INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

TECNOLÓGICO DE MONTERREY®

”Navegación de un Robot Móvil en Ambientes Interiores Usando Marcas Naturales del Ambiente”

Presenta:
Sergio Francisco Hernández Alamilla

Sometido al Programa de Graduados en Informática y Computación en cumplimiento parcial con los requerimientos para obtener el grado de:
Maestría en Ciencias de la Computación

Asesor:
Dr. Eduardo Morales Manzanares

Cuernavaca, Morelos. Noviembre de 2005

Navegación de un Robot Móvil en Ambientes Interiores Usando Marcas Naturales del Ambiente

Presentada por:
Sergio Francisco Hernández Alamilla

Aprobada por:

Dr. Eduardo Morales Manzanares
Profesor-Investigador del Departamento de Computación
ITESM Campus Cuernavaca
Asesor de Tesis

Dr. Luís Enrique Sucar Succar
Profesor-Investigador del Departamento de Computación
ITESM Campus Cuernavaca
Revisor de Tesis

Dr. Leonardo Romero Muñoz
Profesor-Investigador de la Facultad de Ingeniería Eléctrica
Universidad Michoacana de San Nicolás de Hidalgo
Revisor de Tesis

A mi familia

Sucede que en muy pocas y extraordinarias ocasiones un par de personas especiales se conocen. Y pudiendo solamente amarse, se les ocurre hacerlo de la manera difícil: se comprometen de verdad. Quizá ellos no se dan cuenta, pero la evidencia de amor que ofrecen a todo el que se cruza en su camino es su más grande legado. No se lo que me espera en adelante, pero esta vida ya ha valido la pena solamente por el tiempo que hemos compartido. Los amo y me llena de orgullo ser su hijo. A ustedes mamá y papá.

Cuando se tienen unos pocos años y un buen día tu padre te dice: ¡asómate a verla!, ¡es bonita! ¿no crees?, inmediatamente piensas que en adelante tendrás la mitad de los juguetes, la mitad de los dulces, ¡la mitad de la atención!. Varios años después uno se da cuenta de que una hermana no divide, por el contrario multiplica. A mi hermana con todo mi amor. Ha sido muy divertido jugar contigo Yuri.

De entre el montón de recuerdos que guardamos de la infancia, hay algunos que nos inclinan a pensar que hay personas que pueden llegar a querernos como a un hijo sin serlo. Sin importar si lo merecemos o no, su consideración y sus mejores deseos siempre nos acompañan a todas partes. A ustedes tía Azalea y tío Félix con mucho cariño.

Cada vez que me veo reflejado en sus ojos, tan llenos de la vida que se ha ido acumulando, quisiera saber lo que pasa por su mente. ¿Cuál es la causa de semejante emoción?, es posible que sus ojos vean algo que todavía estoy muy lejos de comprender. Mientras tanto y desde mi corto entender, solo puedo agradecerles enormemente por sus plegarias y sus buenos deseos. A mis abuelitas Mari y Rosi con cariño.

Agradecimientos

El correr de la vida trae siempre consigo alegrías y sinsabores. Algunos presentan un reto tan difícil que necesitamos asirnos fuertemente a nuestros más profundos anhelos. Tu apoyo constante y tu compañía me han ayudado a recuperar el ánimo perdido. Has sido más que mi cómplice, me has prestado tus ojos y tus oídos. Mi eterno agradecimiento para ti Gwendy.

Siempre es grato encontrarse con personas amables que además estén dispuestas a compartir su conocimiento sin prejuicios. Pero más valioso resulta cuando todo ello está respaldado por un ejemplo de vida. Gracias por su apoyo y paciencia Dr Eduardo Morales.

A todos mis amigos, los más y los menos, todos ustedes han dejado una pequeña fracción de si mismos en mi bitácora. Ha sido un placer simplemente coincidir.

A todos mis compañeros de posgrado, siempre resulta enriquecedor compartir con personas de mente clara y ojos bien abiertos. En especial a Blanca Vargas, Cinhtia González y Víctor Jáquez; ha sido un enorme placer.

A mis sinodales: Enrique Sucar y Leonardo Romero por sus valiosas observaciones y sugerencias, pero sobre todo por su disposición.

A la Cátedra de Robótica Móvil por el apoyo recibido, formar estos grupos es una estupenda idea.

Gracias a todos mis profesores: en especial a mi maestra de toda la vida que me enseño a leer y escribir, a mi profesor de literatura de secundaria por su estupenda narrativa, a mi profesor de física de la preparatoria por el pueril idealismo, al buen Manuel Martín, mi tutor en la universidad, por su contagioso entusiasmo. ¡Muchos profesores de estos señores!

Por último, gracias a Gabriel García Márquez, Ernesto Guevara, Sixto Valencia, Hans Christian Andersen, Eusebio, Mario Puzo, Giacomo Girolamo, Casiopea, Jaime Sabines, Fernando Delgadillo, Manuel Bernal, Rubén Darío, José Angel Buesa, gracias por todos los momentos.

Resumen

En los inicios de la robótica se pensó en los robots como sustitutos del hombre para realizar tareas repetitivas en ambientes controlados. Poco a poco los robots han ido evolucionando, incursionando en nuevos ambientes hasta llegar a las oficinas o las casas de las personas. Esta evolución ha traído consigo nuevos retos, debido a que la realización de tareas fuera de ambientes controlados exige nuevas habilidades y capacidades diferentes. Esta tesis presenta los resultados de la investigación sobre un problema relacionado con la operación de un robot móvil en ambientes interiores: el de navegación autónoma. Este problema consiste en dotar a un robot móvil con la capacidad de moverse de un lugar a otro del ambiente manteniéndose localizado durante el trayecto.

El problema de navegación se divide en tres partes: planificación de trayectorias, seguimiento de la posición y evasión de obstáculos. De manera adicional, se propone un algoritmo de localización global para determinar la ubicación del robot dentro del ambiente, cuando no se tiene información de la posición previa.

La planificación de trayectorias se hace en base a un mapa del ambiente previamente construido. Para determinar la trayectoria, se utiliza un algoritmo de programación dinámica para asignar costos a las celdas del mapa. La trayectoria se construye reduciendo los giros y manteniéndose alejado de los obstáculos. Para el seguimiento de la posición se utilizan marcas naturales del ambiente, específicamente discontinuidades, esquinas y paredes. Estas marcas son usadas como puntos de referencia para mantener al robot localizado mientras se mueve por lo que no se requiere modificar el ambiente. El método empleado para el seguimiento de la posición considera la imprecisión en los sensores y actuadores del robot. Para la evasión de obstáculos, el robot fusiona la información de dos tipos de sensores: láser y ultrasónico. Ante la presencia de obstáculos imprevistos (cristales en algunos casos) que bloquean su camino, el robot enriquece el mapa del ambiente con los nuevos obstáculos y calcula una nueva trayectoria que le permita llegar a su destino.

Se presentan experimentos realizados tanto en simulación como en ambientes reales con buenos resultados. En ellos se muestra que el robot es capaz de localizarse y navegar de manera efectiva empleando los métodos propuestos. Los experimentos fueron realizados en diversos espacios dentro de las instalaciones del ITESM Campus Cuernavaca.

Índice general

Índice de figuras

Índice de tablas

Índice de algoritmos

Capítulo 1

Introducción

Esta tesis aborda un problema que se presenta al utilizar robots móviles: el problema de navegación. Básicamente el problema consiste en dotar a un robot móvil con la capacidad de desplazarse de un lugar a otro dentro de un ambiente. Para lograrlo el robot debe ser capaz de determinar si existe una trayectoria del lugar donde se encuentra a un punto destino, una vez determinada la trayectoria, el robot debe moverse a través de ella usando sus sensores para detectar obstáculos en su camino y mantenerse localizado en dicho ambiente. Enseguida se presenta el contexto en el que se desarrolla el trabajo de tesis; sus aplicaciones, las principales dificultades asociadas con el problema, el objetivo de la tesis, sus alcances y el enfoque utilizado.

1.1. Motivación

El presente trabajo de tesis se desarrolla como parte de las actividades de investigación de la Cátedra de Robótica Móvil del Tecnológico de Monterrey Campus Cuernavaca. El trabajo consiste en desarrollar un sistema de navegación para un robot móvil en ambientes interiores. Se planea que el sistema de navegación pueda ser incorporado a través de una arquitectura a un sistema robótico completo. El funcionamiento general de dicho sistema estará relacionado con las necesidades de la tarea que el robot realice, aunque en general se piensa en un robot de servicio.

Es importante para la construcción del sistema de navegación tomar en cuenta

la capacidad y precisión de los actuadores y sensores con los que se cuenta físicamente, así como las características de los ambientes interiores reales y de los robots de prueba. La meta final, es que el trabajo desarrollado sea implementado en un robot real con éxito.

1.2. Antecedentes

En la actualidad, es cada vez más común encontrar robots que realizan actividades que anteriormente estaban confinadas exclusivamente a los humanos. Los robots han venido a solucionar la necesidad de realizar con rapidez y precisión un gran número de tareas.

Inicialmente se pensó en los robots desde el punto de vista práctico, como sustitutos del hombre para operar en condiciones peligrosas o en ambientes muy adversos, aprovechando además, su capacidad para realizar tareas repetitivas previamente programadas. Sin embargo, debido a que en las últimas décadas se ha observado un gran desarrollo en el campo de la inteligencia artificial, poco a poco se ha ido llevando a los robots de los parques industriales y laboratorios de pruebas, a museos, hospitales e incluso a los hogares.

Esta evolución ha traído consigo nuevos retos, ya que el desarrollo de actividades fuera de ambientes controlados como un laboratorio o una línea de producción, exige nuevas habilidades y capacidades diferentes. Debido a la incursión de los robots en estos nuevos ambientes, se hace cada vez mayor la necesidad de contar con sistemas que le permitan al robot un alto grado de autonomía; entendiendo por autonomía la capacidad para tomar decisiones y actuar en base a la información percibida del ambiente de acuerdo con un plan [Xie, 2003, Dudek and Jenkin, 2000].

En el caso de los robots de servicio, esta autonomía trae consigo la necesidad de contar con sistemas robustos que le permitan al robot realizar sus actividades dentro de un ambiente dinámico y en principio desconocido. Por otro lado, el mecanismo de toma de decisiones de un robot móvil debe considerar aspectos que afectan directamente el funcionamiento del mismo, como los cambios en el ambiente y la incertidumbre ocasionada por la inexactitud de sensores y actuadores [Dudek and Jenkin, 2000].

1.3. Definición del problema

El sistema de navegación de un robot móvil consiste en varias tareas. Básicamente navegar consiste en determinar un camino que une un punto origen con un punto destino y su correcto seguimiento sin colisiones [Ribeiro and Lima, 2002, Costa et al., 1990]. A continuación se describen las principales tareas de un sistema de navegación.

1.3.1. Principales tareas de un sistema de navegación

El sistema de navegación necesita una representación del ambiente en el que se encuentra el robot. Esta representación le proporciona una referencia para ubicarse dentro de su espacio, además, le da la capacidad de realizar las actualizaciones necesarias de acuerdo con los cambios en el ambiente [Dudek and Jenkin, 2000].

Como se mencionó anteriormente, es necesario un procedimiento que le permita al robot encontrar un camino para llegar de un lugar a otro y determinar el conjunto de movimientos necesarios de acuerdo con sus capacidades motrices.

Otro procedimiento consiste en el seguimiento del camino determinado tomando en cuenta la incertidumbre presente en los sensores y actuadores del robot. Inicialmente se cuenta con el odómetro del robot, sin embargo, debido a factores externos, como el tipo de suelo, el robot puede sufrir deslizamientos sin que el odómetro lo registre. Debido a esto es necesario complementar la información que ofrece el odómetro con un mecanismo de seguimiento de la posición que permita corregir dichos errores [Xie, 2003, Dudek and Jenkin, 2000, Bergasa et al., 2002, Santiso et al., 2003, Wijk and Christenseny, 2000].

Además debe establecer una estrategia que le permita al robot reaccionar ante los cambios imprevistos en el ambiente, evitando obstáculos y estableciendo una estrategia que le permita modificar la trayectoria planeada inicialmente en caso de que no sea posible llegar a su destino a través de ella [Dixon and Henlich, 1997, Hwang and Ahuja, 1992]. Debido a su complejidad, el problema de navegación puede dividirse en varios problemas. Considerando los aspectos mencionados, podemos considerar tres principales problemas que debe resolver un sistema de navegación los cuales se describen en la Tabla 1.1.

Tabla 1.1: Principales tareas de un sistema de navegación

Planeación de trayectorias	Determinar si existe una trayectoria de la posición actual a una meta, que disminuya el error odométrico y la posibilidad de chocar.
Seguimiento de la posición	Mantener localizado al robot en el mapa y apegarse a la trayectoria planeada.
Evasión de Obstáculos	Detectar obstáculos imprevistos y evadirlos.

Esta división permite atacar cada uno de los problemas de manera independiente sin perder de vista el objetivo general. El problema de navegación no es un problema trivial. Para entender la naturaleza del problema, enseguida se describen los factores que causan dificultades a la hora de realizar una propuesta de solución, así como las características de los sensores de los robots móviles utilizados en esta investigación.

1.3.2. Principales dificultades

A continuación se mencionan algunos de los factores que imponen limitaciones prácticas en los robots móviles autónomos para realizar una navegación efectiva [Dudek and Jenkin, 2000]:

- Sensores: Los sensores, en general, proporcionan información de bajo nivel, y no son capaces de medir las características importantes para la tarea de navegación. Por ejemplo, obtienen distancias del robot a los obstáculos a diferentes orientaciones, mientras que para navegación las características importantes serían la cercanía con una puerta o con una pared.
- Ruido en los sensores: Las mediciones en los sensores son generalmente ruidosas, y frecuentemente es muy difícil estimar las distribuciones del ruido que presentan, incluso en el caso de los sensores más precisos como los sensores láser, existen fenómenos como la reflexión o los objetos transparentes que limitan su capacidad perceptual.

- Deslizamientos: El movimiento del robot es en principio impreciso. Si solo se toma en cuenta el odómetro, el error se acumula en el tiempo. Estos errores se presentan, por ejemplo al pasar por algún borde o irregularidad en el piso.

1.3.3. Características del robot

Como se mencionó con anterioridad y para efectos prácticos de este trabajo, el desarrollo de un sistema de navegación estará ligado con el robot para el cual es diseñado, en este caso, se utilizará un robot diferencial de la empresa Nomadic, el modelo es el Nomad Super Scout II, el cual es un robot de forma cilíndrica con capacidad de giro sobre su propio eje. El robot cuenta con computadora Pentium MMX interna, 16 sonares Polaroid con ángulo de dispersión de 25^o organizados en anillo alrededor del robot, 16 sensores táctiles independientes con sensibilidad de 8 onzas organizados también en anillo, un odómetro que estima la distancia recorrida en décimos de pulgada y la orientación en décimos de grados con respecto a la posición y orientación iniciales.

De manera adicional a los sensores del robot, se cuenta con un sensor láser, el modelo es el Sick LMS 200-30106 [SICK-LMS200, 2003]. Este sensor devuelve la distancia al obstáculo más cercano sobre un plano paralelo al piso, en un rango máximo de 180 grados al frente del robot con una resolución de hasta 0.25 grados y un error de $\pm$15mm en la distancia medida. En la Figura 1.1 se pueden observar el robot y sensor mencionados.

1.4. Objetivos

El objetivo general es desarrollar métodos eficientes para dotar a un robot móvil con la capacidad de navegación dentro de un ambiente interior tomando en cuenta la incertidumbre y las limitaciones perceptuales de los sensores.

Los objetivos específicos son los siguientes:

- Desarrollo de un algoritmo de planeación que le permita al robot determinar la trayectoria y el conjunto de movimientos que debe realizar para llegar a su destino de acuerdo con sus capacidades motrices.

(a) Robot Nomad Super Scout II de la empresa Nomadic.

(b) SICK LMS 200

Figura 1.1: Imágenes del robot y sensor con que se cuenta.

- Desarrollo de un algoritmo para el seguimiento de la posición considerando incertidumbre en sensores y actuadores.
- Desarrollo de una estrategia que permita al robot evitar colisiones con obstáculos estáticos imprevistos, durante el seguimiento de una trayectoria.

Adicionalmente se desarrolló también un algoritmo de localización global, con la finalidad de dotar al robot con la capacidad de determinar su ubicación dentro del ambiente. Esta necesidad se presenta cuando el robot es encendido y no conoce su ubicación, por lo que servirá para determinar el punto de partida para realizar la tarea de navegación.

Los alcances y limitaciones de esta tesis, así como las principales ideas que la fundamentan, se describen a continuación.

1.5. Alcances y limitaciones

Los alcances y limitaciones de este trabajo de tesis se resumen en los siguientes puntos:

- La navegación se llevará a cabo en ambientes interiores sin marcas artificiales. El robot se mueve en un ambiente tipo oficina, con un piso plano.
- El planificador de trayectorias será capaz de reducir la cantidad de giros necesarios para la ejecución de una trayectoria.
- El algoritmo de evasión de obstáculos solo considera obstáculos imprevistos estáticos.
- El mecanismo para seguimiento de la posición deberá funcionar en ambientes interiores reales no modificados.
- El sistema de navegación estará diseñado para funcionar en un robot diferencial con las características especificadas en la Sección 1.3.3.
- El proceso de construcción del mapa que se requiere inicialmente no será objeto de estudio de este trabajo.
- El problema de navegación en exteriores presenta problemas y ambientes con características muy distintas a las de los ambientes interiores por lo que no es objeto de estudio de este trabajo.

1.6. Enfoque utilizado en esta tesis

En esta tesis se eligió una representación del ambiente basada en celdas cuadradas de igual tamaño. Cada celda contiene información acerca de la ocupación, la cual indica si el espacio correspondiente es un obstáculo o parte del espacio libre. Un ejemplo de este tipo de mapa se muestra en la Figura 1.2.

El mapa es la información que el robot posee acerca del ambiente en el que se encuentra. En función de dicho mapa, el robot requiere saber cual es su ubicación inicial dentro del ambiente. Ante la necesidad de moverse del lugar donde se encuentra hacia una meta, el robot debe determinar una trayectoria. Enseguida debe moverse hacia ella, realizando durante el recorrido, un seguimiento de la posición para no perderse. En caso de encontrar un obstáculo, debe detenerse y determinar una nueva trayectoria que le permita llegar a la meta. Un esquema general del proceso descrito anteriormente se muestra en la Figura 1.3:

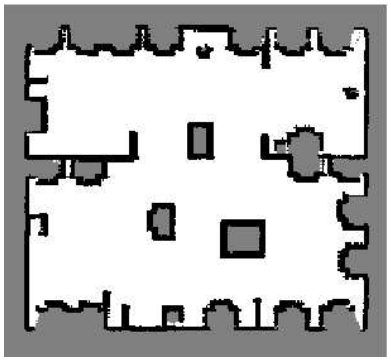


Figura 1.2: Ejemplo de un mapa del ambiente: cada pixel de la imagen mostrada corresponde con una celda.

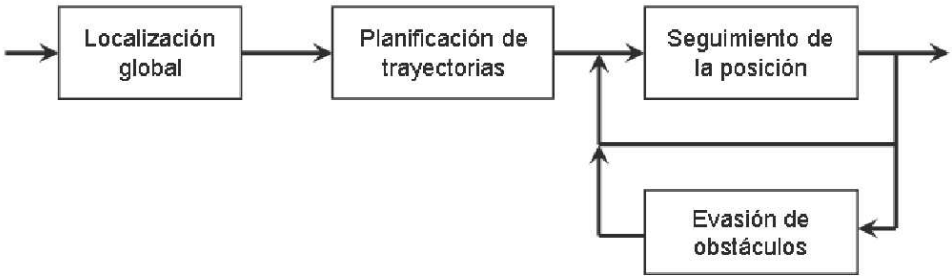


Figura 1.3: Proceso general del sistema de navegación.

De acuerdo con el esquema de la Figura 1.3 los componentes principales son los siguientes:

- **Localización global:** Determina la posición del robot dentro del ambiente en base al mapa.
- **Planificador de trayectorias:** Determina la trayectoria y los movimientos que requiere el robot para ir del lugar donde se encuentra a su destino.
- **Seguimiento de la posición:** Determina el desplazamiento que ha sufrido el robot hasta ese momento fusionando la información adquirida por medio de los sensores del robot con la información de desplazamiento que proporciona el odómetro.

- **Evasión de Obstáculos:** Detecta obstáculos imprevistos que le impidan al robot recorrer la trayectoria planeada inicialmente y determina una trayectoria alterna.

A lo largo de esta tesis, se abordan todos estos componentes en los diferentes capítulos que la conforman. En ellos, se presentan propuestas de solución para cada uno de los problemas en cuestión. Para la elaboración de dichas propuestas se hace uso de las ideas y métodos expuestos por otros investigadores. De manera complementaria se plantean nuevas ideas con la intención de mejorar o hacer más eficientes algunos procesos. Algunas de las ideas planteadas en esta tesis se mencionan a continuación.

1.7. Contribuciones

- Planeación de trayectorias: Se parte de la base de un algoritmo de programación dinámica para determinar una trayectoria inicial que el robot debe seguir para llegar a su destino. Se propone un proceso de refinamiento de la trayectoria inicial que permita reducir la cantidad de giros necesarios para realizar el recorrido.
- Identificación de marcas: Se toman como base diferentes algoritmos para la identificación de marcas naturales del ambiente. Se plantea el uso de una transformada de Hough local para la identificación efectiva de paredes. Se propone el uso de tres diferentes tipos de marcas naturales con la finalidad de proporcionarle al robot información suficiente para mantenerse localizado.
- Seguimiento de la posición: Se parte del uso de marcas naturales como puntos de referencia en el ambiente. Se toman en consideración técnicas como triangulación para determinar la posición del robot. Se propone un esquema de fusión de múltiples estimaciones que reduce el error ocasionado por la imprecisión en la identificación de marcas.
- Localización global: Se parte de la construcción de un modelo del ambiente basado en marcas naturales. Se propone un algoritmo basado en similitud para determinar la ubicación del robot basado en la correspondencia entre las marcas que el robot observa y las almacenadas en el modelo.

1.8. Organización del documento

La tesis está organizada en 9 capítulos que cubren la metodología utilizada en la realización de esta tesis. A continuación se presenta el contenido de la tesis.

En el Capítulo 2 se revisa la investigación previa relacionada con esta tesis.

En el Capítulo 3 se describe el algoritmo de planificación de trayectorias.

En el Capítulo 4 se presentan los métodos de identificación de marcas naturales que serán utilizados en el Capítulo 5 para hacer el seguimiento de la posición.

En el Capítulo 5 se presenta el método para el seguimiento de la posición.

En el Capítulo 6 se describe el control de navegación y la estrategia de evasión de obstáculos propuesta.

En el Capítulo 7 se describe un algoritmo propuesto de localización global.

En el Capítulo 8 se exhiben algunos experimentos y los resultados obtenidos.

Por último en el Capítulo 9 se presentan las conclusiones, comentarios finales y algunas propuestas para trabajos futuros.

Capítulo 2

Trabajos previos

En esté capítulo se describen los trabajos previos relacionados con esta investigación. Para una mayor claridad, esta descripción se divide en cuatro secciones, en las cuales se aborda el trabajo relacionado con cada uno de los tres problemas principales que componen el problema de navegación: planeación de trayectorias, seguimiento de la posición y evasión de obstáculos, definidos en la Tabla 1.1 de la página 4 y una sección referente al problema de localización global. En cada sección se revisan los métodos relacionados señalando las ventajas y desventajas de los mismos, además de la relación entre el problema particular que resuelven y el abordado en esta tesis.

2.1. Planeación de trayectorias

El planificador de trayectorias depende directamente del tipo de mapa que se tiene para realizar dicha tarea [Tomatis, 2001, Hwang and Ahuja, 1992], En términos generales existen dos tipos de planificadores: los métricos y los topológicos.

Los planificadores de trayectorias métricos toman como base un mapa métrico del ambiente y determinan la secuencia de celdas a las que el robot debe moverse para llegar a la meta. Dependiendo del algoritmo empleado, la ruta resultante puede tener diferentes características como: permanecer lejana a los obstáculos, no alejarse a más de cierta distancia de los obstáculos, pasar por el centro de las habitaciones, reducir el número de giros durante la trayectoria, debido a que los giros son los que introducen el

mayor error odométrico [Romero, 2001]. Estas características deseadas dependen del tipo de robot y de la capacidad de sus sensores ya que mientras algunos robots tienen sensores con alcance limitado otros pueden tener sensores con mayor alcance [Hwang and Ahuja, 1992].

En [Romero, 2001] se propone una técnica basada en un algoritmo de iteración de valor que penaliza los giros. En ella el robot se mantiene lo más alejado posible de los obstáculos en lugares estrechos y no se aleja más allá de su capacidad sensorial en espacios abiertos. Este planteamiento se debió a que los sensores utilizados tenían un alcance muy limitado, por lo que se consideró peligroso que el robot se alejara de los obstáculos más allá del alcance de sus sensores. Por otro lado la lejanía deseada en espacios estrechos se debe a que con ello, la posibilidad de que el robot choque se ve reducida.

En el método desarrollado en esta tesis también busca reducir los giros, y se prefiere la lejanía de los obstáculos en lugares estrechos, sin embargo, debido al sensor láser utilizado, no se considera necesario que el robot se mantenga a cierta distancia de los obstáculos en espacios abiertos, por el contrario, el planificador de trayectorias favorecerá las rutas que pasen lejos de los obstáculos en espacios abiertos, como habitaciones o salas. La razón no solamente se debe a la naturaleza del sensor utilizado, además se considera importante que el robot se mueva alejado de los obstáculos, debido a que de esta manera, tiene un campo de visión más amplío y puede identificar un mayor número de marcas naturales del ambiente, las cuales son utilizadas para realizar el seguimiento de la posición. En el Capítulo 3 se presenta un enfoque basado en un algoritmo de programación dinámica que genera trayectorias alejadas de los obstáculos. Para la reducción del número de giros se propone un post-procesamiento de la trayectoria obtenida. Esto último con la finalidad de identificar trayectorias alternativas que impliquen un menor número de giros.

El segundo tipo de planificadores de trayectorias son los topológicos, estos regularmente disponen de un grafo en donde cada nodo representa una región dentro del ambiente. La planificación en este caso se lleva a cabo mediante algoritmos clásicos de teoría de grafos como el del camino más corto de Dijkstra. En [Jefferies et al., 2001] se construye un mapa topológico con un conjunto de habitaciones o espacios locales, los cuales son relacionados por medio de las salidas que llevan de un espacio local a otro, de esta manera el resultado es un grafo que contiene información global del ambiente.

También existen enfoques híbridos como el presentado en [Tomatis, 2001] en donde la planeación de la trayectoria global de realiza en función de un mapa topológi-

co general, posteriormente esta trayectoria se traduce en información métrica que le permite determinar al robot como moverse de un nodo del mapa topológico al siguiente.

En [Hwang and Ahuja, 1992] se exhibe una recopilación de diferentes técnicas usadas en planificación de trayectorias así como una descripción de una gran variedad de algoritmos en detalle.

2.2. Seguimiento de la posición

Algunos sistemas de seguimiento de la posición han empleado sensores ultrasónicos, sin embargo, se ha observado que su resolución es muy limitada por lo que, en general, es necesario hacer un procesamiento adicional para extraer información útil para el seguimiento de la posición [Xiaowei et al., 2000, Wijk and Christenseny, 2000]. Una alternativa son los sensores láser, los cuales proporcionan información más precisa y detallada acerca del ambiente, lo cual es frecuentemente aprovechado para extraer marcas naturales que puedan ser utilizadas para el seguimiento de la posición. Por otro lado la velocidad de actualización es mayor que la de los sensores ultrasónicos [Santiso et al., 2003, Wang et al., 2002, Xiaowei et al., 2000, Ribeiro and Concalves, 1996, Einsele, 2001]. También se han empleado técnicas de visión para afrontar este problema. En [Lowe et al., 2002] se describe un mecanismo basado en visión stereo que en combinación con la identificación de marcas invariantes a la rotación y escala permite calcular de manera aproximada la magnitud del desplazamiento del robot.

En la literatura relacionada se observa que hay dos enfoques que han sido frecuentemente utilizados en la solución del problema de seguimiento: el basado en correlación y el basado en características [Dudek and Jenkin, 2000]. Las propuestas basadas en correlación son ampliamente utilizadas debido a que pueden llegar a ser muy exactas y apegadas al modelo. Sin embargo, este procedimiento puede ser muy costoso en términos de tiempo si consideramos que debe averiguarse en que posición se encuentra el robot utilizando el grueso de los datos sensados. En el caso del uso de celdas este problema aumenta conforme disminuye el tamaño de las celdas. En [Schiele and Crowley, 1994] puede encontrarse un estudio comparativo acerca de diferentes enfoques basados en celdas donde algunos de ellos utilizan correlación.

Por otro lado los enfoques orientados a características aprovechan la informa-

ción de las marcas que el robot es capaz de detectar en el ambiente. Las marcas pueden ser artificiales o naturales, siendo estas últimas las que le dan un mayor grado de autonomía al robot. Uno de los principales argumentos en contra de este enfoque radica en que la identificación de las marcas debe ser muy precisa para obtener una buena estimación de la posición. Por otro lado el tiempo empleado es menor ya que se usan menos datos siempre y cuando al detección de las marcas se realice rápida y efectivamente [Xiaowei et al., 2000, Ribeiro and Concalves, 1996].

Independientemente de si se usa el grueso de la información sensada o solamente un conjunto de marcas extraídas, es necesario contar con un mecanismo que permita fusionar la información del odómetro con las observaciones. Al respecto existen diferentes enfoques, algunos de ellos se basan en la combinación de las estimaciones proporcionadas por las marcas y por medio de triangulación determinan la posición del robot [Xiaowei et al., 2000, Dudek and Jenkin, 2000]. Otros enfoques resuelven el problema por medio de la utilización de un filtro de Kalman, el cual ha sido ampliamente utilizado ya que, en teoría, permite determinar el error de manera óptima; para que la optimalidad sea efectiva, se debe cumplir que exista una relación lineal entre el estado del sistema (x, y, θ) y las observaciones (información obtenida de los sensores). Esta última condición y otras suposiciones como considerar que el ruido es gaussiano, raramente se cumplen en el contexto de robótica móvil por lo que la condición de optimalidad se pierde y en algunos casos el resultado que se obtiene es muy pobre [Dudek and Jenkin, 2000]. En otros sistemas se ha optado por la utilización del filtro de Kalman extendido, el cual básicamente se encarga de linealizar el modelo del sistema por medio de series de Taylor para poder utilizar la teoría del filtro de Kalman básico [Dudek and Jenkin, 2000, Tomatis, 2001].

En esta tesis se utiliza un enfoque basado en marcas naturales del ambiente. Esto evita que el ambiente tenga que ser modificado para que el robot pueda navegar en él. Se plantea el uso de tres tipos de marcas naturales diferentes con la finalidad de que el robot siempre disponga de información para localizarse. Además, se propone un esquema de fusión de múltiples estimaciones para atenuar el error en la estimación de la posición ocasionado por la detección imprecisa de algunas marcas. Las cuestiones relacionadas con la identificación de marcas naturales y seguimiento de la posición serán abordadas en los Capítulos 4 y 5.

2.3. Evasión de obstáculos

La evasión de obstáculos es necesaria cuando el robot se encuentra con un obstáculo imprevisto mientras está ejecutando una trayectoria dada por el planificador de trayectorias. En ambientes reales, esto sucede frecuentemente debido a que las personas suelen dejar fuera de lugar algunos objetos como sillas, mesas o cajas. Cuando esto ocurre, es necesario que el robot sea capaz de modificar la ruta que originalmente tenía para llegar a la meta, o en su defecto, determinar que no es posible llegar a ella [Dudek and Jenkin, 2000, Hwang and Ahuja, 1992].

Para definir un mecanismo de evasión de obstáculos, se debe considerar en primer lugar si los obstáculos imprevistos son estáticos o dinámicos ya que de esto dependen muchas de las características de diseño.

En respuesta a este problema se han desarrollado mecanismos dependientes de técnicas de control reactivo como las basadas en campos de potencial en donde los obstáculos se consideran como emisores de energía, la intención es que el robot se mueva en la dirección de la meta mientras rodea los obstáculos [Hwang and Ahuja, 1992]. También se han empleado enfoques basados en aprendizaje donde se pretende que el robot aprenda cuales son los movimientos necesarios para la evasión de un obstáculo. En [Zeng and Weng, 2004] se describe un esquema con estas características donde el robot aprende a asociar las acciones realizadas con los datos que percibe el robot por medio de sus sensores.

En esta tesis se propone un esquema simple de fusión sensorial para la detección de obstáculos que combina sensores ultrasónicos y láser. Una vez que el obstáculo es detectado se incluye de manera temporal en el mapa del ambiente. Para determinar la trayectoria alterna se utiliza nuevamente el planificador de trayectorias sobre el mapa actualizado. Dicho esquema se describe en detalle en el Capítulo 6.

