen todos aquellos puntos que forman parte de la misma entidad u objeto, $C = \{c_1, c_2, \cdots, c_i\}$. Un punto del DVL se dice que pertenece a una rama cuando es equidistante a 2 puntos bases, y se dice que es un nodo cuando es equidistante a más de 2 puntos base. Todos los puntos equidistantes a los mismos *grupos generadores* pertenecen a la misma rama o nodo.

En las secciones anteriores se describió el algoritmo para agrupar los puntos del espacio ocupado en *grupos generadores*. Estos puntos del espacio ocupado son los puntos proporcionados por el sensor, en este caso un telémetro láser.

Los pasos necesarios para obtener el DVL son:

1. Discretización de la región visible, $\mathbb{CV}_{DVL}$.
2. Calculo de la distancia de cada celda, perteneciente a la región visible, a cada uno de los *grupos generadores*.
3. Obtención de los nodos y ramas que forman el DVL.

A continuación se describen cada uno de los procesos.

4.2.2.1 Discretización de la región visible

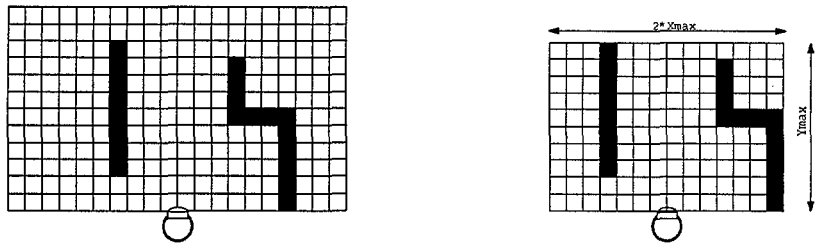
Para facilitar y simplificar el cálculo del DVL la región visible, $\mathbb{CV}$, es dividida en celdillas.

Definición 4.2.11: **Región visible**. Se define región visible a todos los puntos del espacio libre que son visibles por el sensor del robot, es decir, que no están ocultos.

Puede ocurrir que todos los puntos que pertenecen al espacio ocupado, $C = \{(x_{o1}, y_{o1}), \cdots, (x_{on}, y_{on})\} \in S_{DVL}$, estén a una distancia inferior a d_{max}. En este caso se puede reducir el tamaño del mapa reduciendo el número de operaciones a realizar. Para ello se calcula el valor de x máximo y el valor de y máximo de todos los puntos que forman los *grupos generadores*. Como los valores de x pueden ser positivos y negativos se elige el valor absoluto mayor de los dos:

$$\begin{aligned} x_{max} &= \max\{\| x_i \|\}, \quad \forall x_i \in C \\ y_{max} &= \max\{\| y_i \|\}, \quad \forall y_i \in C \end{aligned} \tag{4.14}$$

Las dimensiones de la malla resultante son aproximadamente $(2 \cdot x_{max}, y_{max})$ (ver figura 4.17), teniendo en cuenta que estas han de ser múltiplo del valor



(a) Mapa basado en celdillas de la zona dentro del rango del sensor láser

(b) Reducción de las dimensiones del mapa basado en celdillas

Fig. 4.17: Discretización de la región visible

de la *resolución*. El valor de la resolución elegida tiene que ser un compromiso entre el tiempo de ejecución y la precisión que se requiere en el DVL.

Se considera una malla con valor de resolución igual a *res*, el número de filas igual a n y el número de columnas igual a m. Se considera que t es el tiempo que se emplea en calcular el DVL de una malla de dimensiones $(n \cdot m)$. Si se reduce el valor de *res*, tal que, $res_{nueva} = \frac{res}{k}$ donde $k \in \mathbb{R}$, la malla que se obtiene tiene por dimensiones $(k \cdot n, k \cdot m)$, y el tiempo que se tarda en calcular el nuevo DVL es aproximadamente $t_{nuevo} = k^2 \cdot t$. El tiempo de cálculo del DVL depende de la resolución elegida para la malla, cuanto mayor es el valor de resolución menor es el tiempo de cálculo, pero se obtiene un DVL con información más pobre.

En la figura 4.18 se muestra un DVL para distintos valores de resolución de celdilla. Cuanto mayor es el tamaño de celdilla menor es la precisión del DVL, sin embargo el tiempo de ejecución disminuye considerablemente.

4.2.2.2 Cálculo de la distancia de cada celdilla a cada uno de los *grupos generadores*

Para cada una de las celdillas que componen el espacio discretizado se calcula la distancia del centro de la celdilla a cada uno de los *grupos generadores*. La distancia que se calcula es la distancia euclídea:

$$d = |r_j - r_i| = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{4.15}$$

Como cada *grupo generador* está formado por un conjunto de puntos, $c_i = \{r_j, \cdots, r_n\} = \{(x_j, y_j), \cdots, (x_n, y_n)\} \in S_{DVL}$, la distancia que se considera es la distancia más pequeña que hay desde el centro de la celdilla k, con

(a) DVL para una resolución de celdilla de 2 cm. Tiempo de cálculo 1373 ms

(b) DVL para una resolución de celdilla de 10 cm. Tiempo de cálculo 57 ms

Fig. 4.18: DVL para distintos valores de resolución de la celdilla

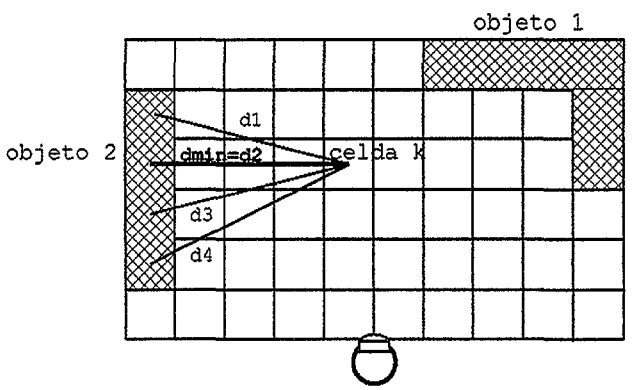


Fig. 4.19: Distancia desde una celdilla a un objeto

coordenadas $p_{ck} = (x_{ck}, y_{ck})$, a cada uno de los puntos que forman el *grupo generador* i.

Definición 4.2.12: **Distancia desde una celdilla a un *grupo generador*** (d^i_{ck}). Se define como distancia desde la celdilla k al *grupo generador* c_i, y se denota por d^i_{ck}, a la distancia tal que:

$$d^i_{ck} = \min\{d_{kp}\} \mid d_{kp} = \|p_{ck} - r_p\| \; \forall r_p \in c_i \tag{4.16}$$

Al discretizar el espacio visible se comete un error, de tal manera que, si d^i_{ck} es la distancia desde el centro de la celda k al *grupo generador* i, cualquier

otro punto de dicha celda tendrá una distancia d_k^i comprendida entre:

$$d_{ck}^i - \frac{\sqrt{2} \cdot res}{2} < d_k^i < d_{ck}^i + \frac{\sqrt{2} \cdot res}{2} \tag{4.17}$$

donde res es la resolución de la celda.

De esta manera se puede calcular el error máximo cometido al considerar el centro de la celdilla como el punto desde donde se calcula la distancia a los *grupos generadores*:

$$E_{max} = \max\{d_{ck}^i - d_k^i\} = \frac{\sqrt{2} \cdot res}{2} \tag{4.18}$$

En la tabla 4.1 se muestran distintos valores de errores máximos cometidos para distintos valores de resolución de discretización.

res(cm)	E_{max}(cm)	$2 \cdot E_{max}$(cm)
1	0.71	1.41
3	2.12	4.24
5	3.53	7.07
7	4.95	9.9
9	6.36	12.73

Tab. 4.1: Errores máximos cometidos para distintos valores de resolución de discretización

Si las distancias desde una celdilla a diferentes *grupos generadores* difieren en una cantidad menor que $2 \cdot E_{max}$ se considera que esta celdilla forma parte del DVL, ya que, puede existir un punto dentro de esta celdilla que sea equidistante a varios *grupos generadores*:

$$\| d_{ck}^i - d_{ck}^j \| < (2 \cdot E_{max}) \tag{4.19}$$

La celdilla formará parte de una rama del DVL si es equidistante a dos *grupos generadores*, y formará parte de un nodo del DVL si es equidistante a más de dos *grupos generadores*.

Si el valor de resolución que se elige es mayor que el error que se comete en las medidas debido al sensor, entonces en E_{max} ya se considera este error.

4.2.2.3 Obtención de las ramas y nodos que forman el DVL

Sea el espacio ocupado formado por un conjunto de puntos proporcionados por el sensor y agrupados en *grupos generadores*, $C = \{c_1, \cdots, c_n\} \in$

S_{DVL}, se denota a la distancia desde el centro de la celdilla i a cada uno de los *grupos generadores* como:

$$D_i = \{d_{ci}^1, \cdots, d_{ci}^n\} \tag{4.20}$$

Se considera que la distancia d_{ci}^j es la distancia menor de todas las distancias desde la celdilla a cada uno de los puntos que forman el *grupo generador* si:

$$d_{ci}^j + 2 \cdot E_{max} < d_{ci}^k \quad \begin{matrix} 1 \leq k \leq n \\ k \neq j \end{matrix} \tag{4.21}$$

Si se cumple esta condición entonces se puede asegurar que la distancia desde cualquier punto desde esta celda al *grupo generador* j es la menor.

Si entre dos distancias, d_{ci}^j y d_{ci}^p, esta condición no se cumple, es decir, si

$$d_{ci}^j - d_{ci}^p < 2 \cdot E_{max} \tag{4.22}$$

implica que la celda es equidistante a dos *grupos generadores*, y por tanto forma parte de una rama del DVL, y se denota por R_{ci}. Si ocurre lo mismo con más de 2 distancias, entonces la celda forma parte de un nodo del DVL, y se denota por N_{ci}.

Definición 4.2.13: **Distancia mínima desde una celdilla a un conjunto de *grupos generadores* (DC_i)** . Se define como distancia mínima desde una celdilla i a un conjunto de *grupos generadores* C, y se denota por DC_i, al conjunto de distancias mínimas que $\in D_i$ y que cumplen el criterio dado por la ecuación 4.22.

Una celda forma parte de una rama, y se denota por R_{ci}, si $dim(DC_i) = 2$. Una celda forma parte de un nodo, y se denota por N_{ci}, si $dim(DC_i) > 2$.

Aquellas celdas que equidisten de los mismos *grupos generadores* formaran parte de la misma rama o nodo del DVL. Una rama estará formada por todos aquellos puntos que equidisten de los mismos 2 *grupos generadores*. Un nodo estará formado por todos aquellos puntos que equidisten de al menos los mismos 3 *grupos generadores*.

Definición 4.2.14: ***Grupos generadores* base (CB_i)**. Se denomina *grupos generadores* base, y se denota por CB_i, al conjunto de *grupos generadores* que son equidistantes a la celdilla c_i.

Definición 4.2.15: **Rama del DVL (R_i).** Se define rama del DVL, y se denota por R_i, al conjunto de puntos tal que:

$$R_i = \{R_{ci}, \cdots, R_{cn}\} \mid \forall R_{ck} \in R_i, \quad CB_k = CB_p \text{ para } k, p \leq n \text{ y } p \neq k \text{ y } dim(DC_k) = 2 \tag{4.23}$$

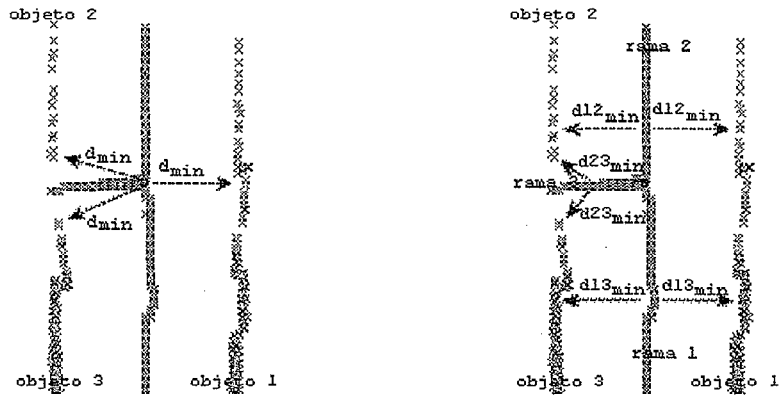
Definición 4.2.16: **Nodo del DVL (N_i).** Se define nodo del DVL, y se denota por N_i, al conjunto de puntos tal que:

$$N_i = \{N_{ci}, \cdots, N_{cn}\} \mid \forall N_{ck} \in N_i, \quad CB_k = CB_p \text{ para } k, p \leq n \text{ y } p \neq k \text{ y } dim(DC_k) > 2 \tag{4.24}$$

El DVL generado está formado un conjunto de puntos que forman las ramas y nodos:

$$\mathbb{DVL} = \{R, N\} = \{\{R_1, \cdots, R_n\}, \{N_1, \cdots, N_k\}\} \tag{4.25}$$

Puede ocurrir que $dim(R) = \emptyset$ o que $dim(N) = \emptyset$, pero siempre se tiene que cumplir que $dim(R) + dim(N) \geq 1$ para que exista el DVL.



(a) Distancias desde el nodo a cada uno de los *grupos generadores* equidistantes

(b) Distancias desde un punto que forma cada rama a los *grupos generadores* a los que son equidistantes

Fig. 4.20: *grupos generadores* que son equidistantes al nodo y a las ramas

En las figuras 4.20(a) y 4.20(b) se representa un entorno correspondiente a un pasillo con una puerta abierta a la izquierda. Los datos del telémetro

láser (puntos en color rojo) se agrupan en tres *grupos generadores* u objetos. El DVL generado está formado por un nodo (color azul) y tres ramas (color verde). En la figura 4.20(a) se observa que los puntos que forman el nodo son equidistantes a los tres objetos. En la figura 4.20(b) se observa que los puntos de la rama 1 son equidistantes a los objetos 1 y 3, los puntos de la rama 2 a los objetos 1 y 2, y los puntos de la rama 3 a los objetos 2 y 3.

5. CONSTRUCCIÓN DEL MAPA TOPO-GEOMÉTRICO LOCAL

5.1 Introducción

Si un robot móvil autónomo no tiene un conocimiento previo del entorno, la planificación solo se puede llevar a cabo si va acompañada de un modelado del entorno. Los mapas topológicos modelan el entorno como un conjunto de estados y sus conexiones, es decir, que estados son alcanzados desde otros estados y en algunas ocasiones que acciones hay que realizar para ir de un estado a otro. Los modelos topológicos determinan la posición del robot con respecto al modelo primario basado en marcas. Por ejemplo, si el robot atraviesa dos zonas muy parecidas el modelo topológico tiene frecuentemente problemas para determinar si estas zonas son la misma o no. Por otra parte, la entrada sensorial depende fuertemente del punto de vista del robot, los modelos topológicos pueden fallar al reconocer geométricamente zonas cercanas incluso en entornos estáticos, haciendo difícil la construcción de mapas de grandes dimensiones, particularmente si la información proporcionada por los sensores es ambigua.

La ventaja principal de los mapas topológicos es que son muy compactos. La resolución de los mapas topológicos depende exclusivamente de la complejidad del entorno. Esta característica hace que los mapas topológicos presenten 3 ventajas sobre los mapas basados en celdillas:

1. Planificación rápida.

2. Facilitan la interfase a planificadores simbólicos.

3. Proporcionan interfases más naturales para las instrucciones humanas (ir a la habitación A).

Por otra parte, los modelos topológicos no requieren la posición geométrica exacta del robot.

Hay dos estrategias típicas para generar mapas topológicos: una consiste en tratar de aprender el mapa topológico directamente y la otra es construirlo a partir de la información proporcionada por un mapa geométrico.

Frecuentemente estos mapas topológicos son enriquecidos con información geométrica dando lugar a mapas topo-geométricos. Esta información geométrica puede ser por ejemplo la longitud y anchura de un pasillo.

En este capítulo se describe como se construye el mapa topo-geométrico local en nuestro sistema de modelado a partir del DVL generado. Este mapa local es la representación que se obtiene del entorno en un instante de tiempo determinado y sólo posee información de una zona restringida del entorno.

La información del mapa local es posteriormente integrada con la información de los mapas locales obtenidos en instantes anteriores, para construir de una manera incremental el mapa del entorno por donde el robot se desplaza.

El mapa local generado es un mapa topológico que se obtiene a partir del Diagrama de Voronoi Local (DVL) utilizando las medidas de un telémetro láser.

El DVL representa el camino, o conjunto de puntos, que son equidistantes a los distintos objetos que se encuentran en el entorno. Es el camino más seguro por el que el robot puede desplazarse. A partir del DVL se obtienen los diferentes nodos y ramas que componen el mapa topológico local.

Este mapa topológico es enriquecido con información geométrica, como es la posición de los nodos en el mapa y los puntos que forman las ramas. El mapa generado es, por tanto, un mapa *topo-geométrico.*

5.2 Construcción del mapa topo-geométrico local

El mapa de estados o topológico se genera a partir del Diagrama de Voronoi Local (DVL). Este mapa no solo contiene información topológica, estados que representan lugares característicos de entornos de interiores como son intersecciones, paso de un vestíbulo a un pasillo, puertas, etc. [BoaPalBla02] y sus correspondientes relaciones, sino que además contiene la posición de cada uno de los puntos que los forman con respecto al sistema de coordenadas centrado en el sensor, y la distancia de estos puntos a los objetos a los que son equidistantes. Se dice, por tanto, que el mapa generado es un mapa topo-geométrico.

5.2.1 Generación del mapa topo-geométrico local a partir de Diagrama de Voronoi Local

Como ya se ha indicado, el mapa topo-geométrico local se construye a partir del DVL. El algoritmo de construcción del DVL se ha explicado en el capítulo 4. Se considera que los nodos del DVL son nodos, o estados, del mapa topológico. Las ramas que unen los nodos del DVL unen también los estados del mapa topológico. El mapa topológico no contiene solamente las nodos y relaciones existentes entre estos nodos, sino que es enriquecido con información geométrica como es la posición de los nodos y la posición de los puntos que forman las ramas.

La manera de proceder es la siguiente:

1. Cada nodo del mapa topológico local esta formado por todos aquellos puntos que son equidistantes a los mismos 3, o más, *grupos generadores*, es decir, los nodos del DVL son nodos (también llamados estados) del mapa topo-geométrico local.

   Definición 5.2.1: **Estado.** Un estado representa un lugar característico de un entorno. En el caso de un entorno de interiores estructurado los estados pueden ser: puertas, intersecciones, etc.. Los estados del mapa topo-geométrico local son los nodos del DVL.

   Cada nodo N_i del mapa está formado por un conjunto de puntos $\{p_k, \cdots, p_n\} = \{N_{ck}, \cdots, N_{cn}\}$, tal que, todos ellos son equidistantes al mismo conjunto de *grupos generadores* CB_j, siendo su dimensión igual o mayor a 3, $dim(CB_j) \geq 3$.

   En cada nodo se almacena también la distancia equidistante de cada punto a los objetos del entorno $\{DC_K, \cdots, DC_n\}$, y un identificador que diferencia los distintos nodos que forman el mapa topo-geométrico local.

2. Una vez obtenidos todos los nodos se calculan las ramas que los unen. Dos nodos estarán unidos si tienen al menos dos *grupos generadores*, a los que son equidistantes, en común. Por tanto, los puntos que forman las ramas, que unen dos nodos, son equidistantes a estos mismos *grupos generadores.*

   Cada rama del mapa local, R_i está formado por un conjunto de puntos $\{p_k, \cdots, p_n\} = \{R_{ck}, \cdots, R_{cn}\}$, tal que, todos ellos son equidistantes

al mismo conjunto de *grupos generadores*, CB_j, siendo su número de elementos igual a 2, $dim(CB_j) = 2$.

Además, en cada rama se almacena la distancia equidistante de cada punto a los objetos del entorno $\{DC_K, \cdots, DC_n\}$, y el conjunto de objetos a los que es equidistante.

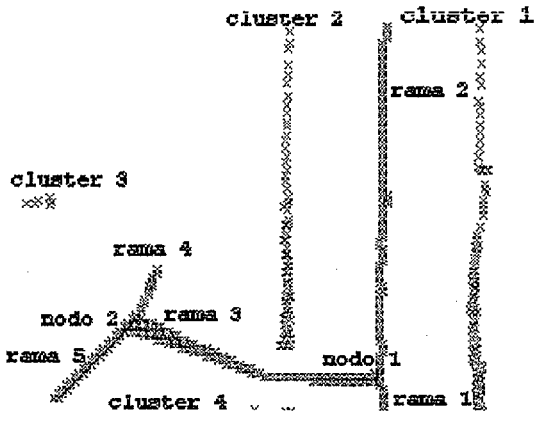


Fig. 5.1: Nodos y ramas que componen el DVL

A diferencia de los modelos puramente topológicos, el modelo construido en esta tesis contiene además información geométrica. A cada nodo se le asigna un punto geométrico que representa su localización en el entorno, y a cada rama se le asocia el conjunto de puntos que la componen.

Los nodos que delimitan las ramas pueden ser de dos tipos:

- **Nodo real**: nodo que tiene un significado físico, por ejemplo, que represente la entrada a una puerta, el paso de un pasillo a un vestíbulo, una bifurcación, etc.. Estos nodos tienen una posición física en el mapa.
- **Nodo ficticio**: nodo que no tiene un significado físico y por tanto no tiene una posición física en el mapa. Estos nodos delimitan las ramas exteriores.

Puede ocurrir que una rama este delimitada por un nodo real y por un nodo ficticio, por dos nodos reales, o por dos ficticios.

En la figura 5.1 se representa el DVL de un pasillo con una puerta abierta a la izquierda. El DVL está formado por dos nodos (puntos en color azul), que serán nodos del mapa topo-geométrico local. El nodo 1 es equidistante

a los objetos 1, 2 y 4. El nodo 2 es equidistante a los objetos 2, 3, 4. Ambos nodos tienen 2 objetos equidistantes en común (2 y 4), por tanto debe existir una rama que los una y cuyos puntos que la forman sean equidistantes a estos mismos objetos. Esta rama es la rama 3. Las ramas 1, 2 y 4 están delimitas por un nodo real y un nodo ficticio, mientras que la rama 3 está delimitada por dos nodos reales.

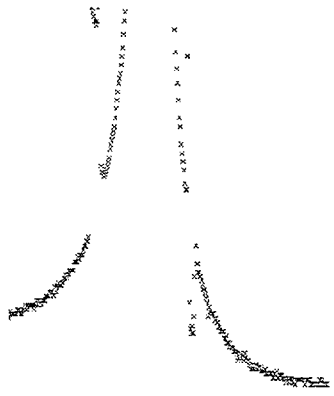
5.2.2 Eliminación de los nodos y ramas del mapa topo-geométrico que están aislados

Al obtener el mapa topo-geométrico local a partir del DVL se obtienen nodos y ramas que no poseen una unión física con el grafo principal.

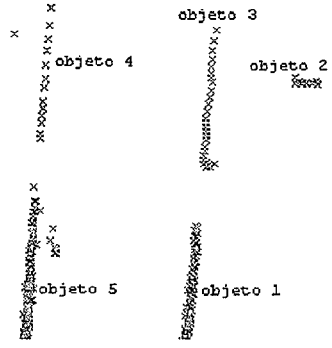
Definición 5.2.2: **Grafo principal.** Se denomina grafo principal al grafo mas cercano a la posición del robot.

El robot utilizará la información del grafo principal para planificar su trayectoria local. El resto de grafos se eliminan para reducir los tiempos de cálculo, y simplificar el mapa.

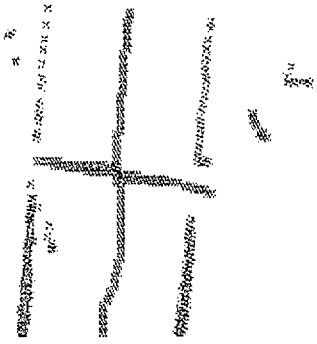
Se considera que el láser proporciona la información que se muestra en la figura 5.2(a). Esta información está dada en coordenadas polares y representa un pasillo con puertas abiertas a ambos lados (ver figura 5.2(b)). El DVL que se obtiene se representa en la figura 5.2(c). Los puntos que forman el nodo (puntos en color azul) son equidistantes a los objetos 1, 3, 4 y 5. El mapa topo-geométrico local que se obtiene directamente a partir del DVL se muestra en la figura 5.2(d). El mapa topo-geométrico local tiene un estado que corresponde al nodo del DVL y que representa una bifurcación. De este nodo salen 4 ramas, habiendo una 5ª rama aislada. Los puntos que forman esta rama aislada son equidistantes a los objetos 1 y 2. Las cuatro ramas tienen en sus extremos un nodo real (nodo que representa la bifurcación) y un nodo falso (extremos de las ramas). La rama aislada del mapa local está limitada en sus extremos por dos nodos ficticios. Esta rama no tiene una unión física con el *grafo principal* y no proporciona ninguna información útil al robot, por lo que es eliminada (ver figura 5.2(e)). En la figura 5.2(f) se representa el mapa topo-geométrico local, indicando los nodos reales (círculos negros), los nodos ficticios (círculos blancos) y las relaciones existentes entre ellos.



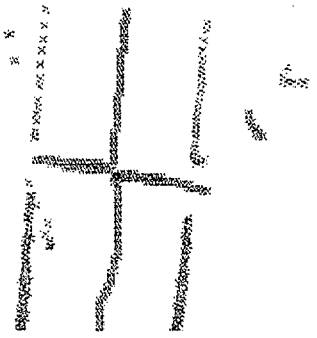
(a) Medidas del telémetro láser en coordenadas polares



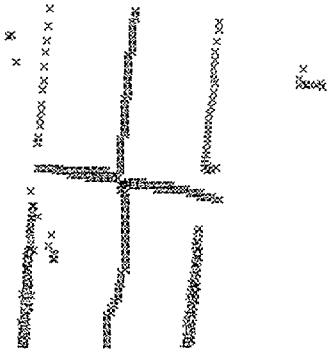
(b) Medidas del telémetro láser en coordenadas cartesianas agrupadas por objetos



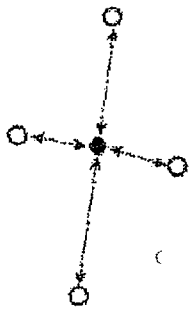
(c) DVL



(d) Mapa topo-geométrico



(e) Mapa topo-geométrico eliminando la rama aislada



(f) Mapa de estados

Fig. 5.2: Pasos para la obtención del mapa topo-geométrico de una intersección

5.2.2.1 Obtención de más estados en el mapa topo-geométrico

Además de los estados obtenidos directamente del DVL, se pueden obtener otros estados que representan lugares típicos de interiores, y que pueden ser útiles para la localización del robot.

Estos estados se obtienen estudiando el comportamiento de los puntos que forman cada rama. Se considera que una rama R_i está formada por un conjunto de puntos, tal que, $R_i = \{p_i, \cdots, p_n\}$ y donde p_i y p_n representan los puntos extremos de la rama. Cada punto tiene asociado una distancia, que representa la distancia del punto a los objetos a los que es equidistante, $\forall p_k \; \exists d_k$.

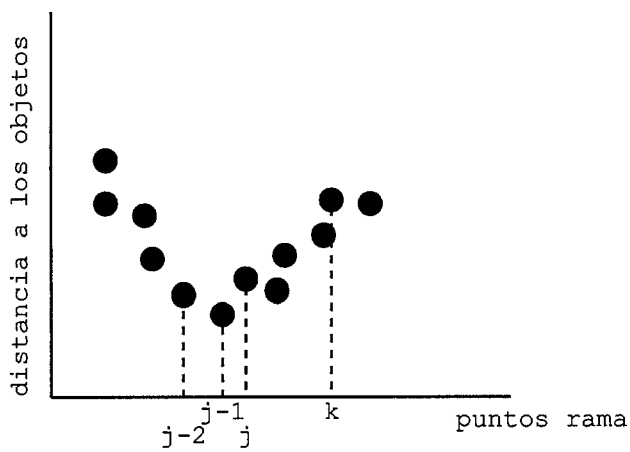


Fig. 5.3: Distancia de cada uno de los puntos que forman una rama a los objetos equidistantes

Si se estudia cómo varía la distancia que existe entre cada uno de los puntos que forman la rama, se observa que un cambio de pendiente en las distancias implica un nuevo estado:

$$signo(d_j - d_{j-1}) \neq signo(d_{j-1} - d_{j-2}), \quad i+2 \leq j \leq n \tag{5.1}$$

Cada vez que se observa un cambio en el signo de la pendiente se genera un nodo. Este nodo está situado en el punto donde se produce tal cambio:

$$N_j = \{p_j\} \tag{5.2}$$

Los datos del telémetro láser tienen errores que no permiten la aplicación del criterio anterior tal cual. Para obtener un nuevo estado hay que considerar

los errores en las medidas. Se considera que un punto p_j es un nuevo nodo, si además de cumplir la condición indicada en la ecuación 5.1, cumple con los requisitos siguientes (ver figura 5.3):

$$signo(d_j - d_{j-1}) = signo(d_k - d_{j-1}), \quad k > j \tag{5.3}$$

y

$$\|p_j - p_k\| > umbral_{\text{error medida}} \tag{5.4}$$

Los estados que se obtienen aplicando este criterio representan: entrada a una puerta, paso de un vestíbulo a un pasillo, paso de un pasillo a un vestíbulo.

En la figura 5.4(a) se muestra como varía la distancia a los objetos de los puntos de una rama cuando se pasa de un vestíbulo a un pasillo. El mapa de estados local obtenido directamente del DVL se muestra en la figura 5.4(b). Aplicando el criterio de obtención de un nuevo nodo estudiando el comportamiento de los puntos que forman la rama, se obtiene el punto p_j que es el punto donde se encuentra el nuevo estado (ver figura 5.4(c)). El mapa topo-geométrico local generado con el nuevo estado y que representa el paso de un vestíbulo a un pasillo se muestra en la figura 5.4(d).

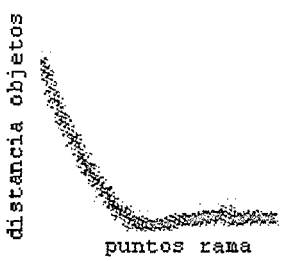
Otro caso de obtención de un nuevo estado, es el paso a través de una puerta. En la figura 5.5(a) se muestra como varía la distancia de cada uno de los puntos que forman la rama con los objetos. Se observa que existe claramente un mínimo donde la pendiente cambia. Este punto p_j (ver figura 5.5(b)) representa un estado que es la entrada por una puerta. En las figuras 5.5(c) y 5.5(d) se muestran el mapa topo-geométrico local obtenido directamente del DVL y el mapa topo-geométrico local con el nuevo estado respectivamente.

Si una rama está delimitada por uno o dos nodos ficticios, para reducir los errores en la obtención de los nuevos estados, primeramente se eliminan los datos alejados de cada una de las ramas. Se considera la rama $R_i = \{p_i, \cdots, p_n\}$, donde p_i es el punto más cercano al nodo real, o en el caso de una rama con 2 nodos ficticios, es el punto más cercano a la posición del robot. Si se cumple que:

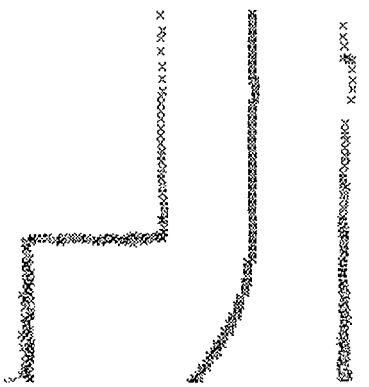
$$\|p_i - p_k\| > umbral_{\text{puntos separados}}, \quad k > i \tag{5.5}$$

entonces la rama R_i quedará formada por los puntos $R_i = \{p_i, \cdots, p_{k-1}\}$.

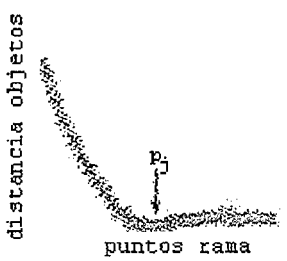
Si se considera el ejemplo de la figura 5.2 se observa que en 5.2(e) hay unos puntos más alejados de una de las ramas a la cual pertenecen. Si no se



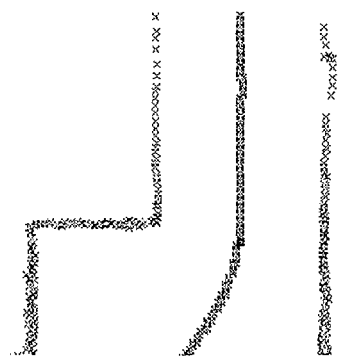
(a) Variación de la distancia de cada uno de los puntos que forman la rama del DVL a los objetos



(b) Mapa topo-geométrico obtenido directamente del Diagrama de Voronoi

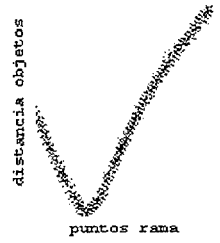


(c) Punto de la rama donde se encuentra el nuevo nodo

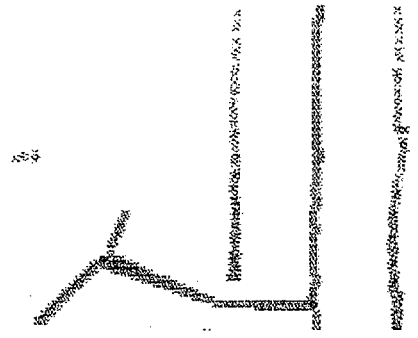


(d) Mapa topo-geométrico con el nuevo estado

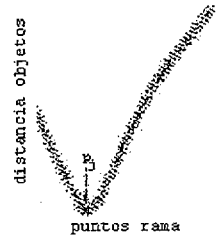
Fig. 5.4: Mapa topo-geométrico del paso de un hall a un pasillo



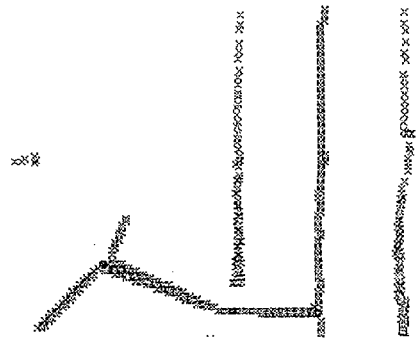
(a) Variación de la distancia de cada uno de los puntos de la rama del DVL que atraviesa la puerta a los objetos



(b) Mapa topo-geométrico obtenido directamente del DVL

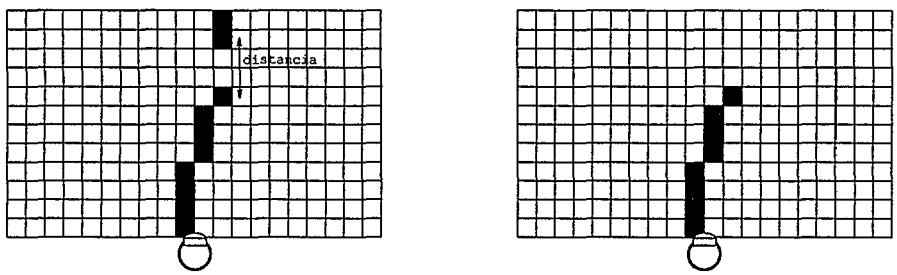


(c) Punto de la rama donde se encuentra el nuevo nodo



(d) Mapa topo-geométrico con el nuevo estado

Fig. 5.5: Mapa topo-geométrico de un pasillo con una puerta abierta a la izquierda



(a) Puntos que forman una rama del DVL

(b) Puntos que forman una rama del DVL después de aplicar el filtro de puntos alejados

Fig. 5.6: Eliminación de puntos de la rama que se encuentran alejados

eliminan estos puntos al realizar el estudio de la pendiente de las distancias se puede generar un nodo que no existe realmente. Si se eliminan los puntos más alejados de las ramas (ver figura 5.7(a)) y luego se aplica el algoritmo de obtención de más estados se genera un nuevo nodo en el mapa topológico local como se muestra en la figura 5.7(b).

En la figura 5.7(b) se observa que se ha generado un estado que corresponde al estrechamiento debido a que la puerta está abierta hacia el pasillo. En la figura 5.7(c) se representa el mapa topológico local que se obtiene con el nuevo estado.

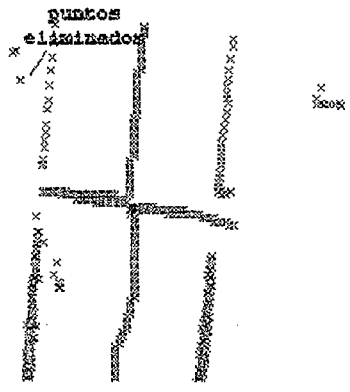
5.2.3 Información almacenada en los nodos y en las ramas

El mapa local construido es un mapa topológico que contiene nodos, que representan lugares característicos de entornos de interiores, y las relaciones existentes entre ellos, facilitando los procesos de planificación y navegación del robot en el entorno.

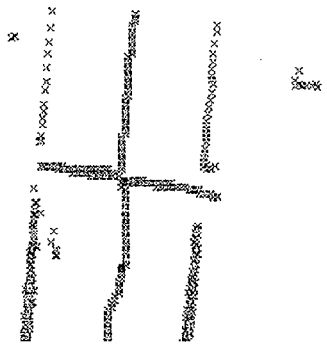
Los nodos y ramas del mapa topológico son enriquecidos con información geométrica, que facilita la localización del robot eliminando posibles ambigüedades que puedan aparecer.

La información almacenada en los nodos es:

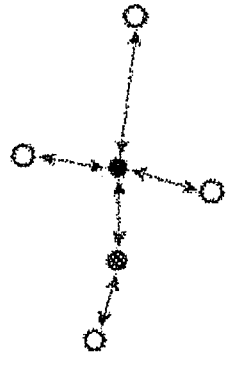
- Puntos en coordenadas cartesianas de cada nodo con respecto al sistema de coordenadas local situado en el sensor $p_1, \cdots, p_n$.



(a) Puntos de las ramas que se eliminan



(b) Mapa topo-geométrico con el estado que define el estrechamiento



(c) Mapa de estados

Fig. 5.7: Mapa topo-geométrico de un pasillo con 2 puertas abiertas a ambos lados y un estrechamiento

- Distancia de cada uno de los puntos anteriores a los objetos a los que son equidistantes $DC_1, \cdots, DC_m$.
- Objetos a los que cada nodo es equidistante CB_i.
- Identificador de cada nodo.

