
**UNIVERSIDAD DEL VALLE**

**PROGRAMA DE POSGRADO EN INGENIERÍA ELÉCTRICA Y ELECTRÓNICA**

FACULTAD DE INGENIERÍA

# MAPEO Y LOCALIZACIÓN SIMULTÁNEA DE UN ROBOT MÓVIL EN AMBIENTES ESTRUCTURADOS BASADO EN INTEGRACIÓN SENSORIAL

## TESIS

QUE PARA OPTAR POR EL GRADO DE:

**MASTER EN INGENIERÍA**

CON ÉNFASIS EN ELECTRÓNICA

P R E S E N T A:

**JULIE STEPHANY BERRÍO PÉREZ**

TUTOR:

**EDUARDO F. CAICEDO BRAVO**


Cali - Valle del Cauca Julio 2012

**Jurado asignado:**

$1^{er}$ Jurado: Breyner Posso Bautista

$2^{o}$ Jurado: Juan Carlos Perafan Villota

Lugar donde se realizó la tesis:

Laboratorio de Robótica Móvil de la
Escuela de Ingeniería Eléctrica y Electrónica
de la Universidad Del Valle.

Director de tesis:

---

**Dr. Eduardo Francisco Caicedo Bravo**

Este trabajo se desarrolló en el Laboratorio de Róbotica Movil del grupo de investigación PSI (Percepción y Sistemas Inteligentes) suscrito a la Escuela de Ingeniería Eléctrica y Electrónica (EIEE) de la Universidad Del Valle (UV) bajo la tutoria de los doctores Eduardo Francisco Caicedo Bravo y Lina Maria Paz Perez. Se contó con el apoyo financiero del programa de becas de asistentes de docencia.

A mis padres Jairo A. Berrío y Aura S. Pérez, y mi abuelita Bertha Narvaez.

# Índice general

**RESUMEN** viii

**1. INTRODUCCIÓN** 1

**2. MAPEO Y LOCALIZACIÓN SIMULTÁNEA (SLAM) EN LA ROBÓTICA** 4

2.1. Metodología para SLAM . . . . . 5
2.2. Formulación del problema . . . . . 6
2.3. Soluciones al problema de SLAM . . . . . 9
2.3.1. Filtro de Kalman: . . . . . 9
2.3.2. Métodos basados en filtros de partículas . . . . . 12
2.3.3. Métodos basados en maximización de la expectativa . . . . . 13
2.3.4. Comparación (pros y contras) de métodos de solución del SLAM 15
2.4. Implementación . . . . . 16
2.5. Complejidad computacional . . . . . 16
2.5.1. Mejoramiento de la eficiencia computacional . . . . . 16
2.6. Representación del ambiente . . . . . 17
2.6.1. Clustering . . . . . 18
2.6.2. Extracción de líneas . . . . . 20
2.6.2.1. Algoritmos no robustos . . . . . 20
2.6.2.2. Algoritmos robustos . . . . . 21
2.7. Asociación de datos . . . . . 22

**3. SISTEMA DEL EKF-SLAM** 25

3.1. Plataforma de Desarrollo . . . . . 26
3.1.1. Sensores para la navegación . . . . . 26
3.1.1.1. Sensores Propioceptivos . . . . . 26
3.1.1.2. Sensores Exteroceptivos . . . . . 27
3.1.1.3. Sensores Comunmente Utilizados en Slam . . . . . 27
3.2. Extracción de Características . . . . . 28
3.2.1. Lecturas de los Sensores . . . . . 28
3.2.2. RANSAC . . . . . 29
3.2.3. Transformada de Hough . . . . . 31
3.2.4. Successive Edge Following (SEF) . . . . . 34

3.2.5. TLS Total Least Squares . . . . . 35
3.3. Asociación de Características . . . . . 38
3.3.1. Distancia de Mahalanobis . . . . . 39
3.3.2. Reducción de hipótesis . . . . . 39
3.3.2.1. Vectores normales a las rectas . . . . . 40
3.3.2.2. Proyección de rectas . . . . . 42
3.3.3. Algoritmo de Compatibilidad Individual . . . . . 43
3.3.4. Actualización de los Puntos Extremos . . . . . 43