2.4. Localización global

Básicamente existen dos tipos de localización: local y global. El problema de localización local se abordará en el Capítulo 5 y consisten en llevar a cabo un seguimiento de la posición del robot, para compensar el error odométrico durante la navegación. Para ello se requiere conocer la posición inicial del robot. El problema de localización

global es un problema difícil, ya que se requiere determinar la ubicación del robot sin conocimiento previo se su posición. Incluso más difícil es el problema del rapto del robot [Fox et al., 1999] en el cual el robot es transportado a otro lugar sin notificárselo. Estos problemas son particularmente difíciles en ambientes dinámicos donde nuevos obstáculos pueden alterar las medidas de los sensores del robot [Burgard et al., 1999].

Varios autores han usado modelos basados en celdas para tratar de resolver el problema de localización global. Algunos utilizan un enfoque orientado a la búsqueda basada en la similitud entre observaciones y el modelo. En [MacKenzie and Dudek., 1994] los autores sugieren un mecanismo de muestreo en combinación con una función de similitud. En cada paso se evalúa la función de similitud en cada una de las muestras y permanecen solo las muestran que presenten una alta similitud. Nuevas muestran son generadas en regiones alrededor de las muestras más prometedoras. El principal problema es que para que este mecanismo funcione adecuadamente es necesario definir una función que crezca o decrezca monotonicamente hacia la región del mapa donde se encuentra el robot. De otra manera el mecanismo de muestreo podría no encontrar la solución final.

Otras propuestas para resolver el problema comparten las mismas bases matemáticas: algunos utilizan localización Markoviana, por ejemplo [Thrun et al., 1999], otros usan Filtros de Kalman con múltiples hipótesis los cuales representan la creencia como una mezcla de Gaussianas, por ejemplo [Roumeliotis and Bekey., 2000]. Otros utilizan algoritmos tipo Monte Carlo como en [Thrun et al., 2001]. Los primeros dos asumen que el ruido es Gaussiano y requieren modelos paramétricos. En el tercero se representa la creencia con distribuciones de probabilidad multimodales. Sin embargo, es difícil estimar el número de partículas a utilizar. Todos estos métodos son probabilísticos y requieren que el robot se mueva por el ambiente mientras la distribución de probabilidad converge hacia un solo pico.

En esta tesis se propone un mecanismo simple pero efectivo para resolver el problema de localización global. El algoritmo considera el ruido en los sensores y la presencia de nuevos obstáculos en el ambiente. Para ello, se parte de la construcción de un modelo del ambiente. El modelo almacenará el conjunto de marcas naturales que el robot debería ver desde cada celda de acuerdo con el mapa. Se propone un algoritmo de búsqueda basado en similitud que permite determinar la ubicación del robot basado en la correspondencia entre las marcas que el robot observa a 360° y las almacenadas en el modelo. El detalle del algoritmo se describe en el Capítulo 7.

2.5. Punto de partida de esta investigación

En resumen, los siguientes trabajos servirán como punto de partida para el desarrollo de esta tesis:

- Planeación de trayectorias: Se tomará como base el algoritmo clásico de programación dinámica. De manera adicional se considerará un crecimiento exponencial del costo cercano a los obstáculos como se propone en [Romero, 2001] y [Batavia and Nourbakhsh, 2000]. Se propone también un algoritmo para la construcción de un mapa de cercanía a los obstáculos.

- Seguimiento de la posición: Se partirá de las ideas expuestas en [Xiaowei et al., 2000], [Santiso et al., 2003] y [Dudek and Jenkin, 2000] acerca del uso de marcas naturales del ambiente por medio del establecimiento de la correspondencia entre conjuntos de marcas desde diferentes puntos. Además se propone una validación de las correspondencias en función de la relación geométrica entre marcas. Se partirá del trabajo de [Kelly, 2003] para discriminar estimaciones en función del error esperado y se determinarán atributos adicionales complementarios.

- Localización global: Se tomarán como base las ideas expresadas en [Thrun et al., 2001], aunque con un diferente enfoque. En [Thrun et al., 2001] se plantea el problema en función de distribuciones de probabilidad; en esta tesis se plantea el problema como una búsqueda basada en similitud. La similitud estará determinada por la distribución espacial de las marcas naturales que el robot observa comparadas con un modelo previamente construido. Para ello se utiliza una versión modificada del algoritmo de relajación discreta expuesto en [Stockman and Goshtasby, 2004].

Capítulo 3

Planeación de trayectorias

En este capítulo se expone el método de planificación de trayectorias propuesto. Dado que el robot posee un mapa del ambiente, este debe servirle para realizar varias tareas; una de ellas es la de encontrar una trayectoria y el conjunto de movimientos para ir de un lugar a otro. La construcción de dicho mapa no es objeto de estudio de este trabajo por lo que se asume que existe un mapa de celdas dado. El algoritmo de construcción de mapas es descrito en la tesis de maestría [Jaquez, 2005] la cual se desarrolla a la par de la presente. La información que ofrece dicho mapa es la probabilidad de ocupación de cada celda. Para fines prácticos este rango de valores se reduce a tres clases de celdas: celdas libres, celdas ocupadas y celdas no determinadas. Las dos primeras clases obviamente se refieren a regiones con una alta probabilidad de estar libres u ocupadas respectivamente; la tercera clase son las celdas que el robot no pudo determinar si pertenecen al espacio libre o a un obstáculo y por lo tanto no existe una certeza acerca de su ocupación.

En la Figura 3.1 se muestra el ejemplo de un mapa probabilístico con estas características, en donde cada pixel de la imagen mostrada corresponde a una celda, el color blanco denota al espacio libre, el negro a los obstáculos y el gris las regiones no determinadas. A lo largo de este capítulo se usarán dos mapas para ejemplificar el funcionamiento de los algoritmos utilizados: el primero es mostrado en la Figura 3.1 y el segundo, más sencillo e ilustrativo se muestra en la Figura 3.2. En el segundo mapa el valor 1 representa a los obstáculos y las celdas vacías son celdas libres.

Para realizar la planificación de trayectorias, es necesario conocer la ubicación

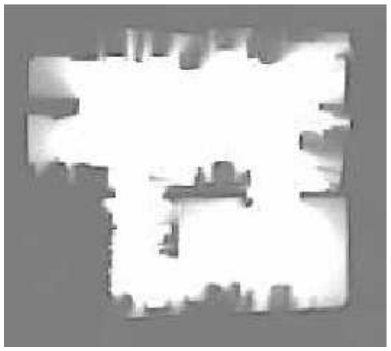


Figura 3.1: Mapa del ambiente.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	1							1							1
3	1							1							1
4	1							1							1
5	1							1							1
6	1							1							1
7	1														1
8	1														1
19	1														1
10	1														1
11	1														1
12	1														1
13	1							1							1
14	1							1							1
15	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figura 3.2: Mapa de celdas simple.

del robot, es decir, la celda en la que el robot se encuentra dentro del ambiente. En el Capítulo 7 se propone un algoritmo de localización global, que permite identificar la celda inicial en cuestión. Para efectos de este capítulo, se asume que el robot conoce su posición dentro del ambiente. El segundo dato que se requiere, es la posición destino del robot, es decir, la celda a la que el robot debe llegar. Estas coordenadas le son proporcionadas por el usuario a través de una interfaz gráfica de manera remota.

3.1. Interfaz remota

Para comodidad del usuario, se desarrolló un programa con interfaz gráfica que le permite indicarle al robot cual es su destino. La interfaz funciona de manera remota por lo que puede presentarse en una pantalla montada encima del robot o en un monitor de cualquier otra computadora. La comunicación con el módulo de planeación de trayectorias se hace por medio de la red. Inicialmente el robot le envía a la interfaz el mapa en forma de imagen en escala de grises. La interfaz le muestra al usuario el mapa del ambiente y basta con hacer un click izquierdo para indicar el lugar dentro del mapa a donde debe trasladarse el robot. En la Figura 3.3 puede verse la interfaz mencionada.

Una vez que el usuario ha indicado la celda destino, las coordenadas le son enviadas al robot y la interfaz quedará en espera de recibir confirmación cuando la meta sea alcanzada o cuando sea imposible llegar a ella. Una vez que el planificador tiene la información de la celda origen y la celda destino, es necesario un procedimiento que le permita al robot determinar cuales son las celdas por las que tiene que pasar y los movimientos necesarios para llegar a la meta. A continuación se presentan los detalles de la planificación de la trayectoria.

3.2. Cálculo de la trayectoria

Dada la posición inicial del robot, si se determina cual es la siguiente acción que debe realizar el robot para llegar a su destino, se tendría una política de movimiento y bastaría con realizar el movimiento indicado de acuerdo con la posición en la que se encuentre el robot. El número de posibles ubicaciones del robot es infinito, por lo que se aprovecha la discretización implícita en el mapa de celdas que se tiene. Se considera

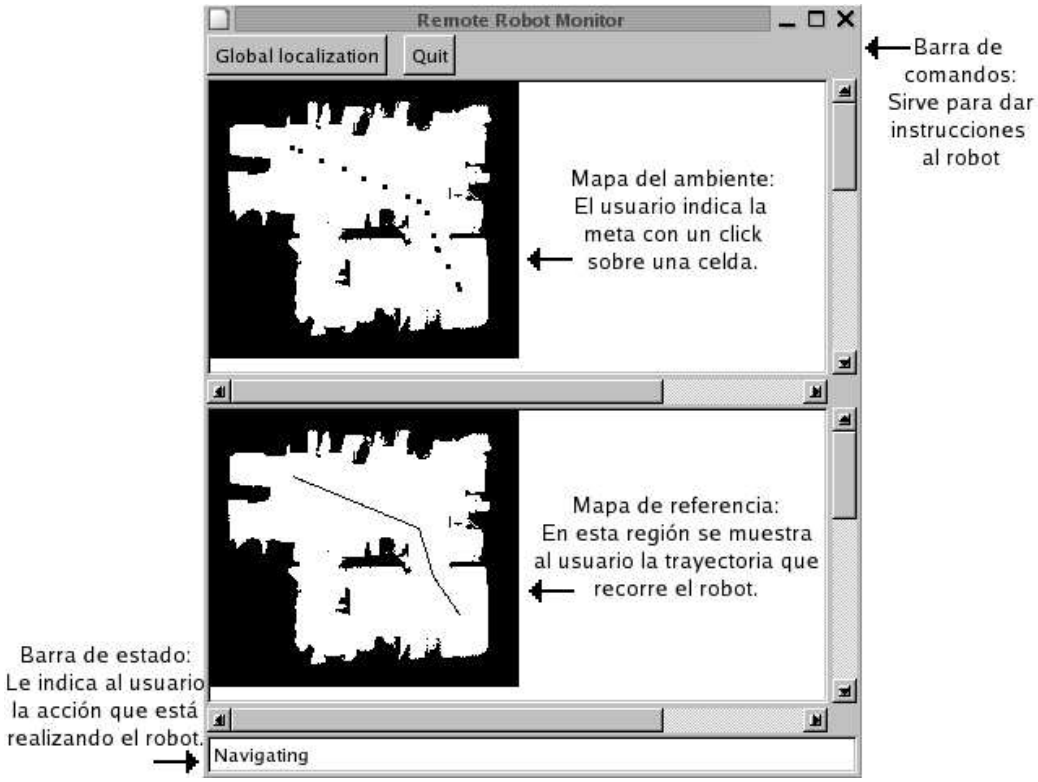


Figura 3.3: Interfaz remota para monitorear las actividades del robot.

que cada celda es una posible ubicación del robot y se define a continuación el esquema de vecindad.

3.2.1. Movimientos permitidos del robot

Se asume que el robot es capaz de moverse hacia cualquiera de las 8 celdas vecinas, en la Figura 3.4 se muestra el esquema de vecindad utilizado.

Cabe hacer notorio, que al moverse a una celda vecina, el robot se acerca, se aleja o se mantiene a la misma distancia de la meta, dependiendo de la celda a la que se mueva. De la misma forma puede requerir de un giro adicional al desplazamiento para llegar a dicha celda. Estos tres factores combinados nos permitirán decidir si un movimiento en general es mejor que otro. Los objetivos generales que se persiguen son

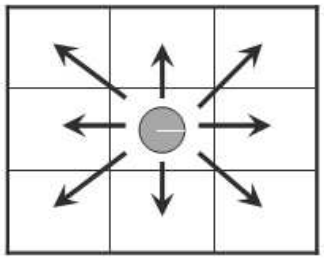


Figura 3.4: Esquema de vecindad de cada celda.

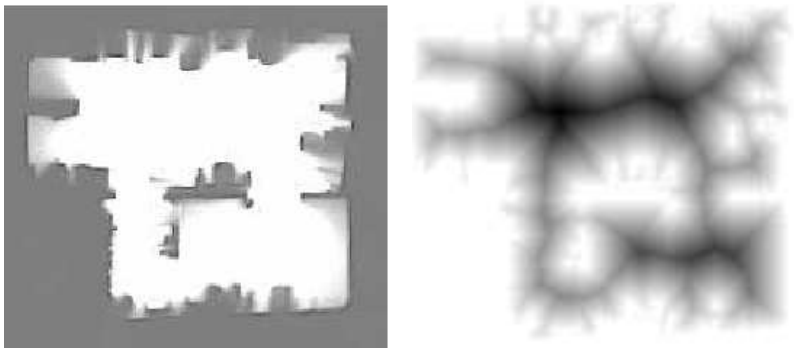
los siguientes en orden de prioridad:

1. Mantenerse alejado de los obstáculos.
2. Acercarse a la meta.
3. Reducir el número de giros requeridos.

Se propone la construcción de un mapa de cercanía a los obstáculos para tener información de los movimientos que acercan al robot a los obstáculos. Se propone también la construcción de un mapa de costos usando el algoritmo de programación dinámica para tener información de los movimientos que acercan al robot a la meta. Por último se propone un proceso de refinamiento de la ruta obtenida para minimizar los giros requeridos para su ejecución.

3.2.2. Cálculo de la cercanía con los obstáculos

Para contar con información de la cercanía de los obstáculos, se construyó un segundo mapa, donde el valor de cada celda corresponde a la distancia entre la celda y el obstáculo más cercano. En la Figura 3.5(b) se muestra el mapa de cercanía a los obstáculos que corresponde con el mapa de la Figura 3.5(a), donde cada celda indica la distancia al obstáculo más próximo. Los pixeles más obscuros representan las celdas más alejadas de los obstáculos. De manera análoga se presenta el mapa de cercanía a los obstáculos en la Figura 3.6 que corresponde con el mapa de la Figura 3.2, donde el valor 0 representa a los obstáculos.



(a) Mapa del ambiente.

(b) Mapa de cercanía a los obstáculos

Figura 3.5: Construcción del mapa de cercanía a obstáculos partiendo del mapa del ambiente.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0
3	0.0	1.0	2.0	2.0	2.0	2.0	1.0	0.0	1.0	2.0	2.0	2.0	2.0	1.0	0.0
4	0.0	1.0	2.0	3.0	3.0	2.0	1.0	0.0	1.0	2.0	3.0	3.0	2.0	1.0	0.0
5	0.0	1.0	2.0	3.0	3.0	2.0	1.0	0.0	1.0	2.0	3.0	3.0	2.0	1.0	0.0
6	0.0	1.0	2.0	3.0	3.0	2.0	1.0	0.0	1.0	2.0	3.0	3.0	2.0	1.0	0.0
7	0.0	1.0	2.0	3.0	3.4	2.4	1.4	1.0	1.4	2.4	3.4	3.0	2.0	1.0	0.0
8	0.0	1.0	2.0	3.0	3.8	2.8	2.4	2.0	2.4	2.8	3.8	3.0	2.0	1.0	0.0
9	0.0	1.0	2.0	3.0	4.0	3.8	3.4	3.0	3.4	3.8	4.0	3.0	2.0	1.0	0.0
10	0.0	1.0	2.0	3.0	4.0	3.8	3.4	3.0	3.4	3.8	4.0	3.0	2.0	1.0	0.0
11	0.0	1.0	2.0	3.0	3.8	2.8	2.4	2.0	2.4	2.8	3.8	3.0	2.0	1.0	0.0
12	0.0	1.0	2.0	3.0	3.0	2.4	1.4	1.0	1.4	2.4	3.0	3.0	2.0	1.0	0.0
13	0.0	1.0	2.0	2.0	2.0	2.0	1.0	0.0	1.0	2.0	2.0	2.0	2.0	1.0	0.0
14	0.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0
15	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Figura 3.6: Mapa de cercanía a los obstáculos.

Para la obtención de dicho mapa, se realiza una propagación partiendo de cualquier celda que corresponda a un obstáculo, esta celda se etiqueta con el valor 0, enseguida las celdas vecinas que pertenezcan al espacio libre se etiquetan con un valor v donde el valor de v es $\sqrt{2}$ ó 1, dependiendo si se trata de un movimiento en diagonal o no. Así sucesivamente para la celda c_n con valor v_n, se etiquetan con valor $v_n + v$ las celdas en la vecindad que no han sido asignadas o que tienen un valor mayor a $v_n + v$. Cuando se encuentra en la vecindad de c_n una celda c_o que corresponde a un obstáculo, c_o se etiqueta con valor 0. Cuando la propagación termina, cada celda tiene el valor que corresponde a la distancia al obstáculo más próximo; el pseudo código correspondiente se muestra en el Algoritmo 3.1.

Algoritmo 3.1 Algoritmo para el cálculo de la cercanía a los obstáculos.

```
Sea (lin_0, col_0) una celda obstáculo
Sea mapa[ ][ ] el mapa del ambiente
Sea obst[ ][ ] el mapa de cercanía a obstáculos inicializado en ∞.

procedimiento cercania_obstaculos() {
   Introducir (lin_0, col_0) a la agenda
   repetir {
      (lin, col) = primer elemento de la agenda
      Para cada celda (lin_i, col_i) en la vecindad de (lin, col)
         Si mapa[lin_i][col_i] es obstáculo entonces
            obst[lin_i][col_i] = 0 e Introducir (lin_i, col_i) a la agenda
         Si (lin_i, col_i) está en diagonal entonces v_i = √2
         Si (lin_i, col_i) no está en diagonal entonces v_i = 1
         Si mapa[lin_i][col_i] es espacio libre entonces
            Si (obst[lin][col]+v_i) < obst[lin_i][col_i] entonces
               obst[lin_i][col_i] = (obst[lin][col]+v_i) e Introducir (lin_i, col_i) a la agenda
   } mientras la agenda no esté vacía
}
```

3.2.3. Algoritmo de programación dinámica

La idea básica del algoritmo de programación dinámica consiste en construir un mapa de costos, donde cada celda indica el costo de ir de dicha celda a la meta. El objetivo es que por medio de este mapa de costos, el robot sepa cual es el siguiente mo-

vimiento para llegar a la meta desde cualquier celda en que se encuentre, simplemente eligiendo moverse hacia la celda vecina con menor costo.

Como se verá a continuación el costo sólo depende de la distancia a la meta y de la cercanía de los obstáculos. La reducción de la cantidad de giros se realizará posteriormente procesando la trayectoria obtenida del algoritmo de programación dinámica.

Para construir el mapa de costos, el usuario debe definir previamente dos parámetros: el primero es la distancia deseada entre el robot y el obstáculo más próximo (D_{pref}) y el segundo es la mínima distancia segura entre el robot y el obstáculo más próximo (d_{min}). La distancia deseada D_{pref} entre el robot y el obstáculo más próximo es usada por el planificador para preferir trayectorias a esa distancia de los obstáculos, mientras que la distancia mínima d_{min} es utilizada para agrandar los obstáculos de tal manera que el robot jamás elija moverse hacia una celda por debajo del valor de d_{min}. Cabe aclarar que para efectos prácticos d_{min} siempre deberá ser menor que D_{pref}.

Para construir el mapa de costos se comienza con la celda que corresponde a la meta y se etiqueta con 0, luego se etiquetan sus vecinos con un valor v donde v es $\sqrt{2}$ ó 1, dependiendo si se trata de una celda vecina en diagonal o no. En las regiones cercanas a los obstáculos por debajo de D_{pref} y por encima de d_{min}, para cada celda c_j con cercanía a los obstáculos d_j ($d_{min} < d_j < D_{pref}$), el costo crece exponencialmente a razón de $e^{D_{pref}-d_j}$. Esto ocasionará que el robot solamente elija pasar por esas celdas si no existe una trayectoria que lo mantenga fuera de ella.

No existe una justificación puntual ó formal que respalde la elección de la función $e^{D_{pref}-d_j}$, sin embargo como se sugiere en [Romero, 2001] y como se afirma en [Batavia and Nourbakhsh, 2000] lo realmente importante es que en dicha región el costo no crezca de manera lineal como en el resto del espacio libre y que la base sea mayor a $1+\sqrt{2}$, pasando a segundo plano la elección de la función que se utilice.

En caso de encontrar una celda que corresponda a un obstáculo o cuya distancia al obstáculo más cercano sea menor que d_{min}, se le asigna un costo infinito con la intención de el robot nunca elija ese movimiento. En la práctica se le asigna un valor muy alto; el pseudo código se muestra en el Algoritmo 3.2.

En la Figura 3.8 se puede apreciar el resultado de la propagación partiendo de la meta indicada en la Figura 3.7. Una vez que el mapa de costos es construido, este puede mostrarse como una imagen en 3D. Por analogía, esta gráfica puede interpretarse como una mesa delimitada con una leve pendiente, de tal manera que en cualquier

Algoritmo 3.2 Algoritmo de programación dinámica.

Sean (lin_{meta}, col_{meta}) las coordenadas de la meta en el mapa de celdas.
Sea obst[ ][ ] el mapa de cercanía a los obstáculos que se construyó.
Sea costo[ ][ ] el mapa de costos que se quiere construir

procedimiento *programación_dinámica*() {
  Introducir (lin_{meta}, col_{meta}) a la agenda con costo 0
  repetir {
    (lin, col) = primer elemento de la agenda
    Para cada celda (lin_i, col_i) en la vecindad de (lin, col)
      Si obst[lin_i][col_i] $< d_{min}$ entonces
        costo[lin_i][col_i] = infinito
      Si (lin_i, col_i) está en diagonal entonces $v_i = \sqrt{2}$
      Si (lin_i, col_i) no está en diagonal entonces $v_i = 1$
      Si obst[lin_i][col_i] $< D_{pref}$ entonces $v_i = v_i + e^{(D_{pref} - obst[lin_i][col_i])}$
      Si (costo[lin][col]+v_i) $<$ costo[lin_i][col_i] entonces
        costo[lin_i][col_i] = (costo[lin][col]+v_i) e introducir (lin_i, col_i) a la agenda
  } mientras la agenda no esté vacía
}

lugar que sea colocada una esfera, esta rodará hasta detenerse finalmente en la celda que corresponde a la meta. Las regiones más obscuras se refieren a los costos más grandes, de tal manera que dada cualquier celda inicial dentro del mapa, la estrategia sería: moverse hacia la celda con menor costo, hasta llegar a la celda con costo 0 que es la meta. En la Figura 3.9 puede observarse el detalle de la propagación que corresponde al mapa de la Figura 3.2. La meta se encuentra en (3,11), el robot se encuentra en (3,3), el valor de D_{pref} es 3 y el valor de d_{min} es 1.5; el valor -2 denota infinito.

Este mapa de costos sirve para llegar a una meta en particular desde cualquier punto inicial. Si cambia la meta, entonces habrá que recalcular el costo de las celdas. Este sencillo mecanismo, converge hacia la meta y su implementación es muy simple.

Como se observa en la Figura 3.10, la trayectoria obtenida solamente del algoritmo de programación dinámica no es adecuada si se toma en cuenta que un robot móvil es quien debe recorrerla; para hacerlo, el robot necesitaría realizar una gran cantidad de giros la mayor parte de ellos innecesarios. Esto se debe en gran parte, a que existe una limitante muy importante al considerar que el robot puede moverse sólo ha-

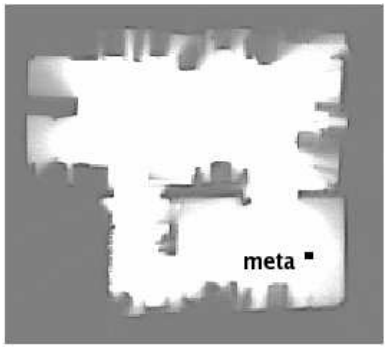


Figura 3.7: La meta es indicada por el usuario

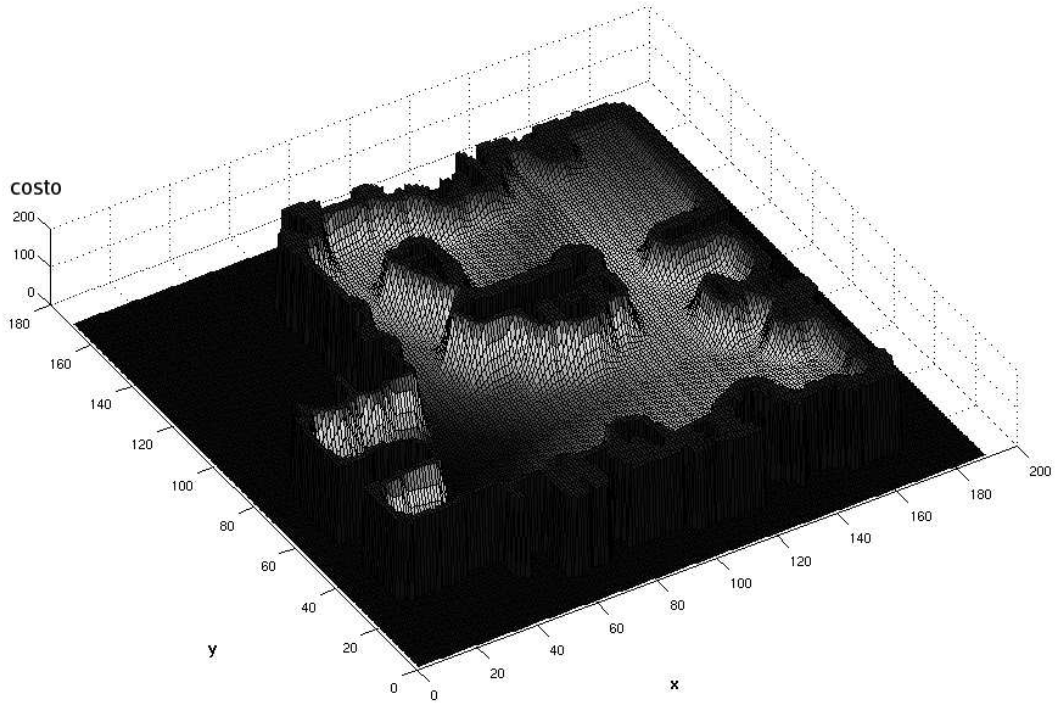


Figura 3.8: Resultado del algoritmo de programación dinámica

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0
2	-1.0	-2.0	-2.0	-2.0	-2.0	-2.0	-2.0	-1.0	-2.0	-2.0	-2.0	-2.0	-2.0	-2.0	-1.0
3	-1.0	-2.0	19.7	19.3	18.9	19.3	-2.0	-1.0	-2.0	05.1	04.1	03.7	04.1	-2.0	-1.0
4	-1.0	-2.0	18.7	15.6	15.2	18.3	-2.0	-1.0	-2.0	04.7	01.0	00.0	03.7	-2.0	-1.0
5	-1.0	-2.0	17.7	14.6	14.2	17.3	-2.0	-1.0	-2.0	05.1	01.4	01.0	04.1	-2.0	-1.0
6	-1.0	-2.0	16.7	13.6	13.2	16.3	-2.0	-1.0	-2.0	05.5	02.4	02.0	05.1	-2.0	-1.0
7	-1.0	-2.0	16.3	12.6	12.2	14.2	-2.0	-2.0	-2.0	05.6	03.4	03.0	06.1	-2.0	-1.0
8	-1.0	-2.0	15.9	12.2	11.2	11.4	11.0	10.9	08.8	06.0	04.4	04.0	07.1	-2.0	-1.0
9	-1.0	-2.0	15.5	11.8	10.8	09.8	08.8	07.8	06.8	05.8	05.4	05.0	08.1	-2.0	-1.0
10	-1.0	-2.0	15.9	12.2	11.2	10.2	09.2	08.2	07.2	06.8	06.4	06.0	09.1	-2.0	-1.0
11	-1.0	-2.0	16.3	12.6	11.6	11.8	11.4	11.3	10.0	09.0	07.4	07.0	10.1	-2.0	-1.0
12	-1.0	-2.0	16.7	13.0	12.6	14.6	-2.0	-2.0	-2.0	10.6	08.4	08.0	11.1	-2.0	-1.0
13	-1.0	-2.0	17.1	16.7	16.3	16.7	-2.0	-1.0	-2.0	12.5	12.1	11.7	12.1	-2.0	-1.0
14	-1.0	-2.0	-2.0	-2.0	-2.0	-2.0	-2.0	-1.0	-2.0	-2.0	-2.0	-2.0	-2.0	-2.0	-1.0
15	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0

Figura 3.9: Mapa de costos calculado usando el algoritmo de programación dinámica.

cia una de las 8 celdas vecinas, ya que al hacerlo se está reduciendo la capacidad de giro del robot a intervalos de 45°. Esto se aprecia mejor en la trayectoria encontrada en el mapa de celdas simple (ver Figura 3.11).

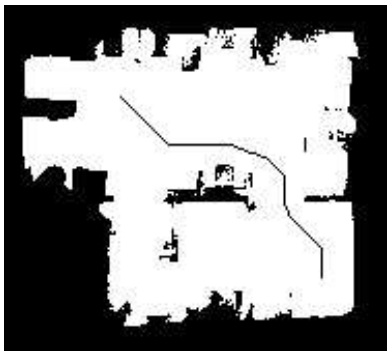


Figura 3.10: Trayectoria obtenida del algoritmo de programación dinámica.

Para mejorar la trayectoria obtenida se realiza un post-procesamiento, el cual es descrito a continuación.

3.2.4. Refinamiento de la trayectoria

Para hacer un refinamiento de la trayectoria el primer paso consiste en abandonar el ámbito de las celdas y entrar en el ámbito del espacio continuo. Para ello, la

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	1							1							1
3	1							1							1
4	1			+				1				+			1
5	1				+			1				+			1
6	1				+			1				+			1
7	1				+							+			1
8	1				+						+				1
9	1					+	+	+	+	+					1
10	1														1
11	1														1
12	1														1
13	1							1							1
14	1							1							1
15	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figura 3.11: Trayectoria inicial obtenida directamente del mapa de costos.

trayectoria obtenida, que es una secuencia de celdas por las que el robot debe pasar, se transforma en un conjunto de puntos de verificación de la siguiente manera.

Un punto de verificación es un punto en el espacio continuo donde el robot necesita girar para cambiar de orientación y seguir con la ruta establecida. Estos puntos de verificación tienen lugar en las celdas donde existe un cambio en la dirección del movimiento. Si se asume que el robot se encuentra en la posición $(0,0)$ del espacio continuo, el primer punto de verificación c_1 tendrá las coordenadas $(0,0)$ el segundo punto de verificación tendrá las coordenadas correspondientes a la siguiente celda donde exista un cambio en la orientación del robot y así sucesivamente. De esta manera se construye el conjunto $C = (c_1, c_2, ..., c_n)$ que contiene los n puntos de verificación por los que el robot debe pasar para llegar a la meta.

Una vez que han sido determinados los puntos de verificación, se revisa, para cada punto de verificación $c_i \in C$ si es posible llegar en línea recta a alguno de los puntos de verificación en el subconjunto posterior $C_i = (c_{i+2}, c_{i+3}, ..., c_n)$. Es decir, se verifica si existe una trayectoria en línea recta que una al punto de verificación c_i con algún punto de verificación posterior $c_j \in C_i$ y que además esa línea recta no viole la distancia deseada de los obstáculos (D_{pref}). Si existe más de un punto de verificación en el conjunto C_i que cumpla con la condición, se elije al punto $c_j \in C_i$ más lejano de c_i. Todos los puntos de verificación entre c_i y c_j son eliminados de C y se repite el procedimiento. En el Algoritmo 3.3 se muestra el pseudo-código del refinamiento de la trayectoria.

Algoritmo 3.3 Refinamiento de los puntos de verificación.

Sea $C = (c_1, c_2, ..., c_n)$ el conjunto inicial de puntos de verificación.
Sea $C_i = (c_{i+2}, c_{i+3}, ..., c_n)$ el conjunto posterior asociado a cada $c_i \in C$.