Las ramas contienen la siguiente información:

- Puntos en coordenadas cartesianas que forman cada rama con respecto al sistema de referencia local situado en el sensor $p_1, \cdots, p_m$.
- Distancia de cada uno de los puntos anteriores a los objetos a los que son equidistantes $DC_1, \cdots, DC_m$.
- Objetos a los que cada rama es equidistante CB_i.

6. CORRECCIÓN DE LA POSICIÓN DE UN ROBOT MÓVIL UTILIZANDO ALGORITMOS GENÉTICOS

6.1 Introducción

Cuando un robot móvil se desplaza por un entorno aparecen dos problemas fundamentales: el problema de modelado y el problema de localización. Estos surgen como consecuencia de la imprecisión de las medidas proporcionadas tanto por los sensores externos (ultrasonidos, láser, infrarrojos etc.), como por los de la odometría.

Estos dos problemas no son independientes sino que se interrelacionan: para que un robot móvil pueda generar un mapa del entorno es necesario que conozca la posición que ocupa con respecto a éste y viceversa, para que un robot pueda localizarse debe tener información del entorno.

En este capítulo se resuelve el problema de localización de un robot móvil autónomo utilizando Algoritmos Genéticos (AG) para estimar la posición del robot. Este proceso se realiza contrastando los mapas topo-geométricos locales obtenidos en instantes de tiempos diferentes.

Diversos trabajos como [Arm97] [Gar99] utilizan los AG para estimar la posición de robot. El primero de ellos utiliza un algoritmo genético simple, mientras que el segundo utiliza un filtro RGO (Restricted Genetic Optimization) no-lineal. Ambos métodos se basan en la detección de marcas artificiales utilizando una cámara y ultrasonidos. En esta tesis, la localización no se basa en el contraste de marcas artificiales, sino que se basa en el contraste de características naturales típicas de entornos de interiores, obtenidas a partir del DVL (ver capítulo 4).

En este trabajo se considera que el robot comienza a desplazarse sin un conocimiento previo ni del entorno por donde se desplaza ni de dónde

se encuentra. El objetivo de esta tesis es que el robot construya un mapa consistente del entorno a medida que se desplaza. En cada desplazamiento el robot adquiere información del entorno a través de los sensores externos, y tiene una primera estimación de lo que se ha desplazado a través de los sensores internos, odometría.

Inicialmente el robot se encuentra en una posición que desconoce, y que considera el origen de coordenadas del sistema de referencia global. La única información de que dispone es la información de los sensores externos, en este caso un telémetro láser. A partir de esta información construye un mapa topo-geométrico local según se ha indicado en el capítulo anterior.

Se considera que el robot se desplaza en incrementos pequeños de tal manera que el entorno no haya podido sufrir cambios significativos y que los errores de la odometría no sean muy grandes. En el primer desplazamiento del robot, la única información de que dispone para corregir su posición es la información de los sensores tanto internos como externos. De estos sensores se obtiene el mapa topo-geométrico generado en ese instante y la información odométrica respectivamente. La corrección de la posición se realiza contrastando el mapa obtenido en ese instante de tiempo con el mapa obtenido en el instante previo. La posición corregida del robot es aquella en la que los mapas contrastan mejor.

A medida que el robot avanza tiene mayor información del entorno con la que puede contrastar la información actual de los sensores para corregir su posición. Si el robot cierra un ciclo, es decir, vuelve a pasar por lugares que ya han sido observados, ya no realizará solamente una corrección incremental de su posición, corrección local, si no que ya se podrá localizar globalmente.

El algoritmo que se emplea para corregir la posición del robot, tanto localmente como globalmente, es un algoritmo que utiliza Algoritmos Genéticos (AG). La corrección de la posición del robot se podría haber realizado utilizando un método clásico de correlación, de tal manera, que éste proporcionara una medida del solapamiento existente entre los diversos mapas topo-geométricos locales obtenidos en instantes de tiempo diferentes para las distintas posiciones posibles del robot. El problema que presenta este método es que para entornos de grandes dimensiones supone un gran coste computacional, ya que, el número de soluciones posibles es muy elevado. Los AG permiten obtener un solución buena del problema para una muestra del conjunto de soluciones posibles consiguiéndose un reducción del tiempo de cálculo.

En las secciones siguientes se explican los fundamentos básicos de los algoritmos genéticos, y el algoritmo desarrollado para llevar a cabo la corrección

de la posición del robot.

6.2 Algoritmos genéticos

Los Algoritmos Genéticos (AG), introducidos por John Holland en los años setenta inspirándose en el proceso observado en la evolución natural de los seres vivos, se han convertido en una herramienta importante en aprendizaje y optimización. La característica principal de los AG es la *selección natural.* Para resolver el problema de aprendizaje, diseño u optimización, un AG mantiene una población de organismos (cadena de bits) y modifica probabilísticamente esta población, buscando una solución cercana a la óptima para la tarea dada (para más detalles se pueden consultar [Gol89] [Raw91]).

Un AG manipula una población de cadenas usando operadores probabilísticos similares a los genéticos, combinando pares de "cadenas buenas" (cruce) y modificando ciertos valores de estas cadenas (mutación), para producir una población actual con la intención de generar cadenas con valores altos de la función que se quiere optimizar. La evolución biológica no tiene memoria en el sentido de que en la formación de los cromosomas únicamente se considera la información del período anterior, la información histórica se basa en la acumulación de cadenas de cierto tipo.

Los algoritmos genéticos presentan muchas ventajas en relación a otros algoritmos más tradicionales:

- Optimizan los resultados tanto con parámetros continuos como con parámetros discretos.
- No requieren información sobre la derivada.
- Realizan la búsqueda mediante toda una generación de objetos, no buscan un único elemento.
- Resuelven el problema con gran cantidad de parámetros.
- Optimizan parámetros con superficies de coste bastante complejas.
- Pueden codificar los parámetros de tal manera que la optimización es realizada con parámetros codificados.
- Trabajan con datos generados numéricamente, datos experimentales o funciones analíticas.

Los inconvenientes principales son:

- No hay ninguna teoría matemática que garantice la convergencia.
- El número de operaciones y por tanto el coste computacional puede ser muy elevado si no se toman precauciones.

Trabajos realizados en el Área de Ingeniería de Sistemas y Automática de la universidad Carlos III de Madrid relacionados con los algoritmos genéticos son: en [ArmMorEsc99] se utilizan los AG para localizar un robot móvil, en [GarMorSal99] se utilizan los AG para estimar los parámetros de un sistema, en [GarMorSal00] se utilizan los AG para el control de sistemas y en [MatArmEsc01] se utilizan los AG para el reconocimiento de patrones.

6.2.1 Espacio de búsqueda

Si se quiere resolver un problema, generalmente lo que se hace es buscar alguna solución que sea la mejor entre varias. El espacio de soluciones posibles se llama *espacio de búsqueda*, o también espacio de soluciones. Cada punto en el espacio de búsqueda representa una solución posible. Cada solución puede ser "marcada" por el valor de su salud. La solución es por tanto un punto, o puntos, entre las soluciones posibles pertenecientes a este espacio de búsqueda. Buscar una solución es igual a buscar un punto perteneciente a este espacio de búsqueda que tenga el valor de salud mayor.

En el caso de la localización de un robot móvil autónomo, lo que interesa es estimar la posición del robot (x, y, θ) dentro del entorno por donde se desplaza. Cuando un robot se desplaza de una posición a otra, su incertidumbre sobre su localización crece con respecto a la posición inicial. De tal manera que, el espacio de soluciones es el espacio del entorno dónde posiblemente se puede encontrar el robot, teniendo en cuenta los errores odométricos que se producen en el desplazamiento del robot. Este *espacio de búsqueda* es la incertidumbre espacial calculada según las ecuaciones del Filtro de Kalman sobre la estimación de la posición del robot. Estas ecuaciones describen como varía la posición del robot en función de las entradas de control introducidas a los motores, considerando una estimación de los errores cometidos en la odometría. Las ecuaciones del filtro de Kalman permiten estimar de una manera sencilla la posición del robot en el instante $t+1$ a partir de un conjunto de medidas de odometría obtenidas en instantes previos considerando que éstas están sometidas a ruidos o imprecisiones.

La restricción del espacio de búsqueda tiene como objetivo mejorar el rendimiento del AG, ya que, reduce el espacio de soluciones posibles.

La mejor posición del robot estimada es aquella que se encuentra dentro de este *espacio de búsqueda* para la que dos mapas topo-geométricos locales hallados en instantes de tiempo diferentes tienen un mejor contraste.

En las secciones siguientes se indica como se calcula la incertidumbre espacial a medida que el robot se desplaza por un entorno. Esta incertidumbre espacial es el *espacio de búsqueda* de las soluciones posibles de la posición del robot. Es en esta zona dónde el AG buscará las mejores soluciones.

6.2.1.1 Determinación de la zona de búsqueda de soluciones. Incertidumbre espacial

En numerosas aplicaciones de robótica, como por ejemplo la automatización industrial y movimiento autónomo, es necesario representar la incertidumbre espacial.

Numerosos métodos propuestos representan esta incertidumbre como un valor mínimo-máximo calculado sobre el error [Bro82] [Tay76]. Estos métodos min-max son muy conservativos comparados con los métodos probabilísticos, debido a que ellos utilizan mucha información, cada una con el caso peor de error cometido. Los métodos probabilísticos para representar la incertidumbre han sido utilizados por Chatila [ChaLau85] y Smith et al. [SmiChe86] [SmiSelChe88]. La diferencia entre ellos es que Chatila et al. [ChaLau85] utilizan una representación escalar mientras que Smith et al. [SmiChe86] [SmiSelChe88] utilizan una representación multivariable. En esta tesis se utiliza este último método para representar la incertidumbre.

A continuación se explica el modelo cinemático que se emplea para definir el movimiento del robot y calcular a partir de él la incertidumbre espacial. Esta incertidumbre, como ya se ha comentado, es al *espacio de búsqueda* de las soluciones posibles de la posición del robot. El AG buscará la mejor solución dentro de este *espacio de búsqueda.*

6.2.1.1.1 Modelo cinemático Se supone que la posición y orientación del robot en un instante de tiempo t es representada por el vector de estados X_t:

$$X_t = \begin{pmatrix} x_t \\ y_t \\ \theta_t \end{pmatrix} \tag{6.1}$$

donde x e y son las coordenadas de un punto del espacio cartesiano y θ es la rotación con respecto al eje z.

El modelo cinemático describe cuál es la nueva posición del robot cuando se introduce una entrada de control $u_t = (\Delta x, \Delta y, \Delta\theta)^T$:

$$X_{t+1} = f(X_t, u_t, t) \tag{6.2}$$

donde $f(X_t, u_t, t)$ es una función no lineal de transición de estado.

En la ecuación 6.2 no se consideran los errores en los datos odométricos. Si se considera una estimación de los errores cometidos por la odometría, el modelo cinemático viene definido por la ecuación siguiente:

$$X_{t+1} = f(X_t, u_t, t) + v_t \tag{6.3}$$

donde v_t es el ruido que se asume blanco, gaussiano, con media cero y covarianza conocida $v_t \rightarrow N(0, Q_t)$ [Arm97].

El modelo usado está basado en el modelo de la cinemática del punto dado en [SmiChe86]. La entrada de control $u_t = [T(t), \Delta\theta(t)]^T$ está formada por una traslación $T(t)$ y una rotación en el sentido antihorario $\Delta\theta(t)$. La función de transición de estado $f(X_t, u_t, t)$ tiene la forma:

$$f(X_t, u_t, t) = \begin{pmatrix} x(t) + T(t) \cdot \cos(\theta(t)) \\ y(t) + T(t) \cdot \sin(\theta(t)) \\ \theta(t) + \Delta\theta(t) \end{pmatrix} \tag{6.4}$$

Existen otros modelos cinemáticos mucho más sofisticados, pero se ha decido utilizar el modelo de la cinemática del punto porque es un modelo sencillo y además ha proporcionado buenos resultados en los experimentos llevados a cabo.

6.2.1.1.2 Cálculo de la covarianza Utilizando el modelo cinemático y la entrada de control se puede predecir la posición del robot en el instante $t+1$ utilizando la ecuación 6.3.:

$$\widehat{X}(t+1|t) = f(\widehat{X}(t|t), u_t) \tag{6.5}$$

La matriz de covarianzas asociada a la predicción se obtiene como:

$$Cov\,(t+1|t) = J\,(t) \cdot Cov\,(t|t) \cdot J\,(t)^T + Q\,(t) \tag{6.6}$$

donde J es el jacobiano de la función de transición de estados $f(\widehat{X}(t|t), u_t)$ obtenido linealizando alrededor del estado estimado $\widehat{X}(t|t)$:

$$J(t) = \Delta f = \begin{pmatrix} 1 & 0 & -T(t) \cdot \sin(\widehat{\theta}(t|t)) \\ 0 & 1 & T(t) \cdot \cos(\widehat{\theta}(t|t)) \\ 0 & 0 & 1 \end{pmatrix} \tag{6.7}$$

y Q es el error. Una aproximación del error puede ser:

$$Q(t) = \begin{pmatrix} (fabs(x_{t+1} - x_t)) & 0 & 0 \\ 0 & (fabs(y_{t+1} - y_t)) & 0 \\ 0 & 0 & (fabs(\theta_{t+1} - \theta_t)) \end{pmatrix} \cdot K_{error} \tag{6.8}$$

donde K_{error} es la matriz de ganancia del error.

En el instante inicial $t = 0$ se considera que el error es 0, debido a que se toma este punto como origen del sistema de referencia global:

$$Cov\,(0|0) = Cov\,(0) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{6.9}$$

6.2.1.2 Representación de la incertidumbre

La incertidumbre espacial se puede representar por una distribución de probabilidad sobre las variables espaciales, $X = (x, y, \theta)^T$. Por ejemplo, una función de densidad de probabilidad que asigna una probabilidad a cada combinación de las variables espaciales, X, puede ser:

$$P(X) = f(X)dX \tag{6.10}$$

El conocimiento de la distribución de probabilidad no es necesario, siempre que el robot sea capaz de llevar a cabo su tarea. La mayoría de los dispositivos proporcionan solamente un valor nominal de la medida, pudiéndose estimar el error medio del error a partir de las especificaciones del sensor. Smith et al. [SmiSelChe88] proponen un modelo para la incertidumbre espacial, estimando los dos primeros momentos de la distribución espacial. La media, $\widehat{X}$, y la covarianza, $Cov(X)$, son definas como:

$$\begin{aligned} \widehat{X} &\triangleq E(X) \\ \widehat{X} &\triangleq X - \widehat{X} \\ Cov(X) &= E(\widehat{X} - \widehat{X}^T) \end{aligned} \tag{6.11}$$

donde E es la esperanza, y $\widehat{X}$ es la desviación de la media. Para un robot

móvil, estos valores son:

$$\widehat{X} = \begin{bmatrix} \widehat{x} \\ \widehat{y} \\ \widehat{\theta} \end{bmatrix}$$
$$Cov(X) = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{x\theta} \\ \sigma_{xy} & \sigma_y^2 & \sigma_{y\theta} \\ \sigma_{x\theta} & \sigma_{y\theta} & \sigma_\theta^2 \end{bmatrix} \tag{6.12}$$

Los elementos de la diagonal de la matriz de covarianzas son las varianzas de las variables espaciales, mientras que los elementos fuera de la diagonal son las covarianzas de las variables espaciales.

Para determinar con que probabilidad se encuentra el robot en un determinado punto del espacio es necesario considerar una distribución en particular. Al igual que en otros trabajos, como en [SmiChe86], en esta tesis se considera un distribución Gaussiana para estimar la probabilidad, ya que, es un método simple y rápido. Se ha demostrado experimentalmente que este método trabaja bastante bien, sobre todo si la contribución del error es pequeña. En este trabajo el robot se desplaza en incrementos pequeños para adquirir información del entorno y construir el mapa. Por tanto, el error de odometria entre una posición y otra no es muy grande.

Dando solamente la media, $\widehat{X}$, y la matriz de covarianzas, $Cov(X)$, de una distribución de probabilidad multivariable, el *principio de máxima entropía* indica que la distribución, que considera la información mínima, es la distribución normal. De este modo, si la relación espacial es calculada combinando informaciones diferentes, el *teorema central del límite* indica que la distribución resultante tenderá a la distribución normal:

$$P(X) = \frac{\exp[-\frac{1}{2}(X-\widehat{X})^T Cov^{-1}(X-\widehat{X})]}{\sqrt{(2\pi)^m \, |Cov(X)|}} dX \tag{6.13}$$

donde m es la dimensión de la variable espacial, que en el caso de un robot móvil es igual a 3: (x, y, θ).

Si se representa la incertidumbre espacial dibujando el contorno de la probabilidad constante de una distribución espacial con los valores de la media y covarianza dadas, estos contornos son elipses concéntricas (elipse para dos dimensiones), cuyos parámetros pueden ser calculados a partir de la matriz de covarianzas, $Cov(X)$ [SmiChe86].

6.2.1.2.1 Parámetros de la elipse Para la toma de decisiones es necesario determinar el contorno equiprobable (elipses o elipsoides) de una distribución Gaussiana multivariable definida por una media $\widehat{X}$ y una matriz de covarianzas $Cov(X)$. Estas elipses pueden ser usadas para determinar que un vector dado se encuentre dentro de la elipse o elipsoide con una probabilidad de por ejemplo el 90%.

La fórmula de una elipsoide es:

$$\left(X - \widehat{X}\right)^T Cov(X)^{-1} \left(X - \widehat{X}\right) = k^2 \tag{6.14}$$

donde X es un punto de la elipsoide y k es una constante elegida para un umbral de probabilidad, es decir, estas elipsoides pueden ser usadas para determinar la probabilidad de que un determinado vector este dentro de una elipsoide con una probabilidad determinada. Al término $(X-\widehat{X})^T\, Cov(X)^{-1}$ $(X - \widehat{X})$ se le denomina distancia de Mahalanobis, y representa la distancia estandarizada entre un vector de variables independientes X y su vector de medias.

La distribución gaussiana tiene la propiedad de que al cortar por planos paralelos al plano (x, y) se obtienen curvas de nivel que son elipses. En el caso particular de dos dimensiones (x, y), la elipse se calcula eliminando de la matriz de covarianzas en $3D(x, y, \theta)$, la fila y columna de la variable no deseada, en este caso θ. Esto es:

$$Cov_3(3x3) \Rightarrow Cov_2(2x2)$$

donde:

$$Cov(2x2) = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{xy} & \sigma_y^2 \end{bmatrix} \tag{6.15}$$

La correspondiente familia de elipses 2D, dada por la ecuación 6.14, viene definida por la expresión:

$$Ax^2 + Bxy + Cy^2 - k^2 = 0 \tag{6.16}$$

donde las variables A, B y C son determinadas a partir de la matriz de covarianzas 2D y de la ecuación 6.14. Si se llama a $InvCov = Cov(2x2)^{-1}$

entonces:

$$\begin{aligned} A &= InvCov\,(0,0) \\ B &= \frac{InvCov(1,0)+InvCov(0,1)}{2} \\ C &= InvCov\,(1,1) \end{aligned} \tag{6.17}$$

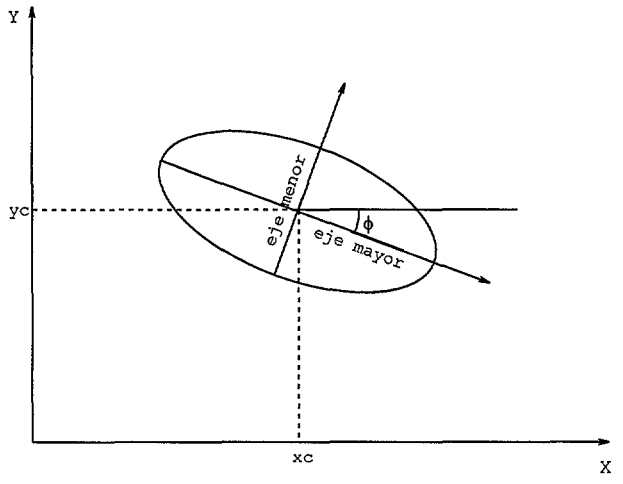


Fig. 6.1: Parámetros de una elipse

El ángulo ϕ, que forma el eje mayor de la elipse con el eje x positivo, viene dado por la expresión:

$$\phi = \frac{1}{2}\arctan\left(\frac{2B}{A-C}\right),\ \phi\left[\frac{-\pi}{2},\frac{\pi}{2}\right] \tag{6.18}$$

Los valores de los ejes mayor y menor que definen a la elipse son:

$$\begin{aligned} \text{semieje mayor} &= \sqrt{\frac{2k^2}{A+C-T}} \\ \text{semieje menor} &= \sqrt{\frac{2k^2}{A+C+T}} \end{aligned} \tag{6.19}$$

con:

$$T = \sqrt{A^2+C^2-2AC+4B^2} \tag{6.20}$$

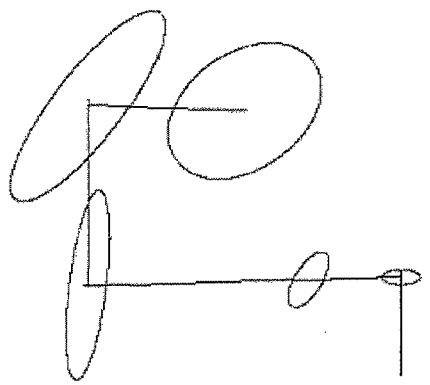


Fig. 6.2: Elipses de incertidumbres para distintos desplazamientos del robot

El valor de k se obtiene a partir de la distribución de probabilidad (ecu. 6.13), es decir la probabilidad de que un punto (x, y) pertenezca a la elipse con un determinado grado de incertidumbre:

$$P\left(x, y \in elipse\right) = 1 - e^{\frac{-k^2}{2}} \tag{6.21}$$

Despejando k se obtiene:

$$k^2 = -2\ln(1 - P) \tag{6.22}$$

En la tabla tabla 6.1 se muestran distintos valores de k para distintos valores de probabilidad. El valor de probabilidad que se usa en este trabajo es del 95%.

Probabilidad %	k^2
50	1.386
90	4.605
95	5.991

Tab. 6.1: Valores de k para distintos valores de probabilidad

6.2.1.3 Rectángulo de incertidumbre

Cuando un robot se desplaza por un entorno el conocimiento de su posición y orientación relativa es importante. Esta información es necesaria para

determinar si una meta especificada ha sido alcanzada, o para conocer la posición del robot móvil en un camino predefinido.

La utilización de la información odométrica es un método simple que integra la traslación y rotación de las ruedas del robot para determinar la posición en coordenadas cartesianas del robot. Sin embargo, debido al deslizamiento de las ruedas y a los errores propios de los sensores odométricos, los encoders no indican de manera muy precisa la posición del robot. Esto implica que a medida que el robot se desplaza, la incertidumbre sobre su posición y orientación con respecto a una posición inicial aumenta. Por tanto, se tiene una incertidumbre en traslación (x, y) y en orientación (θ). Esta incertidumbre es *el espacio de búsqueda* del AG.

En la figura 6.2 se muestra como varía la incertidumbre a medida que el robot se desplaza suponiendo que parte de la posición inicial $(0, 0, 0)$. En esta imagen sólo se muestra la incertidumbre en 2 dimensiones, es decir, que no está representada la incertidumbre en orientación.

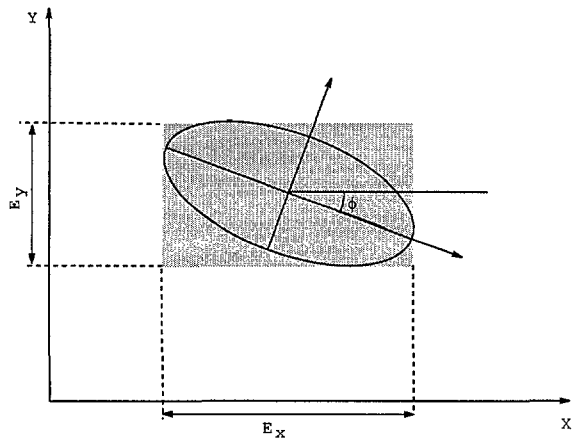
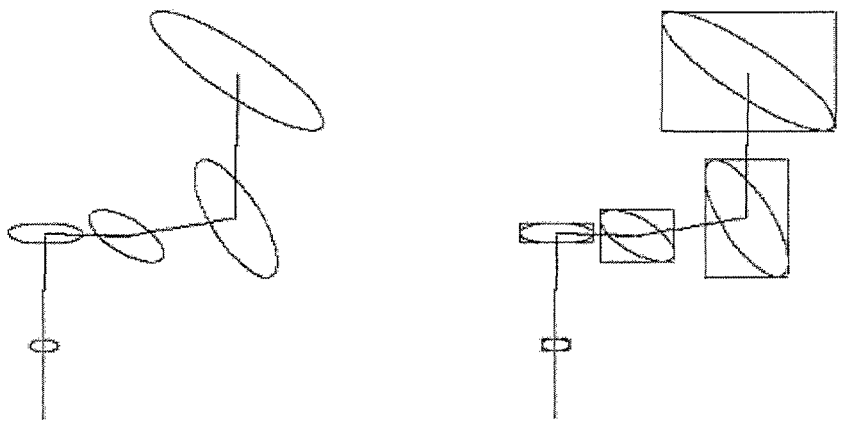


Fig. 6.3: Rectángulo de incertidumbre

Para simplificar el tiempo de cálculo se considera que la incertidumbre espacial no es una elipse sino un rectángulo que delimita la elipse (ver figura 6.3). Al considerar un rectángulo en lugar de una elipse el espacio de búsqueda aumenta, incrementándose también la probabilidad de encontrar la solución en esta zona de búsqueda.

Definición 6.2.1: **Rectángulo de incertidumbre.** Se define como rectángulo de incertidumbre al rectángulo que delimita a la elipse de incertidumbre,

y que representa el *espacio de búsqueda* del AG donde se encuentra las soluciones posibles.



(a) Elipse de incertidumbre (b) Rectángulo de incertidumbre

Fig. 6.4: Incertidumbre espacial a medida que un robot móvil se desplaza por un entorno

Del rectángulo de incertidumbre se obtienen los valores de incertidumbre en traslación: E_x y E_y (ver figura 6.3). La incertidumbre en rotación se obtiene de la matriz de covarianzas (ecu. 6.6):

$$E_\theta = \sqrt{Cov(3,3)} \tag{6.23}$$

Como se ha considerado una distribución normal, esta incertidumbre representa aproximadamente la zona del espacio (campana de Gauss) donde es probable que se encuentre θ con una probabilidad del 68.27%. Si se hubiera considerado que $E_\theta = 3 \cdot \sqrt{Cov(3,3)}$ la probabilidad hubiera sido del 99.73%.

Definición 6.2.2: **Incertidumbre espacial** ($\mathrm{E} = \{E_x, E_y, E_\theta\}$). Se define como incertidumbre espacial, y se denota por $\mathrm{E} = \{E_x, E_y, E_\theta\}$, al *espacio de búsqueda* del AG dónde es posible encontrar la mejor solución de la posición del robot.

El centro de la incertidumbre espacial E es la posición del robot calculada utilizando la información odométrica X_{t+1} (ecu. 6.3), es decir, la posición en

que se supone se encuentra el robot considerando únicamente la información proporcionada por los sensores internos (odometría).

En esta sección se ha definido y se ha indicado cómo se calcula la zona del espacio donde el AG buscará la mejor solución. A continuación se describen brevemente los fundamentos y parámetros de los AG.

6.2.2 Un algoritmo genético simple

El mecanismo de un algoritmo genético simple es muy sencillo ya que no implica más que una copia e intercambio parcial de cadenas. La búsqueda de una población reproductiva comienza con alguna población de cadenas realizándose el ciclo repetidamente hasta que alguna condición es satisfecha (ver figura 6.5).

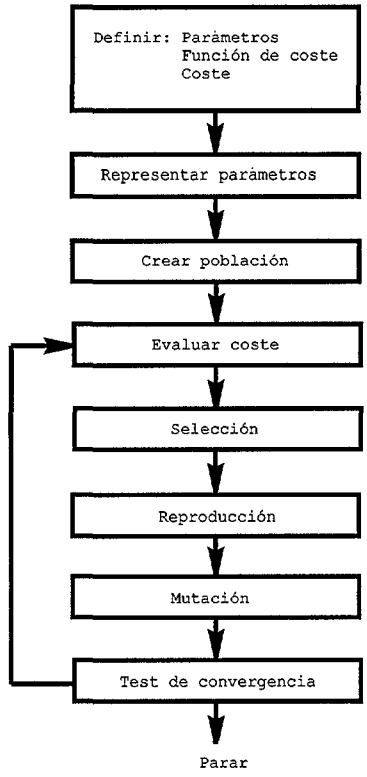


Fig. 6.5: Diagrama de flujo para un AG binario simple

Los pasos del Algoritmo Genético son:

1. **Evaluación**: se evalúa cada cadena de la población.
2. **Selección**: dependiendo de los valores de las cadenas, se selecciona un subconjunto denominado *conjunto de reproducción* de la población.
3. **Reproducción**: de cada *conjunto de reproducción*, se generan nuevas cadenas usando varias estrategias de reproducción, como son el cruce y la mutación.
4. **Reemplazo**: finalmente, se reemplazan algunas o todas las cadenas de la población inicial con nuevas cadenas.

6.2.3 Parámetros de un algoritmo genético

Una decisión también importante a la hora de implementar un AG, es que valores seleccionar para varios parámetros, como son la probabilidad de cruce o mutación, tamaño de la población y longitud del cromosoma. Estos parámetros interactúan unos con otros, así que no se pueden optimizar separadamente.

6.2.3.1 Tamaño de la población

El tamaño de población indica cuántos cromosomas hay en una población (en una generación). Si hay pocos cromosomas es muy probable que el AG tenga pocas probabilidades de dar un resultado bueno, ya que, solamente se considera una parte pequeña del espacio de búsqueda. Pero si se tiene un número elevado, el AG se hace muy lento.

6.2.3.2 Número de cromosomas por individuo

El número de cromosomas por individuo es el número de variables que se quiere manejar. En este caso, lo que interesa es determinar la posición y orientación del robot en $\mathbb{R}^2$. El número de cromosomas por individuo es 3: (x, y, θ).

En la generación inicial se crea un número de posiciones aleatorias que se encuentran dentro del espacio de búsqueda. En esta generación inicial también se añaden las posiciones estimadas utilizando las mejores soluciones de las posiciones halladas en el instante anterior (X_t), y utilizando la información odométrica obtenida en el instante actual (ver ecu. 6.3):

$$\{(x_{t+1,1}, y_{t+1,1}, \theta_{t+1,1}), \cdots, (x_{t+1,i}, y_{t+1,i}, \theta_{t+1,i}), (x_{i+1}, y_{i+1}, \theta_{i+1}), \cdots, (x_n, y_n, \theta_n)\} \tag{6.24}$$

donde $(x_{t+1,k}, y_{t+1,k}, \theta_{t+1,k})$ son las posiciones estimadas, y el resto son las posiciones aleatorias válidas, es decir, que se encuentran dentro del espacio de búsqueda.

6.2.3.3 Longitud del cromosoma

La longitud del cromosoma es el número de bits, si se ha realizado una codificación binaria, que son necesarios para codificar adecuadamente las variables. La longitud del cromosoma es un parámetro a tener en cuenta para el buen funcionamiento del AG. Una longitud pequeña implica que la diferencia numérica entre un cromosoma y otro es elevada y sólo se tienen en cuenta pequeñas porciones del espacio de búsqueda. Si embargo, si el número es elevado el AG se ralentiza.

Si se considera l a la longitud del cromosoma, $valor_{max}$ y $valor_{min}$ a los valores máximos y mínimos del espacio de búsqueda, entonces el incremento entre los valores posibles que pueden tomar los cromosomas vendrá dado por la siguiente expresión:

$$\text{resolución} = \frac{valor_{max} - valor_{min}}{2^l - 1} \tag{6.25}$$

Si se supone que el espacio de búsqueda es el campo de los números enteros comprendidos entre $[-30, 50]$, los valores de los incrementos para diferentes valores de la longitud de cromosoma se muestran en la tabla 6.2. Se observa que ha medida que aumenta la longitud del cromosoma, disminuye el valor de la resolución y aumenta el número de candidatos posibles dentro del espacio de búsqueda.

Longitud cromosoma (número de bits)	**resolución**
2	26.67
4	5.33
6	1.27
8	0.31

Tab. 6.2: Incremento entre los valores de los cromosomas para distintas longitudes de cromosoma

A medida que el robot avanza su incertidumbre aumenta, incrementándose el espacio de búsqueda de las soluciones posibles. Si se mantiene fija la longitud del cromosoma, implica que en los primeros desplazamientos se

tiene más precisión en las soluciones que en los desplazamientos finales. Para resolver este problema, y hacer que la precisión sea igual en cualquier instante de tiempo, se ha implementado un AG en el que la longitud del cromosoma es variable.

De la ecuación 6.25 se puede despejar el valor de la longitud de cromosoma:

$$l = \frac{\log\left(\frac{valor_{\max} - valor_{\min}}{\text{resolución}} + 1\right)}{\log(2)} \tag{6.26}$$

Para cada uno de los elementos que forman un individuo se calcula una longitud:

$$\begin{aligned} l_x &= \frac{\log\left(\frac{2 \cdot incertidumbre_x}{\text{resolución(x)}} + 1\right)}{\log(2)} \\ l_y &= \frac{\log\left(\frac{2 \cdot incertidumbre_y}{\text{resolución(y)}} + 1\right)}{\log(2)} \\ l_\theta &= \frac{\log\left(\frac{2 \cdot incertidumbre_\theta}{\text{resolución}(\theta)} + 1\right)}{\log(2)} \end{aligned} \tag{6.27}$$

La longitud final del cromosoma es:

$$l = max\{l_x, l_y, l_\theta\} \tag{6.28}$$

Experimentalmente se han seleccionado los valores siguientes de resolución:

$$\begin{aligned} &\text{resolución}(x) = 15cm \\ &\text{resolución}(y) = 15cm \\ &\text{resolución}(\theta) = 2^o \end{aligned} \tag{6.29}$$

Aunque a primera vista parezca que estos valores de resolución no son muy buenos hay considerar que se está trabajando en entornos de dimensiones de varios metros, es decir, entornos de grandes dimensiones. Conseguir una precisión de $15cm$ y 2^o en un entorno de alrededor de más de $30m$ es bastante aceptable. También se puede mejorar la precisión de los resultados una vez que se ha obtenido una solución buena del problema.

Hay que considerar que cuanto más precisión se requiere al problema más lento se hace el algoritmo, por tanto, hay que llegar a un compromiso entre rapidez de cálculo del AG y precisión de las soluciones.

6.2.3.4 Probabilidad de cruce

La probabilidad de cruce indica con qué frecuencia se realiza la operación de cruce. Si no hay cruce, los descendientes son una copia exacta de los padres. Si hay cruce los descendientes se forman a partir de los padres.

El cruce se realiza esperando que los nuevos cromosomas tengan las mejores partes de los cromosomas viejos, y que quizá, estos sean mejores. De todas maneras, es conveniente dejar que los mejores cromosomas, sobrevivan en la generación siguiente.

6.2.3.5 Probabilidad de mutación

La probabilidad de mutación indica con qué frecuencia son cambiadas partes de un cromosoma. Si no hay mutación, los descendientes no son modificados, es decir, son una copia de los descendientes procedentes del cruce. Si la probabilidad de mutación es del 100%, todos los cromosomas son cambiados.

6.2.4 Operaciones de un algoritmo genético

6.2.4.1 Codificación

El cromosoma debería de alguna manera contener información sobre la solución que representa. Codificar los cromosomas es uno de los problemas que se plantean a la hora de resolver un AG. Este proceso de codificación depende del problema. Tipos de codificación existentes son: codificación binaria, codificación por permutación, codificación de valores, etc.. Para más detalles ver [Mit96].

El método de codificación empleado en esta tesis es el de codificación binaria. Se ha utilizado este método porque es muy sencillo de implementar. A continuación se explicará brevemente este método.

6.2.4.1.1 Codificación binaria La codificación binaria es el método que más se utiliza y es el que se emplea en esta tesis. En la codificación binaria cada cromosoma es una cadena de bits, 0 o 1. Cada bit en la cadena puede representar alguna característica de la solución. O la cadena completa puede representar un número. En la tabla 6.3 se muestran valores que pueden tomar los cromosomas en codificación binaria.

La codificación binaria proporciona un número alto de cromosomas incluso con un número pequeño de alelos.

Cromosoma 1	1	1	0	1	0	0
Cromosoma 2	0	1	0	1	1	0

Tab. 6.3: Ejemplo de codificación binaria

6.2.4.2 Selección

Los cromosomas generados son seleccionados de una población que son los padres que se van a cruzar para dar lugar a nuevos descendientes. El problema es como seleccionar estos cromosomas.

La selección es un proceso en el que cada cadena individual es copiada de acuerdo a los valores de la función objetivo *f*. En biología a esta función se le denomina función de salud (*fitness*). Esta función *f* mide la utilidad, beneficio o bondad de lo que se quiere maximizar. Copiar cadenas de acuerdo a los valores de su salud implica que las cadenas con un alto valor tienen mayor probabilidad de contribuir en uno o más descendientes en la próxima generación. Este operador es una versión artificial de la selección natural. En poblaciones naturales la salud se determina por la habilidad de las criaturas a sobrevivir a los depredadores, a causas de muerte natural u otros obstáculos, para llegar a ser adultos y consecuentemente reproducirse.

La selección puede ser implementada mediante algoritmos diferentes. Quizá el más sencillo es utilizar una ruleta fragmentada en varias regiones donde cada cadena de la población tiene un tamaño de región asignado que depende del valor de la salud que posea.

Algunos métodos de selección son: método de la ruleta, selección por rango, selección por estado preparado y elitismo entre otros [Mit96].

Los métodos de selección empleados en esta tesis son el método de la ruleta y elitismo que se describirán brevemente a continuación.

6.2.4.2.1 Método de la ruleta Los padres son seleccionados de acuerdo con su salud. Los mejores cromosomas son los que tienen mayor oportunidad para ser seleccionados. A cada individuo se le asigna una porción en una ruleta circular, el tamaño de esta porción es proporcional a la salud de cada individuo. La ruleta gira N veces, donde N es el número de individuos de la población. En cada giro de ruleta, el individuo marcado en ella es seleccionado para estar dentro del conjunto de padres de la próxima generación.

Se considera una población de 4 individuos, como se muestra en la tabla 6.4, donde se representa el valor de salud de cada cromosoma y su probabi-

lidad de selección, que no es más que el cociente entre la salud y el total de esta. Los cromosomas que tienen un valor de salud mayor son los que tienen una probabilidad más alta de selección.

