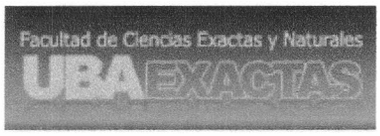

Navegación de Robots en Ambientes Interiores Usando Predicción de Incertezas e Invariates Proyectivos

por

José Gerpe Blanco y Mariano César Buñirigo

Dirigidos por

Dra. Marta Mejail y Dr. Julio Jacobo

Trabajo presentado al Departamento de Computación - Facultad de Ciencias Exactas y Naturales - Universidad de Buenos Aires
en el año 2008,
según requisito para el grado de
Licenciado en Ciencias de la Computación

Nuestro Agradecimiento:

A Marta y a Julio, por la infinita paciencia y dedicación con nuestro proyecto.

A Horacio Abbate, por compartir con nosotros sus conocimientos, fundamentales para la construcción del Simulador.

A nuestras familias, que nos apoyaron y acompañaron siempre para que podamos completar este camino.

Resumen

Presentamos una estrategia de navegación de robots en ambientes interiores, basada en Visión por Computadora. El sistema de navegación resultante no necesita una base de datos compleja para modelar el ambiente y reduce la cantidad de ejecuciones del algoritmo de autolocalización. Ambas caracteristicas producen un sistema económico desde el punto de vista computacional. La estrategia de navegación combina un método para predecir la incerteza de la posición del robot, con un método de localización que usa invariantes proyectivos, basado en visión. Desarrollamos el sistema para navegar en pasillos, utilizando conocimiento específico sobre este tipo de ambientes. Para probar la performance del sistema, construimos un simulador, mediante herramientas de computación gráfica, el cual genera pasillos virtuales. Usando esta herramienta, confirmamos la efectividad de nuestra estrategia de navegación, realizando experimentos en pasillos simulados,

We present an indoor robot navigation strategy, based on computer vision. The resulting navigation system does not need a complex data base to model the environment and, at the same time, reduces the number of executions of the self-localization algorithm. Both characteristics result in a computationally economic navigation system. The navigation strategy combines a method to predict the robot position uncertainty , with a self-localization method using projective invariants, based on computer vision. We developed a system to navigate within hallways, introducing specific knowledge about this kind of environments. To test the performance of the system, we have built a simulator, using computer graphics tools, which creates virtual indoor hallways environments. Using this tool, we confirmed the effectiveness of our navigation strategy, by performing experiments in simulated hallways.

Contenido

Capítulo 1

Introducción

Para comenzar nuestro estudio, definimos qué es un sistema de navegación de robots: es un sistema cuya función es dirigir los movimientos del robot para llevarlo hasta una posición final, definida como objetivo, a partir de una posición inicial determinada. El robot se desenvuelve dentro de un ambiente con características definidas, el cual eventualmente puede incluir obstáculos, y debe tomar decisiones que se reflejen en comandos de movimiento que lo lleven a su destino [KAK/02].

Luego de la ejecución de un movimiento, no es posible predecir exactamente la nueva posición, ya que los movimientos del robot no son perfectos, producto de las irregularidades del suelo, y del nivel de precisión limitado de los sistemas mecánicos de movimiento. Pero para definir una secuencia de comandos de movimientos efectiva, es necesario que el robot conozca su posición. La tarea mediante la cual el robot intenta determinar su posición real en el ambiente se denomina *auto-localización.* Resolver el problema de la auto-localización es uno de los problemas centrales de cualquier sistema de navegación. Los sistemas basados en visión toman decisiones a partir del procesamiento de imágenes capturadas por los sensores del robot. El procesamiento de las imágenes persigue dos objetivos principales: detección de obstáculos y determinación de la posición del robot. Si bien existen tecnologías no basadas en visión para resolver el problema de la auto-localización, como el relevamiento del ambiente mediante scanners lasers o sonares, para implementar estos sistemas es necesario contar con hardware costoso. Por otro lado, los métodos basados en visión se muestran robustos en ambientes estáticos [KAK/02]. En nuestro trabajo, hemos utilizado un método de auto-localización basado en visión, el cual

utiliza las imágenes de una única cámara instalada en el robot.

Kak y Kosaka han propuesto en [KOS/92] un sistema de navegación, llamado FINALE, basado en la predicción de la incerteza de la posición del robot y en un modelo 3D complejo del ambiente, para resolver el problema de la auto-localización. Cuando la incerteza supera un umbral, la posición del robot es calculada comparando el modelo 3D con las imágenes capturadas por la cámara del robot. Otra estrategia de posicionamiento, introducida en [KYO/97], usa un método basado en *Invariantes Proyectivos*, y no necesita ningún modelo geométrico o topológico.

En interiores, existen distintos tipos de ambientes sobre el cual un robot puede navegar. Por ejemplo, en un edificio universitario, encontramos aulas, pasillos, restaurantes, salas de reuniones, etc, los cuales todos conforman tipos de ambientes diferentes, con características topológicas particulares, las cuales deben ser consideradas por la estrategia de navegación del robot para que el mismo pueda moverse con éxito. En nuestro trabajo, hemos implementado el sistema de navegación para un ambiente tipo pasillo . La característica distintiva de un pasillo es que es un ambiente largo, angosto, y eventualmente con puertas en ambas paredes laterales, como el observado en la figura 1-1.

Nuestra estrategia se basa en una técnica de predicción de incertezas utilizando un filtro Extendido de Kalman (EKF) [WEL/03] para determinar cuándo es necesario determinar la posición del robot. Esto se lleva a cabo gracias a un método de auto-localización mediante *Invariantes Proyectivos.*

Introdujimos una modificación en EKF y utilizando la predicción de incertezas se reduce la cantidad de ejecuciones del método de localización evitando correcciones innecesarias. A su vez, los *Invariantes Proyectivos* eliminan la necesidad de que el sistema utilice un modelo 3D del ambiente, de esta forma estamos generando un sistema que reduce el costo computacional del esquema inicial.

Encontramos limitaciones del método de *Invariantes Proyectivos*, por lo que extendimos el método para estimar la orientación del robot, e incorporamos una marca artificial, reconocida mediante descriptores de Fourier, con el objetivo de que el robot gire cuando llega al final del pasillo para proseguir su recorrido.

Para evaluar este sistema de navegación, construimos un simulador usando OpenGL, el cual genera un ambiente virtual tipo pasillo.

Navegación de Robots

Las capacidades electromecánicas del robot determinan, o restringen, las posibles estrategias del sistema de navegación. Concentramos nuestro trabajo en estrategias de navegación basadas en visión, mediante el procesamiento de las imágenes capturadas por el robot. Asumimos la existencia de una única cámara con ángulo de apertura dinámico.

En los últimos años se destacan dos corrientes diferenciadas en el estudio del problema de navegación de robots: navegación en ambientes exteriores y navegación en ambientes interiores. Los problemas planteados por ambos tipos de ambientes son de naturaleza diferente. Por ejemplo, la irregularidad marcada del terreno en ambientes exteriores contrasta con las características de los pisos sobre los cuales el robot ejecuta sus movimientos en ambientes interiores. Nuestro análisis se concentra en las soluciones para ambientes interiores.

Para la resolución del problema de auto-localización, el robot debe poseer algún tipo de conocimiento sobre el ambiente que lo rodea. Siguiendo la clasificación propuesta en [KAK/02], el tipo de conocimiento utilizado define al sistema de navegación como *basado en mapas*, o *no basado en mapas*. Los sistemas basados en mapas, almacenan en el robot un modelo del ambiente, o *mapa*. Esta información puede ser representada de la forma de un modelo 3D del entorno, un modelo topológico, una base de datos de imágenes, etc. La auto-localización se realiza mediante un matching entre el mapa almacenado en el robot, contra las imágenes capturadas por las cámaras del mismo. En otras palabras, el robot "busca en el mapa" lo que está observando, para saber en que lugar se encuentra. Por otro lado, los sistemas no basados en mapas no utilizan ningún tipo de conocimiento previo del ambiente. En este caso, los comandos de movimiento deben determinarse mediante la extracción de información relevante de las imágenes capturadas por la cámara.

El sistema FINALE [KOS/92] basa su estrategia de posicionamiento en un modelo 3D, generado mediante CAD, del cual obtiene las características que espera encontrar en la imagen capturada por el robot, mientras que.el método de posicionamiento basado en *Invariantes Proyectivos* [KYO/97] puede clasificarse como un método no basado en mapas. Presentamos a continuación una introducción a las características que tomamos de [KYO/97] y [KOS/92] para la implementación de nuestro sistema.

Predicción de Incertezas

El sistema FINALE, propuesto en [KOS/92], comienza su ejecución desde una posición inicial determinada, la cual es especificada por un operador humano. El sistema realiza actualizaciones de la incerteza en la ubicación del robot a medida que el mismo se mueve en el ambiente. Esto se efectúa a través de un modelo estadístico de la incerteza sobre la localización del robot, en función de las capacidades de movimiento del mismo. Cuando la incertidumbre supera un umbral, el robot hace uso de sus sensores y de un modelo geométrico 3D del ambiente, para corregir y determinar su posición real. De esta forma, el robot tiene en todo momento una estimación de su posición, en función de la cual determina los comandos de movimiento necesarios para llegar a su objetivo.

Invariantes Proyectivos

El método de localización propuesto en [KYO/97] fue implementado para la navegación del robot en ambientes tipo pasillo, y analiza la distribución de los puntos definidos por la intersección entre el piso del ambiente y las puertas para determinar la posición. Estos puntos se denominan *puntos característicos.* Puede observarse un ejemplo de estos puntos en un pasillo en la figura 1-1.

Sobre subconjuntos de puntos característicos colineales se realiza el cálculo de cocientes cruzados (cross-ratios), los cuales son invariantes ante transformaciones proyectivas [HAR/04]. El sistema de navegación cuenta con una base de datos generada previamente con todos los cross-ratios del ambiente donde se realizará la navegación. El sistema procesa las imágenes capturadas por el robot, detectando los puntos característicos y buscando en la base de datos un matching entre los cocientes calculados a partir de la imagen y los almacenados, lo cual permitirá determinar la posición del robot.

Simulación del Ambiente de Navegación

Hemos desarrollado un simulador sobre el cual ejecutamos el sistema de navegación para evaluar su comportamiento.

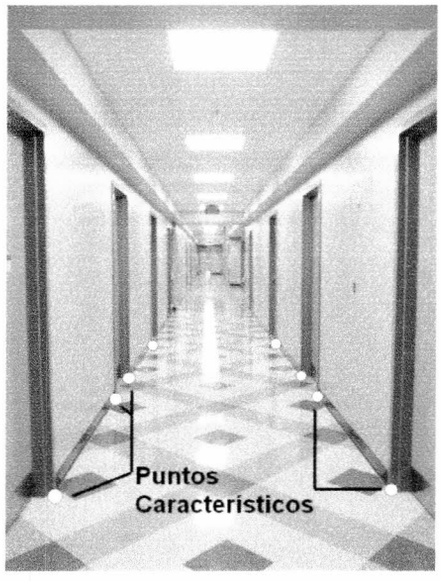

Figura 1-1: Puntos Característicos de un pasillo. El método de Invariantes Proyectivos utiliza estos puntos para determinar la posición del robot.

El desarrollo se basó en la librería OpenGL, utilizando las facilidades que provee para implementar modelos 3D y manipular las matrices de proyección y visualización para generar las imágenes virtuales del pasillo. El simulador reproduce un robot que posee una única cámara, con ángulo de apertura variable, y con la capacidad de hacer movimientos de rotación y de traslación en línea recta.

Organización del trabajo

El presente trabajo se organiza de la siguiente manera: el capítulo 2 describe las características de un sistema de visión y explica con mayor detalle las estrategias de navegación más conocidas. El capítulo 3 presenta la estrategia de navegación que incluye el sistema de movilidad y de autolocalización adoptados en nuestro trabajo. El capítulo 4 detalla la implementación realizada para construir el ambiente de simulación utilizado en este trabajo. El capítulo 5 explica la construcción lógica del sistema de navegación y los módulos que lo componen. El capítulo 6 muestra un resumen de los resultados obtenidos a través de las pruebas realizadas con la

herramienta de simulación. Por último, el capítulo 7 expone las conclusiones finales del trabajo presentando un conjunto de trabajos futuros que pueden realizarse a fin de mejorar y probar nuestra estrategia en el mundo real.

Capítulo 2

Navegación de Robots Usando Visión por Computadora

Presentamos una breve descripción de los sistemas de visión por computadoras. Comenzamos con una definición, tomada de [SHA/02], que indica qué entendemos por sistema basado en visión.

Definición 1

El objetivo de un sistema de visión es tomar decisiones útiles sobre objetos físicos reales y escenas, basadas en el procesamiento de imágenes.

Para poder tomar decisiones sobre objetos reales, es necesario construir alguna descripción o modelo de los objetos, a partir de las imágenes capturadas del mundo real. En función de esto, muchos expertos en el tema presentan una definición alternativa para los sistemas de visión por computadoras.

Definición 2

El objetivo de un sistema de visión es la construcción de una descripción de la escena observada en una imagen del mundo real.

Los problemas centrales que deben atacarse para la construcción de un sistema de visión pueden clasificarse de la siguiente manera:

1. **Sensado**: ¿Cómo se capturan las imágenes del mundo real? ¿De qué forma las imágenes codifican las propiedades del mundo, como iluminación, materiales, relaciones espaciales, etc? La cantidad y tipo de cámaras determina el conjunto de imágenes que se utilizarán como representación del mundo real.

2. **Información codificada**: ¿Cómo se proyectan las características tridimensionales del mundo en las imágenes capturadas? ¿Cómo identificar información sobre la geometría, texturas, movimientos e identidad de los objetos?

3. **Representaciones**: ¿Qué representaciones deben utilizare para la descripción de objetos, sus partes, propiedades y relaciones?

4. **Algoritmos**: ¿Qué métodos son apropiados para el procesamiento de las imágenes para construir descripciones del mundo y sus objetos?

Tanto la definición 1 como la 2 resultan pertinentes a un sistema de navegación de robots basado en visión, dado que el robot debe tomar decisiones de movimiento en función del análisis de las imágenes capturadas y, para ello, debe extraer descripciones de los objetos observados en la imagen para su interpretación. Por ejemplo, la identificación de objetos de formas definidas y el análisis de su posición relativa dentro de la imagen, puede utilizarse para deducir la posición del robot en el mundo real. En la figura 1-1 vemos un ejemplo donde se identifican en la imagen los puntos definidos por la intersección de las líneas del suelo con las puertas. Estos puntos pueden utilizarse para obtener la posición del robot dentro del pasillo, como expondremos más adelante al presentar el método de *Invariantes Proyectivos.*

2.1 Navegación y Localización

Los sistemas de navegación basados en visión, incorporan conocimiento sobre lo que se espera que el robot vea. Esta información puede ser almacenada en el robot previamente a la navegación por un operador humano, o el mismo robot puede hacer un relevamiento del ambiente utilizando su cámara, para extraer automáticamente conocimiento sobre el mismo durante su translación. No se conocen sistemas de navegación que trabajen con ausencia total de información sobre el ambiente [KAK/02].

Desde los primeros desarrollos en el tema, distintos sistemas utilizan información sobre la geometría del espacio en el cual el robot se desenvuelve, desde modelos CAD hasta modelos más simples como mapas de ocupación, mapas topológicos o secuencias de imágenes. Mediante comparación de las imágenes capturadas por el robot con los modelos almacenados se determina la posición actual del robot, en función de la cual se definen los comandos de movimiento para que el robot siga su camino hacia el objetivo. La complejidad computacional de los procesos de posicionamiento hace necesario definir una estrategia que minimize la cantidad de veces que es necesario acudir a este proceso para definir la posición. Luego de la ejecución de los comandos de movimiento no es posible determinar con absoluta certeza la posición del robot, dada que ningún sistema mecánico de movimiento responde con absoluta precisión a estos comandos. Posibles imperfecciones del mecanismo de movimiento, irregularidades del suelo, efectos de la inercia, etc, pueden mencionarse como causas de la imposibilidad de conocer la posición del robot conociendo solo la posición inicial y los movimientos efectuados.

Reproducimos aquí la clasificación propuesta en [KAK/02], sobre los tipos de sistemas de navegación, que agrupa los sistemas en las siguientes clases:

- **Navegación basada en mapas**: estos sistemas utilizan modelos geométricos o topológicos de entorno, generados por un operador humano antes de la ejecución de la navegación. Los mismos se almacenan en el robot, para ser utilizados como información sobre el ambiente para la determinación de la posición y de los comandos de movimiento.

- **Navegación basada en generación de mapas**: estos sistemas utilizan sensores para construir modelos geométricos o topológicos del ambiente, para utilizarlos en la navegación. Estos modelos son construidos por el mismo robot a medida que se va moviendo.

- **Navegación no basada en mapas**: estos sistemas no utilizan una representación explícita del espacio en el cual navega el robot; en su lugar llevan a cabo el reconocimiento de objetos encontrados en el ambiente mediante observación visual.

2.1.1 Métodos basados en mapas

Estos métodos introducen en el robot un modelo del ambiente. Estos modelos pueden ser de diferente grado de detalle, desde un modelo 3D CAD hasta un simple grafo que indica las inter-

relaciones entre los elementos que pueden encontrarse en el ambiente. En los primeros sistemas de visión desarrollados, el conocimiento del ambiente era representado como una grilla. Considerando la proyección a 2D de los objetos presentes en el ambiente sobre el plano horizontal del suelo, se marcaban en la grilla las áreas ocupadas. Estas representaciones son usualmente denominadas *mapas de ocupación.* Estos mapas pueden ser enriquecidos mediante la incorporación de la incerteza relacionada con el error en las mediciones de las coordenadas de los objetos en el espacio realizadas por el robot. Estos mapas son utilizados por el robot para, dada su ubicación, definir el plan de navegación dentro del ambiente para llegar a su objetivo. La utilización de sensores ultrasónicos para detectar los objetos en el ambiente permite comparar los resultados de la observación con el mapa para determinar la posición del robot.

Una implementación de un sistema de navegación basado en mapas puede encontrarse en [THR/98]. De este trabajo reproducimos la figura 2-1, la cual ilustra un mapa de ocupación.

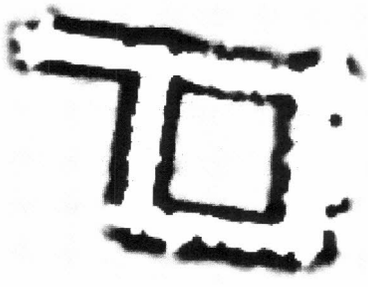

Figura 2-1: Mapa de Ocupación. Las áreas mas oscuras representan alta probabilidad de presencia de objetos sólidos.

Los pasos que se ejecutan en un sistema de navegación basado en mapas pueden enumerarse de la siguiente forma:

1. **Obtención de la imagen**: captura de la imagenes de las cámaras del robot.

2. **Detección de características en la imagen**: extracción de las características que se utilizarán para determinar la posición. Esto puede implicar detección de bordes, filtrado de las imágenes, segmentación de regiones, etc. Mediante una combinación de técnicas se

identifican en la imagen las marcas, objetos o formas geométricas para su comparación con el modelo almacenado como la expectativa del robot.

3. **Matching entre lo observado y lo esperado**: el sistema trata de identificar las características extraídas en el paso anterior dentro del modelo almacenado.

4. **Cálculo de la posición**: determinación de la posición del robot en función de las características extraídas de la imagen en el paso 2 y de la información extraída del modelo en el paso 3.

El tercer paso, el matching entre la observación y lo esperado, presenta la mayor complejidad. Para lograr una navegación efectiva, es necesario utilizar conocimiento específico sobre el ambiente de navegación. Por ejemplo, en un pasillo, podemos considerar las características geométricas comunes en todos los pasillos para detectar objetos como ventanas o puertas. De esta forma, el robot puede reconocer estos objetos en la imagen capturada y buscarlos en el mapa que tiene almacenado, para determinar en que lugar del ambiente se encuentra.

Localización Absoluta, Localización Incremental y Seguimiento de marcas

Es claro que el problema de la localización del robot es uno de los problemas centrales en la definición de la estrategia de navegación. Podemos mencionar tres enfoques diferenciados para encarar la localización del robot, dependiendo de la información con la que cuenta el sistema como dato de entrada.

La *localización incremental* se asume que la posición inicial del robot es conocida aproximadamente al comienzo de la navegación y el objetivo del sistema es deducir la nueva posición en función de los comandos de movimiento ejecutados y de la posición anterior. Esto implica que puede extraerse del mapa del ambiente, al comenzar la navegación, la información de lo que el robot espera ver. Cuando esta expectativa es comparada con la imagen obtenida por el robot, se puede obtener una estimación más precisa de la posición del robot. Ejemplos de sistemas con este tipo de esquema de localización son el sistema FINALE [KOS/92] y el sistema NEURO-NAV [MEN/93]. El primero será descripto en detalle más adelante en este trabajo, por tratarse de la base elegida para la construcción del módulo de propagación de la incerteza de la posición de nuestro sistema. Mencionamos aquí que utiliza un modelo 3D del ambiente

generado por CAD, el cual se utiliza como mapa de la expectativa del robot, para ser comparado con las imágenes capturadas para determinar la posición del robot cuando la incerteza de la misma supera un umbral determinado. Vemos en la figura 2-2, reproducida de [KOS/92], una vista del modelo 3D del sistema FINALE para la navegación de un pasillo.

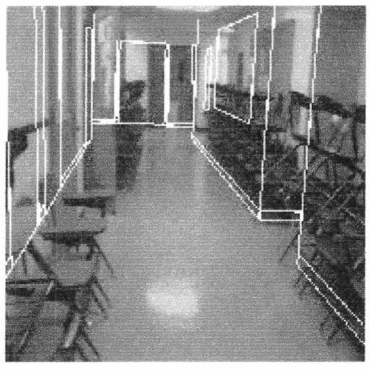

Figura 2-2: Modelo 3D del ambiente.

El sistema NEURO-NAV, en cambio, utiliza una representación topológica del espacio, en la forma de un grafo que representa la estructura física de un pasillo. En la figura 2-3, podemos ver en (a) la descripción de un pasillo, y en (b) una posible representación topológica.

En este grafo de la figura 2-3 vemos tres tipos de nodos para representar pasillos, intersecciones y terminaciones de pasillos. El robot comienza desde un nodo del grafo, por ejemplo C2. Si se le indica al robot que siga el pasillo hasta J2, que doble a la izquierda y que siga el pasillo C3, el robot utiliza una red neuronal para navegar el pasillo de forma tal de mantenerse equidistante a las paredes del mismo. Cuando el módulo detector de marcas detecte que el robot llegó a J2 (por ejemplo, identificando una puerta particular), se ejecutará un comando de rotación, para comenzar una nueva navegación de pasillos, en este caso por C3.

La utilización de un método de *localización absoluta* implica la deducción de la posición inicial del robot comparando las imágenes obtenidas del robot contra toda la base de datos que almacena el mapa del ambiente. Esto presenta un proceso costoso computacionalmente y con un resultado con un menor nivel de certeza inicial.

Otra estrategia de localización, llamada *localización por seguimiento de marcas,* se basa en el seguimiento de marcas u objetos detectadas en la imagen, a través de imágenes consecutivas

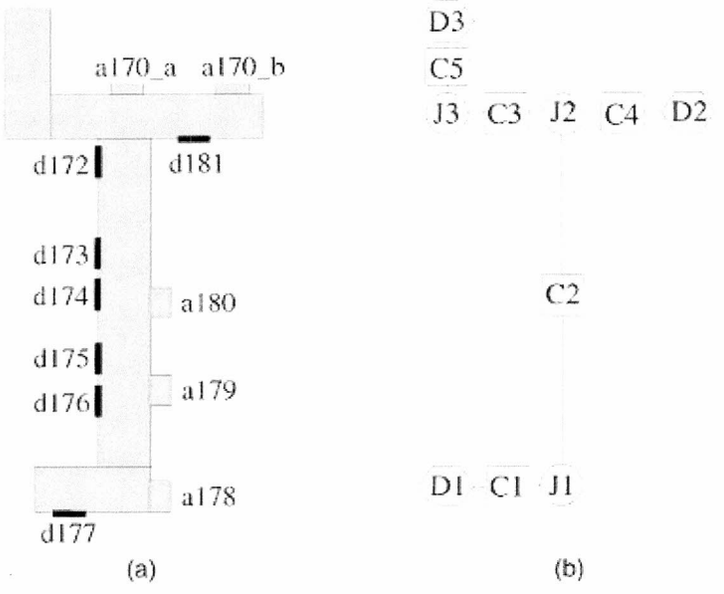

Figura 2-3: Representación topológica de un pasillo.

capturadas por el robot a medida que navega por el ambiente. Las características de las marcas deben ser conocidas por el robot y su identificación en la imagen produce un ajuste de la posición del robot mediante comandos de movimientos. Las marcas puden ser artificiales como en el caso de la figura 2-4, dónde vemos un robot que sigue un línea marcadas en el piso.

2.1.2 Métodos basados en generación de mapas

La generación de mapas requiere que el robot pueda procesar las mediciones del entorno para generar su propio modelo. Estas descripciones del ambiente son difíciles de generar a partir de imágenes, especialmente si hablamos de información métrica. Podemos mencionar dos soluciones propuestas en la literatura: generación de *grillas de ocupación* y creación de *representaciones topológicas del espacio.* En el primer caso se completan las celdas de la grilla, la cual representa al espacio, con información de la certeza de la existencia de un objeto en el lugar representado por la celda. Este dato es obtenido mediante la utilización de sensores ultrasónicos, cuyas lecturas producen la actualización de los datos de la grilla. En el caso de la generación de representaciones topológica, mediante el sensado del ambiente se reconocen

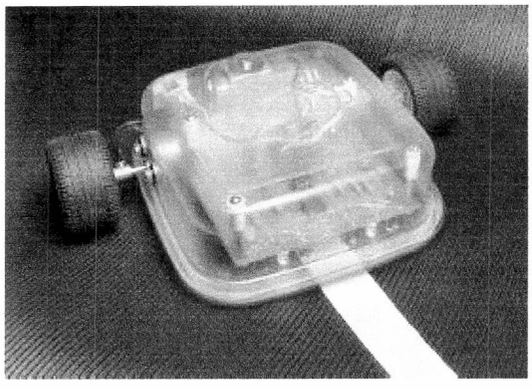

Figura 2-4: Seguimiento del camino marcado.

objetos que se clasificarán como nodos de un grafo, y se relacionarán de acuerdo a lo observado. El robot tiene conocimiento previo sobre los tipos de nodos que puede reconocer y como pueden relacionarse. Un ejemplo de la implementación de grillas de ocupación puede encontrarse en [MOR/85], mientras que [CHO/97] presenta un trabajo orientado a la generación de representaciones topológicas. Alternativamente, numerosos trabajos utilizan métodos estocásticos para la generación de mapas. El trabajo fundacional en este sentido es [SMI/88], dónde se define el concepto de SLAM (Simultaneous Localization And Map Building), dónde se propone un procedimiento probabilístico para construir un mapa estocástico y localizar al robot dentro del mapa. En la actualidad, el trabajo de Lowe sobre SLAM, es referencia ineludible sobre este tema, y define la frontera actual del conocimiento sobre métodos de generación de mapas [LOW/05].