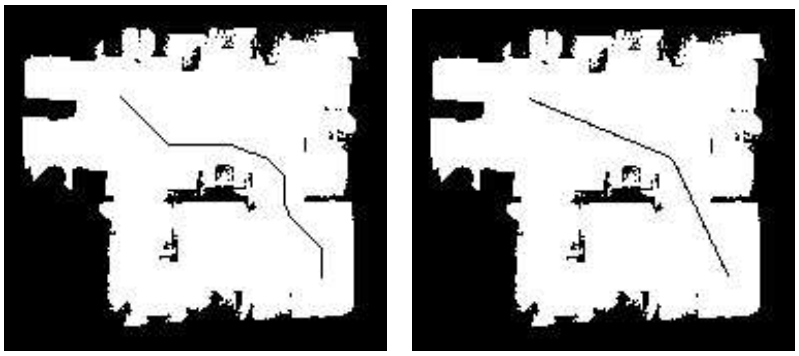
```
procedimiento refinamiento() {
  i = 1;repetir {
    Si existe un punto de verificación c_j ∈ C_i
      tal que se puede unir con una línea recta a c_i con c_j,
      respetando la distancia deseada D_pref
    entonces eliminar los puntos (c_{i+1}, c_{i+2}, ..., c_{j-1}) de C
    i = i + 1
  } mientras i < (n-1)
}
```

En muchos casos, como el que se muestra en la Figura 3.12 la mejora es considerable. De la misma forma en la Figura 3.13 se muestran los puntos de verificación de la trayectoria final para el mapa simple de celdas. Esto se debe a que la limitante en el ángulo de giro impuesta por el esquema de vecindad, ocasiona que sean generadas trayectorias con giros innecesarios. El conjunto de puntos de verificación final será utilizado por el control de navegación para determinar los movimientos que el robot necesita realizar para cumplir con la trayectoria generada. Esto se abordará en el Capítulo 6.

3.3. Comentarios finales

En este capítulo es presentada una propuesta de planificación de trayectorias para un robot móvil en ambientes interiores. Esta propuesta toma en cuenta explícitamente la incertidumbre asociada con los sensores y actuadores del robot; para lograrlo realiza la planificación basándose en tres objetivos: moverse a una distancia prudente de los obstáculos, moverse hacia la meta y minimizar el número de giros requeridos. Las contribuciones realizadas en esta parte son las siguientes.

- Se introduce un algoritmo para la construcción de un mapa de cercanía a obstáculos. Dependiendo de la exactitud de los sensores utilizados es importante que el



(a) Trayectoria inicial considerando programación dinámica.

(b) Trayectoria después del proceso de refinación.

Figura 3.12: Ejemplo del cálculo de una trayectoria.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	1							1							1
3	1							1							1
4	1			●				1							1
5	1			●				1							1
6	1				●			1							1
7	1				●										1
8	1					●									1
9	1					●	●	●	●	●	●	●			1
10	1														1
11	1														1
12	1														1
13	1							1							1
14	1							1							1
15	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figura 3.13: Trayectoria final obtenida de la refinación de los puntos de verificación para el mapa simple.

robot construya trayectorias que se mantengan alejadas de los obstáculos para evitar colisiones accidentales. El concepto de cercanía de los obstáculos ya ha sido abordado anteriormente [Romero, 2001, Batavia and Nourbakhsh, 2000], sin embargo, sólo se considera información de las regiones cercanas a los obstáculos. En cambio en el mapa de cercanía propuesto en este trabajo proporciona información adicional, no sólo en las regiones cercanas a los obstáculos, sino sobre el espacio en general. Este mapa podría ser usado para otras tareas; por ejemplo, si el robot está en una habitación y necesita moverse a la zona más alejada de los obstáculos dentro de la misma, bastaría con moverse hacia las celda con el valor máximo dentro del mapa de cercanía que corresponda a la habitación.

- Se introduce un proceso de refinamiento de la trayectoria con la finalidad de reducir la cantidad de giros necesarios y de compensar las limitaciones impuestas por el esquema de vecindad utilizado. Esta técnica es completamente heurística, sin embargo en la gran mayoría de los casos la mejora a la trayectoria obtenida del algoritmo de programación dinámica es considerable.

Del esquema general de navegación mostrado en la Figura 1.3 de la Sección 1.6, este capítulo aborda el segundo paso: planificación de trayectorias. Como puede apreciarse en el esquema, la siguiente tarea consiste en realizar un seguimiento de la posición mientras el robot se mueve; para hacerlo el robot utiliza la información que percibe por medio de sus sensores. En el siguiente capítulo se describe de que manera se procesa la información sensorial para ubicar puntos de referencia en el ambiente, e inmediatamente después en el Capítulo 5 se describe el algoritmo de seguimiento de la posición en base a los puntos de referencia detectados.

Capítulo 4

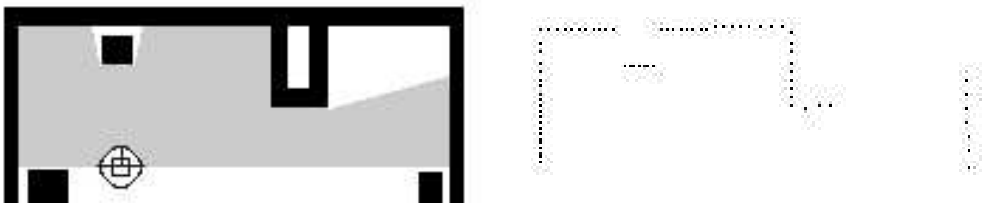
Información sensorial

El robot móvil percibe el mundo que lo rodea a través de sus sensores y la naturaleza de esta información varía en función del tipo de sensor con el que se cuenta. En el caso de este trabajo se cuenta con un sensor láser que proporciona una gran cantidad de datos. En principio, es necesario decidir si se usan todos los datos o si se utiliza solamente una parte de ellos.

En este capítulo se presentan los métodos que le permiten al robot procesar la información que percibe del ambiente, de acuerdo con las necesidades de las tareas que debe realizar y los sensores con los que cuenta.

4.1. Características del sensor

En el caso de este trabajo se tiene un sensor láser tipo SICK LMS 200, el cual proporciona la distancia a los objetos más cercanos sobre un plano paralelo al piso, en un rango de 180 grados frente al robot. La precisión angular es de hasta 0.25 grados por lo que pueden obtenerse hasta 720 lecturas por barrido, donde cada una indica la distancia y el ángulo al obstáculo más próximo en esa dirección. En la Figura 4.1 se muestra un ejemplo de los datos que se obtienen de este tipo de sensor.



(a) Ejemplo de un robot en un ambiente. (b) Lecturas devueltas por un sensor láser.

Figura 4.1: Ejemplo de los datos obtenidos de un sensor láser

4.2. Uso de la información sensorial

Básicamente existen dos enfoques que se refieren al uso de la información sensorial obtenida por el robot: el enfoque basado en marcas y el enfoque basado en información completa [Dudek and Jenkin, 2000]. La decisión gira en torno a la cantidad de datos utilizados, mientras los enfoques basados en información completa utilizan toda la información proporcionada por los sensores, los basados en marcas utilizan un subconjunto de estos datos.

Cuando el robot necesita hacer una comparación entre lo que percibe desde dos lugares diferentes del ambiente, los enfoques basados en información completa emplean técnicas para medir la similitud basadas en correlación como: mínimos cuadrados, ajuste de curvas o máxima verosimilitud [Dudek and Jenkin, 2000]. La efectividad de los métodos varía en cada caso, dependiendo de las características del ambiente y el sensor. La ventaja es que suelen ser precisos debido a que encuentran el mejor ajuste. Sin embargo, la desventaja que presentan es que debido a la gran cantidad de datos que se tienen y a que hacen una comparación de bajo nivel, estos métodos requieren de un tiempo de procesamiento considerable. Por otro lado, es común en sensores reales que existan variaciones abruptas en las medidas; estas variaciones son independientes del desplazamiento y se deben a rebotes en superficies muy lisas, o como en el caso de los sensores láser a superficies negras o plateadas [Romero, 2001].

Los enfoques basados en marcas, se fundamentan en la idea de ubicar puntos o regiones específicas dentro de los datos, que sean representativos del conjunto completo. Para ello se deben definir métodos que permitan identificar estas marcas, con la condición de que puedan ser identificadas desde diferentes orientaciones y distancias.

En este caso, la comparación entre lo que el robot percibe desde dos lugares diferentes se hace en función solamente de las marcas. Para ello se utilizan métodos geométricos como triangulación, o se utiliza la información de su distribución en el espacio para encontrar correspondencias. La ventaja de este enfoque es que los métodos realizan la comparación rápidamente debido a la cantidad de reducida de datos. La desventaja es que, para obtener buenos resultados, las marcas deben ser identificadas rápidamente y con precisión.

En el caso de un robot móvil, el tiempo de respuesta es un aspecto importante. Actualmente se han desarrollado una gran variedad de trabajos con robots móviles, en los que estos interactúan con personas llevando mensajes u objetos. Para la realización de dichas tareas se requiere que el robot se mueva lo más rápido posible, por lo que todos los procesos realizados mientras el robot se mueve deben tomar en cuenta esta necesidad.

En esta tesis se propone utilizar el enfoque basado en marcas naturales del ambiente, las cuales funcionan como puntos de referencia para el robot. Es por esto que surge la necesidad de contar con métodos para identificar las marcas dentro del conjunto de las lecturas de los sensores del robot. El uso de este enfoque, pretende reducir significativamente el conjunto de datos que se utilizarán para el seguimiento de la posición en el Capítulo 5. Enseguida se mencionan varios tipos de marcas naturales y se especifica de que manera se extraen de las lecturas del sensor láser.

4.3. Marcas naturales

Existen diversos tipos de marcas naturales, muchas de ellas han sido usadas en varios sistemas de seguimiento de la posición. Entre las más comunes tenemos las esquinas cóncavas y convexas, las paredes y las obtenidas por medio de visión como bordes o regiones de cierto color o intensidad. La elección del tipo de marca a utilizar depende directamente del tipo de sensores con los que se cuente. Por otro lado, también hay que considerar que uno de los principales problemas de los enfoques basados en marcas es que dependen en gran medida de la detección precisa de un buen número de marcas. Por ejemplo, en [Xiaowei et al., 2000], el autor propone la utilización de un solo tipo de marca: las discontinuidades. Al final, el autor concluye que la mayor debilidad de su propuesta es que, en ausencia de discontinuidades, el sistema de seguimiento de la posición falla. Para evitar estas restricciones, es común el uso de dos o más

tipos de marcas, de tal manera que el robot siempre posea información suficiente para mantenerse localizado. Es necesario también, que el robot pueda identificar las marcas desde diferentes distancias y orientaciones, además de contar con elementos suficientes para diferenciar a una marca de otra.

En este trabajo se usarán tres tipos de marcas: las dos primeras son las discontinuidades y las esquinas, esto obedece a que son muy comunes en ambientes reales de oficina donde existen muchas oclusiones. De manera adicional y con la finalidad de hacer más robusto el sistema de seguimiento, se usarán también las paredes como marcas naturales. La intención de este trabajo es combinar tres tipos de marcas para que, en ausencia de esquinas, las paredes sean los puntos de referencia y viceversa. Además, se identificará una serie de atributos distintivos para cada una de las marcas, con la final de poder diferenciarlas entre las del mismo tipo. El procedimiento de identificación de marcas naturales se explica en detalle a continuación.

4.4. Identificación de marcas naturales

Una de las principales desventajas de algunos sistemas de identificación de marcas naturales es que realizan un procesamiento independiente de los datos para identificar cada tipo de marca. Esto ocasiona que los datos sean procesados tantas veces como tipos de marcas se pretende extraer. Para evitar dicho inconveniente a continuación se presenta un esquema de identificación de marcas en tres niveles. El esquema que a continuación se describe se basa en el trabajo de [Xavier et al., 2004], aunque difiere en la técnica empleada para la detección de paredes.

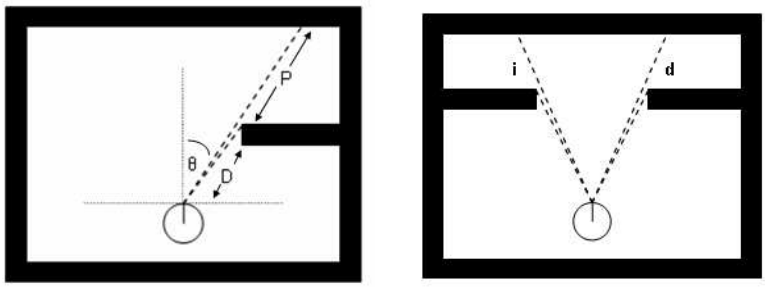
El propósito es aprovechar las relaciones existentes entre tipos de marcas de tal manera que el conjunto total de lecturas sea procesado una sola vez. La idea básica consiste en partir de la identificación de las marcas más sencillas hacia las más complejas. Partiendo del conjunto total de lecturas del sensor láser, el primer paso consiste en identificar las discontinuidades ya que al mismo tiempo, esta identificación permite dividir el conjunto total de lecturas en segmentos. En cada segmento se identifican las esquinas, las cuales a su vez permiten dividir nuevamente al segmento. Una vez que se han detectado todas las esquinas, se verifica cuales de los segmentos resultantes corresponden a una pared aplicando una transformada de Hough local a cada segmento.

Este procedimiento requiere de técnicas específicas de identificación de cada

tipo de marca. A continuación se presenta el detalle de la identificación de cada uno de estos tipos.

4.4.1. Discontinuidades

Una discontinuidad se define como una variación mayor a cierto umbral en la distancia medida en dos lecturas consecutivas del láser (ver Figura 4.2). Cada discontinuidad tiene en principio 2 atributos que la caracterizan: la distancia entre la discontinuidad y el robot, y la orientación a la que se encuentra con respecto al robot. Adicionalmente, se obtienen dos atributos más: la profundidad y el tipo de la discontinuidad el cual está determinado por la ubicación de la región no visible, por ejemplo, en la Figura 4.2(b), *i* es una discontinuidad izquierda y *d* una discontinuidad derecha.



(a) (b)

Figura 4.2: Identificación y caracterización de discontinuidades

Como se mencionó y para efectos de diferenciación, cada discontinuidad es caracterizada con un conjunto de atributos los cuales se describen a continuación:

- **Distancia (D):** Es la distancia en metros entre el robot y la discontinuidad.
- **Orientación (θ):** Es el ángulo en el que se encuentra la discontinuidad con respecto a la orientación del robot.
- **Profundidad (P):** Es la diferencia entre las dos lecturas que dieron origen a la discontinuidad.

- **Tipo de discontinuidad (T):** Determina si una discontinuidad es izquierda (*i*) o derecha (*d*) de acuerdo con la ubicación de la región no visible (ver Figura 4.2(b)).

Para la identificación del conjunto de discontinuidades que el robot observa, basta con recorrer una sola vez las lecturas del láser. Mientras se hace el recorrido, se obtiene la magnitud de la diferencia aritmética entre cada par de lecturas consecutivas del láser, si es mayor a un cierto umbral μ_d, se ha encontrado una discontinuidad. En la práctica, el valor de μ_d es la menor diferencia entre lecturas consecutivas del láser que será considerada como una discontinuidad.

Si tomamos en cuenta las consideraciones anteriores, podemos esperar obtener un buen número de discontinuidades de los datos del sensor láser en ambientes como oficinas donde existen una gran cantidad de objetos. Como se mencionó anteriormente, estas discontinuidades sirven también para determinar el conjunto inicial de segmentos. En la Figura 4.3 puede verse un ejemplo de la identificación de discontinuidades y la división inicial en segmentos.

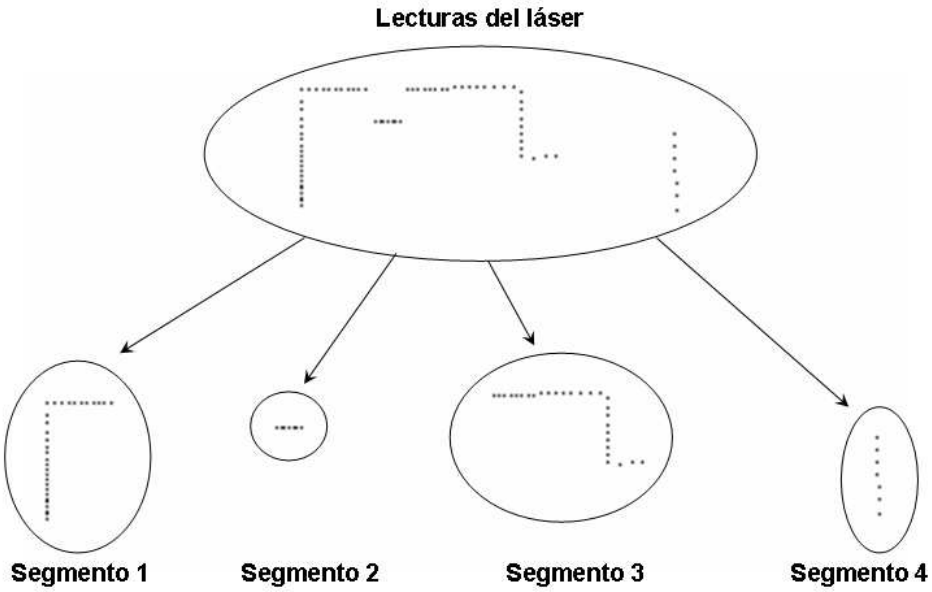


Figura 4.3: Identificación de discontinuidades y primera segmentación.

Las discontinuidades extraídas corresponden a oclusiones en el ambiente, y son

ocasionadas por objetos que se encuentran en él. Esto permite que puedan ser identificadas como un solo punto desde diferentes perspectivas y a diferentes distancias del robot. La precisión de la detección depende de la precisión del sensor, su hoja técnica reporta un error de $\pm$ 15 mm [SICK-LMS200, 2003]. en la distancia, lo cual sería el error en el atributo D, y una resolución máxima de 0.25 grados lo cual correspondería al error en el atributo θ.

Una vez que se han identificado las discontinuidades y en consecuencia los segmentos iniciales, se procede a la identificación de esquinas, procedimiento que se describe a continuación.

4.4.2. Esquinas

La detección de esquinas es muy simple: Sea $P = (p_1, p_2, ..., p_n)$ el conjunto de puntos de un segmento. Se traza una línea recta imaginaria entre el primero (p_1) y el último (p_n) punto del segmento , enseguida se busca al punto p_i más alejado de dicha línea y si la distancia entre este punto y la recta es mayor a cierto umbral μ_e entonces p_i es una esquina y el segmento se parte en dos [Xavier et al., 2004]. En la práctica μ_e puede interpretarse como la máxima variación que puede permitirse para considerar a un conjunto de puntos como candidato a una recta. En la Figura 4.4 puede observarse este procedimiento para el segmento 3 de la Figura 4.3.

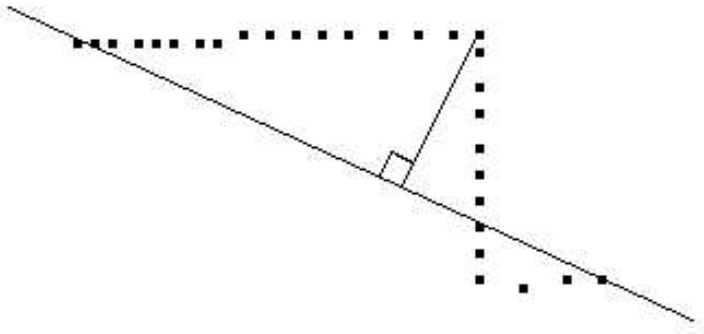


Figura 4.4: Ejemplo de la identificación de una esquina dentro de un segmento.

El mismo procedimiento se realiza para cada uno de los segmentos generados.

El proceso termina cuando ningún segmento puede ser dividido. El objetivo es que al final solamente queden segmentos que sean posibles líneas rectas como paredes, mesas, o cualquier superficie que genere una recta. En la Figura 4.5 puede apreciarse el árbol de segmentos que resulta del procedimiento antes descrito. Las hojas son los segmentos que serán considerados como candidatos a paredes dependiendo de su tamaño y número de puntos.

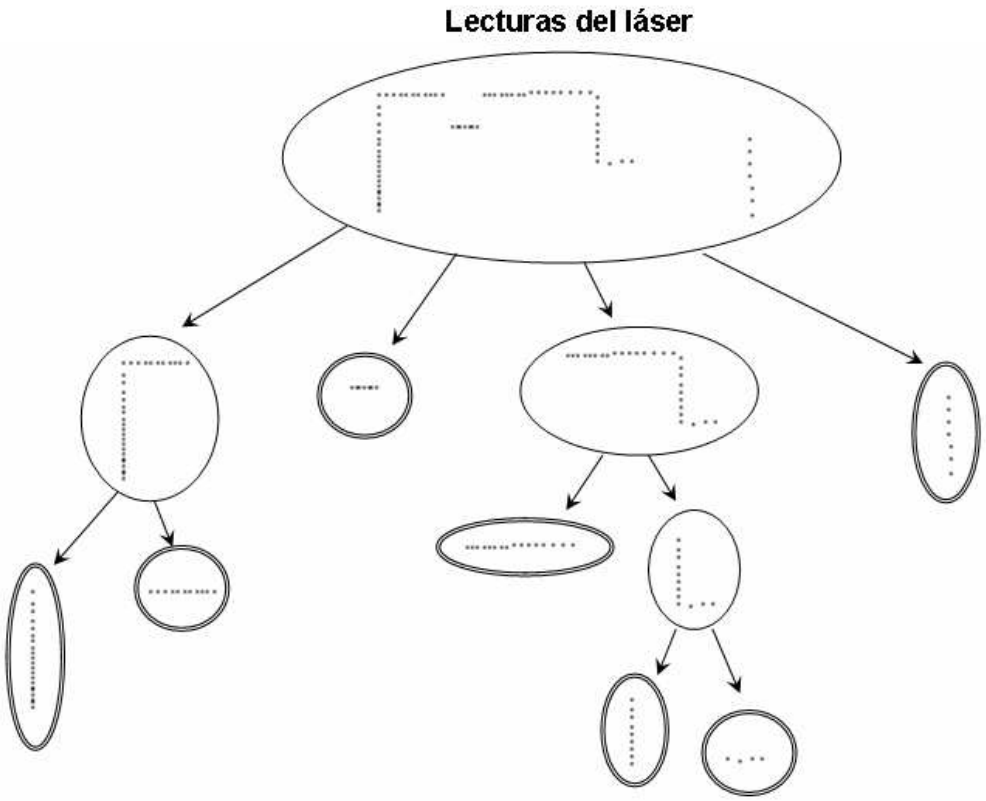


Figura 4.5: Árbol donde se observa la segmentación progresiva de las lecturas del láser.

Al final del proceso se descartan todos aquellos segmentos cuya distancia entre el punto inicial y el punto final no sea mayor a un umbral μ_p, donde μ_p es la mínima longitud admisible para que un segmento sea considerado candidato a pared. Esta última consideración permite descartar inmediatamente segmentos muy pequeños o con muy pocos puntos, los cuales podrían deberse al ruido ocasionado por alguna superficie u objeto.

4.4.3. Paredes

El tercer tipo de marcas naturales usadas en este trabajo de tesis son las paredes. Dado que las lecturas del láser pueden verse como un plano paralelo al piso a la altura del sensor láser, la detección de paredes se reduce a la detección de líneas dentro del conjunto de datos que devuelve el sensor láser.

Para la identificación de estas líneas se utiliza la transformada de Hough [Duda and Hart, 1972], la cual es una técnica que se emplea para identificar una gran variedad de formas y que ha sido ampliamente usada en el dominio del procesamiento digital de imágenes. Esta transformación es comúnmente usada para la identificación de curvas regulares, como líneas, círculos, elipses y cualquier curva que pueda expresarse en forma parametrizada. La principal ventaja de esta transformada es la tolerancia a las irregularidades en los datos, además se ve poco afectada por ruido en los mismos. Esta última propiedad es muy importante ya que, aunque la precisión del sensor láser utilizado es buena, no se observan líneas rectas perfectas en las lecturas. Por otro lado aunque el sensor fuera perfecto las paredes regularmente tienen objetos colgados, pegados o cercanos a ellas, que no permiten obtener una línea recta, las paredes de hecho no son perfectamente rectas en si mismas.

Para el caso que nos ocupa se aplica una transformada de Hough local en los segmentos considerados como candidatos a paredes de acuerdo con el parámetro μ_p. Una de las ventajas de la división en segmentos que se realiza es que solo puede haber, a lo más, una recta en cada segmento. Esta consideración es de gran ayuda al aplicar la transformada de Hough local ya que de no ser así tendrían que emplearse técnicas de *agrupamiento* para encontrar los máximos locales en el acumulador que correspondan con rectas. Como se sabe de antemano que solo puede haber una recta, basta con tomar el máximo del acumulador en cada caso.

Comúnmente se considera a la transformada de Hough un procedimiento muy lento, sin embargo gran parte de esta lentitud se debe al proceso de *agrupamiento* que debe realizarse cuando existe la posibilidad de encontrar más de una línea. Otro factor que hace lento el procedimiento es que comúnmente se aplica a imágenes digitales binarizadas, donde la cantidad de pixeles que se procesan es en ocasiones muy alta. En cambio, en el caso de un conjunto de lecturas láser los puntos que deben procesarse son solamente dichas lecturas. Si a esto se añade que el rango posible en distancia está limitado de 0 a λ, donde λ es el alcance del sensor (en este caso 8m.) entonces el proceso se agiliza y puede emplearse de manera efectiva.

Transformada local de Hough

Sea L el conjunto total de lecturas del láser. Dado que en nuestro caso la posición de cada uno de los puntos está definida en coordenadas polares (ρ_i, γ_i) la ecuación de las rectas se define de acuerdo con la Ecuación 4.1.

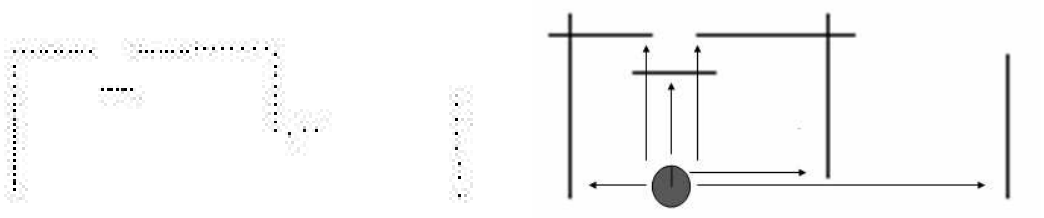
$$\rho_r = \rho_i cos(\gamma_i - \gamma_r) \tag{4.1}$$

Donde (ρ_r, γ_r) es el vector normal que define a la recta y (ρ_i, γ_i) las coordenadas de cada uno de los puntos en L, cuyos valores se obtienen directamente del láser.

Para aplicar la transformada de Hough se divide el margen de variación de ρ_r $(-90^\circ <= \rho_r <= 90^\circ)$ en M partes. En nuestro caso $M = 720$ por lo que los posibles valores de ρ_r están separados por intervalos de 0.25°. De la misma forma se divide el margen de variación de γ_r, que es de $0 <= \gamma_r <= 8m.$, en P intervalos. En nuestro caso $P = 800$, por lo que los posibles valores de γ_r están separados por intervalos de $1cm$. Sea γ_{rk} el k-ésimo valor posible de γ_r con $k = 1, 2, ..., M$. Para cada γ_{rk} y cada punto (ρ_i, γ_i) se obtienen, a partir de la Ecuación 4.1 los diferentes valores de ρ_r, que denotaremos por ρ_{rj}.

Se define un acumulador para cada uno de los posibles pares (ρ_{rj}, γ_{rk}) y se inician a 0. Estos acumuladores se identifican por $A(j,k)$. Para cada ρ_{rk} y cada punto (ρ_i, γ_i) se incrementa en una unidad el acumulador correspondiente, esto es: $A(j,k) = A(j,k) + 1$. Cuando finaliza el proceso, el acumulador $A(j,k)$ que tenga el valor máximo corresponderá a la recta normal a la superficie encontrada cuyos parámetros son: (ρ_{rj}, γ_{rk}). Los cuales a su vez no son más que los atributos D y θ de las marca correspondiente.

Usando la transformada local de Hough se obtiene la distancia y orientación de la recta normal a cada recta identificada; esto es: los parámetros D y θ, además para una mejor diferenciación con el resto de las superficies, se almacena el tamaño de la pared (en metros) en el atributo P, el cual no se determina por Hough sino con los puntos en los extremos del segmento. En la Figura 4.6(b) pueden observarse las paredes detectadas; las flechas representan las rectas normales a las paredes, para las cuales son determinados los atributos D y θ.



(a) Lecturas del sensor láser

(b) Rectas normales a las paredes identificadas

Figura 4.6: Identificación de paredes usando la transformada de Hough.

4.5. Conjunto final de marcas

Como resultado de aplicar el procedimiento descrito se obtiene una lista con el conjunto de atributos que corresponden a cada tipo de marca. En ella se muestra que los atributos D y θ se refieren a la ubicación de la marca con respecto al robot. Los atributos P y T son características distintivas que se refieren al tipo de marca y al tamaño de la misma. Los umbrales se definieron de la siguiente manera: $\mu_d = 20cm.$, $\mu_e = 15cm.$ y $\mu_p = 30cm.$.

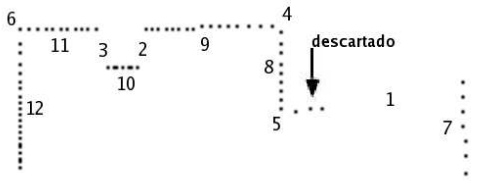
En la Figura 4.7(a) se muestra el conjunto de marcas extraídas de las lecturas del robot. En la Figura 4.7(b) se muestran las lecturas del robot con las marcas etiquetadas de acuerdo con el número consecutivo que se observa en la Figura 4.7(a). Cabe mencionar que los segmentos con pocos puntos (menos de 5) no son considerados como candidatos a paredes ya que no habría información suficiente para determinar sus atributos. Esta es la razón por la que el segmento que se encuentra señalado como descartado a la derecha de la marca número 5, en la Figura 4.7(b) no fue identificado como candidato a pared.

4.6. Comentarios finales

En este capítulo se presentó un método para la extracción de marcas naturales del ambiente. Las marcas naturales elegidas son discontinuidades, esquinas y paredes

No	D	θ	P	T	Tipo de marca
1	2.18	-73.0	1.41	i	discontinuidad
2	1.10	-6.0	0.40	i	discontinuidad
3	1.11	10.0	0.41	d	discontinuidad
4	2.17	-47.0	-	e	esquina
5	1.71	-68.0	-	e	esquina
6	1.84	36.0	-	e	esquina
7	3.46	-89.50	1.06	p	pared
8	1.59	-89.75	0.85	p	pared
9	1.49	0.0	1.36	p	pared
10	1.10	0.0	0.39	p	pared
11	1.50	0.0	0.80	p	pared
12	1.10	89.75	1.43	p	pared

(a) Lista de marcas naturales identificadas.



(b) Lecturas devueltas por el sensor láser.

Figura 4.7: Identificación de marcas naturales del ambiente.

debido a que se complementan unas a otras. En ambientes muy estructurados o creados artificialmente es frecuente encontrarse con varias superficies planas como paredes mientras que en ambientes reales tipo oficina es común encontrar un gran número de discontinuidades y esquinas. En este trabajo se propone su uso combinado para hacer el seguimiento de la posición del robot mientras se mueve. Sin importar si se identifican marcas de los tres tipos o solo de uno, el robot las utilizará como puntos de referencia para hacer el seguimiento de la posición.

Otra de las ventajas que proporciona el procedimiento descrito es que separa el conjunto total de lecturas del láser en segmentos e identifica cuales de ellos son candidatos a paredes. Esto se realiza con la finalidad de hacer más eficiente la transformada de Hough convirtiéndola en local. De esta manera se aplica varias veces una transformada de Hough local en lugar de hacer una sola vez la transformada de Hough sobre el total de lecturas del láser.

En un sentido más amplio, las discontinuidades, esquinas y paredes pueden ser utilizadas de manera combinada debido a que las tres cuentan con los atributos D (distancia) y θ (orientación). Sólo hay que tener en cuenta que las paredes requieren un trato especial ya que no son puntos en el espacio sino líneas rectas y sus atributos D y θ se refieren a la recta normal a la pared. Esta distinción se puntualizará en el siguiente capítulo. El resto de los parámetros pueden ser utilizados en la tarea de establecer correspondencias o en la de diferenciar a una marca en particular de otras de su mismo

tipo.

De aquí en adelante se hará referencia a las discontinuidades, esquinas y paredes simplemente como *marcas* ó *marcas naturales* estableciendo la diferencia explícitamente sólo en los casos que lo ameriten. En el capítulo siguiente se muestra de que manera pueden usarse las marcas naturales detectadas en el problema del seguimiento de la posición del robot.

Capítulo 5

Seguimiento de la Posición

En este capítulo se presenta el algoritmo de seguimiento de la posición propuesto. Como se mencionó en el primer capítulo, el seguimiento de la posición es necesario debido a que el odómetro, que es el sensor encargado de proporcionar la información del movimiento, no es suficientemente preciso. En consecuencia, el robot tiene que valerse de la información que percibe a través del resto de sus sensores para determinar con mayor exactitud su ubicación dentro del ambiente.