N° cromosoma	Salud asociada	Probabilidad selección %
1	150	24.4
2	15	2.5
3	360	58.5
4	90	14.6
Total	615	100

Tab. 6.4: Cromosomas de una población y su salud asociada considerando el método de selección de ruleta

6.2.4.2.2 Elitismo Cuando se crea una nueva población por cruce o mutación, se tiene una gran probabilidad de que se pierda el mejor cromosoma. Este método fuerza al AG a retener algún número de los mejores individuos en cada generación. El resto de operaciones del AG se realizan de la manera tradicional.

El método de elitismo puede mejorar rápidamente el rendimiento del AG, debido a que previene la pérdida de la mejor solución encontrada.

6.2.4.3 Cruce

Después de la selección, se realiza un cruce. Este proceso se realiza en dos pasos. Primeramente, los miembros de las cadenas nuevas generadas son emparejados aleatoriamente. En segundo lugar, cada par de cadenas son cruzadas de la manera siguiente: una posición intermedia y entera k de la cadena es seleccionada aleatoriamente entre los valores de $[1, l-1]$. Las dos nuevas cadenas son creadas mezclando todos los caracteres entre las posiciones $k+1$ y l.

El mecanismo de cruce está representado en la figura 6.6 para una codificación binaria.

Los mecanismos de reproducción y cruce son muy simples, ya que implican la generación de números aleatorios, copia de cadenas y algún intercambio parcial entre ellas.

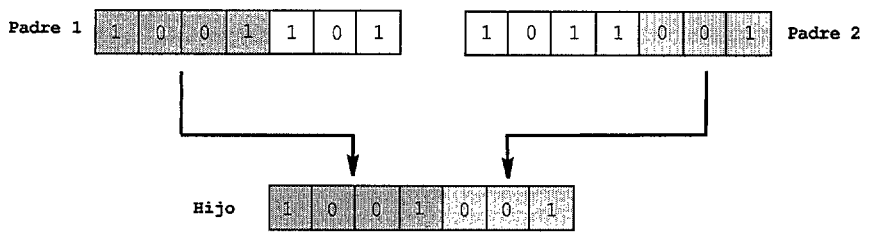


Fig. 6.6: Algoritmos genéticos: cruce

6.2.4.4 Mutación

Si no se selecciona un método adecuado, el algoritmo genético puede converger demasiado rápido dentro de una región de la función de coste. Si en este área está el mínimo global, esto es bueno. Sin embargo, hay muchas funciones que tienen muchos mínimos locales y si no se hace nada para evitar que el algoritmo converja rápidamente a este mínimo, puede acabar en un mínimo local en vez de en el global. Para evitar este problema de convergencia rápida, se puede forzar a que el algoritmo explore otras áreas de la función de coste introduciendo cambios aleatorios, o mutaciones, en algunos de los parámetros. En el algoritmo genético binario, este proceso se lleva a cabo intercambiando un bit de 0 a 1 o viceversa (ver figura 6.7).

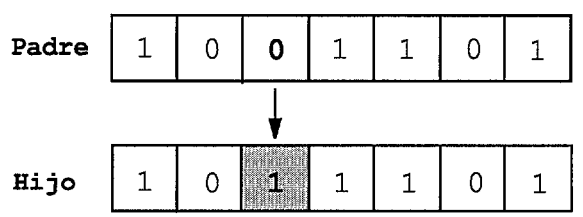


Fig. 6.7: Algoritmos genéticos: mutación

Los algoritmos genéticos binarios con un rango de mutación comprendido entre 1% y 20% trabajan bien frecuentemente. Si el rango de mutación está por encima del 20%, muchos parámetros buenos son mutados y el algoritmo se atasca. Multiplicando el radio de mutación por el número total de parámetros se obtiene el número de parámetros que deberían ser mutados. A continuación, números aleatorios son generados para seleccionar la columna y fila del parámetro que va a ser mutado. Un parámetro de mutación es reemplazado por un nuevo parámetro aleatorio.

6.2.5 Función objetivo a maximizar. La función de salud

Los AGs generan una población de cromosomas, reemplazando sucesivamente cada población por otra. Los AGs más corrientes requieren una función de *salud* que asigna un valor (salud) a cada cromosoma de la población actual. La salud de un cromosoma depende de lo bien que el cromosoma resuelva el problema a tratar.

La función de salud elegida para resolver el problema de localización debe ser tal que sea capaz de medir el contraste entre dos mapas topo-geométricos locales obtenidos en instantes de tiempo diferentes.

Una manera de medir el solapamiento o contraste entre dos mapas es calcular las distancias mínimas existentes entre todos los puntos que forman estos mapas como se ha realizado en [BlaBoaMor01], pero esto supone un gran coste computacional. En esta tesis se mide este solapamiento utilizando las características topo-geométricas de los mapas generados a partir del DVL. Los parámetros topo-geométricos empleados para calcular el solapamiento son posición de nodos, distancias a los objetos equidistantes y orientación de las ramas entre otros.

El mapa generado en este trabajo es un mapa topológico, que contiene nodos que representan lugares característicos, y ramas que unen estos nodos. Tanto los nodos como las ramas contienen información geométrica, es decir, las coordenadas de los puntos que los definen y la distancia de estos puntos a los objetos a los que son equidistantes. Esta información ha sido obtenida a partir del DVL (ver capítulo 4). Una manera de medir el contraste entre estos mapas es calcular la distancia que existe entre los nodos y ramas equivalentes de ambos mapas. La función de salud usada es:

$$f = \frac{1}{\frac{\sum^{n} d_{\text{nodos equivalentes}} + \sum^{m} d_{\text{ramas equivalentes}}}{n+m} + 1} \tag{6.30}$$

Definición 6.2.3: **Nodos equivalentes.** Se dice que N_i, perteneciente al mapa topo-geométrico g_i, es equivalente al nodo N_j del mapa topo-geométrico g_j si tienen características geométricas parecidas.

Proposición 6.2.1: Se dice que dos nodos tienen características geométricas similares si están próximos en el espacio y los valores de las distancias de estos nodos a los objetos a los que son equidistantes son parecidos.

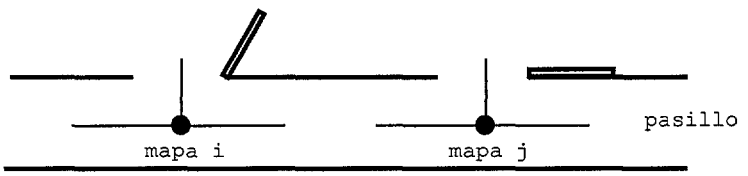


Fig. 6.8: Mapas topo-geométricos con características geométricas similares pero que no están próximos: mapas no equivalentes

Definición 6.2.4: **Ramas equivalentes.** Se dice que R_i, perteneciente al mapa topo-geométrico g_i, es equivalente a la rama R_j del mapa topo-geométrico g_j si tienen características geométricas parecidas.

Proposición 6.2.2: Se dice que dos ramas tienen características geométricas similares si están próximas en el espacio y los valores de la inclinación y la distancia de sus puntos a los objetos a los que son equidistantes son parecidos.

Entre dos mapas topo-geométricos que se solapan existen al menos dos ramas equivalentes, sin embargo no tiene porque haber nodos equivalentes. Este es el caso de dos mapas con una sola rama cada uno. Si entre dos mapas no existen nodos equivalentes, entonces la función de salud se calcula de la manera siguiente:

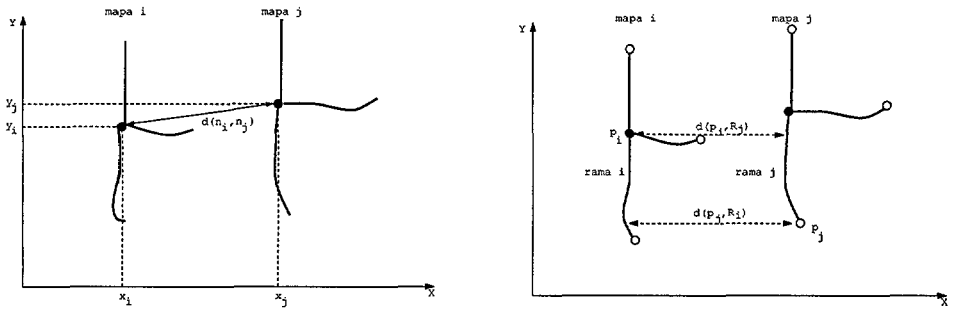
$$f = \frac{1}{\frac{\sum^{m} d_{\text{ramas equivalentes}}}{m} + 1} \tag{6.31}$$

Cuánto mayor es la distancia entre los nodos y ramas equivalentes menor es el valor de la función salud. El valor mínimo de la función salud es 0, y el valor máximo es 1, $f \in [0, 1]$. Un valor de $f = 0$ indica que no hay solapamiento entre los mapas (no existen ni nodos ni ramas equivalentes), y un valor $f = 1$ indica que los mapas se solapan totalmente.

6.2.5.1 Distancia entre nodos

Sea N_i un nodo del mapa topo-geométrico g_i, que tiene por coordenadas (x_i, y_i), y N_j un nodo del mapa topo-geométrico g_j que tiene por coordenadas (x_j, y_j) referidos ambos al mismo sistema de referencia. La distancia entre los dos nodos, es la distancia euclídea definida por la expresión:

$$d(n_i, n_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{6.32}$$



(a) Distancia entre nodos pertenecientes a distintos mapas

(b) Distancia entre ramas pertenecientes a distintos mapas

Fig. 6.9: Distancia entre los elementos que caracterizan un mapa topo-geométrico

Definición 6.2.5: **Distancia entre nodos ($d(n_i, n_j)$).** Se define como distancia entre el nodo i y el nodo j pertenecientes a distintos mapas topo-geométricos y referidos al mismo sistema de referencia, y se denota por $d(n_i, n_j)$, a la distancia euclídea entre los puntos que definen a ambos nodos.

6.2.5.2 Distancia entre ramas

Sea R_i una rama del mapa topo-geométrico g_i formada por los puntos $R_i = \{(x_{i1}, y_{i1}), \cdots, (x_{in}, y_{in})\}$, y R_j una rama del mapa topo-geométrico g_j formada por los puntos $R_j = \{(x_{j1}, y_{j1}), \cdots, (x_{jn}, y_{jn})\}$, referidas ambas al mismo sistema de referencia. Una manera de calcular la distancia entre estas dos ramas sería hallar la media de las distancias mínimas entre cada uno de los puntos que las forman, pero calcular la distancia de esta manera supone un gran coste computacional.

Para reducir el tiempo de cálculo, se calcula una distancia equivalente que no es más que el cuadrado medio de las dos distancias mínimas entre los puntos extremos de las ramas que caen dentro de la rama equivalente, y los puntos que forman la otra rama (ver figura 6.9(b)):

$$d(R_i, R_j) = \frac{\sqrt{d(p_i, R_j)^2 + d(p_j, R_i)^2}}{2} \tag{6.33}$$

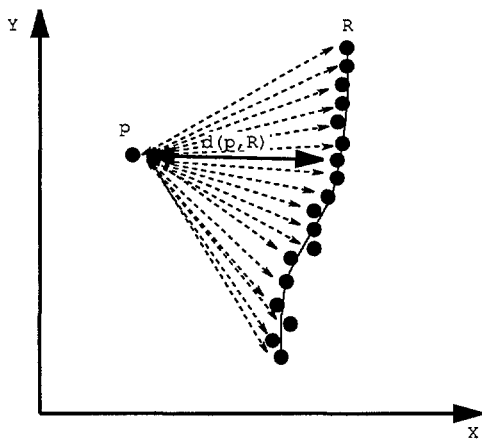


Fig. 6.10: Distancia de un un punto a una rama

Definición 6.2.6: **Distancia de un punto a una rama del mapa topo-geométrico** ($d(p,R)$). Sea p_i un punto del espacio y $R_i = \{(x_{i1}, y_{i1}), \cdots, (x_{in}, y_{in})\}$ una rama del mapa topo-geométrico. Se define como distancia de un punto p_i a una rama R_i, y se denota por $d(p_i, R_i)$ a la distancia euclídea menor entre el punto p_i y cada uno de los puntos que definen la rama R_i:

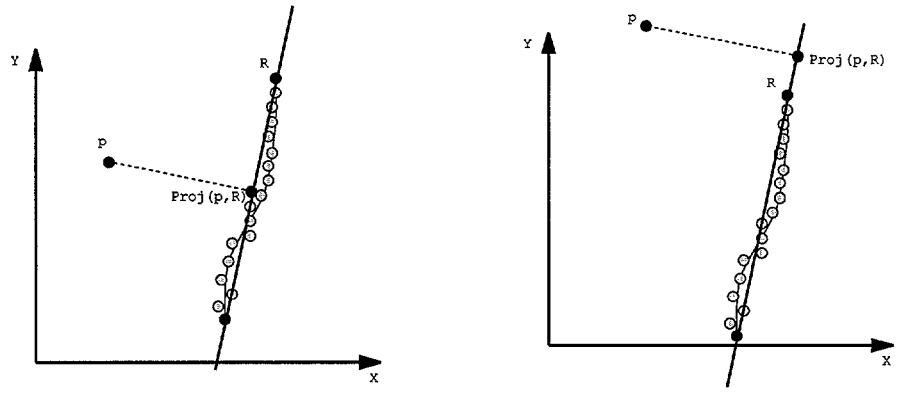
$$d(p_i, R_i) = min(d(p_i, p_R))\forall p_R \in R_i \tag{6.34}$$

donde $d(p_i, p_R)$ es la distancia euclídea entre dos puntos del espacio.

Definición 6.2.7: **Proyección de un punto p a una rama del mapa topo-geométrico** ($Proj(p,R)$). Se define como proyección de un punto p_i a una rama $R_i = \{(x_{i1}, y_{i1}), \cdots, (x_{in}, y_{in})\}$, y se denota por $Proj(p_i, R_i)$, a la proyección de dicho punto sobre la línea formada por los dos extremos de la rama R_i.

Definición 6.2.8: **Punto proyector** (p_{proj}). Se define como punto proyector, y se denota por $p_{proj_i}(R_j)$, al punto extremo de la rama R_i cuya proyección está dentro del segmento formado por los extremos de la rama R_j (ver figura 6.11).

Sea p_{i1} y p_{in} los puntos extremos de la rama R_i y p_{j1} y p_{jn} los puntos extremos de la rama R_j, donde R_i y R_j son dos ramas equivalentes, entonces al menos dos de estos puntos son *puntos proyectores* (ver figura 6.12).



(a) Punto dentro de la rama (b) Punto fuera de la rama

Fig. 6.11: Proyección de un punto a una rama

Si hay más de dos *puntos proyectores*, implica que la proyección de uno de los puntos extremos coincide con uno de los extremos de la otra rama (ver figura 6.12).

Proposición 6.2.3: Si p_k es un punto extremo de la rama R_k y p_l es un punto extremo de la rama R_l equivalente a R_k, tal que, $p_k = Proj(p_l, R_k)$ entonces los dos puntos p_k y p_l son *puntos proyectores*. Las distancias de estos puntos a sus correspondientes ramas equivalentes son: $d(p_k, R_l)$ y $d(p_l, R_k)$. Para el cálculo de la distancia entre dos ramas sólo se emplea de estas dos distancias la que tiene un valor menor, $min(d(p_k, R_l), d(p_l, R_k)))$.

Definición 6.2.9: **Distancia entre dos ramas equivalentes** ($d_{equi}(R_i, R_j)$). Se define distancia entre dos ramas equivalentes, y se denota por $d_{equi}(R_i, R_j)$, a la distancia cuadrática media (ver ecu. 6.35) de las distancias entre los puntos proyectores y las ramas equivalentes, teniendo en cuenta la proposición 6.2.3.

Definición 6.2.10: **Distancia cuadrática media.** Si d_1 y d_2 son dos distancias, se define como distancia cuadrática media a:

$$d = \frac{\sqrt{d_1^2 + d_2^2}}{2} \tag{6.35}$$

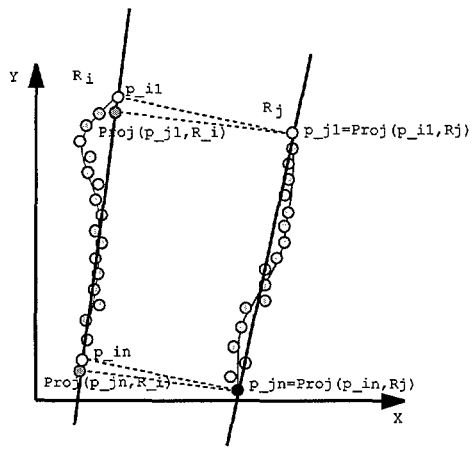


Fig. 6.12: Dos ramas equivalentes con tres *puntos proyectores* (puntos blancos).

Para aumentar la precisión del método de localización, el valor de $d_{equi}(R_i, R_j)$ es multiplicado por un factor que mide la diferencia de orientación entre dos ramas equivalentes:

$$factor_\theta(R_i, R_j) = \frac{abs(\theta(R_i) - \theta(R_j))}{\theta(R_i) + \theta(R_j)} \tag{6.36}$$

donde $\theta(R_i)$ es el ángulo que forma la rama R_i con respecto al eje x positivo del sistema de coordenadas global (ver figura 6.13).

La distancia entre dos ramas se calcula de la siguiente manera:

$$d_\theta(R_i, R_j) = factor_\theta(R_i, R_j) \cdot d_{equi}(R_i, R_j) \tag{6.37}$$

Este factor es importante ya que mide cómo de solapadas están dos ramas. Puede ocurrir que dos ramas tengan el mismo valor de $d_{equi}(R_i, R_j)$ pero distinto valor de orientación (ver figura 6.14). La orientación de una rama tiene que ver con la orientación del robot. Un error pequeño en la orientación del robot, provoca errores grandes a distancias largas.

Un entorno puede estar sujeto a modificaciones, por ejemplo en un entorno de interiores puede haber una puerta abierta en un instante dado que podía haber estado cerrada en un instante previo y viceversa (ver figura 6.15). Para tener en cuenta este caso, se considera que una rama puede ser equivalente a varias ramas.

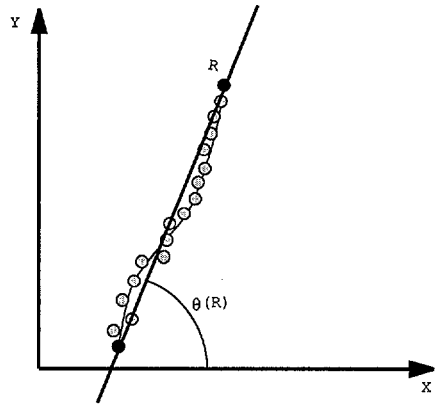
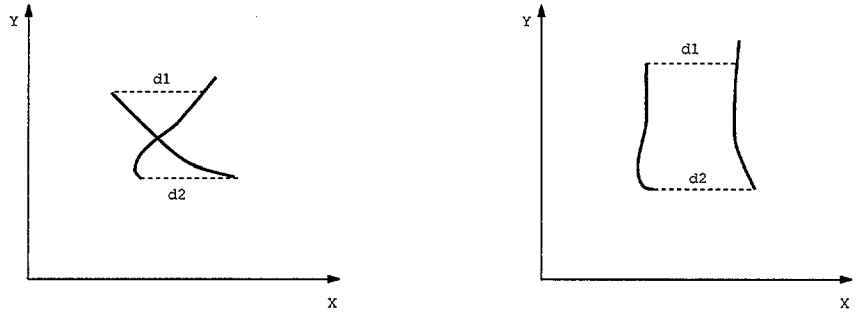


Fig. 6.13: orientación de una rama



(a) Ramas con distinta orientación (b) Ramas con orientación parecidas

Fig. 6.14: Ramas con valores de distancias iguales pero valor de factor de orientación distinto

Sea R_i una rama del mapa g_i equivalente a varias ramas del mapa g_j, $R_{j1}, \cdots, R_{jn}$. Se calcula la distancia entre la rama R_i y el resto de ramas equivalentes, $d_\theta(R_i, R_{j,1}) \cdots, d_\theta(R_n, R_{j,n})$. La distancia resultante final es la media de todas estas distancias.

6.3 Algoritmo de corrección de la posición

En esta sección se presenta el algoritmo que se utiliza para corregir la posición de un robot móvil mediante el contraste de los mapas topo-geométricos

(a) Mapa topo-geométrico en el instante de tiempo $t-n$

(b) Mapa topo-geométrico en el instante de tiempo t

Fig. 6.15: Modificación del mapa del entorno

obtenidos a partir de la información del telémetro láser en instantes de tiempo diferentes.

Los pasos del algoritmo para obtener la posición del robot para la que dos mapas tienen un mejor contraste son:

Entrada: Dos mapas topo-geométricos locales g_i y g_j referidos al mismo sistema de referencia, la posición estimada desde la que se observa el mapa g_i $(\widehat{x}_i, \widehat{y}_i, \widehat{\theta}_i)$ y la posición del robot desde la que se observa el mapa g_j calculada a partir de la información odométrica (x_j, y_j, θ_j).

Salida: Posición del robot estimada, $(\widehat{x}_j, \widehat{y}_j, \widehat{\theta}_j)$, para la que el valor de la función salud tiene el valor máximo.

Paso 1: Calcular la incertidumbre espacial de la posición (x_j, y_j, θ_j) con respecto a la posición $(\widehat{x}_i, \widehat{y}_i, \widehat{\theta}_i)$: E.

Paso 2: Calcular la longitud del cromosoma: l.

Paso 3: Calcular la población inicial. Esta población tiene la posición dada por la odometría y el resto de elementos son determinados aleatoriamente: $\{(x_j, y_j, \theta_j), (x_1, y_1, \theta_1), \cdots, (x_n, y_n, \theta_n)\} \in E$.

Paso 4: Inicializar el resto de parámetros del AG: probabilidad de cruce p_c, probabilidad de mutación p_m, número de iteraciones k.

Paso 5: Evaluar la función de coste para todos los elementos de la población.

Paso 6: Para $i = 1, \cdots, k$:

Paso 6.1: Seleccionar los m mejores individuos.

Paso 6.2: Proceso de reproducción.

Paso 6.3: Proceso de mutación.

Paso 6.4: Añadir a la nueva población los p mejores individuos de la población anterior.

Paso 6.5: Evaluar la función de coste para todos los elementos de la población.

Paso 6.6: Volver al paso 6.

Paso 7: Devolver $(\widehat{x}_j, \widehat{y}_j, \widehat{\theta}_j)$.

En la figura 6.16 se muestran valores de salud para distintas posiciones del robot en la que se contrastan dos mapas topo-geométricos locales con una rama equivalente. El mejor valor de salud, se obtiene cuando los dos mapas están alineados. La posición para la que se obtiene este valor de salud es la mejor estimación de la posición del robot.

En la figura 6.17 se muestran distintos valores de salud para distintas posiciones en las que se contrastan dos mapas topo-geométricos locales con un nodo y 4 ramas equivalentes. El valor mayor de salud se obtiene cuando la posición de los dos nodos coinciden y las ramas están alineadas.

En el capítulo siguiente se explica cómo se utiliza este algoritmo de corrección de la posición junto a la construcción de un mapa global del entorno a medida que el robot se desplaza, mostrando ejemplos en distintos entornos de interiores. En los primeros desplazamientos del robot, como no se tiene mucha información del entorno, este algoritmo es utilizado únicamente para corregir la posición del robot de manera local. Cuando el robot vuelve a pasar por una zona del espacio de la que ya se tiene información previa, este algoritmo se emplea para corregir la posición del robot de manera global.

(a) Valor de $f = 0.0$

(b) Valor de $f = 0.0011$

(c) Valor de $f = 0.002$

(d) Valor de $f = 0.0052$

(e) Valor de $f = 0.0468$

(f) Valor de $f = 0.1087$

Fig. 6.16: Distintos valores de la función salud que mide el contraste entre dos mapas topo-geométricos locales con 1 rama equivalente

(a) Valor de $f = 0.0$

(b) Valor de $f = 0.0002$

(c) Valor de $f = 0.05$

(d) Valor de $f = 0.21$

(e) Valor de $f = 0.33$

(f) Valor de $f = 0.50$

Fig. 6.17: Distintos valores de la función salud que mide el contraste entre dos mapas topo-geométricos locales con un nodo y 4 ramas equivalentes.

7. CONSTRUCCIÓN DEL MAPA TOPO-GEOMÉTRICO GLOBAL Y LOCALIZACIÓN DE MANERA SIMULTÁNEA

7.1 Introducción

Un aspecto muy importante en un robot móvil autónomo que se desplaza por un entorno desconocido a priori es que construya un mapa del entorno que le ayude a localizarse y a navegar por él.

El problema de construir un mapa del entorno es determinar la localización de lugares de interés (como son marcas, obstáculos), frecuentemente relativos a un sistema de coordenadas global. Para construir un mapa del entorno, el robot debe conocer donde está en relación con localizaciones pasadas. Como el movimiento del robot no es preciso, debido a los errores de los sensores internos y al deslizamiento, el robot debe resolver el problema de conocer su posición actual, cuya dificultad se incrementa a medida que aumenta el tamaño del entorno. De este modo el problema de construcción del mapa es el problema del huevo y la gallina [ThrBurFox98a]: para determinar la posición de entidades de interés, el robot debe saber donde está y para saber donde está, el robot necesita conocer las localizaciones de las entidades de interés.

Resolver el problema SLAM (Simultaneous Localization And Mapping) es muy costoso computacionalmente, ya que, se han de considerar una gran cantidad de soluciones posibles. Es por esta razón por la que el algoritmo desarrollado no se puede realizar on-line. Una vez que se obtiene un mapa consistente del entorno, es decir, que en cada ciclo no se obtiene una gran diferencia entre las soluciones obtenidas, ya no será necesario seguir ejecutando este algoritmo, sino que a partir de este momento se empleará el mapa topo-geométrico global construido para llevar a cabo las diferentes tareas,

como son navegación y localización.

En las secciones siguientes se presenta un método que permite construir un mapa topo-geométrico del entorno a medida que el robot se desplaza. Este mapa se construye integrando la información del mapa topo-geométrico local generado en un instante de tiempo con el mapa/s topológico/s generado/s en los instantes anteriores.

7.2 El modelo

El mapa del entorno es construido a partir de una secuencia de acciones de control intercaladas con observaciones. Sin perder la generalidad se considera que el movimiento y percepción del robot se realizan de una manera alterna. Por tanto, la información disponible para generar el mapa global del entorno es de la forma:

$$M = \{o_1, \xi_2, o_2, \xi_3, \cdots, o_{T-1}, \xi_T, o_T\} \tag{7.1}$$

donde o_t representa la observación que el robot ha hecho en el instante de tiempo t, y ξ_t representa la lectura odométrica que caracteriza la acción ejecutada por el robot entre el instante de tiempo $t-1$ y t.

En este trabajo, las observaciones utilizadas para generar el mapa global son mapas topo-geométricos locales obtenidos en cada desplazamiento del robot:

$$M = \{g_1, \xi_2, g_2, \xi_3, \cdots, g_{T-1}, \xi_T, g_T\} \tag{7.2}$$

donde g_i es el mapa topo-geométrico local obtenido en la posición i para un desplazamiento del robot ξ_i desde la posición $i-1$. El mapa local g_1 es calculado en la posición desde la que el robot comienza a desplazarse y por tanto no se tiene ninguna información odométrica.

A partir de la información odométrica y de los mapas locales obtenidos en cada desplazamiento se construye el mapa global. El modelo presentado en esta tesis representa al entorno como un mapa topológico donde los nodos representan lugares característicos. Estos lugares son obtenidos del DVL y representan lugares físicos típicos de entornos de interiores estructurados como son puertas, intersecciones, estrechamientos, etc.. Las ramas representan las uniones entre estos nodos. El mapa global se puede representar como:

$$M = G(\sum Nodo, \sum Rama) \tag{7.3}$$

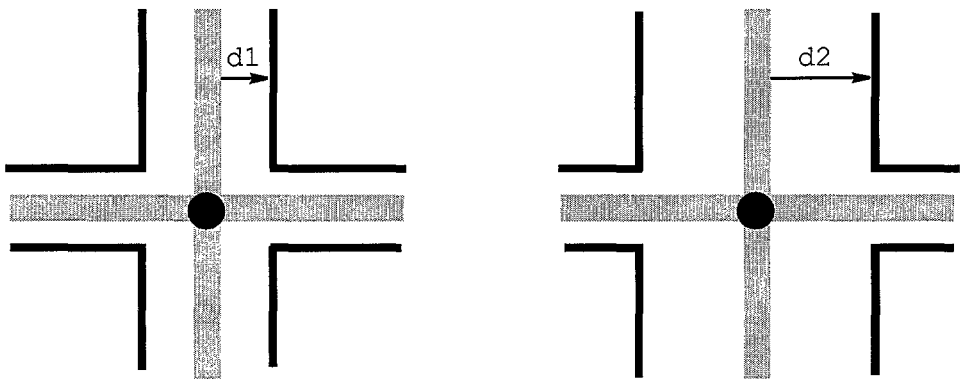


Fig. 7.1: Grafos con las mismas características topológicas pero distintas características geométricas

donde G es el mapa topo-geométrico global formado por un conjunto de nodos y ramas. Tanto los nodos como las ramas del mapa global M contienen información geométrica sobre las posiciones de los puntos que los forman con respecto a un sistema de referencia global situado en la posición inicial del robot, y las distancias de estos puntos a los objetos más cercanos equidistantes. Esta información es importante para eliminar posiciones ambiguas y facilitar la localización del robot. En la figura 7.1 se representan dos zonas distintas del espacio que tienen la misma representación topológica. La información geométrica, en este caso la anchura del pasillo, es la que permite distinguir una zona de otra.

Inicialmente se considera que el robot comienza a explorar el entorno desde una posición desconocida a priori, y a partir de ella construye el mapa global del entorno integrando la información de los mapas locales obtenidos en cada desplazamiento. Se considera, por tanto, que el origen de coordenadas del sistema global se encuentra en la posición de la que parte el robot (ver figura 7.2).

Inicialmente la única información de que dispone el robot es la información proporcionada por el telémetro láser. A partir de esta información se genera el mapa topo-geométrico local g_1. En este caso, el mapa topo-geométrico global coincide con el mapa topo-geométrico local, ya que, no se dispone de más información con la que comparar este mapa local:

$$M_1 = \{g_1\} \text{ para } t = 1 \tag{7.4}$$

A medida que el robot avanza adquiere más información del entorno, te-

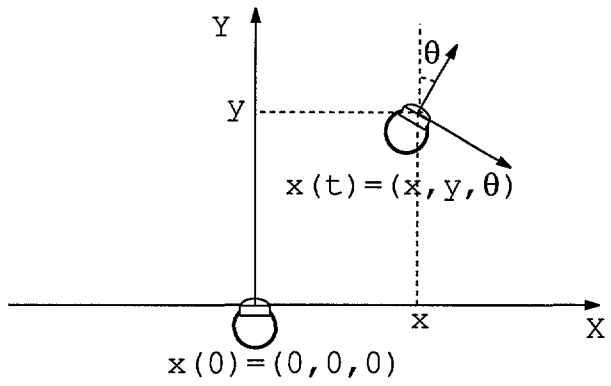


Fig. 7.2: Posición actual del robot con respecto al sistema de coordenadas global

niendo una aproximación de lo que se ha desplazado utilizando la información odométrica. Debido a los errores de odometría, en cada desplazamiento se comete un error en la estimación de la posición que crece incrementalmente a medida que el robot se desplaza. Por tanto, es necesario corregir esta posición. Para la corrección de la posición se utiliza el algoritmo, que mide el solapamiento entre dos mapas, explicado en el capítulo 6.

En el primer desplazamiento del robot la información disponible es el mapa generado en la posición actual g_2, la información odométrica ξ_2 y el mapa topo-geométrico local obtenido en el instante anterior g_1. Si el desplazamiento no ha sido muy grande se puede considerar que el entorno no ha sufrido modificaciones grandes y que el error cometido en la odometría es pequeño. La posición actual del robot se corrige contrastando la información del mapa topo-geométrico local g_2 con la del mapa obtenido en el instante de tiempo anterior g_1. A medida que el robot avanza adquiere más información del entorno, de tal manera que para corregir su posición se puede contrastar el nuevo mapa topo-geométrico local no solamente con el mapa local obtenido en el instante previo sino también con los mapas locales obtenidos en los instantes anteriores.

En las secciones siguientes se explica el algoritmo de construcción del mapa del entorno y localización del robot de manera simultánea. En este algoritmo se alternan dos pasos. Un primer paso en el que se corrige la posición del robot considerando que los mapas locales obtenidos no contienen errores y un segundo paso en el que se corrige el mapa global considerando que la posición obtenida en el paso anterior es la posición correcta. Este proceso de

alternar estos dos pasos se puede realizar iterativamente en cada desplazamiento para mejorar los resultados obtenidos, pero hay que considerar que esto hace más lento el proceso. En cada desplazamiento se ejecutan estos dos pasos tantas veces como se considere oportunas para mejorar el mapa global obtenido. Una vez que ya no se consigue una mejora substancial en los resultados obtenidos entre dos desplazamientos sucesivos este proceso deja de ejecutarse y se trabaja ya con el mapa global generado.

7.3 Construcción incremental del mapa

En esta sección se considera el problema de construcción del mapa del entorno y localización de manera simultánea (SLAM) usando los mapas locales obtenidos en cada nueva posición del robot.

La idea de este método es combinar la idea de estimación posterior, que permite construir mapas de grandes dimensiones en entornos con ciclos. La idea de estimación posterior consiste en corregir no solamente la información obtenida en un instante de tiempo, sino la obtenida también en todos los instantes anteriores.

El conjunto de entidades utilizadas en la resolución de este problema son el conjunto de nodos N_i y ramas R_i que caracterizan un mapa topo-geométrico local g_i junto con sus características geométricas:

$$g_i = \left\{ \sum_{j=1}^{n_N} N_{i,j}, \sum_{k=1}^{n_R} R_{i,k} \right\} \tag{7.5}$$

En la figura 7.3 se muestra el algoritmo de obtención del mapa topo-geométrico global. Al igual que en el algoritmo EM [BurFoxJan99] se utilizan dos pasos para resolver el problema SLAM (Simultaneous Localization and Mapping):

1. Corrección de la posición del robot.

2. Construcción del mapa utilizando la posición estimada en el paso anterior.

Estos dos pasos se ejecutan cada vez que el robot se desplaza y toma información del entorno. Este proceso deja de realizarse cuando se obtiene un mapa global que ya no se modifica en el tiempo, o que sufre pocas modificaciones. La diferencia principal del algoritmo desarrollado en esta tesis con el algoritmo EM es que no se calculan densidades de probabilidad.

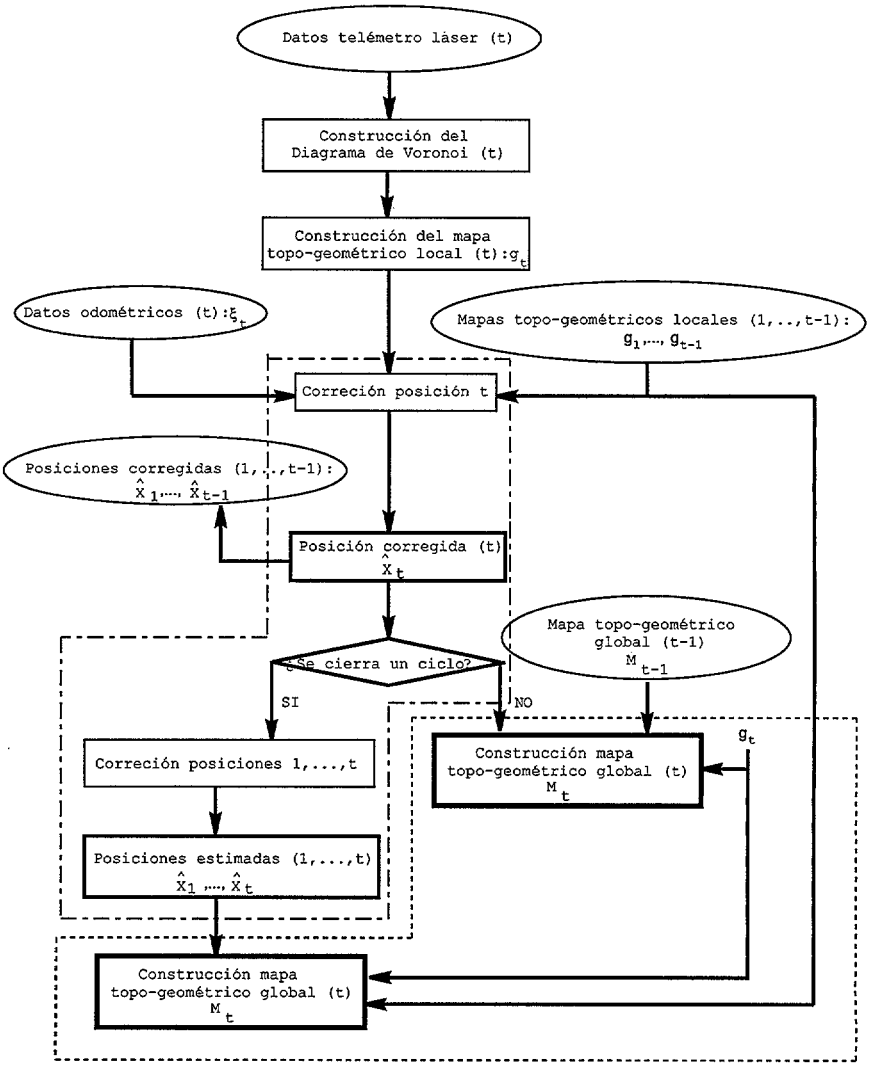


Fig. 7.3: Algoritmo de construcción del mapa topo-geométrico global

A continuación se explican con detalle cada uno de los pasos mencionados anteriormente.

7.3.1 Corrección de la posición del robot

La corrección de la posición se realiza considerando que los mapas locales obtenidos a partir de la información sensorial no poseen ningún error. La corrección de la posición se realiza utilizando el método de localización explicado en el capítulo 6.

Se considera que la posición de la que parte el robot es el origen de coordenadas del sistema de referencia global, ya que, no se dispone de ninguna información adicional que permita estimar esta posición. El robot comienza a desplazarse adquiriendo información del entorno utilizando un telémetro láser. La información odométrica obtenida en cada desplazamiento posee errores que hay que corregir para construir un mapa consistente. Debido a estos errores odométricos, es necesario corregir la posición del robot en cada desplazamiento.