**4. FILTRO DE KALMAN, EXTENCIÓN E IMPLEMENTACIÓN 47**
4.1. Filtro de Kalman . . . . . 47
4.1.1. Proceso a ser estimado . . . . . 47
4.1.2. Origenes Computacionales del Filtro . . . . . 48
4.1.3. Origenes Probabilisticos del Filtro . . . . . 49
4.2. Filtro de Kalman Extendido . . . . . 50
4.3. Implementación del Filtro de Kalman Extendido . . . . . 52
4.3.1. Filtro de Kalman Extendido en el Proceso del SLAM . . . . . 54
4.3.1.1. Predicción . . . . . 55
4.3.1.2. Re-observación de los hitos (Actualización) . . . . . 57
4.3.1.3. Integración del nuevo hito . . . . . 58
4.3.2. Complejidad Computacional . . . . . 59
4.3.2.1. Coste computacional por paso . . . . . 59
4.3.2.2. Coste computacional total . . . . . 60

**5. PRUEBA Y ANALISIS DE RESULTADOS 61**
5.1. Medio Ambiente Artificialmente Modelado . . . . . 61
5.2. Experimentos en entornos de interiores . . . . . 66
5.3. Aplicaciones . . . . . 70

**6. CONCLUSIONES 73**
6.1. Segmentación y Asociación de datos . . . . . 73
6.2. Filtro de Kalman Extendido . . . . . 73
6.3. Aplicaciones . . . . . 74
6.4. Resumen . . . . . 74

**A. PUBLICACIONES 76**

**BIBLIOGRAFÍA 77**

# Índice de figuras


2.1. Desarrollo del proceso de localización y mapeo simultáneo . . . . . 6
2.2. Esquema de localización y mapeo simultáneo . . . . . 7

3.1. Sistema del EKF-SLAM. . . . . 25
3.2. Parámetros del escáner láser . . . . . 29
3.3. Transformaciones punto a linea. a) Un punto con lineas de varios parametros $(m, b)$ a través de estos se transforma en una línea recta en el plano de parametros, b) Puntos colineales se trasnforman en líneas rectas que se intersectan en un único punto en el plano de parametros . . . . 31
3.4. Parametrización normal . . . . . 32
3.5. Transformaciones punto-curva. a) un punto con lineas a con varios parámetros $(\varphi, \rho)$ pasando a través de ella se transforma en una curva senosoidal en el plano de parámetros. b) Puntos colineales se transforman en curvas que se intersectan en un unico punto en el plano de parámetros. 33
3.6. Transformada de Hough con grilla 2D . . . . . 34
3.7. Seguimiento de Extremos Consecutivos . . . . . 35
3.8. Compatibilidad Individual. El mapa color naranja corresponde al predicho en el instante $k$ cuando las observaciones en negro son medidas. . . 39
3.9. Características que han sido predecidas, con sus respectivos parámetros. 40
3.10. Robot Navegando . . . . . 40
3.11. Parámetros de las rectas detectadas . . . . . 41
3.12. Asociación de datos. . . . . 42
3.13. Extremos de rectas proyectados. . . . . 42
3.14. Matriz de compatibilidad individual. . . . . 43
3.15. Punto extremo de líneas horizontales. . . . . 45
3.16. Punto extremo de líneas verticales. . . . . 45

4.1. Parámetros del filtro del Kalman . . . . . 53
4.2. Proceso de SLAM . . . . . 54

5.1. Resultado de la ejecución del EKF-SLAM en un pasillo, utilizando un escaner Laser. . . . . 62
5.2. Resultado de la ejecución del EKF-SLAM en un pasillo, utilizando un cinturon de ultrasonidos. . . . . 63

5.3. Mapa modelado en Mapper3 . . . . . 63
5.4. Resultado de la ejecución del EKF-SLAM para el mapa modelado. . . . 64
5.5. Resultado de la ejecución del EKF-SLAM para el mapa modelado. . . . 65
5.6. Error en cada paso iteración del proceso de EKF-SLAM (Laser), Error de odometría en magneta y error de la posición estimada en azul . . . . 66
5.7. Error en cada paso iteración del proceso de EKF-SLAM (Ultrasonido), Error de odometría en verde y error de la posición estimada en azul . . 66
5.8. Resultado de EKF-SLAM (laser) para un mapa cerrado. . . . . 67
5.9. Error de las características del mapa cerrado. . . . . 67
5.10. Resultado EKF-SLAM (laser) con datos reales. . . . . 68
5.11. Resultado EKF-SLAM (ultrasonido) con datos reales. . . . . 68
5.12. Error del mapa resultante del EKF-SLAM (laser) . . . . . 69
5.13. Mapa según los datos sin procesar de los sensores. . . . . 70
5.14. Resultado de EKF-SLAM (laser) . . . . . 71
5.15. Mapa en Autocad . . . . . 71
5.16. Mapa en Autocad . . . . . 72