La idea general para realizar el seguimiento de la posición, consiste en la utilización de marcas naturales como puntos de referencia. Se pretende que el robot se mueva y realice el seguimiento de la posición sin detenerse. Se espera también, que el mecanismo empleado sea lo más simple y eficiente posible debido a que el procesamiento debe realizarse muchas veces mientras el robot se mueve, a intervalos regulares de tiempo.

5.1. Esquema general

El planteamiento consiste en la extracción del conjunto de marcas en el tiempo t desde la ubicación A, En la Figura 4.7(a) se muestra el conjunto de marcas extraídas de las lecturas del robot. En la Figura 4.7(b) se muestran las lecturas del robot con las marcas etiquetadas de acuerdo con el número consecutivo que se observa en la Figura 4.7(a). posteriormente el robot se mueve y después de un cierto intervalo de

tiempo se vuelven a extraer las marcas en el tiempo $t + \Delta t$ desde la posición *B*. Se encuentra la correspondencia entre marcas que estén presentes en ambos conjuntos y en base a ellas, se realiza el cálculo del desplazamiento y la orientación del robot con respecto al instante t, por medio de triangulación. En la Figura 5.1 se muestra el esquema general de este proceso. Durante el resto del capítulo se abordarán cada uno de los pasos mencionados, comenzando con el procedimiento para establecer la correspondencia entre conjuntos de marcas. El control de navegación será descrito en el Capítulo 6.

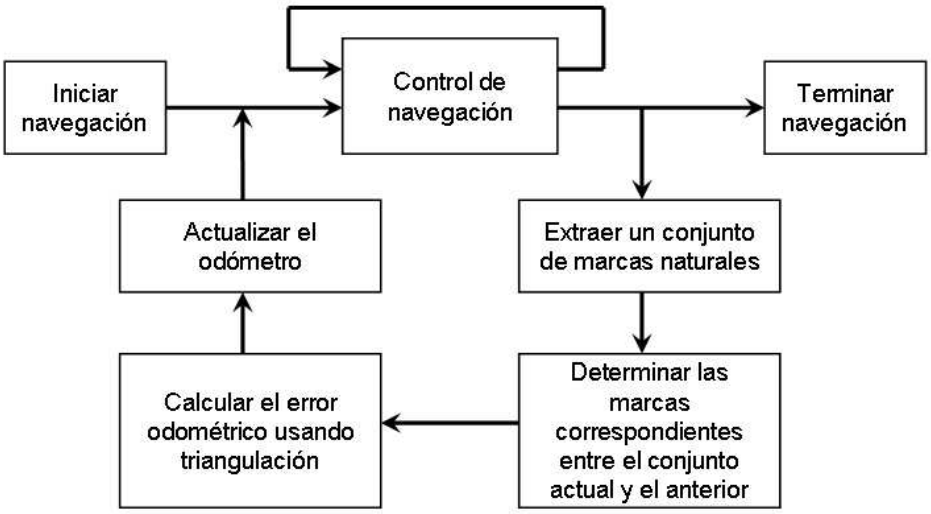


Figura 5.1: Esquema general del proceso de seguimiento de la posición.

5.2. Correspondencia entre conjuntos de marcas

Aún cuando las ubicaciones *A* y *B* son cercanas una de la otra, el establecimiento de la correspondencia entre conjuntos de marcas no es un problema sencillo. La dificultad radica en que al identificar dos conjuntos de marcas desde dos lugares diferentes, podrían haber marcas que ya no están presentes por haber quedado ocultas tras algún objeto o podrían haber también nuevas marcas provenientes de regiones que no eran visibles desde la ubicación anterior.

Además hay que tener en consideración que si se establece una correspondencia

equivocada, esto afectaría fuertemente la creencia que el robot tiene acerca de su ubicación. En este caso, el problema puede ser visto como la búsqueda de correspondencias entre dos conjuntos de puntos, aunque hay que considerar también la información adicional que se tiene.

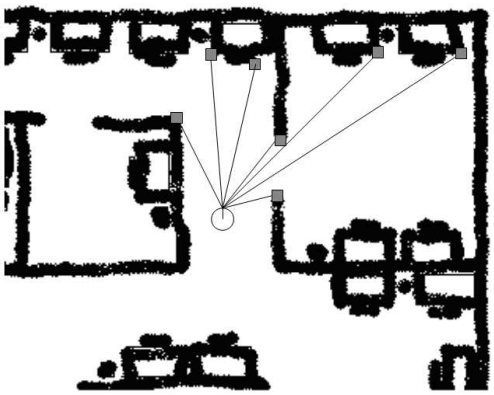
En principio se conocen los atributos de cada una de las marcas de ambos conjuntos y se tiene una estimación de la distancia recorrida proporcionada por el odómetro, que si bien puede no ser precisa, si es aproximada. El proceso de establecimiento de correspondencias se realiza en dos pasos: en el primer paso, se hace una asignación inicial basada en la estimación del odómetro, en el segundo paso se revisan las asignaciones iniciales verificando que sean consistentes geométricamente. En la Figura 5.2(a) los cuadrados son las marcas detectadas por el robot en el tiempo t, en la Figura 5.2(b) los círculos son las marcas detectadas en el tiempo $t + \Delta t$.

5.2.1. Correspondencia inicial entre conjuntos de marcas

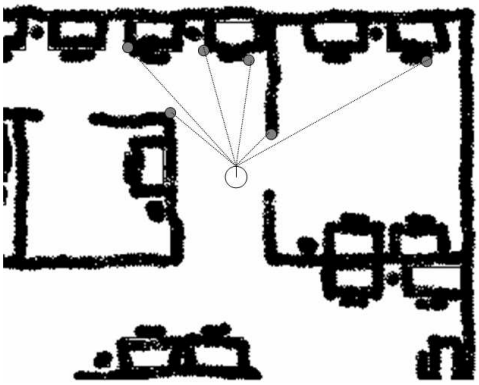
Sean M_t el conjunto de marcas detectadas en el tiempo t y $M_{t+\Delta t}$ el conjunto de marcas detectadas en el tiempo $t + \Delta t$. Para hacer el cálculo inicial de la correspondencia entre marcas de ambos conjuntos, se utiliza la estimación del odómetro. En base a ella se calcula la posición (x_i, y_i) en donde se esperaba encontrar cada marca $m_i \in M_t$. Para cada marca m_i, se busca la marca más cercana $m_k \in M_{t+\Delta t}$ y se asocian, formando así la relación $rel(m_i, m_k)$.

Cabe aclarar que la cercanía se define como la distancia Euclidiana y que solamente se busca dentro de una región delimitada por un círculo alrededor de la posición esperada (x_i, y_i). El radio de este círculo está relacionado con la precisión del odómetro; si el odómetro fuera perfecto, la marca m_k se encontraría justo en el lugar donde se espera encontrarla, como sabemos que no lo es, se busca en una región alrededor de la posición esperada. De esta manera el radio del círculo de búsqueda es el máximo error que puede arrojar la estimación del odómetro. Además de ser la más cercana, la marca m_k correspondiente, debe ser del mismo tipo (pared, esquina o discontinuidad) que m_i.

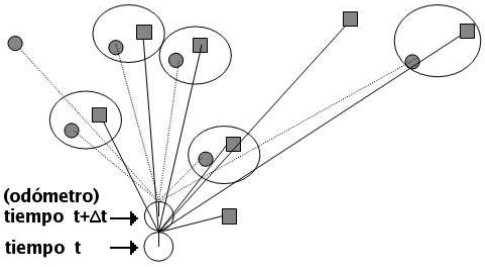
En la Figura 5.2(c) se ilustra este procedimiento: los cuadrados representan las marcas en el tiempo t mientras que los círculos pequeños representan las marcas en el tiempo $t + \Delta t$; los círculos que se observan representan el área donde se busca la correspondencia más cercana.



(a) Marcas en el tiempo t.



(b) Marcas en el tiempo $t + \Delta t$.



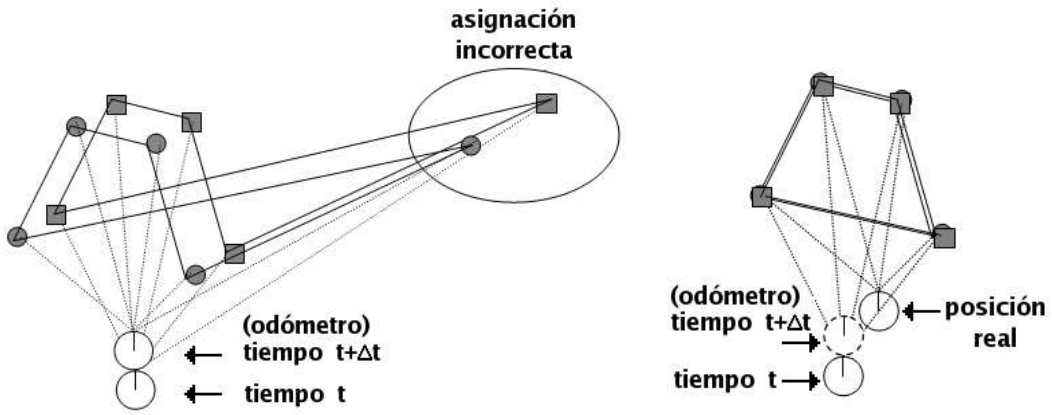
(c) Asignación inicial de las marcas correspondientes en ambos conjuntos.

Figura 5.2: Relación inicial de correspondencia entre conjuntos de marcas.

5.2.2. Revisión de la consistencia geométrica entre conjuntos de marcas

Una vez que se hace la asignación inicial, el siguiente paso es verificar que dichas asignaciones sean válidas. Sea d_{ij} la distancia Euclidiana entre las marcas m_i y m_j. Se verifica que si existen las relaciones $rel(m_i, m_k)$ y $rel(m_j, m_l)$ entonces se debe cumplir que $d_{ij} \approx d_{kl}$. Dicho de otra manera, la distancia entre pares de marcas debe persistir en ambos conjuntos de marcas.

Nótese que la relación de igualdad estricta no es posible debido a que las marcas no se detectan con precisión, para efectos prácticos se utiliza un umbral determinado por la precisión del sensor, de ahí la notación. Usando este criterio se descartan las asignaciones que no cumplan con la condición establecida. En la Figura 5.3(a) se observa el efecto del procedimiento descrito. En la imagen puede verse que la correspondencia inicial encerrada en un círculo no cumple con las relaciones de distancia. Si revisamos las imágenes 5.2(a) y 5.2(b) podemos observar que se debe a que no corresponden a la misma esquina. En la Figura 5.3(b) se muestran las relaciones finales posteriores a la revisión de consistencia geométrica. Como puede observarse, una vez que se establece la correspondencia entre conjuntos de marcas es posible determinar el error en la estimación de la posición obtenida del odómetro. En la siguiente sección se explican en detalle las técnicas empleadas para determinar la posición real del robot.



(a) Verificación de consistencia geométrica (b) Correspondencia final

Figura 5.3: Verificación de la consistencia geométrica entre conjuntos de marcas.

5.3. Estimación de la posición

Una vez que se tienen las marcas que pasaron la prueba de la consistencia geométrica, se toman por pares y se calcula la posición del robot. La técnica empleada depende del tipo de marca, si se trata de discontinuidades o esquinas la estimación

se hace por medio de triangulación; si se trata de paredes la técnica empleada es la intersección de líneas. La diferencia radica en que mientras las discontinuidades y esquinas pueden verse como puntos en el espacio, las paredes son una superficie y sólo se conocen la magnitud y orientación de la normal a la pared.

Cada par de marcas ofrece una estimación de la posición del robot, sin embargo, debido a que la detección de las marcas no es exacta, a causa de la incertidumbre en los sensores, es necesario el uso de un esquema que permita fusionar el conjunto de estimaciones para obtener una sola. A continuación se describen las técnicas de triangulación e intersección de líneas empleadas y enseguida el esquema de fusión para la obtención de la estimación final de la posición.

5.3.1. Técnica de triangulación

Dado que el robot se mueve de una posición $A = (x_A, y_A)$ a una posición $B = (x_B, y_B)$, las marcas sirven como puntos de referencia para el cálculo de la posición del robot. Como se muestra en la Figura 5.4, m_0 y m_1 son dos marcas dentro del conjunto de correspondencias válidas. Sus coordenadas medidas desde el punto A son (x_0, y_0) y (x_1, y_1) respectivamente. Desde la nueva posición del robot B las marcas m_0 y m_1 son detectadas a una distancia d_0 y d_1 respectivamente. De esta forma la estimación de la nueva posición del robot $(\hat{x_B}, \hat{y_B})$ recae en la intersección de dos círculos cuyo centro son las marcas m_0 y m_1. Esto puede plantearse por medio del siguiente sistema de ecuaciones:

$$(\hat{x_B} - x_0)^2 + (\hat{y_B} - y_0)^2 = {d_0}^2 \tag{5.1}$$

$$(\hat{x_B} - x_1)^2 + (\hat{y_B} - y_1)^2 = {d_1}^2 \tag{5.2}$$

Por simplicidad, consideremos el origen del sistema de coordenadas en la marca m_0, la marca m_1 estará ubicada sobre el eje x a una distancia a; donde a es la magnitud de la distancia entre las dos marcas (ver Figura 5.5). De esta forma, las coordenadas de m_0 y m_1 son $(0,0)$ y $(a,0)$ respectivamente. Las ecuaciones 5.1 y 5.2 quedan de la siguiente manera:

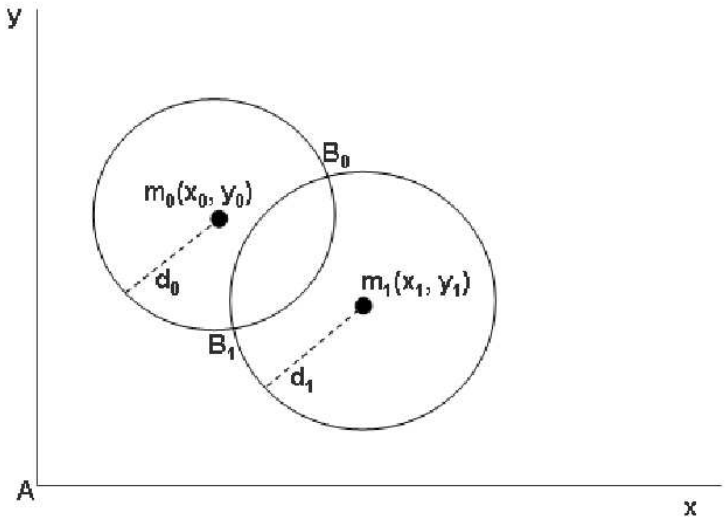


Figura 5.4: Técnica de triangulación para la estimación de la posición del robot.

$$\hat{x_B}^2 + \hat{y_B}^2 = {d_0}^2 \tag{5.3}$$

$$(\hat{x_B} - a)^2 + \hat{y_B}^2 = {d_1}^2 \tag{5.4}$$

Despejando $\hat{y_B}$ de 5.3 y sustituyendo en 5.4:

$$(\hat{x_B} - a)^2 + ({d_0}^2 - \hat{x_B}^2) = {d_1}^2 \tag{5.5}$$

Desarrollando 5.5:

$$\hat{x_B}^2 - 2a\hat{x_B} + a^2 - \hat{x_B}^2 = {d_1}^2 - {d_0}^2 \tag{5.6}$$

Resolviendo 5.6 para obtener $\hat{x_B}$:

$$\hat{x_B} = \frac{a^2 - {d_1}^2 + {d_0}^2}{2a} \tag{5.7}$$

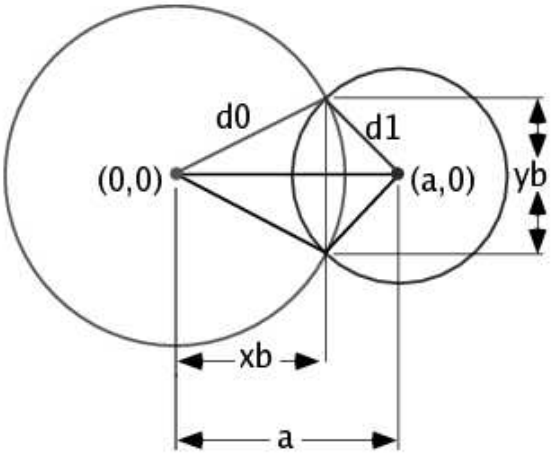


Figura 5.5: Cálculo de la intersección de dos círculos.

Sustituyendo 5.7 en 5.3 para obtener $\hat{y_B}$:

$$\hat{y_B}^2 = {d_0}^2 - \hat{x_B}^2 = {d_0}^2 - \left(\frac{a^2 - {d_1}^2 + {d_0}^2}{2a}\right)^2 \tag{5.8}$$

$$\hat{y_B}^2 = \frac{4a^2{d_0}^2 - (a^2 - {d_1}^2 + {d_0}^2)^2}{4a^2} \tag{5.9}$$

Resolviendo 5.9 para $\hat{y_B}$:

$$\hat{y_B} = \frac{1}{2a} \pm \sqrt{4a^2{d_0}^2 - (a^2 - {d_1}^2 + {d_0}^2)^2} \tag{5.10}$$

Como se observa existen dos intersecciones que representan las dos posibles ubicaciones del robot, Para decidir cual de las dos es la correcta se utiliza la aproximación proporcionada por el odómetro. La intersección que más se aproxime a la estimación del odómetro se asume que es la correcta.

5.3.2. Técnica de intersección de líneas

De manera complementaria, para obtener una estimación de la posición del robot en función de un par de paredes se usa la técnica de intersección de líneas. Como se observa en la Figura 5.6 inicialmente el robot se encuentra en el punto A donde conoce su posición y orientación. Desde ese punto detecta dos paredes no paralelas entre si: *pared 1* y *pared 2*. Además, como se tienen las rectas normales a dichas paredes, se conocen la distancia y orientación a la que se encuentran: (d_{A1}, θ_{A1}) para la *pared 1* y (d_{A2}, θ_{A2}) para la *pared 2*. El origen del sistema de coordenadas (X, Y) corresponde a la posición del robot, es decir, el punto A.

Enseguida el robot se mueve al punto B y detecta nuevamente ambas paredes. Desde este punto el robot no conoce su posición, pero conoce la distancia entre él y cada pared: d_{B1} a la *pared 1* y d_{B2} a la *pared 2*. Para el caso de la *pared 1* la posición del robot $B = (\hat{x_B}, \hat{y_B})$ recae sobre una línea paralela a dicha pared $L1$ que se encuentra a una distancia $d_{A1} - d_{B1}$ del punto A. Dado que se tiene la recta normal a la pared y su pendiente m, la pendiente de la recta paralela a la pared estaría dada por $-1/m$. En el caso de la *pared 2* la posición del robot $B = (\hat{x_B}, \hat{y_B})$ recae sobre una línea paralela a dicha pared $L2$ que se encuentra a una distancia $d_{A2} - d_{B2}$ del punto A.

De esta manera la posición $B = (\hat{x_B}, \hat{y_B})$ estará dada por la solución del sistema de ecuaciones formado por estas dos rectas $L1$ y $L2$. Una de las formas de resolver este problema es la siguiente: se utiliza la forma general $ax + by = c$, donde a, b, y c son las constantes que definen a la línea. Las líneas raramente se definen de esta forma, sin embargo esta puede derivarse de la ecuación punto-pendiente o de la ecuación de la recta que pasa por dos puntos. Sean los puntos (x_1, y_1), (x_2, y_2) que definen a una recta, los valores de a, b y c se obtendrían de la siguiente forma.

$$a = x_1 - x_2 \tag{5.11}$$

$$b = y_2 - y_1 \tag{5.12}$$

$$c = ax_1 - by_1 \tag{5.13}$$

Una vez determinados estos valores para cada una de las dos rectas paralelas el problema se reduce a resolver el siguiente sistema de ecuaciones para $\hat{x_B}$ y $\hat{y_B}$.

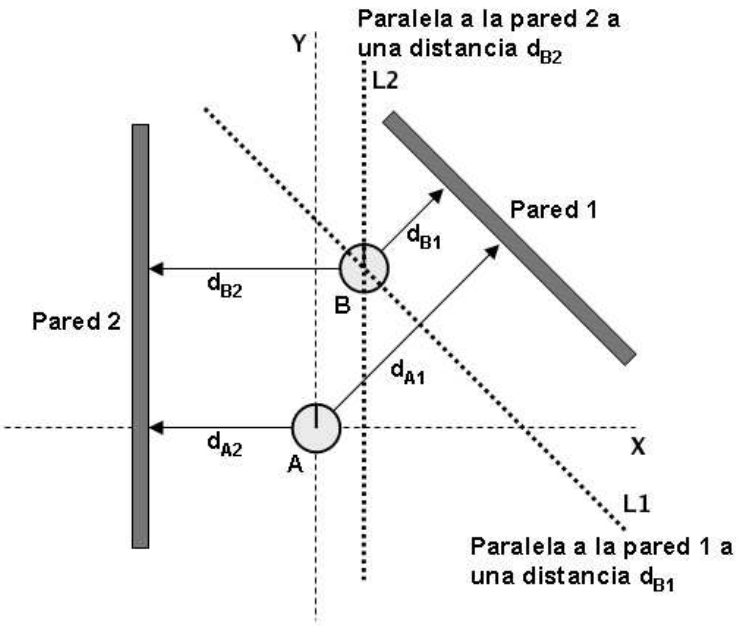


Figura 5.6: Cálculo de posición del robot por intersección de líneas.

$$a_1\hat{x_B} + b_1\hat{y_B} = c_1 \quad (L1) \tag{5.14}$$
$$a_2\hat{x_B} + b_2\hat{y_B} = c_2 \quad (L2) \tag{5.15}$$

Multiplicando 5.14 por b_2 y 5.15 por b_1:

$$a_1b_2\hat{x_B} + b_1b_2\hat{y_B} = b_2c_1 \tag{5.16}$$
$$a_2b_1\hat{x_B} + b_2b_1\hat{y_B} = b_1c_2 \tag{5.17}$$

Restando 5.17 de 5.16:

$$a_1b_2\hat{x_B} - a_2b_1\hat{x_B} = b_2c_1 - b_1c_2 \tag{5.18}$$

Finalmente dividiendo ambos lados de 5.18 entre $a_1b_2 - a_2b_1$ se obtiene la solución para $\hat{x_B}$:

$$\hat{x_B} = \frac{b_2c_1 - b_1c_2}{a_1b_2 - a_2b_1} \tag{5.19}$$

La ecuación para $\hat{y_B}$ se deriva de manera similar.

Las técnicas de intersección de círculos e intersección de líneas son utilizadas para obtener una estimación de la posición del robot. El lector debe notar que las coordenadas $(\hat{x_B}, \hat{y_B})$ que se obtienen en cada caso, son una aproximación a las coordenadas (x_B, y_B) que representan la posición real del robot. La diferencia entre la posición real y la estimada, depende de la precisión con que sean detectadas cada una de las marcas.

Otros factores influyen en la precisión de dicha estimación y son ajenos a la técnica de triangulación o intersección; estos se refieren al ángulo de separación entre las marcas utilizadas, a la distancia entre el robot y las marcas y a la posición de las mismas. Antes de presentar el esquema que permita fusionar las estimaciones, se abordan un par de consideraciones adicionales que permiten identificar la tolerancia al ruido de cada par de marcas. La primera consideración de refiere a la justificación formal de la tolerancia al ruido en función del ángulo de separación entre marcas. La segunda se refiere a la justificación de la relación entre la tolerancia al ruido y la distancia del robot a las marcas y la posición de estas.

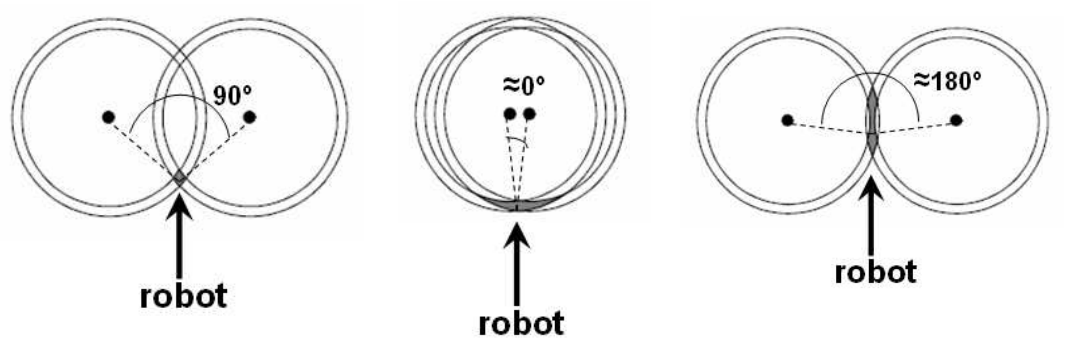
5.3.3. Dilución de la precisión

En [Kelly, 2003] el autor aborda formalmente el problema de la precisión, relacionado con el método de triangulación. Básicamente intenta determinar cual es la relación entre las marcas utilizadas y la calidad de la estimación obtenida. Para ello estudia la respuesta a pequeñas perturbaciones en las observaciones, en relación con el impacto que tienen en la estimación de la posición. Por perturbaciones en las observaciones se refiere al ruido en los sensores.

Como se explicó en la sección anterior, la información relativa a las marcas, que se utiliza en el conjunto de marcas del tiempo $t + \Delta t$ es la distancia del robot a las marcas. En base a las distancias a dos marcas se localiza al robot en la intersección de

dos círculos cuyos radios son dichas distancias. Sean r_0 y r_1 el radio de las marcas m_0 y m_1 respectivamente. Supongamos que esta distancia está sujeta a un margen de error de $\pm\varepsilon$ donde ε es la magnitud del error en la detección de una marca.

En la Figura 5.7 puede observarse que si se toma en cuenta el área de intersección de los anillos formados por $r_0 \pm \varepsilon$ y $r_1 \pm \varepsilon$, esta varía notablemente en tres casos particulares: cuando el ángulo de separación entre marcas relativo al robot es igual a 90° (ver Figura 5.7(a)), cuando el ángulo de separación es muy cercano a 0° (ver Figura 5.7(b)) y cuando el ángulo de separación es cercano a 180° (ver Figura 5.7(c)). De estas observaciones se desprende que el área de intersección decrece en el intervalo $[0^\circ, 90^\circ]$ y crece en el intervalo $[90^\circ, 180^\circ]$. Cuando el área de intersección es mínima (90°) la perturbación afecta en menor medida la estimación de la posición.



(a) Separación entre marcas cercana a 90°.

(b) Separación entre marcas cercana a 0° grados.

(c) Separación entre marcas cercana a 180°.

Figura 5.7: Dilución de la precisión en la estimación de la posición del robot.

El análisis matemático descrito en [Kelly, 2003] de la relación entre la magnitud de la perturbación y la precisión de la estimación, arroja que la variación de la precisión está determinada por la ecuación 5.20.

$$H = sin(\theta) \tag{5.20}$$

Donde H describe la relación entre las perturbaciones y la precisión de la estimación, θ es el ángulo de separación entre las marcas. En la Figura 5.8 se presenta la

gráfica asociada a la Ecuación 5.20.

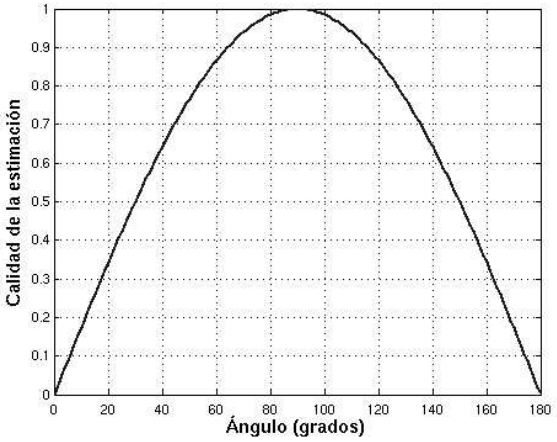


Figura 5.8: Calidad de la estimación en función del ángulo de separación entre marcas.

Claramente el valor de la función tiende a 0 cuando la separación entre las marcas se aproxima a 0° ó cuando se aproxima a 180°. El valor máximo se registra, cuando el ángulo de separación es un ángulo recto, por lo que pares de marcas con 90° de separación serán impactadas en menor medida por los errores en la detección de las marcas. En consecuencia la probabilidad de obtener una buena estimación crece conforme el valor de H se acerca a 90°. Este factor se considera dentro del esquema de fusión de múltiples estimaciones para asignarle un peso a cada estimación.

5.3.4. Calidad de la estimación

Varios argumentos se han planteado acerca de la estimación del error en la posición de un robot móvil. La mayor parte de ellos están basados en atributos de las marcas utilizadas, relacionados con aspectos geométricos de su distribución en el espacio. Para determinar cuales atributos son importantes se propone realizar la estimación de parámetros de la distribución de probabilidad de varios atributos.

La idea básica consiste en la recolección de un conjunto de datos de ejemplo, cada tupla de datos corresponderá a los atributos que han sido mencionados en la literatura como relevantes. Mientras el robot se mueve por el ambiente, utilizará pares de marcas para el cálculo de su posición, se etiquetará cada tupla con la clase que co-

rresponda de acuerdo a la magnitud del error. Determinar con precisión el error de la estimación en un robot real no es posible, a menos que se mida físicamente y se etiquete ejemplo por ejemplo. Para evitar este infructuoso esfuerzo se usará un simulador, se agregará un ruido aleatorio a las lecturas de los sensores y en base al odómetro del simulador, se determinará el error en la estimación de la posición y se discretiza en tres clases: buena, regular y mala.

Los atributos usados son: ángulo de separación entre marcas, distancia euclidiana entre marcas, distancia entre el robot y la marca más lejana y marcas en ambos lados del robot. Este último atributo se refiere a la ubicación de las marcas con respecto a la orientación del robot. Como se observa en la Figura 5.9 si suponemos que el sensor láser tiene un rango de -90° a 90° y trazamos una línea imaginaria a 0° el valor del atributo será 1 si las marcas m_1 y m_2 están separadas por dicha línea y 0 si no lo están. En la Tabla 5.1 se muestra un ejemplo de los datos recolectados.

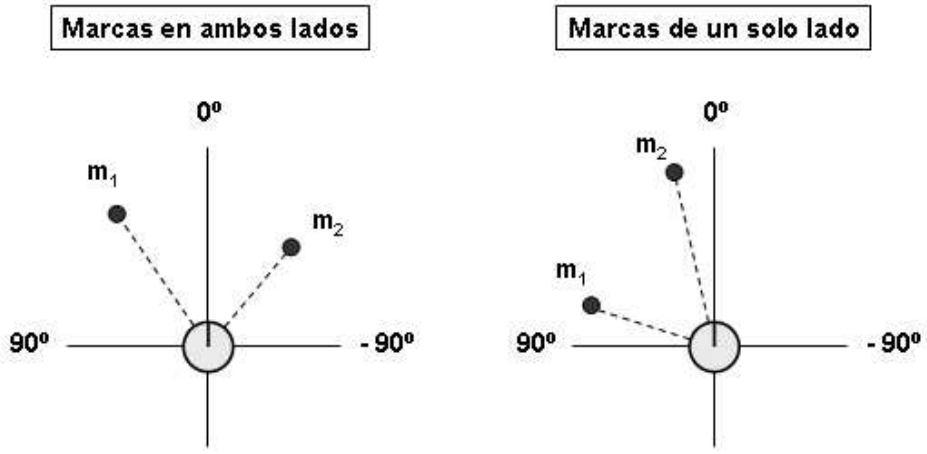


Figura 5.9: Valor del atributo *Marcas en ambos lados*.

El objetivo es la estimación de parámetros de la función de densidad de probabilidad de cada atributo. Para los datos continuos se asume que se trata de una Gaussiana con media u y varianza σ, para el atributo discreto basta con calcular la tabla de probabilidad condicional en base al número de ocurrencias de cada valor con respecto al total para cada valor de la clase.

La recolección de los ejemplos se realizó utilizando el simulador *Stage* [Gerkey et al., 2003]. Para ello se hicieron varios recorridos en dos ambientes simulados (ver

Tabla 5.1: Ejemplo de datos de entrenamiento

Separación θ	**Distancia**	**Lejanía**	**Ambos lados**	**Clase**
40	1.2	2.0	0	regular
90	0.8	2.7	1	buena
10	2.2	2.3	1	mala
15	1.8	6.0	0	mala
102	3.2	2.1	1	buena
48	4.2	7.0	0	regular

Figuras 8.2 y 8.3 en el Capítulo 8). El robot simulado fue guiando a control remoto por una persona mientras recolectaba los ejemplos.

El simulador no es capaz de simular ruido por lo que se introdujo un ruido aleatorio dentro de un rango de $\pm$2cm., sobre el 50 % de las lecturas del sensor laser. Mientras el robot se movía se recolectaron aproximadamente 6,000 ejemplos de acuerdo con el formato mostrado en la Tabla 5.1. Para efectos prácticos cada tupla de atributos se etiquetó con la clase de acuerdo con la magnitud del error de la siguiente manera:

$$\text{Clase} = \begin{cases} \text{buena} & 0\text{cm.} <= \varepsilon <= 2\text{cm.} \\ \text{regular} & 2\text{cm.} < \varepsilon <= 4\text{cm.} \\ \text{mala} & 4\text{cm.} < \varepsilon \end{cases} \tag{5.21}$$

El atributo *distancia entre marcas* no aportaba información adicional, de alguna manera es redundante con el de *ángulo de separación entre marcas* o es irrelevante ya que el valor de la media era prácticamente el mismo en todas las clases.