Para corregir la posición del robot se contrasta la información del mapa local obtenido en un instante de tiempo con la información de los mapas locales obtenidos en instantes de tiempos anteriores. Para mejorar el proceso de localización no se emplea únicamente la información del mapa local obtenido en el instante previo sino que se utiliza la información de todos los mapas locales obtenidos anteriormente, ya que, puede ocurrir que el mapa local del instante $t = i$ no se solape con el mapa local del instante anterior $t = i-1$, pero que si lo haga con algún mapa local de algún instante anterior. Por otra parte, considerar toda la información disponible permite determinar si una zona del espacio ya ha sido observada por el robot y de esta manera corregir todas las posiciones anteriores (estimación posterior). La estimación posterior es un factor clave para poder construir un mapa consistente en entornos cíclicos.

Se trabaja con los mapas locales en lugar de con el mapa global porque es menos costoso computacionalmente y se eliminan los problemas que implica utilizar un único mapa monolítico. La información local sólo posee los errores debidos al sensor externo, que se pueden considerar poco importantes si se utiliza un buen telémetro láser, mientras que un mapa global posee este error mas el debido al que se comete en la estimación de la posición, el cual, tiene una magnitud considerable. Por otra parte, ya se ha comentado que la ventaja principal del algoritmo presentado en esta tesis es que permite corregir no solamente la última posición en la que se encuentra el robot, sino

que también permite corregir todas las posiciones anteriores cuando se estima que un ciclo se ha cerrado, el problema que se plantea entonces si se trabaja con el mapa global es ¿cómo modificar el mapa global si una de las posiciones estimadas no es la correcta?.

En cada desplazamiento del robot se corrige tanto su última posición cómo sus posiciones anteriores. Este último paso sólo se ejecuta cuando una zona del espacio vuelve a ser observada. El proceso de corrección de las posiciones presentado en esta tesis se divide en dos pasos:

1. Un paso *hacia adelante* o *forward* en el que se corrige la última posición del robot.

2. Un paso *hacia atrás* o *backward* en el que se corrigen todas las posiciones del robot. Este paso sólo se ejecuta cuando se estima que un ciclo se ha cerrado.

En las secciones siguientes se explica detalladamente cada uno de los pasos mencionados.

7.3.1.1 Corrección *hacia adelante*

La corrección *hacia adelante* se ejecuta siempre que el robot realiza un nuevo desplazamiento y una nueva observación.

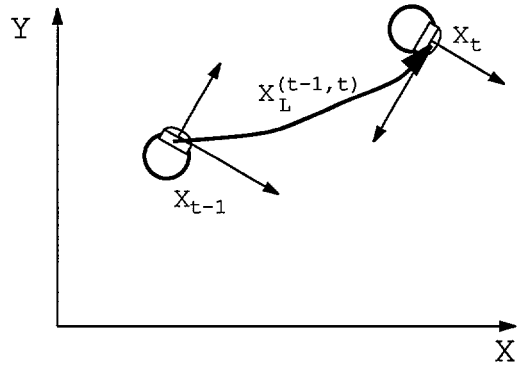


Fig. 7.4: Posición actual del robot

Para un desplazamiento del robot desde una posición X_i a una posición X_{i+1}, una primera estimación de lo que se ha desplazado es la proporcionada por los sensores odométricos ξ_{i+1}.

Estos nos proporcionan la posición relativa del robot con respecto a la posición en el instante de tiempo anterior, $X_L^{(i,i+1)} = \xi_{i+1}$. La nueva posición del robot, con respecto al sistema de coordenadas global, viene dada por la expresión:

$$X_t = \begin{pmatrix} x_t \\ y_t \\ \theta_t \end{pmatrix} = T(\theta_{t-1}) \cdot X_L^{(t-1,t)} + X_{t-1} \tag{7.6}$$

donde $T(\theta)$ es la matriz de transformación:

$$T(\theta) = \begin{pmatrix} \cos(\theta) & sen(\theta) & 0 \\ -sen(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{7.7}$$

Conociendo la posición del robot en los instantes de tiempo anteriores $(X_1, \cdots, X_{i-1})$ y la posición actual del robot X_i, todas ellas referidas al sistema de coordenadas global, se puede calcular (ec. 6.6) cual es la incertidumbre espacial de la posición actual del robot con respecto a cada una de las posiciones anteriores $\mathrm{E}_i^j = \{E_{i,x}^j, E_{i,y}^j, E_{i,\theta}^j\}$. E_i^j es la incertidumbre espacial de la posición del robot en el instante $t = i$ con respecto a la posición en el instante $t = j$ con $j = 1, \cdots, i-1$ (ver definición 6.2.2). Esta incertidumbre determinará el espacio de búsqueda del algoritmo genético donde se busca la mejor estimación de la posición del robot (ver sección 6.2.1). El centro de cada incertidumbre espacial es la posición del robot en ese instante referido a cada una de las posiciones anteriores:

$$X_L^{(j,i)} = T(\theta_j) \cdot (X_i - X_j), \text{ con } j = 1, \cdots, i-1 \tag{7.8}$$

donde X_i y X_j son los valores de la posición del robot en los instantes $t = i$ y $t = j$ respectivamente referidos al sistema de referencia global. Como $X_L^{(j,i)}$ es una estimación de la posición del robot en el instante i referida a la posición j, esta forma parte de la población inicial del AG.

Los valores de la incertidumbre espacial E_i^j no están referidos al sistema de referencia global sino que están referidos a distintos sistemas de referencia, cada uno de los cuales tiene como origen de coordenadas la posición del robot desde la que se calcula la incertidumbre. Por tanto, los valores de las posiciones dadas por el algoritmo de contraste no están referidos al sistema de referencia global sino que están referidos al sistema de referencia situado en cada una de las posiciones anteriores. El algoritmo de contraste se aplica para cada uno de estos valores de incertidumbre, obteniéndose una solución

de la posición corregida para cada uno de ellos $(\widehat{X}_L^{(1,i)}, \cdots, \widehat{X}_L^{(i-1,i)})$. Cada uno de estos valores, que representan la posición corregida, se puede referir al sistema de coordenadas global mediante la expresión siguiente:

$$\widehat{X}_i^j = T(\widehat{\theta}_j) \cdot \widehat{X}_L^{(j,i)} + \widehat{X}_j, \text{ con } j = 1, \cdots, i-1 \tag{7.9}$$

donde $\widehat{X}_L^{(j,i)}$ es la posición corregida, obtenida del algoritmo de contraste, en el instante de tiempo $t = i$ con respecto al sistema de referencia situado en la posición $t = j$, X_j es la posición en el instante $t = j$ y $\widehat{X}_i^j$ es la posición estimada en el instante $t = i$ con respecto a la posición del instante $t = j$ referida ya al sistema de coordenadas global.

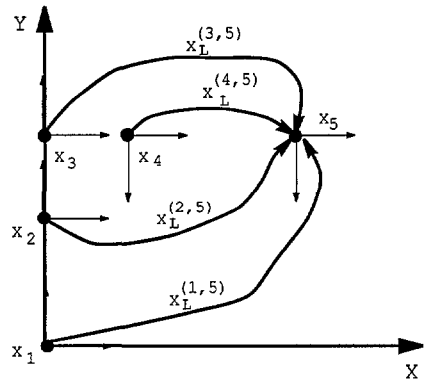


Fig. 7.5: Posiciones relativas

En la figura 7.5 se muestran las posiciones relativas de la posición X_5 con respecto a cada una de las posiciones anteriores. Se considera que el robot parte de la posición X_1 y se desplaza hasta la posición X_2. Una primera estimación de X_2 se obtiene a partir de la información odométrica ξ_2:

$$X_2 = \begin{pmatrix} x_2 \\ y_2 \\ \theta_2 \end{pmatrix} = T(\theta_1) \cdot \xi_2 + X_1 \tag{7.10}$$

El valor de la incertidumbre de la posición X_2 con respecto a X_1, se calcula a partir de los valores de la matriz de covarianzas (ecu. 6.6) según se ha explicado en la sección 6.2.1:

$$Cov\,(2|1) = J\,(1) \cdot Cov\,(0|0) \cdot J\,(1)^T + Q\,(1) \tag{7.11}$$

A partir de la matriz $Cov(2|1)$ se obtiene el valor de la incertidumbre espacial E_2^1. Aplicando el algoritmo de contraste se obtiene la posición estimada $\widehat{X}_L^{(1,2)}$ con respecto a la posición 1. En este caso como la posición 1 es el origen de coordenadas del sistema de referencia global no es necesario hacer un cambio de coordenadas: $\widehat{X}_1 = \widehat{X}_L^{(1,2)}$.

A continuación el robot se desplaza desde la posición X_2 a la posición X_3. Una primera estimación de X_3 se obtiene a partir de la información odométrica ξ_3:

$$X_3 = \begin{pmatrix} x_3 \\ y_3 \\ \theta_3 \end{pmatrix} = T(\theta_2) \cdot \xi_3 + X_2 \tag{7.12}$$

Se tiene entonces dos valores de incertidumbre de la posición X_3, uno con respecto a la posición X_1:

$$Cov\,(3|1) = J\,(2) \cdot Cov\,(2|1) \cdot J\,(2)^T + Q\,(2) \tag{7.13}$$

y otro con respecto a la posición X_2:

$$Cov\,(3|2) = J\,(2) \cdot Cov\,(0|0) \cdot J\,(2)^T + Q\,(2) \tag{7.14}$$

De la matriz $Cov(3|1)$ se obtiene el valor de la incertidumbre espacial E_3^1, y de la matriz $Cov(3|2)$ se obtiene el valor de la incertidumbre E_3^2. Aplicando el algoritmo de contraste para cada incertidumbre espacial se obtienen dos valores estimados de la posición 3, $\widehat{X}_L^{(1,3)}$ referido a la posición 1 y $\widehat{X}_L^{(2,3)}$ referido a la posición 2. Haciendo un cambio de coordenadas según la expresión 7.9 se obtienen los valores corregidos en la posición 3 referidos al sistema de referencia global, $\widehat{X}_3^1$ y $\widehat{X}_3^2$.

El robot sigue avanzando hasta una posición X_4. Los valores de la incertidumbre con respecto a cada una de las posiciones anteriores son, con respecto a la posición 1:

$$Cov\,(4|1) = J\,(3) \cdot Cov\,(3|1) \cdot J\,(3)^T + Q\,(3) \tag{7.15}$$

con respecto a la posición 2:

$$Cov\,(4|2) = J\,(3) \cdot Cov\,(2|1) \cdot J\,(3)^T + Q\,(3) \tag{7.16}$$

y con respecto a la posición 3:

$$Cov\,(4|3) = J\,(3) \cdot Cov\,(0|0) \cdot J\,(3)^T + Q\,(3) \tag{7.17}$$

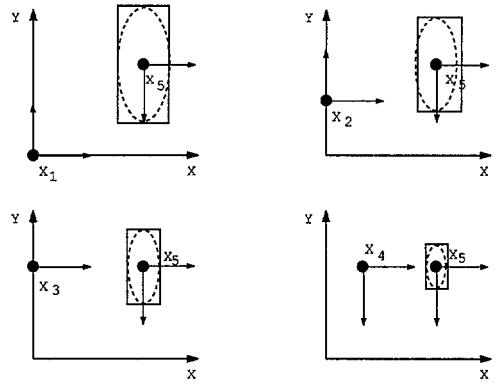


Fig. 7.6: Incertidumbre de una posición con respecto a posiciones anteriores

A partir de estas matrices de covarianza se obtienen los valores de las incertidumbres espaciales correspondientes, E_4^1, E_4^2 y E_4^3. Aplicando el algoritmo de contraste para cada uno de estos valores de incertidumbre espacial se obtienen los valores de las posiciones corregidas y referidas a cada una de las posiciones anteriores $\widehat{X}_L^{(1,4)}$, $\widehat{X}_L^{(2,4)}$ y $\widehat{X}_L^{(3,4)}$. Realizando un cambio de coordenadas según la ecuación 7.9 se refieren los valores anteriores al sistema de coordenadas global: $\widehat{X}_4^1$, $\widehat{X}_4^2$ y $\widehat{X}_4^3$. Este proceso se repite en cada desplazamiento del robot.

En la figura 7.6 se muestra como varía la incertidumbre espacial en traslación de la posición X_5 con respecto a las posiciones anteriores $(E_{5,x}^i, E_{5,y}^i)$. El valor de la incertidumbre espacial de X_5 con respecto a X_1, E_5^1, está referido al sistema de referencia que tiene como origen la posición X_1, el valor de incertidumbre de X_5 con respecto a X_2, E_5^2, está referido al sistema de referencia que tiene como origen la posición X_2, y así sucesivamente. Se observa que la incertidumbre espacial de la posición X_5 es mayor cuanto más se aleja de la posición desde la que se calcula dicha incertidumbre.

Conociendo el espacio de búsqueda de una posición en un instante de tiempo dado con respecto a cada una de las posiciones anteriores se puede corregir el valor de esta posición aplicando el algoritmo de corrección para cada uno de estos espacios de búsqueda, contrastando el mapa local obtenido en esta posición con el mapa local obtenido en la posición desde la que se calcula la incertidumbre espacial. Para cada espacio de búsqueda se obtiene una posición corregida, que se puede referir al sistema de coordenadas global, y su salud correspondiente.

Dado $< g_T, X_T >$ para el instante de tiempo $t = T$, donde X_T es el valor

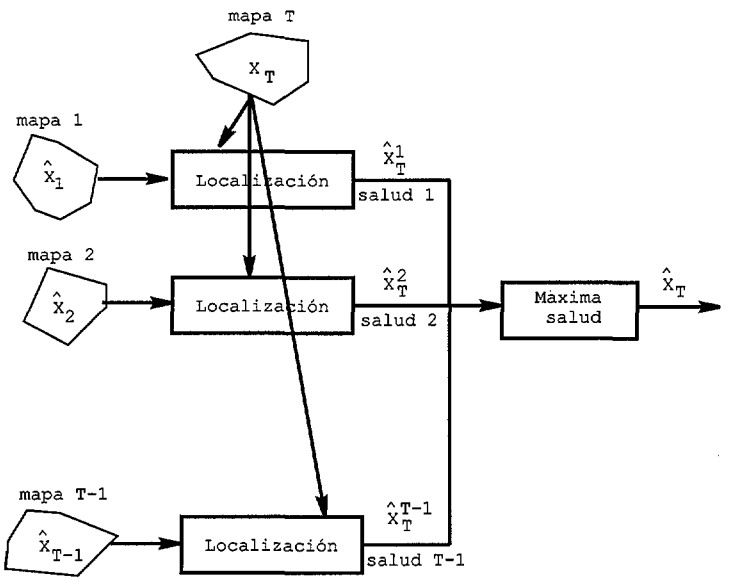


Fig. 7.7: Algoritmo *hacia atrás* para la estimación de la posición actual

de la posición del robot con respecto al sistema de referencia global teniendo en cuenta únicamente la información odométrica, se calcula el valor estimado de esta posición $\widehat{X}_L^{(T,i)}$ con $i = 1, \cdots, T-1$ con respecto a cada mapa local anterior para el que el valor de la función salud f_i es máximo. Hay que considerar que estas posiciones corregidas están referidas al sistema de referencia situado en la posición desde la que se calcula la incertidumbre espacial, por lo que es necesario, realizar a continuación un cambio de coordenadas para referir cada una de estas posiciones al sistema de referencia global, $\widehat{X}_T^i$. La posición estimada en el instante de tiempo T es:

$$\widehat{X}_T = \widehat{X}_T^K \mid f_k = \max[f_i] \text{ con } i = 1, \cdots, T-1 \tag{7.18}$$

Para reducir el número de operaciones sólo se realiza el proceso de contraste entre aquellos mapas locales cuyos campos de visión se solapan.

Definición 7.3.1: **Campo visión del mapa local** g_i ($\mathbb{CV}_i$). Se define como campo de visión del mapa topo-geométrico local g_i, y se denota por $\mathbb{CV}_i$, a la zona rectangular del espacio comprendida entre el robot y los puntos que delimitan al mapa g_i.

Para calcular el conjunto de mapas locales con los que se solapa el mapa

local g_i se transforman los puntos que definen dicho mapa a cada uno de los sistemas de referencia definidos por las posiciones anteriores.

El mapa local g_i está formado por un conjunto de puntos que definen los nodos y ramas. Estos puntos están referidos a un sistema de referencia local centrado en el sensor: $g_i = \{\sum N_{i,j}, \sum R_{i,k}\} = \{(x_1^i, y_1^i), (x_2^i, y_2^i), \cdots, (x_n^i, y_n^i)\}$. Para calcular el campo de visión del mapa local g_i con respecto a cada una de las posiciones anteriores basta con realizar un cambio de coordenadas según la expresión (ver figura 7.8):

$$\begin{pmatrix} x_k^{(i,j)} \\ y_k^{(i,j)} \end{pmatrix} = \begin{pmatrix} \cos(\theta_j) & sen(\theta_j) \\ -sen(\theta_j) & \cos(\theta_j) \end{pmatrix} \cdot \begin{pmatrix} x_k^i \\ y_k^i \end{pmatrix} + \begin{pmatrix} x_j \\ y_j \end{pmatrix} \tag{7.19}$$

Donde (x_j, y_j, θ_j) son las posiciones estimadas del robot referidas a cada uno de los sistemas de referencia situados en las posiciones anteriores, es decir, $X_L^{(j,i)} = (x_j, y_j, \theta_j)$; y (x_k^i, y_k^i) son las coordenadas de cada uno de los puntos que componen el mapa local g_i.

Definición 7.3.2: **Campo de Visión del mapa g_i referido al sistema de referencia j ($\mathbb{CV}_i^j$)** Se define como campo de visión del mapa g_i referido al sistema de coordenadas situado en la posición j, y se denota por $\mathbb{CV}_i^j$, a la zona rectangular del espacio definida por $\mathbb{CV}_i$ y referida al sistema de coordenadas situado en la posición j.

Definición 7.3.3: Se define MS_i al conjunto de mapas con los que se solapa el mapa local calculado en el instante $t = i$, $MS_i = \{g_j, \cdots, g_k\}$.

Realizando el cambio de coordenadas para cada punto del mapa local g_i, según la ecuación 7.19, se obtiene el mapa local referido a cada uno de los sistemas de coordenadas situados en cada una de las posiciones anteriores:

$$\begin{aligned} g_{(i,j)} = & \{\sum N_{i,p}^j, \sum R_{i,k}^j\} \\ = & \{(x_{j,1}^i, y_{j,1}^i), (x_{j,2}^i, y_{j,2}^i), \cdots, (x_{j,n}^i, y_{j,n}^i)\} \end{aligned} \tag{7.20}$$

donde $g_{(i,j)}$ es el mapa topo-geométrico local obtenido en la posición i y referido al sistema de coordenadas situado en la posición j. A la hora de calcular el campo de visión $\mathbb{CV}_i^j$ hay que considerar que existe un espacio de búsqueda, es decir, una incertidumbre espacial de la posición i con respecto a la posición j. Esta incertidumbre espacial E_i^j representa el espacio de soluciones posibles. Para cada una de estas soluciones se obtienen campos de visión diferentes, $\mathbb{CV}_i^{j,k}$. Por tanto, el campo de visión para el mapa

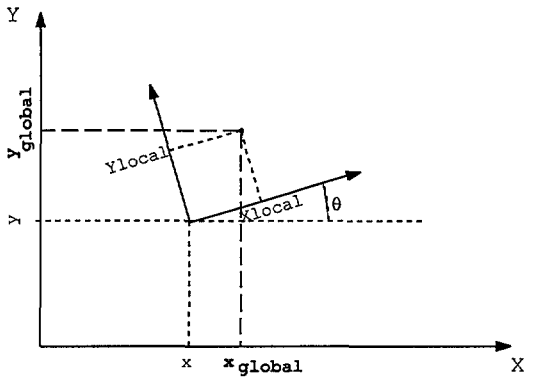


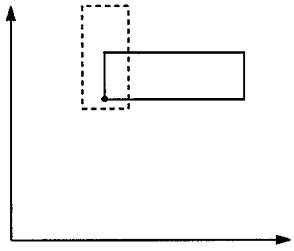
Fig. 7.8: Cambio de coordenadas de un sistema local a un sistema global

g_i con respecto a la posición j se obtiene fusionando todos los campos de visión obtenidos para todas la posiciones posibles definidas por el espacio de búsqueda: $\mathbb{CV}_i^j = \mathbb{CV}_i^{j,1} \bigcup \mathbb{CV}_i^{j,2} \bigcup \cdots \bigcup \mathbb{CV}_i^{j,k}$. Calcular $\mathbb{CV}_i^j$ de esta manera es muy costoso computacionalmente, debido a que existen infinitas soluciones posibles en el espacio de búsqueda. Para simplificar el problema, $\mathbb{CV}_i^j$ se calcula obteniendo la envolvente de los campos de visión para los puntos extremos del espacio de incertidumbre (espacio de búsqueda). En la figura 7.9 se representan los campos de visión para cada uno de los extremos del espacio de búsqueda considerando ya la incertidumbre angular en cada uno de ellos, y el campo de visión total obtenido de la fusión de los campos de visión en cada extremo.

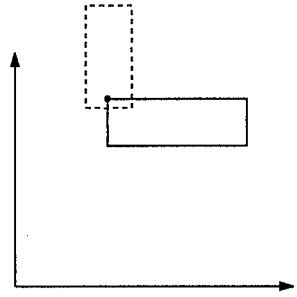
El algoritmo de localización se aplica entre dos mapas cuyos campos de visión se solapan.

Proposición 7.3.1: Para que dos mapas se solapen al menos lo deben hacer sus campos de visión correspondientes: $\mathbb{CV}_i^j \bigcap \mathbb{CV}_j \neq \emptyset$ con $i \neq j$.

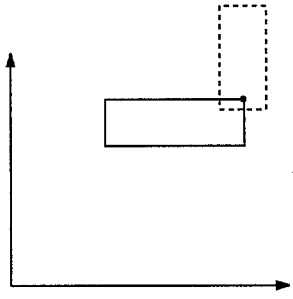
En la figura 7.10 se muestran diferentes mapas topo-geométricos locales referidos al sistema de referencia global con sus correspondientes campos de visión. El origen del sistema de coordenadas global se encuentra en la posición desde la que se genera el mapa con ramas en color rojo (círculo negro). El rectángulo de color negro es el rectángulo de incertidumbre de la posición 2 (círculo en azul) con respecto a la posición 1 (círculo en color negro). Este rectángulo define el espacio de búsqueda de la posición 2. La posición 2 es calculada considerando la información odométrica. El campo de visión del mapa topo-geométrico generado en la posición 1 es de color azul



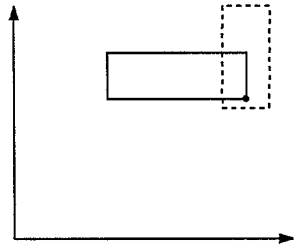
(a) Campo de visión para el punto inferior izquierdo del espacio de búsqueda



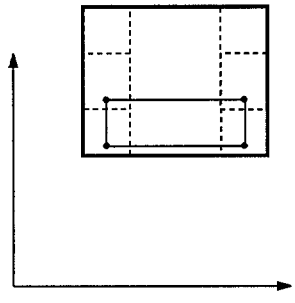
(b) Campo de visión para el punto superior izquierdo del espacio de búsqueda



(c) Campo de visión para el punto superior derecho del espacio de búsqueda

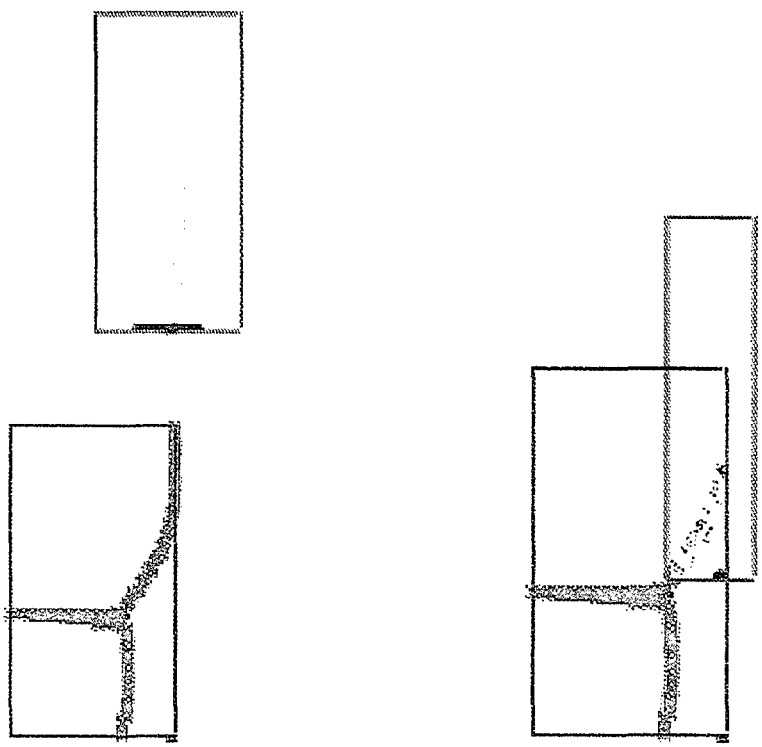


(d) Campo de visión para el punto inferior derecho del espacio de búsqueda



(e) Campo de visión total

Fig. 7.9: Fusión de los campos de visión de los extremos del espacio de búsqueda



(a) Caso en el que el campo de visión de dos mapas no se solapan

(b) Caso en el que el campo de visión de dos mapas se solapan

Fig. 7.10: Campos visión globales

y el campo de visión del mapa topo-geométrico generado en la posición 2 es de color verde. Este último tiene dimensiones mayores que las dimensiones del mapa topo-geométrico local, debido a que se considera la incertidumbre espacial. En la figura 7.10(a) se muestra el caso en el que los campos de visión de dos mapas no se solapan, y en la figura 7.10(b) se muestra el caso en el que los campos de visión de dos mapas se solapan.

A continuación se describen los pasos a seguir para llevar a cabo la localización *hacia adelante*.

7.3.1.1.1 Algoritmo *hacia adelante*. Los pasos del algoritmo *hacia adelante* son:

Entrada: Mapa local obtenido a partir de la información sensorial y la odometría: $< g_T, \xi_T >$, y los mapas locales y sus respectivas posiciones corregidas obtenidos en los instantes de tiempo anteriores: $< g_i, \widehat{X}_i >$ con $i = 1, \cdots, T-1$.

Salida: Posición estimada: $\widehat{X}_T$

Paso 1: Calcular el valor de la posición del robot con respecto al sistema de referencia global considerando la información odométrica: X_T.

Paso 2: Calcular el valor de la posición en el instante T con respecto a cada una de las posiciones anteriores: $X_L^{(i,T)}$ con $i = 1, \cdots, T-1$

Paso 3: Calcular la incertidumbre de la posición X_T con respecto a cada una de las posiciones anteriores $\widehat{X}_i$: E_T^i con $i = 1, \cdots, T-1$.

Paso 4: Calcular el campo de visión de cada mapa local: $\mathbb{CV}_i$ con $i = 1, \cdots, T-1$.

Paso 5: Calcular los campos de visión del mapa obtenidos en la posición X^T con respecto a cada una de las posiciones anteriores: $\mathbb{CV}_T^i$ con $i = 1, \cdots, T$.

Paso 6: Calcular el conjunto de mapas con los que se solapa g_T: MS_T (ver definición 7.3.3).

Paso 7: Calcular la mejor posición para cada uno de los mapas con los que se solapa g_T utilizando el algoritmo de localización: $\widehat{X}_L^{(i,T)}$ con $i = 1, \cdots, T-1$.

Paso 8: Transformar los valores anteriores al sistema de coordenadas global: $\widehat{X}_T^i$ con $i = 1, \cdots, T-1$.

Paso 9: Calcular la mejor posición estimada en el instante de tiempo $t = T$: $\widehat{X}_T$.

Paso 10: Devolver $\widehat{X}_T$.

Este proceso se repite en cada desplazamiento del robot. Con el algoritmo *hacia adelante* sólo se corrige la última posición del robot. Se corregirán también las posiciones anteriores cuando se considere que un ciclo se ha cerrado. En este caso se aplicará también el algoritmo *hacia atrás.*

Proposición 7.3.2: Un ciclo se cierra cuando $MS_T \not\subseteq (MS_{T-1} + g_{T-1})$.

7.3.1.1.2 Resultados experimentales Se considera que el robot parte de una posición desconocida a priori y que se supone el origen de coordenadas del sistema de referencia global. En esta posición el robot adquiere información del entorno a través del telémetro láser y construye el mapa local g_1. En la figura 7.11 se muestra el mapa generado en una posición inicial cualquiera (círculo negro).

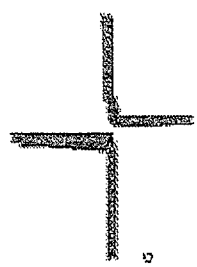
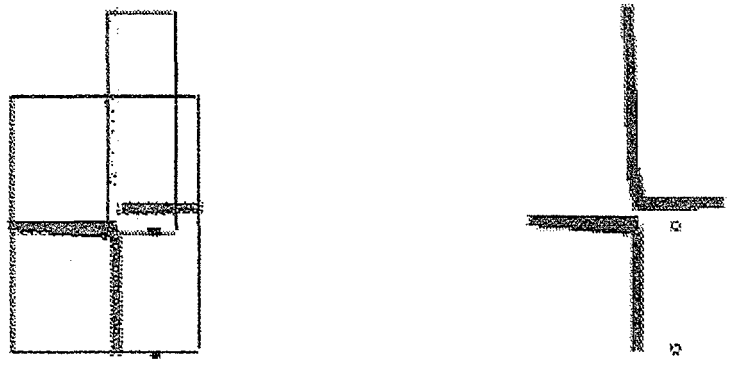


Fig. 7.11: Mapa generado desde la posición en la que parte el robot

El robot avanza desde esta posición hasta una posición 2. Los datos odométricos proporcionados por los sensores internos son $\xi_2 = (0.0, 2.484, 0.0)$, donde la traslación viene dada en metros y la rotación en radianes. En esta nueva posición el robot adquiere información del entorno y genera el mapa local g_2. La única información de que dispone el robot para corregir su posición es el mapa generado en el instante anterior g_1. Como los mapas g_1 y $g_{(2,1)}$ se solapan (ver figura 7.12) se aplica el algoritmo de corrección de la posición. Este algoritmo proporciona el siguiente valor estimado de la posición con respecto a la posición 1: $\widehat{X}_L^{(1,2)} = (0.008, 2.502, 0.0246)$. En este caso el valor de la posición del robot con respecto a la posición inicial $\widehat{X}_2$ coincide con $\widehat{X}_L^{(1,2)}$. El conjunto de mapas con los que se solapa el mapa obtenido en la posición 2 es $MS_2 = \{g_1\}$.

El robot avanza hasta la posición 3 ($\xi_3 = (0.0, 4.986, 0.0)$). El nuevo mapa $g_{(3,2)}$ se solapa con el mapa g_2 (ver figura 7.13). Aunque los campos de visión de los mapas calculados en las posiciones 1 y 3 se solapan (ver figura 7.13(a)), es decir $\mathbb{CV}_1 \bigcap \mathbb{CV}_3^1 \neq \emptyset$, e incluso lo hacen también algunos puntos de ambos mapas, de g_1 y de $g_{(3,1)}$, no se aplica entre ellos el algoritmo de localización debido a que son muy pocos los puntos equivalentes para contrastar la información y si se aplica el algoritmo de corrección se pueden producir grandes errores en la corrección de la posición. El valor de la posición 3 corregida utilizando la información del mapa 2 es



(a) Campos de visión de los mapas obtenidos en las posición 1 y 2

(b) Posiciones estimadas (círculo rojo) y posiciones calculadas de los datos odométricos (círculo negro)

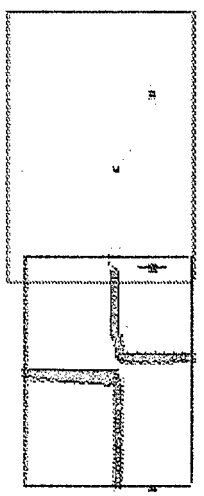
Fig. 7.12: Corrección *hacia adelante*: posición 2

$\widehat{X}_L^{(2,3)} = (0.0587, 2.496, 0.0027)$. El valor de la posición 3 referida al sistema de referencia global es $\widehat{X}_3 = (0.0024, 4.982, 0.0516)$. El conjunto de mapas con los que se solapa el mapa 3 es $MS_3 = \{g_2\}$. Como $MS_3 \subset MS_2 + g_2$ el robot no vuelve a pasar por una zona de la que ya tiene información, es decir, no se cierra un ciclo.

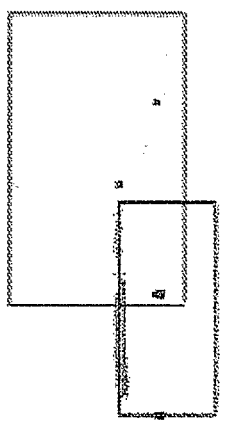
El robot sigue avanzando realizando el mismo proceso que anteriormente. Desde la figura 7.14 a la figura 7.21 se muestra el desplazamiento del robot desde la posición 4 hasta la posición 11. El círculo de color negro representa la posición del robot considerando únicamente la información de la odometría. El círculo en color rojo representa la posición del robot aplicando el algoritmo de corrección de la posición.

Considerando únicamente la información odométrica la posición 11 con respecto al sistema de referencia global queda $X_{11} = (0.15, 24.21, 0.0245)$. Aplicando el algoritmo de corrección el valor de esta posición queda $\widehat{X}_{11} = (-1.02, 24.15, 0.058)$. Se observa que hay una corrección importante en el eje x de aproximadamente 1 metro. La corrección en orientación es de aproximadamente de 2^o. En ningún desplazamiento el robot ha vuelto a pasar por una zona de la que ya disponía información, por lo que no ha sido necesario aplicar el algoritmo *hacia atrás*.

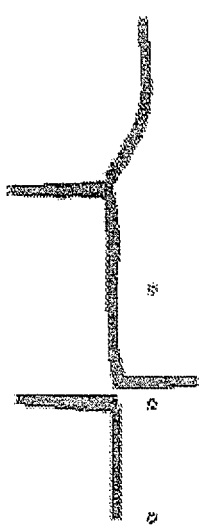
Si el robot hubiera vuelto a pasar por una zona conocida se aplicaría



(a) Campos de visión de los mapas obtenidos en las posición 1 y 3



(b) Campos de visión de los mapas obtenidos en las posición 2 y 3



(c) Posiciones estimadas (círculo rojo) y posiciones calculadas de los datos odométricos (círculo negro)

Fig. 7.13: Corrección *hacia adelante*: posición 3

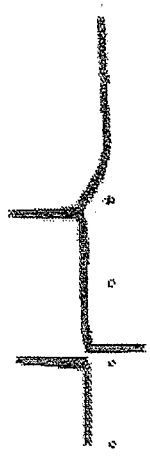


Fig. 7.14: Corrección *hacia adelante*: posición 4

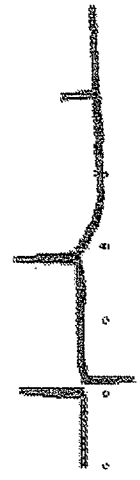


Fig. 7.15: Corrección *hacia adelante*: posición 5

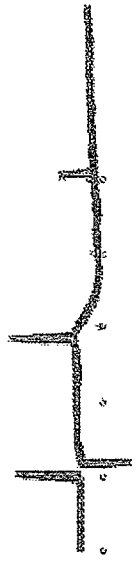


Fig. 7.16: Corrección *hacia adelante*: posición 6

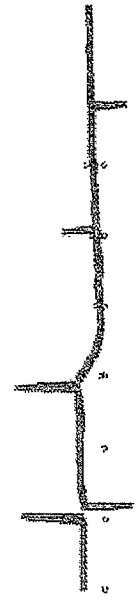


Fig. 7.17: Corrección *hacia adelante*: posición 7

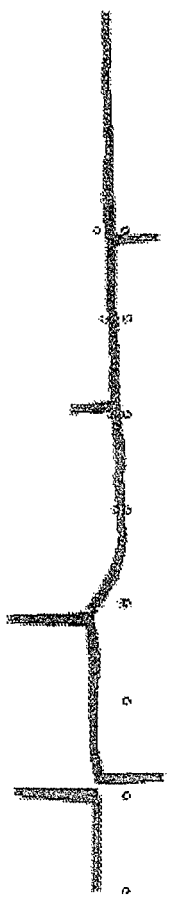


Fig. 7.18: Corrección *hacia adelante*: posición 8

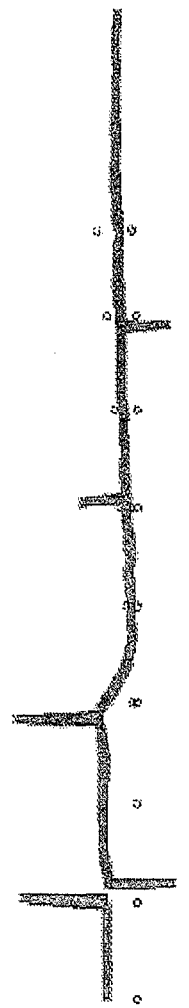


Fig. 7.19: Corrección *hacia adelante*: posición 9

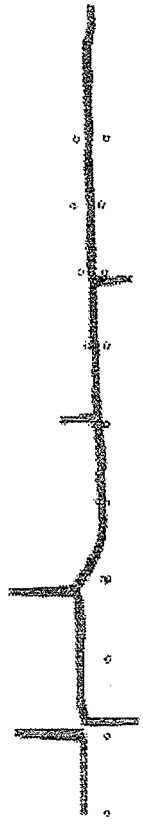


Fig. 7.20: Corrección *hacia adelante*: posición 10

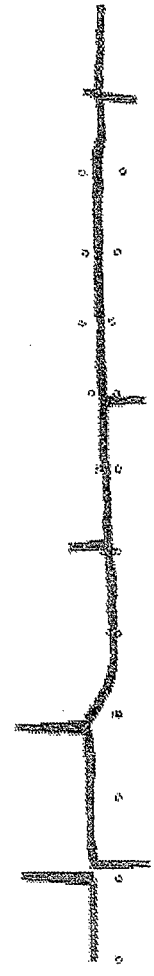


Fig. 7.21: Corrección *hacia adelante*: posición 11

a continuación el algoritmo *hacia atrás* corrigiendo no solamente la última posición en la que se encuentra el robot, sino también se corregirían las posiciones anteriores. En la sección siguiente se explica este algoritmo.