2.1.3 Métodos no basados en mapas

Los métodos que no basan su navegación en mapas, no utilizan ningún tipo de conocimiento previo del ambiente. Los comandos de movimiento del robot son determinados por observación y extracción de información relevante de las imágenes capturadas. Los elementos extraidos pueden ser paredes, escritorios, puertas, etc. Un sistema representativo de este tipo de estrategias es *robee* [SAN/93], el cual reproduce los reflejos de una abeja. Si el robot se posiciona en el centro de un pasillo, la diferencia de velocidad entre las imagenes capturadas por la cámara

instalada a la derecha del robot con las imágenes capturadas por la cámara de la izquierda, determina comandos de movimientos para mantener esa diferencia en un valor cercano a cero, que es el caso en el cual el robot navega por el centro del pasillo (aunque sea curvo, lo cual implica comandos de rotación). Si las velocidades son diferentes el robot se mueve hacia el lado cuya imagen cambia con velocidad menor. En resumen, la idea básica es la medición de diferencia de velocidad entre las imágenes capturadas por cámaras laterales, y la utilización de esta información para guiar al robot. Es importante notar que en una estrategia como esta es central la existencia de algún tipo de textura en las paredes, sin lo cual el robot no puede determinar las variaciones de las imágenes para deducir la velocidad de movimiento. La figura 2-5 esquematiza el funcionamiento de un sistema de este tipo.

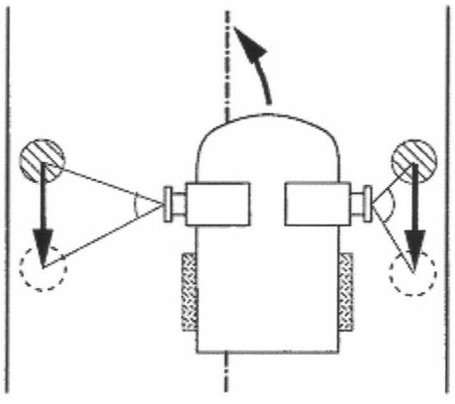

Figura 2-5: Esquema del sistema Robee

Capítulo 3

Estrategia de Navegación

La estrategia de navegación de un robot autónomo incluye principalmente la interacción de dos sistemas: uno de movilidad, encargado de decidir y realizar los movimientos necesarios para alcanzar el objetivo, y otro de localizacion que permite determinar la posición actual aproximada del robot dentro de su ambiente.

3.1 Sistema de Movilidad

Este sistema se encarga de construir el plan de movimientos a realizar para alcanzar el objetivo. Para ello se utiliza la localización actual del robot que se representa a través de la posición y orientación del robot en su universo, la posición se establece como un par de coordenadas que refieren al centro del robot, mientras que la orientación es representada por un ángulo que muestra la dirección dónde apunta la cámara del robot. Inicialmente el robot comienza su recorrido en una localización conocida y realiza movimientos para desplazarse en el ambiente.

Siendo conocida la posición actual, se determina el próximo movimiento que deberá realizar el robot para alcanzar su objetivo. Existen factores internos relacionados al sistema odométrico del robot y externos, del ambiente, que introducen incertidumbre en la posición y orientación final que se planea alcanzar luego del movimiento que se realiza. La incertidumbre de los desplazamientos realizados es acumulativa, esto hace que el robot construya una idea errónea acerca de su localización a lo largo del tiempo.

El grado de certeza sobre la localización actual debe ser entonces calculado y controlado

luego de cada movimiento, dado que el mismo determina si el sistema de movilidad debe interactuar con el sistema de autolocalización para reducir la incertudimbre generada y obtener así información cierta sobre la localización real del robot.

3.1.1 Modelo de Desplazamiento

El modelo de desplazamiento permite estimar la posición del robot luego de la ejecución de un comando de movimiento. Los tipos de movimientos que nuestro robot realizará son: traslaciones y rotaciones. El robot se modela como uno de dos ruedas unidas por un eje común que pasa por el centro del mismo.

Ambos tipos de desplazamiento se parametrizan con dos variables: la distancia que el robot recorre y el ángulo que el robot rota. Al realizarse una traslación el ángulo de rotación es igual a cero, y al realizarse una rotación la distancia de traslación es igual a cero. Estos valores se consideran relativos a la última localización asumida por el robot.

La localización del robot se modela como un vector de tres componentes: las dos primeras componentes especifican la posición del robot en el espacio y refieren al centro del mismo, mientras que la tercera es el ángulo de rotación del robot y refiere a la dirección dónde la cámara esta apuntando. La cámara se enfoca hacia el frente del robot y se ubica en el centro del mismo, coincidiendo de esta manera la posición del robot con la posición de la cámara.

Definimos $\mathbf{u}_k$, la localización del robot en el instante k, como $\mathbf{u}_k = \begin{pmatrix} u_{[x],k}, u_{[y],k}, u_{[\phi],k} \end{pmatrix}^t$ donde, $u_{[x],k}$, $u_{[y],k}$ son las coordenadas de la posición del robot en su universo, y $u_{[\phi],k}$ es la orientación del robot.

Los desplazamientos son medidos desde la posición actual del robot, y están relacionados con el centro del robot y la orientación de su cámara. Definimos el desplazamiento relativo como $\mathbf{d}_k = \begin{pmatrix} d_{[\Delta D],k}, d_{[\Delta\phi],k} \end{pmatrix}^t$, donde $d_{[\Delta D],k}$ es la distancia relativa recorrida por el centro del robot, y $d_{[\Delta\phi],k}$ es el cambio de orientación relativo durante el traslado.

La actualización de la posición del robot en el instante k, se modela como una función $\mathbf{f}$ cuyos parámetros son la posición actual y el desplazamiento relativo llevado a cabo en el instante $k - 1$. Por lo tanto, definimos

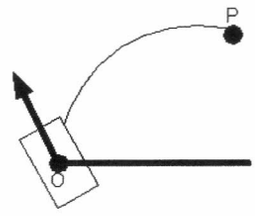

Figura 3-1: Posición Inicial

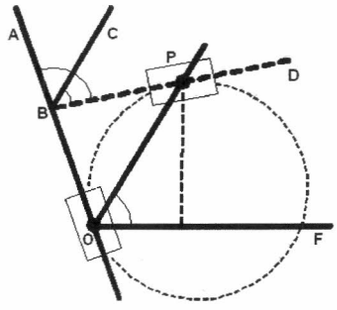

Figura 3-2: Posición Final

$$\mathbf{u}_k = \mathbf{f}(\mathbf{u}_{k-1}, \mathbf{d}_{k-1}) = \begin{pmatrix} f_x(\mathbf{u}_{k-1}, \mathbf{d}_{k-1}) \\ f_y(\mathbf{u}_{k-1}, \mathbf{d}_{k-1}) \\ f_\phi(\mathbf{u}_{k-1}, \mathbf{d}_{k-1}) \end{pmatrix} = \begin{pmatrix} u_{[x],k} \\ u_{[y],k} \\ u_{[\phi],k} \end{pmatrix}$$

La figura 3-1 muestra un desplazamiento de un robot desde la posición O con orientación $\widehat{FOA}$, hasta la posición P. La nueva orientación $u_{[\phi],k}$ está dada por $u_{[\phi],k-1} + d_{[\Delta\phi],k-1}$, pero la nueva posición del robot es desconocida (figura 3-2). Para determinar la posición del robot, necesitamos hacer algunas suposiciones sobre el camino que sigue. Asumimos un camino circular, entonces $d_{[\Delta D],k-1} = arc(OF)$ y $d_{[\Delta\phi],k-1} = \widehat{ABD}$.

Como $\overline{BC}$ biseccIona al ángulo $\widehat{ABD}$ y $\overline{BC}$ // $\overline{OF}$ entonces

$$\widehat{ABC} = \widehat{AOF} = \frac{d_{[\Delta\phi],k-1}}{2}$$

$$\widehat{POF} = \widehat{AOF} - \widehat{AOF} = u_{[\phi],k-1} + \frac{d_{[\Delta\phi],k-1}}{2}$$

Con esta aproximación, el sistema de movilidad que utilizamos queda planteado de la siguiente manera:

$$\begin{aligned} f_x(\mathbf{u}_{k-1}, \mathbf{d}_{k-1}) &\approx u_{[x],k-1} + d_{[\Delta D],k-1} * \cos(u_{[\phi],k-1} + \frac{d_{[\Delta\phi],k-1}}{2}) = u_{[x],k} \\ f_y(\mathbf{u}_{k-1}, \mathbf{d}_{k-1}) &\approx u_{[x],k-1} + d_{[\Delta D],k-1} * \sin(u_{[\phi],k-1} + \frac{d_{[\Delta\phi],k-1}}{2}) = u_{[y],k} \\ f_\phi(\mathbf{u}_{k-1}, \mathbf{d}_{k-1}) &= u_{[\phi],k-1} + d_{[\Delta\phi],k-1} = u_{[\phi],k} \end{aligned} \tag{3.1}$$

Este sistema trabaja como un sistema perfecto, donde los factores internos y externos de error no son considerados, para que el sistema se aproxime a uno real necesitamos modelar en forma separada estos factores que afectan la navegación del robot. Estas fuentes de error se modelan como ruido blanco Gaussiano.

El ruido $\mathbf{q}$ en el instante k es el error producido por factores internos durante el desplazamiento, relacionado directamente a los mecanismos de movimiento del robot. El mismo se modela como un vector de ruido blanco Gaussiano $\mathbf{q}_k$, de componentes independientes con media $\tilde{q}$ y matriz de covarianza Q, que se define como $\mathbf{q}_k \sim N(\tilde{q}_k, Q_k)$.

El ruido $\mathbf{w}$ en el instante k es el error del sistema producido por factores externos, por ejemplo las imperfecciones y el material de construcción del suelo que modifican el desplazamiento esperado. El mismo es modelado como un vector de ruido blanco Gaussiano $\mathbf{w}_k$, de componentes independiente con media $\tilde{w}$ y matriz de covarianza W, que se define como $\mathbf{w}_k \sim N(\tilde{w}_k, W_k)$.

Finalmente, el modelo de navegación queda determinado como

$$\mathbf{u}_k = \mathbf{f}(\mathbf{u}_{k-1}, \mathbf{d}_{k-1}) + \mathbf{w}_k = \begin{pmatrix} f_x(u_{k-1}, d_{k-1}) \\ f_y(u_{k-1}, d_{k-1}) \\ f_\phi(u_{k-1}, d_{k-1}) \end{pmatrix} + \begin{pmatrix} w_{[x],k-1} \\ w_{[y],k-1} \\ w_{[\phi],k-1} \end{pmatrix}$$

$$\mathbf{d}_k = \begin{pmatrix} d_{[\Delta D],k} \\ d_{[\Delta\phi],k} \end{pmatrix} + \begin{pmatrix} q_{[\Delta D],k} \\ q_{[\Delta\phi],k} \end{pmatrix}$$

3.1.2 Propagación de la Incertidumbre

La incertidumbre en la localización del robot se relaciona directamente con el error producido por los factores internos y externos que influyen sobre los desplazamientos. Cada vez que el robot realiza un movimiento en su ambiente la incertidumbre sobre su posición aumenta, por lo que es necesario controlar la incertidumbre para corregir la localización y contar información más precisa y confiable.

Es necesario entonces poder representar la incertidumbre para conocer el grado de certeza de la información con que cuenta el robot sobre su localización. Utilizamos un filtro Kalman extendido (EKF) [WEL/03], modelando los dos tipos de desplazamientos que soporta nuestro robot: traslación y rotación, y calculando la incertidumbre de la localización luego de cada movimiento realizado.

El proceso de modelado de la incertidumbre comienza encontrando una relación funcional entre la posición actual del robot $\mathbf{u}_{k-1} = (u_{[x],k-1}, u_{[y],k-1}, u_{[\phi],k-1})^t$, antes de la ejecución de un comando de movimiento, y su nueva posición $\mathbf{u}_k = (u_{[x],k}, u_{[y],k}, u_{[\phi],k})^t$, despues de la ejecución del comando de movimiento. Esta relación se define como:

$$\mathbf{u}_k = \mathbf{f}(\mathbf{u}_{k-1}, \mathbf{d}_{k-1}) + \mathbf{w}_k$$

donde f es la relación funcional entre la posición actual y la anterior.

La incertidumbre se modela como una distribución de probabilidad Gaussiana [KOS/92], definiendo su media $\overline{\mathbf{u}}$ y matriz de covarianza Σ_u. Para actualizar estos parámetros luego de la ejecución de un comando de movimiento, introducimos el sistema $\overline{\mathbf{u}}_k = \mathbf{f}(\overline{\mathbf{u}}_{k-1})$ y $\Sigma_{u_k} = (\frac{\delta \mathbf{f}}{\delta \mathbf{u}})_{k-1} \Sigma_{u_{k-1}} (\frac{\delta \mathbf{f}}{\delta \mathbf{u}})^t_{k-1}$, donde, $\overline{\mathbf{u}}_{k-1}, \Sigma_{u_{k-1}}$ y $\overline{\mathbf{u}}_k, \Sigma_{u_k}$ son, respectivamente, las medias y matrices de covarianza antes y despues de los comandos de movimiento, y $(\frac{\delta \mathbf{f}}{\delta \mathbf{u}})$ es la matriz Jacobiana de la transformación $\mathbf{f}$.

3.1.3 Modelo de Medición y Localización usando el Filtro Extendido de Kalman

Este modelo realiza la estimación de la posición del robot luego de un desplazamiento en el instante k y la representa por un vector de posición llamado $\mathbf{z}_k$. La incertidumbre es representada utilizando la matriz de covarianzas de las variables que definen la localización del robot,

las componentes de la diagonal indican el valor de incerteza, siendo la coordenada cero el valor en el eje X, la coordenada uno en el eje Y y la coordenanda dos en la orientación.

El filtro extendido de Kalman (EKF) define dos pasos: predicción y medición ([WEL/03]). Representamos con el símbolo $^-$ a los valores de las variables obtenidas en el paso de predicción y con el símbolo $^+$ a los valores de variables obtenidas en el paso de corrección, por ejemplo $\widehat{\mathbf{u}}_k^-$ es la posición del robot obtenida en la predicción y $\widehat{\mathbf{u}}_k^+$es la posición del robot obtenida en el paso de corrección.

El primero se compone de dos ecuaciones, la primera predice el próximo estado del sistema luego de la ejecución de una acción (en nuestro caso, la próxima posición luego de un desplazamiento del robot), y la segunda calcula la matriz de covarianza del paso predictivo. De esta forma, el paso predictivo de nuestro modelo de navegación se define como:

$$\begin{aligned} \widehat{\mathbf{u}}_k^- &= \mathbf{f}(\widehat{\mathbf{u}}_{k-1}^+, \mathbf{d}_{k-1}) + \mathbf{w}_{k-1} \\ P_k^- &= A_{\widehat{\mathbf{u}}_{k-1}^+} P_{k-1}^+ A_{\widehat{\mathbf{u}}_{k-1}^+}^T + A_{\mathbf{d}_{k-1}} U_{k-1} A_{\mathbf{d}_{k-1}}^T + W_{k-1} \end{aligned}$$

donde,

$\widehat{\mathbf{u}}_k^-$: predicción de la posición

$\widehat{\mathbf{u}}_{k-1}^+$: posición antes del movimiento

$\mathbf{w}_{k-1}$: ruido del sistema

P_k^-: matriz de covarianza en el paso predictivo

$A_{\widehat{\mathbf{u}}_{k-1}^+}$: matriz Jacobiana de $\mathbf{f}$ con respecto a la posición

P_{k-1}^+: matriz de covarianza

$A_{\mathbf{d}_{k-1}}$: matriz Jacobiana de $\mathbf{f}$, con respecto al desplazamiento relativo

U_{k-1}: matriz de covarianza del movimiento relativo

W_{k-1}: matriz de covarianza del ruido del sistema

Las matrices Jacobianas se definen como:

$$\begin{aligned} A_{\widehat{\mathbf{u}}_{k-1}^+} &= \frac{\delta \mathbf{f}(\widehat{\mathbf{u}}_{k-1}^+, \mathbf{d}_{k-1})}{\delta \widehat{\mathbf{u}}_{k-1}^+} \\ A_{\mathbf{d}_{k-1}} &= \frac{\delta \mathbf{f}(\mathbf{d}_{k-1}, \mathbf{d}_{k-1})}{\delta \mathbf{d}_{k-1}} \end{aligned}$$

Podemos predecir la posición del robot luego de la ejecución de un desplazamiento relativo y la matriz de covarianza del paso predictivo, lo cual nos permite medir la incerteza del sistema.

El segundo paso de EKF (el paso correctivo), definido por tres ecuaciones, se utiliza para corregir los resultados obtenidos en el paso predictivo. El sistema de visión encargado de realizar la medición de la localización del robot, sólo es ejecutado cuando la incertidumbre generada en el paso predictivo supera un umbral estipulado. De esta manera, el proceso con mayor costo computacional es utilizado sólo cuando es necesario. En los casos en que la incertidumbre se mantiene dentro de los umbrales esperados, se utiliza la localización obtenida en el paso predictivo como medición para el paso correctivo.

Definimos $\Phi(P_{k,x}^{-}, umbral)$ como una función que retorna un valor de verdad que indica si la incertidumbre es o no superada. Para conocer si la incertidumbre supera el umbral establecido, se utilizan los valores de la diagonal de la matriz de covarianza P_k^{-}. La variable *umbral* es un vector de tres componentes que son las cotas de la incertidumbre para los ejes x e y y la orientación θ. Entonces la función Φ queda estlecida de esta forma:

$$\Phi(P_k^{-}, umbral) = (P_{k,x}^{-} > umbral_x) \vee (P_{k,y}^{-} > umbral_y) \vee (P_{k,\theta}^{-} > umbral_\theta)$$

donde,

$P_{k,x}^{-} = P_k^{-}[1,1]$: incertidumbre en el eje x,

$P_{k,y}^{-} = P_k^{-}[2,2]$: incertidumbre en el eje y,

$P_{k,\theta}^{-} = P_k^{-}[3,3]$: incertidumbre en la orientación θ,

$umbral = [umbral_x, umbral_y, umbral_\theta]$: vector de umbrales de incertidumbre,

$umbral_x$: cota de incertidumbre para el eje x,

$umbral_y$: cota de incertidumbre para el eje y,

$umbral_\theta$: cota de incertidumbre para la orientación θ,

$\vee$: relación binaria Ó-Lógico

Las ecuaciones para el paso de corrección de EKF son tres: la primera calcula la ganancia de Kalman, la segunda la nueva localización del robot y la tercera la matriz de covarianzas. Se muestran las ecuaciones del paso correctivo:

$$
\begin{aligned}
K_k &= \text{Si } \Phi(P_k^-, umbral) \\
&\quad \text{entonces } P_k^- H_k^T (H_k P_k^- H_k^T - R_k)^{-1} \\
&\quad \text{sino } 0 \\
\widehat{\mathbf{u}}_k^+ &= \widehat{\mathbf{u}}_k^- + K_k(\mathbf{z}_k - \mathbf{h}(\widehat{\mathbf{u}}_k^-)) \\
P_k^+ &= (I - K_k H_k) P_k^-
\end{aligned}
$$

donde,

K_k: ganancia de Kalman en el instante k,

$\Phi(P_k^-, umbral)$: función que indica si la incertidumbre es superada,

$\mathbf{h}(\widehat{\mathbf{u}}_k^-)$: función que predice la medición, en nuestro caso es la función identidad, por lo que $\mathbf{h}(\widehat{\mathbf{u}}_k^-) = \widehat{\mathbf{u}}_k^-$,

H_k: matriz Jacobiana de $\mathbf{h}$ con respecto al desplazamiento relativo en el instante k,

R_k: matriz de covarianza para la medición

La matriz Jacobiana H_k se define como

$$
H_k = \frac{\delta \mathbf{h}(\widehat{\mathbf{u}}_{k-1}^-)}{\delta \widehat{\mathbf{u}}_{k-1}^-} = I
$$

donde I el la matriz identidad de dimensión 3×3. Concluyendo, el paso de corrección se define como:

$$
\begin{aligned}
K_k &= P_k^- H_k^T (H_k P_k^- H_k^T - R_k)^{-1} = \\
&\quad P_k^- (P_k^- - R_k)^{-1}
\end{aligned}
$$

$$
\begin{aligned}
\widehat{\mathbf{u}}_k^+ &= \widehat{\mathbf{u}}_k^- + K_k(\mathbf{z}_k - \mathbf{h}(\widehat{\mathbf{u}}_k^-)) = \\
&\quad \widehat{\mathbf{u}}_k^- + K_k(\mathbf{z}_k - \widehat{\mathbf{u}}_k^-)
\end{aligned}
$$

$$
P_k^+ = (I - K_k H_k) P_k^-
$$

La matriz de ganancia de Kalman es el resultado de una ponderación entre la localización obtenida en el paso predictivo de EKF y la localización obtenida en la medición realizada por el sistema de autolocalización. Se considera el resultado del sistema de autolocalización como verdad absoluta la ganancia sea igual a la matriz Identidad.

Sin embargo, cuando el sistema de autolocalización no es utilizado debido a que la incertidumbre en el sistema es baja, se toma como valor de corrección a la localización calculada en el paso predictivo de EKF.

3.2 Autolocalización: Invariantes proyectivos

La imagen capturada por el robot es una transformación proyectiva del mundo real en el plano de la imagen. Por lo tanto, es posible definir un invariante proyectivo para identificar correspondencias entre objetos en la imagen y objetos en el mundo real, para determinar la posición del robot [KYO/97].

3.2.1 Geometría proyectiva aplicada a Visión por Computadora

El principal resultado de la geometría proyectiva aplicado a problemas de visión por computadoras es la definición de *Invariantes Proyectivos.* Estos invariantes se definen para un conjunto de puntos, líneas, planos y otras curvas algebraicas [HAR/04].

Una transformación proyectiva de un plano P en un plano p es un mapping $h : P \rightarrow p$ de forma tal que si $\{P_i\}$ es un conjunto de puntos colineales en P, entonces $\{h(P_i)\}$ es un conjunto de puntos colineales en p. Para cada transformación proyectiva de planos existe una matríz H no singular de dimensión 3×3, que define unívocamente a la transformación, de forma tal que $p_i = HF_i$, para todo punto P_i del plano P. Las transformaciones proyectivas producen un mapping en perspectiva sobre un plano (por ej, el plano de la imagen). Las líneas paralelas proyectadas sobre el plano de proyección convergen en un punto llamado *punto de fuga (o vanishing point)*, como se muestra en la figura 3-3.

Identificando el plano de proyección de la figura 3-3 como el plano de la imagen de la cámara del robot, podemos ver como el mundo real es proyectado sobre este plano.

Definimos el *cociente cruzado* para cuatro puntos colineales P_i, $i = 1, ..., 4$, como:

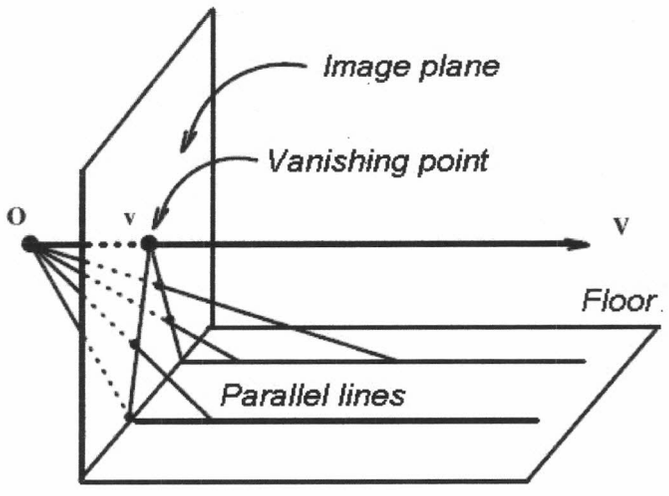

Figura 3-3: Transformación Proyectiva. Observamos la proyección de los puntos del plano del suelo sobre el plano de la imágen.

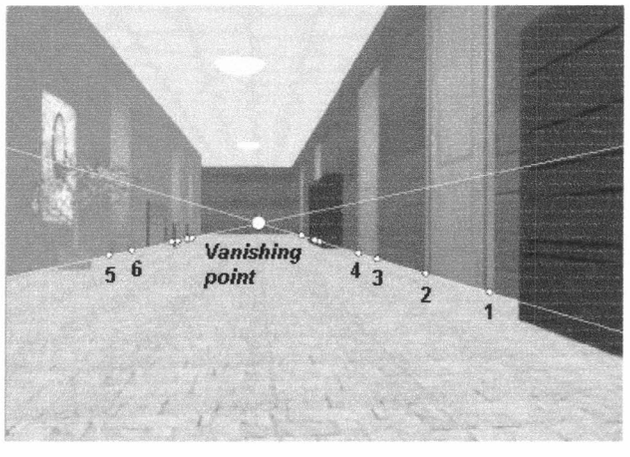

Figura 3-4: Punto de Fuga

$$C_r(P_1, P_2, P_3, P_4) = \frac{|P_3-P_1||P_4-P_2|}{|P_3-P_2||P_4-P_1|}$$

donde $|P_i - P_j|$ es la distancia entre los puntos P_i y P_j. El cociente $C_r(P_1, P_2, P_3, P_4)$ es un invariante proyectivo [HAR/04], por lo que $C_r(P_1, P_2, P_3, P_4) = C_r(p_1, p_2, p_3, p_4)$, donde p_i, $i = 1, ..., 4$, son los puntos colineales del mundo real P_i, proyectados sobre el plano.

Para mostrar la invarianza del cociente cruzado, primero debemos exponer el efecto de una transformación proyectiva sobre las coordenadas de los puntos de una línea. Cualquier punto en una línea puede ser representado solo con dos coordenadas homogéneas. Esto es, $P_i = (X_{1,}X_2)^t$ donde X_1 y X_2 son coordenadas homogéneas de la línea. La posición cartesiana de un punto P_i en la línea está dada por $X_{P_i} = X_1/X_2$. La transformación proyectiva entre líneas se reduce a una matríz H homogénea de 2x2, de forma tal que $p_i = HF_i$, donde $p_i = (x_1, x_2)^r$ son las coordenadas homogéneas del punto p_i en la línea transformada. La posición cartesiana de punto p_i en la línea transformada es $x_{p_i} = x_1/x_2$. Por lo tanto, $x_{p_i} = \frac{h_{11}X_{P_i}+h_{12}}{h_{21}X_{P_i}+h_{22}}$, donde h_{ij} es el valor de la fila i y columna j de la matríz de transformación H. Luego,

$C_r(p_1, p_2, p_3, p_4) = \frac{|p_3-p_1||p_4-p_2|}{|p_3-p_2||p_4-p_1|} =$

$$= \frac{(\frac{(h_{11}X_{P_1}+h_{12})}{(h_{21}X_{P_1}+h_{22})} - \frac{(h_{11}X_{P_3}+h_{12})}{(h_{21}X_{P_3}+h_{22})})(\frac{(h_{11}X_{P_4}+h_{12})}{(h_{21}X_{P_4}+h_{22})} - \frac{(h_{11}X_{P_2}+h_{12})}{(h_{21}X_{P_2}+h_{22})})}{(\frac{(h_{11}X_{P_2}+h_{12})}{(h_{21}X_{P_2}+h_{22})} - \frac{(h_{11}X_{P_3}+h_{12})}{(h_{21}X_{P_3}+h_{22})})(\frac{(h_{11}X_{P_4}+h_{12})}{(h_{21}X_{P_4}+h_{22})} - \frac{(h_{11}X_{P_1}+h_{12})}{(h_{21}X_{P_1}+h_{22})})} =$$

$$= \frac{(h_{11}h_{22}-h_{12}h_{21})(X_{P_1}-X_{P_3})(h_{11}h_{22}-h_{12}h_{21})(X_{P_4}-X_{P_2})}{(h_{11}h_{22}-h_{12}h_{21})(X_{P_1}-X_{P_2})(h_{11}h_{22}-h_{12}h_{21})(X_{P_4}-X_{P_3})} =$$

$$= \frac{(X_{P_1}-X_{P_3})(X_{P_4}-X_{P_2})}{(X_{P_1}-X_{P_2})(X_{P_4}-X_{P_3})} = \frac{|P_3-P_1||P_4-P_2|}{|P_3-P_2||P_4-P_1|} = C_r(P_1, F_2, F_3, F_4)$$

El concepto de invarianza en la proyección de líneas puede ser extendido a invarianza en la proyección de planos. Considerando los puntos coplanares P_i, $i = 1, ..., 5$,

$$I_1 = I_1(P_1, ..., F_5) = \frac{|P_5, F_2, F_3|.|P_4, F_1, F_2|}{|P_4, F_2, F_3|.|P_5, F_1, F_2|} \tag{3.2}$$

$$I_2 = I_2(P_1, ..., F_5) = \frac{|P_5, F_3, F_1|.|P_4, F_1, F_2|}{|P_4, F_3, F_1|.|P_5, F_1, F_2|} \tag{3.3}$$

donde $|P_i, F_j, F_k|$ es el determinante de la matriz de 3×3 cuyas columnas son las coordenadas de los puntos P_i, F_j y P_k. Estos invariantes de planos son utilizados para determinar la posición

del robot, como se describe en la sección 3.2.2.