El resto de los atributos muestran una clara tendencia: El atributo *Ángulo de separación* no resultó ser una Gaussiana, sin embargo, la distribución de probabilidad puede ser representada por un par de Gaussianas para cada clase. La primera de ellas corresponde a los datos cuyo valor es menor a 90° y la segunda corresponde a los datos mayores a 90°. Las medias y desviaciones estándar para las 3 clases se muestran en la Tabla 5.2. De la tabla se puede apreciar que los resultados son consistentes con lo expuesto en la sección anterior acerca del trabajo de [Kelly, 2003]: la media de las Gaussianas se aleja del valor de 90° al mismo tiempo que la calidad de la estimación decrece.

Tabla 5.2: Parámetros calculados para el atributo *Ángulo de separación*.

	u_1	σ_1	u_2	σ_2
buena	62.32	22.43	121.78	19.23
regular	32.73	23.54	159.11	25.11
mala	12.22	5.21	177.63	12.62

Sorpresivamente el atributo *Marcas en ambos lados* muestra claramente que cuando las marcas están ubicadas una de cada lado del robot (valor 1) la estimación es mejor que si ambas están ubicadas a la izquierda o a la derecha de él (valor 0). Los parámetros para este atributo se muestran en la Tabla 5.3

Tabla 5.3: Parámetros calculados para el atributo *Marcas en ambos lados*.

	0	**1**
buena	0.39	0.61
regular	0.79	0.21
mala	0.88	0.12

Por último el atributo *Distancia entre el robot y la marca más lejana* muestra que entre más lejos del robot esté una de las marcas, el impacto del error en la detección será mayor, vea la Tabla 5.4.

Tabla 5.4: Parámetros para el atributo *Distancia entre el robot y la marca más lejana*.

	u	σ
buena	1.17	0.80
regular	3.23	0.94
mala	4.93	1.15

Finalmente y retomando el objetivo de asociar atributos de los pares de marcas con el error esperado, se asigna un peso a cada estimación dependiendo de 3 atributos: ángulo de separación entre marcas, distancia del robot a la marca más lejana y marcas en ambos lados del robot. Para cada uno de ellos se define una función: f_{ang}, f_{dis} y f_{lados} respectivamente. Cada función devolverá un valor entre 0 y 1 que representa la calidad de la estimación en función de cada atributo y los valores devueltos por las tres funciones serán promediados para obtener el peso de cada estimación.

En el caso del atributo *ángulo de separación entre marcas* se tomará la Ecuación 5.20 propuesta en [Kelly, 2003]. En el caso del atributo *distancia entre el robot y*

la marca más lejana se utilizará una función que aproxime el comportamiento observado en la Tabla 5.4. En la Figura 5.10 se presentan las Gaussianas que representan las distribuciones de probabilidad para cada clase. Como se observa la calidad de la estimación decrece conforme la marca se encuentra más lejos del robot. En el conjunto de los datos recolectados para elaborar la gráfica la gran mayoría de ellos tienen valores entre 1 y 5.5 metros. Esto último se debe a que el robot fue guiado a través del ambiente sin acercarse mucho a los obstáculos. Por otro lado, debido a que los ambientes no tienen espacios libres muy grandes el robot tampoco detectó marcas cerca del límite perceptual del sensor láser (8 metros).

Con la finalidad de definir una función que abarque todo el rango de valores posibles (de 0 a 8 metros) y que además observe un comportamiento similar al observado en la Tabla 5.4, se utiliza la función mostrada en la Ecuación 5.23 cuya gráfica se muestra en la Figura 5.11

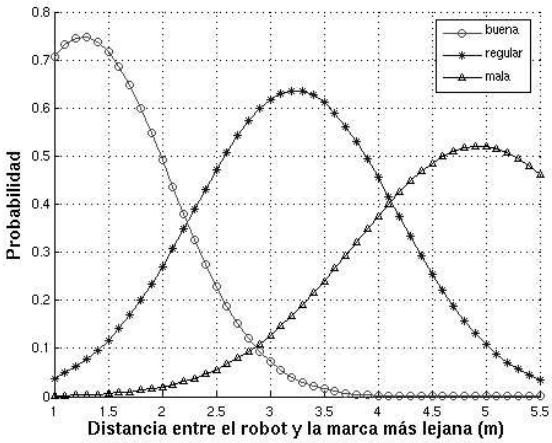


Figura 5.10: Distribuciones de probabilidad para el atributo *distancia entre el robot y la marca más lejana*.

Por último, para el atributo *marcas en ambos lados del robot* se toma directamente de la Tabla 5.3 la probabilidad condicional de que la estimación sea buena dado que las marcas se encuentran en ambos lados del robot (valor 1) o ambas del mismo lado (valor 0). Las funciones quedan entonces de la siguiente manera:

$$f_{ang}(\theta) = sin(\theta) \tag{5.22}$$

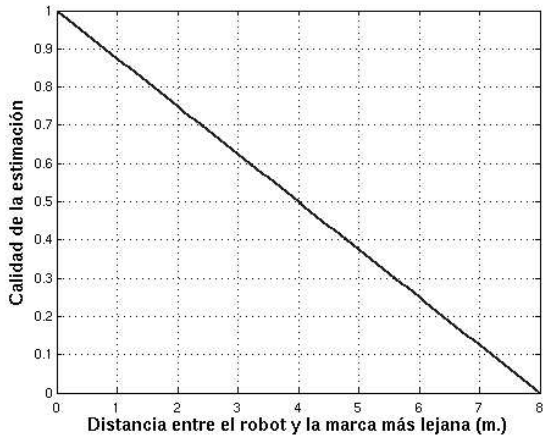


Figura 5.11: Función monotónica decreciente que aproxima la distribución de probabilidad para el atributo *distancia entre el robot y la marca más lejana*.

$$f_{dis}(m) = 1 - \frac{m}{8.0} \tag{5.23}$$

$$f_{lados}(\lambda) = \begin{cases} 0.61 & \text{si } \lambda = 1 \\ 0.39 & \text{si } \lambda = 0 \end{cases} \tag{5.24}$$

En la Ecuación 5.23 se usa el valor de 8.0 debido a que es la máxima distancia posible a la que puede identificar una marca, por lo que la función decrece linealmente para m en el intervalo $[0, 8.0]$. En la Ecuación 5.24, λ es 1 si las marcas se encuentran en ambos lados del robot y 0 si se encuentran en un solo lado. A continuación se muestra el esquema de fusión de estimaciones en donde son utilizadas estas funciones.

5.3.5. Fusión de las estimaciones

La triangulación y la intersección de rectas se hacen tomando pares de marcas, así que para un conjunto de N marcas se hacen aproximadamente $L = N * (N-1)/2$ veces el proceso. Estas estimaciones, aunque no corresponden exactamente a un mismo punto, están muy cercanas una de la otra. De acuerdo con lo visto en la subsección anterior, se asocia un peso p_i con cada estimación $(\hat{x_{Bi}}, \hat{y_{Bi}})$ donde p_i es el promedio del

valor devuelto por las funciones $f_{ang}(\theta_i)$, $f_{dis}(m_i)$ y $f_{lados}(\lambda_i)$. θ_i es el ángulo de separación entre las marcas que dieron origen a la estimación $(\hat{x_{Bi}},\hat{y_{Bi}})$, de la misma forma m_i es la distancia del robot a la marca más lejana y λ_i es 1 si las marcas se encuentran en ambos lados del robot ó 0 en caso contrario. Para obtener una sola estimación se calcula el promedio pesado de todas las estimaciones como se muestra en la Ecuación 5.25. En este proceso se incluye la estimación del odómetro como una estimación más con un peso de 0.5.

$$(\bar{x_B},\bar{y_B}) = \frac{\sum_{i=1}^{L} p_i(\hat{x_{Bi}},\hat{y_{Bi}})}{P} \tag{5.25}$$

donde:

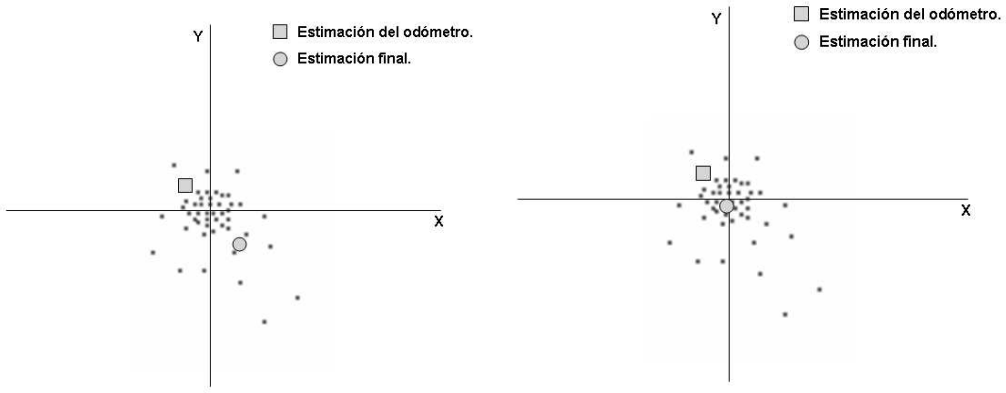
$$P = \sum_{i=1}^{L} p_i \tag{5.26}$$

con:

$$p_i = \frac{f_{ang}(\theta_i) + f_{dis}(m_i) + f_{lados}(\lambda_i)}{3} \tag{5.27}$$

En la Figura 5.12 puede verse un ejemplo de la necesidad de utilizar un promedio pesado para fusionar múltiples estimaciones de la posición del robot. Cada estimación es generada por un par de marcas. En las Figuras 5.12(a) y 5.12(b) se muestra un conjunto de estimaciones que deben ser fusionadas en una sola. Los puntos negros son estimaciones obtenidas de pares de marcas, el cuadrado indica la estimación que ofrece el odómetro y el círculo indica la estimación final de la posición del robot. Considere que la posición real del robot es el origen del plano cartesiano (X,Y).

Obviamente la estimación final será mejor en la medida que se acerque a la posición real del robot, que en este caso es el origen. En la Figura 5.12(a) se observa que debido a que se hace un promedio simple de las estimaciones, la estimación final no resulta adecuada. Esto ocurre porque las estimaciones lejanas a la posición real ocasionadas frecuentemente por ruido en los sensores, afectan fuertemente al promedio simple, lo cual conduce a una mala estimación. Por el contrario, en la Figura 5.12(b) se muestra el mismo conjunto de estimaciones las cuales fueron fusionadas utilizando un



(a) Fusión de las estimaciones sin asignar un peso a cada estimación.

(b) Fusión de las estimaciones asignando un peso a cada estimación.

Figura 5.12: Fusión de múltiples estimaciones de la ubicación del robot.

promedio con un peso asociado a cada estimación. La estimación final mejora considerablemente debido a que las estimaciones lejanas a la real tienen un peso mucho menor en función de los atributos asociados a los pares de marcas que las generaron.

5.4. Estimación de la orientación

Una vez calculada la posición $\overline{B} = (\overline{x_B}, \overline{y_B})$, es necesario calcular el cambio en la orientación del robot. Como ya se conoce $\overline{B}$, este cambio puede ser calculado usando cualquier correspondencia uno a uno entre las marcas de un conjunto y otro. Sean $m_i \in M_t$ y $m_k \in M_{t+\Delta t}$ un par de marcas correspondientes, la diferencia de orientación entre la posición A y la posición B esta dada por:

$$\hat{\theta} = \beta - \beta' \tag{5.28}$$

Donde $\hat{\theta}$ es el cambio en la orientación del punto A al punto B. β es el ángulo con respecto a la orientación del robot de la marca m_k desde el punto B. β' es calculada usando las coordenadas de la marca m_i y la nueva posición del robot $(\overline{x_B}, \overline{y_B})$. En

términos prácticos β' es la orientación a la que el robot debería ver la marca m_i desde el punto B si no existiera variación en la orientación. En la Figura 5.13 puede apreciarse la relación geométrica entre β y β'. De manera análoga al cálculo de la posición del robot, el cálculo de la orientación del robot se obtiene del promedio de las diferencias en orientación que proporcionan las marcas correspondientes en los conjuntos $M_{t+\Delta t}$ y M_t.

$$\bar{\theta} = \frac{\sum_{i=1}^{N} \hat{\theta}}{N} \tag{5.29}$$

Donde $\bar{\theta}$ es el promedio de las diferencias en orientación obtenidas de las marcas correspondientes entre M_t y $M_{t+\Delta t}$, N es el número de correspondencias encontradas entre el conjunto M_t y el conjunto $M_{t+\Delta t}$.

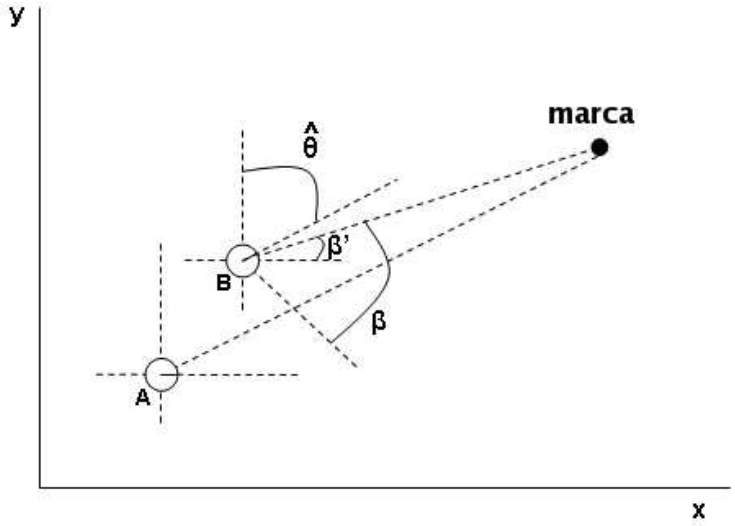


Figura 5.13: Estimación de la orientación del robot.

5.5. Comentarios finales

En esta sección se presentó un esquema de seguimiento de la posición usando marcas naturales del ambiente. De acuerdo con lo visto en el Capítulo 4 la base de este

esquema es la identificación de marcas de las lecturas del sensor láser. Posteriormente se determina una correspondencia entre conjuntos de marcas que el robot ve desde lugares diferentes y se verifica que sean consistentes geométricamente, es decir, que estén distribuidas de la misma manera en el espacio.

Si solamente se hiciera uso de la correspondencia entre marcas, la precisión dependería enteramente de la robustez de los algoritmos de identificación de marcas. Esta robustez es difícil de garantizar debido al ruido inherente a la naturaleza de los sensores y el ambiente. Es por esto que la verificación de la consistencia geométrica es de gran ayuda ya que permite descartar correspondencias que no sean válidas. Por ejemplo, supongamos que una persona camina frente al robot mientras este se encuentra realizando el proceso de seguimiento de la posición; esto añadirá marcas que no estaban presentes anteriormente. Cabe entonces la posibilidad de que el robot las identifique y asocie con alguna de las que observó previamente, sin embargo al verificar la consistencia geométrica estas marcas serán eliminadas debido a que no serán consistentes con el resto de las marcas que se mantuvieron en el mismo lugar. La única situación en que el robot podría establecer una correspondencia errónea sería en en caso de que varios objetos estuvieran moviéndose a la misma velocidad siguiendo la misma trayectoria. Esto ocasionaría que existieran dos subconjuntos consistentes entre si: el de las marcas ocasionadas por los objetos inmóviles y el de las marcas ocasionadas por los objetos que se mueven. El algoritmo propuesto no sería capaz de identificar cuál de estos dos subconjuntos es el correcto y se incurriría en errores de localización. La situación planteada, aunque problemática, es muy poco probable que se dé en un ambiente real.

En cuanto a la estimación del error esperado, esta se hizo necesaria debido a que una vez obtenidas las estimaciones de la posición del robot, estas se presentaban a menudo concentradas alrededor de la posición real, con el inconveniente de que algunas pocas estaban dispersas. Aunque eran pocas, a la hora de obtener el promedio de las estimaciones, las malas influían negativamente en la estimación final. Este fenómeno se debe a que no todos los pares de marcas arrojan una estimación igualmente buena, aún cuando cumplen con la consistencia requerida. Con la finalidad de identificar a las malas estimaciones se asignó un *peso* a cada una de ellas, donde el *peso* depende de los factores identificados como determinantes sobre la calidad de la estimación.

Otra de las razones por las que es necesaria una estimación de la posición lo más exacta posible es porque de ella depende la estimación de la orientación. Una vez calculada la posición del robot, se calcula la diferencia entre la orientación a la que el robot esperaba encontrar una marca y la orientación en la que realmente se encuentra.

Como puede suponerse, otra de las limitantes de este esquema es que el robot necesita identificar al menos 2 esquinas o discontinuidades para localizarse ó, en su defecto, 1 esquina y 2 paredes no paralelas. Los únicos casos en que el robot se quedaría sin información para localizarse son: un cuarto circular (en el que incluso una persona perdería la orientación) y un pasillo muy largo (más de 8 metros de largo). El problema del pasillo es considerablemente difícil ya que si este se encuentra despejado, el robot solo dispondría de dos paredes paralelas. Esto le impediría determinar con precisión su posición en al ambiente ya que solo sabría que se encuentra en algún lugar sobre una línea paralela a las paredes.

Capítulo 6

Navegación

En este capítulo de abordarán los aspectos relacionados con el movimiento del robot. Para ello se considera que el robot conoce su ubicación dentro del ambiente, la meta a donde debe dirigirse y que además dispone de una trayectoria a seguir. En el Capítulo 3 se expuso el método propuesto para la planificación de trayectorias; como se mencionó, lo que se obtiene es un conjunto de puntos de verificación por los que el robot debe pasar. También se considera, de acuerdo con lo expuesto en el Capítulo 5, que debe realizarse un seguimiento de la posición mientras el robot se mueve.

En la primera parte de este capítulo se presenta el control de navegación empleado el cual es el encargado de determinar los movimientos que el robot debe realizar para recorrer la trayectoria planeada pasando por los puntos de verificación que la conforman. En la segunda parte, se aborda el problema de la detección y evasión de obstáculos imprevistos estáticos. Esta necesidad surge cuando existen obstáculos en el ambiente que no fueron contemplados en la construcción del mapa. En consecuencia, es factible que el robot genere trayectorias por regiones que actualmente contienen obstáculos.

6.1. Control de navegación

El control de navegación es el componente que determina cuales son los movimientos que el robot debe realizar para recorrer una trayectoria determinada. Como se indica en el Capítulo 3 el planificador de trayectorias genera un conjunto con los puntos

de verificación por los que el robot debe pasar para llegar a la meta. Este conjunto se denota por $C = (c_1, c_2, ..., c_n)$ donde $c_i = (x_i, y_i)$ es el *i-ésimo* punto de verificación por el que debe pasar. En principio se asume que el robot se encuentra en el punto c_1.

Enseguida, el control de navegación debe ser capaz de determinar cuál es el siguiente movimiento (girar o avanzar) que el robot debe realizar con la finalidad de seguir la ruta planeada. Además del movimiento requerido, deberá determinar la magnitud del giro o desplazamiento necesario y la velocidad a la que debe realizarse el movimiento. A continuación se aborda cada uno de estos aspectos.

6.1.1. Seguimiento de la trayectoria planeada

Para moverse de un punto de verificación $c_i = (x_i, y_i)$ al siguiente punto de verificación $c_{i+1} = (x_{i+1}, y_{i+1})$ se toma en cuenta la orientación actual del robot θ y la orientación requerida θ_r para moverse en línea recta de c_i a c_{i+1}. Sean $\Delta y = y_{i+1} - y_i$ y $\Delta x = x_{i+1} - x_i$; para determinar θ_r se calcula el valor de $\arctan(\Delta y / \Delta x)$ incluyendo el cálculo del cuadrante en el que se encuentra el resultado en función de los signos de Δy y Δx. En la Figura 6.1 se observa la manera en que se determina el cuadrante en el que se encuentra la solución de la función arctan dependiendo de los signos, esto último de realiza para que el ángulo resultante θr tenga la orientación correcta de acuerdo con los puntos c_i y c_{i+1}.

Cuando el robot se encuentra en una orientación θ diferente de θ_r es necesario hacer un giro Θ para colocarse en la orientación requerida θ_r. Para ello se calcula el giro necesario para ir de θ a θ_r en sentido horario Θ_h y también el giro necesario para ir de θ a θ_r en sentido anti-horario Θ_a. Una vez hecho esto de toma el giro cuya magnitud sea la menor entre Θ_h y Θ_a.

$$\Theta = \begin{cases} \Theta_h & \text{si } \mid \Theta_h \mid \leq \mid \Theta_a \mid \\ \Theta_a & \text{si } \mid \Theta_a \mid < \mid \Theta_h \mid \end{cases} \tag{6.1}$$

Una vez que el robot ha girado y su orientación es θ_r la magnitud d del desplazamiento para llegar de c_i a c_{i+1} es:

$$d = \sqrt{(y_{i+1} - y_i)^2 + (x_{i+1} - x_i)^2} \tag{6.2}$$

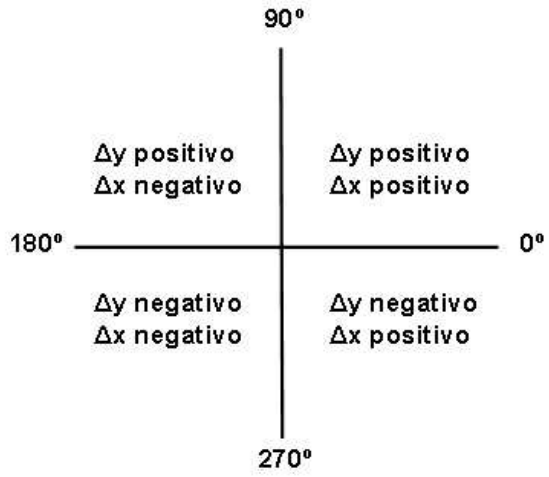


Figura 6.1: Cálculo del cuadrante para la solución de la función arctan.

Esta misma operación se repite para llegar a los siguientes puntos de verificación. Como puede suponerse el robot necesita detenerse en cada punto de verificación para realizar el giro que lo oriente en dirección del siguiente punto de verificación. Tanto para detenerse cuando está en movimiento como para ponerse en movimiento cuando está detenido se requiere de un control de la velocidad debido a que detenerse bruscamente podría ocasionar que el robot perdiera el equilibrio e incluso podría caerse. A continuación se aborda el problema del control de la velocidad.

6.1.2. Control de la velocidad

Para controlar la velocidad a la que se desplaza el robot de definen 3 parámetros para la velocidad lineal y 3 parámetros para la velocidad angular. Los primeros 3 son $velL_{min}$, $velL_{med}$ y $velL_{max}$; velocidad lineal mínima, velocidad lineal media y velocidad lineal máxima respectivamente. De manera análoga los 3 restantes son $velA_{min}$, $velA_{med}$ y $velA_{max}$; velocidad angular mínima, velocidad angular media y velocidad angular máxima respectivamente.

Cuando el robot inicia su movimiento lo hace a la velocidad mínima, después de cierta cantidad de metros o grados, aumenta la velocidad a la velocidad media y posteriormente a la velocidad máxima. Se mantiene en la velocidad máxima hasta que

se encuentra cerca del siguiente punto de verificación o de la orientación deseada, según sea el caso. Enseguida pasa de la velocidad máxima a la velocidad media, luego a la velocidad mínima y por último a velocidad 0. En la Figura 6.2 puede apreciarse la variación de la velocidad lineal en función de la cercanía con los puntos de verificación. Los puntos c_1, c_2 y c_3 representan tres puntos de verificación consecutivos en donde el robot se detiene para girar y posteriormente comenzar a desplazarse nuevamente. En la Figura 6.3 puede apreciarse la variación en la velocidad angular en función de la cercanía con el inicio y el fin del giro. En el caso específico del ejemplo mostrado en la gráfica se trata de un giro de 70°.

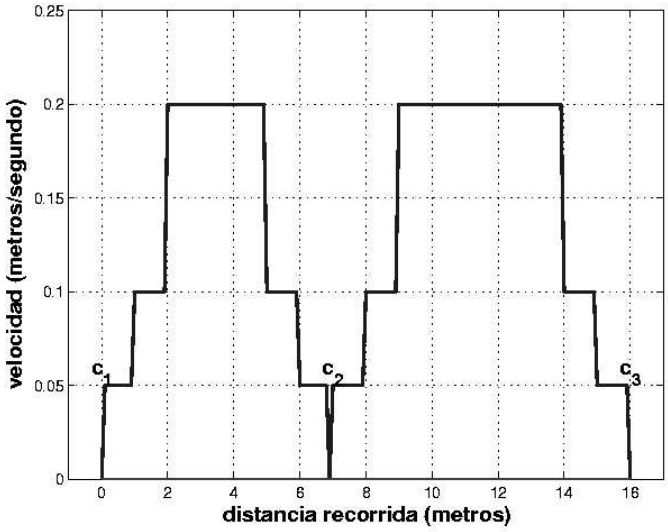


Figura 6.2: Control de la velocidad de desplazamiento del robot en función de la proximidad con los puntos de verificación.

6.2. Evasión de obstáculos

En esta sección se aborda el problema de evasión de obstáculos, el objetivo es que el robot sea capaz de detectar la presencia de un obstáculo estático que impida la ejecución de la trayectoria planeada. Usando sus sensores el robot debe ser capaz de detectarlos y posteriormente de encontrar una trayectoria alterna que lo lleve a la meta. Además existen situaciones en las que el obstáculo en cuestión obstruye la única vía de

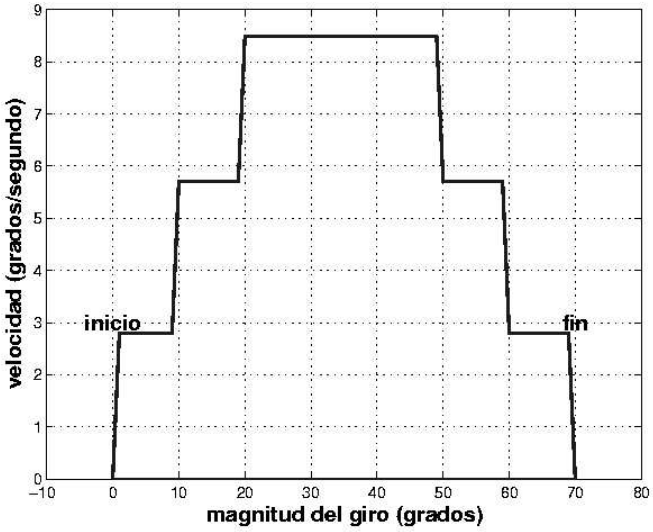


Figura 6.3: Control de la velocidad de giro del robot en función de la proximidad con el inicio y fin del giro.

acceso hacia la meta por lo que el robot deberá notificar que la meta está en un lugar inaccesible.

6.2.1. Detección de obstáculos

Para asegurarse de no chocar, el robot vigila la región frente a él permanentemente. Mientras navega, realiza una fusión de las lecturas del sensor láser que corresponden al frente del robot y los 3 sonares frontales del robot. El uso de sólo 3 sonares frontales obedece a que el robot solamente se mueve hacia el frente en línea recta y no se consideran obstáculos dinámicos en el ambiente.

También se relaciona con la necesidad de que ambos sensores vigilen la misma región frente al robot, ya que mientras el sensor láser tiene una alta precisión los sonares forman conos cuya zona de cobertura es difícil determinar con precisión. Esta fusión obedece a una estrategia conservadora y es en extremo simple, consiste en encontrar el mínimo entre las medidas devueltas por ambos sensores.

El uso de dos sensores para esta tarea en particular, tiene una razón muy importante: el sensor láser utilizado no opera adecuadamente cuando los obstáculos son cristales o superficies semi transparentes. Si sólo se tomara en cuenta la información del sensor láser el robot correría el riesgo de chocar, por ejemplo, contra una puerta de cristal. Por otro lado, los sonares, aunque no son muy precisos en la distancia medida, responden mejor que el láser ante puertas de cristal, cuando éstas se encuentran frente al sonar, debido a que se fundamentan en la reflexión de sonido de alta frecuencia.

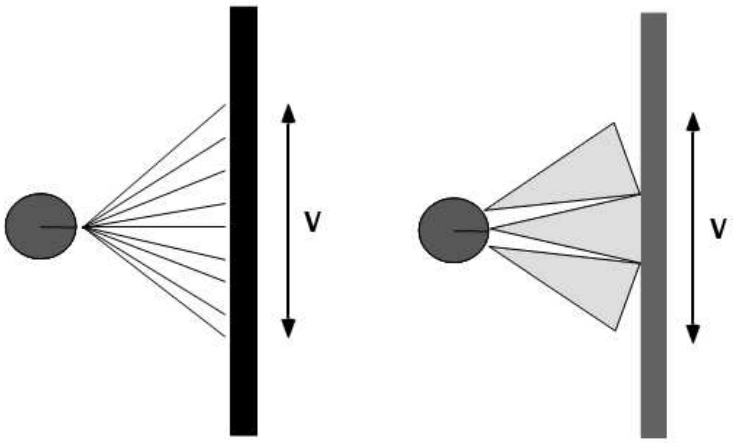
Se define una distancia mínima permitida de acercamiento a los obstáculos que se denota por M. Al registrar un obstáculo a una distancia menor que M se considera que el robot está en peligro de colisionar y se lleva a cabo la estrategia de evasión. Para efectos prácticos $M = 30cm.$ se considera una distancia crítica prudente y necesaria. Prudente porque el robot puede detenerse sin problema y necesaria porque, como se mencionará posteriormente en este capítulo, en ocasiones el robot debe acercarse a los obstáculos para determinar si puede o no pasar por un espacio reducido.

La amplitud de la región que el robot debe vigilar está en función del diámetro del robot ya que debe ser suficientemente grande para evitar que el robot choque y suficientemente pequeña para que le permita moverse en espacios reducidos como puertas. Dado que nuestro robot tiene un diámetro de aproximadamente 40 cm. y una puerta tiene una apertura estándar de 90 cm. el tamaño de la región vigilada por el robot es de $V = 60cm.$ con la finalidad de que el robot pueda pasar por lugares estrechos.

En la Figura 6.4 se ilustra la estrategia utilizada: el rectángulo que se observa en ambas figuras representa un obstáculo imprevisto, la amplitud de la región vigilada al frente del robot es V y las lecturas que se indican corresponden al láser (Figura 6.4(a)) y a los sonares (Figura 6.4(b)). Mientras el robot se mueve se calcula el mínimo de las lecturas de ambos sensores y si éste es menor que M se determina que se ha detectado un obstáculo.

6.2.2. Planificación de una trayectoria alterna

La evasión del obstáculo una vez que ha sido detectado, es muy simple y está estrechamente relacionada con la planificación de trayectorias. Si el mínimo de los sensores es menor que un cierto límite crítico de cercanía con los obstáculos (M), el robot se detiene y se realiza el procedimiento de evasión de obstáculos descrito a continuación.



(a) Obstáculos imprevistos detectados con el sensor láser.

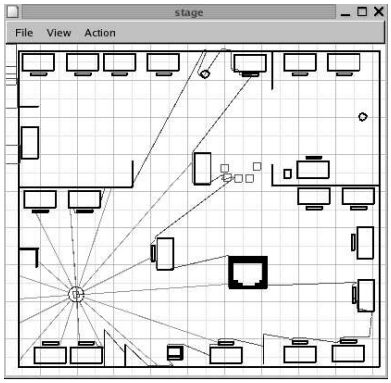
(b) Obstáculos imprevistos detectados con los sonares frontales.

Figura 6.4: Vista superior del robot mientras navega. Permanentemente vigila la región frontal para prevenir colisiones.

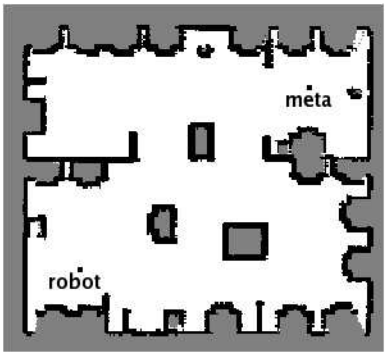
1. Alterar temporalmente el mapa del ambiente, incluyendo los nuevos obstáculos detectados.
2. Propagar la región del mapa de cercanía a obstáculos con los nuevos obstáculos incluidos.
3. Usar el mismo algoritmo de planificación de trayectorias descrito en el Capítulo 3.

En la Figuras 6.5 y 6.6 puede verse un ejemplo de la reacción del robot ante la presencia de obstáculos imprevistos.