# Índice de tablas

2.1. Lista de ventajas y desventajas de las diferentes estrategias de SLAM [3] [49] . . . . . 15

5.1. Comparación entre distancias características del mapa construido utilizando un escaner laser y sensores de ultrasonido. . . . . 65
5.2. Comparación entre distancias características de un mapa real utilizando sensor laser. . . . . 69

# Mapeo y Localización Simultánea de un robot móvil en ambientes estructurados basado en integración sensorial

Tesis de Maestría

## Resumen

Julie Stephany Berrío Pérez
**Escuela de Ingeniería Eléctrica y Electrónica**
**Universidad del Valle**

La odometría que inicialmente provee la localización de los robots móviles tiene un alto nivel de incertidumbre, por lo que aumenta el nivel de error, además, este error es acumulativo y por lo tanto no es lo suficientemente confiable para especificar el estado del robot en el medio. Razón que motivó el planteamiento del principal objetivo de la tesis de investigación, la cual busca proponer y programar una metodología que utilice información del medio ambiente (donde se mueve el robot) para restablecer la posición de un robot móvil, realizando una tarea conocida como mapeo y localización simultanea o SLAM (construyendo y/o actualizando un mapa del entorno mientras que el robot se mueve por el mismo).

Como refuerzo a los sensores de odometría se recurre a los de alcance para corregir el estado del robot, en este caso el láser y ultrasonido. Esta corrección se logra por medio de la extracción de características (del medio ambiente), al mismo tiempo que se examina de manera continua como el robot se mueve alrededor de las características detectadas en cada paso. El propósito del algoritmo implementado es mantener y actualizar un mapa estocástico a medida que el robot está elaborando el mapa automáticamente. Cuando se encuentran cambios de odometría correspondientes al desplazamiento del robot, la incertidumbre relativa se actualiza, los puntos de referencia son extraídos del entorno y asociados a observaciones detectadas con anterioridad, luego esta información se utiliza para corregir la posición del robot.

Para el presente trabajo las características extraídas del entorno (aplicando diferentes métodos de segmentación, como Ransac, transformada de Hough, TLS, entre otras) son representadas por líneas rectas que se adquieren de los datos obtenidos a través de un escáner laser y un cinturón de sonares equipados en un robot móvil. La disponibilidad de un modelo estocástico, tanto para el mapa como para las mediciones, permite comprobar cada correspondencia de las características observadas mediante una ventana de validación que define el error máximo admisible entre la observación y la predicción (en este caso una ventana de validación basada en la distancia de Mahalanobis).

Un estimador óptimo (combina de forma recursiva datos con ruido provenientes de sensores con un modelo de la dinámica del sistema) es utilizado para generar la mejor estimación de estados, dada la información disponible, este estimador seleccionado es conocido como el filtro de Kalman extendido (EKF).

Para el análisis de resultados se utilizaron tanto simulaciones como conjuntos de datos obtenidos por medio de un láser y un anillo de ultrasonidos embarcados en un robot móvil PIONNER 3-DX, mediante recorridos realizados en el laboratorio de robótica

móvil de la Universidad de Valle, obteniendo un mapa cuya varianza del vector de estados es reducida en los diferentes pasos del proceso.
El algoritmo desarrollado estima la posición del robot a través de sensores que poseen cierta cantidad de ruido, esto no solo con el ánimo de crear un modelo del ambiente, sino también para abrir camino a nuevas aplicaciones como el monitoreo de zonas donde cámaras o sistemas de posicionamiento global no funcionen correctamente. Para mostrar de manera práctica la potencialidad del estudio del problema de SLAM, se generan archivos que contienen los mapas procesados, que luego son cargados y dibujados automáticamente en un ambiente AutoCad. Brindando una aproximación a un levantamiento de mapas de interiores para determinar la configuración del ambiente y la posición sobre la superficie de elementos o instalaciones, que podría ser utilizado en fases preliminares de proyectos arquitectónicos y de ingeniería en general.