7.3.1.2 Corrección *hacia atrás*

El algoritmo *hacia adelante* estima la posición del robot incrementalmente, sin embargo, este algoritmo falla cuando el robot cierra un ciclo o cuando atraviesa un entorno cíclico. Esto es debido a que este algoritmo no revisa las estimaciones pasadas a medida que recibe nueva información sensorial.

Cuando se cierra un ciclo, el error del robot puede ser arbitrariamente grande, requiriéndose una corrección *hacia atrás* en el tiempo para generar un mapa consistente. El algoritmo presentado en esta tesis es capaz de revisar las posiciones estimadas en instantes de tiempo anteriores.

El algoritmo *hacia atrás* únicamente se ejecuta cuando se cierra un ciclo (ver prop. 7.3.2), en este caso es necesario realizar una corrección de las posiciones estimadas en instantes de tiempo anteriores.

El conjunto MS_i para cada mapa local se obtiene durante el algoritmo *hacia adelante*, almacenando esta información para reducir los tiempos de cálculo.

Una vez que un ciclo se cierra se ejecuta el algoritmo *hacia atrás*. Este algoritmo calcula las m mejores soluciones que se obtienen al ejecutar el algoritmo de contraste entre el mapa local actual g_i y los mapas locales que, perteneciendo a MS_i, no pertenecen a $MS_{i-1} + g_{i-1}$.

Para reducir el tiempo de cálculo durante el algoritmo *hacia adelante* también se almacenan las n mejores soluciones que se obtienen al ejecutar este algoritmo. Para cada mapa local g_i se almacenan las n mejores soluciones: $\widehat{\mathbb{X}}_L^{(j,i)}$ con $j = 1, \cdots, i-1$. Algunas de estas soluciones son modificadas al ejecutar el algoritmo *hacia atrás*.

Definición 7.3.4: Se define $\widehat{\mathbb{X}}_L^{(j,i)}$ al conjunto de soluciones de la posición del instante de tiempo i que hacen que el valor de la salud f sea máxima al contrastar el mapa $g_{(i,j)}$ con el mapa g_j con respecto al sistema de coordenadas situado en el origen del mapa local g_j :

$$\widehat{\mathbb{X}}_L^{(j,i)} = \{\widehat{X}_{L,1}^{(j,i)}, \cdots, \widehat{X}_{L,n}^{(j,i)}\} \text{ con } j = 1, \cdots, i-1 \tag{7.21}$$

Las soluciones almacenadas en el algoritmo *hacia atrás* son más que las almacenadas en el algoritmo *hacia adelante*. Esto es debido a que la incertidumbre espacial es mayor con respecto a los mapas con los que se ejecuta el

algoritmo *hacia atrás*, que con respecto a los mapas con los que sólo se ejecuta el algoritmo *hacia adelante*, ya que, el espacio recorrido por el robot para volver a una zona que ya ha sido observada es mayor que el espacio recorrido con respecto a las posiciones con las que sólo se ejecuta el algoritmo *hacia adelante*.

Las soluciones obtenidas para la posición en la que se cierra un ciclo son transformadas al sistema de coordenadas global $\widehat{\mathbb{X}}_T = \{\widehat{X}_T^1, \cdots, \widehat{X}_T^n\}$. A partir de los valores de estas posiciones se realiza una corrección *hacia atrás* de las posiciones anteriores. El nuevo valor de la posición anterior se obtiene mediante la expresión:

$$\widehat{X}_{i-1} = -T(\widehat{\theta}_i) \cdot T(-\widehat{\theta}_{L,k}^{(i-1,i)}) \cdot \widehat{X}_{L,k}^{(i-1,i)} + \widehat{X}_i \tag{7.22}$$

Esta ecuación es aplicable si el mapa del instante i se solapa con el mapa obtenido en el instante de tiempo justamente anterior $i-1$. Pero puede ocurrir que el mapa g_i no se solape con el mapa anterior sino que lo haga con un mapa obtenido en instantes anteriores y por tanto el valor que se obtiene a partir del algoritmo de corrección sea $\widehat{X}_{L,k}^{(n,i)}$ y no $\widehat{X}_{L,k}^{(i-1,i)}$. Se considera que g_n es el mapa local obtenido en el instante n que se solapa con el mapa g_i, tal que, $n \neq (i-1)$. A partir del algoritmo de contraste se obtienen los mejores valores de las posiciones estimadas en el instante i con respecto a la posición n, $\widehat{\mathbb{X}}_L^{(n,i)} = \{\widehat{X}_{L,1}^{(n,i)}, \cdots, \widehat{X}_{L,k}^{(n,i)}\}$. Las ecuaciones para calcular las nuevas posiciones corregidas del instante $i-1$ a partir de estas posiciones corregidas son las que se muestran a continuación. Primeramente se calcula las nuevas posiciones corregidas del instante n:

$$\widehat{X}_n = -T(\widehat{\theta}_i) \cdot T(-\widehat{\theta}_{L,k}^{(n,i)}) \cdot \widehat{X}_{L,k}^{(n,i)} + \widehat{X}_i \tag{7.23}$$

Y a partir de ellas se calcula las posiciones corregidas en el instante $t = i-1$:

$$\widehat{X}_{i-1} = T(\widehat{\theta}_n) \cdot [-T(\widehat{\theta}_{L,k}^{(n,i)}) \cdot T(-\widehat{\theta}_{L,m}^{(i-1,i)}) \cdot \widehat{X}_{L,m}^{(i-1,i)} + \widehat{X}_{L,k}^{(n,i)}] + \widehat{X}_n \tag{7.24}$$

El problema que surge es determinar los valores de las posiciones del instante $t = i$ con respecto a la posición anterior $t = i-1$, es decir, $\widehat{X}_{L,m}^{(i-1,i)}$. Si entre los mapas locales de ambos instantes no hay un solapamiento y por tanto el algoritmo de contraste no proporciona ningún resultado, una solución es considerar la información odométrica, es decir, $\widehat{X}_{L,1}^{(i-1,i)} = \xi_i$.

Otro de los problemas que surgen es si el mapa local g_i no se solapa con ningún mapa local anterior. En este caso se utiliza la ecuación 7.22 considerando que, el conjunto $\widehat{\mathbb{X}}_L^{(i-1,i)}$ está formado por un conjunto de soluciones

que están dentro del espacio de incertidumbre E_i^{i-1} y separadas entre ellas por unos valores de resolución igual a los utilizados en el AG.

A continuación se describen los pasos a seguir para llevar a cabo la localización *hacia atrás.*

7.3.1.2.1 Algoritmo hacia atrás. Los pasos del algoritmo *hacia atrás* son:

Entrada: Las entradas son:

- Las n mejores soluciones estimadas por el algoritmo *hacia adelante* en el instante de tiempo T: $\widehat{\mathbb{X}}_T$.
- El conjunto de mapas con los que se solapa cada uno de los mapas: MS_i con $i = 1, \cdots, T$.
- El conjunto de los valores de las incertidumbres para cada uno de los mapas: E_i^j con $i = 2, \cdots, T$ y $j = 1, \cdots, T-1$).
- El conjunto de las mejores soluciones locales para cada uno de los mapas: $\widehat{\mathbb{X}}_L^{(j,i)}$ con $i = 2, \cdots, T$ y $j = 1, \cdots, T-1$).

Salida: Las mejores posiciones estimadas para cada uno de los mapas locales: $\widehat{\mathbb{X}}_i = \{\widehat{X}_i^1, \cdots, \widehat{X}_i^n\}$ con $i = 1, \cdots, T-1$.

Paso 1: Para $i = T, \cdots, 2$ calcular las posiciones anteriores:

- Aplicando la ecuación 7.22 para calcular la posición anterior si el mapa del instante $t = i$ coincide con el mapa en el instante $t = i - 1$.
- Aplicando la ecuación 7.24 para calcular la posición anterior si el mapa del instante $t = i$ coincide con el mapa en el instante $t = j$ con $j < i$ y $j \neq (i-1)$.

Paso 2: Comprobar si la posición actual estimada $\widehat{X}_i$ está dentro del espacio de búsqueda E_i^1. Es decir, si la posición estimada del instante i está dentro de la zona de incertidumbre con respecto a la primera posición.

Paso 3.1: Si está dentro del espacio de búsqueda, seguir con las posiciones posteriores: $i = i - 1$ y volver al paso 1.

Paso 3.2: Si está fuera del espacio de búsqueda:

Paso 3.2.1: Elegir otra posición calculada en el paso 1 del instante de tiempo i, y volver al paso 2.

Paso 3.2.2: Si el número de posiciones es igual a cero hacer $i = i + 1$:

Paso 3.2.2.1: Si el número de posiciones actuales estimadas del instante i es igual a 0, volver al paso 3.2.2.

Paso 3.2.2.2: Si el número de posiciones actuales estimadas del instante i es distinto de 0, volver al paso 1.

Fin del ciclo : Cuando el número de posiciones del instante T sea igual a cero.

Estas nuevas posiciones estimadas se consideran como soluciones buenas si el bucle se ejecuta hasta la posición 2. Se considera que la primera posición (de la que parte el robot) no tiene error y por tanto solamente tiene un valor posible que es $X_1 = (0, 0, 0)$.

Puede ocurrir que el algoritmo *hacia atrás* no dé solamente una solución posible. En este caso se considera como solución los valores de las posiciones que cierran un ciclo y que tengan un valor de salud mayor. Este valor de salud se calcula sumando los valores de salud f_i obtenidos a partir del algoritmo de correción para cada nueva posición.

7.3.1.2.2 Resultados experimentales Se continúa con el ejemplo de la sección 7.3.1.1.2. El robot sigue avanzando por el pasillo corrigiendo su posición mediante el algoritmo *hacia adelante*. En la posición 44 el conjunto de mapas con los que se solapa el mapa topo-geométrico local g_{44} es $MS_{44} = \{g_{42}, g_{43}\}$. En esta posición el robot gira 180^o y adquiere información del entorno generando el mapa topo-geométrico local g_{45}. La posición del robot se corrige utilizando el algoritmo *hacia adelante*. Para mejorar la eficiencia del AG no se guarda únicamente la mejor solución obtenida cada vez que se ejecuta el algoritmo *hacia adelante*, sino que se almacenan las n mejores soluciones obtenidas cada vez que se ejecuta el AG, en la práctica el número de soluciones que se almacenan es igual a 3. En la figura 7.22 se muestran las mejores soluciones que se obtienen en cada posición. Para generar el mapa topo-geométrico global únicamente se empleará aquella solución de cada posición que tiene un valor de salud mayor (ver figura 7.23).

El mapa topo-geométrico local g_{45} se solapa con los mapas $MS_{45} = \{g_{33}, g_{34}, g_{35}, g_{36}, g_{37}, g_{38}, g_{39}, g_{40}, g_{41}, g_{42}, g_{44}\}$. Como no se cumple que $MS_{45} \subseteq MS_{44} + g_{44}$, implica que el robot vuelve a observar una zona del espacio que ya había sido observada anteriormente, por lo que se ejecuta el algoritmo *hacia atrás*. Se calcula de nuevo las n mejores soluciones con cada

Fig. 7.22: Mejores soluciones

Fig. 7.23: Mejor solución

uno de los mapas de MS_{45} que no coinciden con MS_{44}, en este caso, con $g_{33}, g_{34}, g_{35}, g_{36}, g_{37}, g_{38}, g_{39}, g_{40}$ y g_{41}. En el algoritmo *hacia atrás* se almacenan más soluciones que en el algoritmo *hacia adelante* debido a que la incertidumbre espacial es mayor, ya que, el espacio recorrido por el robot para regresar a una zona ya observada es mayor que el espacio recorrido con respecto a las posiciones donde se observan los mapas con los que únicamente se ejecuta el algoritmo *hacia adelante*. Por ejemplo, puede ocurrir que para un mapa obtenido en una posición, al intentar solaparlo con un mapa obtenido en otra posición se tenga no una única solución buena sino que se tenga varias soluciones buenas como se muestra en la figura 7.24. En la práctica se almacenan como máximo las 40 mejores soluciones.

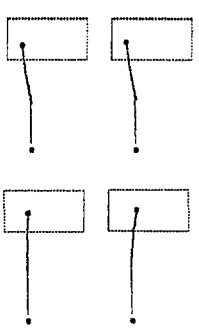


Fig. 7.24: Posibles soluciones buenas

En la figura 7.25 se muestran las mejores soluciones del mapa g_{45} con respecto al mapa g_{33}, es decir $X_{L,m}^{(33,45)}$. En la figura 7.26 se muestran estas soluciones referidas ya al sistema de referencia global $X_{(45,n)}^{33}$, los puntos en color rojo son las mejores soluciones de la posición 33, y los puntos en color azul las mejores soluciones de la posición 45. Con cada una de las soluciones de la posición 45 se ejecuta el algoritmo *hacia atrás*. De la figura 7.27(a) a la figura 7.29(o) se muestran las posiciones que se obtienen aplicando el algoritmo *hacia atrás*.

Fig. 7.25: Mejores soluciones de la posición 45 con respecto a la posición 33

Fig. 7.26: Mejores soluciones de la posición 45 con respecto a la posición 33 referidas al sistema de referencia global (primera posición)

Se parte de una de las soluciones obtenidas para las posición 45 (ver figura

7.27(a)), y se empieza a calcular las nuevas posiciones para $i = 44$ según la expresión 7.22, debido a que el mapa g_{45} se solapa con el mapa anterior g_{44}, utilizando las mejores soluciones obtenidas en el algoritmo *hacia adelante*: $\widehat{\mathbb{X}}_{L,n}^{(44,45)}$. Las nuevas posiciones para $i = 44$ serán válidas si están dentro del espacio de incertidumbre E_{44}^{1}. En el caso de que no existiera ninguna posición válida para $i = 44$ se seleccionaría una nueva posición de $i = 45$.

Si existen nuevas soluciones válidas para $i = 44$, se selecciona una de las nuevas posiciones obtenidas para $i = 44$ (ver figura 7.27(b)) y se repite el mismo proceso que para $i = 45$. Si se obtuviera nuevas posiciones válidas para $i = 43$ se continuaría con el resto de posiciones anteriores repitiendo el proceso, pero si no se obtuviera ninguna posición válida se continuaría con alguna de las posiciones válidas obtenidas anteriormente.

Dos posibles situaciones por las que una solución nueva no es válida son:

1. La solución corregida para una posición determinada se encuentra fuera de su espacio de incertidumbre.

2. Los mapas topo-geométricos locales que se solapan al aplicar el algoritmo *hacia adelante* no lo hacen cuando se ejecuta el algoritmo *hacia atrás*. En la figura 7.30 se muestra un ejemplo de este caso, donde se puede observar que el mapa topo-geométrico local g_{45} no se solapa con el mapa g_{33} (mapas en color rojo).

En la figura 7.29(o) se muestra una de las soluciones generadas. Se observa que todas las posiciones nuevas calculadas se encuentran dentro de su correspondiente espacio de incertidumbre referido a la posición primera. Esta solución es peor que la obtenida para el algoritmo *hacia adelante* (ver figura 7.23), pero que hay que considerar que esta es una de las soluciones que se obtienen. También hay que tener en cuenta, que al utilizar los AG para estimar la posición, no se considera todo el espacio de búsqueda, lo que hace que el algoritmo sea más rápido que si se considerara todo el espacio de búsqueda, pero en contraoposición se obtienen soluciones un poco distorsionadas. Esto tampoco es un inconveniente, ya que una vez que se ha obtenido una solución aceptable se puede aplicar de nuevo el algoritmo de corrección de la posición con un valor de resolución menor para obtener mejores resultados.

Al aplicar el algoritmo *hacia atrás* no se tiene porque obtener una única solución, sino que se pueden obtener varias soluciones. Para generar el mapa global, se emplea aquella solución para la que se obtiene un mejor solapamiento entre los mapas locales referidos a las nuevas posiciones.

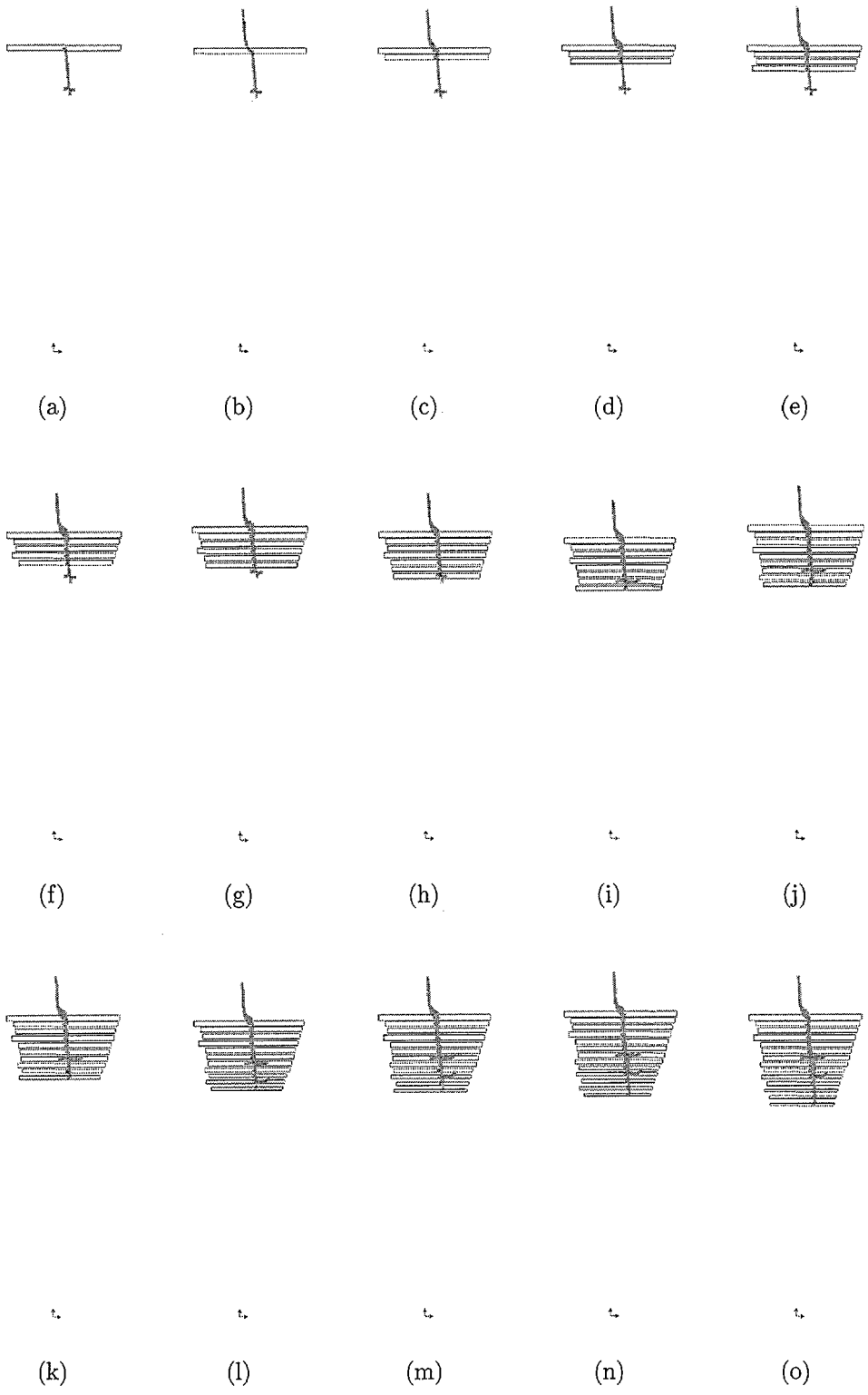


Fig. 7.27: Algoritmo *hacia atrás* desde la posición 45 a la posición 1

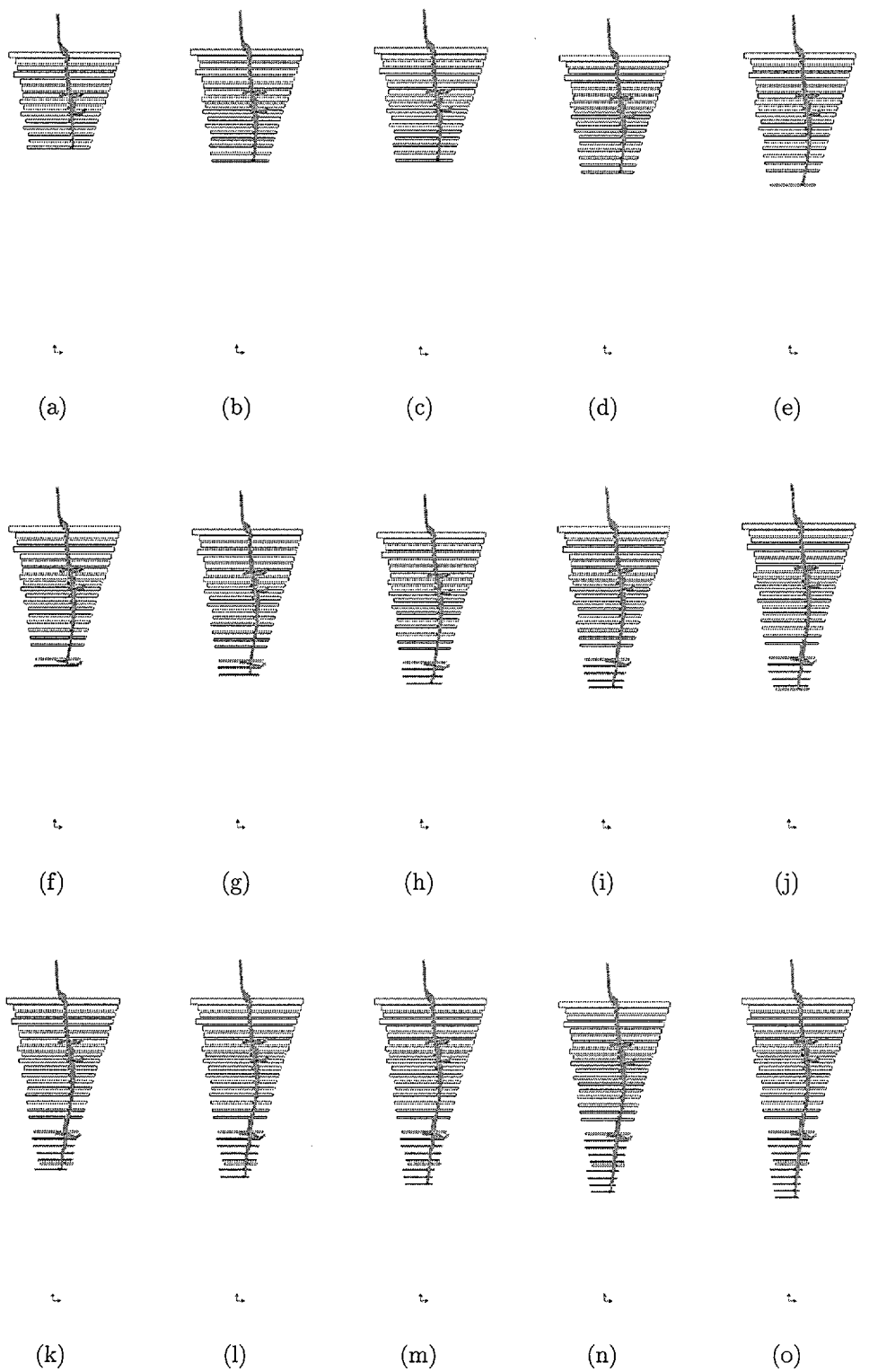


Fig. 7.28: Algoritmo *hacia atrás* desde la posición 45 a la posición 1 (continuación)

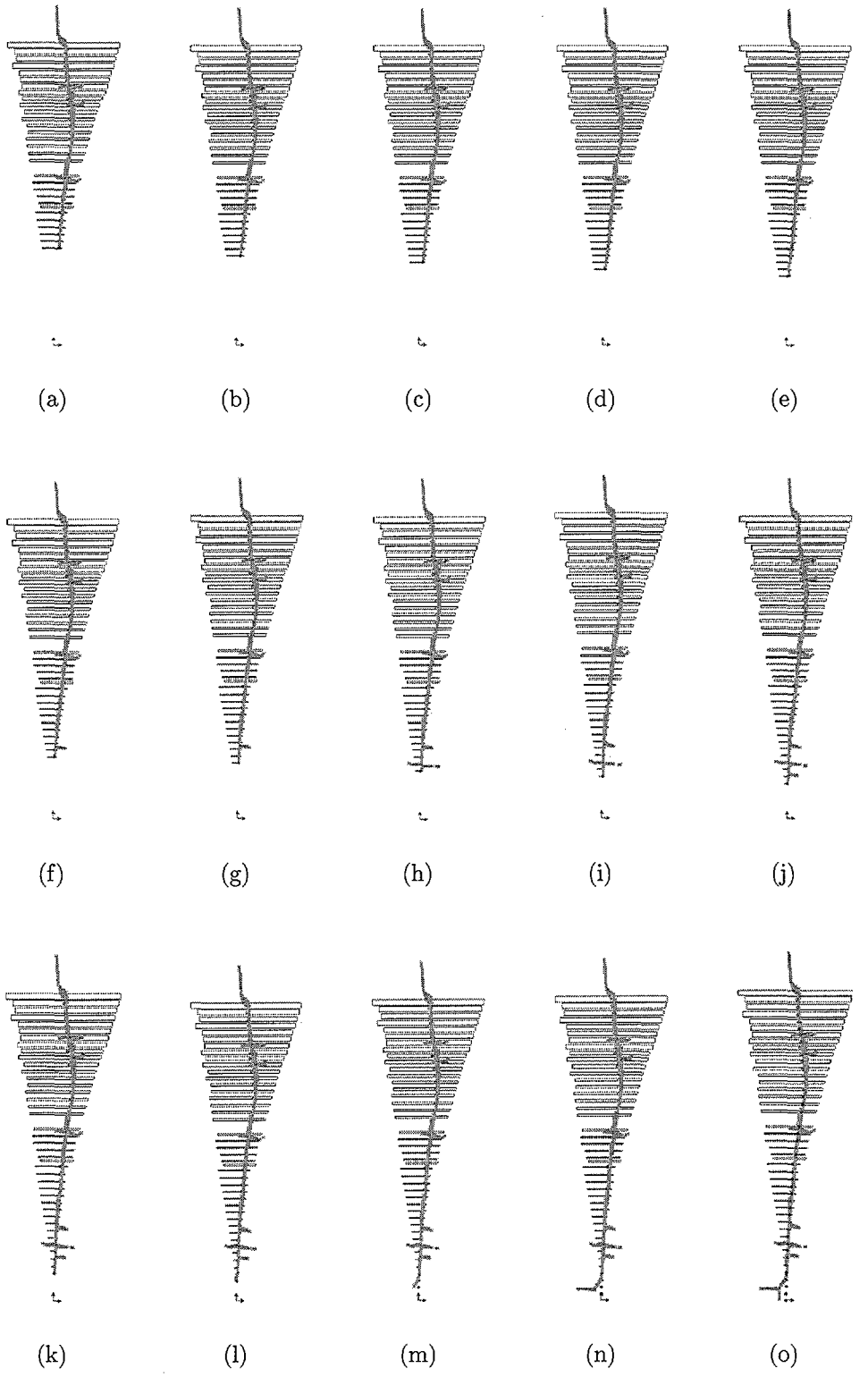


Fig. 7.29: Algoritmo *hacia atrás* desde la posición 45 a la posición 1 (continuación)

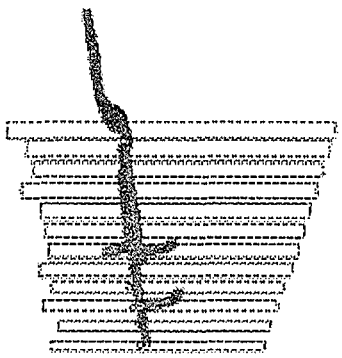


Fig. 7.30: Caso en el que dos mapas topo-geométricos locales no se solapan al ejecutar el algoritmo *hacia atrás*

7.3.2 Integración del mapa local

Una vez que se estima la posición del robot en un instante de tiempo dado, se integra el mapa local obtenido en ese instante con el resto de mapas para generar el mapa topo-geométrico global.

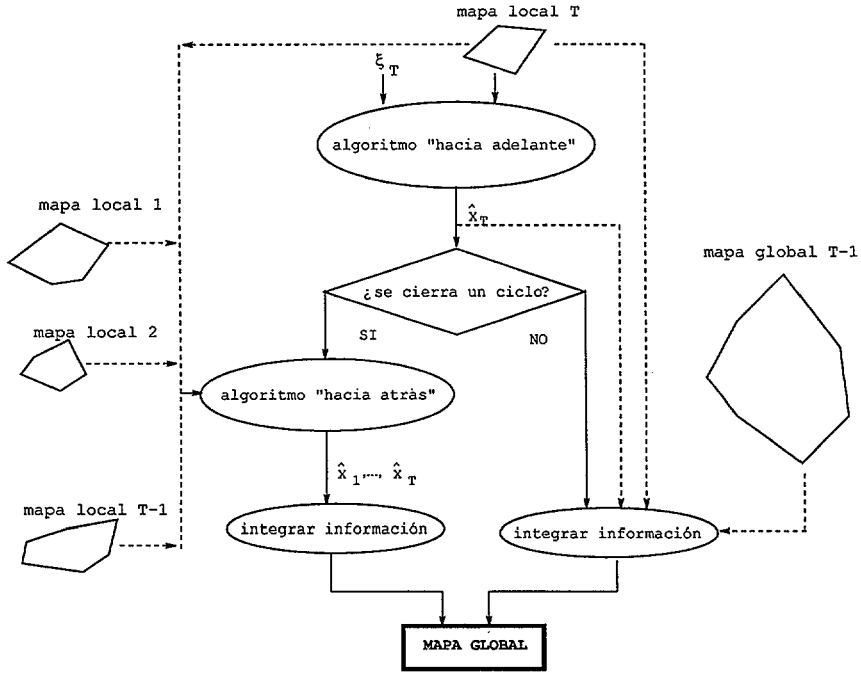


Fig. 7.31: Proceso de integración de los mapas para dar lugar al mapa global

Sea g_T el mapa topo-geométrico local del instante de tiempo dado T y $\widehat{X}_T$ su posición estimada. Si no se ha cerrado ningún ciclo entonces el algoritmo *hacia atrás* no se ha ejecutado y las posiciones anteriores estimadas $\widehat{X}_1, \cdots, \widehat{X}_{T-1}$ no se han corregido. El mapa global del instante T, M_T, se genera integrando la información del mapa local $g_{(T,1)}$ con la información del mapa global M_{T-1}.

Sin embargo, si el algoritmo *hacia atrás* se ha ejecutado, las posiciones anteriores han sido corregidas. En este caso el mapa global M_T se calcula integrando de nuevo todos los mapas locales considerando sus posiciones corregidas.

En el instante de tiempo inicial $t = 0$, el mapa global coincide con el mapa local: $M_1 = g_1$. El robot avanza hasta la posición $\widehat{X}_2$ que ha sido estimada aplicando el algoritmo *hacia adelante* y si se requiere también el

algoritmo *hacia atrás*. A continuación se integra la información de ambos mapas, realizando previamente un cambio de coordenadas con respecto al sistema de referencia global (punto de partida) del mapa local actual (ver ecu. 7.8).

El robot se encuentra en una posición del espacio que ha sido estimada. Las ramas y nodos del mapa global, que son vistos desde la posición actual del robot, son los que pueden coincidir con alguna rama o nodo del mapa local actual referido al sistema de coordenadas global.

Proposición 7.3.3: Sea $M_{T-1} = \{\sum Nodo_{T-1,i}, \sum Rama_{T-1,j}\}$ el mapa global calculado en el instante de tiempo $T-1$ y $\mathbb{CV}_T^1$ el campo de visión del mapa local g_T obtenido en el instante T con respecto a la posición 1, entonces una rama o nodo del mapa M_{T-1} pueden coincidir con alguna rama o nodo del mapa local $g_{(T,1)}$ si se cumple que $Rama_{T-1,j} \bigcap \mathbb{CV}_T^1 \neq \emptyset$ o que $Nodo_{T-1,i} \bigcap \mathbb{CV}_T^1 \neq \emptyset$.

Si no existe ningún nodo del mapa global que esté dentro del campo de visión del robot implica que ningún nodo del mapa local coincide con algún otro del mapa global. Los nodos del mapa local son entonces añadidos al nuevo mapa global. El nuevo nodo del mapa global tendrá por coordenadas la posición del nodo del mapa local referido al sistema de coordenadas global.

Por el contrario, si existe algún nodo del mapa global dentro del campo de visión este puede coincidir o no con un nodo del mapa local. Si no coincide se procede de la misma manera que en el caso anterior, pero si coincide se calcula la posición media de los dos nodos equivalentes (ver definición 6.2.3).

Si una rama del mapa global esta delimitada por uno o dos nodos que están dentro del campo de visión se comprueba si esta coincide con alguna rama del mapa local (ver definición 6.2.4). En caso afirmativo se integra la información de ambas ramas, es decir, la nueva rama generada tendrá información de las dos ramas anteriores.

Por último se añaden los nodos y ramas del mapa local que no han sido añadidos todavía al mapa global. Esto se debe a que no coinciden con ningún nodo o rama del mapa global.

Se producen entonces varios casos tanto para los nodos como para las ramas. En el caso de los nodos puede ocurrir:

- Un nodo del grafo global coincide con un nodo del nuevo grafo local.
- Un nodo del grafo global que esta dentro del campo de visión no coincide con ningún nodo del grafo local.

- Un nodo del grafo global esta fuera del campo de visión.
- Un nodo del grafo local no coincide con ninguno del grafo global.

En el caso de las ramas:

- Una rama del grafo global esta fuera del campo de visión.
- Una rama del grafo global esta dentro del campo de visión y coincide con una rama del grafo local.
- Una rama del grafo global esta dentro del campo de visión y no coincide con ninguna rama del grafo local.
- Una rama del grafo local no coincide con ninguna rama del grafo global.

La integración de información entre dos mapas se realiza únicamente entre los nodos y ramas equivalentes (ver sección 6.2.5). En las secciones siguientes se explica el mecanismo de integración de la información tanto de los nodos como de las ramas.

Debido a la resolución utilizada en la obtención del mapa, a los errores propios de las medidas y a que la posición corregida puede tener errores, la información tanto de los nodos como de las ramas equivalentes no tiene porque coincidir totalmente. Esto obliga a fusionar o integrar de alguna manera la información tanto de los nodos como de las ramas equivalentes.

7.3.2.1 Integración de nodos equivalentes

Sea N_i un nodo del mapa g_i cuyas coordenadas son $N_i = \{x_i, y_j\}$, y N_j un nodo del mapa g_j equivalente a N_i y cuyas coordenadas son $N_j = \{x_j, y_j\}$. El nuevo nodo que se genera en el mapa global es un nodo que integra la información de ambos nodos y tiene por coordenadas:

$$N_k = \{x_k, y_k\} = \{\frac{x_i + x_j}{2}, \frac{y_i + y_j}{2}\} \tag{7.25}$$

Se podría haber seleccionado otro método de integración, pero el método del promedio es muy sencillo de implementar y con él se han obtenido buenos resultados.

Sea DC_i (ver definición 4.2.13) la distancia desde el nodo N_i a los objetos equidistantes y DC_j la distancia desde el nodo N_j a los objetos equidistantes. La distancia para el nuevo nodo generado es:

$$DC_k = \frac{DC_i + DC_j}{2} \tag{7.26}$$

7.3.2.2 Integración de ramas equivalentes

Sea R_i una rama del mapa g_i cuyos puntos que la definen son $R_i = \{p_{i1}, \cdots, p_{in}\}$ y R_j una rama del mapa g_j equivalente a R_i cuyos puntos que la forman son $R_j = \{p_{j1}, \cdots, p_{jm}\}$. Se supone que los puntos de ambas ramas coincidentes son $R_i^{equi} = \{R_{ik}, \ldots, R_{il}\}$ y $R_j^{equi} = \{R_{jp}, \ldots, R_{jq}\}$.

Proposición 7.3.4: Los puntos de la rama $R_i \in g_{(i,j)}$ y los puntos de la rama $R_j \in g_j$, donde R_i y R_j son dos ramas equivalentes, son coincidentes si pertenecen a la zona del espacio definida por $\mathbb{CV}_i^j \bigcap \mathbb{CV}_j$.

La nueva rama generada del mapa global está formada por los puntos de ambas ramas que no son coincidentes y por los puntos coincidentes de la rama del mapa obtenido más recientemente.

$$R_k = p_{ir} \in R_i \mid p_{ir} \text{ no } \in R_i^{equi} \cup p_{jr} \in R_j \mid p_{jr} \text{ no } \in R_j^{equi} \cup R_i^{equi} \tag{7.27}$$

A la hora de integrar la información de dos ramas no se considera que pueda haber obstáculos o personas moviéndose por el entorno. Cuando se introduce un obstáculo en una zona, que no existía anteriormente, el mapa topo-geométrico local cambia. Estudiando la información tanto topológica como geométrica del nuevo mapa se puede saber si hay un nuevo obstáculo en el entorno, en este caso, la información de este mapa no se añade al mapa global, pero sí se tiene en cuenta para la navegación del robot.

Sin embargo, a la hora de fusionar la información de dos mapas si se ha considerado que puertas, que no estaban abiertas en un instante anterior, puedan estarlo en instantes posteriores. En este caso los nodos y la ramas que se puedan generar como consecuencia de que una puerta se ha abierto si se añaden al mapa global.

La distancia de cada punto que forma la nueva rama R_k a los objetos a los que son equidistantes es igual a la distancia de cada punto que pertenece a cada rama del mapa local correspondiente.

7.3.2.3 Resultados experimentales

Se considera que el robot comienza a explorar el entorno desde una posición desconocida a priori y que considera el origen de coordenadas del sistema global. En esta posición el robot genera el mapa topo-geométrico local que se muestra en la figura 7.32. El mapa global generado del instante $t = 1$

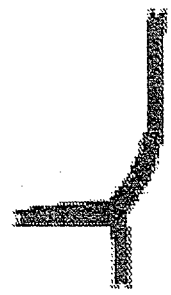


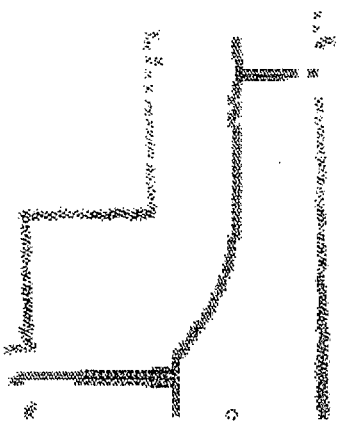
Fig. 7.32: Mapa topo-geométrico observado en la posición desde la que el robot comienza a explorar el entorno

coincide con el mapa local g_1, ya que, no se dispone de más información con la que comparar este mapa.