3.2.2 Método de localización del robot

El sistema de navegación asume que existen en el pasillo dos líneas paralelas formadas por la unión entre el piso y las paredes laterales. Los puntos formados por la intersección entre el piso y los marcos de las puertas los denominamos *puntos característicos*. Estos puntos colineales se usan para calcular los cocientes cruzados. Para cada subconjunto de cuatro puntos característicos del ambiente donde navegará el robot, se calcula el cociente cruzado correspondiente y se almacena un registro en la memoria del robot. Este registro relaciona el valor del cociente cruzado con las posiciones dentro del pasillo de los puntos característicos utilizados para su cálculo. El conjunto de todos estos registros conforman la base de datos que usará el robot para el proceso de matching entre los puntos característicos detectados en la imagen y los puntos característicos del pasillo, como se expone a continuación.

Matching de puntos colineales

El primer paso del método de localización es el reconocimiento de los puntos característcos. En la sección 3.2.3, exponemos el algoritmo implementado para identificar estos puntos en la imagen. En la figura 3-4, se observan los puntos característicos reconocidos en una imagen de un pasillo. Una vez que los puntos p_1, p_2, p_3 y p_4 son reconocidosen la imagen, se calcula el valor $C_r(p_1, p_2, p_3, p_4)$, el cual se usa para buscar en la base de datos el conjunto de puntos característicos P_1, P_2, P_3 y P_4del pasillo que se corresponde con los puntos reconocidos en la imagen.

Auto-localización

Sean $P_i = (X_i, Y_i)^t$, $i = 1, ..., 5$, las coordenadas inhomogéneas de cinco puntos coplanares en el plano del suelo. Los puntos P_1 y P_2 son 2 puntos de la pared derecha, mientras que P_3 y P_4 son dos puntos de la pared izquierda. Además, asumimos que P_5 es la posición del robot en el piso del pasillo. Las ecuaciones (3.2) y (3.3) pueden ser escritas de la siguiente manera:

$$AX_5 - BY_5 = -C \tag{3.4}$$

$$DX_5 - EY_5 = -F \tag{3.5}$$

donde:

$$A = \frac{|P_4, F_2, F_3|}{|P_4, F_1, F_2|}(Y_1 - Y_2) - (Y_2 - Y_3)$$

$$D = \frac{|P_4, F_3, F_1|}{|P_4, F_1, F_2|}(Y_1 - Y_2) - (Y_3 - Y_1)$$

$$B = \frac{|P_4, F_2, F_3|}{|P_4, F_1, F_2|}(X_1 - X_2) - (X_2 - X_3)$$

$$E = \frac{|P_4, F_3, F_1|}{|P_4, F_1, F_2|}(X_1 - X_2) - (X_3 - X_1)$$

$$C = \frac{|P_4, F_2, F_3|}{|P_4, F_1, F_2|}(X_1Y_2 - X_2Y_1) - (X_2Y_3 - X_3Y_2)$$

$$F = \frac{|P_4, F_3, F_1|}{|P_4, F_1, F_2|}(X_1Y_2 - X_2Y_1) - (X_3Y_1 - X_1Y_3)$$

Resolviendo el sistema formado por las ecuaciones (3.4) y (3.5), las coordenadas (X_5, Y_5) correspondientes a la posición del robot P_5 , son determinadas unívocamente.

Como se expone en la sección 3.2.1, sabemos que

$$\begin{aligned} I_1(P_1, ..., F_5) &= I_1(p_1, ..., p_5) \\ I_2(P_1, ..., F_5) &= I_2(p_1, ..., p_5) \end{aligned}$$

donde $p_i = (x_i, y_i)^t$, con $i = 1, ..., 5$, son las coordenadas inhomogeneas de los puntos P_i proyectados en el plano de la imagen. Luego, para resolver el sistema de ecuaciones, debemos definir cual es el punto p_5 dentro del plano de la imagen. La figura 3-5 ayuda a ver cómo considerar la coordenada y_5 del punto $p_5 = (x_5, y_5)$. Considerando la inclinación de la cámara del robot, la figura 3-5 muestra las coordenadas de la cámara del robot (x_5, y_5) en el plano de la imagen. Si el eje óptico de la cámara es paralelo al suelo, podemos asumir que y_5 es infinito. Luego, siendo $p_i = (x_i, y_i)$, sabemos que:

$I_1 = I_1(p_1, ..., p_5) = \frac{|p_5,p_2,p_3|.|p_4,p_1,p_2|}{|p_4,p_2,p_3|.|p_5,p_1,p_2|} = \frac{|p_4,p_1,p_2|.[x_5(x_2-x_3)-y_5(y_2-y_3)+(x_2y_3-x_3y_2)]}{|p_4,p_2,p_3|.[x_5(x_1-x_2)-y_5(y_1-y_2)+(x_1y_2-x_1y_2)]}$

Por lo tanto, asumiendo que el eje de visión de la cámara es horizontal, podemos hacer la

siguiente aproximación:

$x_5(x_2 - x_3) - y_5(y_2 - y_3) + (x_2y_3 - x_3y_2) \approx -y_5(y_2 - y_3)$

$x_5(x_1 - x_2) - y_5(y_1 - y_2) + (x_1y_2 - x_1y_2) \approx -y_5(y_1 - y_2)$

Entonces,

$I_1 = I_1(p_1, ..., p_5) = \frac{|p_5,p_2,p_3|.|p_4,p_1,p_2|}{|p_4,p_2,p_3|.|p_5,p_1,p_2|} \approx \frac{|p_4,p_1,p_2|.[-y_5(y_2-y_3)]}{|p_4,p_2,p_3|.[-y_5(y_1-y_2)]} = \frac{|p_4,p_1,p_2|.(y_2-y_3)}{|p_4,p_2,p_3|.(y_1-y_2)}$

Haciendo un análisis similar para I_2, vemos que podemos aproximar, I_1 y I_2 por:

$$I_1(P_1, ..., F_5) = I_1(p_1, ..., p_5) \approx \frac{|p_4,p_1,p_2|}{|p_4,p_2,p_3|} \frac{(x_2-x_3)}{(x_1-x_2)}$$
$$I_2(P_1, ..., F_5) = I_2(p_1, ..., p_5) \approx \frac{|p_4,p_1,p_2|}{|p_4,p_3,p_1|} \frac{(x_3-x_1)}{(x_1-x_2)}$$

Los valores aproximados de I_1 y I_2 pueden usarse para resolver el sistema de las ecuaciones 3.4 y 3.5, obteniendo la posición del robot (X_5, Y_5), utilizando el matching entre un conjunto de cuatro puntos característicos del suelo y un conjunto de cuatro puntos de la imagen.

Estimación de la Orientación del Robot

En la sección anterior se expuso como obtener la posición del robot, a partir de la imagen capturada por la cámara del mismo. Pero para navegar por el pasillo este dato no es suficiente. Dado que es necesario ejecutar los comandos de movimientos necesarios para que el robot pueda corregir su trayectoria y trasladarse hacia su objetivo, es preciso conocer su orientación, para definir el movimiento de giro adecuado para prosegir su camino.

Si bien hemos implementado el método basado en *Invariantes Proyectivos* propuesto en [KYO/97], este trabajo no presenta un método para estimar la orientación del robot. Por esto, hemos extendido la aplicación del matching de cocientes cruzados para lograr estimar la orientación del robot.

En la imagen izquierda de la figura 3-6, observamos un ejemplo de detección de puntos característicos y su identificación con puertas del pasillo, realizada de acuerdo a lo expuesto en la sección 3.2.1. Luego de la localización de las coordenadas del robot, conocemos los puntos del pasillo P_i, $i = 1, ..., 4$, que se corresponden con los puntos p_1, p_2, p_3 y p_4, de la imagen del robot.

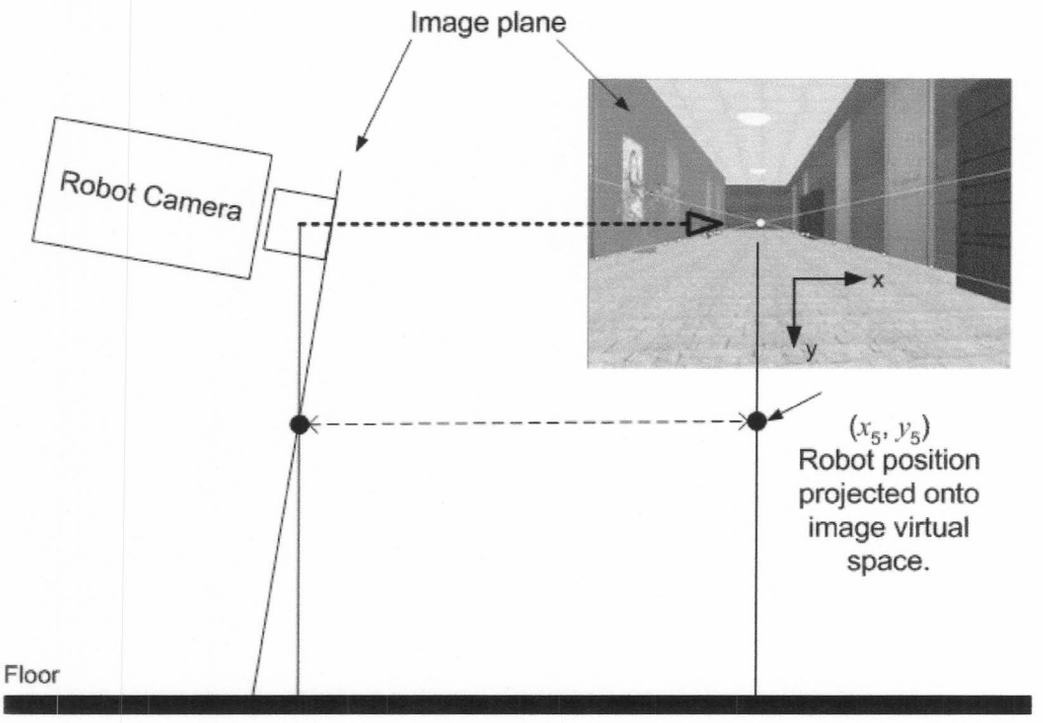

Figura 3-5: Proyección de la posición del robot en el plano de la imagen. Podemos considerar que el punto (x_5, y_5) está en el plano de la imágen del robot, aunque está fuera de la sección visible de este plano. Este punto es la posición del robot en el plano de la imágen. Se observa que disminuyendo la inclinación de la cámara, el punto (x_5, y_5), se aleja hacia abajo. Si la cámara se instala horizontalmente, podemos asumir que y_5 es infinito.

Ahora observemos por un momento el diagrama de la parte derecha de la figura 3-6. La posición localizada del robot se muestra en esta figura como $P_5 = (X_5, Y_5)$. Si logramos estimar la distancia A, entre la coordenada Y_5 de la posición del robot, y el punto de la pared lateral del pasillo que se encuentra justo en frente del robot, podríamos estimar la orientación del robot Or como $Or = \arctan(\frac{X_5}{A})$.

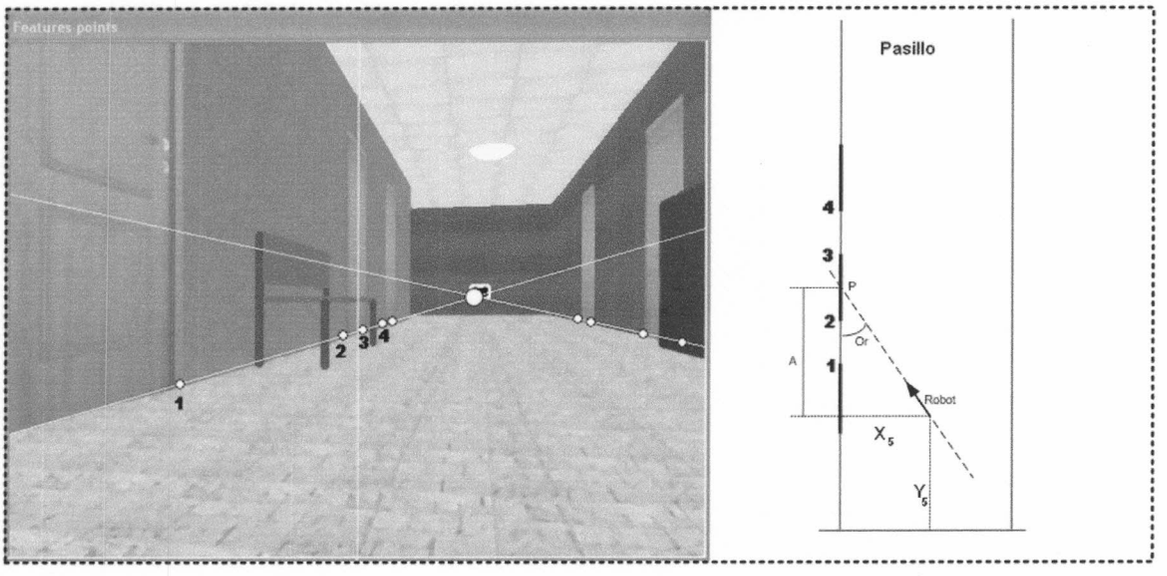

Figura 3-6: Estimación de la Orientación del Robot. Una vez obtenida la posición del Robot (X_5, Y_5), la orientación del mismo es $Or = \arctan(\frac{X_5}{A})$.

El robot modelado cuenta con una única cámara apuntando hacia el frente del robot y, como fue expuesto en 3.2.1, la imagen capturada por el robot consiste en una transformación proyectiva de los elementos del ambiente, sobre el plano de la imagen. Por esto, sabemos que el punto P del diagrama de la figura 3-6 será proyectado horizontalmente en el centro de la imagen, marcado con la línea vertical amarilla en la figura 3-6. También sabemos que el punto P se proyectará en la intersección de la línea vertical que marca el centro de la imagen, con la línea que marca el límite entre el suelo del pasillo y la pared del mismo. Llamaremos p a este punto de la imagen, obtenido luego de la determinación de la posición del robot. Para definir que línea de pasillo utilizar, simplemente observamos si el vanishing point se encuentra corrido hacia la derecha del centro de la imagen, en cuyo caso se utiliza el lateral izquierdo del pasillo,

o el lateral derecho en caso contrario. Si el vanishing point se encuentra exactamente en el centro de la imagen, podemos asumir que el robot se encuentra alineado en forma paralela a las paredes del pasillo.

Sabemos que en el paso de localización fue realizada la identificación $C_r(P_1, F_2, F_3, F_4) = C_r(p_1, p_2, p_3, p_4)$, donce C_r es el cociente cruzado definido en la sección 3.2.1. Si reemplazamos uno de los puntos de la imagen en esta ecuación, por ejemplo el 3, tenemos que $C_r(P_1, F_2, P, F_4) = C_r(p_1, p_2, p, p_4)$, obteniendo de esta forma una ecuación con una única incógnita, que corresponde al punto P del diagrama. Simplemente despejando P, podemos estimar que $A = P_y - Y_5$, donde P_y el la componente longitudinal al pasillo del punto P. Luego, solo queda estimar la orientación como $Or = \arctan(\frac{X_5}{A})$.

Estimación del Fin del Pasillo

En las secciones anteriores vimos como estimar la posición robot dentro del pasillo, así como su orientación. El método utilizado y los features points definidos para su aplicación, imponen una restricción sobre la cantidad de puertas que deben encontrarse dentro del campo visual del robot, para que la localización sea efectiva. Inevitablemente al avanzar lo suficiente en el pasillo, se acabarán las puertas del mismo, con lo cual el robot se queda sin referencias para auto-localizarse. Para que el robot pueda recorrer el último segmento del pasillo aunque no tenga más features points de referencia, introdujimos una marca artificial al final del pasillo. La forma de la marca definida puede verse en la figura 3-7.

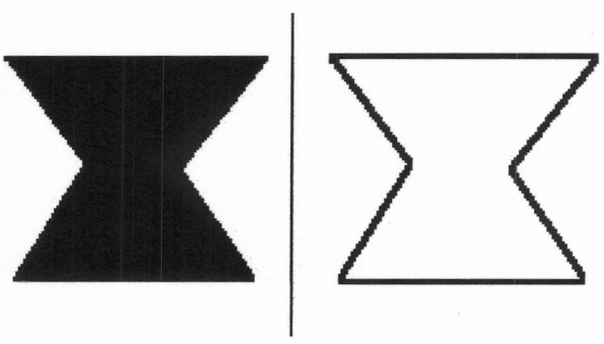

Figura 3-7: Izquierda: Marca de Fin de Pasillo. Derecha: Silueta de la Marca.

En la figura 3-8, se observa la ubicación de la marca artificial en el ambiente de navegación. Cuando el robot no consigue localizarse por ausencia de features points, busca la marca de fin de pasillo en la imagen capturada por la cámara. Al encontrarla, realiza pequeños giros hasta que la misma se posicione en el centro de la imagen, con lo cual se asegura que la marca se encuentra frente a él. Luego, mediante traslaciones acotadas y giros de ajuste para mantener la marca al frente y dentro de su campo de visión, el robot se traslada en dirección a ella, hasta que su tamaño crezca lo suficiente dentro de la imagen capturada. Cuando el tamaño de la marca alcanza un límite específico, el robot asume que se encuentra en el final del pasillo.Al llegar, éste realiza un giro de 90^o hacia la derecha o hacia la izquierda, dependiendo de la topología de los pasillo, la cual se encuentra almacenada en su base de datos, según se expone en la sección 5.1.5.

El reconocimiento de la marca fue realizado mediante Descriptores de Fourier, cuya utilización es bien conocida para la clasificación de formas ([SHA/02]). La forma de la marca de fin de pasillo es analizada a partir de su silueta, la cual observamos en la figura 3-7. La silueta es una curva de Jordan, parametrizada por su longitud. Construimos la curva recorriendo los pixels de la silueta en una dirección determinada, hasta volver al pixel inicial. El k-ésimo pixel corresponderá a la posición (x_k, y_k), de manera que podemos describir al contorno o silueta con dos ecuaciones paramétricas:

$x(k) = x_k$

$y(k) = y_k$

Luego, tomando la transformada de Fourier de cada función, obtenemos el descriptor de Fourier $(a_x, a_y)_v$:

$a_x(v) = F(x(k))$

$a_y(v) = F(y(k))$

Analizando las siluetas que se pueden encontrar en la imagen capturada por la cámara del robot, y calculado los descriptores de Fourier de las mismas, podemos decidir si alguna de ellas corresponde a la marca de fin de pasillo, comparando sus descriptores ([SHA/02])

Para facilitar la extracción de la marca de fin de pasillo de la imagen del robot, la definimos de color negro, sobre un fondo blanco, como se observa en la figura 3-7, De esta forma, simple-

mente aplicando un umbral sobre la intensidad de cada pixel, obtenemos una imagen binaria con la marca del pasillo bien definida. En la figura 3-8 se observa el resultado del umbralado de la imagen. En la imagen resultante se observarán secciones en blanco, una de las cuales corresponderá a la marca de fin de pasillo, si la misma se encuentra en el campo de visión del robot. Dependiendo como sea la imagen capturada por el robot, es posible que aparezcan más de una sección blanca en la imagen resultante si, por ejemplo, hubieran otros objetos negros o muy oscuros en el pasillo. Luego de esto, obtenemos las siluetas de la secciones en blanco de la imagen umbralada, mediante la extracción de los contornos de las mismas. Para identificar cual de los contornos identificados corresponde a la marca de fin de pasillo, utilizamos Descriptores de Fourier.

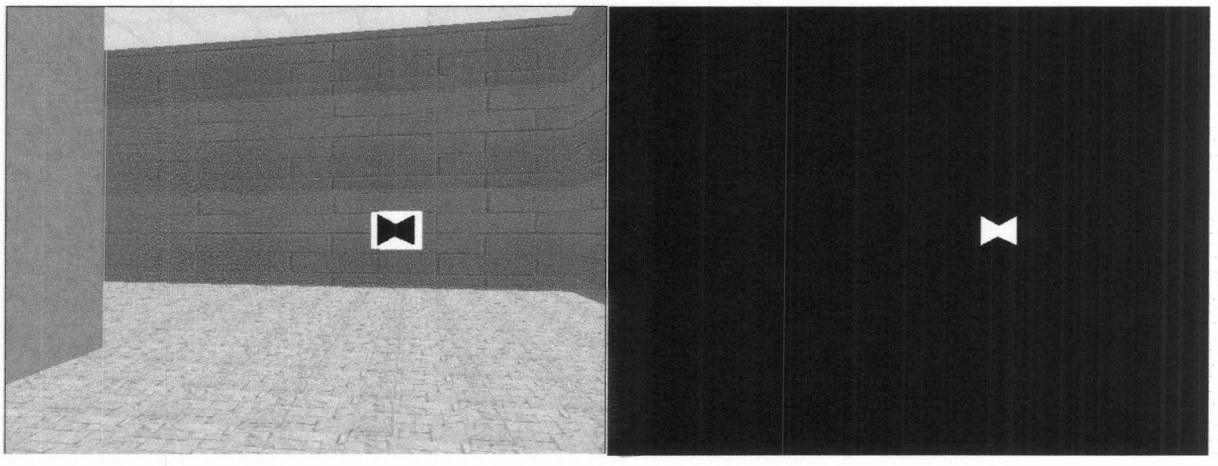

Figura 3-8: Izquierda: Imágen original con Marca de Fin de Pasillo. Derecha: Resultado de la aplicación del umbral para detectar la marca. Como la marca de fin de pasillo es negra sobre un fondo blanco, aplicando un umbral de baja intensidad, se detecta claramente la marca de fin de pasillo.

Los descriptores de Fourier fueron elegidos para identificar la marca de fin de pasillo por su capacidad para clasificar formas pero, no menos importante, por sus propiedades de invarianza. Ignorando el primer término del descriptor de Fourier, el mismo es invariante a la traslación y a la rotación. Para nuestro caso, la invarianza a la rotación no es importante dado que la cámara se encuentra fija en el robot y por lo tanto no se observarán rotaciones importantes en la marca dentro de la imagen, salvo alguna irregularidad en el suelo que produzca que el robot

no mantenga una posición vertical perfecta. Por el contrario, la invarianza de la traslación es fundamental ya que no es posible predecir en que posición dentro de la imagen se encontrará la silueta de la marca. Otra propiedad de invarianza fundamental para el éxito de esta estrategia es la invarianza a escalado. Esto significa que la silueta es reconocida independientemente de su tamaño, solo la forma es considerada. Si modificamos la silueta sólo en su tamaño, el efecto en el descriptor de Fourier correspondiente es la multiplicación por un escalar. Por lo tanto, normalizando los descriptores de Fourier, obtenemos un método invariante a diferencias de tamaño. Esto también es importante, ya que el robot buscará la marca desde diferentes posiciones en el pasillo y la misma será más grande o más pequeña, dependiendo de la distancia entre el robot y la marca. En la figura 3-9 podemos observar el reconocimiento de la marca desde dos posiciones diferentes, con la marca ubicada en distintos lugares de la imagen, y con el tamaño de la marca visiblemente diferente.

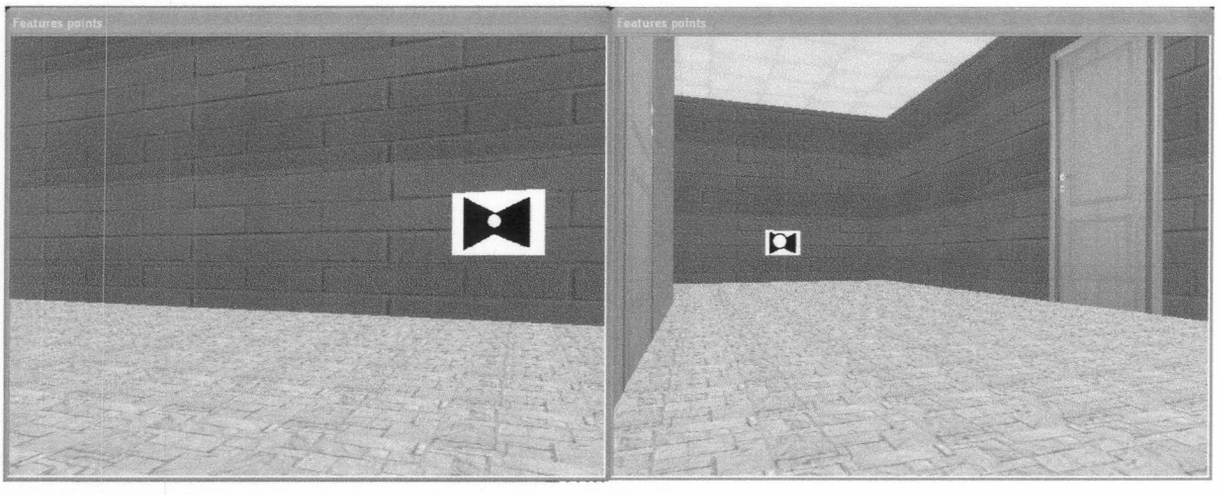

Figura 3-9: Detección de la Marca de Fin de Pasillo desde diferentes ángulos y con distintos tamaños en la proyección de la marca.

3.2.3 Reconocimiento de Puntos Característicos

El método de reconocimiento de puntos característicos se presenta en el siguiente Algoritmo:

Algoritmo de Reconocimiento de Puntos Característicos

1. *Segmentación de la Imagen*
2. *Generación del Mapa de Bordes*
3. *Reconocimiento del Punto de Fuga*
4. *Reconocimiento de los Límites del Pasillo*
5. *Reconocimiento de Puertas*
6. *Extracción de los Puntos Característicos*

Figura 3-10: Pasillo Simulado

Segmentación de la Imagen

El objetivo del algoritmo es distinguir los puntos característicos del pasillo, por lo que es necesario identificar las áreas de la imagen que corresponden a las puertas y al suelo, ya que la intersección de los límites laterales de las puertas con el suelo son los puntos que buscamos. Por lo tanto, el primer paso del algoritmo consiste en la segmentación de la imagen para separar las áreas que corresponden al suelo y a las puertas. Los puntos de la imagen que correspondan a cualquier otro elemento del pasillo, por ejemplo el techo o las paredes, no son de nuestro interés por lo que son segmentados como un único componente.