# Capítulo 1

# INTRODUCCIÓN

La navegación autónoma es el conjunto de técnicas y procedimientos que permiten conducir eficientemente y de manera segura un robot a su lugar de destino sin necesidad de intervención directa de terceros (control remoto). El problema de la navegación de un robot móvil se resume en tres preguntas claves: ¿Dónde está el robot?, ¿Cuál es el destino? y ¿Cómo llegar a éste?, a las que se puede adicionar una cuarta pregunta que interviene directamente en la estrategia de navegación ¿Como es el entorno en el que se navega?. La primera y cuarta pregunta es donde se focalizan los mayores esfuerzos de estudio de gran cantidad de grupos de investigación en el mundo. La segunda aún no se ha abordado con suficiente profundidad, ya que para la mayoría de aplicaciones, el destino que debe alcanzar el robot está definido por el usuario del sistema. La tercera pregunta encuentra su respuesta en un grado de madurez de los algoritmos de planificación de trayectorias suficiente para su eficaz implementación y funcionamiento en un robot móvil, así como en diversas técnicas de control reactivo de bajo nivel, adecuadas para la implementación en tiempo real.
La pregunta ¿Dónde está el robot? Ha dado origen desde el principio de la robótica móvil al desarrollo de distintos algoritmos de autolocalización que se basan en un mapa que se introduce previamente al robot, seguidamente a finales de los años 80 comienzan a aparecer los primeros sistemas que intentan construir o actualizar un mapa del entorno mientras que el robot se mueve por el mismo (resolviendo la pregunta ¿Como es el entorno en el que se navega? de forma paralela), donde los algoritmos utilizados desacoplan el problema de la construcción del mapa y de la localización del robot, más la solución rigurosa del problema no es posible sin considerar ambos aspectos simultáneamente. Con la aparición del término SLAM *Simultaneous localization and mapping* (1988), se ha intentado dar una solución óptima a este problema, la cual no se ha conseguido en su totalidad y continua siendo tema de investigación, a pesar de que en los últimos años se han realizado considerables avances en el área. Siguiendo estas tendencias, el presente trabajo de investigación aborda el problema del SLAM con el fin de obtener un mapa del ambiente y la auto-localización de un robot que se encuentra en movimiento en un entorno estructurado.

El presente trabajo está apoyado en el marco científico nacional e internacional buscando realizar transferencia y apropiación de conocimiento que aporte al desarrollo tecnológico en la línea de estudio de la robótica móvil (navegación) de la institución, el cual es un campo de investigación abierto, con múltiples problemas sin solución completa. Los robots móviles son trascendentales para la industria, no solo como ayudantes en procesos de manufactura, sino también como productos comercializables, el conjunto de aplicaciones abarca: el sector manufacturero, pasando por la medicina, el sector militar, hasta el ocio y entretenimiento, el conjunto de aplicaciones abarca: sistemas de producción altamente flexibles, manipulación de explosivos, desactivación de bombas, inspección visual y sensado de zonas con gases tóxicos o bajo peligro de derrumbes, y en general cualquier tipo de tareas donde la presencia de seres humanos implique un riesgo. El presente proyecto se enfoca en el diseño e implementación de un algoritmo capaz de crear mapas del entorno, estimando la posición del robot a través de sensores que poseen cierta cantidad de ruido, esto no solo con el ánimo de crear un modelo del ambiente, sino también para abrir camino a nuevas aplicaciones como el monitoreo de zonas donde cámaras o sistemas de posicionamiento global no funcionen correctamente. Entre otras razones del desarrollo del presente proyecto están:

- Comenzar una nueva línea de investigación en el grupo de Investigación de Percepción y Sistemas Inteligentes (PSI), en el área de robótica, donde todavía no se ha explorado de manera profunda el problema de localizacion y mapeo simultáneo.
- Proyectar a la Universidad del Valle y al Grupo de Investigación PSI a nivel internacional a través de la publicación de los resultados obtenidos en este campo de investigación de robótica móvil.
- Generar conocimientos que permitan el planteamiento de nuevos proyectos complementarios y de mejora de los algortimos implementados.