El robot se desplaza a una nueva posición X_2. El mapa local que se genera en esta posición se muestra en la figura 7.33(a). Aplicando el algoritmo de corrección de la posición se obtiene $\widehat{X}_2$. Los mapas $g_{(2,1)}$ y M_1 se muestran en la figura 7.33(b) para esta nueva posición. Como no se cierra ningún ciclo el proceso de construcción del mapa global consiste en integrar la información del mapa M_1 con la información del mapa $g_{(2,1)}$. Se observa que hay varias ramas y nodos equivalentes cuya información se fusiona. También se genera un nuevo nodo y dos nuevas ramas. El mapa global M_2 que se obtiene se muestra en la figura 7.33(c).

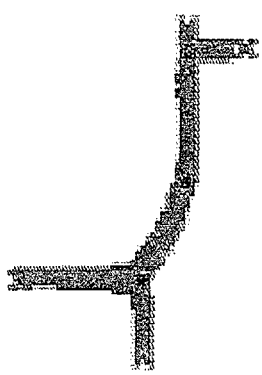
El robot sigue avanzando hasta una posición X_3, donde se genera el mapa local que se muestra en la figura 7.34(a). Se aplica el algoritmo de corrección de la posición para obtener $\widehat{X}_3$. El mapa M_2 obtenido en el instante anterior junto con el mapa $g_{(3,1)}$ calculado para la posición $\widehat{X}_3$ se muestra en la figura 7.34(b). La información tanto de las ramas como de los nodos equivalentes se fusiona para dar lugar al mapa global M_3 que se muestra en la figura 7.34(c). En este caso no se genera ningún nodo y rama nueva.

El robot sigue avanzando hasta una posición X_4, donde se genera el mapa local que se muestra en la figura 7.35(a). Se aplica el algoritmo de corrección de la posición para obtener $\widehat{X}_4$. El mapa M_3 obtenido en el instante ante-



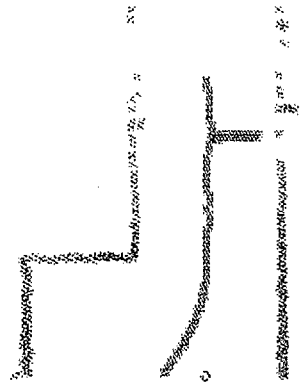
(a) DVL

(b) M_1 con $g_{(2,1)}$

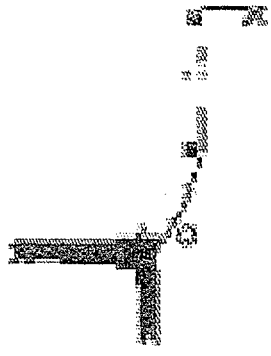


(c) M_2

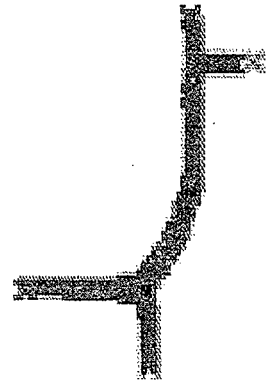
Fig. 7.33: Integración de la información del mapa global M_1 con la información del mapa local $g_{(2,1)}$



(a) DVL



(b) M_2 con $g_{(3,1)}$

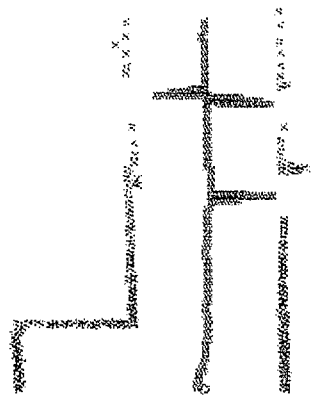


(c) M_3

Fig. 7.34: Integración de la información del mapa global M_2 con la información del mapa local $g_{(3,1)}$

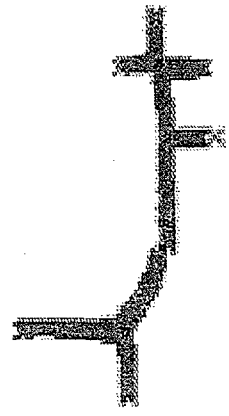
rior junto con el mapa $g_{(4,1)}$ calculado para la posición $\widehat{X}_4$ se muestra en la figura 7.35(b). Hay un nodo y varias ramas equivalentes cuya información se fusiona, y además se generan dos nuevos nodos y dos nuevas ramas.

En ningún desplazamiento, el robot ha vuelto a pasar por una zona del espacio de la que ya se tenía información. El proceso de integración de la información se realiza entonces fusionando el mapa global generado en el instante de tiempo anterior con el mapa local obtenido en la posición actual. Si un ciclo se hubiera cerrado se tendría que haber repetido este proceso desde el comienzo con los nuevos valores de las posiciones corregidas.



(a) DVL

(b) M_3 con $g_{(4,1)}$



(c) M_4

Fig. 7.35: Integración de la información del mapa global M_3 con la información del mapa local $g_{(4,1)}$

8. RESULTADOS EXPERIMENTALES

8.1 Introducción

En este capítulo se presentan los resultados experimentales obtenidos al aplicar el algoritmo de construcción de un mapa y localización de manera simultánea descrito en los capítulos anteriores. Un robot móvil recorre un entorno estructurado típico de interiores y construye un mapa, al mismo tiempo que corrige su posición.

Los resultados experimentales descritos en este capítulo han sido obtenidos en diferentes escenarios del edificio Betancourt de la Universidad Carlos III de Madrid utilizando el robot B21 (apéndice A) y el telémetro láser PLS-220 (apéndice B), haciendo que el robot se desplace por el entorno y que adquiera información del mismo a través del telémetro láser, cada vez que recorre uno o dos metros aproximadamente. La información odométrica, dada por los sensores internos, también es almacenada para mejorar el proceso de localización.

El robot B21 es un robot holónomo de la casa RWI (Real World Interface) equipado con sensores de ultrasonidos, de infrarrojos y de un telémetro láser. Los experimentos se han realizado utilizando únicamente el telémetro láser, ya que, tiene más precisión que los otros dos tipos de sensores. El telémetro láser es de la casa SICK y proporciona una rango de medidas horizontales en 2D de 180^o con una resolución angular de 0.5^o.

Los algoritmos de construcción de los mapas topo-geométricos y corrección de la posición del robot han sido programados en C y C++ bajo el sistema operativo Linux. Para facilitar el manejo de los grafos generados y de la información geométrica que contienen, se han utilizado las librerías de LEDA [Max].

Los experimentos se han realizado en un microprocesador Pentium II a $400MHz$. El tiempo de cálculo para construir el mapa topo-geométrico local a partir de la información del telémetro láser está comprendido entre $400ms$

y $900ms$, dependiendo de la complejidad del entorno, con una resolución de celdilla de $5cm$ y un valor de la región visible limitada a $4m$. El tiempo de cálculo que se requiere al contrastar un mapa topo-geométrico local con otro, empleando AG con una población de 100 individuos y 50 generaciones, está comprendido entre $10sg$ y $30sg$ dependiendo de la complejidad del entorno. Cuanto mayor es el número de mapas con los que se solapa un mapa determinado mayor es el número de veces que se tiene que ejecutar el AG y, por tanto, mayor es el tiempo de cálculo. El algoritmo *hacia atrás* tarda en ejecutarse varias horas dependiendo del número de mapas con los que se solapa un mapa determinado.

Según los valores de los tiempos de ejecución se puede concluir que, el algoritmo de construcción del mapa topo-geométrico local se puede ejecutar al mismo tiempo que el robot se desplaza por el entorno, debido a que los tiempos de cálculo son pequeños. Sin embargo, los algoritmos de corrección de la posición y de construcción del mapa global se han de realizar off-line. Esto tampoco supone una gran desventaja, ya que, como se ha comentado anteriormente, el robot utiliza la información del mapa local y no la del mapa global para navegar por el entorno.

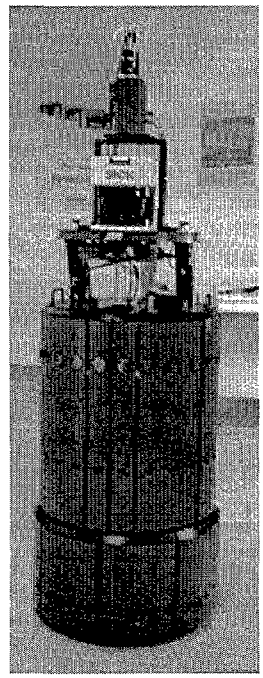


Fig. 8.1: Robot B21 de la casa RWI con el telémetro láser PLS-220 de SICK

Se han realizado varios experimentos para demostrar la consistencia del algoritmo propuesto:

- El robot recorre un pasillo varias veces (planta 3 del edificio Betancourt).
- El robot recorre un entorno cíclico de grandes dimensiones (planta 1 del edificio Betancourt).

8.2 Experimento 1: Construcción de un mapa topo-geométrico de un pasillo

El primer experimento consiste en que el robot recorra un pasillo varias veces y que genere un mapa topo-geométrico global del mismo, corrigiendo su posición en cada avance. Este pasillo se encuentra situado en la planta 3 del edificio Betancourt de la Universidad Carlos III de Madrid. El plano de la zona recorrida se representa en la figura 8.2. Imágenes reales de la planta 3 que recorre el robot, se muestran en la figura 8.3.

En la figura 8.4 se muestran los datos del telémetro láser obtenidos en cada avance del robot considerando únicamente la información odométrica. En la figura 8.5 se representan los mapas topo-geométricos locales generados en cada posición del robot considerando solamente la información odométrica. Se observa que si no se corrige la posición del robot se obtiene un mapa distorsionado del entorno. Es necesario, por tanto, aplicar el algoritmo SLAM desarrollado en esta tesis, para corregir la posición del robot y obtener un mapa que represente fielmente la topología del entorno recorrido.

El robot avanza por el pasillo adquiriendo información del entorno, a través del telémetro láser, cada vez que recorre uno o dos metros aproximadamente. El mapa global que se obtiene cuando el robot llega al final del pasillo, y que se genera por la fusión de los mapas locales que se obtienen en cada desplazamiento del robot, se muestra en la figura 8.6.

Hasta este momento el robot no ha vuelto a pasar por una zona de la que ya tenía información, por lo que sólo se ha aplicado el algoritmo *hacia adelante* para corregir su posición. Como no se han corregido las posiciones anteriores, el mapa global se genera fusionando el mapa local obtenido en el instante de tiempo actual con el mapa global obtenido en el instante anterior. El mapa topo-geométrico global generado está formado por un conjunto de nodos y ramas que caracterizan lugares típicos de interiores como son puertas, intersecciones y estrechamientos. Se observa que este mapa global no

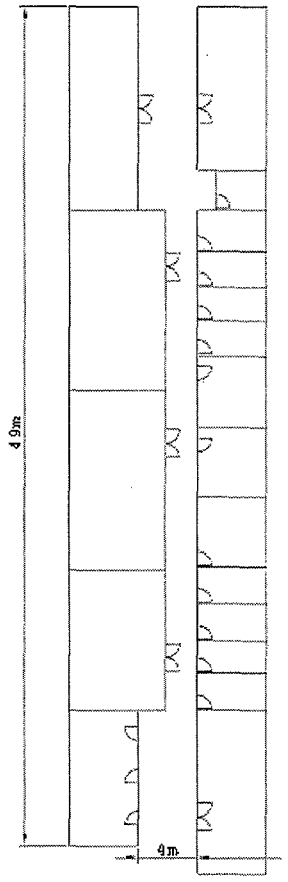
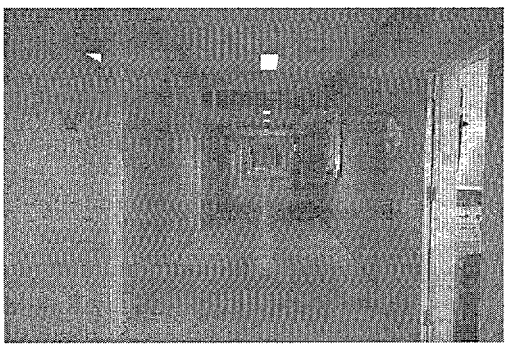


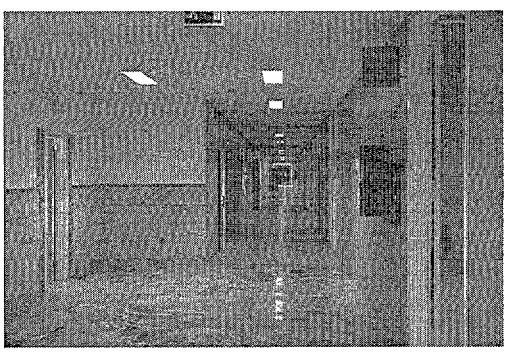
Fig. 8.2: Planta 3 del edificio Betancourt de la Universidad Carlos III de Madrid

representa totalmente la topología del mapa representado en la figura 8.2. Esto se debe a que, en este recorrido del robot no todas las puertas estaban abiertas y, por tanto, no han podido ser observadas por el telémetro láser. Si el robot vuelve a pasar por una zona y observa una puerta abierta, que no lo estaba anteriormente, la rama y nodo que la caracterizan se añaden al mapa topo-geométrico global.

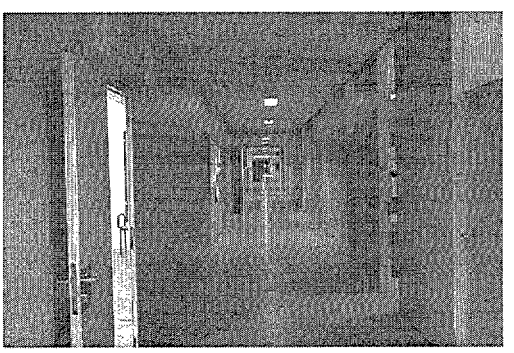
El robot, cuando llega al final del pasillo, gira 180° recorriendo de nuevo el pasillo. Como el robot vuelve a observar una zona de la que ya tiene información se empieza a aplicar el algoritmo *hacia atrás*. En el algoritmo *hacia atrás* se modifican todos los valores de las posiciones, por lo que el mapa global se genera integrando de nuevo la información de todos los mapas topo-geométricos locales obtenidos para las nuevas posiciones.



(a)



(b)



(c)

Fig. 8.3: Imágenes de la planta 3 del edificio Betancourt de la Universidad Carlos III de Madrid

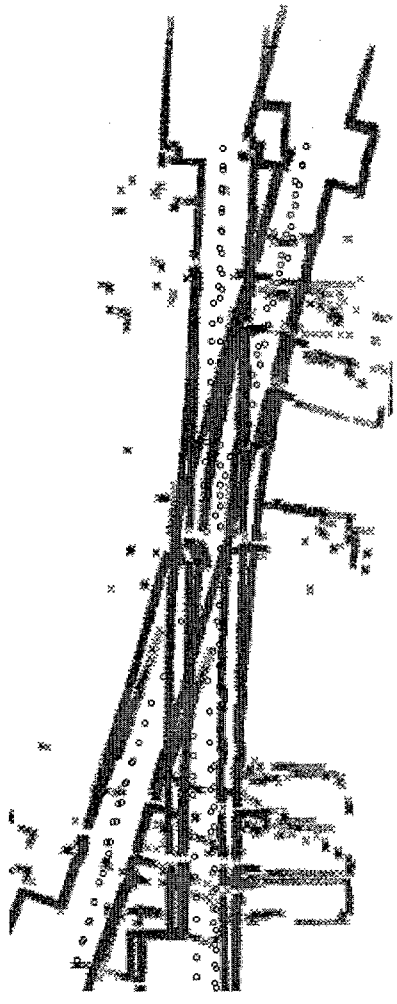


Fig. 8.4: Datos obtenidos del láser en la planta 3 del edificio Betancourt considerando solamente la información odométrica

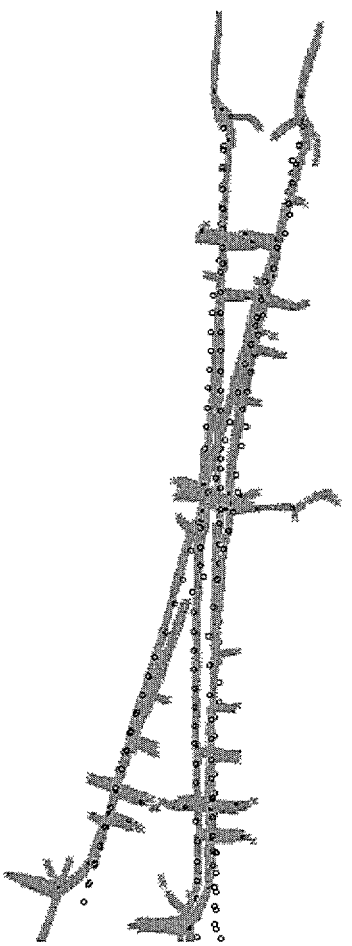


Fig. 8.5: Mapas topo-geométricos locales de la planta 3 del edificio Betancourt considerando solamente la información odométrica

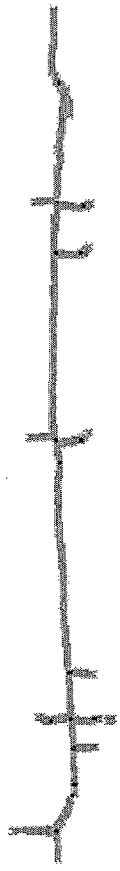


Fig. 8.6: Mapa global generado de la planta 3 cuando el robot recorre el pasillo

El robot, al recorrer de nuevo una zona de la que ya tiene información, ejecuta alternativamente el algoritmo *hacia adelante* y el algoritmo *hacia atrás* en cada desplazamiento. En la figura 8.7 se muestran las distintas soluciones que se obtienen al ejecutar el algoritmo *hacia atrás* en cada nuevo desplazamiento. Se observa que se obtienen varias soluciones válidas. Para generar el mapa global sólo se utiliza la solución en la que se obtiene un mejor solapamiento entre los mapa topo-geométricos locales obtenidos para cada posición. Este solapamiento se calcula sumando el valor de salud que se obtiene para cada nueva posición corregida.

El robot al recorrer de nuevo el pasillo, no solamente corrige las posiciones, sino que también modifica el mapa global que ha generado anteriormente. Puede ocurrir que el robot al desplazarse observe una intersección que no existe realmente, como ocurre en la figura 8.8(b). Esto se debe a que existe una puerta que se abre hacia el pasillo y que queda prácticamente paralela a la pared si se abre completamente, por lo que el robot considera que existe una puerta (ver figura 8.8(a)). En la figura 8.8(b) se representa en color verde el mapa topo-geométrico global generado hasta la posición en la que se encuentra el robot, y en color amarillo el mapa topo-geométrico local obtenido en la posición, ya corregida, en la que se encuentra el robot. El nuevo mapa global contendrá el nuevo nodo y la nueva rama que representan esta intersección falsa.

La generación de ramas y nodos falsos tampoco supone un gran inconveniente porque el robot al avanzar obtendrá una información más completa de esta zona del entorno y dejará de observar esta intersección falsa (ver figura 8.8(c)). En la figura 8.8(c) se muestra que el mapa local (mapa en color amarillo), generado en el nuevo avance del robot, ya no observa la intersección falsa. Hay que considerar que en la navegación, el robot utilizará la información del mapa topo-geométrico local obtenido en cada avance y no la del mapa topo-geométrico global.

El robot, al seguir avanzando por el pasillo, puede observar una puerta que anteriormente podía haber estado cerrada. Como consecuencia de ello se generan los nodos y ramas que la caracterizan. En la figura 8.9(a) se muestra el mapa topo-geométrico global (mapa en color verde) generado hasta la posición actual del robot, con el mapa topo-geométrico local (mapa en color amarillo) que se observa en la nueva posición corregida del robot. El robot observa una puerta que anteriormente no estaba abierta. El nodo y rama que la caracterizan son añadidos al mapa global (ver figura 8.9(b)).

El robot sigue avanzando por el pasillo, y observa el paso del pasillo al vestíbulo generándose un nodo que caracteriza este ensanchamiento (mapa de

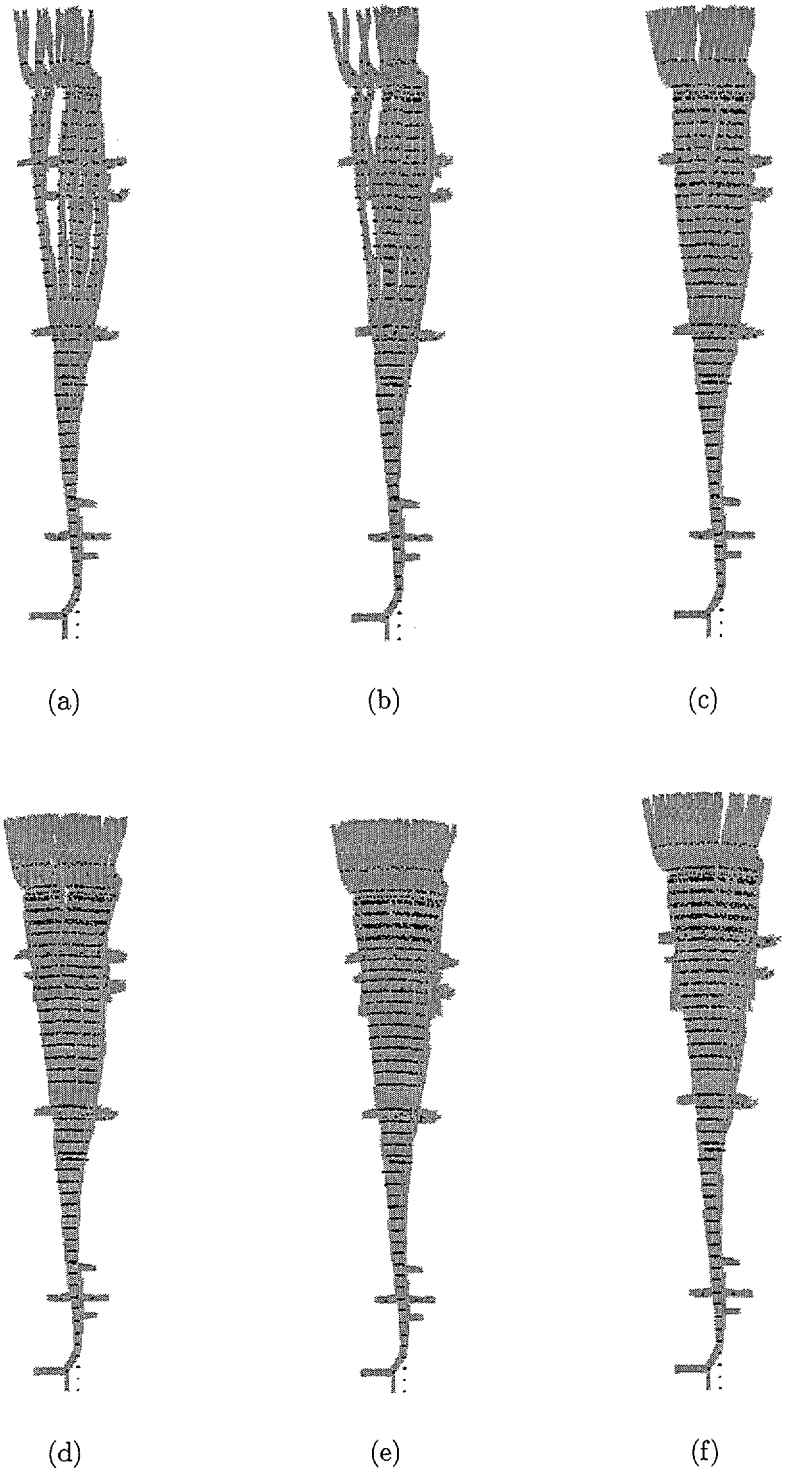
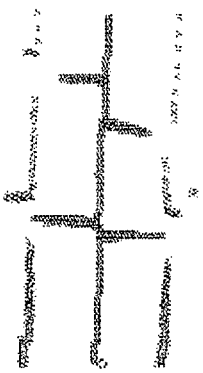
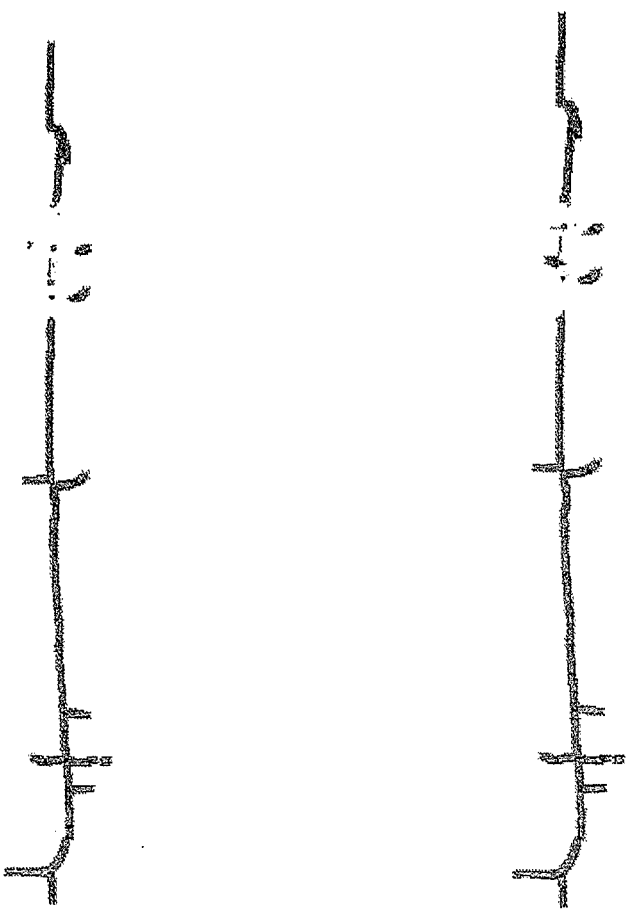


Fig. 8.7: Soluciones obtenidas al aplicar el algoritmo *hacia atrás* en cada nuevo desplazamiento del robot

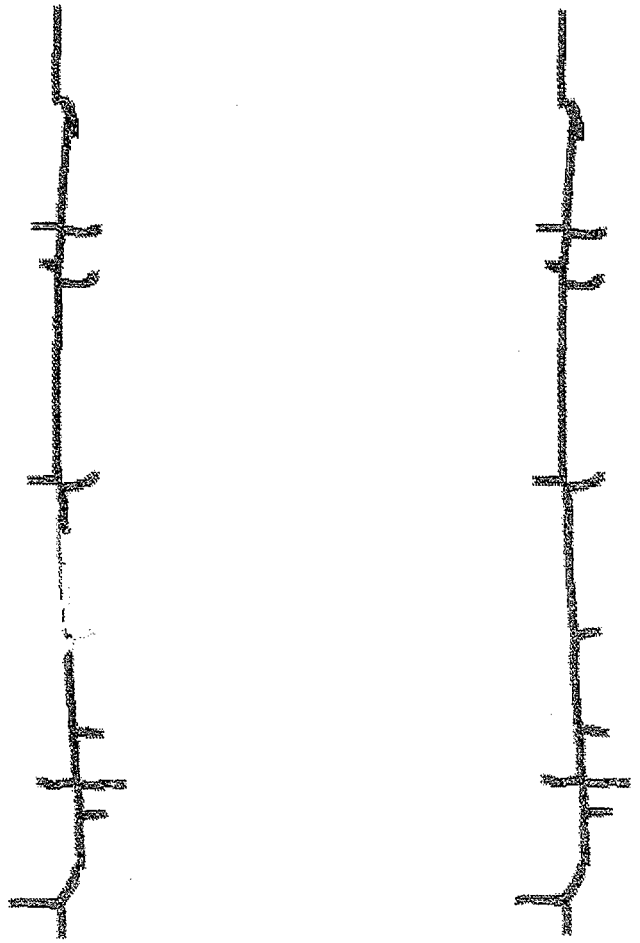


(a)



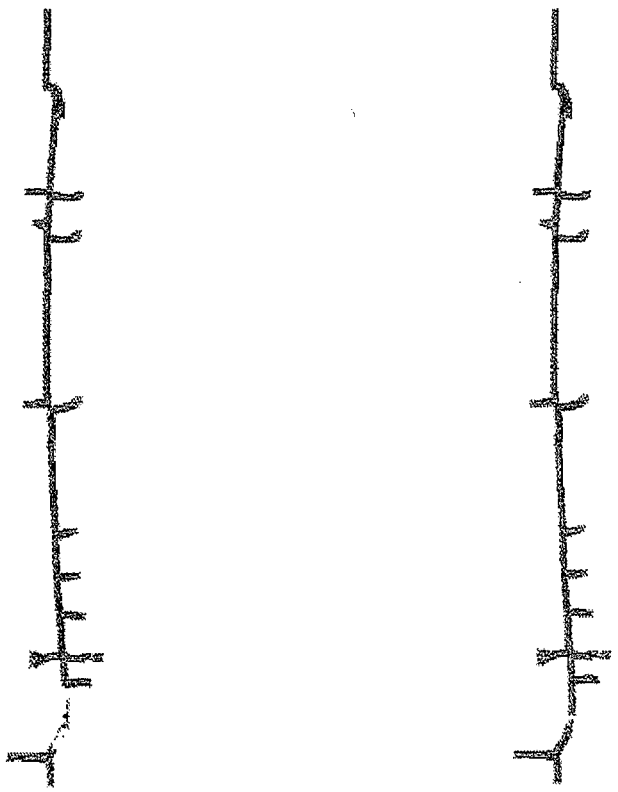
(b) (c)

Fig. 8.8: Generación de una rama y un nodo falsos



(a) (b)

Fig. 8.9: Generación de una nueva rama y nodo que caracterizan una intersección



(a) (b)

Fig. 8.10: Fusión de la información de varios nodos

color amarillo de la figura 8.10(a)). Este nodo está próximo a dos nodos que se habían generado anteriormente cuando el robot había pasado del vestíbulo al pasillo (ver mapa de color verde de la figura 8.10(a)). Estos dos nodos aunque representan el mismo paso del vestíbulo al pasillo no se habían fusionado, debido a que los propios errores de las medias y a los errores cometidos en la obtención de un nuevo nodo estudiando el comportamiento de los puntos de la rama (ver sección 5.2.2.1), habían hecho que estos nodos no se encontraran próximos en el espacio. El robot al pasar por esta zona, genera un nodo que se encuentra próximo a estos dos nodos, y por tanto la información de estos 3 nodos se fusiona generándose un único nodo (ver figura 8.10(b)).

8.3 Experimento 2: Construcción de un mapa topo-geométrico de un entorno cíclico

El segundo experimento consiste en que el robot recorra un entorno cíclico de grandes dimensiones y que genere un mapa topo-geométrico global del mismo, corrigiendo su posición en cada avance. Este entorno cíclico se encuentra situado en la planta 1 del edificio Betancourt de la Universidad Carlos III de Madrid. El plano de la zona recorrida se representa en la figura 8.11. Imágenes reales de la planta 1 que recorre el robot, se muestran en la figura 8.12.

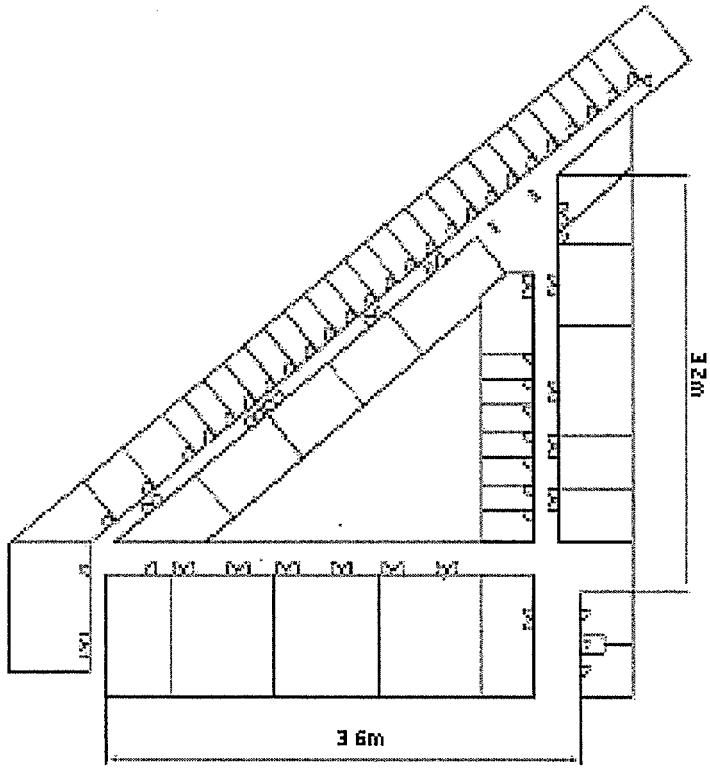


Fig. 8.11: Planta 1 del edificio Betancourt de la Universidad Carlos III de Madrid

En la figura 8.13 se muestran los datos del telémetro láser obtenidos en

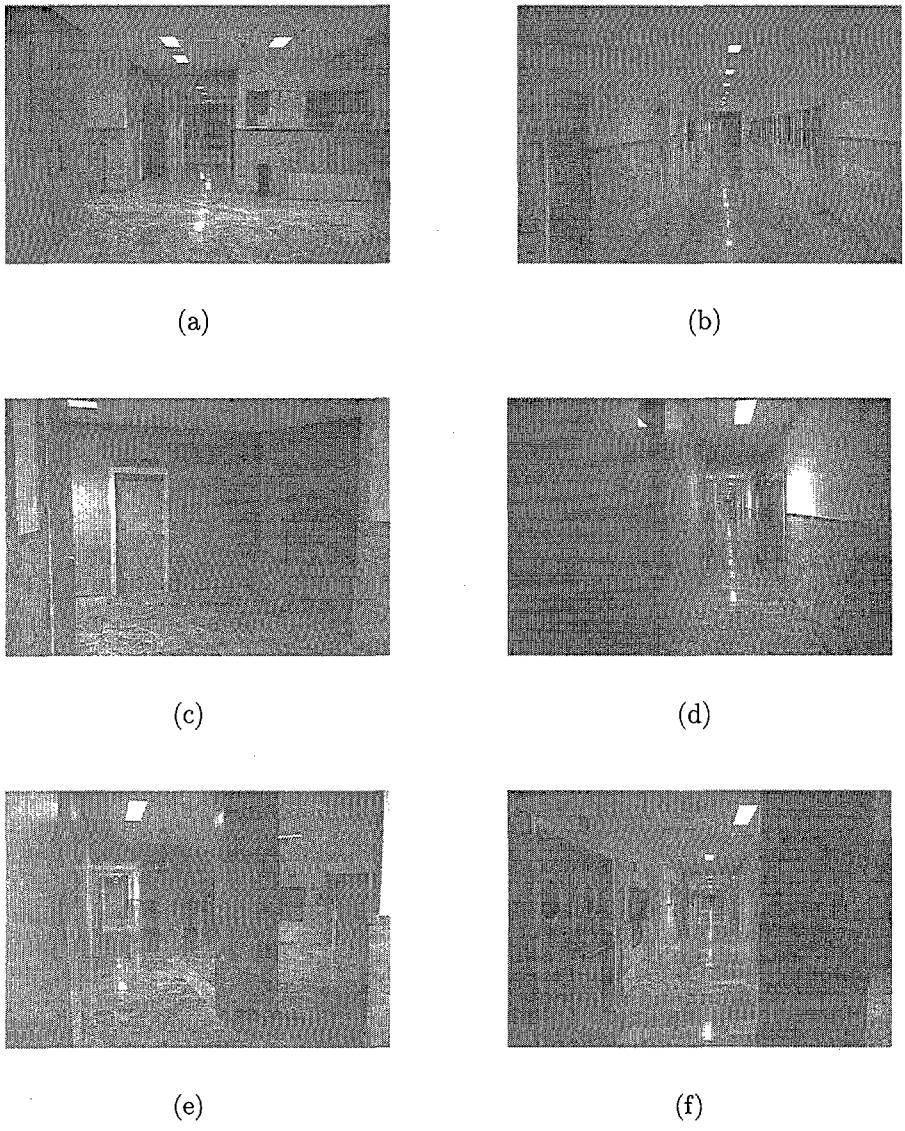
cada avance del robot considerando únicamente la información odométrica. En la figura 8.14 se muestran los mapas topo-geométricos locales generados en cada posición del robot considerando solamente la información odométrica. Al igual que en el caso anterior, se observa que si no se corrige la posición del robot se obtiene un mapa distorsionado del entorno, en este caso el mapa no se cierra. Es necesario, por tanto, aplicar el algoritmo SLAM desarrollado en esta tesis, para corregir la posición del robot y obtener un mapa que represente fielmente la topología del entorno recorrido.

El robot parte del vestíbulo situado en la parte inferior derecha de la figura 8.11 y gira a la izquierda para recorrer el pasillo. Cuando llega al final del mismo gira a la derecha para recorrer el nuevo pasillo, llegando a un nuevo vestíbulo. En este vestíbulo sortea las dos columnas que se encuentran en él, y se aproxima al pasillo que se dirige a la zona de la que partió. El robot adquiere información del entorno a través del telémetro láser cada vez que recorre uno o dos metros aproximadamente.

En este experimento se muestra mejor la necesidad de realizar una estimación posterior de todas las posiciones. En la figura 8.15 se representa la mejor solución que se obtiene al aplicar el algoritmo *hacia adelante* desde una posición desde la que se observa una zona que ya ha sido observada anteriormente. Según la información odométrica el robot se encuentra situado en la posición $(-6.1, 28.28, -2.83)$ con respecto a la posición de la que partió, donde la traslación viene dada en metros y la rotación en radianes. Al aplicar el algoritmo *hacia adelante*, la posición que se obtiene es de $(2, 24.43, -2.893)$. Aunque la posición se ha corregido algo, sin embargo no se consigue cerrar el mapa. En esta posición, el mapa topo-geométrico local que se obtiene se solapa con el mapa topo-geométrico local obtenido en la posición 4. Como el robot observa una zona del espacio de la que ya se tiene información previa, se ejecuta el algoritmo *hacia atrás*.