La segmentación fue implementada mediante el método de *Nearest Neighborhood Clustering*, asumiendo que la distribución de colores de las áreas del suelo y las puertas siguen una distribución normal. En la actualidad existen muchas alternativas bien estudiadas para el problema de la segmentación de una imagen, pero no hemos explorado otros métodos ya que este no es el foco de nuestro trabajo. El método de clustering que implementamos se comporta bien en el entorno de simulación implementado, lo cual nos permitió testear la estrategia de navegación definida.

Para realizar la segmentación, asumimos que el color (r,g,b) de cada pixel de la imagen se corresponde con un punto en el espacio euclideano, con lo cual utilizamos como medida de cercanía en el método de *Nearest Neighborhood* la distancia euclideana entre los píxeles de la imagen y los puntos en el espacio que caracterizan a las puertas y al suelo. Si la distancia de un pixel al punto representativo de las clases "suelo" y "puertas" es mayor a un determinado umbral, asumimos que el punto no pertenece a ninguna de estas clases. La obtención de los colores que caracterizan las áreas del suelo y las puertas son obtenidos utilizando muestras de las texturas que se utilizan en el renderizado del pasillo. La utilización de texturas en el renderizado del ambiente se expone en la sección 4.1. Para las texturas del suelo y de las puertas se calculan la media y la varianza de los colores de los pixeles que las componen. Para implementar este método en un robot físico, será necesario tomar imagenes del suelo y de las puertas, utilizando la cámara del robot, con el objetivo de usarlas como muestras, en forma análoga a como usamos las texturas en la simulación.

La utilización del método *Nearest Neighborhood* produce algunos puntos mal clasificados en áreas cuyos colores no presentan una distribución acorde a la modelada, o si la muestra utilizada para obtener los parámetros de la distribución no es lo suficientemente representativa. En la figura 3-11 se observa el resultado del la segmentación de una imagen renderizada por el simulador, donde también se observan algunos puntos mal clasificados en el área del suelo, luego de la segmentación de la imagen. Para minimizar este efecto y producir una mejora, el sistema procesa esta imagen mediante operadores morfológicos, utilizando un disco como elemento estructurante. Si observamos con cuidado la imagen de la figura 3-11 vemos que dilatando la imagen sería posible hacer desaparecer la mayoría de los puntos mal clasificados. Pero como no es deseable que se modifique el área de las puertas, ya que podría modificar la

ubicación de los puntos característicos, hemos aplicado el operador de cerradura con el cual obtuvimos buenos resultados en la eliminación los puntos espúreos. El tamaño del elemento estructurante puede variar para distintas texturas. En la figura 3-12, se observa el resultado de aplicar el operador morfológico de cerradura a la imagen de la figura 3-11. Se observa una clara mejora en el "ruido" producido por los puntos mal clasificados.

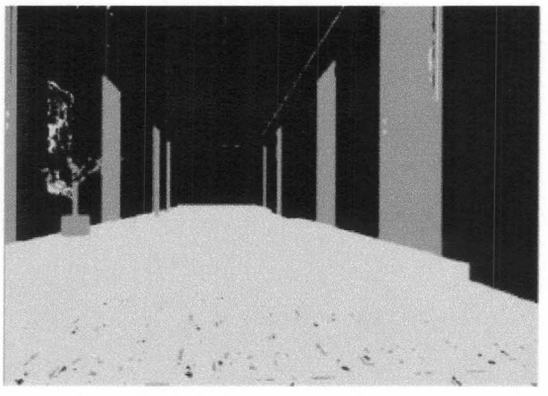

Figura 3-11: Clasificación inicial

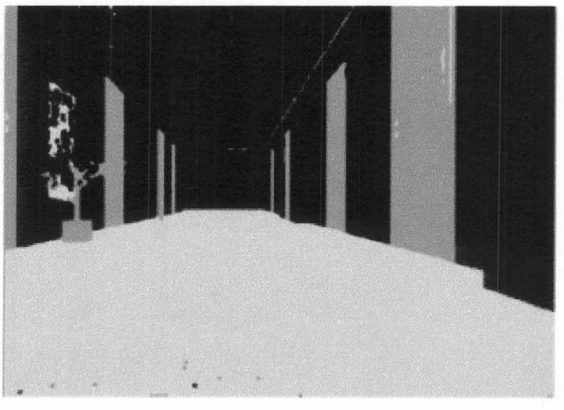

Figura 3-12: Aplicación del Operador Morfológico de Cerradura

Generación del Mapa de Bordes

Un Mapa de Bordes de una imagen es una colección de puntos que marcan qué pixels de la imagen corresponden a bordes de objetos presentes en la misma. Podemos representar a un mapa de bordes con un imagen con fondo negro y con los puntos correspondientes a los bordes marcados en blanco, como se observa en la figura 3-13. Nuestro algoritmo obtiene el Mapa de Bordes para identificar las líneas rectas que nos permitan llegar a reconocer los puntos característicos de la imagen. La detección de las líneas rectas se realiza en un paso posterior del algoritmo, mediante la Transformada de Hough, la cual se expone en el apéndice B. La generación del Mapa de Bordes se realiza mediante el *Operador de Canny [CAN/86],* aplicado a la imagen segmentada obtenida en el paso anterior del algoritmo. Es posible obtener el Mapa de Bordes desde la imagen original en lugar de la segmentada, pero esto produciría una cantidad mayor de bordes, detectados por causa de las texturas de los elementos del pasillo (puertas, suelo, etc) o elementos en el area de las paredes (cuadros, por ej). Estos bordes estarían ubicados en areas de la imagen que no son de nuestro interés y podrían generar líneas rectas no deseadas al aplicar la Transformada de Hough.

El proceso de histéresis propio del Operador de Canny produce contornos más contínuos y minimiza el ruido en la imagen resultante, lo que ayuda a disminuir la cantidad de líneas espúreas que puede producir la Transformada de Hough (ver apéndice B). En la figura 3-13 podemos observar el Mapa de Bordes obtenido de la imagen segmentada observada en la figura 3-12.

Reconocimiento del Punto de Fuga

La *Transformada de Hough,* descripta en el apéndice B , es utilizada sobre la imagen del Mapa de Bordes para identificar las líneas que definen los bordes laterales del pasillo. Una de las características de la Transformada de Hough es que genera líneas espúreas. Si bien este efecto puede optimizarse seleccionando adecuadamente el umbral de votación y las celdas de acumulación, no es posible eliminarlo totalmente. Para mejorar el reconocimiento de líneas, hemos implementado una segunda pasada de la Transformada de Hough, haciendo que cada pixel vote sólo por la mejor línea a la cual pertenece, lo cual permite disminuir la cantidad de líneas espúreas detectadas. En la figura 3-14 se ilustra la mejora obtenida mediante la segunda

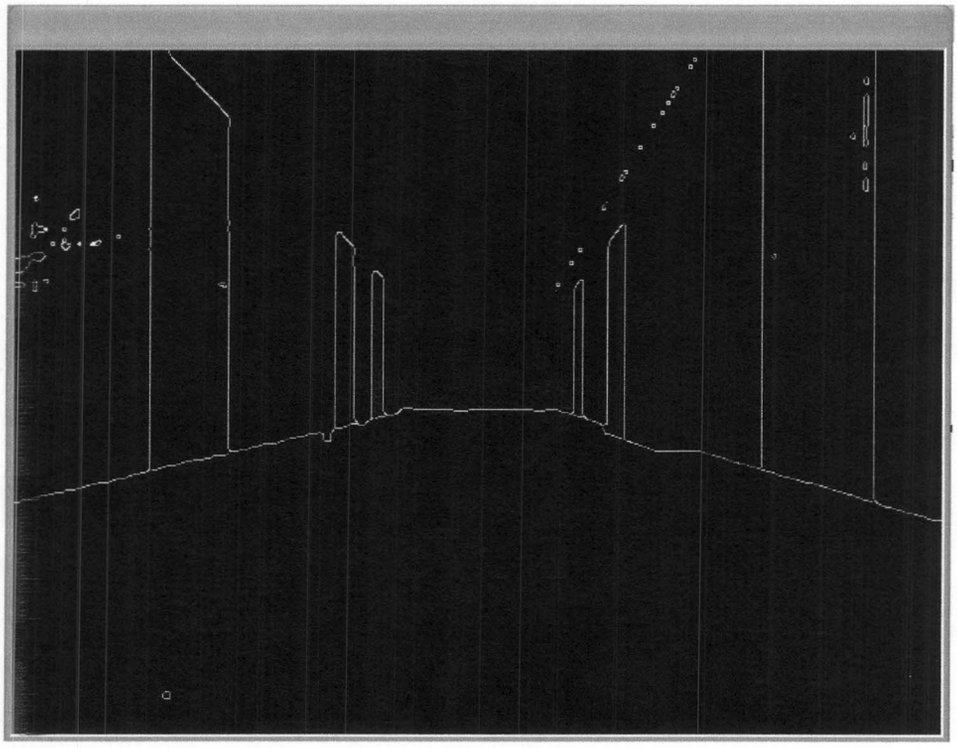

Figura 3-13: Mapa de Bordes obtenido con el Operador de Canny. Los pixels en blanco corresponden a los bordes detectados en la imágen original.

pasada.

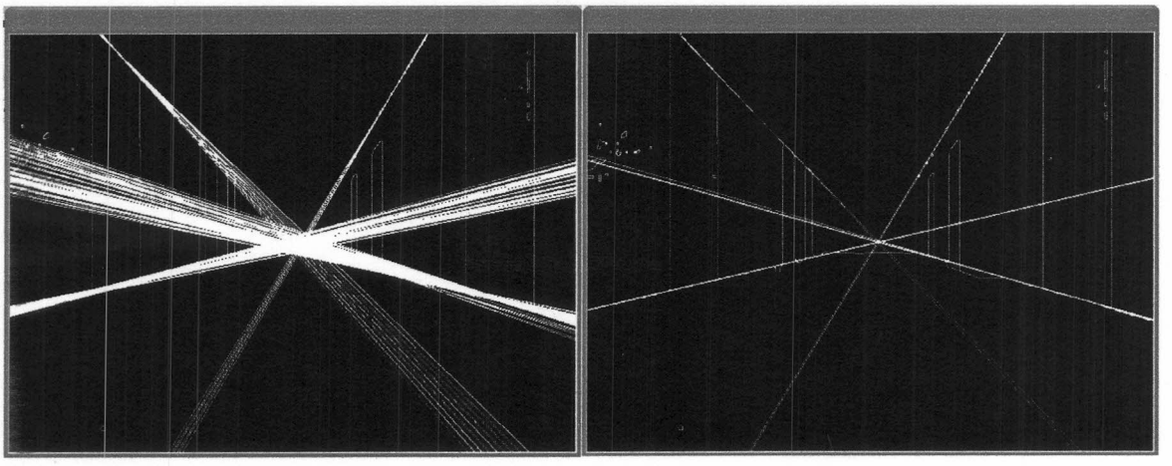

Figura 3-14: Izquierda: Primera pasada de Houg, con líneas espúreas. Derecha: eliminación de líneas espúreas con una segunda pasada.

Como sabemos que las líneas del pasillo se intersectan en el punto de fuga, las líneas horizontales y verticales pueden descartarse del espacio de Hough.. El punto de fuga se determina como el centroide del conjunto de puntos compuesto por todas las intersecciones de las líneas identificadas. La utilización del centroide permite ubicar al punto de fuga en el punto con alta densidad de intersecciones. Con esto, estamos ubicando al punto de fuga cerca del lugar donde se intersectan la mayoría de las líneas detectadas.. En la figura 3-15 se ilustra la identificación de las líneas laterales del pasillo y la detección del Punto de Fuga.

Reconocimiento de los Límites del Pasillo

Tomando las líneas encontradas en el paso anterior del algoritmo, tenemos un conjunto de líneas rectas candidatas para describir los límites laterales del pasillo. El sistema implementado busca, en cada lateral del pasillo, la línea que pasa más cerca del punto de fuga detectado, y la considera como el límite entre el piso y la pared correspondiente del pasillo. En caso de encontrar más de una línea que pasa exactamente por el punto de fuga, se toma la más votada en el espacio de Hough.

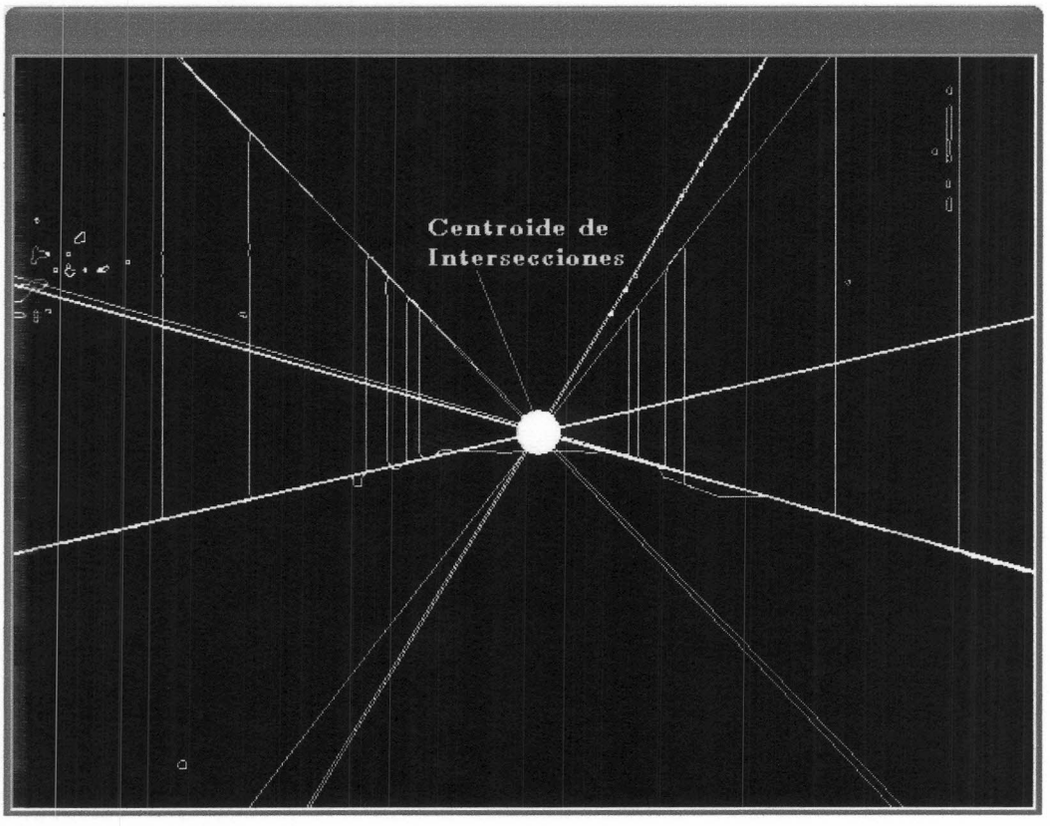

Figura 3-15: Punto de Fuga. El centro del círculo blanco es el centroide del conjunto de puntos de intersección entre las rectas laterales.

Dado que sólo necesitamos extraer dos líneas, correspondientes a la pared derecha e izquierda, se selecciona la mejor línea con pendiente negativa y la mejor línea con pendiente positiva, respectivamente. El reconocimiento de estas líneas permite mejorar la segmentación de la imagen, ya que el área del piso queda totalmente delimitada, y podemos re-clasificar los pixeles que se encuentren fuera del area correspondiente. Sabemos que toda el área de la imagen que se encuentre por debajo de las líneas que decriben los límite del pasillo corresponde al suelo. Por lo tanto los pixels que se encuentren por debajo de estas líneas, que no hayan sido clasificados como parte del suelo, pueden reclasificarse. La figura 3-16 muestra el reconocimiento de los límites y la mejora en la segmentación, que puede observarse comparando esta figura con la figura 3-12.

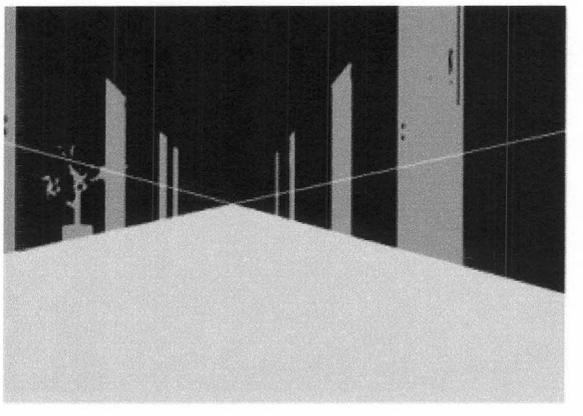

Figura 3-16: Punto de Fuga

Reconocimiento de Puertas

En este paso aplicamos nuevamente la Transformada de Hough, buscando solo líneas verticales, para identificar los marcos de las puertas. Dado que, como se ha descripto en el paso anterior del algoritmo, se obtiene una mejora de la segmentación de la imagen luego de identificar el punto de fuga y los bordes laterales del pasillo, volvemos a aplicar el operador de Canny sobre la segmentación mejorada, para obtener mejores resultados con la Transformada de Hough. Estas mejoras sucesivas en la detección de las líneas rectas, permiten encontrar los Puntos Característicos, con la consecuente mejora en la precisión del proceso de auto-localización. La

figura 3-17 muestra un ejemplo de líneas verticales encontradas en una imagen segmentada.

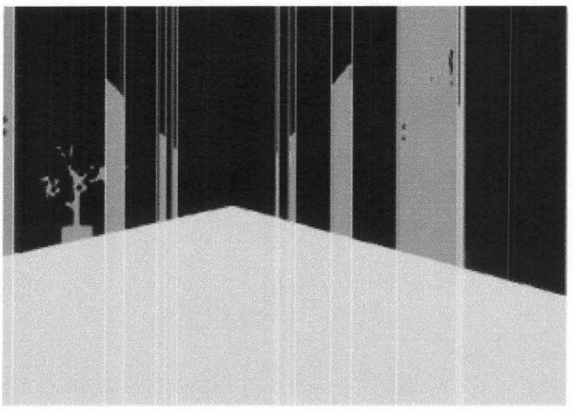

Figura 3-17: Lineas Rectas Verticales

Extracción de Puntos Característicos

Finalmente, el algoritmo obtiene la intersección entre las líneas del pasillo y las líneas verticales para determinar los puntos característicos. El resultado se muestra en la figura 3-18.

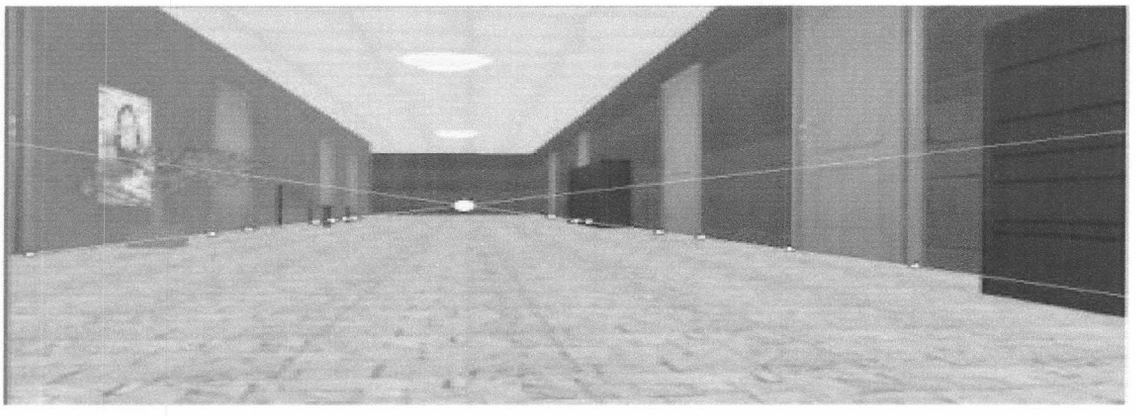

Figura 3-18: Puntos Característicos detectados. Se observan como pequeños círculos blancos los puntos característicos detectados. La imágen de la izquierda los muestra sobre las rectas detectadas por Hough, mientras que la imágen de la derecha los muestra sobre la imágen original.

Capítulo 4

Simulación del Ambiente de Navegación

Los experimentos con el sistema de navegación se realizaron en un simulador que genera imágenes de un pasillo virtual. La figura 3-10 (Pag. 33) muestra un ejemplo de una imagen generada por el simulador.

4.1 Modelado del Pasillo

Para la construcción del simulador sobre el cual validamos la estrategia de navegación, elegimos OpenGL y GLUT (OpenGL Utility Toolkit) [WWW/02]. OpenGL es una librería standard para la creación de aplicaciónes 2D y 3D. Es multi-plataforma y dado su nivel de estabilidad y evolución es utilizada ampliamente como herramienta base para la construcción de simuladores. GLUT es una herramienta que permite construir interfaces para aplicaciones OpenGL, mediante la implementación de ventanas en las cuales pueden visualizarse las escenas 2D o 3D renderizadas con OpenGL. GLUT también permite la creación de ventanas simples capaces de contener controles de interfaz de usuario que pueden usarse como interfaz de la aplicación.

OpenGL permite generar una escena en un espacio virtual con el objetivo de visualizarla desde distintos puntos de vista en el espacio. Como nuestro simulador debe reproducir lo que nuestro robot observa por medio de su cámara, es necesario definir los elementos que componen la escena o ambiente donde se desenvuelve el robot y la relación entre ellos El tipo

de ambiente interior que definimos para probar nuestra estrategia de navegación es un pasillo, con los siguientes elementos básicos:

- Paredes laterales
- Piso
- Techo
- Puertas

Las operaciones primitivas que implementa OpenGL permiten genear el modelo y visualizarlo desde una posición determinada en el espacio virtual. Los parámetros básicos que definen el ambiente del robot son los siguientes:

- Largo del Pasillo
- Altura del Techo
- Ancho de las Puertas
- Altura de las Puertas
- Cantidad de Puertas de la pared izquierda.
- Posición de cada puerta de la pared izquierda
- Cantidad de Puertas de la pared derecha.
- Posición de cada puerta de la pared derecha.

Luego de establecido el modelo del ambiente, OpenGL permite definir la posición en el espacio virtual desde la cual queremos observar el modelo. Para entender esto, podemos hacer una analogía con la presencia de un observador en una posición determinada del espacio, observando la escena, apuntando su mirada en una dirección específica. Llamamos Punto de Vista a la posición de observación, junto con la dirección de observación y otros parámetros que detallaremos más adelante. OpenGL renderiza una imagen que corresponde a lo que el observador

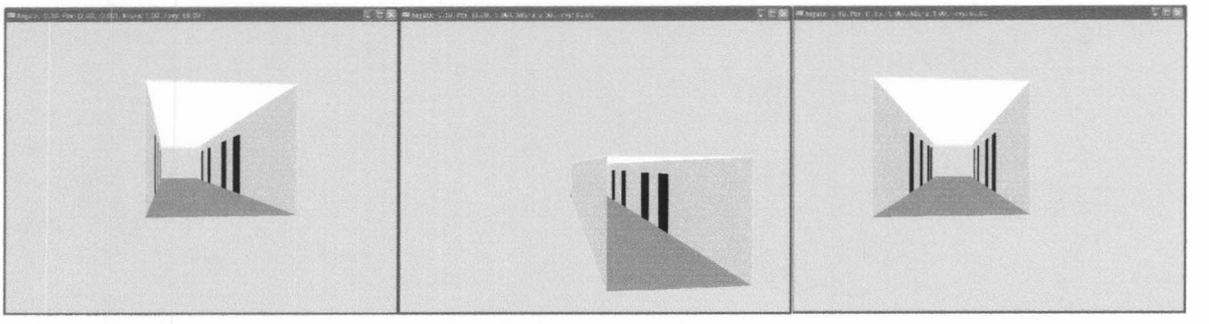

Figura 4-1: Distintos Puntos de Vista del Modelo del Pasillo. El color celeste de las imágenes representa el vacío en OpenGL.

puede ver desde su posición. En la figura 4-1 podemos ver un modelo de un pasillo de 15 metros de largo, 2.7 metros de alto y 2.3 metros de ancho, observado desde distintos Puntos de Vista.

Como mencionamos anteriormente, es posible determinar un "observador" en el espacio virtual definido en OpenGL, En nuestro caso, identificamos a ese observador como el robot, brindándonos la posibilidad de incluir un Robot Virtual dentro del espacio generado. Si posicionamos al robot dentro del pasillo, podemos renderizar las imágenes que el mismo ve desde una posición determinada. En el alcance de este trabajo definimos que el Robot está provisto con una única cámara fija, sin posibilidad de movimientos de Pan o Tilt, apuntando en dirección paralela al piso del ambiente. Por lo tanto, desde el punto de vista del posicionamiento de la cámara dentro del espacio virtual nos interesa su posición dentro del pasillo, a qué distancia vertical del suelo se encuentra y en que dirección apunta. En la figura 4-2 podemos observar lo que vería un robot posicionado al comienzo del pasillo de la figura 4-1, apuntando hacia el final del pasillo.

Para poder probar la estrategia de navegación en pasillos con distintas características, hemos implementado en el sistema la capacidad de modelar una secuencia de pasillos, de forma tal que cuando se termina uno, el camino sigue por otro nuevo que se empalma perpendicularmente con el anterior. Este esquema intenta representar las redes de pasillos de edificios reales. En la figura 4-3 se observa el punto de vista del robot en un corredor que se empalma con un segundo pasillo al final. Como vemos, ya no se observa una sección del espacio vacío que se ve en la figura 4-2, sino la pared del próximo camino a tomar.

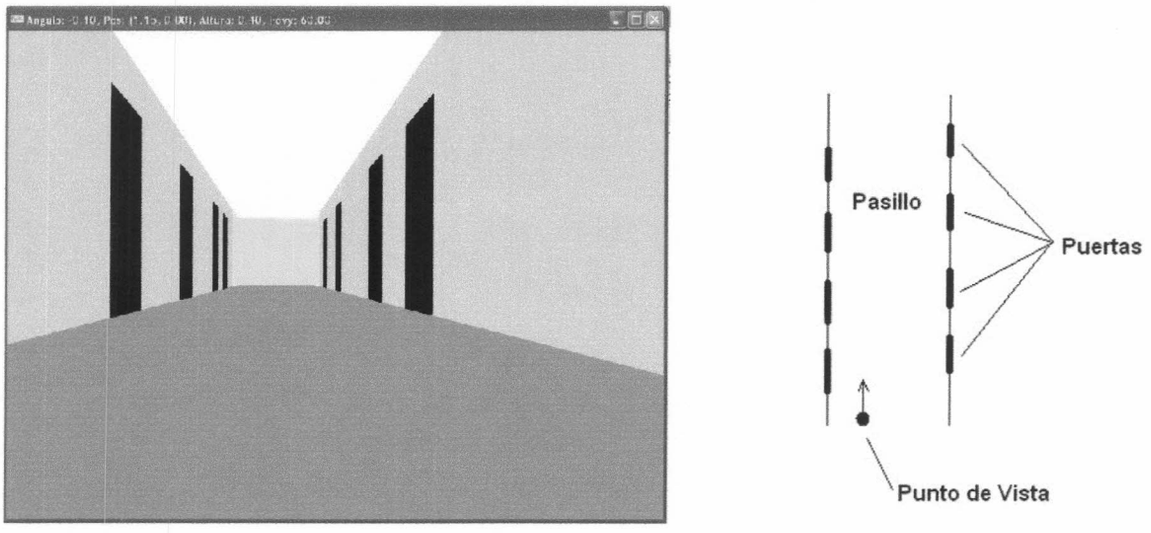

Figura 4-2: Punto de Vista del Robot. Cámara a 45 cm del suelo.