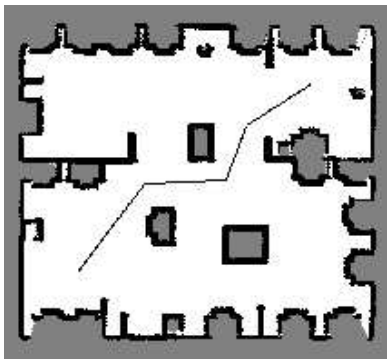
Es importante mencionar que algoritmo de evasión de obstáculos se encargará de que el robot no choque con los obstáculos siempre y cuando estos sean estáticos, la evasión de obstáculos dinámicos presenta diferentes retos y no será abordado en este trabajo. Ahora bien, si una persona se encuentra en el camino del robot obstruyendo la trayectoria planeada, la persona será considerada como un obstáculo estático cualquiera.



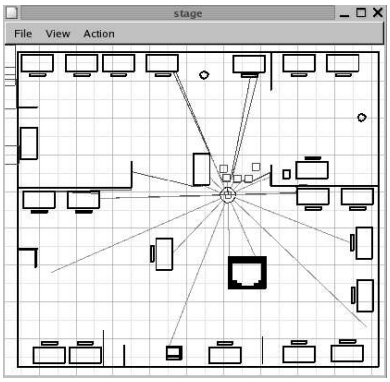
(a) Ambiente donde se indica la posición inicial del robot.



(b) Puntos inicial y final indicados.

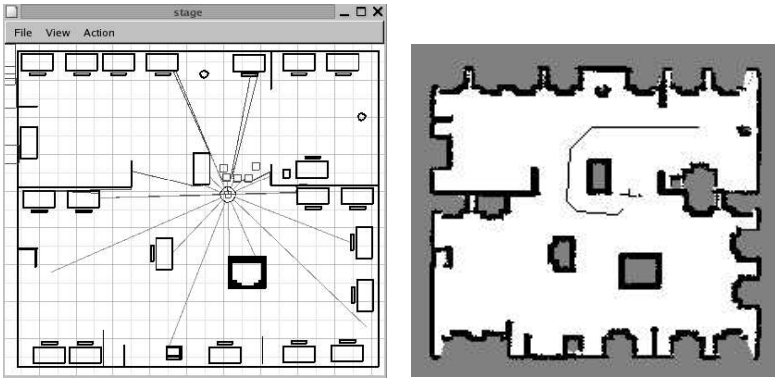


(c) Ruta inicial trazada por el planificador.



(d) Obstáculos imprevistos detectados durante el recorrido de una trayectoria.

Figura 6.5: Primer paso: Inicialmente el robot conoce su posición y la ubicación de la meta, determina una ruta inicial e intenta recorrerla



(a) Obstáculos imprevistos detectados durante el recorrido de una trayectoria.

(b) Nueva trayectoria para llegar a la celda destino usando el mapa modificado.

Figura 6.6: Segundo paso: El robot detecta un obstáculo imprevisto, modifica temporalmente el mapa original incluyendo en nuevo obstáculo y determina una nueva trayectoria.

6.2.3. Lugares inaccesibles

Cuando se le da al robot la instrucción de llegar a una meta, lo primero que necesita es una trayectoria inicial. Mientras se mueve, podría necesitar el procedimiento de evasión de obstáculos en varias ocasiones. No existe un límite definido para el número de intentos posibles. El robot seguirá intentando llegar a la meta siempre que exista una trayectoria alterna. En el momento que no le sea posible determinar una trayectoria a la meta, el robot informará al usuario que la meta se encuentra en una lugar inaccesible. Esta situación podría darse por diferentes razones.

Por ejemplo: supongamos que el proceso de construcción de mapas de un ambiente de oficina se lleva a cabo con todas las puertas abiertas. Posteriormente utilizando la interfaz remota, se le indica al robot que su destino se encuentra dentro de una de las oficinas cuya puerta se encuentra cerrada. Desde su ubicación, el robot utiliza el planificador de trayectorias para determinar un camino hacia la meta. Como la trayectoria se construye de acuerdo al mapa, el robot determina los movimientos necesarios para llegar a la meta y comienza a moverse. Al acercarse a la puerta, el robot detecta un obstáculo sobre la trayectoria que tenía planeada; enseguida se incluyen temporalmente

los nuevos obstáculos observados al mapa, y se intenta determinar una nueva trayectoria. Es en este punto cuando el planificador le indica al robot que no existe ninguna trayectoria posible, el robot recibe el mensaje y por último se lo comunica al usuario a través de la interfaz remota.

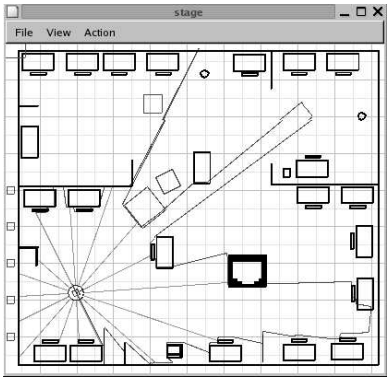
6.2.4. Inclusión de nuevos obstáculos en el mapa

Es claro que durante el recorrido que el robot debe realizar para llegar a la meta puede encontrar más de un obstáculo imprevisto. Como se asume a lo largo de esta tesis, existe un mapa previamente construido del ambiente el cual es alterado temporalmente ante la presencia de obstáculos imprevistos. Ahora bien, las alteraciones ocasionadas por dichos obstáculos persisten hasta el momento en que el robot llega a la meta o hasta que determina que no es posible llegar a la meta. Cuando el usuario le indica al robot que debe ir hacia una ubicación dentro del ambiente el robot inicia con un mapa temporal idéntico al original. Durante el trayecto el robot incorpora al mapa temporal todos los obstáculos imprevistos que le impidan seguir con una trayectoria. Una vez que el robot llega a la meta ó determina que no es posible llegar a ella de acuerdo con la nueva distribución de obstáculos, el mapa temporal es desechado y el mapa original volverá a ser usado en la siguiente ocasión.

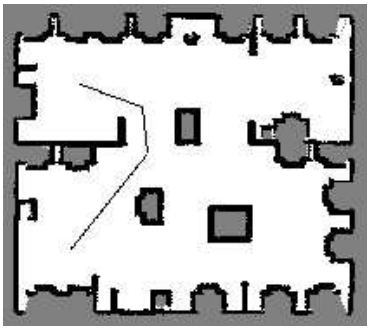
Estás últimas consideraciones tienen la ventaja de reflejar la nueva distribución de los obstáculos en el ambiente lo que le permite al robot determinar cuando no es posible llegar a la meta. Por otro lado la desventaja radica en que si algún obstáculo imprevisto que ya fue incorporado al mapa se mueve de su lugar, entonces dejará de bloquear el camino, lo cual ya no será considerado por el robot. Esto último podría ocasionar que el robot determine que no es posible llegar a la meta cuando en realidad si existe una trayectoria posible. En las Figuras 6.7 y 6.8 se muestra un caso que refleja la ventaja de incluir de manera acumulativa los obstáculos imprevistos detectados por el robot.

6.3. Comentarios finales

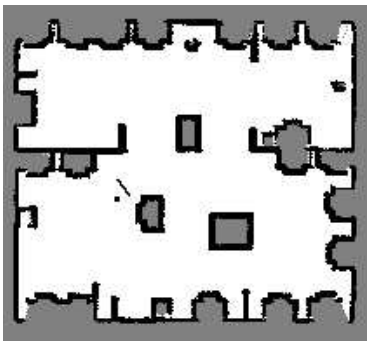
En este capítulo se abordó el problema de detección de obstáculos estáticos imprevistos. Como puede apreciarse, el problema se concentra en la detección del



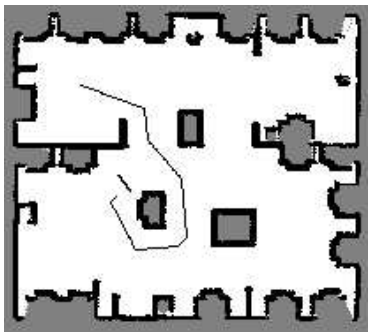
(a) Ambiente y posición inicial del robot.



(b) Ruta inicial trazada por el planificador.

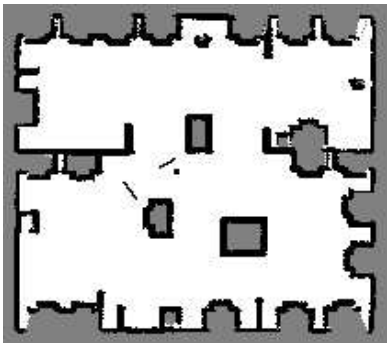


(c) Obstáculo inesperado encontrado durante el seguimiento de una trayectoria.

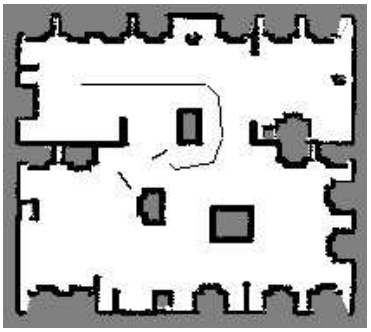


(d) Trayectoria alterna planeada de acuerdo con el mapa alterado.

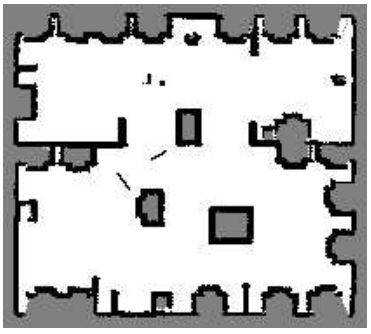
Figura 6.7: Primer paso: El robot traza una trayectoria inicial y altera el mapa del ambiente al encontrar un obstáculo imprevisto para determinar una trayectoria alterna.



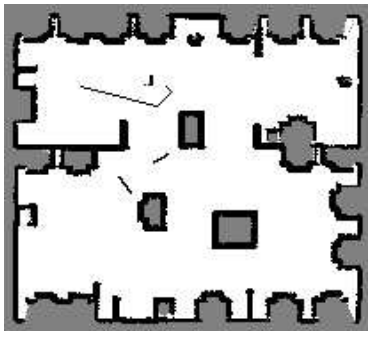
(a) Obstáculo inesperado encontrado durante el seguimiento de una trayectoria.



(b) Trayectoria alterna planeada de acuerdo con el mapa alterado.



(c) Obstáculo inesperado encontrado durante el seguimiento de una trayectoria.



(d) Trayectoria alterna planeada de acuerdo con el mapa alterado.

Figura 6.8: Segundo paso: El robot incluye sucesivamente los obstáculos que va encontrando en su camino y construye en cada caso una ruta alterna.

obstáculo ya que una vez detectado, la nueva trayectoria a seguir estará determinada por el mismo algoritmo de planeación de trayectorias descrito en el Capítulo 3. En lo que se refiere a los obstáculos, la gran mayoría son detectados con precisión solamente con el sensor láser. La razón que justifica el uso de los sonares está relacionada con las superficies de cristal (paredes ó puertas) ya que estas no son detectadas por el láser debido a que se basa en reflexión de la luz.

Por su parte el sonar se basa en la reflexión del sonido. En principio, esto haría suponer que la detección de cristales no debe presentar ningún problema. En la práctica la suposición no siempre se cumple. Si bien es cierto que el sonar emite sonido que es reflejado por las superficies de cristal también es cierto que tiene el problema de la reflexión especular. Este problema se presenta en superficies muy lisas, como es justamente el caso de los cristales. En este tipo de superficies el haz de sonido rebota de tal manera que el receptor ultrasónico no registra la señal devuelta o la registra después de haber rebotado varias veces, obteniendo medidas mayores a las reales. En el Capítulo 8 se muestran varios experimentos realizados a este respecto.

Capítulo 7

Localización global

La capacidad de un robot para localizarse por si mismo dentro de un ambiente, no es solo un problema fundamental dentro del campo de la robótica móvil, también es un requisito para la realización de diversas tareas como la navegación. En este Capítulo se introduce un algoritmo de localización global basado en marcas naturales para un robot móvil en ambientes interiores.

Para resolver el problema de localización global, la idea general consiste en el pre procesamiento de un mapa de celdas previamente construido, para obtener un modelo del conjunto de marcas y sus atributos, que son visibles desde cada celda. El problema de localización global se reduce entonces a encontrar una correspondencia entre la información asociada a una celda y las marcas observadas por el robot desde su posición. Esta búsqueda se realiza en dos fases. En la primera fase se emplea un filtro inicial, que permite determinar cuales son las celdas del mapa que probablemente correspondan con las observaciones del robot, durante esta fase son eliminadas un gran número de celdas que no poseen información similar a la observada por el robot. En la segunda fase se toman las celdas más probables obtenidas de la fase 1 y se hace una búsqueda más detallada de la correspondencia para eliminar nuevamente, a las celdas que no se asemejen a las observaciones del robot. Este algoritmo se enfoca en ambientes interiores tipo oficina aprovechando las marcas naturales que en este tipo de ambientes se encuentran; el algoritmo considera la presencia de nuevos obstáculos y el ruido en los sensores. Para la extracción de marcas naturales se utiliza el mismo algoritmo descrito en el Capítulo 4. Cabe aclarar que en el proceso de localización global solamente se usan las discontinuidades y las paredes, esto se debe a que ambas tienen atributos que

permiten diferenciarlas de otras marcas de su mismo tipo. En el caso de las esquinas, estas no cuentan con ningún atributo que distinga individualmente a una esquina de otras de su mismo tipo por lo que no serán tomadas en cuenta para la localización global.

El resto de este capítulo está organizado de la siguiente manera: Primero se describe en detalle el pre procesamiento del mapa de celdas y las características del modelo del ambiente que es extraído. Enseguida se describe el filtro inicial que permite eliminar las celdas del mapa donde la probabilidad de correspondencia es muy baja. Posteriormente se expone el proceso de búsqueda de correspondencia dentro del espacio reducido de celdas. Por último se obtiene la posición final del robot y la orientación con respecto al mapa. También se muestran algunos ejemplos que ilustran el procedimiento.

7.1. Pre procesamiento del mapa de celdas

Se asume que existe un mapa del ambiente previamente construido por el robot. Cada celda del mapa está asociada con una probabilidad de ocupación. Sin embargo, en la etapa de pre procesamiento, solo se necesita la información de ocupación. De esta manera, el valor de cada celda deberá indicar solamente si la celda corresponde a un obstáculo o es parte del espacio libre. Las celdas cuya ocupación no pueda ser determinada con claridad no serán consideradas debido a que las marcas identificadas en esa región no necesariamente reflejarían la distribución real de los obstáculos en esa zona. El objetivo del pre procesamiento es asociar un conjunto de marcas naturales y sus atributos a cada celda del mapa. En la práctica, resulta imposible colocar al robot real en cada celda para obtener el conjunto de marcas asociadas a ella; en cambio, se hace una simulación de las lecturas del láser por medio de *ray tracing*. En base a estas lecturas simuladas se obtienen las marcas naturales correspondientes y se asocian cada celda. Al final de este proceso, cada celda del mapa esta asociada con el conjunto de marcas, en un rango de 360°, que el robot debería observar desde esa celda.

Para cada celda c_j, se almacena un conjunto M_j de marcas donde para cada marca m_i se almacena (d_i, θ_i, A_i) donde d_i es la distancia entre la celda en cuestión y la marca, θ_i es la orientación a la que se encuentra la marca con respecto a la celda y A_i es el conjunto de atributos asociados a la marca. Dependiendo de si se trata de una discontinuidad o de una pared los atributos serán diferentes. Es importante recalcar que

el atributo θ es relativo al sistema de coordenadas del mapa, una vez que la posición del robot es determinada, el atributo θ será usado para calcular la orientación del robot con respecto al mapa.

Como es de esperarse, el tamaño de la celda impacta directamente en la precisión del modelo. Entre más pequeña sea la celda, mayor será la precisión de la localización, pero también será mayor el espacio de memoria y el tiempo de procesamiento requeridos.

7.2. Proceso de localización global

El objetivo del método de localización global propuesto es determinar la posición y la orientación del robot dentro del mapa utilizando para ello el modelo construido. El primer paso consiste en extraer las discontinuidades que el robot observa a 360°. El sensor láser (SICK LMS 200) con que cuenta el robot tiene un rango de visión de solo 180°, por lo que el conjunto total de marcas se obtiene en tres pasos:

1. Se obtiene el conjunto de marcas M_1 visibles desde la posición y orientación en la que se encuentre el robot.
2. El robot gira 180° sobre su propio eje.
3. Se obtiene el conjunto de marcas M_2.

El conjunto final de marcas observadas M_o es la unión de M_1 y M_2 como se muestra en la Figura 7.1. De ahora en adelante $M_o = (m_{o1}, m_{o2}, ..., m_{ok})$ se referirá al conjunto de k marcas observadas por el robot y $M_m = (m_{m1}, m_{m2}, ..., m_{mj})$ se referirá al conjunto de j marcas asociadas con una celda del modelo.

La información obtenida en la etapa de pre-procesamiento constituye el modelo del ambiente y servirá para calcular la posición y orientación del robot. Como se muestra en la Figura 7.2 este proceso consiste en encontrar la celda en la que se encuentra el robot en función de la similitud entre las observaciones y el modelo. En muchos casos no será posible encontrar una correspondencia exacta debido a cambios en el ambiente o al ruido inherente a los sensores utilizados. En ambientes reales de oficina, es común

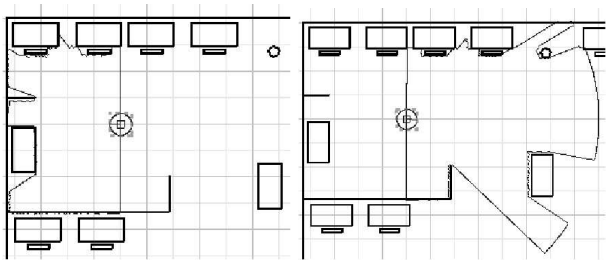


Figura 7.1: El conjunto total de marcas a 360° es obtenido es 3 pasos debido a la limitación en el rango de visión del láser.

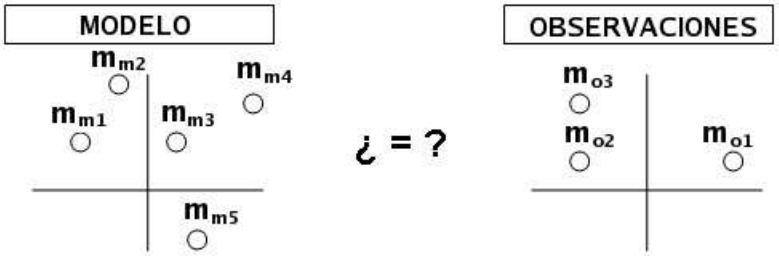
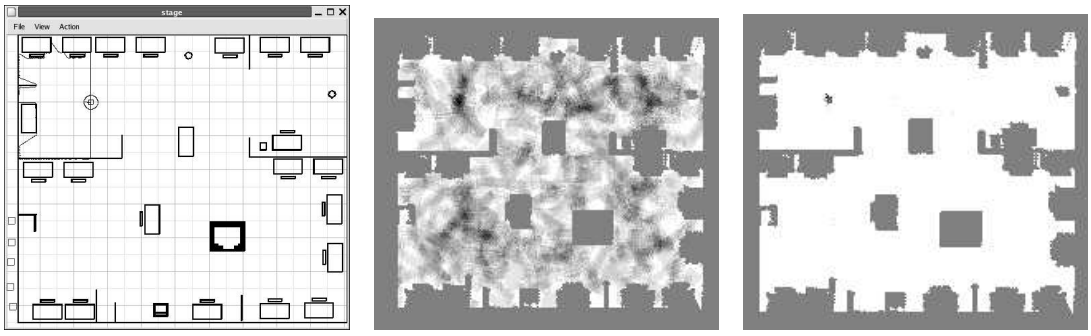


Figura 7.2: El problema de encontrar una correspondencia entre el modelo y las observaciones del robot.

que las personas frecuentemente cambien de lugar objetos como sillas y botes de basura. En la Sección 7.2.2 se presenta un algoritmo de relajación discreta que toma en cuenta esto último al establecer la correspondencia entre las observaciones y el modelo.

Determinar en detalle el grado de correspondencia entre las observaciones y cada una de las celdas del modelo podría resultar prohibitivo en tiempo de procesamiento y recursos. Para agilizar este proceso, se realiza un procedimiento en dos fases. En la primera se utiliza un filtro inicial para determinar cuales son las celdas que presentan mayor evidencia de corresponder con las observaciones del robot. En la segunda se lleva a cabo un proceso más detallado para encontrar la correspondencia correcta solamente en aquellas celdas que hayan pasado la primera etapa. Cada una de estas etapas es descrita a continuación.



(a) Ambiente simulado.

(b) Después de aplicar el filtro inicial.

(c) Después del algoritmo de relajación discreta.

Figura 7.3: La ubicación del robot se determina en dos etapas.

7.2.1. Filtro inicial

El filtro inicial cuenta el número de marcas observadas que son similares a las marcas que corresponden a las celdas del modelo. Para ello utiliza los atributos distintivos que le son asignados a cada marca. En esta etapa no interesa de que forma están distribuidas en el espacio, ni la relación entre ellas, solamente importa si existe una marca con las mismas características que la observada por el robot. Obviamente la comparación de las distancias se hace tomando en cuenta cierta tolerancia debido a que el mapa que se utilizó es construido por el robot y por lo tanto puede tener imprecisiones. Para cada celda c_j, el número de marcas con características similares a las observadas es dividido entre N_j, donde N_j es el total de discontinuidades asociadas en el modelo a la celda c_j. En otras palabras, se obtiene el porcentaje del total de marcas de la celda c_j que corresponden con las observadas por el robot. La Figura 7.3(b) muestra el resultado de aplicar este filtro inicial al mapa, dadas las observaciones del robot. Las regiones más obscuras representan los valores más grandes obtenidos, es decir, las celdas más probables en donde podría encontrarse el robot.

7.2.2. Algoritmo de relajación discreta

En la segunda fase, sólo son tomadas en cuenta las celdas con un porcentaje mayor a la mitad del porcentaje más alto. Para determinar la similitud entre las observaciones y dichas celdas se utiliza un algoritmo de relajación discreta. Este algoritmo puede ser altamente ineficiente si todas las asignaciones son posibles inicialmente. Sin embargo, usando la información de los atributos y el tipo de marca, se puede reducir considerablemente el número de correspondencias posibles para cada observación.

Un ejemplo de lo anterior se muestra en la Tabla 7.2(a): las asignaciones iniciales posibles son marcadas con 0, mientras que el resto no son consideradas en el proceso. Esto último reporta un gran beneficio debido a que en la práctica es muy difícil encontrar dos marcas a la misma distancia del robot, del mismo tipo y con los mismo atributos. Para obtener una medida de la similitud más precisa en el algoritmo de relajación discreta se toma en cuenta la distancia Euclidiana entre marcas y la separación angular entre ellas. En el Algoritmo 7.1 se presenta el pseudo código del procedimiento descrito. La idea básica del algoritmo consiste en verificar que si la marca $m_{oi} \in M_o$ corresponde a la marca $m_{mi} \in M_m$ entonces m_{oi} guarda la misma relación de distancia y ángulo con las marcas en M_o que m_{mi} con las marcas en M_m. Después de haber aplicado el algoritmo el valor máximo de la i-ésima línea en la matriz A tiene el número de relaciones satisfechas para la marca observada m_{oi}.

En la Tabla 7.2(b) el máximo de cada línea determina la correspondencia entre las marcas del conjunto M_o y el conjunto M_m. Obviamente el máximo de cada línea puede ser a lo mas $i-1$ para i observaciones. En la Tabla 7.1 $R(P_i, P_j)$ denota la relación entre dos marcas, esta relación se cumple si la distancia Euclidiana y la separación angular son similares, el operador $\approx$ se usa debido a que las medidas no pueden ser exactamente iguales por la imprecisión de los sensores y el mapa, en la práctica se usa un umbral definido como μ_s para determinar la similitud. Una descripción más amplia del algoritmo básico de relajación discreta puede ser consultado en [Stockman and Goshtasby, 2004].

En la Figura 7.3(c) se muestra el resultado de aplicar el algoritmo de relajación discreta al mapa de la Figura 7.3(b). El algoritmo de relajación discreta no necesita encontrar una correspondencia exacta con cada observación, si existen algunas marcas que no se encuentren en el modelo simplemente se tomarán en cuenta aquellas que cumplan con las relaciones geométricas. Después del algoritmo de relajación discreta sólo restan unas pocas celdas posibles concentradas en una región. Es claro suponer que

Tabla 7.1: Ejemplo de la aplicación del algoritmo de relajación discreta.

	m_{m1}	m_{m2}	m_{m3}	m_{m4}	m_{m5}
m_{o1}	-	-	-	0	0
m_{o2}	0	0	0	-	-
m_{o3}	-	0	0	-	-

(a) Posibles asignaciones iniciales.

	m_{m1}	m_{m2}	m_{m3}	m_{m4}	m_{m5}
m_{o1}	-	-	-	0	2
m_{o2}	2	0	1	-	-
m_{o3}	-	2	1	-	-

(b) Después de aplicar el algoritmo de relajación discreta.

Algoritmo 7.1 Algoritmo de relajación discreta.

Sea $P_i, i = 1 \ldots N$ el conjunto de marcas observadas
Sea $S(P_i), i = 1 \ldots M$ el conjunto de posibles asignaciones iniciales
Sea $A[N][M]$ una matriz de contadores
procedure *Relajación_discreta*(P,S) {
repetir {
 Para cada$(P_i, S(P_i))$ hacer {
  Para cada marca Lk en $S(P_i)$
   Para cada relación $R(P_i, P_j)$ en las observaciones
    Si $\exists Lm \in S(P_j)$ en el modelo con $R(Lk, Lm) \approx R(P_i, P_j)$
    entonces $A[Pi][S(Pi)] = A[Pi][S(Pi)] + 1$
  }
 } hasta $i = N$;
}

todas estas celdas tienen el mismo porcentaje de marcas observadas que corresponden con el modelo de cada celda. Para escoger sólo una de ellas, se utiliza un criterio de mínimos cuadrados, el cual se describe a continuación.

7.2.3. Criterio basado en mínimos cuadrados

Después de aplicar el algoritmo de relajación discreta, sólo resta un conjunto reducido de celdas, todas ellas tienen el mismo porcentaje de asignaciones consistentes. Para determinar cuál de ellas corresponde con la posición del robot se emplea un criterio basado en mínimos cuadrados. Para este propósito se utiliza el atributo D de las marcas (distancia entre el robot y la marca). El mejor ajuste de las observaciones M_o con el modelo se obtiene mediante el cuadrado de la diferencia del valor del atributo D en cada asignación válida. En la Ecuación 7.1, N es el número de asignaciones entre las marcas observadas y el modelo, $E(c_j)$ es el error de ajuste de la celda c_j; d_{oi} y d_{mi} son el atributo D de las marcas m_{oi} y m_{mi} respectivamente.

$$E(c_j) = \sum_{i=1}^{N} (d_{oi} - d_{mi})^2 \tag{7.1}$$

De esta forma la celda c_k que corresponde con la posición del robot está dada por:

$$c_k = argmin(E(c_j)) \tag{7.2}$$

7.2.4. Cálculo de la orientación del robot

Por último, calcular la orientación del robot, una vez que se conoce su posición es relativamente simple. Sean m_{oi} una marca en el conjunto de las observaciones y m_{mi} una marca del modelo correspondiente a la celda de la posición final. Para cada correspondencia (m_{mi}, m_{oi}), se calcula la diferencia aritmética $\hat{\theta}_i$ entre el atributo θ_{oi} de m_{oi} y el atributo θ_{mi} de m_{mi}. Hay que recordar que θ_{mi} es la orientación de una marca relativa al sistema de coordenadas del mapa y θ_{oi} es la orientación a la que se encuentra

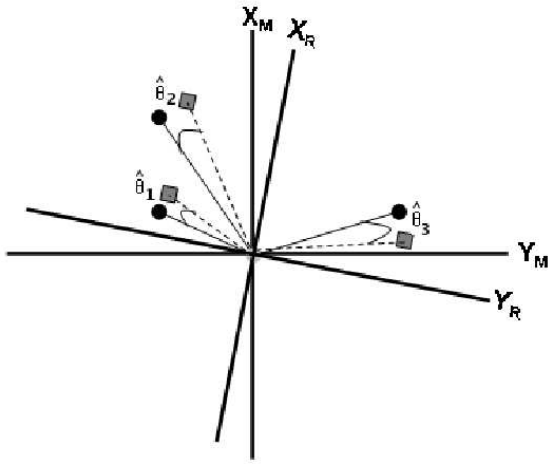


Figura 7.4: Orientación del robot relativa al mapa en base a la diferencia en la orientación de las marcas observadas con las del modelo.

la marca con respecto a la orientación del robot. La orientación del robot con respecto al mapa es el promedio de todas las diferencias $\hat{\theta}_i$ calculadas.

En la Figura 7.4, $\hat{\theta}_i$ es la *i*-ésima diferencia entre la orientación de la marca en las observaciones y el modelo, (X_M, Y_M) son los ejes del sistema de coordenadas del modelo y (X_R, Y_R) son los ejes del sistema de coordenadas del robot. Pequeños círculos y cuadrados representan a las marcas m_{mi} y m_{oi} respectivamente. En la ecuación 7.3, N es el número de asignaciones entre las marcas observadas y las marcas de la celda que corresponde a la ubicación del robot, θ_{oi} y θ_{mi} son el atributo θ de m_{oi} y m_{mi} respectivamente..

$$\theta_R = \frac{\sum_{i=1}^{N} \hat{\theta}_i}{N} \tag{7.3}$$

Donde:

$$\hat{\theta}_i = \theta_{oi} - \theta_{mi} \tag{7.4}$$

7.3. Comentarios finales

En este capítulo de presentó un algoritmo de localización global basado en marcas naturales del ambiente. Para ello se construye un modelo de ambiente conformado por las marcas naturales asociadas a cada celda. La idea básica consiste en determinar la posición del robot estableciendo una correspondencia entre las marcas que el robot observa y el modelo.

Las principales ventajas del enfoque propuesto son las siguientes:

- Se presenta un algoritmo modificado de relajación discreta para la solución del del problema de correspondencia, la solución se va construyendo de acuerdo con las marcas que cumplan con las restricciones geométricas por pares, esto permite obtener una solución incluso en los casos en que existen marcas que el robot observa y que no están en el modelo (nuevos objetos), o cuando existen marcas en el modelo que el robot no observa (objetos faltantes).
- Debido a la combinación de discontinuidades y paredes, regularmente se dispone de un buen número de marcas para realizar la localización, en contraste, se requieren al menos 2 marcas que correspondan con el modelo para realizar la localización.

Las principales limitaciones son las siguientes:

- Debido a que el enfoque es basado en celdas, la complejidad computacional del filtro inicial aumenta con el número de celdas. En los experimentos reportados en el Capítulo 8 se observa este inconveniente. Para evitar este problema, la reducción inicial del espacio de búsqueda podría hacerse en términos de un clasificador que determine directamente la clase de celda en la que se encuentra el robot, un enfoque con estas características es presentado en [Romero, 2001].
- En el caso de que existieran dos lugares dentro del ambiente exactamente iguales, este algoritmo sería incapaz de determinar en cual de ellos se encuentra. Sin embargo, cabe mencionar que en ambientes reales de oficina esta condición es muy poco probable. Aunque en muchas ocasiones los espacios son similares, el arreglo del mobiliario y demás objetos frecuentemente establecen una diferencia.

Capítulo 8

Experimentos realizados

En este capítulo se hará un resumen de los principales experimentos realizados con el sistema de navegación y localización. Se realizaron experimentos tanto en simulación como en el robot real. Para los ambientes simulados se utilizó el simulador Stage del proyecto de software libre Player/Stage [Gerkey et al., 2003]. Para los experimentos en un robot real se utilizó uno con las características descritas en la Sección 1.3.3 del Capítulo 1. Los experimentos reales se llevaron a cabo en 2 espacios diferentes: el Laboratorio de Sistemas Inteligentes y el Departamento de Computación del ITESM Campus Cuernavaca, los cuales presentaron diferentes retos y características. Los dos tipos de robot, el simulado y el real cuentan con un odómetro, un anillo de sonares, y un sensor láser con un rango de 180°. El robot tiene forma redonda y puede girar sobre si mismo y avanzar.

8.1. Condiciones de prueba

Los experimentos se dividen en 4 grupos: localización global, planeación de trayectorias, seguimiento de la posición y detección de obstáculos. Dentro de cada grupo se realizaron pruebas específicas para observar el desempeño de cada uno de los componentes del sistema de navegación y localización. Enseguida se indica brevemente la forma en que se llevaron a cabo estas pruebas, así como el objetivo buscado.