Al aplicar el algoritmo *hacia atrás* se obtienen dos posibles soluciones (ver figura 8.16). Los valores obtenidos para esta última posición son de $(-1.12, 16.83, -3.047)$ y de $(-0.5, 16.9, -3.11)$. Se observa que hay una corrección de la posición, respecto a la información odométrica, de aproximadamente 5 metros en el eje x, de aproximadamente 12 metros en el eje y, y de aproximadamente 11^o en rotación. Con respecto a la posición obtenida aplicando el algoritmo *hacia adelante*, se obtiene una corrección de aproximadamente 2 metros en el eje x, de aproximadamente 8 metros en el eje y, y de aproximadamente 11^o en rotación.

En la figura 8.16 se observa que no solamente se ha corregido la última posición, sino que se han corregido todas las posiciones. Con esto se consigue



(a) (b)

(c) (d)

(e) (f)

Fig. 8.12: Imágenes de la planta 1 del edificio Betancourt de la Universidad Carlos III de Madrid

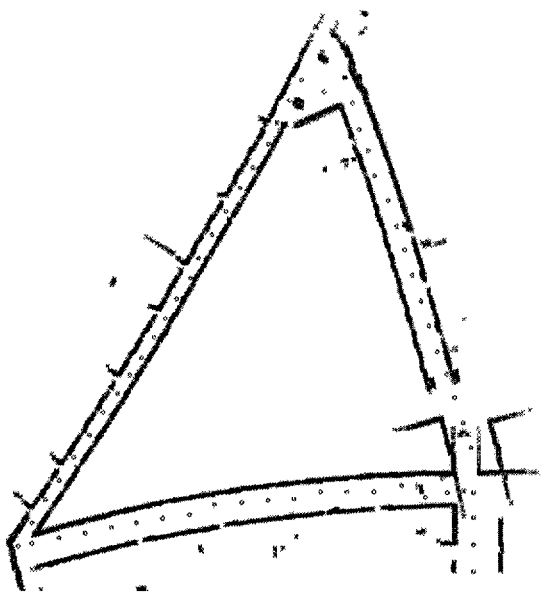


Fig. 8.13: Datos obtenidos del láser en la planta 1 del edificio Betancourt considerando solamente la información odométrica

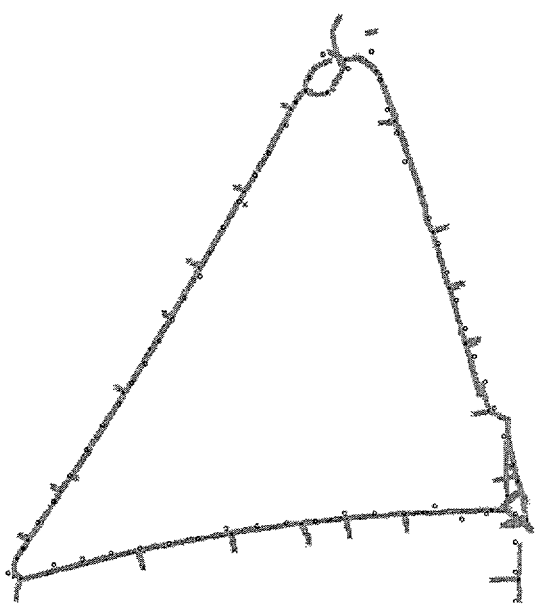


Fig. 8.14: Mapas topo-geométricos locales de la planta 1 del edificio Betancourt considerando solamente la información odométrica

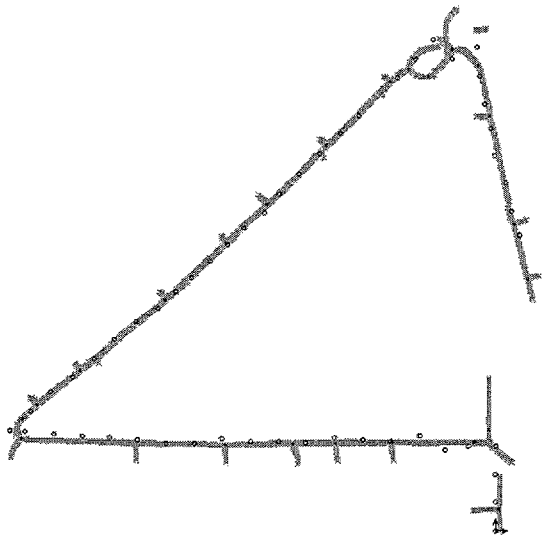


Fig. 8.15: Mejor solución para la posición 59 aplicando el algoritmo *hacia adelante*

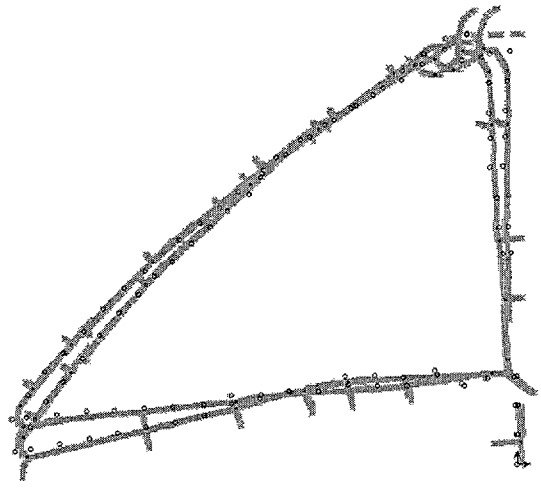


Fig. 8.16: Soluciones obtenidas para la posición 59 aplicando el algoritmo *hacia atrás*

que el mapa se cierre manteniendo una continuidad en el mapa global. Para generar el mapa global se utiliza, de las dos soluciones obtenidas, aquella en la que se consigue un mejor solapamiento entre los mapas topo-geométricos locales obtenidos en cada posición. Este solapamiento se calcula sumando el valor de salud obtenido para cada posición.

Los algoritmos *hacia adelante* y *hacia atrás* se pueden ejecutar iterativamente para mejorar las soluciones obtenidas. Sin embargo, hay que considerar que cuantos más ciclos se realice más lento se hace el proceso. Experimentalmente se ha realizado un único ciclo, es decir, un único proceso *hacia adelante* y un único proceso *hacia atrás*, obteniéndose buenos resultados.

De la figura 8.17 a la figura 8.22 se muestran los resultados al aplicar el algoritmo *hacia adelante* y *hacia atrás* sucesivamente en cada nuevo desplazamiento del robot. En las figuras en las que se representa el resultado obtenido al aplicar el algoritmo *hacia adelante* (figuras 8.17, 8.19 y 8.21) no se consideran las soluciones obtenidas al aplicar el algoritmo *hacia atrás* en el paso anterior, con ello se demuestra que es necesario realizar una corrección, no solamente de la última posición, sino de todas las posiciones anteriores para conseguir que el mapa se cierre. En las figuras en las que se representan las soluciones obtenidas al aplicar el algoritmo *hacia atrás*, después de haber ejecutado el algoritmo *hacia adelante*, (figuras 8.16, 8.18, 8.20 y 8.22) se observa que las posiciones son corregidas cada vez que se ejecuta este proceso.

En la figura 8.22 se muestra la mejor solución obtenida al aplicar el algoritmo de corrección con la que se genera el mapa global. Según la información odométrica la posición última del robot es $(-3.93, 21.57, -2.83)$. Al ejecutar el algoritmo *hacia adelante* (ver figura 8.21) se obtiene una nueva posición corregida cuyos valores son $(3.69, 17.68, -2.852)$. Al aplicar el algoritmo *hacia atrás* los valores obtenidos para esta última posición son $(-0.41, 13.8, -3.1)$. Se obtiene una corrección de la posición, con respecto a la información odométrica, de aproximadamente 3 metros en el eje x, de aproximadamente 8 metros en el eje y, y de aproximadamente 17^o en rotación.

En este experimento, también se observa que hay dos mapas topo-geométricos locales que no coinciden (ver figura 8.23(a)). En este caso no se puede corregir la posición del robot en la que se obtiene el mapa topo-geométrico local que no coincide con ninguno anterior, porque no se dispone de información que se pueda contrastar. En este caso, al ejecutar el algoritmo *hacia adelante*, se considera la información odométrica como buena. Cuando se ejecuta el algoritmo *hacia atrás* se consideran como posibles soluciones aquellas que se encuentran dentro del espacio de búsqueda con una resolución de $15cm$ en traslación y de 2^o en rotación, que es la misma que se ha utilizado en el AG.

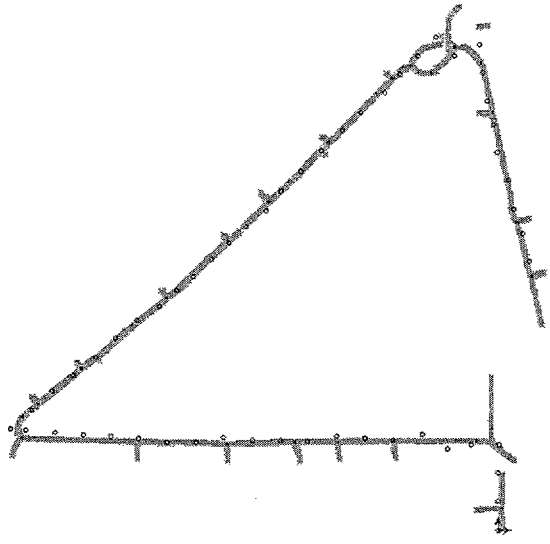


Fig. 8.17: Mejor solución para la posición 60 aplicando el algoritmo *hacia adelante*

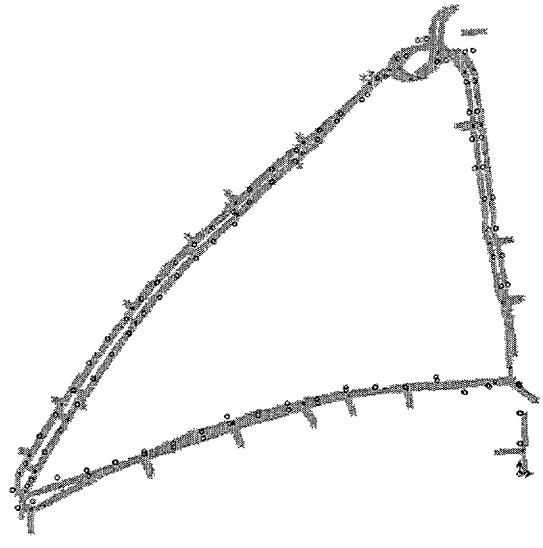


Fig. 8.18: Soluciones obtenidas para la posición 60 aplicando el algoritmo *hacia atrás*

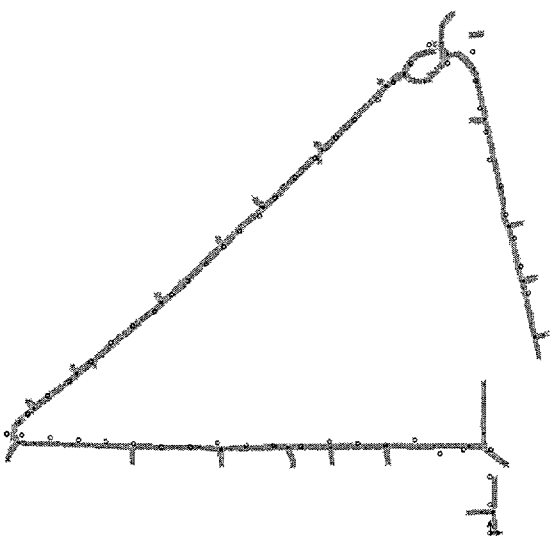


Fig. 8.19: Mejor solución para la posición 61 aplicando el algoritmo *hacia adelante*

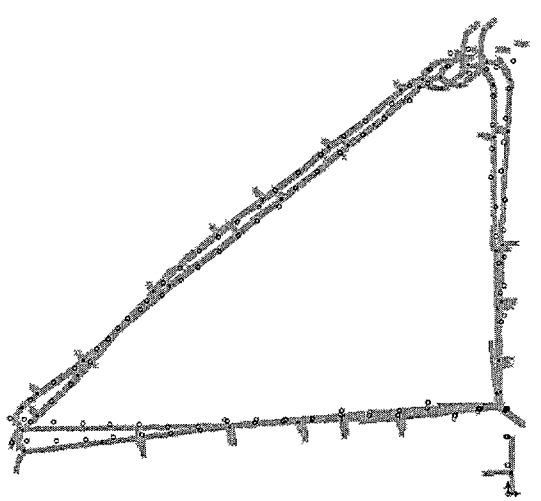


Fig. 8.20: Soluciones obtenidas para la posición 61 aplicando el algoritmo *hacia atrás*

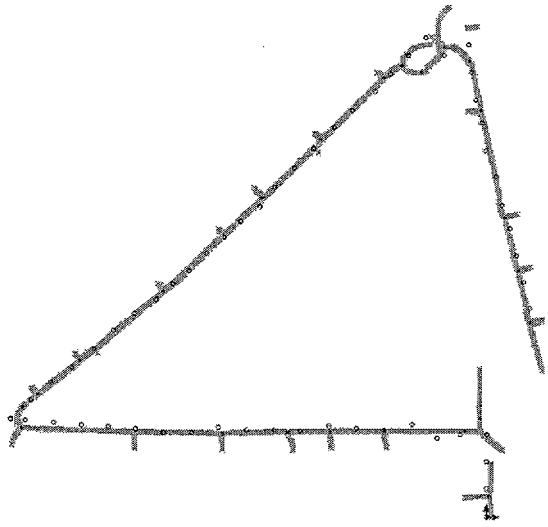


Fig. 8.21: Mejor solución para la posición 62 aplicando el algoritmo *hacia adelante*

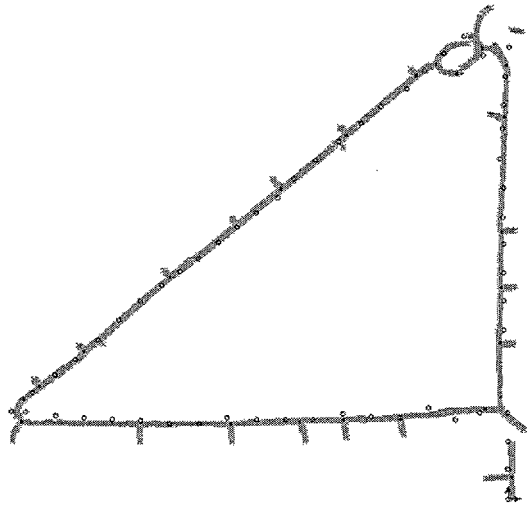


Fig. 8.22: Soluciones obtenidas para la posición 62 aplicando el algoritmo *hacia atrás*

(a) Dos mapas topo-geométricos locales no se solapan

(b) Generación de un DVL virtual

Fig. 8.23: Caso en el que dos mapas topo-geométricos locales no se solapan

Otro problema que se plantea, es que no existe una continuidad en el mapa generado. Esto se soluciona generando un diagrama virtual que une los dos mapas topo-geométricos locales (puntos en color amarillo de la figura 8.23(b)). Los puntos virtuales únicamente se utilizan para la navegación del robot, pero no se tienen en cuenta en el proceso de localización.

9. CONCLUSIONES

En esta tesis se ha desarrollado un nuevo algoritmo que resuelve el problema de construcción de un mapa del entorno desconocido a priori por el que se desplaza un robot móvil autónomo, y localización de manera simultánea (SLAM).

El objetivo de este trabajo era proponer un método de modelización eficaz en un entorno de interiores estructurado de grandes dimensiones como es el edificio donde se han realizado los experimentos, edificio Betancourt de la Universidad Carlos III de Madrid. El modelo propuesto es un modelo topo-geométrico que combina los dos paradigmas de modelado: modelado métrico y modelado topológico. El modelado topológico facilita la navegación y exploración del robot, mientras que el modelado métrico facilita la localización del robot.

Inicialmente el robot se encuentra en un punto del espacio que desconoce. La única información de que dispone es la información proporcionada por un telémetro láser a partir de la que genera un diagrama de Voronoi (DVL). El DVL (Diagrama de Voronoi Local) representa los puntos del espacio que son equidistantes a dos o más objetos. Los objetos del espacio son identificados analizando los datos del sensor. A partir del DVL se genera un mapa topo-geométrico local.

El mapa topo-geométrico contiene nodos y ramas. Los nodos representan lugares típicos de entornos de interiores como son intersecciones, puertas, etc., y las ramas representan las relaciones existentes entre los nodos. Tanto los nodos como las ramas son enriquecidos con información geométrica como son los puntos del espacio que los caracterizan y la distancia de estos puntos a los objetos que son equidistantes.

El robot comienza a desplazarse y en cada desplazamiento adquiere información del entorno a través del telémetro láser construyendo un mapa topo-geométrico local en cada posición. Una primera estimación de lo que se ha desplazado el robot viene dada por la información odométrica. El problema es que esta información posee errores debido al deslizamiento de las ruedas y a los errores propios de todo elemento de medida. Este error se va

acumulando, de tal manera que, en desplazamientos grandes el error cometido en la estimación de la posición del robot es muy grande. Es necesario, por tanto, llevar a cabo un proceso de localización de manera simultanea al proceso de construcción del mapa del entorno para corregir la posición del robot.

En el segundo desplazamiento, la única información de que dispone el robot para corregir su posición es el mapa generado en el instante previo. En desplazamientos pequeños se puede considerar que el entorno no ha sufrido grandes cambios y los errores odométricos no son muy grandes. La corrección de la posición se realiza contrastando la información de los mapas, que se disponen, utilizando algoritmos genéticos (AG). A medida que el robot avanza la información que posee del entorno con la que puede contrastar el mapa actual es mayor.

Si el robot no vuelve a pasar por una zona que ya ha sido observada, solamente puede corregir su posición última. De esta manera, el proceso de localización es solamente incremental y se denomina localización local. Si el robot vuelve a pasar por una zona de la que ya tiene información, puede corregir no solamente la posición última sino que también puede corregir las posiciones anteriores, llevándose a cabo una localización global.

El proceso de localización se puede dividir entonces en dos procesos: un proceso *hacia adelante* en el que se corrige la posición incrementalmente y un proceso *hacia atrás* en el que se corrigen todas las posiciones. Este último proceso sólo se realiza cuando el robot vuelve a pasar por una zona ya observada, y es fundamental para obtener un mapa consistente en entornos de grandes dimensiones con ciclos.

Una vez que se han corregido las posiciones del robot se integra la información de los mapas topo-geométricos obtenidos en cada posición para obtener un mapa global del entorno.

El proceso de SLAM se realiza en dos pasos: un primer paso en el que se corrige la posición considerando que los mapas locales obtenidos no poseen errores y un segundo paso en el que se considera que el valor de las posiciones estimadas en el paso anterior son las correctas y se modifica la información del mapa.

9.1 Aportaciones principales

La primera aportación de esta tesis es un algoritmo nuevo para la construcción de un diagrama de Voronoi generalizado a partir de la información de un telémetro láser, al que se le denomina Diagrama de Voronoi Local (DVL).

El término local es añadido porque este diagrama de Voronoi sólo contiene información de un área restringida del entorno. El DVL está definido por los puntos del espacio que son equidistantes a los obstáculos presentes en el entorno, por tanto, es el camino más seguro que puede seguir el robot. Este algoritmo ha sido desarrollado junto con Dolores Blanco en el Departamento de Ingeniería de Sistemas y Automática.

También se ha realizado un algoritmo que construye un mapa topo-geométrico local a partir del DVL. El tiempo de cálculo de obtención de este mapa a partir de la información del telémetro láser es muy pequeño lo que permite ser utilizado on-line en el propio robot. El robot utilizará el mapa topo-geométrico local para navegar, de tal manera que, será capaz de sortear los nuevos obstáculos presentes en el entorno o personas que se están desplazando por él, ya que, estos cambios serán reflejados en el mapa.

Pero la aportación principal de esta tesis es un algoritmo que permite resolver el problema SLAM (Simultaneous Localization And Mapping) para entornos estructurados de interiores de grandes dimensiones. La mayoría de los métodos que se pueden encontrar en la literatura y que pretenden resolver este problema no dan resultados muy consistentes, ya que, fallan cuando la dimensión del entorno es grande y contiene ciclos. El método presentado en esta tesis funciona bien en entornos de este tipo.

La utilización de Algoritmos Genéticos (AG) en la solución del problema de localización permite reducir los tiempos de cálculo, ya que, sólo se analiza una muestra del espacio de búsqueda. Por otra parte, la diferencia de este algoritmo con otros algoritmos presentes en la literatura es que no está basado en el cálculo de una función de densidad de probabilidad como por ejemplo los basados en el algoritmo EM o los basados en el Filtro de Kalman, sino que proporciona una solución estocástica obtenida a partir del AG. Sin embargo, la utilización de AG permite también, si se quisiera, dar una solución probabilística a este problema.

Otra de las aportaciones de esta tesis es la generación de un mapa que combina las metodologías tanto métrica como topológica, es decir, es un mapa híbrido. En la literatura se pueden encontrar varios algoritmos de construcción de mapas híbridos a partir de un diagrama de Voronoi, pero la novedad de este mapa es que no sólo se almacenan las coordenadas de los puntos que caracterizan tanto los nodos como las ramas, sino que también se almacena otro tipo de información, como es la distancia de estos puntos a los objetos a los que son equidistantes y que se obtiene directamente del DVL.

El mapa construido permite que sea utilizado para resolver distintos pro-

blemas presentes en los robots móviles autónomos. Su configuración topo-geométrica permite planificar tanto local como globalmente las trayectorias del robot. La planificación local se realiza considerando la información del mapa topo-geométrico local calculado en cada instante. Esta planificación local es importante para que el robot pueda desplazarse por un entorno que es dinámico, es decir, sujeto a cambios. La planificación global se realiza considerando el mapa global construido y permite que el robot lleve a cabo tareas que le han sido encomendadas: ir a la habitación A.

El robot puede utilizar este mapa topo-geométrico también para localizarse. Inicialmente el robot solo se podrá localizar localmente pero cuando se genere un mapa completo del entorno ya lo podrá utilizar para localizarse globalmente. El método de localización se basa en el contraste de características extraídas directamente de la información proporcionada por un telémetro láser y que representan lugares típicos de entornos de interiores por lo que no es necesario introducir marcas artificiales o modificar el entorno para que el robot se pueda localizar.

9.2 Trabajos futuros

El algoritmo presentado en esta tesis permite construir de una manera consiste un mapa del entorno por donde se desplaza un robot móvil al mismo tiempo que se localiza.

Una primera mejora de este algoritmo sería reducir los tiempos de cálculo. Esto nos permitiría trabajar con una resolución menor que con la que se ha estado trabajando, mejorando los resultados obtenidos. También permitiría que el algoritmo pudiera ser utilizado on-line en el propio robot.

En el futuro, sería interesante fusionar este algoritmo con métodos de navegación y planificación de trayectorias que permitan de una manera autónoma que el robot explore una zona del entorno y que sea capaz de emplear el mapa generado o de incorporar nueva información dependiendo de si se tiene o no información de la zona.

También sería interesante añadir un módulo que permitiera al robot desplazarse por un entorno dinámico, es decir, en un entorno en el que se pueden introducir nuevos obstáculos o en el que hay personas moviéndose.

APÉNDICE

A. ROBOT B21 DE RWI

A.1 Introducción

El B21 de Real World Interface Inc. (RWI) [Rea96] es un robot móvil diseñado como soporte para aplicaciones robóticas de investigación y desarrollo en medios interiores. Este robot (ver figura A.1), está dividido en tres secciones principales: la base, el cuerpo y la consola. El cuerpo y la consola se hallan físicamente unidos, sin posibilidad de separación, mientras que la base es fácilmente separable del cuerpo.

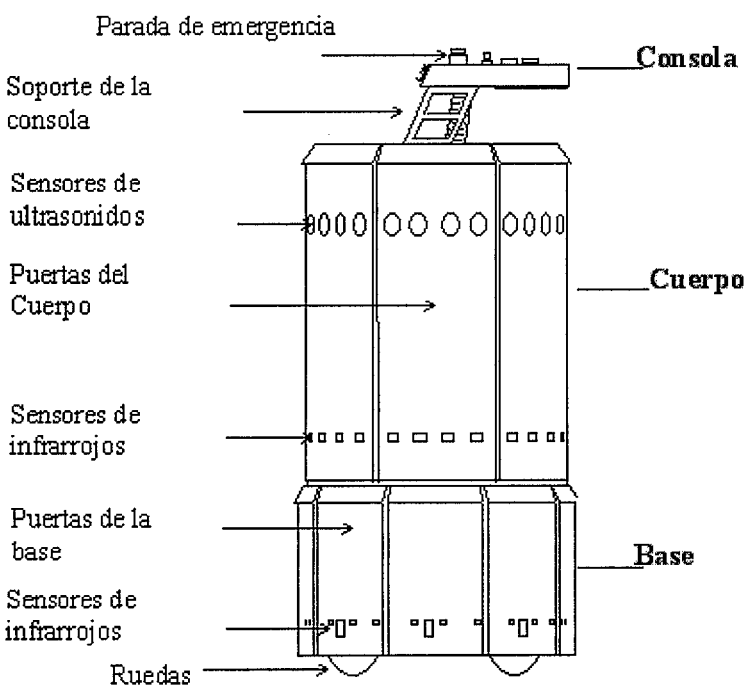


Fig. A.1: El robot B21

El robot se mueve mediante 4 ruedas que son guiadas de forma sincroniza-

da. La base no rota con las ruedas, al ser éstas dirigidas, sino que permanece siempre con la misma orientación. Es el cuerpo el que se orienta según el ángulo de rotación de las ruedas.

La base contiene los motores, las baterías, las ruedas, los circuitos de freno, y otros componentes de potencia y movimiento. En el cuerpo se hallan los sensores, las CPUs, los equipos de comunicación, el interfaz con la consola, ciertos indicadores, interruptores, y otros equipos de medida y procesado. La consola se encuentra en la parte superior del cuerpo y contiene la mayor parte de los indicadores e interruptores del B21. Además, sirve como plataforma física para montar ciertos equipos y sensores como cámaras o, como en este caso, un telémetro láser (PLS Laser Scanner [SIC]).

Toda la superficie exterior tanto del cuerpo como de la base está cubierta de paneles curvos que funcionan como sensores táctiles, y que pueden ser abiertos fácilmente para acceder al interior del robot. Sobre estos paneles van montados todos los sensores de ultrasonidos, táctiles y de infrarrojos (IR).

Los robots B21 pueden tener uno o dos PCs instalados en el interior del cuerpo. Se puede disponer además de un ordenador portátil situado en la parte superior del cuerpo y que se puede utilizar como interfaz con el robot. Las CPUs instaladas por Real World Interface, Inc. (RWI), vienen configuradas con el sistema operativo Linux y el software de RWI Robotic Applications Interface (RAI). El RAI incluye las librerías hardware de bajo nivel, servidores, protocolos de comunicación entre procesos, y utilidades específicas del robot. Cuando se tienen varias CPUs, se conectan entre sí mediante una red Ethernet.

En la tabla A.1 se muestran las especificaciones y las principales características del robot B21 de la casa RWI.

A.2 Arquitectura física

A.2.1 La base

La base es la parte inferior del robot, con una altura de unos 30 cm (12 pulgadas) y un diámetro de unos 53 cm (21 pulgadas). En su interior se encuentran los motores, las correas, las ruedas, los engranajes, las baterías, y la electrónica de control del movimiento. Su parte exterior está totalmente rodeada por una capa protectora, con ocho puertas para acceder al interior, sobre las que se encuentran 32 sensores infrarrojos y 32 sensores táctiles.

Posee dos motores que proporcionan la traslación y la rotación, y cuatro baterías de 12 V y 31 A, que suministran energía a todo el robot.

Una gran parte de las funciones de bajo nivel se realiza enteramente en la base, con lo que se libera en parte a la CPU principal situada en el cuerpo. Para ello, el MCP (Motion Control Processor) se encarga de realizar las tareas de gestión de movimiento y monitorización de condiciones como la corriente del motor, la posición del encoder, y el voltaje de las baterías.

Un conector de expansión de 24 hilos montado en su parte superior proporciona al cuerpo toda la potencia y las señales necesarias.

A.2.2 El cuerpo

El cuerpo se halla montado sobre la base mediante cuatro enganches y es fácilmente separable de ella para simplificar el transporte. En su interior se encuentran las CPUs, los sensores y controladores, muchos de los sistemas de distribución de potencia, los interruptores de potencia, cableado, redes de comunicación, y las zonas reservadas para la incorporación de sistemas opcionales que pueden ser añadidos al B21.

Hay tres tipos de sensores, 24 sensores de ultrasonidos, 24 infrarrojos y 24 táctiles, montados sobre las puertas del cuerpo y controlados por tarjetas MSP (Multi Sensor Processor).

En el interior del robot, en su lado izquierdo, se encuentra la CPU principal. En este caso, la CPU está equipada con el sistema operativo Linux 6.0 y se comunica con los sensores del cuerpo mediante un bus de alta velocidad y con la base vía protocolo serie RS-232 a 9600 baudios.

Además de esta CPU principal se puede instalar otra, que en este caso se ha equipado con el sistema operativo MS-DOS 6.0 y se encarga del control de la conexión con una cámara CCD y de la adquisición y procesamiento de imágenes.

También se puede colocar, en la parte superior del cuerpo, bajo la consola, un ordenador portátil. Este irá conectado a la CPU principal mediante Ethernet y servirá como interfaz con el robot, de modo que el usuario no tenga la necesidad de conectar un monitor y un teclado a dicha CPU principal.

A.2.3 La consola

La consola está integrada en la parte superior del cuerpo y en ella se localizan la mayor parte de los controles de usuario del B21 (ver figura A.2), entre ellos el interruptor de habilitado y la parada de emergencia, los LEDs

de la CPU, los LEDs de potencia, los botones de función, las conexiones Ethernet, los puntos para el montado de la antena, etc. Además, pueden montarse sobre ella algunos sistemas opcionales como cámaras, unidades de pan-tilt, o como en este caso un telémetro láser en la parte frontal.

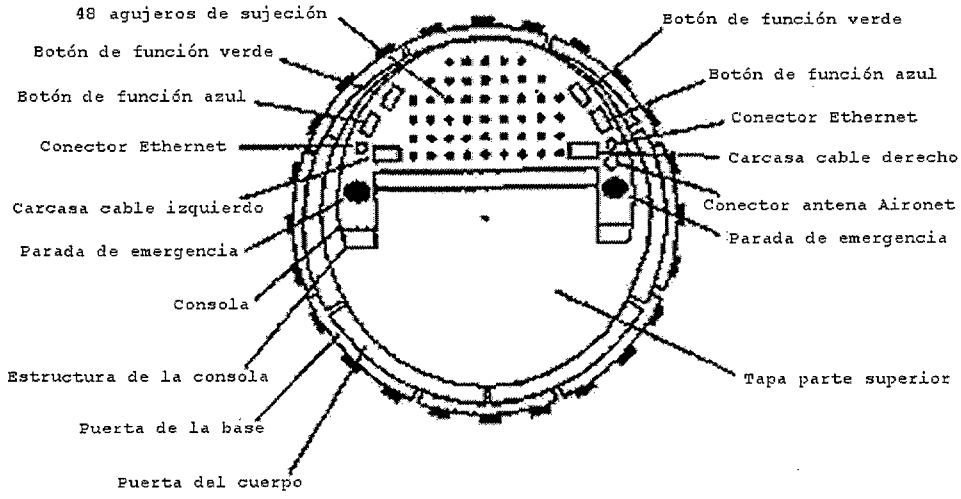


Fig. A.2: Componentes de la consola del robot B21

A.3 Arquitectura electrónica

El sistema electrónico del B21 incluye los dispositivos de potencia, comunicaciones y cableado de señales.

A.3.1 Distribución de potencia

El B21 está alimentado a 48 V mediante cuatro baterías localizadas en la base. Los 48 V se llevan desde la base a dos circuitos que actúan como interruptores, uno de ellos alimenta a los motores y al convertidor DC-DC, el otro envía esa tensión hacia el cuerpo por 8 de los 24 circuitos del contacto rotatorio que comunica el cuerpo con la base, llevándola hasta la tarjeta principal de distribución de potencia del cuerpo. Después, los 48 V se convertirán a +5 V, +/-12 V, y +24 V, mediante convertidores DC-DC.

A.3.2 Buses de comunicación

El B21 utiliza varios tipos de buses de comunicación:

- **RS-232 asíncrono serie.** Cada CPU viene con dos puertos RS-232. La CPU principal utiliza el primero de sus puertos para comunicarse con el MCP (Motion Control Processor) de la base, el otro queda libre para otras aplicaciones opcionales.
- **Red Ethernet.** La CPU del robot está equipada con un adaptador de Ethernet que le permite comunicarse con otros computadores a bordo o fuera del robot. Esta red se halla configurada para cableado 10base2, también llamada 'thinnet', y utiliza cable coaxial y estructura en malla.
- **Bus de acceso rápido multi-maestro asíncrono serie.** Este bus de acceso se utiliza en el B21 para enlazar siete MSPs (Multi Sensor Processor) distribuidos entre sí y con la CPU principal. Como es un bus multi-maestro, cualquier dispositivo conectado a él puede enviar un mensaje en cualquier momento, arbitrariamente.
- **Puerto Paralelo.** Por cada CPU instalada en el robot se dispone de un puerto paralelo. El de la CPU principal se usa por los controles del B21 de varias formas:
  - Lee los cuatro botones de función que aparecen en lo alto de la consola.
  - Controla las luces de los cuatro botones de función.
  - Es capaz de controlar los seis conectores de potencia auxiliares.
  - Controla las luces de los interruptores de parada de emergencia.
- **Bus de los sensores de ultrasonidos.** Es un cable de 10 hilos que conecta todas las tarjetas de los sensores de ultrasonidos de las puertas del cuerpo del robot con los MSPs (Multi Sensor Processor) que las controlan. Cuatro de los hilos se usan para el direccionamiento (los MSP solo usan 3 de estas líneas), otros cuatro llevan los +5 V y la tierra, y los otros dos restantes disparan y leen los sensores.
- **Bus de los sensores de infrarrojos.** Es un cable de 10 hilos que conecta todas las tarjetas de los sensores IR del cuerpo y la base a los MSPs que las controlan. Tres hilos se usan para el direccionamiento, cuatro para llevar los +5 V y la tierra, y los otros tres para disparar y leer los sensores IR.

En la figura A.3 se muestra una visión general del sistema, en la que se representan las distintas conexiones entre las CPUs principales y los distintos tipos de sensores, dispositivos periféricos, etc.

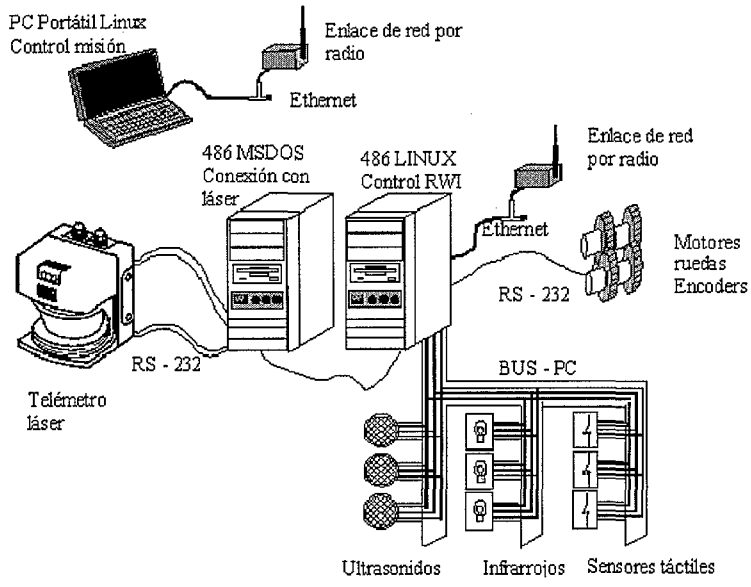


Fig. A.3: Arquitectura electrónica del robot B21

A.4 Accesorios

A.4.1 CPUs

La configuración estándar del B21 está equipada con una CPU Pentium en el lado izquierdo, pudiendo instalar otra en el derecho. Las CPUs vienen con dos puertos serie RS-232 de 9 pines y un puerto paralelo. El equipamiento estándar también incluye las tarjetas del bus de acceso, de Ethernet y del adaptador de vídeo SVGA. También están provistas de un botón de *reset* mediante hardware.

A.4.2 Cargador

El cargador proporciona la potencia externa necesaria para el B21, recargando sus baterías y proporcionando energía a los sistemas de a bordo

mientras éstas se recargan. El cargador está diseñado para que el B21 pueda permanecer encendido 24 horas al día. El tiempo de recarga varía dependiendo de lo descargadas que se encuentren las baterías, pero unas 8 horas suelen ser suficientes para asegurar una buena carga independientemente del grado de descarga de las baterías. El cargador posee un interruptor de potencia que le permite aceptar voltajes de entrada entre 100 y 240 V.

A.4.3 Baterías

Las cuatro baterías de la base del B21 son de ácido de plomo. Proporcionan un total de 1488 vatios hora de potencia para la base y todos los equipos a bordo. Se recargan mediante una fuente de continua regulada a unos 59 V. La corriente de carga inicial es de unos 6 A, mientras que la corriente por las baterías a plena carga es de 0,25 A.

A.4.4 Joystick

El joystick permite el movimiento controlado de la base del B21 para operaciones de traslado o de demostración. La activación del joystick impide que los comandos de movimiento provenientes de la CPU sean enviados a los motores, para devolver el control al puerto serie hay que resetear la base. Para mayor seguridad, el joystick está provisto de un sistema de 'hombre muerto'.

A.4.5 Modem de radio

El B21 puede estar equipado con un par de modems de radio Comrad RS-232 opcionales. Uno de los modems se monta en la consola, el otro es externo al robot y viene con un adaptador de potencia de 120 V AC a 12 V DC. La frecuencia de la radio está en el rango de los 900 Hz. Para establecer el enlace, ambos modems deberán funcionar en el mismo canal, A o B, y estar en modo SEND. Si el enlace es establecido correctamente se encenderán los LEDs de enlace ("link") y de señal ("signal") de los modems.

A.5 Subsistemas

A.5.1 Convertidores DC-DC

Dentro del robot se encuentran diversos convertidores, tanto en la base como en el cuerpo.

Convertidores en la base:

- Un convertidor de 50 Vatios, de +5 V, necesario para el MCP, los MSPs, los sensores IR, y otros circuitos.

Convertidores en el cuerpo:

- Un convertidor de 200 Vatios, de +5 V, necesario para las CPUs, los MSPs, los sensores de ultrasonidos, los conectores de potencia auxiliares, la unidad de pan-tilt, y otros circuitos.
- Un convertidor de 200 Vatios, de +12 V, necesario para las CPUs, las cámaras y los conectores de potencia auxiliares.
- Un convertidor de 200 Vatios, de +24 V, necesario para la unidad de pan-tilt, el compás, la radio Ethernet y los conectores de potencia auxiliares.
- Un convertidor de 3 Vatios, de -12 V, necesario para las CPUs.
- Un convertidor opcional de 200 Vatios, de +5 V.