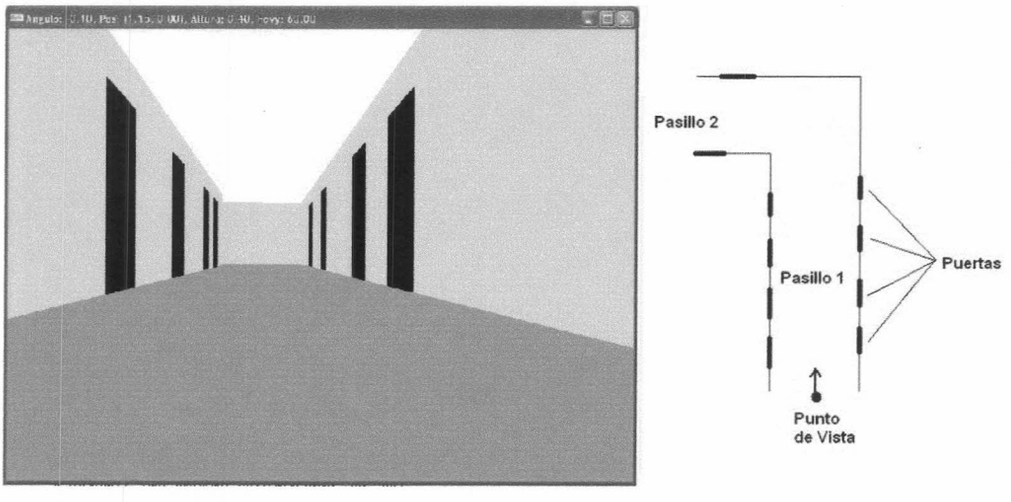

Figura 4-3: Sobre el final del pasillo, se observa la pared lateral del próximo pasillo.

En la figura 4-4, se observa el punto de vista del robot posicionado al final del primer pasillo. Podemos ver como el robot se encuentra dentro del ambiente virtual generado por OpenGL. Este ambiente es donde el robot virtual desarrollará sus desplazamientos de acuerdo a los comandos de movimientos definidos por la estrategia de navegación.

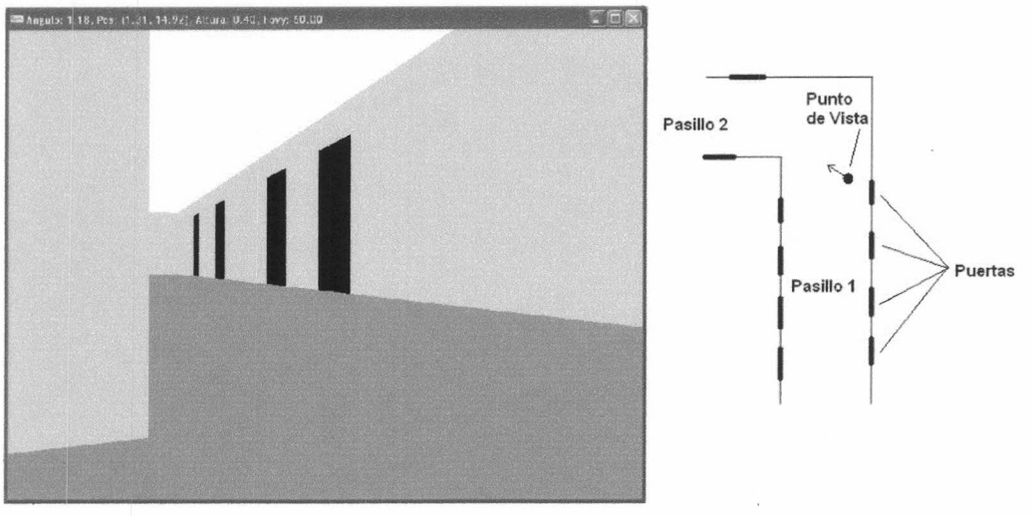

Figura 4-4: Punto de Vista del Robot sobre el final del primer pasillo.

En las figuras anteriores de esta sección hemos observado distintas imágenes renderizadas mediante OpenGL, correspondientes a vistas del ambiente virtual donde se desarrollará la navegación del robot. Estas imágenes se identifican con las imágenes que capturaría un robot físico si se encontrara posicionado dentro de un pasillo real con las mismas dimensiones y configuración que el modelo representado en OpenGL. Pero para simular las imágenes que tomaría el robot, debemos considerar las características de una cámara real e incorporarlas en el renderizado de OpenGL.

Las imagenes renderizadas mediante OpenGL corresponden a proyecciones del espacio virtual en un plano de proyección que debe establecerce junto con la definición del observador. Las imágenes capturadas por una cámara digital son modeladas utilizando la renderización correspondiente a una proyección proyectiva de OpenGL, la cual se ilustra en la figura 4-5.

Otras características que debemos modelar en la renderización de las imágenes del ambiente

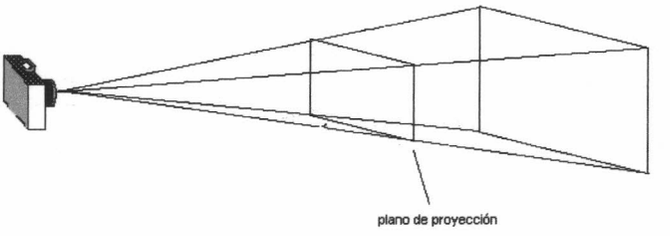

Figura 4-5: Proyección Proyectiva

virtual son el *aspect ratio,* o *resolución* de la cámara y el *ángulo de apertura* de la misma. La apertura de la cámara es particulamente importante ya que el método de posicionamiento depende de la geometría del pasillo y la distribución de las puertas, lo cual se observa de diferente forma en la imagen capturada o renderizada, variando la apertura En la figura 4-6 observamos tres imágenes renderizadas en una resolución de 640x480 pixels, con distintos ángulos de apertura. Se observa claramente cómo se modifica la distribución de las puertas dentro de la imagen, lo cual impacta directamente en el método de extracción de los puntos característicos necesarios para obtener la posición del robot. Es importante mencionar que no hemos modelado posibles efectos de distorsión óptica que pueden producirse en la imagen con algunos ángulos de apertura, ya que enfocamos nuestro trabajo en la definición de la estrategia de navegación y no en la corrección de las imágenes capturadas por el robot. Para la construcción de un robot físico que utilice nuestra estrategia será necesario considerar y estudiar este aspecto.

Como veremos en el capítulo de Resultados, hemos experimentado los resultados del posicionamiento con distíntos ángulos de apertura, para determinar la más adecuada para la cámara del robot. Esta información puede utilizarse en el momento de elegir la cámara física a instalar en un robot gobernado por nuestra estrategia de navegación.

Hasta ahora hemos expuesto el modelado de las características geométricas del ambiente tipo pasillo donde el robot realizará sus desplazamientos. OpenGL permite incorporar texturas en los objetos que componen en ambiente, con el objetivo de agregar realismo a la escena. Si

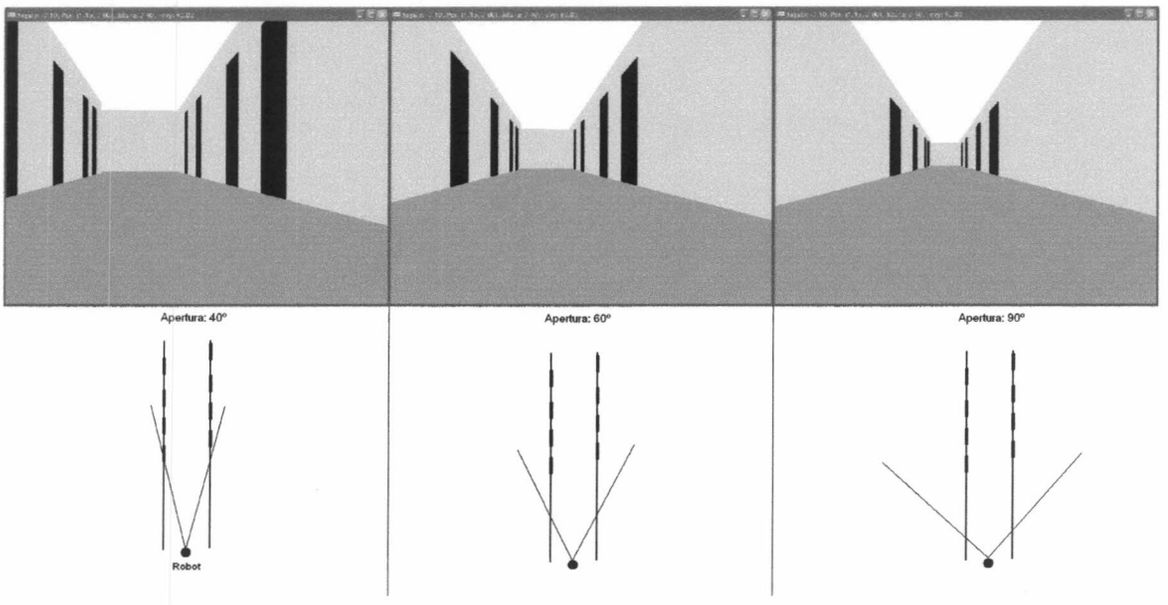

Figura 4-6: Renderizado con distintos ángulos de apertura.

bien no hemos implementado texturas del tipo *fotorealistas*, que permiten un nivel extremo de realismo, hemos incorporado texturas que presentan patrones característicos de los componentes del pasillo para evaluar cómo se comporta el método de extracción de puntos característicos ante algunos problemas del mundo real. OpenGL permite tomar muestras de estas texturas para aplicarlas completamente al objeto definido. En la figura 4-7 se observar ejemplos de las muestras de texturas que utilizamos para decorar el pasillo.

En la figura 4-8 podemos observar el pasillo renderizado con las texturas de la figura 4-7.

Para finalizar el modelado, hemos incorporado algunos elementos que pueden aparecer en pasillos reales, como sillas y armarios, que si bien no representan obstáculos para el desplazamiento del robot, pueden ocluir parcialmente las puertas, dificultando la extracción de los puntos característicos. La figura 4-9 muestra la renderización del pasillo con estos objetos incorporados.

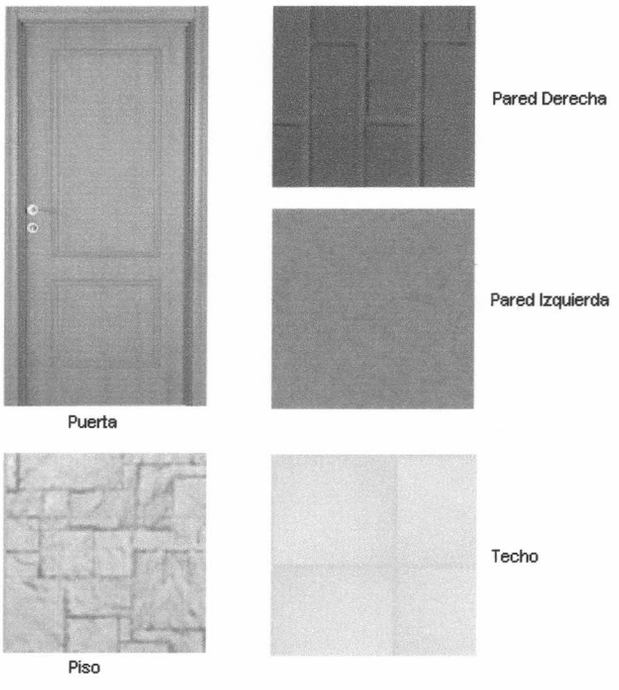

Figura 4-7: Texturas aplicadas a los componentes del pasillo.

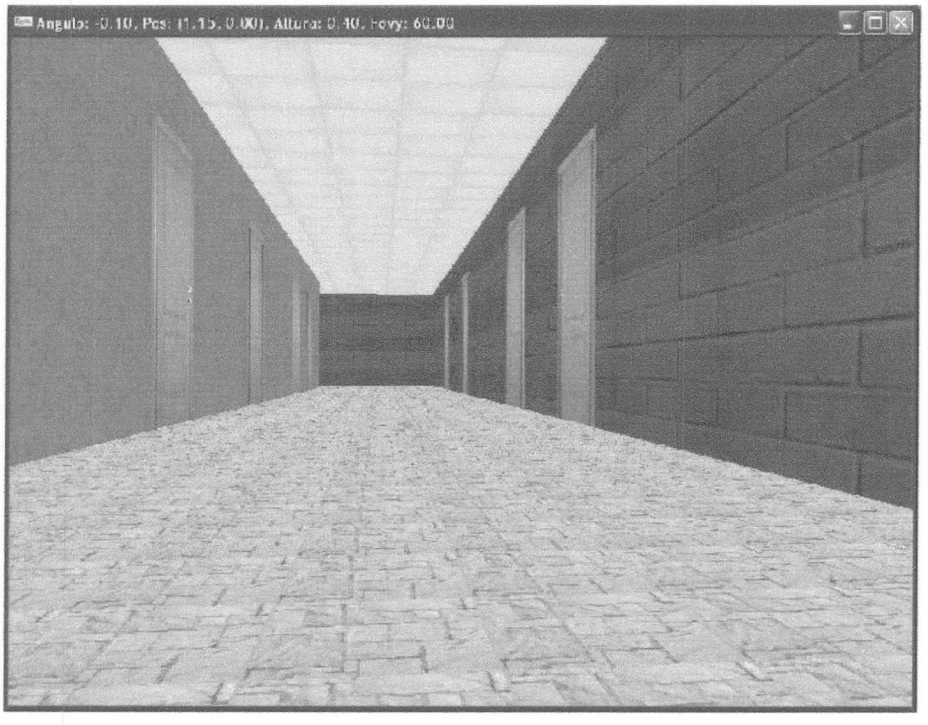

Figura 4-8: Renderización del Pasillo con aplicación de texturas de la Figura 4-8

Figura 4-9: Pasillo con elementos que ocluyen parciamente las puertas.

4.2 Desplazamiento del Robot

Para crear el ambiente de navegación del robot fue necesario crear un conjunto de objetos geométricos y ubicarlos en el espacio 3D del ambiente virtual para formar el modelo del ambiente físico, el cual puede verse desde diferentes posiciones y orientaciones creándose así diversos puntos de vista del mismo modelo. Utilizamos OpenGL como herramienta para implementar el modelo virtual del ambiente de navegación.

En la sección anterior mencionamos que el ambiente de simulación de nuestro proyecto se observa exclusivamente desde el punto de vista del robot de acuerdo a su posicionamiento y orientación, los cuales se corresponden con la cámara que el mismo posee. La cámara está instalada en el centro del robot y mirando hacia el frente.

El proceso necesario para modelar un punto de vista de la escena en un ambiente virtual se compone de una serie de transformaciones geométricas las cuales son análogas a los pasos necesarios para la toma de una fotografía con un cámara. Primero elige el tipo de lente de la cámara, luego se ubica la cámara en una posición específica y se apunta la misma hacia una dirección determinada, de forma tal de capturar los objetos que componen la escena a fotografiar. En OpenGL se identifica a la cámara virtual con un Punto de Vista, desde donde se observará la escena. OpenGL ofrece herramientas para crear el modelo o escena (operación Transformada del Modelo), para decidir el tipo de lente de la "cámara virtual" (Transformada de la Proyección), para ubicar y orientar el punto de vista en el espacio virtual (Transformada de la Vista), y para seleccionar la resolución de las imágenes a generar (Transformada de Mapeo hacia el visualizador de la imagen).

Para que el simulador imite el movimiento real del robot dentro del ambiente, los desplazamientos se realizan a través de traslaciones y giros pequeños de acuerdo a la velocidad de movimiento definida en la interfaz del simulador. Se utiliza la función *gluLookAt* de OpenGL que permite observar una escena desde diferentes puntos de vista, en nuestro caso nos permite reproducir el resultado visual luego de cada movimiento, para construir la secuencia completa de navegación del robot dentro de su entorno de pasillos.

Uno de los problemas que enfrentamos fue como poder visualizar las variaciones que se producen en los desplazamientos del robot producidas por los factores internos y externos, como vimos en el capítulo anterior dos fuentes de ruido afectan al sistema y debiamos representarlo.

En caso de hacer grandes desplazamientos sólo era posible incorporar la dispersión producida los factores internos y externos al final del movimiento lo cual no reflejaba la sensación de movimiento contínuo en el simulador. Es por eso que se generan movimientos pequeños para poder observar como el sistema de ruido afecta el normal desplazamiento del robot a lo largo de su recorrido.

A continuación mostramos un recorrido realizado por nuestro simulador, donde la secuencia verde corresponde al recorrido real que realiza el robot por el pasillo que navega, mientras que la secuencia roja corresponde a la creencia del robot sobre el recorrido que realiza, en azul se muestran los ajusten realizados al detectarse un incerteza excesiva y ejecutarse el mecanismo de auto-localización.

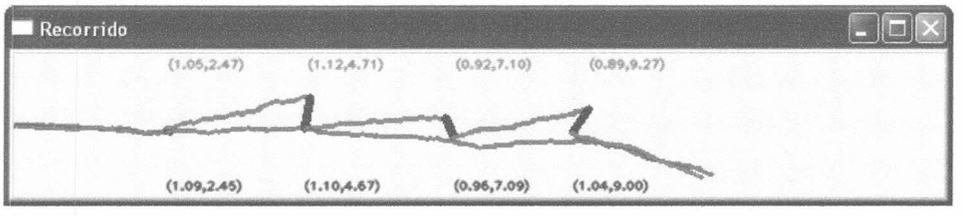

Figura 4-10: Recorrido de un pasillo.

En nuestro simulador, el modelo del error de los movimientos de robot es utilizado para generar un recorrido virtual del robot. Las posiciones de ese recorrido virtual, y las orientaciones que el robot va tomando a lo largo del camino, forman los puntos de vista desde los cuales el simulador va generando las imágenes, para mostrar la vista del entorno en el cual el robot se está desplazando. A su vez, el mismo modelo es utilizado por la estrategia de navegación para que el robot estime su posición y la incerteza de la misma.

Por otro lado es necesario mostrar la posición real en la que se encuentra el robot, es decir que la posición y punto de vista de la cámara se actualiza de acuerdo a esta localización, mostrando así el camino real navegado por el robot.

En esta imagen también podemos visualizar que los recorridos no son perfectos, esto se debe a la incorporación de ruido en el sistema de movilidad. Este ruido se aplica tanto al mecanismo de control que produce los desplazamientos del robot como a la interacción con el ambiente donde navega 3.1.

En nuestro simulador los valores de ruido sobre el controlador de movimientos se especifican a través de un porcentaje, el cual indica que la varianza de la distribución es el porcentaje aplicado al desplazamiento a realizar, de esta manera los valores aleatorios de ruido generados estarán relacionados acorde al movimiento solicitado y al nivel de ruido que se desea generar.

Los factores de ruido se aplican tanto a las traslaciones como a las rotaciones. Estos factores se obtuvieron a través de los resultados experimentales en valores de media y varianza, y se aplican a cada una de las componentes independientes que forman la información de localización. Utilizando el mecanismo de estandarización de variables aleatorias se transforman las distribuciones de cada componente para que sean de la forma $N(0, \sigma)$.

Capítulo 5

Construcción del Sistema de Simulación de Navegación de Robots

5.1 Arquitectura

El sistema que implementa la simulación de la navegación del robot en su ambiente de pasillo se diseñó siguiendo una arquitectura modular, con el objetivo de encapsular las funciones específicas que se deben realizar. La figura 5-1 expone la arquitectura general definida.

5.1.1 Organizador de Tareas

El organizador de tareas es el nivel superior de la arquitectura y se encarga de llevar adelante el control completo del sistema, conoce la existencia del resto de los módulos y coordina la interacción entre ellos.

Cuando se inicia la ejecucción del Organizador de Tareas se solicita al módulo Simulador OpenGL (5.1.5) la construcción del ambiente por el cual navegará el robot. El Organizador de Tareas también informa la ubicación inicial del robot para iniciar la visualización del ambiente.

El Organizador de Tareas conoce las especificaciones de los pasillos a través de los cuales el robot debe desplazarse. Los objetivos parciales del robot consisten en alcanzar el final de cada uno de los pasillos que debe recorrer, es por eso que interactúa con el Planificador de Movimientos (5.1.2) para conocer el próximo desplazamiento a realizar de acuerdo a su objetivo

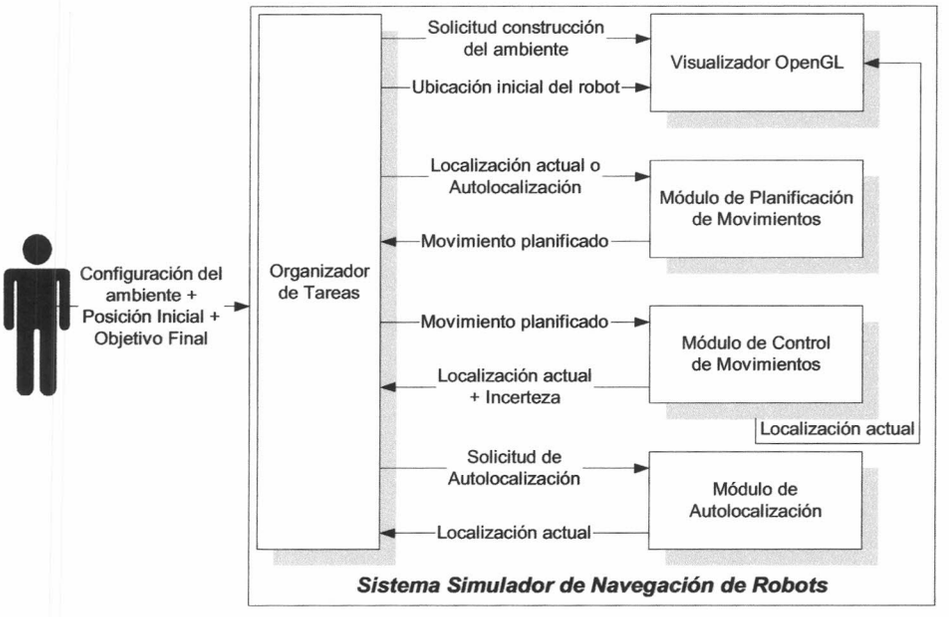

Figura 5-1: Arquitectura Simulador de Navegación de Robots

parcial a alcanzar.

El Organizador de Tareas recibe como respuesta del Planificador de Movimientos un desplazamiento a ejecutar, éste puede ser una traslación o un giro que son los movimientos implementados en nuestro modelo, es entonces que solicita al Controlador de Movimientos (5.1.3) la ejecución de dicha operación. Una vez concluído el desplazamiento, el Organizador de Tareas recibe como respuesta la localización actual y la información sobre la incertidumbre acumulada sobre la situación del robot.

La secuencia de interacciones del Organizador de Tareas con el Planificador y el Controlador de Movimientos se repite en forma contínua, logrando de esta manera el desplazamiento del robot en el ambiente, hasta que la incertidumbre alcanzada en la ubicación u orientación sobrepasa un umbral preestablecido. En ese momento, el Organizador de Tareas utiliza el Módulo de Autolocalización (5.1.4), que mediante la ejecución del método de *Invariantes Proyectivos* sobre la imagen capturada por la cámara del robot, permite ajustar con mayor exactitud su localización en el ambiente.

Cuando el Módulo de Autolocalización informa la ubicación y orientación del robot en el ambiente, el Organizador de Tareas comienza nuevamente su ciclo de desplazamientos, solicitando al Planificador de Movimientos que ajuste la orientación del robot de acuerdo a su objetivo parcial. La respuesta será un movimento de giro en la dirección apropiada que será ejecutada por el Controlador de Movimientos, una vez alcanzada la orientación deseada el Planificador definirá los movimientos de traslación necesarios para alcanzar el final del pasillo.

Una vez que el robot alcanza el final del pasillo que actualmente recorre, el Organizador de Tareas realiza la rotación del sistema de coordenadas del ambiente de acuerdo a la dirección del nuevo pasillo que se debe recorrer. Esta última acción permite reutilizar la funcionalidad de recorrido de un pasillo para cualquier pasillo orientado en cualquier dirección.

El Organizador de Tareas evalúa luego de cada desplazamiento o autolocalización el alcance de los objetivos intermedios o su objetivo final, para ello calcula la diferencia entre la posición objetivo y su posición actual, y determina si su objetivo ha sido alcanzado.

La figura 5-2 muestra el esquema detallado en esta sección

5.1.2 Planificador de Movimientos

El módulo Planificador de Movimientos es invocado por el Organizador de Tareas para obtener el próximo desplazamiento a realizar y así alcanzar el objetivo parcial. Los movimientos que el robot puede realizar son traslaciones y rotaciones y dependen de la situación informada por el Organizador de Tareas.

El robot se mueve en forma permanente hasta que se realice una autolocalización, en ese caso las decisiones a tomar varían de acuerdo al resultado de la misma. La autolocalización tiene asociada dos modos: el primero que realiza la búsqueda de los puntos característicos y el segundo que realiza la búsqueda de la marca de fin de pasillo.

Al iniciar el recorrido de un pasillo la búsqueda se centra sobre los puntos característicos cuando se hace necesario obtener una autolocalización, pero cuando falla se cambia el modo a búsqueda de la marca de fin de pasillo.