- Localización global
  - Pruebas con el simulador: Se consideraron 3 mapas diferentes y se realizó 30 veces el procedimiento de localización global en cada uno. Se colocó al robot simulado en lugares diferentes aleatoriamente. Se evaluaron: el tiempo de procesamiento del filtro inicial, el número de celdas que son procesadas en la segunda etapa, el tiempo de procesamiento del algoritmo de relajación discreta, el error en la posición y orientación y el espacio de memoria requerido para almacenar el modelo de cada mapa.
  - Pruebas reales: Se consideraron dos ambientes reales diferentes: el Laboratorio de Sistemas Inteligentes y el Departamento de Computación. En cada uno se realizó 10 veces el proceso de localización global reportando los mismos aspectos del punto anterior.
- Planeación de trayectorias.
  - Se utilizaron 3 diferentes mapas construidos en un editor de imágenes y los 2 mapas de los ambientes reales obtenidos por el robot. Se realizaron pruebas de planeación en cada ambiente, tratando de que éstas fueran largas y complejas.
- Seguimiento de la posición.
  - Pruebas con el simulador: Se realizaron 30 recorridos en 3 diferentes ambientes simulados. Durante el recorrido del robot, existieron objetos moviéndose frente a él. Se reporta la longitud de la trayectoria, el error final en la posición y orientación (tomando como referencia al odómetro perfecto del simulador). Se introdujeron diferentes cantidades de ruido a las lecturas de los sensores.
  - Pruebas reales: Se realizaron 10 pruebas en cada uno de los 2 ambientes reales antes mencionados. Una persona se movía frente al robot mientras navegaba. Se reporta el error final en la posición y orientación (estimado).
- Detección de obstáculos.
  - Pruebas reales: En virtud de que se utilizan dos tipos de sensores para la detección de obstáculos se hicieron 20 pruebas con obstáculos reales en los dos ambientes antes mencionados. Se le indicó una meta al robot y se le colocaron diversos objetos en el camino (cajas, sillas, puertas de cristal y personas).

8.2. Ambientes utilizados

Para una mejor interpretación de los experimentos realizados y las condiciones establecidas a continuación se muestran los mapas utilizados.

8.2.1. Ambientes simulados

Como se mencionó se construyeron diferentes ambientes para las pruebas de simulación. El simulador Stage proporciona la facilidad de definir un mapa a través de una imagen que indique con diferentes colores el espacio libre y los obstáculos. Los ambientes que se muestran en las Figuras 8.1, 8.2 y 8.3 son los 3 mapas utilizados para las pruebas. La Figura 8.1 es un ambiente de aproximadamente 5×5 metros, el de la Figura 8.2 es de aproximadamente 10×10 metros y el de la Figura 8.3 es de aproximadamente 20×20 metros. Los ambientes mostrados corresponden a espacios típicas de oficinas o cubículos.

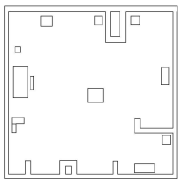


Figura 8.1: Ambiente de aproximadamente 5×5 metros.

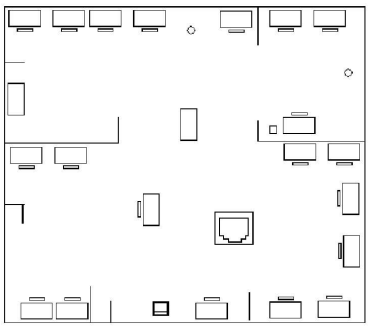


Figura 8.2: Ambiente de aproximadamente 10×10 metros.

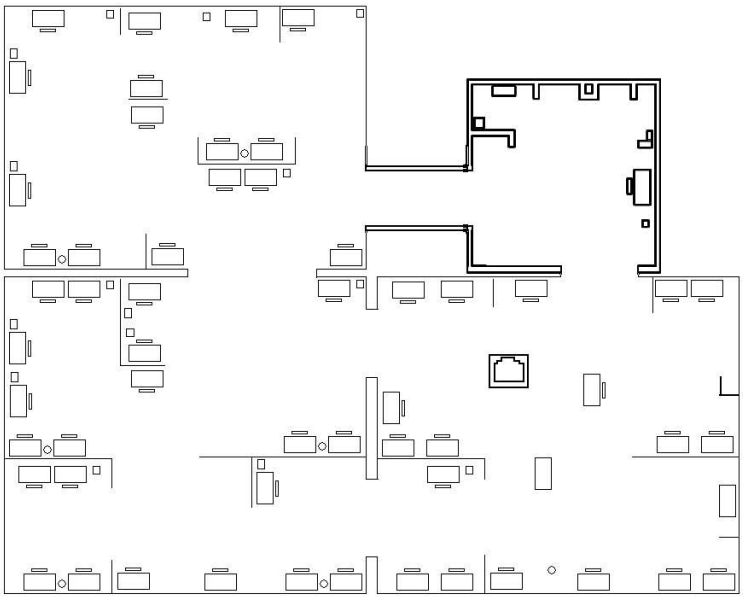


Figura 8.3: Ambiente de aproximadamente 20×20 metros.

8.2.2. Ambientes reales

Como se mencionó los ambientes reales son el Laboratorio de Sistemas Inteligentes y el Departamento Computación del ITESM Campus Cuernavaca. Los mapas de los ambientes reales usados son proporcionados por un módulo de construcción de mapas. Este módulo es independiente del presente trabajo aunque está directamente relacionado ya que el mapa que se construye es utilizado por el módulo de navegación y localización para realizar sus tareas. El detalle de la construcción de mapas puede encontrarse en [Jaquez, 2005]. En las Figuras 8.4 y 8.5 pueden verse los mapas de los ambientes reales considerados. En la Figura 8.4 puede verse el mapa del Laboratorio de Sistemas Inteligentes del ITESM Campus Cuernavaca. El color blanco representa el espacio libre, el negro los obstáculos y el gris las celdas cuya ocupación no se pudo determinar. En la Figura 8.5 se muestra un mapa similar del Departamento de Computación del ITESM Campus Cuernavaca. En el caso del Departamento de Computación el ambiente es especialmente difícil ya que las puertas de los cubículos son de cristal transparente. Además el piso no es completamente plano, existen pequeñas hendiduras entre un mosaico y otro, que afectan el movimiento del robot y en consecuencia hacen más difícil el problema de localización. El tamaño de las celdas en ambos mapas es de 5 centímetros cada una.

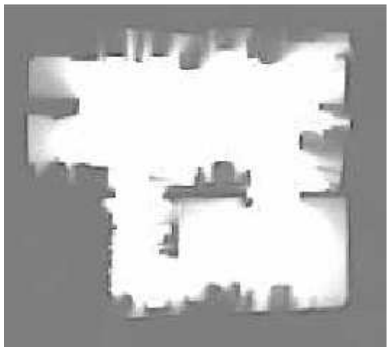


Figura 8.4: Mapa del Laboratorio de Sistemas Inteligentes de aproximadamente 8×8 metros.

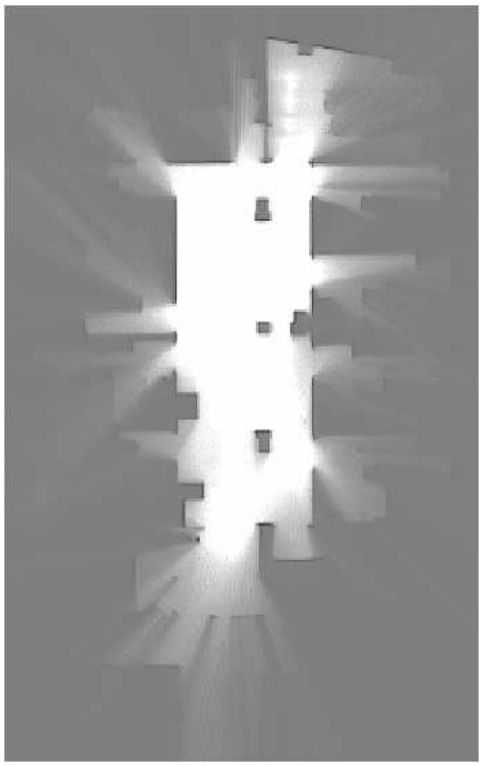


Figura 8.5: Mapa del Departamento de Computación de aproximadamente 15×8 metros.

8.3. Localización global

8.3.1. Experimentos simulados

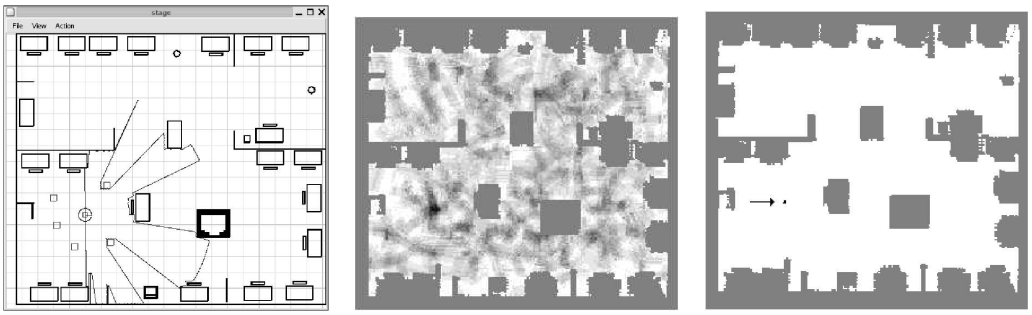
En el caso de la localización global se introdujo ruido en los sensores con la finalidad de hacer que las pruebas fueran más útiles para fines de evaluación. El ruido se distribuyó de manera uniforme sobre el 20 % de las marcas en promedio. La magnitud del ruido introducido fue de $\pm 2cm.$ sobre la distancia medida entre la marca y el robot (atributo D) y de $\pm 5^{\circ}$ sobre el atributo θ que es la orientación a la que se encuentra la marca con respecto al robot.

Para los experimentos, tanto simulados como reales, el umbral μ_s que determina la similitud se fijó en $5cm$. Los resultados se muestran en la Tabla 8.1. La localización se considera exitosa si el algoritmo arroja un resultado dentro de un radio de $10cm$ alrededor de la posición real del robot. Esto es, si cada celda del mapa es de $5 \times 5cm$ la localización se considera exitosa si el resultado es la celda en la que realmente se encuentra el robot o cualquiera de sus celdas vecinas.

Tabla 8.1: Promedio de los resultados de 30 pruebas del algoritmo de localización global sobre diferentes mapas.

Mapa	Número de celdas	Tiempo etapa 1 (seg.)	No. de celdas etapa 2	Tiempo etapa 2 (seg.)	Memoria (Mb)	Loc. exitosa (%)
Fig 8.1	106x106	0.176	297.1	0.130	1.9	100
Fig 8.2	240x220	0.509	790.8	0.164	3.93	100
Fig 8.3	480x580	1.7	2,246.5	0.3	18.66	100

En 5 de los 30 experimentos de cada mapa se agregaron obstáculos adicionales al ambiente. Estos obstáculos fueron introducidos o movidos de su lugar después de que el mapa del ambiente fue construido. La localización no se vio afectada en ninguno de los casos en que se introdujeron nuevos objetos. En la Figura 8.6 puede verse un ejemplo de la agregación de obstáculos durante el proceso de localización global.



(a) Obstáculos adicionales añadidos (cuadrados cercanos al robot).

(b) Celdas más probables después del filtro inicial

(c) Posición final encontrada

Figura 8.6: Experimento con nuevos obstáculos agregados al ambiente

8.3.2. Experimentos reales

En el caso de los experimentos en los ambientes reales el ruido no necesitó ser introducido. El robot fue colocado en 10 posiciones diferentes y se realizó el proceso de localización, los resultados se muestran en la Tabla 8.2.

Tabla 8.2: Promedio de los resultados de 10 pruebas del algoritmo de localización global en ambientes reales.

Mapa	Número de celdas	Tiempo etapa 1 (seg.)	No. de celdas etapa 2	Tiempo etapa 2 (seg.)	Memoria (Mb)	Loc. exitosa (%)
Departamento	250x400	0.645	567.3	0.279	14.93	90
Laboratorio	190x170	0.398	1120.2	0.180	7.3	100

En 5 de los 10 experimentos de cada mapa se agregaron obstáculos adicionales al ambiente. Estos obstáculos fueron introducidos o movidos de su lugar después de que el mapa del ambiente fue construido. En el caso del Departamento de Computación se obtuvo una efectividad del 90 %. Esto quiere decir que en un caso la localización no fue exitosa.

En el caso en que la localización no fue exitosa se debió a que el robot fue colocado en un lugar donde se encontraba rodeado de áreas de celdas no determinadas.

Específicamente fue colocado dentro de la sala de juntas que es la habitación que se encuentra en la parte superior derecha dentro del mapa que se muestra en la Figura 8.5. En la etapa de pre-procesamiento del mapa descrita en la Sección 7.1 las celdas no determinadas no se toman en cuenta a la hora de identificar el conjunto de marcas que el robot debería ver (modelo), por lo que estas zonas no poseen información (marcas) para localización. La única información que el robot detectó fueron las discontinuidades ocasionadas por el marco de la puerta desde el interior de la habitación. Con sólo esta información el robot no pudo determinar dentro de cual de las habitaciones se encontraba, ya que en todas, las puertas son del mismo tamaño.

Otro factor que no se ve reflejado en las tablas, pero que afecta fuertemente al proceso de localización es cuando se colocan obstáculos muy cerca del robot, esto limita en gran medida el rango de visión del sensor láser y en consecuencia el robot no identifica un gran número de marcas. Este problema se ve atenuado debido a que el robot da media vuelta para obtener las lecturas a 360°.

8.4. Planeación de trayectorias

Para probar el planificador de trayectorias se tomó cada uno de los mapas: los 3 construidos a mano y los 2 obtenidos de los ambientes reales. Enseguida se colocó al robot en un extremo del ambiente y se marcó como la meta un punto lejano dentro del ambiente. La intención es que la ruta construida sea larga y compleja con la finalidad de observar los resultados. En las Figuras 8.7, 8.8, 8.9, 8.10, 8.11 pueden observarse los experimentos realizados con cada uno de los mapas.

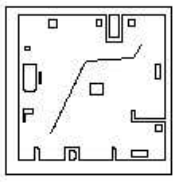


Figura 8.7: Experimento con el mapa simulado de 5×5 m.

Como se mencionó en la Sección 3.2.3 existen dos parámetros se le deben proporcionar al planificador de trayectorias: la distancia deseada entre el robot y el obstáculo más próximo (D_{pref}) y el segundo es la mínima distancia segura entre el robot y el

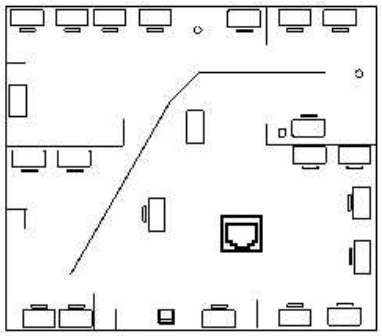


Figura 8.8: Experimento con el mapa simulado de 10×10 m.

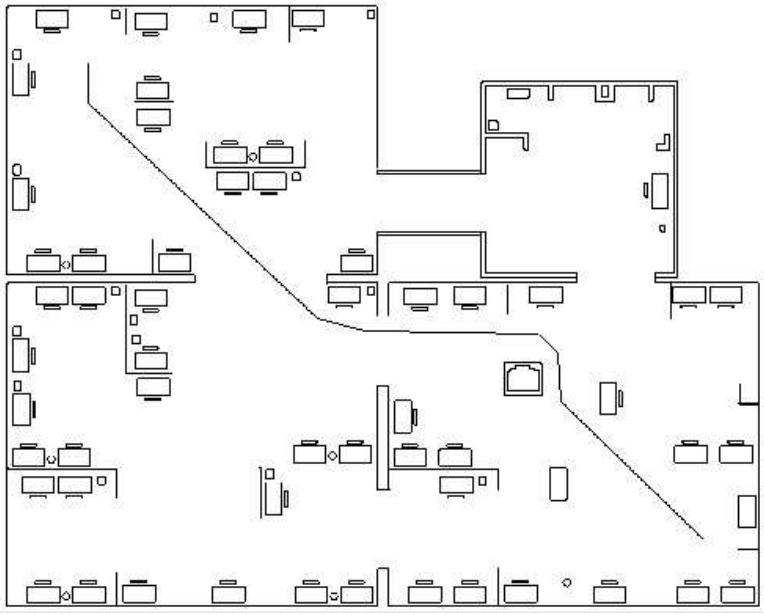


Figura 8.9: Experimento con el mapa simulado de 20×20 m.

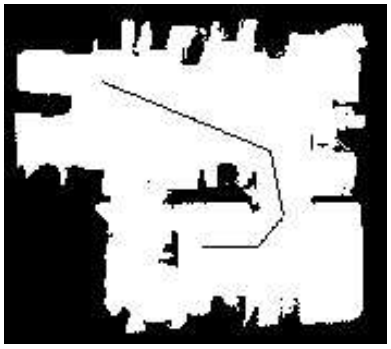


Figura 8.10: Experimento de planeación de trayectorias con el mapa del Laboratorio de Sistemas Inteligentes.

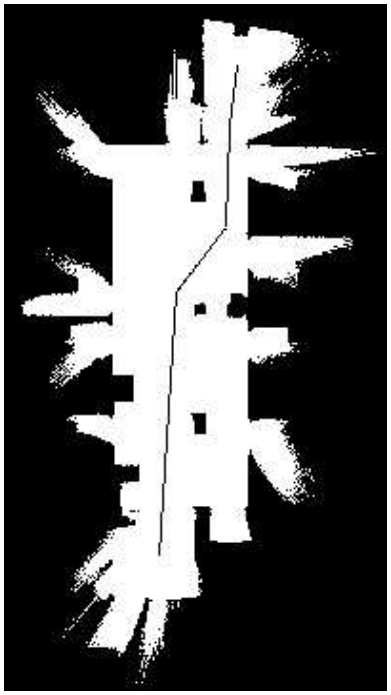


Figura 8.11: Experimento de planeación de trayectorias con el mapa del Departamento de Computación.

obstáculo más próximo (d_{min}). La distancia deseada D_{pref} entre el robot y el obstáculo más próximo es usada por el planificador para preferir trayectorias a esa distancia de los obstáculos, mientras que la distancia mínima d_{min} es utilizada para agrandar los obstáculos de tal manera que el robot jamás elija moverse hacia una celda por debajo del valor de d_{min}. En el caso de los experimentos realizados se utilizaron los valores de $30cm.$ y $45cm.$ para los parámetros d_{min} y D_{pref} respectivamente. En los casos como el del Departamento de Computación hay que ser cuidadoso al seleccionar estos parámetros, ya que si el valor de d_{min} es muy grande, las entradas a las habitaciones podrían quedar bloqueadas debido al agrandamiento de los obstáculos. En la práctica el valor de d_{min} deberá ser menor a la mitad del pasaje más reducido por el que el robot tenga que pasar. En el Apéndice A se presentan más experimentos de planificación de trayectorias.

8.5. Seguimiento de la posición

8.5.1. Experimentos simulados

Las pruebas relacionadas con el seguimiento de la posición se realizaron de la siguiente manera: en ambientes simulados se hicieron 30 recorridos de diversas longitudes. Además se introdujo ruido en los sensores con la finalidad de hacer que las pruebas fueran más útiles para fines de evaluación. El ruido se distribuyó uniformemente sobre el 20 % de las marcas en promedio. La magnitud del ruido introducido fue de $\pm 2cm.$ sobre la distancia medida entre la marca y el robot (atributo D) y de $\pm 5^{\circ}$ sobre el atributo θ que es la orientación a la que se encuentra la marca con respecto al robot. Los umbrales para la identificación de marcas se definieron tanto para experimentos simulados como para experimentos reales de la siguiente manera: $\mu_d = 20cm.$, $\mu_e = 15cm.$ y $\mu_p = 30cm.$. Tanto en los experimentos simulados como en los reales el seguimiento de la posición se realiza a intervalos constantes de tiempo considerando un Δt de 4 segundos. El esquema de velocidades utilizado, de acuerdo con lo expuesto en el Capítulo 5 es el siguiente: $velL_{min}$ = 0.05m/s, $velL_{med}$ = 0.1m/s, $velL_{max}$ = 0.2m/s, $velA_{min}$ = 2.8grados/s, $velA_{med}$ = 5.7grados/s y $velA_{max}$ = 8.5grados/s. Los resultados se muestran en la Tabla 8.3.

Tabla 8.3: Promedio de los resultados de 30 recorridos haciendo seguimiento de la posición en diferentes ambientes simulados.

Mapa	Número de celdas	Longitud de la trayectoria (m)	Error final en x (cm)	Error final en y (cm)	Error final de orientación (grados)
Fig 8.1	106x106	3.8	4.9	4.1	0.9
Fig 8.2	240x220	7.6	5.8	6.0	1.3
Fig 8.3	480x580	18.5	8.2	7.9	1.8

8.5.2. Experimentos reales

Para las pruebas en ambientes reales se hicieron 10 recorridos en cada uno de los 2 ambientes reales. Mientras el robot se movía una persona se movía también dentro del rango de visión del láser. Esto último no afectó al proceso de seguimiento de la posición por lo que se asume que las marcas detectadas a causa del objeto en movimiento fueron descartadas adecuadamente al revisar la consistencia geométrica.

El error en la posición (x,y) y orientación finales no puede ser calculado con exactitud debido a que para ello habría que medir físicamente para determinar el lugar donde se encuentra realmente el robot. Por otro lado aunque esta medida se obtuviese, restaría el problema de encontrar el lugar dentro del mapa al que corresponde la posición del robot para medir el error en la posición. Debido a lo anterior, el error final en x, y y la orientación se reporta como una medida aproximada, obtenida de la medición física de la distancia entre el robot y los obstáculos más cercanos, comparándolas con el mapa. Se agrega también un campo que indica el porcentaje de ocasiones en que el robot fue capaz de llegar al punto indicado como la meta. En la Tabla 8.4 se muestran los resultados.

Tabla 8.4: Promedio de los resultados de 10 recorridos haciendo seguimiento de la posición en ambientes reales.

Mapa	Número de celdas	Longitud de la trayectoria (m)	Error final en x,y (cm)	Error final de orientación (grados)	Llegó a la meta (%)
Departamento	250x400	7.8	< 15	< 4	90
Laboratorio	190x170	4.5	< 5	< 2	100

El error final en la orientación también se refiere a una aproximación la cual

fue medida en base al trazado del mosaico del suelo. En el caso del Departamento de Computación se presentó un problema muy notorio.

El problema se relaciona con la superficie del piso, la dificultad radica en que existe una pequeña hendidura entre un mosaico y otro. Esta hendidura tiene aproximadamente un ancho de 1.5 cm. Las dos ruedas de tracción que se encuentran a los costados del robot no se ven afectadas por estos pequeños canales, sin embargo, el robot posee una pequeña rueda de soporte en la parte trasera la cual si se ve afectada.

La razón es que su circunferencia es mucho menor que la de las ruedas laterales. En el momento en que esta rueda de soporte pasa por una de las hendiduras el robot completo se balancea de un lado a otro ligeramente. Si en ese momento casualmente se hace el proceso de seguimiento de la posición, el láser registra lecturas que no corresponden exactamente con la posición del robot y que están afectadas por el balanceo del robot. Esto introduce imprecisiones que ocasiona que el error final en la posición (x,y) sea mayor en el caso de las trayectorias recorridas en el Departamento de Computación.

Incluso hubo un caso en el que el error en el seguimiento de la posición ocasionado por las irregularidades del suelo ocasionó que el robot realizara una estimación errónea de su posición, lo que le llevo a detectar como obstáculo el marco de una puerta, cuando la trayectoria generada indicaba pasar por el centro de esta; a eso se debe que el robot llegó el 90 % de veces a la meta sin perder su posición.

8.6. Detección de obstáculos

Para verificar que los obstáculos sean detectados adecuadamente, se realizaron 30 pruebas con diferentes tipos de obstáculos en ambientes reales. Hay que recordar que debe definirse una distancia crítica M de cercanía a los obstáculos. En los experimentos se considera una distancia $M = 30cm$.

Los problemas que se detectaron fueron los siguientes:

- La altura de los obstáculos: debido a que el sensor láser se mantiene sobre un plano paralelo al piso a una cierta altura, no le es posible detectar los obstáculos a una altura menor o mayor. Específicamente en el caso de algunas mesas, estas tienen un espacio vació por debajo de su superficie. Esto ocasiona que el sensor

láser no detecte el peligro de chocar con el borde superior de la mesa cuando esta se encuentra por encima del plano del sensor láser.

- Lo mismo sucede con obstáculos cuya altura está por debajo de la del sensor láser como por ejemplo las cajas y CPUs que se encuentran en el Laboratorio de Sistemas Inteligentes. El robot es incapaz de detectar el peligro de choque con estos objetos.

- El tercer problema tiene que ver con los obstáculos de cristal como las puertas o paredes de cristal. En este caso, de las 20 pruebas realizadas con puertas y paredes de cristal sólo el 60 % fueron detectadas correctamente. Obviamente el sensor láser no tiene nada que hacer y son los sonares los encargados de realizar la detección. A pesar de que se emplean los tres sonares frontales, hay ocasiones en las que los cristales no son detectados. En los experimentos se observó que debido a que los cristales son superficies extremadamente lisas, el sonido reflejado solamente es detectado de manera correcta por el sonar si este se encuentra sobre la línea imaginaria perpendicular a la superficie de cristal. Esta limitante es física y es consecuencia de la naturaleza de los sensores ultrasónicos que posee el robot y no depende tanto del esquema de fusión sensorial empleado.

Cabe mencionar también que un persona estática que obstruye el camino del robot es detectada correctamente y que las superficies negras no presentaron problemas diferentes de las superficies de otros colores.

Capítulo 9

Conclusiones y trabajo futuro

Esta tesis ha abordado los problemas de localización global y navegación. Este capítulo resalta las ideas más significativas que subyacen detrás de los métodos propuestos así como algunas sugerencias para extenderlo.

9.1. Conclusiones

Durante el proceso de planteamiento y propuesta de solución de un problema como el de navegación o el de localización se desarrollan procedimientos, algoritmos y técnicas que sustentan las ideas generales. Algunas de las ideas más importantes planteadas en este trabajo de tesis son las siguientes:

- La planificación de trayectorias presentada en este trabajo de tesis es una combinación de varias ideas relacionadas. En primer lugar se aprovecha la información del mapa de celdas que se tiene. Se propone un algoritmo para la construcción de un mapa de distancia a los obstáculos por medio de una propagación simple. Este mapa de cercanía a los obstáculos es muy útil debido a que es tan preciso como lo es el mapa. Además proporciona información de cercanía en todo el ambiente y no solo en zonas cercanas a los obstáculos. Posterior a la utilización del clásico algoritmo de programación dinámica se utiliza un procedimiento para refinar la ruta obtenida. Ciertamente se trata de una heurística, sin embargo, compensa en

gran medida las limitaciones impuestas por el esquema de vecindad del enfoque basado en celdas. En la gran mayoría de los experimentos realizados con diversos ambientes se observa una mejora considerable.

- Se propone un esquema para seguimiento de la posición basado en marcas naturales. Este esquema se apoya en algoritmos de identificación de marcas naturales tradicionales con la diferencia de que además de identificarlas les asocia atributos que permitan diferenciarlas de otras de su mismo tipo. Además identifica al conjunto de marcas útiles en base a la consistencia de su distribución espacial. Esto evita que una imprecisión en la identificación de algunas marcas lleve a errores en la estimación de la posición del robot. De manera adicional se establece un mecanismo para etiquetar las estimaciones en función de atributos fácilmente identificables con la finalidad de dar más peso a aquellas estimaciones que muestran evidencia de ser más precisas. Los experimentos realizados con el robot real en ambientes demandantes como el Departamento de Computación corroboró que el esquema de fusión de las estimaciones le permite al robot realizar el seguimiento de su posición de manera rápida y eficiente.

- En lo que se refiere a la localización global se propone la construcción de un modelo del ambiente utilizando para ello el mapa del que se dispone. La idea básica es asociar celdas del mapa con conjuntos de marcas naturales, esto permite eliminar inicialmente las regiones donde no se encuentren marcas similares a las que el robot observa. Posteriormente se usa un algoritmo para encontrar correspondencias aprovechando tanto los atributos de las marcas como la relación espacial entre ellas. La orientación del robot se determina en base a la diferencia entre el sistema de coordenadas local del robot y el sistema de coordenadas del mapa utilizando para ello las marcas correspondiente. En los experimentos realizados en ambientes reales el algoritmo de localización global propuesto funciona adecuadamente y le permite al robot conocer su posición y orientación dentro del ambiente.

De la misma manera, durante el desarrollo de un trabajo de esta naturaleza se visualizan diferentes alternativas y nuevas ideas que pueden ser desarrolladas como trabajo futuro de investigación; algunas de ellas se mencionan a continuación.

9.2. Sugerencias para trabajos futuros

Mejorar el algoritmo de localización global. Particularmente el filtro inicial, ya que como se observó en los experimentos realizados el tiempo de procesamiento de esta etapa se incrementa con el tamaño del ambiente. Además es necesario optimizar el modelo del ambiente construido, ya que actualmente si una marca es vista desde dos celdas diferentes se almacena en ambas con sus respectivos atributos. Si se pudiera identificar que en realidad de trata de la misma marca podría reducirse significativamente el tamaño del modelo.

Mejorar la evasión de obstáculos imprevistos. Dado que en este trabajo sólo se contemplan obstáculos imprevistos estáticos; el siguiente paso consistiría en que el robot sea capaz de evadir obstáculos imprevistos dinámicos.

Mejorar el esquema de fusión sensorial. En este trabajo se toman en cuenta dos tipos de sensores: láser y ultrasónico. La fusión sensorial se basa en obtener el mínimo de ambos sensores en un instante de tiempo. Esta fusión podría mejorarse tomando en cuenta no solamente un instante de tiempo sino las lecturas anteriores, con la finalidad de mejorar la capacidad de los sonares para detectar cristales.

Enriquecer y actualizar el mapa del ambiente. Actualmente el mapa dado que se tiene del ambiente se mantiene sin modificaciones permanentes. Sin embargo mientras el robot navega podría considerarse la posibilidad de actualizar el mapa, añadiendo nuevos obstáculos o eliminando los que ya no estén presentes en la realidad.

Referencias

[Batavia and Nourbakhsh, 2000] Batavia, P. and Nourbakhsh, I. (2000). Path planning for the cye robot. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1:15–20.

[Bergasa et al., 2002] Bergasa, L., Lopez, E., and Barea, R. (2002). Sistemas robóticos para la asistencia a personas mayores. Technical report, Departamento de Electrónica, Universidad de Alcalá.

[Burgard et al., 1999] Burgard, W., Cremers, A., Fox, D., Hähnel, D., Lakemeyer, G., Schulz, D., Steiner, W., and Thrun, S. (1999). Experiences with an interactive museum tour-guide robot. *Artificial Intelligence*, pages 114(1–2):355.

[Costa et al., 1990] Costa, L. D. F., Ben-Tzvi, D., and Sandlert, M. (1990). Performance improvements to the hough transformation. Technical report, King's College, University of London, UK.

[Dixon and Henlich, 1997] Dixon, J. and Henlich, O. (1997). Mobile robot navigation. *Surveys and Presentations in Information Systems Engineering (Surprise 97)*, 4.

[Duda and Hart, 1972] Duda, R. O. and Hart, P. E. (1972). Use of the hough transformation to detect lines and curves in pictures. *Graphics and Image Processing*, 15(1):11–15.

[Dudek and Jenkin, 2000] Dudek, G. and Jenkin, M. (2000). *Computational principles of mobile robotics*. Cambridge University Press, New York, NY, USA.

[Einsele, 2001] Einsele, T. (2001). *Localization in Indoor Environments Using a Panoramic Laser Range Finder*. PhD thesis, Universidad Tecnológica de Munich.

[Fox et al., 1999] Fox, D., Burgard, W., and Thrun, S. (1999). Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, page 11:391.

[Gerkey et al., 2003] Gerkey, B. P., Vaughan, R. T., and Howard, A. (2003). The player/stage project. Technical report, http://playerstage.sourceforge.net.

[Hwang and Ahuja, 1992] Hwang, Y. K. and Ahuja, N. (1992). Gross motion planning, a survey. *ACM Computing Surveys*, 24:219–291.

[Jaquez, 2005] Jaquez, V. (2005). Construcción de mapas y localización simultanea con robots móviles. Master's thesis, ITESM Campus Cuernavaca.

[Jefferies et al., 2001] Jefferies, M. E., Yeap, W. K., Smith, L., and Ferguson (2001). Building a map for robot navigation using a theory of cognitive maps. *International Conference on Artificial Intelligence and Application*, pages 348–253.

[Kelly, 2003] Kelly, A. (2003). Precision dilution in triangulation based mobile robot position estimation. Technical report, Carnegie Mellon University, Pittsburg, PA 15213-3890.

[Lowe et al., 2002] Lowe, D., Little, J., and Se, S. (2002). Global localization using distinctive visual features. *Conference on intelligent robots and systems*, pages 226–231.