Después de la conversión, las tensiones bajas son distribuidas por interruptores de estado sólido manuales hacia los dispositivos que las necesitan. Seis interruptores de potencia de estado sólido auxiliares pueden controlarse manualmente o desde el puerto paralelo de alguna de las CPUs.

A.5.2 El MCP de la base

A.5.2.1 Conexiones serie del MCP de la base

Los cables de comunicación serie se pueden conectar a la base del B21 de dos formas:

- El conector de 9 pines del puerto serie sub miniatura-D del panel de control de la base se utiliza para controlar ésta desde un computador o terminal externo al robot.

- Cuando se instala el cuerpo del robot las líneas serie RS-232 de la base son guiadas por separado a través de un conector hembra de 9 pines hacia la CPU principal.

A.5.2.2 Protocolo de comunicación del MCP

El MCP de la base bajo control serie funciona como un terminal full-duplex. Todos los caracteres son duplicados (salvo en el modo half-duplex o modo binario). La base se comunica con 8 bits de datos y un bit de parada. No se utiliza paridad, cabecera ni control de flujo.

El MCP no soporta el uso de las señales de "data carrier detect", "data terminal ready", "data set ready" y "clear to send".

A.5.3 Sensores

La información leída por los distintos tipos de sensores del robot se envía a unos módulos software controlados por un programa denominado RAI (Robotic Applications Interface). Este software es proporcionado con las interfaces para los sensores de ultrasonidos, los infrarrojos y los táctiles, pero se pueden escribir fácilmente otros módulos de control para sensores adicionales, por ejemplo el telémetro láser.

Los módulos del RAI recogen los datos procedentes de los sensores, los transforman a un formato adecuado para trabajar con ellos y los introducen en un vector, avisando al usuario cuando llegan nuevos datos mediante el uso de mensajes.

Se dispone de un vector de estructuras para los sensores, que indica la correspondencia entre los sensores físicos y sus correspondientes valores, y de ciertas constantes ya definidas que proporcionan un método de iteración a través de un vector o por cálculo de un índice para un sensor dado de forma que pueda ser utilizado por cualquiera de los robots.

A.5.3.1 Ultrasonidos

Como se acaba de indicar, existe un módulo RAI que se encarga de recoger las lecturas de los 24 sensores de ultrasonidos que se encuentran alrededor del cuerpo del robot, de transformar estos datos a milímetros y dejarlos a disposición de la CPU.

Algunas características teóricas de los sensores de ultrasonidos son:

- **Radio máximo**: 8 metros.

- **Radio mínimo**: 10 centímetros.
- **Ángulo de apertura del sensor**: 15°.

A.5.3.2 Infrarrojos

Los sensores de infrarrojos envían una señal, una radiación infrarroja, y de la señal reflejada se puede extraer información sobre el entorno en el que se encuentra el robot. Los IR no son estrictamente un tipo de sensor, ya que la señal de retorno depende del color y la textura del material en el que se refleja. Los IR se utilizan generalmente para cubrir áreas en las que los ultrasonidos no obtienen medidas correctas, como lugares a baja altura. Los valores que proporcionan no son muy precisos.

Algunas características teóricas de estos sensores son:

- **Radio máximo**: 50 centímetros.
- **Radio mínimo**: 4 centímetros.
- **Ángulo de apertura del sensor**: 15°.

A.5.3.3 Táctiles

Los sensores táctiles indican si el robot ha colisionado con algún obstáculo, enviando la orden de parar para evitar daños tanto en el robot como en el objeto de la colisión.

A.5.3.4 Telémetro láser

El telémetro láser, A.4, realiza un barrido del entorno proporcionando la magnitud de las distancias a los objetos que se encuentran en el plano de medida. Está colocado en la parte frontal de la consola, por lo que el plano de medida está situado a 110 cm del suelo, aproximadamente. La conexión con el PC del robot se realiza vía protocolo RS-232 a 9600 baudios. Toda la información referente al telémetro láser se encuentra en el apéndice B

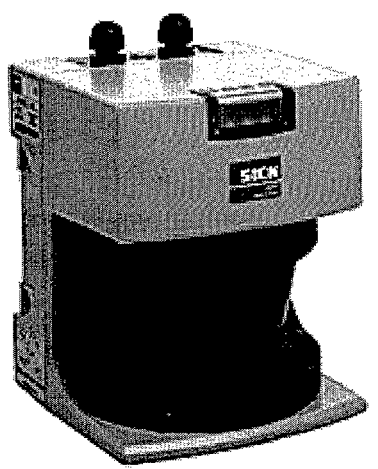


Fig. A.4: Telémetro láser PLS de la casa Sick

B21	**Especificaciones**
Sónar	24(cuerpo)
IR	56 (32 base, 24 cuerpo)
Táctiles	56 (32 base, 24 cuerpo)
CPU´s	Pentium II simple o dual, o Computer System
Comunicaciones	RS-232 opcional sin cable o Ethernet
Trabajo en red	A bordo: 10baseT y 10base2
Baterías	41440 W-hr
Tiempo de autonomía	6 horas
Motor	4 high torque, 48 VDC servo motores
Conducción	4 ruedas síncronas
Puertos de E/S	Joystick, RS-232
Radio de giro	Cero
Velocidad de traslación	90 cm/s
Velocidad de rotación	167°/s
Resolución de traslación	1 mm
Resolución de rotación	0.35°
Carga **Torreta direccionable** **Paneles protectores** **Diámetro**	90 kg. No necesita potencia 14 (8 alrededor de la base, 6 alrededor del cuerpo) 52.5 cm
Altura	106 cm
Peso	122.5 kg.
Uniformidad del suelo	2.54 cm
Accesorios	Ampliación de computadores Pack de baterías de repuesto Sistema de visión Triclops 3D Sistema de visión estéreo de alta definición Componentes de visión individuales Radio Ethernet sin cable, de 2.4.GHz Modem RS-232 sin cable, de 900 Mhz SICK® PLS Scanner Interfaz de ASCII a palabra Compás de navegación computerizado

Tab. A.1: Especificaciones del robot B21

B. TELÉMETRO LÁSER PLS-220 DE SICK

B.1 Introducción

El telémetro láser PLS [SIC] es un dispositivo capaz de medir las distancias a los objetos que se encuentran en su entorno, a partir del tiempo de tránsito de la luz. Para ello, el láser efectúa una operación de barrido o escaneado (de forma similar a un radar) entre 0 y 180°, lo que proporciona información sobre la posición de los objetos presentes en el plano de medida.

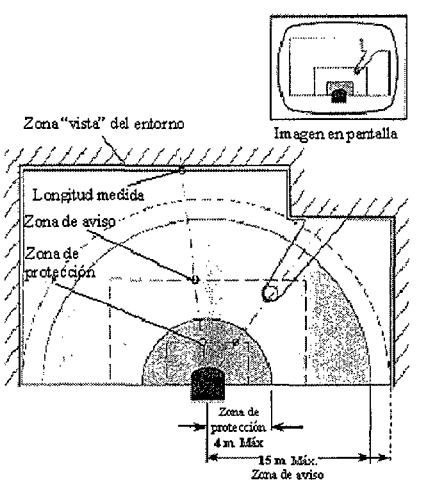


Fig. B.1: Rangos de medida del telémetro láser PLS-220

Permite definir unos rangos de medida de manera que cuando un objeto entra dentro de esta zona predeterminada, este dispositivo dispara una señal de alarma. Estos rangos de medida se denominan zona de aviso y zona de protección. Este tipo de sensor es utilizado tanto en sistemas de seguridad como en AGVs (Automated Guided Vehicle) y puede proporcionar, mediante

el ajuste de la zona de aviso, una rápida señal de advertencia antes de que el objeto entre en la zona de protección (ver figura B.1).

Las zonas de aviso y protección pueden ser definidas con la ayuda del software apropiado, que transmite esta información al telémetro. La zona de protección queda almacenada en la memoria residente del escáner, por lo que la conexión con el ordenador sólo es necesaria cuando se requiere cambiar estos rangos de medida (ver figura B.2).

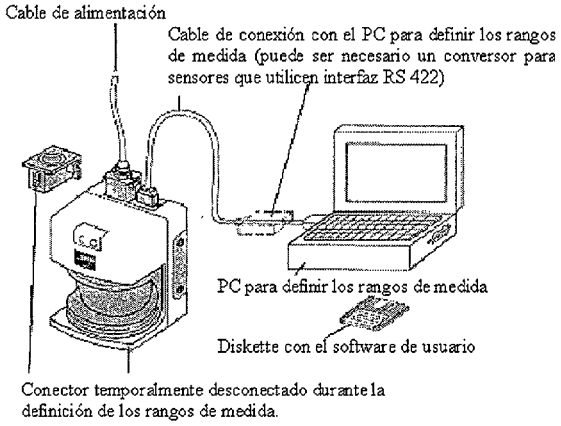


Fig. B.2: Configuración del telémetro láser PLS-220

B.2 Condiciones de aplicación

El telémetro PLS está diseñado para su utilización en espacios cerrados y es adecuado para su uso en plantas de fabricación o en vehículos de transporte terrestre. Otras posibles aplicaciones son:

- Reconocimiento posicional de objetos
- Medida del tamaño de objetos
- Reconocimiento de formas

B.3 Instalación

La instalación del PLS se realiza dependiendo de la zona de protección que se defina (ver figura B.3). Es importante que se asegure que la zona de

protección elegida pueda ser registrada completamente por el láser (en un semicírculo completo) y con un radio máximo de $4m$ (ver figura B.1).

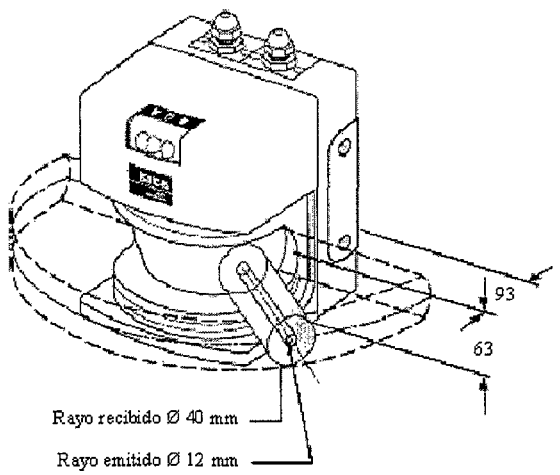


Fig. B.3: Instalación del PLS-220 de acuerdo a la zona de protección

A la hora de elegir la posición óptima para el sensor, habrá que tener en cuenta los siguientes factores:

- Si es posible, se evitará la presencia de objetos estáticos en la zona de medición del láser.
- La posición deberá facilitar las conexiones eléctricas del sensor.
- La posición elegida deberá proporcionar al sensor la mayor protección posible.

La sujeción del sensor se realiza mediante tornillos y, con los accesorios adecuados puede orientarse girando el sensor según las direcciones que marca la figura B.4.

B.4 Conexiones eléctricas

La alimentación eléctrica es suministrada a través de un conector especial que se acopla al sensor mediante un conector Sub-D de 9 pines. La tensión de alimentación necesaria es de $14V \pm 15\%$ con una potencia de $14W$, más la carga de las tres posibles salidas (máximo $2x250mA$ y $1x100mA$). Estas

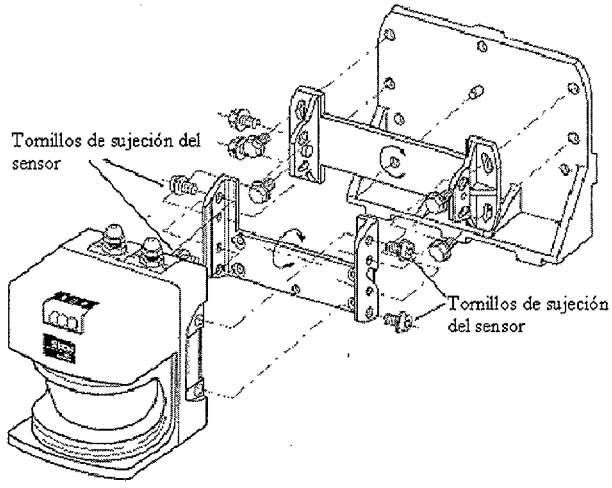


Fig. B.4: Montaje del telémetro láser PLS-220

salidas son OSSD1 y OSSD2, que indican el estado de la zona de protección (ocupada o no) y WEAK-SIGNAL que indica el estado de la zona de aviso. La conexión con el interfaz (que utiliza un protocolo *RS*232 o *RS*422) se hace a través de otro de estos conectores especiales, como se puede ver en la figura B.5.

Los dos conectores son siempre suministrados junto con el telémetro, y mientras el dispositivo se encuentre en funcionamiento, deberán estar insertados y atornillados.

B.5 Luces indicadoras

El sensor tiene tres indicadores luminosos (ver figura B.6). Sólo la luz roja y la verde tienen relación con la zona de protección (tabla B.1). La luz amarilla indica que existe un objeto en la zona de aviso o que existe una acumulación de polvo en la pantalla frontal del láser (tabla B.2). La pantalla frontal es un componente óptico que debe ser limpiado utilizando un paño suave.

B.6 Características técnicas

En la tabla B.3 se detallan las especificaciones de telémetro.

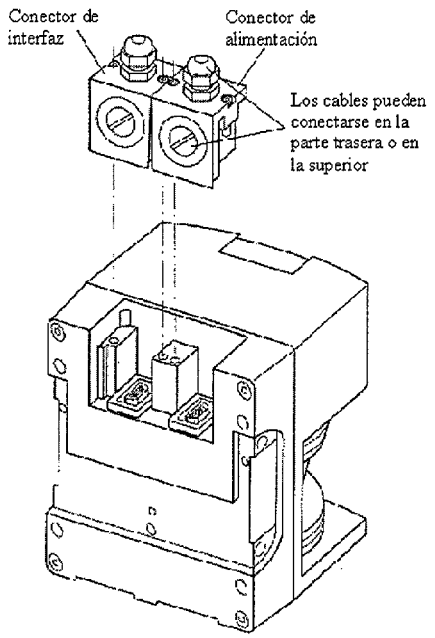


Fig. B.5: Conectores del telémetro láser PLS-220

Zona de protección	**Indicadores luminosos**		**Salida** OSSD1 y OSSD2
	Rojo	Verde	
Objeto presente	Encendido	Apagado	Nivel bajo ($0V$)
Vacía	Apagado	Encendido	Nivel alto ($24V$)

Tab. B.1: Función de los indicadores y estado de la salida de la zona de protección

Estado	**Indicador** Amarillo	**Salida** WEAK SIGNAL
Zona de aviso vacía Sin acumulación de polvo	Apagado	Nivel alto ($24V$)
Aviso de acumulación de polvo	Parpadeo ($1/s$)	Nivel bajo ($0V$)
Objeto en zona de aviso Acumulación de polvo	Encendido	Nivel bajo ($0V$)

Tab. B.2: Función de indicador y del estado de la salida para la zona de aviso y acumulación de polvo

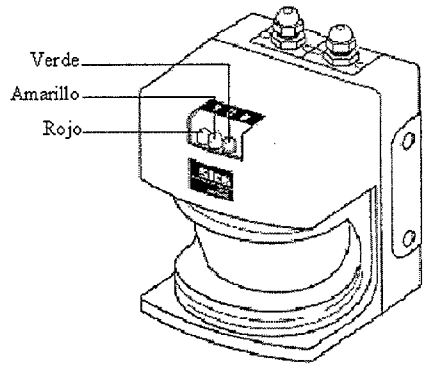


Fig. B.6: Indicadores luminosos del telémetro láser PLS-220

Datos generales	
Tensión de alimentación	$24VCC \pm 15\%$
Potencia	$< 17W$ más la carga de las salidas (máx. $24V/2x250mA + 1 < 100mA$)
Dimensiones (anchoxaltoxlargo)	$155x185x156mm^3$
Máximo ángulo de barrido	180°
Resolución angular	0.5°
Rango de medida	$50m$
Resolución	$\pm 50mm$
Periodo de barrido	40 ms
Emisor	Diodo láser infrarrojo
Zona de protección	
Rango	$4m$
Tiempo de respuesta	$< 80ms$
Reflectancia mínima	1.8%, difusa
Reflectancia máxima	Sin límite
Resolución	$< 70mm$
Salida	2 salidas semiconductoras PNP, $24V/250mA$
Zona de aviso	
Rango	$15m$ aprox.
Salida	Salida semiconductor PNP, $24V/100mA$

Tab. B.3: características técnicas del PLS

BIBLIOGRAFíA

[AmaEstMán95] Josep Amat, Francesc Esteva, and Ramón López de Mántaras. Autonomous navigation troup for cooperative modelling of unknown environments. In *Seventh International Conference on Advanced Robotics ICAR'95*, pages 383–389, Sant Feliu de Guixols, Catalonia, Spain, September 1995.

[AntPet01] G. I. Antonaros and L. P. Petrou. Real time map building by means of n ellipse spatial criterion and sensor-based localization for mobile robot. *Journal of Intelligent and Robotic System*, 30:331–358, 2001.

[AraAlm98] Rui Araújo and Aníbal T. de Almeida. Map building using fuzzy ART, and learning to navigate a mobile robot on an unknown world. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation.*, 1998.

[Arm97] José María Armingol. *Localización de Robots Móviles Autónomos.* PhD thesis, Universidad Carlos III de Madrid, 1997.

[ArmMorEsc98] J.M. Armingol, L. Moreno, A. de la Escalera, and M.A. Salichs. Landmark perception planning for mobile robot localization. In *IEEE International Conference on Robotics and Automation*, 1998.

[ArmMorEsc99] J. M. Armingol, L. E. Moreno, A. de la Escalera, and M. A. Salichs. A genetic algorithm for mobile robot localization using ultrasonic sensors. In *14th World Congress of IFAC International Federation of Automatic Control*, 1999.

[BarEgiSal01] R. Barber, V. Egido, and M.A. Salichs. Algorithm of topological map generation for the EDN navigation system. In *IFAC Workshop on Mobile Robot Technology*, 2001.

[BlaBoaMor00] D. Blanco, B.L. Boada, L. Moreno, and M.A. Salichs. Local mapping from on-line laser voronoi extraction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2000.

[BlaBoaMor01] D. Blanco, B.L. Boada, and L. Moreno. Localization by voronoi diagrams correlation. In *International Conference on Robotics and Automation*, 2001.

[BoaPalBla02] B.L. Boada, D. Palazon, D. Blanco, and L. Moreno. Voronoi based place recognition using hidden markov models. In *International Federation of Automatic Control 15th IFAC World Congress*, 2002.

[BooOveSta99] Valérie Boor, Mark H. Overmars, and A.Frank Van der Stappen. The gaussian sampling strategy for probabilistic roadmap planners. In *Proceedings of the 1999 IEEE Conference on Robotics and Automation*, pages 1018–1023, 1999.

[BorBru95] G. Borghi and D. Brugali. Autonomous map learning for a multi-sensor mobile robot using diktiometric representation and negociation mechanism. In *Seventh International Conference on Advenced Robotics ICAR'95*, pages 521–528, Sant Feliu de Guixols, Catalonia, Spain, September 1995.

[BreChaDev94] Stéphane Betgé Brezetz, Raja Chatila, and Michel Devy. Natural scene understanding for mobile robot navigation. In *IEEE International Conference on Robotics and Automation*, pages 730–736, San Diego, California, 1994.

[BreChaDev95] Stéphane Betgé Brezetz, Raja Chatila, and Michel Devy. Object-based modelling and localization in natural environments. In *IEEE International Conference on Robotics and Automation*, pages 2920–2927, 1995.

[Bro82] R.A. Brooks. Symbolic error analysis and robot planning. *Int. Journal Robotics Research*, 1(4):29–68, 1982.

[BulDev96] Hanna Bulata and Michel Devy. Incremental construction of a landmark-based and topological model of indoor environments by a mobile robots. In *Proceedings of the 1996*

IEEE International Conference on Robotics and Automation, pages 1054–1060, Minneapolis, Minnesota, April 1996.

[BurDerFox98] Wolfram Burgard, Andreas Derr, Dieter Fox, and Armin B. Cremers. Integrating global position estimation and position tracking for mobile robots: The dynamic markov localization approach. In *Proceedings of the 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems.*, pages 730–735, 1998.

[BurFoxHen96a] Wolfrag Burgard, Dieter Fox, and Daniel Henning. Fast grid-based position tracking for mobile robots. In *In Proc. Of the First Euromicro Workshop on Advanced Mobile Robots (EUROBOT'96). IEEE Computer Sicity Press.*, pages 2–9, 1996.

[BurFoxHen96b] Wolfram Burgard, Dieter Fox, Daniel Henning, and Timo Schmidt. Estimating the absolute position of a mobile robot using position probability grids. In *Proc. of the Fourteenth National Conference on Artificial Intelligence (AII'96)*, pages 896–901, 1996.

[BurFoxJan99] Wolfram Burgard, Dieter Fox, Hauke Jans, Christian Matenar, and Sebastian Thrun. Sonar-based mapping with mobile robots using EM. In *In Proc. Of the International Conference on Machine Learning (ICML)*, 1999.

[CasKaeKur96] Anthony R. Cassandra, Leslie Pack Kaelbling, and James A. Kurien. Acting under uncertainty: Discrete bayesian models for mobile-robot navigation. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems IROS'96*, 1996.

[CasMarNei98] J.A. Castellanos, J.M. Martínez, J.Ñeira, and J.D. Tardós. Simultaneous map building and localization for mobile robots: A multisensor fusión approach. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, pages 1244–1249, Leuven, Belgium, May 1998.

[CasMorSal01] C. Castejon, L. Moreno, and M.A. Salichs. Traversability models in 3d environments. In *3rd International Conference on Field and Service Robotics FSR2001*, 2001.

[CasTarSch97] J.A. Castellanos, J.D. Tardós, and G. Schmidt. Building a global map of the environment of a mobile robot: The importance of correlations. In *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, pages 1053–1059, Albuquerque, New Mexico, April 1997.

[Cha95] Raja Chatila. Deliberation and reactivity in autonomous mobile robots. *Robotics and Autonomous Systems*, 16:197–211, 1995.

[ChaLau85] R. Chatila and J.P. Laumond. Position referencing and consistent world modelling for mobile robots. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 138–145, 1985.

[Cho96] Howie Choset. *Sensor Based Motion Planning: The Hierarchical Generalized Voronoi Graph.* PhD thesis, California Institute of Technology, Pasadena, California, March 1996.

[Cho01] H. Choset. Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization. *IEEE Transactions on Robotics and Automation*, 17(2):125–137, 2001.

[ChoBur95] Howie Choset and Joel Burdick. Sensor based planning, part ii: Incremental construction od the generalized voronoi graph. In *IEEE Conference on Robotics and Automation*, 1995.

[ChoNag01] Howie Choset and Keiji Nagatani. Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization. *IEEE Transactions on Robotics and Automation*, 17(2):125–137, April 2001.

[Cso97] Michael Csorba. *Simultaneous Localisation and Map Building.* PhD thesis, Robotics Research Group. Department of Engineering Science. University of Oxford, 1997.

[DelPégMou98] Laurent Delahoche, Claude Pégard, El Mustapha Mouaddib, and Pascal Vasseur. Incremental map building for mobile robot navigation in an indoor environment. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation.*, 1998.

[DemLaiRub77] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Roy. Stat. Soc.*, 39(1):1–38, 1977.

[DevBul95] M. Devy and H. Bulata. Landmark-based us feature-based localization of a mobile robot in a structure environment. In *Seventh International Conference on Advenced Robotics ICAR'95*, pages 998–1007, Sant Feliu de Guixols, Catalonia, Spain, September 1995.

[DevPar98] Michel Devy and Carlos Parra. 3d scene modelling and curve-based localization in natural environments. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, pages 309–3096, Leuven, Belgium, May 1998.

[DisNewCla01] M.W.M. Gamini Dissanayake, Paul Newman, Steven Clark, and H.F. Durrant-Whyte. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, 17(3):229–241, 2001.

[DubSie98] Artur Dubrawski and Barbara Siemiatkoska. A method for tracking pose of a mobile robot equipped with a scanning laser range finder. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, pages 2518–2523, Leuven, Belgium, May 1998.

[FilDev93] Philippe Fillatreu and Michel Devy. Localization of an autonomous mobile robot from 3d depth images. In *Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1881–1888, Yokohama, Japan, July 1993.

[Fox98] Dieter Fox. *Markov Localization: A Probabilistic Framework for Mobile Robot Localization and Navigation.* PhD thesis, Institute of Computer Science III. University of Bonn. Germany, 1998.

[FoxBurDel99] Dieter Fox, Wolfram Burgard, Frank Dellaert, and Sebastian Thrun. Monte carlo localization: Efficient position

estimation for mobile robots. In *Proceedings of AAAI-99*, 1999.

[FoxBurThr98] Dieter Fox, Wolfram Burgard, Sebastian Thrun, and Armin B. Cremers. Position estimation for mobile robots in dynamic environments. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, Madison, WI, 1998.

[FoxBurThr99] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. *Planning for Sensor-Based Intelligent Robot Systems*, chapter Markov Localization for Reliable Robot Navigation and People Detection. Springer Verlag, 1999.

[Gar99] Santiago Garrido. *Identificación, Estimación Y Control de Sistemas No-Lineales Mediante RGO.* PhD thesis, Depart. de Ingeniería Eléctrica, Electrónica y Automática, Universidad Carlos III de Madrid, 1999.

[GarMorSal99] S. Garrido, L. Moreno, and M. A. Salichs. Identification of state-space models with RGO. In *7th European Congress on Intelligent Techniques and Soft Computing. EUFIT´99*, 1999.

[GarMorSal00] S. Garrido, L. Moreno, and M. A Salichs. Predictive control with restricted genetic optimization. In *European Symposium on Intelligent Techniques. ESIT 2000*, pages 233–238, 2000.

[Gol89] David E. Goldberg. *Genetic Algoritmhs in Search, Optimization and Machine Learning.* Addinson-Wesley Publishing Company, Inc., 1989.

[GuiNeb01] José E. Guivant and Eduardo Mario Nebot. Optimization of the simultaneous localization and map-building algorithm for real-time implementation. *IEEE Transactions on Robotics and Automation*, 17(3):242–257, 2001.

[IocMasNar01] Luca Iocchi, Dominico Mastrantuono, and Daniele Nardi. A probabilistic approach to hough localization. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, pages 4250–4255, 2001.

[KaeLitCas98] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1-2):99–134, 1998.

[KanHayLi99] Toshiniko Kanbara, Akira Hayashi, Shigang Li, Jun Miura, and Yoshiaki Shirai. A method of planning movement and and observation for a mobile robot considering uncertainties of movement, visual sensing, and a map. In *Proceedings of the Ninth International Conference on Advanced Robotics'99 (ICAR)*, pages 375–381, 1999.

[KasKum99] Ashraf A. Kassim and B.V.K. Vijaya Kumar. Path planners based on the wave expansion neural network. *Robotics and Autonomous Systems*, 26:1–22, 1999.

[KeeOngHua99] S.S. Keerthi, C.J. Ong, E. Huang, and E.G. Gilbert. Equidistance diagram- a new roadmap method for path planning. In *Proceedings of the 1999 IEEE Conference on Robotics and Automation*, pages 682–687, 1999.

[KnoNouMor98] Ryan M. Knotts, Illah R. Nourbakshs, and Robert C. Morris. Navigates: A benchmark for indoor navigation. In *Proceedings of the Third International Conference and Exposition on Robotics for Challenging Environments. ASCE.*, 1998.

[KoeSim96] Sven Koenig and Reid G. Simmons. Passive distance learning for robot navigation. In *Proceedings of the Thirteenth International Conference (ICML'96)*, pages 266–274, Bari, Italy, July 1996.

[KorWey94] David Kortenkamp and Terry Weymouth. Topological mapping for mobile robots using a combination of sonar and vision sensing. In *Proceedings of the Twelfth National Conference on Artificial Intelligence.*, pages 979–984, 1994.

[KuiByu91] Benjamin Kuipers and Yung-Tai Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Robotics and Autonomous Systems*, 8:47–63, 1991.

[KunWilNou99] Clayton Kunz, Thomas Willeke, and Illah R. Nourbakshs. Automatic mapping of dynamic office environments. 7, 7(2):131–142, 1999.

[KweKan90] In So Kweon and Takeo Kanade. High resolution terrain map from multiple sensor data. In *IEEE International Workshop on Intelligent Robots and Systems IROS'90*, pages 127–134, 1990.

[Lat91] Jean-Claude Latombe. *Robot Motion Planning.* Kluwer Academic Publishers, 1991.

[LeeChoRiz01] Ji Yeong Lee, Howie Choset, and Alfred A. Rizzi. Sensor based planning for rod shaped robots in three dimensions: Piece-wise retracts of R^3xS^2. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, pages 991–999, 2001.

[LeoWhi91] John J. Leonard and Hugh F. Durrant White. Simultaneous map building and localization for an autonomous mobile robot. In *IEEE/RSJ International Workshop on Intelligent Robots and Systems IROS'91*, pages 1442–1447, Osaka, Japan, Nov. 1991.

[LevHutBur99] Peter Leven, Seth Hutchinson, Darius Burschka, and Georg Färber. Perception-based motion planning for indoor exploration. In *Proceedings of the 1999 IEEE Conference on Robotics and Automation*, pages 695–701, 1999.

[LinHanJud98] Long-Ji Lin, Thomas R. Hancock, and J. Stephen Judd. A robust landmark-based system for vehicle location using low-bandwidth vision. *Robotics and Autonomous Systems*, 25:19–32, 1998.

[MahSli98] R. Mahkovic and T. Slivnik. Generalized local voronoi diagram of visible region. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, pages 349–355, Leuven, Belgium, May 1998.

[MarThr98] Dimitris Margaritis and Sebastian Thrun. Learning to locate an object in 3d space from a sequence of camera images. In *IMCL'98*, 1998.

[MatArmEsc01] M. Mata, J.M. Armingol, A. de la Escalera, and M.A. Salichs. A visual landmark recognition system for topological navigation of mobile robots. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, pages 1124–1129, 2001.

[Max] Max-Planck-Institut für Informatik, Saarbrücken. *The LEDA User Manual. Version 3.7.* http://www.mpi-sb.mpg.de/LEDA.

[Mit96] Melanie Mitchell. *An Introduction to Genetic Algorithms.* The MIT Press, Cambridge, Massachusetts, London, England, 1996.

[MouCha91] Philippe Moutarlier and Raja Chatila. Incremental free-space modelling from uncertain data by an autonomous mobile robot. In *IEEE/RSJ International Workshop on Intelligent Robots and Systems IROS'91*, pages 1052–1058, Osaka, Japan, Nov. 1991.

[NagCho99] K.Ñagatani and H. Choset. Toward robust sensor based exploration by constructing reduced generalized voronoi graph. In *Proc. Of IEEE/RSJ Int. Conference on Intelligent Robots and Systems (IROS'99)*, pages 1687–1698, 1999.

[NagChoThr98] Keiji Nagatani, Howie Choset, and Sebastian Thrun. Towards exact localization without explicit localization with the generalized voronoi graph. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, pages 342–348, Leuven, Belgium, May 1998.

[NasFilWri94] Fawzi Nashashibi, Philippe Fillatreu, Benoit Dacre Wright, and Thierry Simeon. 3d autonomous navigation in a natural environment. In *IEEE International Conference on Robotics and Automation*, pages 433–439, San Diego, California, 1994.

[NeiHorTar96] J.Ñeira, J. Horn, J. D. Tardós, and G. Schmidt. Multisensor mobile robot localization. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, pages 673–679, Minneapolis, Minnesota, April 1996.

[OkaBooSug92] A. Okabe, B. Boots, and K. Sugihara. *Spatial Tessellations. Concepts and Applications of Voronoi Diagrams.* John Wiley and Sons, 1992.

[OlsMat98] Clark F. Olson and Larry H. Matthies. Maximum likelihood rover localization by matching range maps. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, pages 272–277, Leuven, Belgium, May 1998.

[PiaZac98] Maurizio Piaggio and Renato Zaccaria. Using roadmaps to classify regions of space for autonomous robot navigation. *Robotics and Autonomous Systems*, 25:209–217, 1998.

[Rab89] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.

[Raw91] Gregory J.E. Rawlins. *Foundations of Genetic Algorithms.* Morgan Kaufmann Publishers, Inc., 1991.

[Rea96] Real World Interface, Inc. *B21 Users Guide*, 1996.

[RekDujDud99] Ioannis M. Rekleitis, Vida Dujmovic, and Gregory Dudek. Efficient topological exploration. In *Proceedings of the 1999 IEEE Conference on Robotics and Automation*, pages 676–681, Detroit, Michigan, May 1999.

[RoyBurFox99] Nicholas Roy, Wolfram Burgard, Dieter Fox, and Sebastian Thrun. Coastal navigation- mobile robot navigation with uncertainty in dynamic environments. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, pages 35–40, Detroit, Michigan, May 1999.

[SalMor00] M.A. Salichs and L. Moreno. Navigation of mobile robots: Open questions. *Robotica*, 18:227–234, 2000.

[SchAdaYam99] Alan C. Schultz, William Adams, Brian Yamauchi, and Mike Jones. Unifying exploration, localization, navigation and planning though a common representation. In *Proceedings of the 1999 IEEE Conference on Robotics and Automation*, pages 2651–2658, 1999.

[SeLowLit01] Stephen Se, David Lowe, and Jim Little. Vision-based mobile robot localization and mapping using scale-invariant features. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, pages 2051–2058, 2001.

[Sha98] Hagit Shatkay. *Learning Models for Robot Navigation.* PhD thesis, Department of Computer Science. Brown University, December 1998.

[ShaKae97] Hagit Shatkay and Leslie Pack Kaelbling. Learning topological maps with weak local odometric information. In *Proceedings of IJCAI-97. IJCA, Inc., 1997.*, 1997.

[SIC] SICK optic electronic. *PLS and PLS User Software Laser Scanner. Operating Instructions.*

[SimKoe95] Reid Simmons and Sven Koenig. Probabilistic robot navigation in partially observable environments. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJ-CAI)*, pages 1080–1087, 1995.

[SmiChe86] Randall C. Smith and Peter Cheeseman. On the representation and estimation of spatial uncertainty. *The International Journal of Robotics Research*, 5(4):56–68, 1986.

[SmiSelChe88] Randall Smith, Matthew Self, and Peter Cheesman. Estimation uncertain spatial relationships in robotics. *Uncertainty in Artificial Intelligence*, 2:435–461, 1988.

[SudNanSri99] N. Sudha, S.Ñandi, and K. Sridharan. A parallel algorithm to construct voronoi diagram and its VLSI architecture. In *Proceedings of the 1999 IEEE Conference on Robotics and Automation*, pages 1683–1688, 1999.

[Sug93] Kokichi Sugihara. Approximation of generalized voronoi diagrams by ordinary voronoi diagrams. *CVGIP: Graphical Models and Image Processing*, 55(6):522–531, 1993.

[Tay76] R. H. Taylor. A synthesis of manipulator control programs from task-level specifications. Technical Report AIM-282, Stanford University Artificial Intelligence Laboratory, 1976.

[Thr98a] Sebastian Thrun. Bayesian landmark learning for mobile robot localization. *Machine Learning*, 33(1), 1998.

[Thr98b] Sebastian Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 99(1):21–71, 1998.

[Thr01] Sebastian Thrun. A probabilistic on-line mapping algorithm for teams of mobile robots. *The International Journal of Robotics Research*, 20(5):335–363, 2001.

[ThrBüc96] Sebastian Thrun and Arno Bücken. Integrating grid-based and topological maps for mobile robot navigation. In *Proceedimgs of the Thirteenth National Conference on Artificial Intelligence AAAI'96*, 1996.

[ThrBurFox98a] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. A probabilistic approach to concurrent mapping and localization for mobile robots. *Autonomous Robots*, 5:253–271, 1998.

[ThrBurFox98b] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning and Autonomous Robots (joint issue)*, 31(5):1–25, 1998.

[ThrBurFox00] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping. In *IEEE International Conference on Robotics and Automation*, 2000.

[ThrFoxBur98] Sebastian Thrun, Dieter Fox, and Wolfram Burgard. Probabilistic mapping of an environment by a mobile robot. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 1998.

[ThrGutFox98] Sebastian Thrun, Jens-Steffen Gutmann, Dieter Fox, Wolfram Burgard, and Benjamin J. Kuipers. Integrating topological and metric maps for mobile robot navigation: A statical approach. In *Proceedings AAAI'98*, 1998.

[TomNouArr01] N. Tomatis, I.Ñourbakhsh, K. Arras, and R. Siegwart. A hybrid approach for robous and precise mobile robot navigation with compact environment modeling. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, pages 1111–1116, 2001.

[UlrNou00] I. Ulrich and I.Ñourbakshs. Appearance-based place recognition for topological localization. In *IEEE International Conference on Robotics and Automation*, pages 1023–1029, 2000.

[VicRivBor00] Alessandro Corrêa Victorino, Patrick Rives, and Jean-Jacques Borrely. Localization and map building using a sensor-based control strategy. In *Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2000.

[VlaTsa98] N.A. Vlassis and P. Tsanakas. A sensory uncertainty field model for unkown and non-stacionary mobile robot environments. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation.*, 1998.

[Yam97] Brian Yamauchi. A frontier-based approach for autonomous exploration. In *Proceedings of the 1997 IEEE International Symposioum on Computacional Intelligence in Robotics and Automation*, pages 146–151, Monterey, CA, July 1997.

[YamSchAda98] Brian Yamauchi, Alan Schultz, and William Adams. Mobile robot exploration and map-building with continuous localization. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, Leuven, Belgium, May 1998.

[ZhaGho00] Li Zhang and Bijoy K. Ghosh. Geometric feature based 2 1/2d map building and planning with laser, sonar and tactile sensors. In *Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2000.

[Zwy00] Dominique Van Zwynsvoorde. *Construction Incrémentale de Modèles Topologiques Pour la Navigation D'un Robot*

Mobile. PhD thesis, Groupe Robotique et IA du LAAS, 2000.

[ZwySimAla00] D. Van Zwynsvoorde, T. Simeon, and R. Alami. Incremental topological modeling using local voronoi-like graphs. In *Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2000.

[ZwySimAla01] D. Van Zwynsvoorde, T. Simeon, and R. Alami. Building topological models for navigation in large scale environments. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, 2001.