En el primer caso, cuando se utiliza la autolocalización, se debe tener en cuenta el resultado de la misma para determinar el próximo paso a seguir. Por eso es necesario conocer si la posición y la orientación del robot pudieron ser obtenidas correctamente. Para cada combinación posible

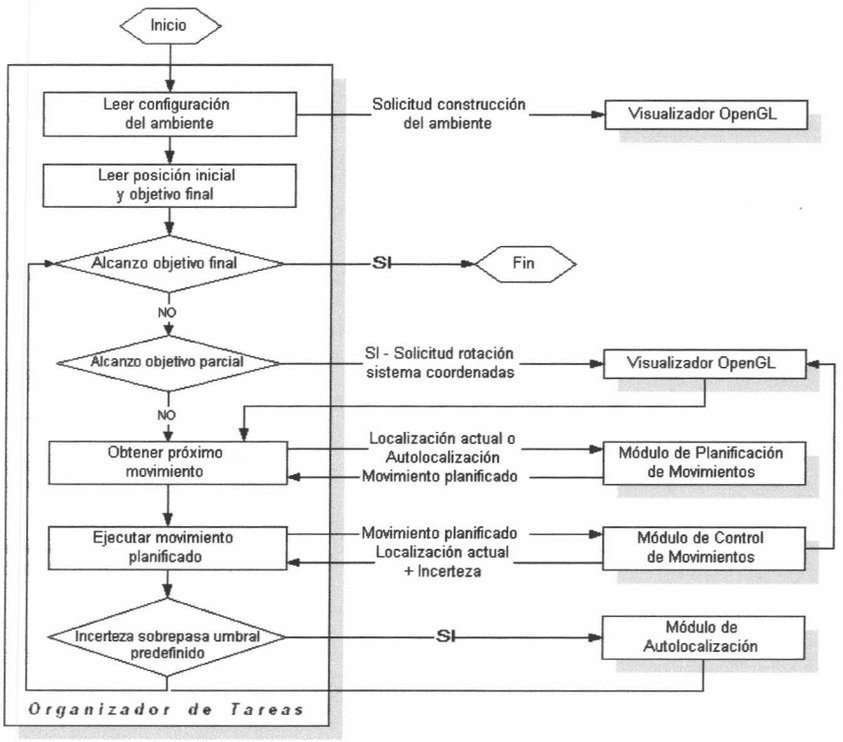

Figura 5-2: Módulo Organizador de Tareas

de dicho resultado se explica cual es el próximo paso a seguir de acuerdo al Planificador de Movimientos:

- Posición y orientación obtenidas: El robot conoce su ubicación en el entorno y sólo debe estimar la rotación necesaria para direccionarse hacia su objetivo, por lo que próximo movimiento a realizar es una rotación con el ángulo de giro calculado.
- Posición obtenida: El próximo movimiento a realizar es una traslación, esperando que los próximos pasos guíen al robot hacia el objetivo.
- Orientación obtenida: El próximo movimiento a realizar es un giro en dirección hacia su objetivo.
- Posición y orientación no obtenidas: Se modifica el modo de búsqueda para obtener la marca de fin de pasillo

  La figura 5-3 muestra un esquema para este módo de búsqueda:

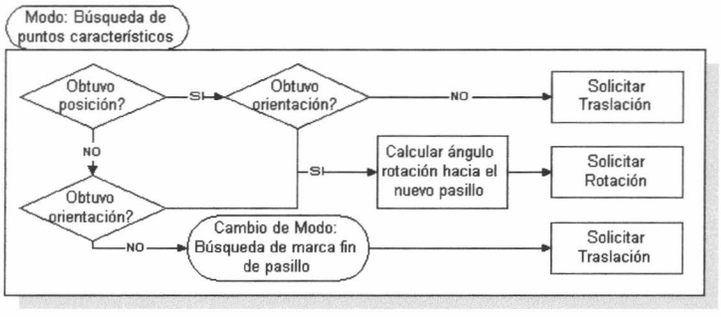

Figura 5-3: Diagrama de decisión - Búsqueda de Puntos Característicos

Cuando el modo haya cambiado hacia la búsqueda de la marca de fin de pasillo solo habrá dos situaciones que contemplar:

- El robot esta orientado: Se realizará una traslación para acercarnos al objetivo.
- El robot no esta orientado: Se realizará un giro con el ángulo necesario para orientar al robot hacia la marca de fin de pasillo.

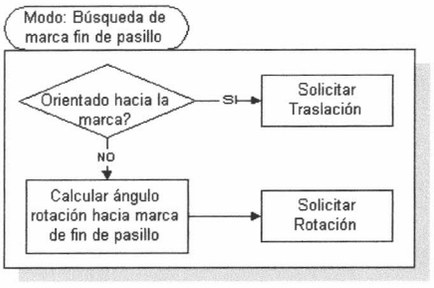

Figura 5-4: Diagrama de decisión - Búsqueda Marca Fin de Pasillo

La figura 5-4 muestra un esquema para este módo de búsqueda:

Si no se ha realizado una autolocalización, el Planificador de Movimientos controla si se ha alcanzado el objetivo. Si ése fuera el caso, el robot debería realizar un giro en la dirección en que se encuentra el próximo pasillo, por lo que el movimiento planificado será una rotación de 90^o en el sentido en que se acopla el próximo.

La figura 5-5 muestra un esquema del Planificador de Movimientos

5.1.3 Controlador de Movimientos

El módulo Controlador de Movimientos se encarga de actualizar la posición del robot en el simulador con el movimiento que se le solicita realizar desde el Organizador de Tareas.

Este módulo contiene un componente que implementa el sistema de movilidad, modelando los posibles desplazamientos del robot. El sistema de movilidad, a su vez, está contenido en un filtro extendido de Kalman que se utiliza para medir la incertidumbre en el sistema luego de cada movimiento. En nuestro trabajo el Controlador de Movimientos utiliza dos instancias de este componente: una que modela los desplazamientos dentro del ambiente, mientras que la otra para modelar la creencia real del robot.

La primera instancia modela las posiciones reales en la que se halla el robot luego de cada desplazamiento, por lo tanto esta información es utilizada por el sistema para posicionar la visualización del entorno.

La segunda instancia no solo modela la posición y orientación en la cual el robot supone

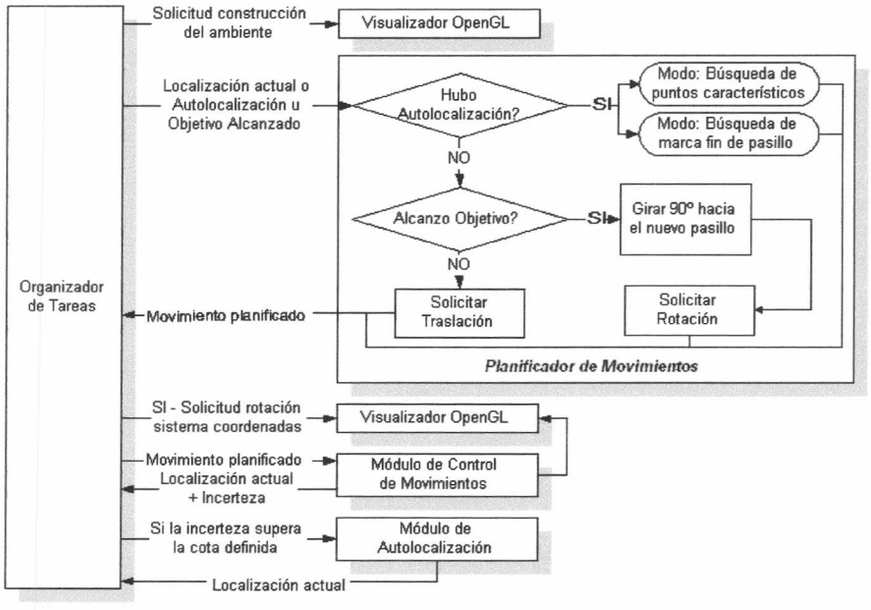

Figura 5-5: Módulo Planificador de Movimientos

encontrarse, sino que también calcula la incertidumbre de su creencia. Esta información es la que devuelve al módulo Organizador de Tareas para decidir si es necesario o no utilizar el mecanismo de autolocalización. Cuando el módulo Planificador de Movimentos decide el próximo movimiento a realizar lo hace en base a la localización y orientación generada por esta instancia.

La figura 5-6 muestra el esquema del Controlador de Movimientos

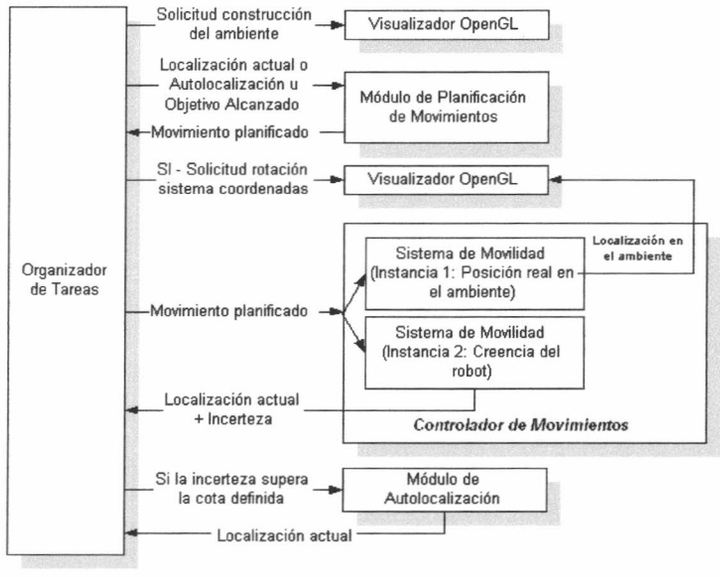

Figura 5-6: Módulo Controlador de Movimientos

5.1.4 Módulo de Autolocalización

De acuerdo a lo expuesto en la sección 5.1.2, cuando la incerteza de la posición supera un umbral, el Sistema de Navegación ejecuta el procedimiento de autolocalización basado en *Invariantes Proyectivos*. El Módulo de Localización recibe la imagen renderizada por el simulador, correspondiente a lo que observa la cámara del robot virtual en el momento que la incerteza supera el umbral. Esta imagen es utilizada como entrada al algoritmo expuesto en la sección 3.2.3, para la localización de los puntos característicos de la imagen.

Como fue expuesto en el capítulo 3.2, la posición del robot es estimada contrastando los

cocientes cruzados de los puntos característicos de la imagen, contra los cocientes cruzados de las posiciones de las puertas en el ambiente donde se mueve el robot. Para calcular los cocientes cruzados del ambiente es preciso que el robot cuente con la información sobre las dimensiones y posiciones de los componentes del pasillo. La distribución de las puertas en el ambiente virtual, esto es su posición dentro del pasillo, se define mediante un archivo de configuración, que podemos observar en la figura 5-10. Este archivo también es usado por el simulador para modelar y renderizar el ambiente virtual, como expondremos en la sección 5.1.5.

Para el cálculo de la posición, el sistema toma el cociente cruzado más cercano de la pared izquierda y de la pared derecha del pasillo, esto es, se calcula el cociente cruzado correspondiente a los puntos característicos más cercanos al robot. En la imagen renderizada, los puntos caracterísiticos más cercanos corresponden a los que se encuentran más abajo en la imagen. En la figura 5-7 observamos un ejemplo de los puntos característicos seleccionados para el cálculo de los cocientes cruzados observados por el robot.

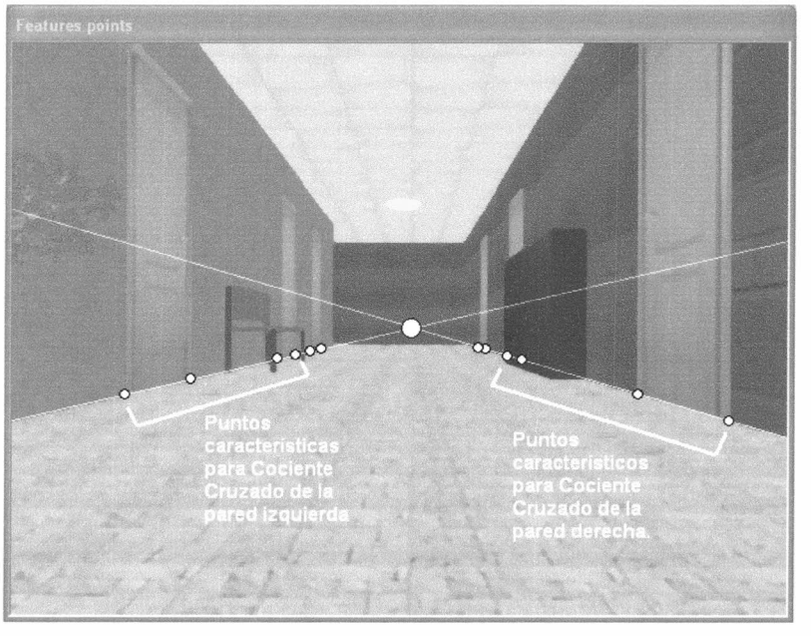

Figura 5-7: Puntos característicos extraidos de la imagen para el cálculo de los Cocientes Cruzados

Dado que los puntos característicos seleccionados de la imagen de la cámara virtual del

robot son los más cercanos a la cámara y dado que el Planificador de Movimientos sólo decide movimientos hacia adelante, según lo expuesto en la sección 5.1.2, la tabla de cocientes cruzados del ambiente donde se buscan los cocientes cruzados observados en la imagen es armada tomando subconjuntos consecutivos de cuatro puntos característicos del ambiente. Ilustramos esto en la figura 5-8.

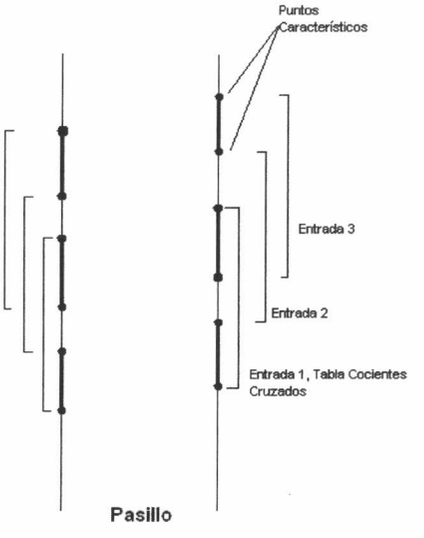

Figura 5-8: La tabla de cocientes cruzados del pasillo se arma tomando subconjuntos consecutivos de cuatro puntos característicos. Se observan en la imágen los puntos característicos utilizados para calcular el cociente cruzado almacenado en las entradas 1, 2 y 3 de la tabla. La tabla se completa relevando todas las puertas del pasillo.

Al identificar los cocientes cruzados calculados a partir de la imagen observada, con los cocientes cruzados correspondientes de la tabla, logramos identificar cuáles son las puertas que el robot está observando con su cámara, cuya posición es conocida por el robot, ya que se encuentra indicada en el archivo de configuración.

El último paso para la autolocalización, según lo expuesto en la sección 3.2 (Pag. 29), consiste en la resolución del sistema de ecuaciones 3.4 (Pag. 32) y 3.5 (Pag. 33), para lo cuál debemos seleccionar cuatro puntos característicos de la imagen. Por los efectos de la

perspectiva, una diferencia de pocos pixels representa una distancia mayor en el ambiente cuanto más alejados al robot estén los puntos indicados por esos pixels. Unos pocos pixels cercanos al Punto de Fuga pueden representar varios centímetros en el pasillo, mientras que la misma cantidad de pixels cerca de la posición del robot pueden representar solo unos milímetros. Esta diferencia tiene un efecto importante en la precisión de la auto-localización. Por esto, seleccionamos los cuatro puntos característicos más cercanos al robot para calcular la posición del robot, como se ejemplifica en la figura 5-9.

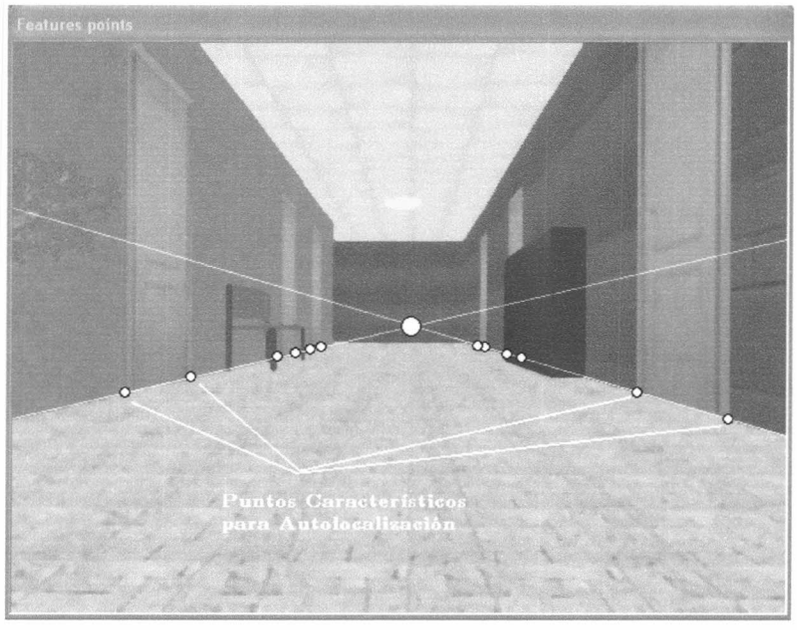

Figura 5-9: Puntos característicos mas cercanos al robot.

5.1.5 Simulador OpenGL

El SimuladorOpenGL, brinda dos servicios: provee al Módulo de Localización la imagen renderizada correspondiente a la captura de la cámara del robot virtual y ofrece al operador humano una ventana donde puede observarse la simulación del movimiento del robot en el ambiente modelado, según lo expuesto en el Capítulo 4.

El Simulador obtiene la información necesaria para generar el modelo de los pasillos donde

se trasladará el robot, desde un archivo de configuración donde se especifican los siguientes parámetros:

- Cantidad de Pasillos
- Largo, Alto y Ancho de cada pasillo.
- Cantidad de Puertas de cada pasillo, para las dos paredes laterales.
- Dimensiones de las puertas.
- Dirección de giro al final de cada pasillo.
- Texturas a aplicar a los elementos del pasillo.

En la figura 5-10 se observa una pequeña sección del archivo de configuración.

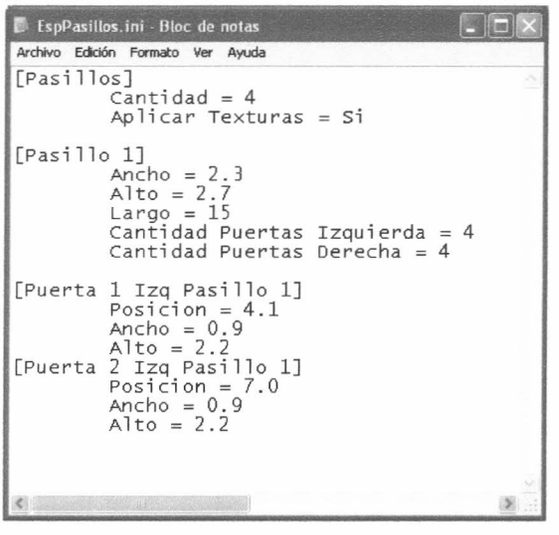

Figura 5-10: Configuración del Modelo de Pasillos Virtuales

La interfaz del simulador presenta tres ventanas, las cuales describimos a continuación.

Ventana de Simulación

Esta ventana presenta al usuario una vista del ambiente simulado mediante OpenGL. En esta ventana pueden observarse los movimientos del robot, observados desde la cámara virtual, así también como el resultado del método de autolocalización cuando el mismo es invocado para corregir la posición estimada del robot. En la figura 5-11 puede observarse la ventana de Simulación en el momento en el cual el Sistema de Navegación muestra la posición calculada a partir de la imagen de la cámara virtual. También es posible observar los pasos intermedios del procedimiento de localización. En la figura 5-12 puede observarse, por ejemplo, el resultado de la segmentación de la imagen y la detección de las líneas verticales que definen los límites de las puertas del pasillo.

Panel de Control

El Panel de Control presenta una serie de controles que permiten modificar distintos parámetros que gobiernan el comportamiento de los componentes de Sistema de Navegación. La modificación de los parámetros resulta en el cambio inmediato del comportamiento del Simulador, reflejando las nuevas condiciones para los movimientos posteriores del Robot virtual. En la figura 5-11 se observan los parámetros, cuya descripción es la siguiente:

Cámara Virtual

- Resolución :permite modificar la resolución de la cámara virtual
- Altura: especifica la distancia de la cámara al suelo
- Apertura: ángulo de apertura de la cámara virtual.

Control de Incerteza

- Umbral X: especificación del umbral de Incerteza en la dirección perpendicular al pasillo, cuando la incerteza supera este umbral es ejecutado el procedimiento de autolocalización.
- Umbral Y: especificación del umbral de Incerteza en la dirección longitudinal al pasillo, cuando la incerteza supera este umbral es ejecutado el procedimiento de autolocalización.

Simulación Permite especificar los paràmetros necesarios para simular el movimiento del robot en el ambiente virtual. El robot tiene dos capacidades de movimiento: giro y avance. Estas capacidades son independientes, esto es, en un instante determinado el robot puede avanzar o girar, pero no ambas cosas a la vez, por lo cual el robot solo puede girar cuando está detenido, o trasladarse en línea recta.

- Velocidad Lineal: velocidad de traslación del robot en línea recta.
- Velocidad Giro: velocidad de giro del robot.
- Error del Controlador de Movimientos: es una variable aleatoria cuya media es cero pero su varianza es un porcentaje relacionado al desplazamiento realizado. Con esos valores se genera el ruido asociado al sistema de desplazamiento del robot.
- Error de Posicionamiento del sistema: es el valor de la varianza de una variable aleatoria con media cero, que indica el error externo que se produce en el posicionamiento al realizar el desplazamiento.
- Error de Orientación del sistema: es el valor de la varianza de una variable aleatoria con media cero, que indica el error externo que se produce en la orientación al realizar el desplazamiento.

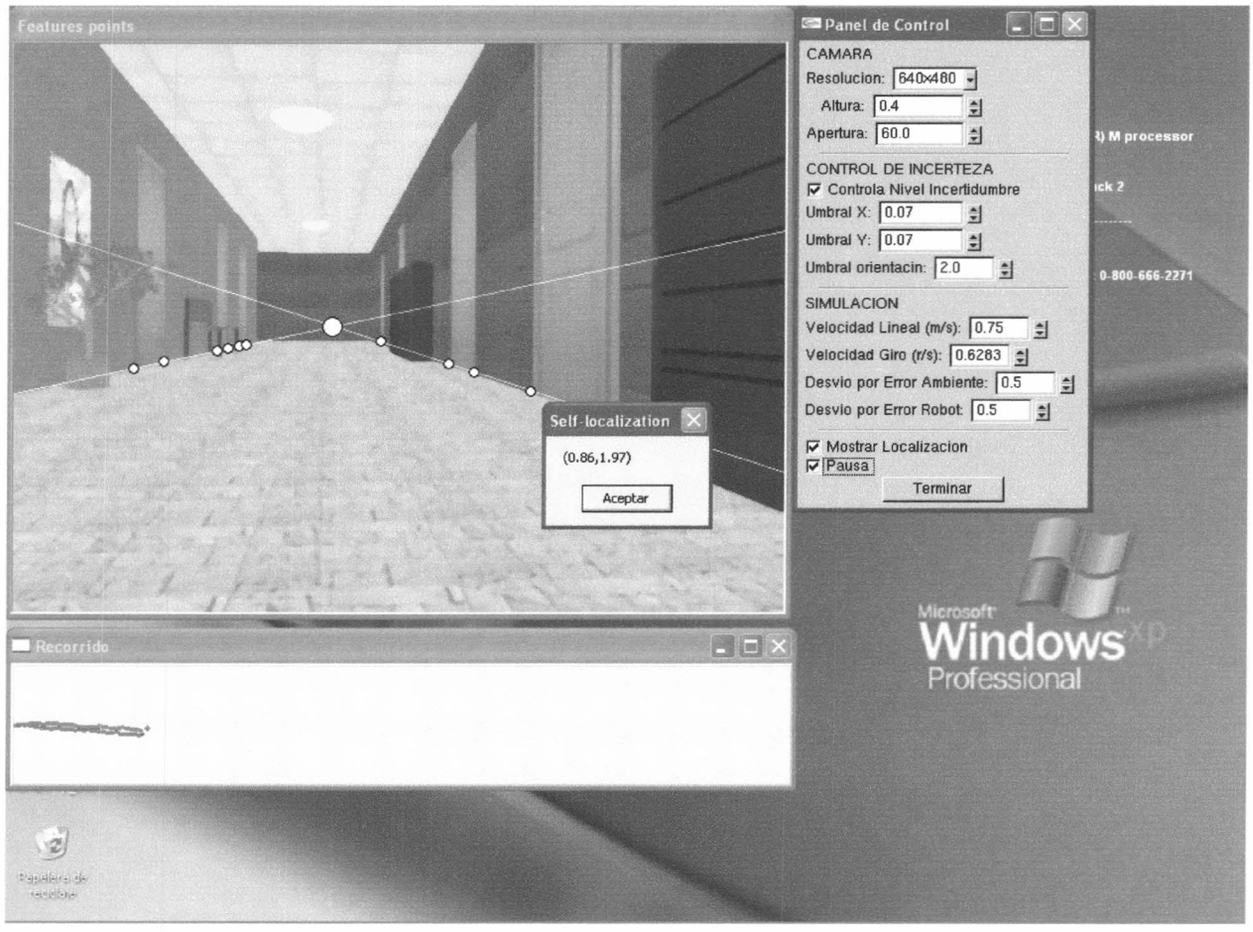

Figura 5-11: Simulador OpenGL: Detección de Puntos Característicos.

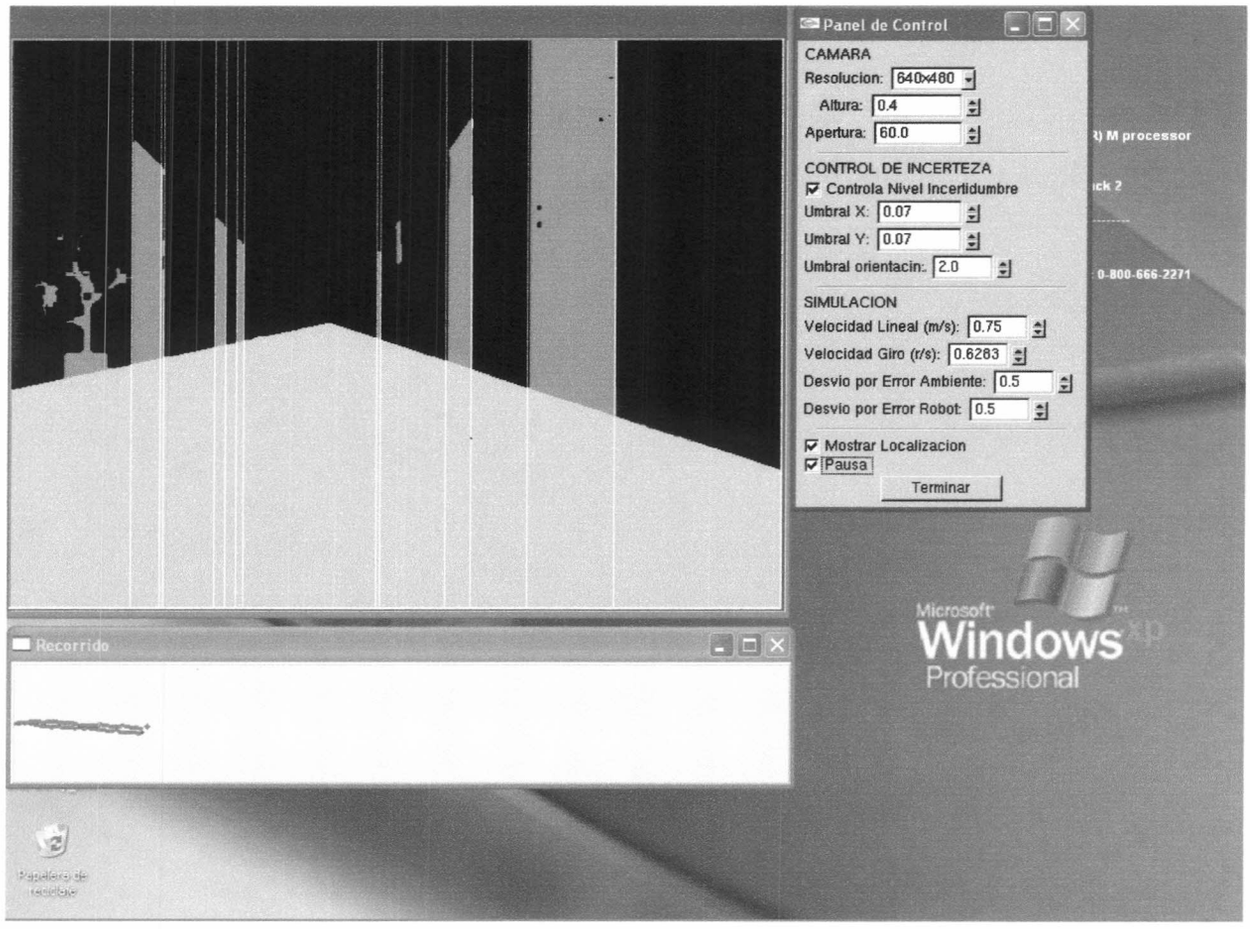

Figura 5-12: Simulador OpenGL: segmentación y detección de lineas verticales.