[MacKenzie and Dudek., 1994] MacKenzie, P. and Dudek., G. (1994). Precise positioning using model-based maps. *Proc. Int. Conf. Robotics and Automation*, pages 1615–1651.

[Ribeiro and Concalves, 1996] Ribeiro, M. I. and Concalves, J. G. (1996). Natural landmark based localization of mobile robots using laser range data. *Proceedings of Eurobot 96*.

[Ribeiro and Lima, 2002] Ribeiro, M. I. and Lima, P. (2002). Navigation of mobile robots. Technical report, Instituto Superior Técnico de Lisboa, Avenida Rovisco Paris 11049-001, Lisboa, Portugal.

[Romero, 2001] Romero, L. (2001). *Map building and localization for mobile robots: a probabilistic approach*. PhD thesis, ITESM Campus Cuernavaca.

[Roumeliotis and Bekey., 2000] Roumeliotis, S. and Bekey., G. (2000). Bayesian estimation and kalman filtering: A unified framework for mobile robot localization. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2985–2992.

[Santiso et al., 2003] Santiso, E., Mazo, M., Ureña, J., Jiménez, J. A., and García, J. (2003). Extracción de características para posicionamiento absoluto de robots móviles en interiores. Technical report, Universidad de Alcalá.

[Schiele and Crowley, 1994] Schiele, B. and Crowley, J. L. (1994). A comparison of position estimate techniques using ocupancy grids. *International Conference on Robotics and Automation*, pages 1628–1634.

[SICK-LMS200, 2003] SICK-LMS200 (2003). *Product Information: Laser Measurement System LMS 200*. Laser Measurement Technology, Sebastian-Kneipp D-79183 Waldkirch, Germany.

[Stockman and Goshtasby, 2004] Stockman, G. and Goshtasby, A. (2004). 2-d and 3-d image registration:a tutorial. *Computer Vision and Pattern Recognition 2004*. http://www.cse.msu.edu/ stockman/RegTutorial/.

[Thrun et al., 1999] Thrun, S., Bennewitz, M., Burgard, W., Cremers, A., Dellaert, F., Fox, D., Hahnel, D., Rosenberg, C., Roy, N., Schulte, J., and Schulz., D. (1999). Minerva: A second generation museum tour-guide robot. *Proceedings of IEEE International Conference on Robotics and Automation*.

[Thrun et al., 2001] Thrun, S., Fox, D., Bugard, W., and Dellaert, F. (2001). Robust monte carlo localization for mobile robots. *Artificial Intelligence*.

[Tomatis, 2001] Tomatis, N. (2001). *Hybrid, Metric - Topological Robot Navigation*. PhD thesis, Polytechnique Fédérale de Lausanne.

[Wang et al., 2002] Wang, L. C., Yong, D. L. S., and Ang, M. (2002). Mobile robot localization for indoor environment. Technical report, Singapore Institute of Manufacturing Technology.

[Wijk and Christenseny, 2000] Wijk, O. and Christenseny, H. (2000). Extraction of natural landmarks and localization using sonars. *Robotics and Autonomous Systems*, 31:31–42.

[Xavier et al., 2004] Xavier, J., Pacheco, M., Castro, D., and Ruano, A. (2004). Fast line, arc/circle and leg detection from laser scan data in a plater driver. *International Conference on Robotics and Automation*.

[Xiaowei et al., 2000] Xiaowei, Z., Khing, H. Y., Sen, C. C., and Yi, Z. (2000). The localization of mobile robot based on laser scanner. *Canadian Conference on Electrical and Computer Engineering*, 2:841–845.

[Xie, 2003] Xie, M. (2003). *Fundametals of Robotics*, volume 54 of *Series on Machine Perception and Artificial Intelligence*. Singapore-MIT Alliance.

[Zeng and Weng, 2004] Zeng, S. and Weng, J. (2004). Obstacle avoidance through incremental learning with attention selection. *International Conference on robotics and automation*, pages 115–121.

Apéndice A

Experimentos adicionales.

En este apéndice se muestran experimentos adicionales bajo la misma estructura del Capítulo 8. Estos experimentos se reportan con la intención de ofrecer un panorama más amplio del funcionamiento de los algoritmos propuestos.

A.1. Planeación de trayectorias

En las Figuras A.1 y A.2 se muestran dos mapas que serán utilizados en los experimentos mostrados en esta sección.

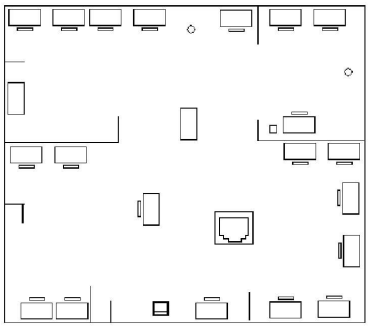


Figura A.1: Ambiente de aproximadamente 10×10 metros.

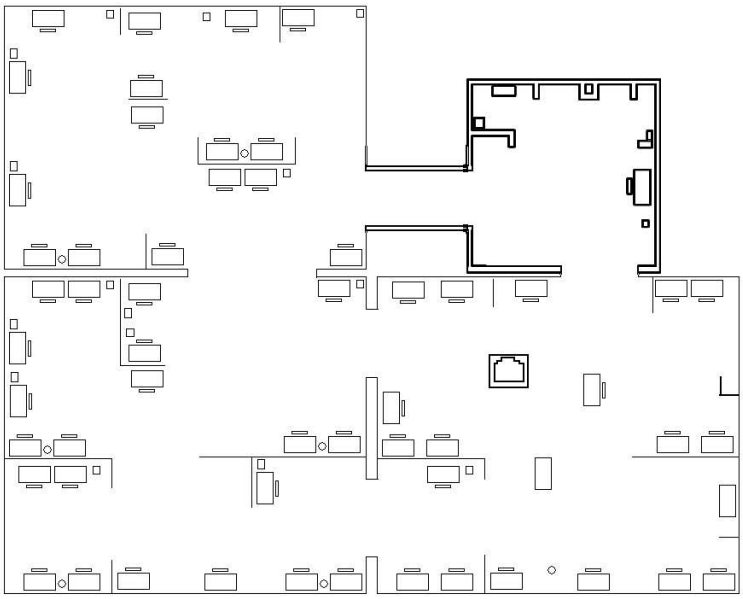


Figura A.2: Ambiente de aproximadamente 20×20 metros.

En las Figuras A.3 y A.4 se muestran diferentes rutas trazadas por el planificador en base al mapa mostrado en la Figura A.1.

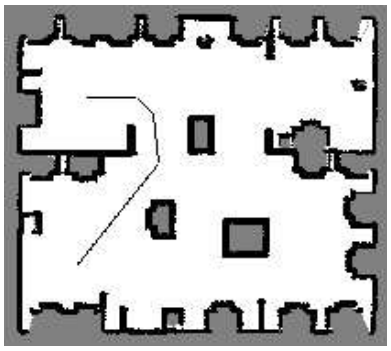


Figura A.3: Caso 1: Ruta obtenida por el planificador de trayectorias.

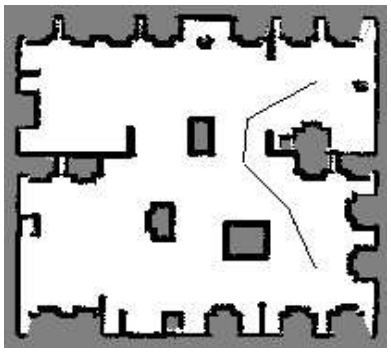


Figura A.4: Caso 2: Ruta obtenida por el planificador de trayectorias.

En las Figuras A.5, A.6, A.7 y A.8 se muestran diferentes rutas trazadas por el planificador en base al mapa mostrado en la Figura A.2.

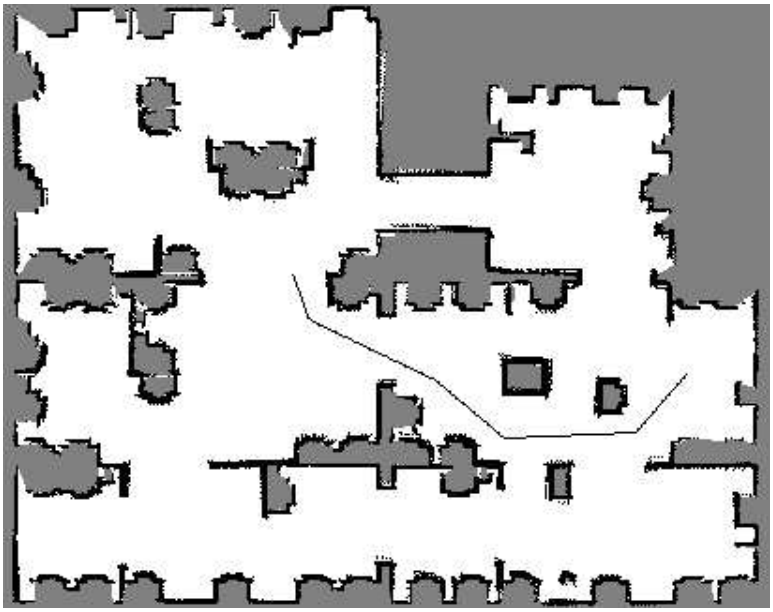


Figura A.5: Caso 3: Ruta obtenida por el planificador de trayectorias.

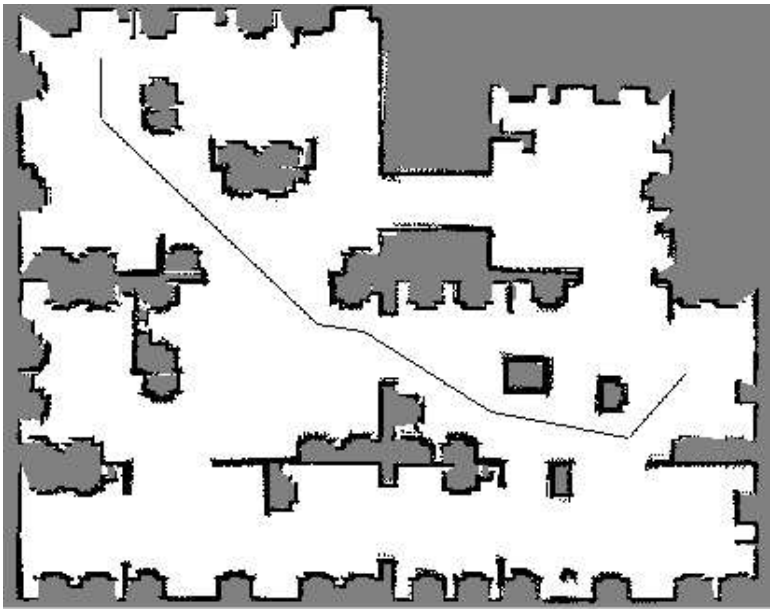


Figura A.6: Caso 4: Ruta obtenida por el planificador de trayectorias.

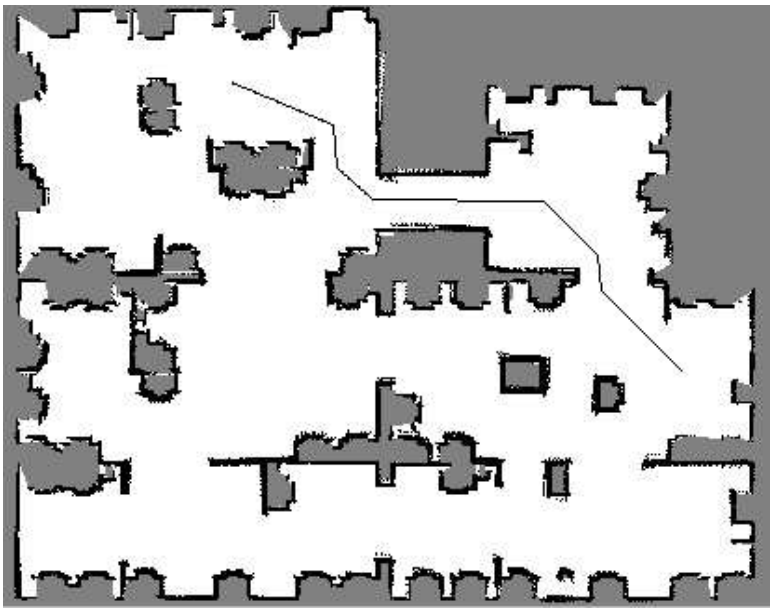


Figura A.7: Caso 5: Ruta obtenida por el planificador de trayectorias.

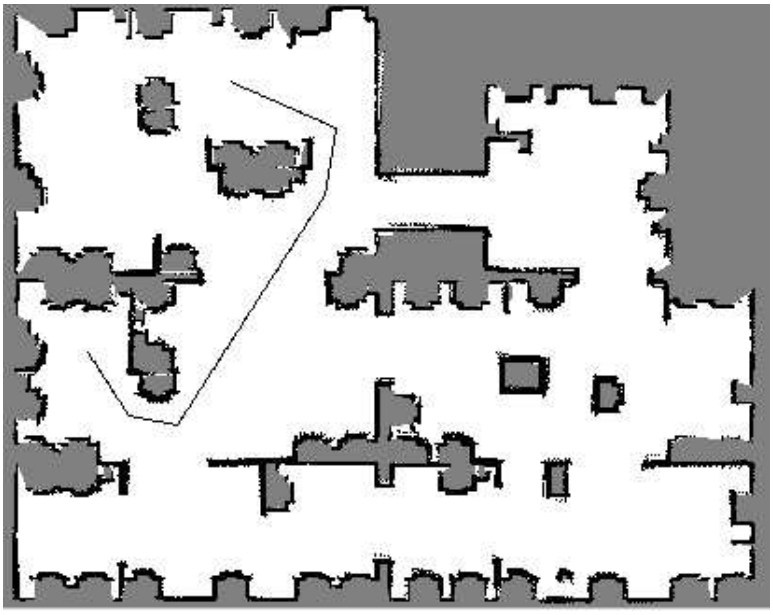


Figura A.8: Caso 6: Ruta obtenida por el planificador de trayectorias.

Apéndice B

Manual del módulo de navegación.

Esta aplicación proporciona a un robot móvil la capacidad para navegar en ambientes interiores. Fue desarrollado en el Laboratorio de Sistemas Inteligentes del ITESM Campus Cuernavaca.

Las tareas que realiza son las siguientes:

- Planificación de trayectorias.
- Seguimiento de la posición.
- Evasión de obstáculos.

De manera adicional se incluye una interfaz gráfica llamada Whiteboard para el monitoreo de las actividades del robot. Este manual abarca los siguientes aspectos:

- Compilación
- Ejecución
- Distribución física de los módulos
- API para su uso de manera remota (por HTTP)
- Interfaz de monitoreo: Whiteboard

B.1. Compilación

B.1.1. Antes de compilar

Requerimientos:

- glib 2.0
- gthread 2.0
- libsoup 2.2
- player 1.6.x
- gtk 2.0

Estos paquetes deben estar instalados y sus bibliotecas deben estar accesibles por medio de la utilidad pkg-config. Para verificarlo escriba:

$ make dep

Si la salida en pantalla reporta algún error alguno de los paquetes no está instalado o no esta accesible por medio de pkg-config.

B.1.2. Compilación

Para compilar el módulo de navegación escriba:

$ make

Para compilar la interfaz gráfica Whiteboard que monitorea las actividades del robot escriba:

$ make whiteboard

B.2. Ejecución

Para inicializar el sistema de navegación escriba:

$ ./modulo_navegacion

Para inicializar Whiteboard escriba:

$ ./whiteboard

Ambos comandos tienen opciones de línea de comandos. Para obtener un listado escriba el nombre del comando seguido de -?. En el caso de modulo_navegacion estas opciones le permiten al usuario indicar la ubicación de la interfaz remota y/o de player, los puertos de conexión para la comunicación, etc.

B.3. Distribución física de los módulos

Son varios los módulos que intervienen en el sistema robótico completo: construcción de mapas, interacción humano-robot, coordinador de tareas, etc. Cada uno cumple con una función específica y es controlado por medio de un coordinador de tareas. El módulo de navegación expone un API por medio del protocolo HTTP con la finalidad de que el coordinador de tareas pueda solicitarle información ó enviarle instrucciones. Por otro lado se dispone de una interfaz gráfica remota, la cual se comunica con el módulo de navegación por medio de sockets. Esta interfaz es opcional y el sistema puede funcionar sin problemas en su ausencia. Para interactuar con el robot el módulo de navegación necesita interactuar con player a través de un API proporcionado por player (libplayerc). Player a su vez se vale de un driver específico para el robot que se desee utilizar.

En la Figura B.1 se presenta un diagrama de despliegue donde se observa la relación entre cada uno de los módulos que intervienen. PC1, PC2 y PC3 son computadoras conectadas por medio de una red inalámbrica. En la Figura B.1, player y el módulo de navegación se presentan en una misma computadora (PC2), esto no es una restricción ya que player podría estar en una computadora diferente del módulo de navegación.

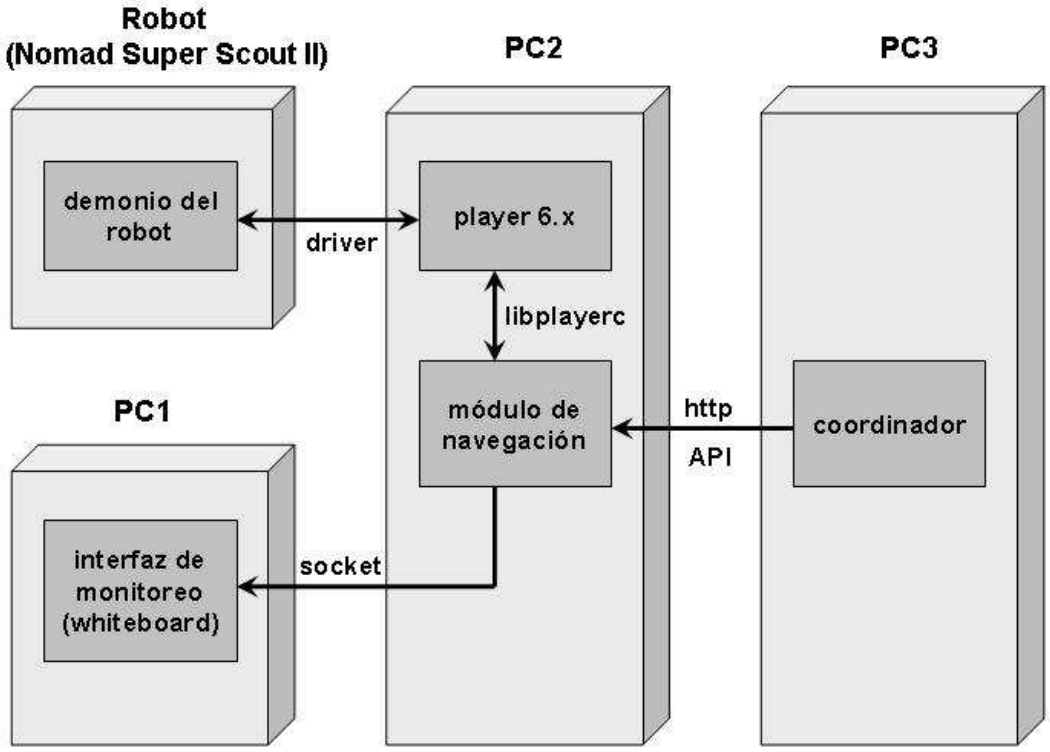


Figura B.1: Diagrama de despliegue donde se observa la interacción entre los módulos del sistema.

B.4. API para su uso de manera remota (por HTTP)

Una vez iniciado, el módulo de navegación puede ser utilizado de manera remota. Para ello se define una interfaz por medio de HTTP. Esto permite a cualquier aplicación enviar y recibir información del módulo de navegación utilizando una URL, de la misma manera en que se hace referencia a una página WEB.

Para entender mejor las instrucciones que forman parte de la interfaz es necesario conocer un poco el funcionamiento interno del módulo de navegación. En principio, el módulo de navegación está representado por un autómata (ver Figura B.2). El módulo de navegación siempre inicia en el estado *sin mapa* y cambia a otros estados dependiendo de las instrucciones que se le envíen.

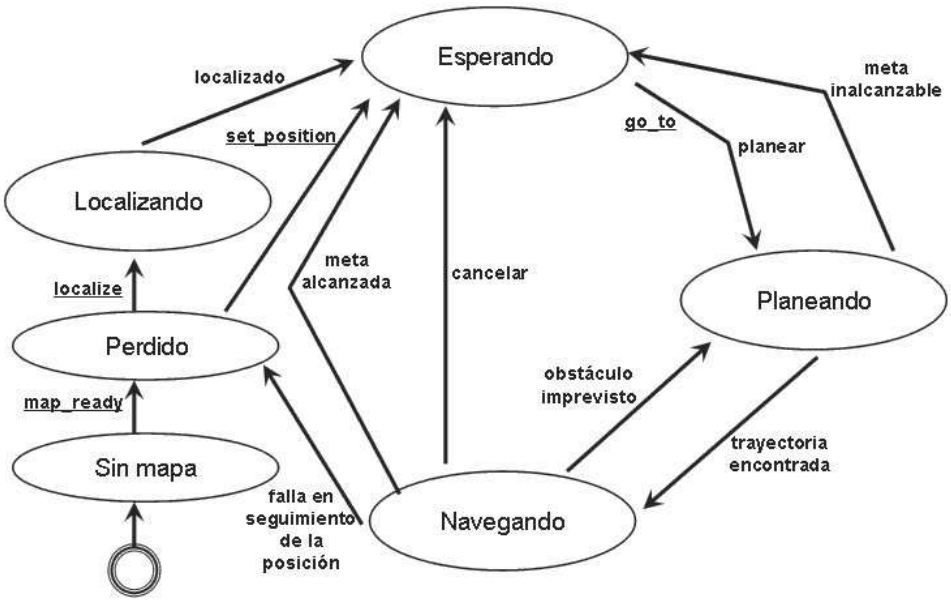


Figura B.2: Autómata que representa el estado del módulo de navegación

Dependiendo del estado en el que se encuentre el autómata se podrán enviar diferentes instrucciones, por ejemplo: no de le puede indicarle al módulo de navegación que vaya a un lugar del ambiente si no posee un mapa del mismo.

En adelante para hacer referencia al nombre de la máquina donde se encuentra el módulo de navegación se utilizará la palabra *host*. Para hacer referencia al puerto HTTP donde el módulo de navegación escucha peticiones se utilizará la palabra *puerto*. De esta manera, las peticiones son de la forma:

http://host:port/instruccion?param1=val1&param2=val2&...&paramN=valN

Siempre que se haga una petición al módulo de navegación lo primero que debe verificarse es el *status_code* de la petición HTTP. Este código no tiene nada que ver con el estado del módulo de navegación sino con el protocolo HTTP; los códigos 2xx son utilizados para indicar que se obtuvo una respuesta a la petición mientras que los 4xx y 5xx se usan para indicar errores en el cliente y servidor respectivamente. Es muy importante que el usuario se asegure que el *status_code* indique que se ha obtenido una respuesta a la petición. De no ser así deberá revisar la URL empleada y asegurarse que esté bien escrita, que el nombre del host y el puerto correspondan y que el nombre de

la petición y sus parámetros sean los correctos.

Una vez que se ha asegurado que el *status_code* es 2xx el siguiente paso es interpretar la respuesta a la petición. Dicha respuesta es una cadena de caracteres. El primer caracter es un número: 0 indica que no hubo ningún problema y que la instrucción está siendo procesada, 1 indica que hubo un error. Después del número hay un espacio y enseguida un mensaje que indica el error o información acerca de la petición que está siendo procesada según sea el caso. Una excepción a esta regla es la instrucción *get_status* ya que después del primer caracter hay un espacio y enseguida el código de estado que se solicitó. A continuación se muestra el detalle de las instrucciones.

B.4.1. Instrucción get_status

Solicita el estado actual de módulo de navegación. La instrucción *get_status* es la que se usa con más frecuencia, sobre todo para verificar el estado del módulo de navegación antes de enviar otra instrucción.

Parámetros: No necesita ningún parámetro. La petición debe quedar de la siguiente manera:

http://host:port/get_status

Valor devuelto: Devuelve una cadena de caracteres con el siguiente formato: el caracter 0 ó 1 dependiendo de si hubo un error o no, enseguida un espacio y a continuación un valor numérico que representa el estado del módulo de navegación. Este valor debe ser interpretado de acuerdo con la siguiente definición (Se recomienda copiarla en algún lugar dentro del código).

```
typedef enum{
   GSTATUS_ESPERANDO    = 2,
   GSTATUS_NAVEGANDO    = 3,
   GSTATUS_LOCALIZANDO  = 4,
   GSTATUS_SIN_MAPA     = 5,
   GSTATUS_PERDIDO      = 6
}t_gstatus;
```

B.4.2. Instrucción map_ready

Le indica al módulo de navegación que hay un mapa que está listo para ser utilizado. Para ello, le debe indicar también el nombre del host donde está el mapa, el nombre del archivo del mapa (.pgm) y si debe procesarlo o no para construir un modelo para localización global. Cuando el robot recibe esta instrucción intenta obtener el mapa indicado y en caso de hacerlo con éxito cambia del estado *sin_mapa* al estado *perdido*.

Parámetros: *host* que indica el nombre de la máquina donde está el mapa, *filename* que indica el nombre del archivo del mapa y *procesar* que indica si debe o no ser procesado el mapa. La petición debe quedar de la siguiente manera:

http://host:port/map_ready?host=maquina&filename=archivo&procesar=codigo_procesar

Donde *codigo_procesar* es un valor que le indica al módulo de navegación si debe o no procesar el mapa de acuerdo con la siguiente definición:

```
typedef enum{
   PROCESAR_NUNCA                = 0,
   PROCESAR_EN_AUSENCIA_DE_MODELO = 1,
   PROCESAR_SIEMPRE              = 2
} t_procesar;
```

Procesar el mapa implica una cantidad de tiempo considerable, es por eso que se le debe indicar cuando hacerlo. El mapa debe procesarse solo si el robot va a necesitar localizarse globalmente (PROCESAR_EN_AUSENCIA_DE_MODELO ó PROCESAR_SIEMPRE). En caso de no necesitar esta funcionalidad el parámetro *procesar* debe tener el valor PROCESAR_NUNCA. La posición puede enviarse al robot usando la instrucción *set_position*.

Valor devuelto: Devuelve una cadena de caracteres con el siguiente formato: el caracter 0 ó 1 dependiendo de si hubo un error o no, enseguida un espacio y a continuación una cadena de caracteres con un mensaje informativo.

B.4.3. Instrucción set_position

Le indica al módulo de navegación cual es la posición y orientación del robot con respecto al mapa. Para ello el robot ya debe poseer un mapa, es decir, debe estar en el estado *perdido*. Cuando esta instrucción es recibida por el robot su estado cambia del estado *perdido* al estado *esperando*.

Parámetros: *lin* y *col* indican la línea y columna en la que se encuentra el robot dentro del mapa, considerando que se trata de un mapa de celdas donde cada pixel del mapa (archivo .pgm) representa una celda; *ori* le indica al robot cual es su orientación dentro del mapa en radianes. La petición debe quedar de la siguiente manera:

http://host:port/set_position?lin=xxx&col=xxx&ori=xxx

Valor devuelto: Devuelve una cadena de caracteres con el siguiente formato: el caracter 0 ó 1 dependiendo de si hubo un error o no, enseguida un espacio y a continuación una cadena de caracteres con un mensaje informativo.

B.4.4. Instrucción go_to

Le indica al módulo de navegación cuál es la ubicación del lugar a donde debe dirigirse. Para ello el robot ya debe poseer un mapa y conocer su posición, es decir, debe estar en el estado *esperando*. Cuando esta instrucción es recibida por el robot su estado cambia del estado *esperando* al estado *navegando* y permanece en ese estado mientras se mueve, una vez que ha llegado a su destino cambia nuevamente al estado *esperando*.

Parámetros: *lin* y *col* indican la línea y columna dentro del mapa a la que debe ir el robot, considerando que se trata de un mapa de celdas donde cada pixel del mapa (archivo .pgm) representa una celda. *ori* le indica al robot cual es la orientación final deseada, es decir, una vez que haya llegado a la meta, cual es el ángulo en el que debe quedar orientado en radianes. La petición debe quedar de la siguiente manera:

http://host:port/go_to?lin=xxx&col=xxx&ori=xxx

Valor devuelto: Devuelve una cadena de caracteres con el siguiente formato: el caracter 0 ó 1 dependiendo de si hubo un error o no, enseguida un espacio y a conti-

nuación una cadena de caracteres con un mensaje informativo.

B.4.5. Instrucción localize

Le indica al módulo de navegación que debe realizar el procedimiento para localización global. Para ello el robot ya debe poseer un mapa y haberlo procesado para tener un modelo del ambiente, es decir, debe estar en el estado *perdido* o en el estado *esperando*. Cuando esta instrucción es recibida por el robot su estado cambia al estado *localizando* y permanece en ese estado mientras se localiza, una vez que ha terminado de localizarse cambia al estado *esperando*.

Parámetros: No necesita ningún parámetro. La petición debe quedar de la siguiente manera:

http://host:port/localize

Valor devuelto: Devuelve una cadena de caracteres con el siguiente formato: el caracter 0 ó 1 dependiendo de si hubo un error o no, enseguida un espacio y a continuación una cadena de caracteres con un mensaje informativo.

B.5. Interfaz de monitoreo: Whiteboard

Con la finalidad de tener una interfaz de monitoreo se desarrolló Whiteboard. La aplicación fue desarrollada usando las bibliotecas de GTK versión 2.0, una captura de pantalla se muestra en la Figura B.3. Esta interfaz puede colocarse en una máquina para monitorear de manera remota las actividades del robot.

Dispone de una barra de herramientas donde están las instrucciones que pueden enviársele al robot. La ventana principal de la aplicación está dividida en dos partes: la parte superior muestra el mapa del ambiente para que el usuario indique la meta haciendo *click* sobre el punto correspondiente. La parte inferior muestra un mapa cuyo contenido es meramente informativo y le muestra al usuario el detalle relacionado con las actividades del robot, por ejemplo: la ruta a seguir, las ubicaciones más probables del robot, etc. La barra de estado en la parte inferior de la ventana muestra en estado actual del robot, de acuerdo con el autómata de la Figura B.2. Dependiendo del estado

en el que se encuentre el módulo de navegación puede realizar ciertas tareas por lo que ésta información es muy importante.

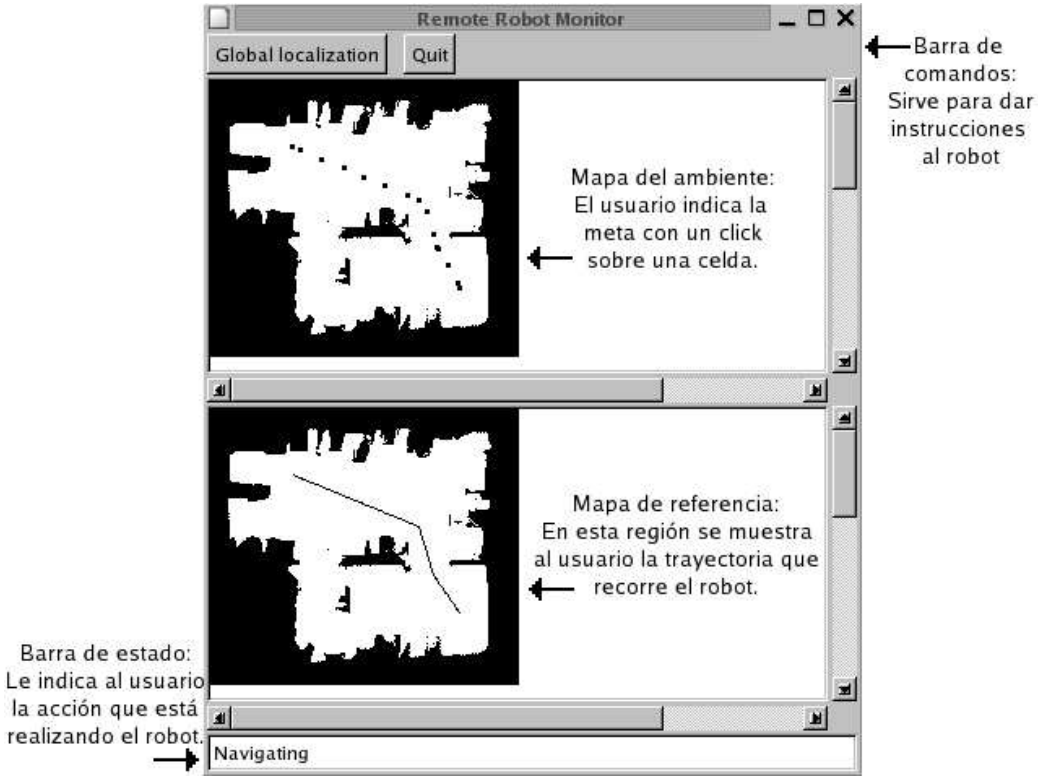


Figura B.3: Captura de pantalla de la interfaz de monitoreo Whiteboard.