Capítulo 6

Resultados

Realizamos experimentos para determinar cual es la mejor altura en la cual instalar la cámara y cual sería la mejor apertura. Hemos tomado una muestra al azar de 100 posiciones dentro del pasillo generado por el simulador y, para cada una, generamos la imagen correspondiente a diferentes alturas y aperturas de la cámara. Conociendo la posición desde la cual fue generada, para cada imagen ejecutamos el método de localización para estudiar el error de la localización. El resultado puede verse en la siguiente tabla.

Altura	Apertura	Localizaciones Positivas	Promedio Error Localización	Desviación Error Localización	Promedio Error Orientación	Desvio Error Orientación
0.3	30	15%	0.29	0.34	0.05	0.09
0.3	60	48%	0.43	0.72	0.07	0.15
0.3	90	51%	0.64	0.82	0.06	0.07
0.3	120	33%	0.87	0.77	0.18	0.27
0.6	30	33%	0.34	0.41	0.08	0.19
0.6	60	82%	0.31	0.61	0.12	0.22
0.6	90	88%	0.73	0.91	0.1	0.13
0.6	120	68%	0.95	0.89	0.17	0.27
1.2	30	34%	0.44	0.43	0.06	0.09
1.2	60	87%	0.5	0.81	0.1	0.13
1.2	90	87%	0.65	0.86	0.11	0.16
1.2	120	65%	1.03	0.94	0.24	0.29

Resultados de auto-localizaciones con diferentes parámetros de altura y apertura para la cámara montada en el robot

Si consideramos los casos con tasas de localizaciones positivas superiores al 80%, observamos que el mejor resultado corresponde a una altura de 0.6cm y una apertura de 60 grados, ya que

este caso muestra los menores valores de error promedio de localización y desviación del error. En la tabla se observa este caso resaltado en verde. Si bien se observan en la tabla algunas filas con un menor error promedio de localización, la tasa de localizaciones positivas es muy baja en estos casos. Es importante mencionar que, como las imágenes fueron generadas al azar para este experimento, algunas mostraban posiciones del robot donde no es posible realizar la localización ya que no se encuentran puntos característicos en el campo de visión, o al estar el robot demasiado cerca de una pared, los puntos característicos no se pueden extraer con precisión. Analizando el lote de imágenes de prueba, encontramos que un 16% de las mismas corresponden a este caso. Esto impone un techo de 84% al porcentaje de localizaciones positivas que puede alcanzarse en este experimiento. En esos casos la estrategia de navegación utilizará la marca del fin del pasillo o el ajuste de orientación, para seguir su recorrido, como fue descripto en los capítulos anteriores.

El experimento anterior nos muestra la precisión del método de autolocalización y lo hemos utilizado para estimar los mejores valores de altura y apertura para la cámara, aunque recordemos que el lote de imágenes fue generado al azar. Para evaluar como se comporta el método de auto-localización en condiciones reales de navegación, medimos la precisión del módulo de auto-localización al procesar imágenes generadas en una navegación real. Para esto, luego de determinar la altura y apertura de la cámara, ejecutamos el simulador con estos valores para medir el Error Promedio de Localización durante la ejecución de la estrategia completa. Medimos el resultado de 100 localizaciones durante la ejecución del sistema de navegación en el ambiente simulado, obteniendo un 96% de localizaciones positivas, con un error promedio de localización de 0.11 cm, y un desvío de 0.19 cm. La mejora de los valores con respecto al experimiento anterior se debe a que el sistema de navegación mantiene al robot en posiciones cercanas al centro del pasillo, orientado generalmente hacia el objetivo, con lo cual no se presenta el problema del 16% de imágenes dificultosas para la extracción de los puntos característicos que se observó en el experimiento anterior. El tiempo de ejecución promedio del método de autolocalización resultó ser de 2.5 segundos, en una PC Pentium IV, 2.4 Ghz.

Realizando el experimento de variación de apertura de la cámara, observamos que uno de los problemas del método de autolocalización, consistente en la ausencia de suficientes puntos característicos en la imagen, puede resolverse en algunos casos reintentando la localización con

otras aperturas. En las figuras 6-1 y 6-2 podemos observar dos casos donde la apertura óptima de 60 grados no obtiene suficientes features points, pero modificando la apertura de la cámara a 90^o aparecen más puertas en el campo de visión del robot. Esto nos sugiere investigar sobre la posibilidad de instalar cámaras de apertura variable en el robot físico.

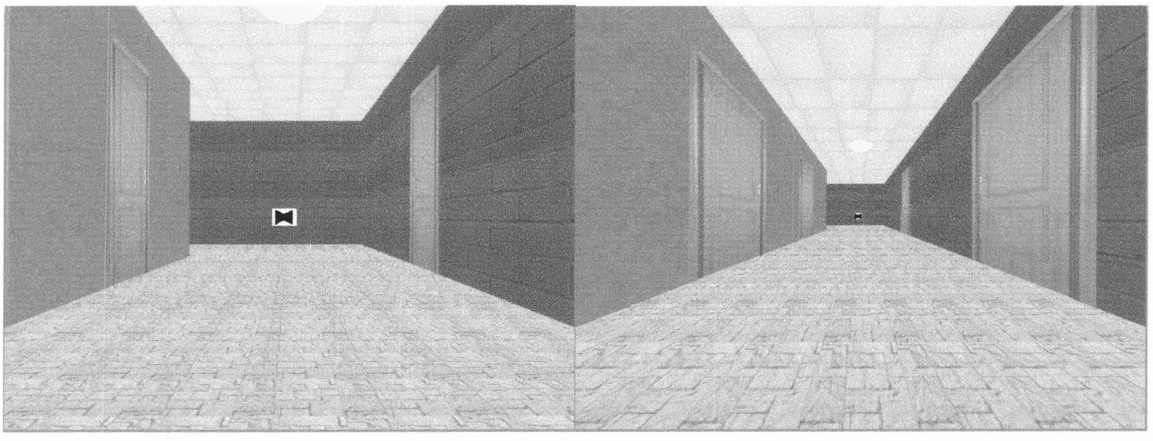

Figura 6-1: Izquierda: Apertura 60^o. Derecha: Apertura 90^o

Otro problema que se observa en el método de autolocalización es el siguiente: si el robot se acerca demasiado a una pared del pasillo, los puntos característicos proyectados en la imagen quedan demasiado cercanos unos a otros y la precisión necesaria para determinar la posición del robot no puede ser alcanzada. La figura 6-3 muestra un ejemplo de una imagen donde los marcos de las puertas más lejanas de la pared izquierda están demasiado cerca en la imagen.

En este caso, si la marca artificial de fin de pasillo es identificada, es posible determinar la orientación del robot analizando las puertas de la pared opuesta, entonces trasladamos al robot en dirección al centro del pasillo para intentar localizarlo nuevamente. Pero si la marca de fin de pasillo no hubiese sido detectada entonces la localización no sería posible, en estos casos se intentan realizar pequeños movimientos para reubicar al robot en el centro del pasillo.

El sistema completo se probó inicialmente sin factores de ruido que afecten al modelo de movimiento propuesto, comprobándose que el robot navega perfectamente por su ambiente de pasillos sin perturbación alguna. Además del recorrido de cada pasillo también realiza los giros en las esquinas para tomar su nuevo rumbo, esto demuestra que el modelo de movimiento imple-

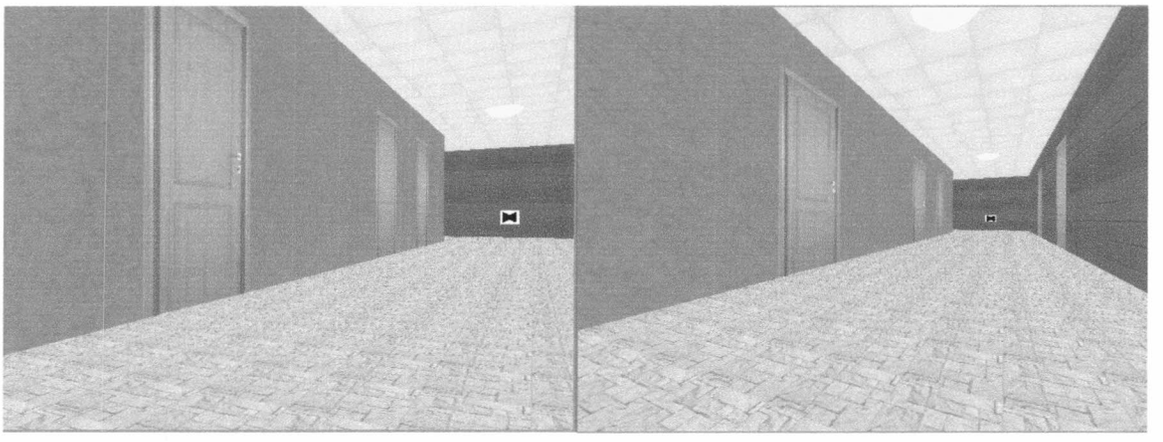

Figura 6-2: Izquierda: Apertura 60^o. Derecha: Apertura 90^o.

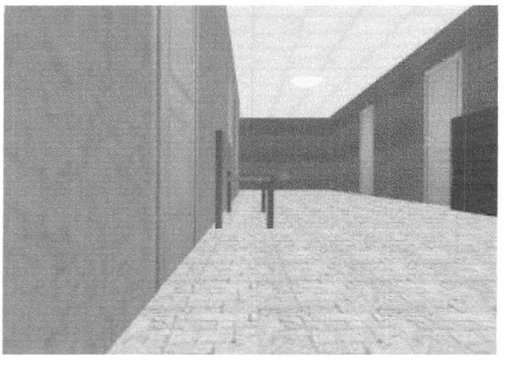

Figura 6-3: Posición indeterminada

Porcentaje Error Control	Varianza error en el sistema	Cantidad de localizaciones	Localizaciones positivas	Promedio Error	Desvío Error	Orientaciones positivas	Promedio Error	Desvío Error
5%	1 cm	4,3	80%	0,27cm	0,31 cm	60%	6°	15°
10%	1 cm	5,2	82%	0,19 cm	0,27 cm	68%	4°	11°
15%	1 cm	6,1	85%	0,23 cm	0,19 cm	70%	3°	5°
20%	1 cm	6,8	81%	0,23 cm	0,08 cm	63%	5°	10°

Figura 6-4: Resultados obtenidos utilizando diferentes porcentajes de error en el Sistema de Control de Movimiento con 50 ejecuciones

mentado funciona correctamente sin la incorporación de ruido generado por factores externos.

Se utilizaron los valores calculados en los experimentos realizados con el sistema de localización para generar las matriz de covarianza de la medición, que se utiliza en el paso correctivo de Kalman. Se fijó como umbral de incertidumbre para el posicionamiento un valor de 30 cm (± 5 cm) para los dos ejes que controlan el posicionamiento y un valor de 20^o ($\pm 10^o$) para la orientación. Valores por debajo de estas cotas producen que el sistema de autolocalización sea utilizado de manera muy seguida alejándonos del objetivo de tener un sistema bajo en cuanto al costo computacional. Valores por encima de estas cotas producen que el robot se acerque o navegue hacia las paredes haciendo que el sistema de autolocalización no tenga efecto.

Con estos valores se realizaron pruebas con diferentes valores de error para el sistema de control de movimientos del robot dejando fijo el desvío del error para el entorno donde el robot se mueve. La tabla 6-4 muestra los resultados obtenidos.

Con los resultados obtenidos podemos concluir que al aumentar el tamaño del error en el sistema de control del robot la incertidumbre sobre la localización crece con mayor velocidad. En consecuencia, el algoritmo de localización y el paso correctivo de Kalman se utilizan más veces haciendo que el desplazamiento sea más lento, además conlleva a que el ajuste realizado en la mediciones consecutivas generan errores más pequeños. Se usó el 20% como cota superior dado que utilizar un 25% de error en el control de movimientos produce que la cantidad de autolocalizaciones convierta al sistema en ineficiente ya que debe utilizar las mismas casi en forma continua.

Una alternativa a este problema fue aumentar el umbral de la incertidumbre para tener autolocalizaciones y ajustes más espaciados, pero el resultado de este cambio mostró que el

robot se acercaba más rápidamente hacia las paredes del pasillo. Este hecho produjo que la autolocalización no sea posible y en algunas circunstancias produjo que el robot chocara contra la pared antes de detectar que la incertidumbre era excesiva.

Concluimos entonces que era necesario un equilibrio entre el umbral de la incertidumbre en la posición y el porcentaje de error en el sistema de control. Los valores, que consideramos óptimos, que se utilizaron para realizar diferentes ejecuciones de prueba son 30 cm y 20^o como umbrales de incertidumbre en la posición y orientación respectivamente, además un valor de entre 5% y 15% como porcentaje de error en el controlador de movimientos del robot.

En caso de implementarse en un robot real, los valores de error del sistema de control son conocidos, y en el caso de que sean menores a estos porcentajes, nos puede permitir reducir el umbral de la incertidumbre. Esto significará que las autolocalizaciones y ajustes tienden a ser aún más espaciados en relación al valor empírico obtenido en estas pruebas.

Acerca del error en el sistema, al realizar recorridos con varianzas mayores a 2 cm encontramos que como nuestros desplazamientos son pequeños (5 cm), valores de error altos desnaturalizaban los movimientos originales. También comprobamos que al aumentar la varianza, la cantidad de autolocalizaciones tiende es mayor por el crecimiento de la incertidumbre. En definitiva, establecimos que una varianza de 1 cm para el error en el sistema era óptima para nuestro modelo.

La distancia a trasladarse y el ángulo de rotación utilizado en cada paso se definieron con valores de 5 cm y 3^o respectivamente, los cuales se obtuvieron realizando pruebas y observando los resultados, de manera que pueda reflejarse en el simulador la sensación de movimiento continuo en cada paso. Al utilizar valores más grandes se visualizaban saltos en los movimientos debido a que los errores producidos son relativos al tamaño del movimiento, y valores más pequeños producían errores imperceptibles que terminaban siendo casi nulos y no factibles de ser acumulados a lo largo del recorrido. Por lo tanto, estos valores mostraron ser óptimos para reflejar tanto el desplazamiento continuo del robot como para acumular el error generado en cada paso.

Un punto que no fue tenido en cuenta en nuestro modelo de desplazamiento fueron la perturbaciones, que entendemos como características particulares del ambiente que afectan los movimientos del robot. Estas características aparecen en forma imprevista en diferentes lugares

del ambiente y afectan el resultado de los desplazamientos, ejemplos pueden ser un hueco en el piso del ambiente, una imperfección en la lámina que recubre el piso del ambiente, etc.

La figura 6-5 muestra un ejemplo de una ejecución de una navegación en un pasillo donde las posiciones reales simuladas están en verde y las posiciones estimadas por el robot, en rojo. Puede observarse claramente en esta figura los puntos de color azul de la trayectoria donde la posición estimada por el robot es corregida por el método de autolocalización y el ajuste del paso correctivo de Kalman, luego de lo cual la posición estimada se acerca correctamente a la posición del robot en el simulador. Puede verse una navegación típica en el sitio web [WWW/01].

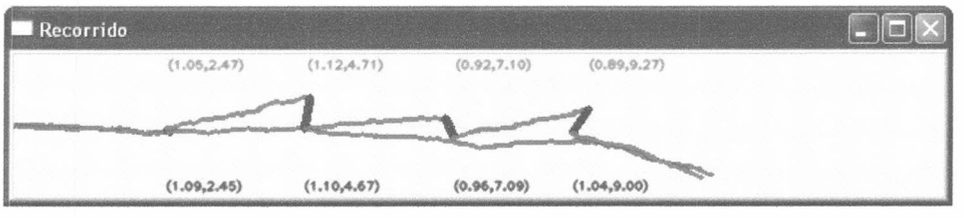

Figura 6-5: Recorrido de un pasillo

Capítulo 7

Conclusiones

Hemos desarrollado un sistema de navegación de robots para ambientes interiores tipo pasillo usando un filtro extendido de Kalman (EKF), al cual se le introdujo una ligera modificación para economizar el gasto computacional ocasionado por el método de autolocalizacion.

Recordemos: El paso predictivo ejecuta los movimientos del robot, estima su posición final y acumula la incertidumbre luego de cada movimiento. El paso correctivo emplea la información de localización obtenida mediante el método basado en *Invariantes Proyectivos*, el cual nos permite concentrarnos especificamente en calibrar la cámara para eliminar las aberraciones ópticas y además evita el almacenamiento de complejos esquemas para representar el ambiente.

En nuestra propuesta se itera repetidas veces utilizando sólo el paso predictivo y esperando a que el valor de la incerteza en la localización supere un umbral predeterminado. Cuando esto ocurre, se ejecuta el paso correctivo que utiliza el mecanismo de autolocalización haciendo que el robot conozca de manera más precisa su localización en el ambiente y permitiendo que el ciclo de nuevos desplazamientos comience nuevamente.

Para modelar el sistema de movilidad de un robot y asemejarlo a la realidad se introdujeron dos variables aleatorias en la definición del EKF, una modela la incerteza en los movimientos del robot, y la otra modela los factores externos de distorsión en el desplazamiento.

El método de autolocalización propuesto no necesita de complejos esquemas para representar el ambiente por el cual navega, esto es otra manera de economizar en lo que respecta a la representación y procesamiento del ambiente.

Encontramos limitaciones en el método de *Invariantes Proyectivos*, de forma tal que el

método fue complementado para resolver o atenuar estos problemas. El problema principal radica en el hecho de no poder detectar los puntos característicos necesarios, ya sea debido a estar próximos a una de las paredes laterales del pasillo o por estar próximos al fin de pasillo. Para esta situación particular se estableció un método que permite, además de localizar, estimar la orientación del robot a partir de la imagen, y definimos para ello una marca artificial que se ubica al final de los pasillos, la cual es reconocida mediante descriptores de Fourier.

Realizamos experimentos que comprobaron la eficacia de la estrategia de navegación, cuyos resultados brindan importantes datos de entrada para la definición de un robot físico adecuado para la implementación de esta estrategia.

Nueva investigación debe realizarse para definir qué otras características naturales se pueden utilizar para extender el método a otros tipos de ambiente.

7.1 Futuros Trabajos

En esta sección enumeramos distintos puntos que consideramos de importancia, como futuros trabajos para mejorar y completar la estrategia de navegación, así como para avanzar en la construcción de un robot físico adecuado para la aplicación de la misma.

- Utilizar otros modelos de movimiento para corroborar si el esquema propuesta se adapta fácilmente a otros modelos de robots.
- Definición de la arquitectura de hardware y software de base más apropiada para la implementación de la estrategia de Navegación que presentamos en este trabajo.
- Validar los resultados implementando el sistema en un robot real. Será necesario resolver los problemas que introducen la reflexión de la luz o las sombras en las imágenes, debidos a la iluminación natural o artificial.
- Definición y reconocimiento de marcas artificiales para resolver la localización cuando el robot se encuentra demasiado cerca de una pared.
- Utilización de cámaras de ángulo de apertura variable para aumentar la tasa de localización efectiva, intentando utiliizar otros valores cuando la apertura definida como óptima fracasa en casos aislados.

- Estudiar, y eventualmente corregir, los problemas que pueden producir algunos efectos de distorsión óptica, producto del ángulo de apertura de la cámara.
- Definición de nuevas clases de puntos característicos colineales en ambientes interiores.
- Complementar el método de navegación con métodos de relevamiento del ambiente para la extracción automática de puntos característicos. Ampliar el método para que no trabaje basado en un mapa, sino que releve el ambiente buscando puntos colineales que puedan utilizarse como puntos característicos.
- Implementación de la detección de obstáculos mediante el método de Invariantes Proyectivos.
- Ampliar el modelo de desplazamientos para incorporar la existencia de perturbaciones del ambiente.

Referencias

[BRO/92] R.G. Brown y P.Y.C. HWang, "Introduction to Random Signals and Applied Kalman Filtering", Second Edition, John Wiley & Sons, Inc., 1992

[CAN/86] Canny, "A Computational Approach to Edge Detection", *IEEE Trans. Pattern Analysis Mach. Intell*, 8(6), 1986.

[CHO/97] H. Choset, I. Konukseven, and A. Rizzi, "Sensor Based Planning: A Control Law for Generating the Generalized Voronoi Graph," *Proc. Eighth IEEE Int'l Conf. Advanced Robotics*, pp. 333-8, 1997

[HAR/04] R. Hartley y A. Zisserman, "Multiple View Geometry in Computer Vision", Cambridge University Press, March 2004.

[KAK/02] G. DeSouza and A. Kak, "Vision for Mobile Robot Navigation: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 2, February 2002

[KOS/92] A. Kosaka and A. C. Kak, "Fast Vision-Guided Mobile Robot Navigation Using Model-Based Reasoning and Prediction of Uncertainties," *CVGIP: Image Understanding*, Vol. 56, No. 3, November, pp. 271~329, 1992

[KYO/97] Kyoung Sig Roh, Wang Heon Lee, In So Kewon, "Obstacle Detection and Self Localization without Camera Calibration Using Projective Invariants," *IEEE IROS'97*, September, 1997.

[LOW/05] D. G. Lowe S. Se and J. J.Little, "Vision-based global localization and mapping for mobile robots," *IEEE Transactions on Robotics*, vol. 21, pp. 364-375, 2005

[MEN/93] M. Meng and A.C. Kak, "NEURO-NAV: A Neural Network Based Architecture for Vision-Guided Mobile Robot NavigationUsing Non-Metrical Models of the Environment," *Proc. IEEE Int'l Conf. Robotics and Automation*, vol. 2, pp. 750-757, 1993

[MOR/85] H.P. Moravec and A. Elfes, "High Resolution Maps from Wide Agle Sonar," *Proc. IEEE Int'l Conf. Robotics and Automation*, pp. 116-121, 1985

[SAN/93] J. Santos, G. Sandini, F. Curotto, y S. Garibaldi, "Divergent Stereo for Robot Navigation: Learning from Bees," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition*, 1993

[SHA/02] Shapiro and Stockman, *Computer Vision*, Prentice Hall, 2002

[SMI/88] R. C. Smith, M. Self, y P. Cheeseman, "Estimating uncertain spacial relationships in robotics", *Uncertainty in Artificial Intelligence 2*, pp. 435-461, 1988

[SOR/70] H.W.Sorenson, "Least-squares estimation: from Gauss to Kalman", *IEEE Spectrum*, vol. 7, pp.63-68, Julio 1970

[THR/98] Thrun S, Burgard W. y Fox D., "A Probabilistic Approach to Concurrent Mapping and Localization for Mobile Robots", *Machine Learning and Autonomous Robots* (joint issue), 31/5, 1–25 (1998)

[WAN/98] C.Ming Wang, "Location Estimation and Uncertainty Analysis for Mobile Robots", IEEE 1998

[WEL/03] Greg Welch y Gary Bishop, "An Introduction to the Kalman Filter ", Technical Report 95-041, Univ North Carolina, Dept Computer Science, May 23, 2003

[WWW/01] University of Buenos Aires Image Processing Group, http://www.dc.uba.ar/people/grupinv/imagenes/".

[WWW/02] http://www.opengl.org

Apéndice A

El Operador de Canny

Presentamos en esta sección una descripción conceptual de los fundamentos de la definición del Operador de Canny para la detección de bordes. Puede encontrarse la definición completa del operador en [CAN/86].

Para facilitar la exposición, consideramos señales unidimensionales, concentrándonos en la definición de un borde como una función escalón ideal, lo cual puede definirse como la función Signo: $Sgn(x) = \frac{|x|}{x}$, ilustrada en la Figura A-1.

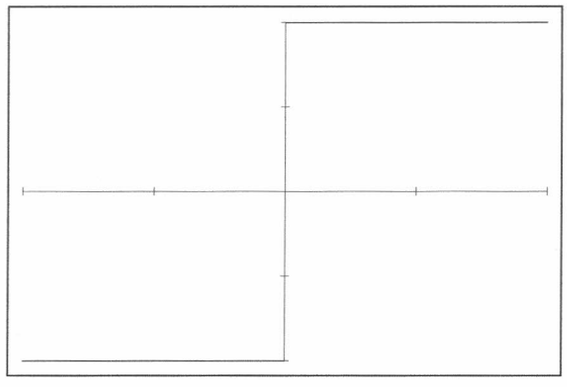

Figura A-1: Función Signo

Si bien no es un modelo exacto, para modelar el efecto del ruido introducido en la imagen por los sensores, el muestreo y la cuantización, Canny suaviza la función escalón utilizando un operador Gaussiano, ya que asume que el ruido del proceso es Gaussiano. Esto se lleva a cabo

mediante la convolución de la imagen con dicho operador Gaussiano y sus derivadas. Siguiendo con el caso unidimensional, ilustramos en las figuras A-2, A-3 y A-4, la función Gaussiana, centrada en cero, y sus derivadas primera y segunda, respectivamente.

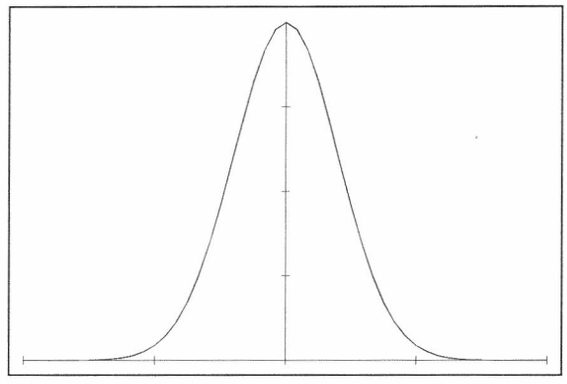

Figura A-2: Función Gaussiana $G(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-x^2}{2\sigma^2}}$

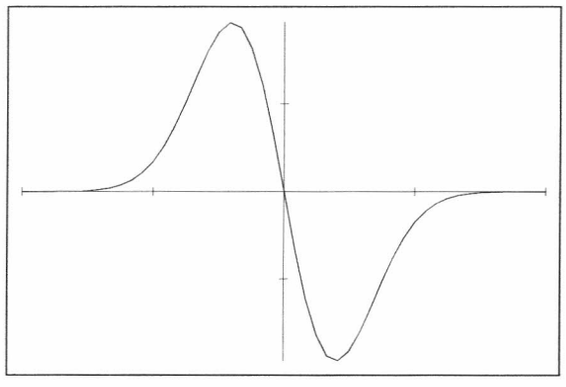

Figura A-3: Derivada primera de la Función Gaussiana: $G'(x) = \frac{-x}{\sqrt{2\pi}\sigma^3} e^{\frac{-x^2}{2\sigma^2}}$

Considerando el borde escalón ideal, cuando realizamos la convolución del borde con la función Gaussiana obtenemos un borde suavizado, como vemos en la Figura A-5.

Luego, podemos ver que la presencia de un borde puede detectarse como un extremo local en F' o como un cero en F''(figuras A-6 y A-7).

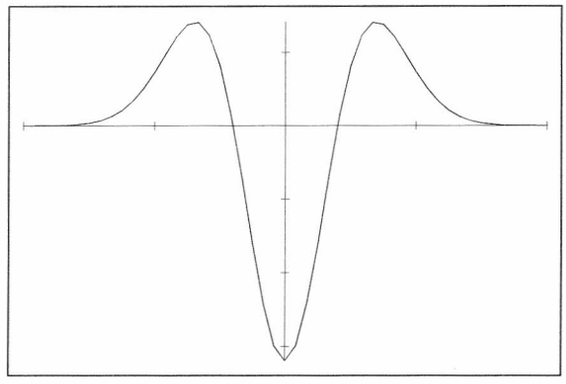

Figura A-4: Derivada segunda de la Función Gaussiana: $G''(x) = -\frac{1}{\sqrt{2\pi\sigma^3}} e^{\frac{-x^2}{2\sigma^2}} \left[1 - \frac{x^2}{\sigma^2}\right]$

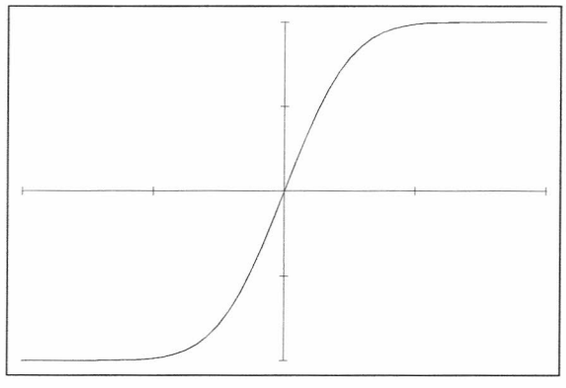

Figura A-5: Borde Ideal Suavizado $F(x)$

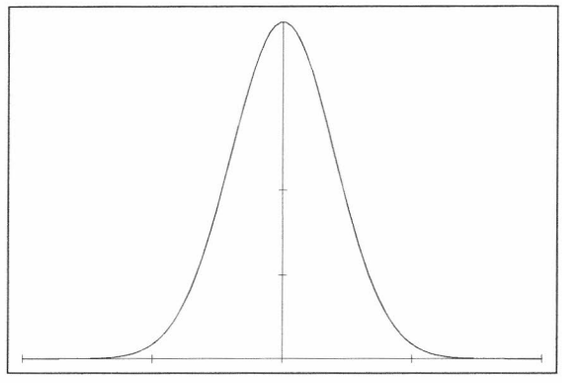

Figura A-6: Derivada primera del borde suavizado

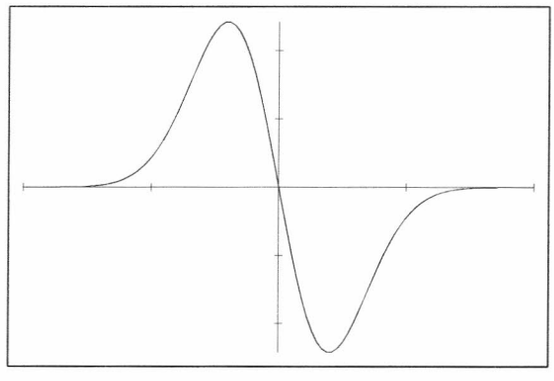

Figura A-7: Derivada segunda del borde suavizado

Para la implementación del método de detección podemos ver además que es posible combinar las etapas de suavizado de la imagen y detección de bordes en una única convolución con la primera derivada de la función Gaussiana, con una búsqueda posterior de máximos locales, o con la segunda derivada Gaussiana para una búsqueda de ceros. Esto es así pues la primera derivada de la imagen, convolucionada con la función Gaussiana es equivalente a la imagen resultante de la convoluciónpara con la primera derivada de la Gaussiana. Como mencionamos al comienzo de este apéndice, hemos considerado hasta aquí señales unidimensionales para facilitar la exposición, pero el análisis se extiende directamente para señales bidimensionales. Por lo tanto, siendo f la imagen y G la Gaussiana, entonces

$$\nabla [G(x,y) * f(x,y)] = \nabla [G(x,y)] * f(x,y) \tag{A.1}$$

Implementación

Los objetivos del operador de Canny son los siguientes:

1. **Buena Detección:** habilidad para detectar todos los bordes reales.
2. **Buena Localizacion**: minimización de la distancia entre el borde detectado y el real.
3. **Respuesta Clara**: obtención de una única respuesta por borde.

Para lograr estos objetivos, el proceso de detección de bordes incluye las siguientes etapas:

Etapa 1. Suavizado de la Imagen

La imagen es suavizada mediante una función Gaussiana de dos dimensiones cuyo ancho es definido mediante un parámetro. En la práctica la convolución en dos dimensiones con la función Gaussiana es muy costosa computacionalmente por lo que es común aproximar esto mediante dos Gaussianas unidimensionales, una alineada con el eje x, y la otra alineada con el eje y. Esto produce dos valores para cada pixel, en vez de uno.

Etapa 2. Diferenciación

En esta etapa se implementa la diferenciación de la imagen suavizada en ambas direcciones. El cálculo del gradiente en cualquier dirección puede calcularse a partir del gradiente conocido en las direcciones x e y.

Etapa 3. Detección de máximos

Habiendo encontrado una medida de la intensidad del cambio en cada punto de la imagen, los bordes deben ser ubicados en los puntos de máximo cambio local. Un máximo local ocurre en un pico en la función gradiente o, alternativamente cuando existe un cero su derivada. En esta etapa se eliminan los puntos no-máximos del gradiente, aunque sólo los que se encuentren en dirección perpendicular al borde. Los puntos no-máximos paralelos al borde se mantienen dada la expectativa de continuidad del borde.

Etapa 4. Umbralado

El umbralado en el método de Canny se lleva a cabo mediante un proceso llamado *histéresis.* La mayoría de los procesos de umbralado utilizan un único umbral, esto produce la detección de bordes con discontinuidades cuando los pixels que forman un borde real en la imagen fluctúan por encima y por debajo del umbral. Para contrarrestar este efecto el proceso de hysteresis define dos valores para decidir si un punto pertenece a un borde o no: umbral superior y umbral inferior. Si el valor del gradiente en un punto es mayor al umbral superior, inmediatamente se asume que este punto pertenece a un borde. Por otro lado, si el valor es menor al umbral inferior, el punto es deshechado directamente. En cambio, si el valor de un punto está entre el umbral superior y el inferior se aceptan siempre y cuando el punto se encuentre conectado a puntos con altos valores. Esto disminuye drásticamente la discontinuidad de los bordes detectados.

Apéndice B

La Transformada de Hough

La transformada de Hough es una técnica utilizada para identificar características de alguna forma particular dentro de una imagen. La transformada de Hough *clásica* se basa en la definición paramétrica de las características que se desean identificar y es comunmente utilizada para la detección de curvas regulares con líneas, círculos, elipses, etc. Se define también la transformada de Hough *generalizada* la cual no se basa en descripciones analíticas de las características o las formas, brindando una técnica más general, pero de mayor complejidad computacional. Exponemos en este apéndice la transformada clásica, ya que se brinda como herramienta suficiente para la detección de líneas rectas, necesarias para la detección de los puntos característicos utilizados por el método de auto-localización del robot.

La idea central detrás de la transformada de Hough para la detección de líneas es que cada punto de la imagen procesada aporta su contribución a las líneas que pasan por ese punto. En la figura B-1, vemos tres puntos que hacen su aporte a la definición de la línea *L1*, mientras que para las siguientes líneas, *L2*, *L3* y *L4*, vemos dos puntos que las definen en cada caso.

Podemos describir analíticamente una línea recta de diferentes maneras. Una alternativa es hacerlo mediante una definición *paramétrica* de la siguiente forma:

$$x \cos\theta + y \sin\theta = r \tag{B.1}$$

donde r el la distancia de la recta al origen y θ es la orientación de la recta con respecto a la

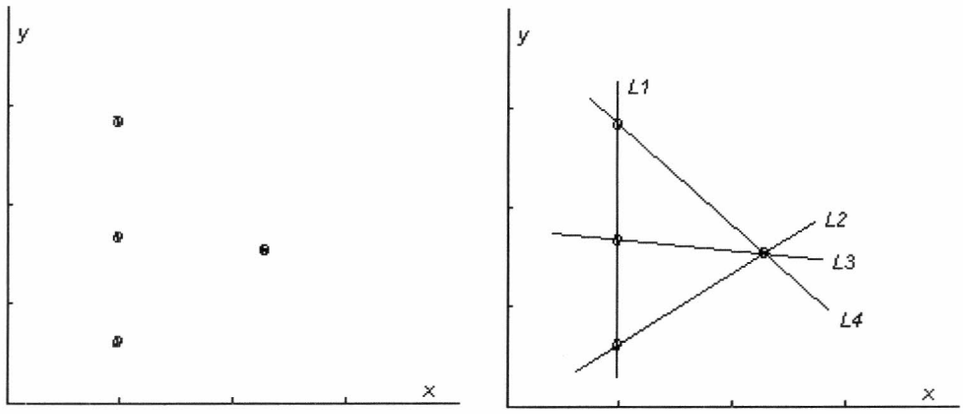

Figura B-1: Votación de puntos. A la izquierda observamos un conjunto de 4 puntos en el plano (x, y). A la derecha observamos las cuatro posibles líneas rectas en este espacio. La línea L1 es "votada" por 3 puntos, mientras que las líneas L2, L3 y L4 son votadas por dos puntos.

abcisa, como se ilustra en la figura B-2.

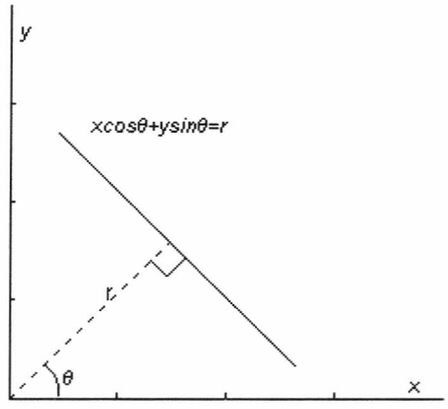

Figura B-2: Definición paramétrica de la recta

Desde el punto de vista del procesamiento de imágenes, podemos considerar al mapa de bordes de una imagen como un conjunto de puntos en un plano. Cada punto (x_i, y_i) en la imagen correspondiente al mapa de bordes puede considerarse como constantes en la ecuación paramétrica de la recta, mientras que θ y r se asumen como variables desconocidas. Para cada punto (x_i, y_i) de la imagen, se obtiene una curva sinusoidal, formada por el conjunto de todos

los posibles (θ, r), correspondientes a todas las posibles rectas que pasan por (x_i, y_i). Al espacio de (θ, r) lo llamamos *Espacio de Parámetros Polares de Hough.* Este proceso corresponde a una transformación de puntos en rectas, del espacio cartesiano de la imagen al Espacio de Hough. Los puntos que son colineales en el espacio de la imagen generan curvas que se intersectan en un punto (θ, r) común, por lo que cuantas más curvas se intersecten en (θ, r), más evidencia hay de que existen puntos en el espacio de la imagen que forman una recta, determinada por la ecuación $x \cos\theta + y \sin\theta = r$.

Esta transformación se implementa cuantizando el Espacio de Hough en intervalos finitos, llamados *celdas de acumulación.* El algoritmo transforma cada (x_i, y_i) de la imagen en una curva discretizada (θ, r), incrementando el acumulador de la celda correspondiente. Los máximos locales en el Espacio de Hough presentan evidencia de la existencia de una línea en la imagen.

B.1 Ejemplo e Implementación

Para ilustrar la aplicación de la transformada de Hough, podemos observar las figuras B-3 y B-4, correspondientes a una imagen, y su correspondiente mapa de bordes, en este caso obtenido mediante el operador de Canny.

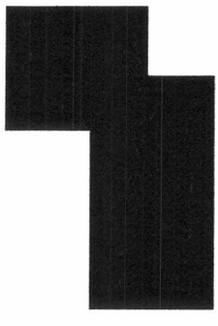

Figura B-3: Figura con bordes rectos.

Si utilizamos los puntos del mapa de bordes como los datos de entrada de la transformada de Hough, obtenemos el correspondiente conjunto de curvas en el Espacio de Hough. Utilizando

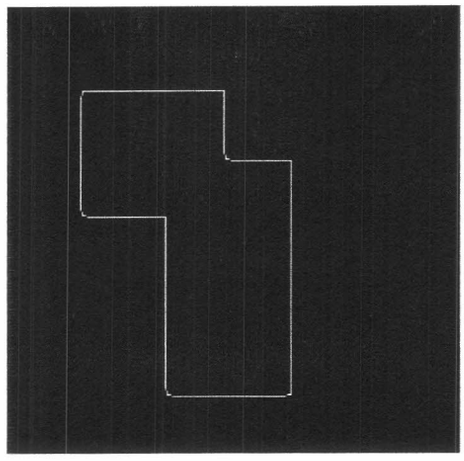

Figura B-4: Mapa de Bordes

las magnitudes correspondiente a las celdas de acumulación como intensidades de puntos de una imagen, podemos visualizar el Espacio de Hough en figura B-5.

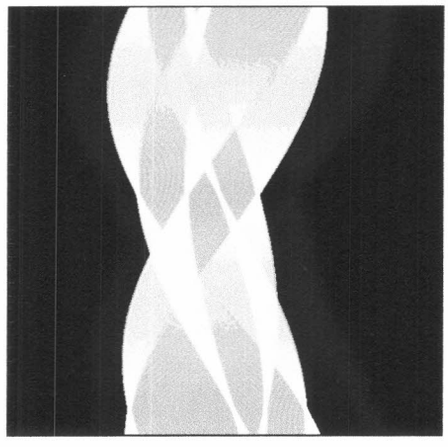

Figura B-5: Espacio de Hough

Los puntos más brillantes de esta imagen se corresponden con máximos locales en la matriz de acumulación lo cual indica fuerte evidencia de la existencia de una línea recta en la imagen original. Obteniendo los máximos locales en el Espacio de Hough, obtenemos un conjunto de pares (θ_i, r_i), que corresponden a rectas paramétricas en la imagen original. Siguiendo nuestro

ejemplo, superponiendo las rectas obtenidas sobre la imagen original, podemos ver las rectas detectadas en la figura B-6.

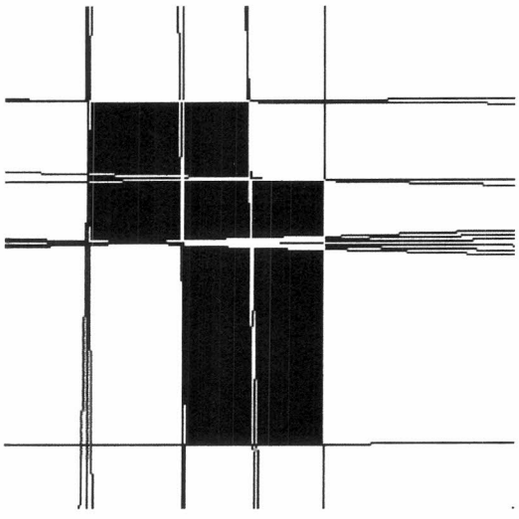

Figura B-6: Rectas Detectadas

Puede observarse en la figura B-6 que se detectaron algunas líneas espúreas, similares a las líneas correctas, lo cual se debe a la cuantización realizada sobre el espacio (θ, r).Por esto, es necesario seleccionar cuidadosamente los intervalos de cuantización, junto con la implementación de técnicas accesorias para eliminar líneas no deseadas. No existe una regla general para la elección del intervalo de cuantización, ni para afinar la selección de líneas; la decisión debe tomarse en función de las características particulares de la imagen a procesar.

Apéndice C

El Filtro de Kalman

El filtro de Kalman es una herramienta matemática formada por un conjunto de ecuaciones que proveen una solución recursiva pero eficiente, desde el punto de vista computacional, al problema de mínimos cuadrados. Actualmente es utilizado en temas de investigación y aplicación en el área de navegación autónoma o asistida de robots.

C.1 Método de Mínimos Cuadrados

El método de mínimos cuadrados fue descripto por Gauss en su libro "Teoría Motus". El punto de partida de éste método fueron observaciones y mediciones, las cuales según su descripción eran "sólo aproximaciones a la verdad", por lo tanto el objetivo de este método era lograr que los cálculos a realizar nos acerquen tanto a la verdad como sea posible.

El método de mínimos cuadrados es definido entonces como un mecanismo matemático que nos permite determinar la función que mejor se aproxima a un conjunto de datos observados minimizando la suma de los cuadrados de los desvíos, siendo estos últimos la diferencia entre los valores observados y los esperados.

Supongamos que m mediciones están disponibles en un grupo de momentos discretos en el tiempo, hay que determinar el parámetro x que se relaciona en forma lineal de acuerdo a la ecuación $z_k = H_k x + v_k$, donde

z_k, medición en el instante k

x, parámetro para obtener la medición

H_k, relación líneal del parámetro con la medición

v_k, es el error ocurrido en cada medición

El residuo asociado con cada medición se calcula como $r_k \widehat{=} z_k - H_k \widehat{x}_k$, y además $k = 0, 1, 2, ..., n$

El método de mínimos cuadrados esta relacionado con determinar el valor más probable de x (es decir $\widehat{x}_n$), el cual se define como el valor que minimiza la suma de los cuadrados de los residuos. Hay que elegir x tal que $L_n = \frac{1}{2} \sum_{k=0}^{n} [z_k - H_k x]^t W [z_k - H_k x]$ sea minimizado, donde W indica el grado de confianza en la medición que se utiliza, de esta manera es posible otorgar pesos a cada una de las mediciones en forma independiente.

La solución al problema acarrea un costo computacional muy alto, debido a que la solución se obtiene al resolver un sistema de ecuaciones, el sistema es más complejo de acuerdo a la cantidad de variables y ecuaciones que contiene, esto último depende de las mediciones utilizadas para realizar la estimación.

C.2 Teoría del filtro de Kalman

Al considerarse el problema de estimación de una señal s_n, variante en el tiempo, a partir de un grupo de mediciones realizadas $z = (z_0, z_1, ...z_n)$. Asumimos que la estimación de s_n llamada $\widehat{s}_n$ puede ser calculada como una combinación lineal de z_i, siendo $\widehat{s}_n = \sum_{i=0}^{n} H_{n,i} * z_i$.

La ganancia del filtro $H_{n,i}$ se selecciona para que el error cuadrático medio sea minimizado, es decir que se debe elegir $H_{n,i}$ tal que $M = E[(s_n - \widehat{s}_n)^t (s_n - \widehat{s}_n)]$ sea minimizado. Es sabido que para minimizar M es condición necesaria y suficiente que el error en la estimación sea ortogonal a la medición, es decir que $E[\widehat{s}_n z_i^t] = 0$.

El problema aquí planteado es bastante diferente al problema de mínimos cuadrados planteado por Gauss pero según se explica en [SOR/70] si se modifica la estructura de los procesos para obtener la señal como un sistema dinámico teniendo la forma $x_{i+1} = \Phi_i x_i + w_i$, donde Φ_i es una función de control sobre el estado actual del proceso y w_i una variable aleatoria que representa el ruido blanco, y las mediciones se describen ahora como $z_i = s_i + v_i = H_i x_i + v_i$, donde v_i es una variable aleatoria que representa ruido blanco, estas nuevas ecuaciones permiten resolver la ecuación mencionada en el parráfo anterior de manera más sencilla.

Resumiendo, el filtro de Kalman ataca el problema de tratar de estimar un estado $x \in R^n$ de un proceso discreto en el tiempo, la ecuación que modela los cambios de estados está dada por $x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1}$con una medición $z \in R^n$ tal que $z_k = Hx_k + v_k$ donde

A es una matriz que relaciona el estado anterior con el próximo

B es una matriz que relaciona el estado con una entrada de control sobre el proceso

H es un matriz que relaciona el estado con la medición

x_{k-1}es el estado anterior del sistema

z_k es la medición obtenida

Las variables aleatorias w_k y v_k representan el ruido en el proceso y en la medición respectivamente. Son independientes y representan ruido blanco gaussiano, teniendo una distribución

$p(w) = N(0, Q)$

$p(v) = N(0, R)$

Representamos con el símbolo $^-$ a los valores de las variables obtenidas en el paso de predicción y con el símbolo $^+$ a los valores de variables obtenidas en el paso de medición, por ejemplo $\widehat{x}_k^-$ es la posición del robot obtenida en la predicción y $\widehat{x}_k^+$es la posición del robot obtenida en el paso de medición.

El primer objetivo al derivar las ecuaciones del filtro de Kalman es encontrar una expresión para el estado posterior $\widehat{x}_k^+$ como una combinación lineal del estado anterior $\widehat{x}_k^-$ y utilizar una diferencia pesada entre la medición actual z_k y una predicción de la medición $H\widehat{x}_k^-$ quedando establecida entonces la siguiente ecuación $\widehat{x}_k^+ = \widehat{x}_k^- + K(z_k - H\widehat{x}_k^-)$. El término $(z_k - H\widehat{x}_k^-)$ se denomina innovación o residuo y refleja la diferencia entre la medición predicha y la medición real obtenida.

La matriz K es el factor que minimiza el error que se acumula en la matriz covarianza. Para obtener esta matriz se substituye en la ecuación anterior la definición del error, luego derivando en K e igualando a cero. Por último, se despeja K quedando la ecuación de la siguiente manera:

$K_k = P_k^- H^t (HP_k^- H^t + R)^{-1} = \frac{P_k^- H^t}{HP_k^- H^t + R}$

Cuando $R \to 0$ entonces $lim\ K = H^{-1}$, por lo tanto el residuo actúa con más peso, siendo la medición real la que gana peso para el cáculo del estado próximo. Pero Si $P_k^- \to 0$, entonces $lim\ K = 0$, el residuo pierde peso, siendo el estado anterior quien gana peso para calcular el nuevo estado del sistema.

C.3 Algoritmo discreto del filtro de Kalman

Las ecuaciones que forman el filtro de Kalmna pueden ser manipuladas algebraicamente de diversas maneras. Por ejemplo, la ecuación que representa la ganancia es de las más populares conocidas.

El filtro de Kalman modela un proceso estimando el estado actual en algún momento del tiempo, y obtiene mediciones para corrobar las estimaciones realizadas. Las ecuaciones se agrupan en dos conjuntos: las de predicción del estado y las de corrección. Las primeras son responsables de hacer una proyección en el tiempo del estado del sistema y del error con el que es generado. Las segundas se relacionan con las mediciones y corrigen las estimaciones que se realizan en la primer etapa. Las ecucaciones de predicción-corrección trabajan juntas y en forma recursiva para resolver problemas numéricos.

Las ecuaciones del paso predictivo son dos: la primer ecuación se encarga de estimar el nuevo estado luego de realizar una acción en el proceso, la segunda estima el valor de la covarianza del error en el sistema

$\widehat{x}_k^- = A\widehat{x}_{k-1}^+ + Bu_{k-1} + w_{k-1}$

$P_k^- = AP_{k-1}^+A^t + Q$

Las ecucaiones del paso correctivo son tres: la primera calcula la ganancia de Kalman que representa la incidencia de la medición realizada para el cálculo del nuevo estado del sistema, la segunda calcula el nuevo estado del sistema que es una suma pesada entre a la predicción realizada en el paso anterior y el término que utiliza la ganancia sobre la innovación, la tercera ajusta el valor de la matriz de covarianza del error en el sistema.

$K_k = P_k^- H^t (HP_k^- H^t + R)^{-1}$

$\widehat{x}_k^+ = \widehat{x}_k^- + K_k(z_k - H\widehat{x}_k^-)$

$P_k^+ = (I - K_k H)P_k^-$

El proceso se repite en forma continua, primero se realiza el paso predictivo, obteniendo la estimación del nuevo estado y de la matriz de covarianza del error, luego se ajusta la predicción utilizando una medición en el sistema, al terminar el paso correctivo el proceso vuelve a comenzar. La naturaleza recursiva del Filtro de Kalman es una de las características principales que hacen que sea una implementación práctica y computacionalmente factible.

C.4 El filtro extendido de Kalman

Como vimos anteriormente el filtro Kalman ataca el problema de estimar un estado $x \in R^n$ de un proceso discreto en el tiempo el cual esta gobernado por una ecuación lineal estocástica. Pero la mayoría de los procesos reales se modelan con funciones no lineales, entonces al modificar el filtro de Kalman, creando un nuevo modelo que linearice el proceso utilizando términos de la serie de Taylor nos referimos al Filtro Extendido de Kalman (EKF).

El proceso está gobernado ahora por ecuaciones estocásticas no lineales, por lo cual es necesario modificar las ecuaciones iniciales que se plantearon al presentar el filtro de Kalman, siendo la función del cambio de estado $x_k = f(x_{k-1}, u_{k-1}, w_{k-1})$ con una medición $z \in R^n$ tal que $z_k = h(x_k, v_k)$.

Las ecuaciones para el paso predictivo del EKF son

$\widehat{x}_k^- = f(\widehat{x}_{k-1}^+, u_{k-1}, w_{k-1})$

$P_k^- = A_k P_{k-1}^+ A_k^t + W_k Q_k W_k^t$

donde

A_k, es la matriz jacobiana de derivadas parciales de f con respecto a x - $A_{[i,j]} = \frac{\partial f_{[i]}}{\partial x_{[j]}}(x_{k-1}, u_{k-1}, w_{k-1})$

W_k, es la matriz jacobiana de derivadas parciales de f con respecto a w - $W_{[i,j]} = \frac{\partial f_{[i]}}{\partial w_{[j]}}(x_{k-1}, u_{k-1}, w_{k-1})$

Q_k, es la matriz de covarianza del ruido en el proceso

Las ecuaciones para el paso correctito del EKF son

$K_k = P_k^- H_k^t (H_k P_k^- H_k^t + V_k R_k V_k^t)^{-1}$

$\widehat{x}_k^+ = \widehat{x}_k^- + K_k(z_k - h(\widehat{x}_k^-))$

$P_k^+ = (I - K_k H) P_k^-$

donde

H_k, es la matriz jacobiana de derivadas parciales de f con respecto a x - $H_{[i,j]} = \frac{\partial h_{[i]}}{\partial x_{[j]}}(x_k, v_k)$

V_k, es la matriz jacobiana de derivadas parciales de f con respecto a v - $V_{[i,j]} = \frac{\partial f_{[i]}}{\partial v_{[j]}}(x_k, v_k)$

R_k, es la matriz de covarianza del ruido en la medición

Los pasos predictivo y correctivo se repiten en forma recursiva al igual que el filtro normal de Kalman, agregando el beneficio de que la aproximación del sistema no lineal es más precisa.

C.5 Contribuciones del filtro de Kalman

Las contribuciones del filtro de Kalman son dos:

1. Es un procedimiento conveniente para ser ejecutado por una computadora digital. Se pueden implementar sistemas que utilicen directamente las ecuaciones del filtro de Kalman. Si bien existen procedimientos para resolver los sistemas de ecuaciones estos son complejos y consumen mucho tiempo computacional.

2. Kalman ubicó el problema en un marco general, lo cual motivó la generación de numerosos trabajos de investigación basados en sus ideas. El propio Kalman reconoció el potencial de este proceso para ser aplicado en distintas áreas, este fue sin duda un hecho poco habitual dado que supo comunicar el valor intrínseco de su método.