El objetivo general del proyecto es diseñar un algoritmo de Localización y Mapeo Simultanea (SLAM) para un robot móvil que navega en ambientes estructurados, el cual se logró tras el cumplimiento de objetivos especificos donde se tuvieron como metas: obtener el estado del arte sobre técnicas de localización y mapeo simultánea (SLAM), seleccionar el algoritmo a ser implementado como estrategia de SLAM, proponer y programar un sistema de integración sensorial para la construcción de mapas y localización de un robot móvil y por último desarrollar un protocolo de pruebas que permita evaluar el desempeño del algoritmo.

En esta tesis, se discute el problema de autonomía total para un robot móvil y el desarrollo de un sistema de mapeo y localización simultánea SLAM. Se plantea una descripción del marco teórico resultante de la recolección, sistematización, clasificación y selección de información, donde se explica con mayor profundidad el problema, el proceso y los pasos que se deben realizar para realizar SLAM, además de la formulación matemática y la presentación de soluciones planteadas en la literatura. Seguidamente se muestra el sistema total de desarrollo, presentando las características del sistema

(robot y sensores) en cuanto a descripción física, las formas de representar el ambiente y extraer características del mismo a través del procesamiento de la información proveniente de los sensores, y la asociación de las características para obtener correspondencias entre aquellas que han sido vistas en el pasado con las detectadas en el presente. Se introduce entonces el método probabilístico seleccionado para resolver el problema SLAM, llamado Filtro de Kalman, el cual, para su aplicación en la solución del problema debe ser modificado, a esta variación se le denomina Filtro Extendido de Kalman, el cual se expone de forma genérica, para luego ser aplicado en el mapeo y localización simultaneo de un robot móvil utilizando líneas rectas como características extraídas del entorno. Finalmente se muestran la valoración de los resultados del proceso de SLAM programado, junto con una aplicación en el área de ingeniería que manifiesta el impacto de la investigación, las conclusiones y trabajo futuro del presente trabajo.

# Capítulo 2

# MAPEO Y LOCALIZACIÓN SIMULTÁNEA (SLAM) EN LA ROBÓTICA

En general el problema del SLAM ha sido un importante tema de investigación desde los comienzos de la robótica móvil (finales de siglo XX), y antes de esto, en áreas como sistemas de navegación vehiculares y evaluación geofísica. El SLAM puede ser implementado de muchas maneras (en robots de interiores, exteriores, subacuáticos y aéreos), en primer lugar existe gran cantidad de hardware y software que puede ser utilizado, segundo, SLAM es más un concepto que un único algoritmo. Existen varios pasos implicados en el SLAM y cada uno de ellos se puede efectuar haciendo uso de diferentes algoritmos. El problema del SLAM puede considerarse resuelto en un nivel teórico y conceptual, sin embargo algunos puntos en la práctica se realizan de forma general, lo cual se ve reflejado en el proceso de construcción de mapas (información y tiempo).

Numerosas aproximaciones han sido propuestas para el tratamiento del problema del SLAM y de la simplificación de los problemas de navegación aportando información adicional del mapa o localización del robot. En términos generales, estas aproximaciones adoptan una de las tres principales filosofías. La más popular de éstas es la estimación teórica o filtro de Kalman basado en aproximación, debido a dos factores importantes; primero, provee directamente una solución recursiva al problema de navegación y un medio de cómputo coherente de estimación para la incertidumbre en la ubicación del vehículo y de las referencias en el mapa en base a modelos estadísticos del movimiento del robot y las observaciones relativas de las referencias; segundo, una cantidad sustancial de métodos y experimentos han sido desarrollados en navegación aeroespacial, marítima y otros, desde donde la comunidad de vehículos autónomos puede aproximarse.

Una segunda filosofía se basa en eliminar la necesidad de estimar la posición absoluta para calcular las medidas de la incertidumbre y en cambio utilizar más datos cualitativos de la posición relativa de las marcas y el vehículo, para la construcción del mapa

y la guía de movimiento. Esta filosofía ha sido desarrollada por diferentes grupos en diferentes maneras [8] [31] [32]. La tercera, una amplia filosofía lejana del riguroso filtro de Kalman o formalismos estadísticos mientras conserva esencialmente la aproximación numérica o computacional del problema de navegación y SLAM. Tales aproximaciones incluyen la utilización de la correspondencia de puntos de referencias icónicos [55], registro del mapa global [14], regiones limitadas [50] y otras medidas para describir la incertidumbre [15]. Es de tener en cuenta que el problema del SLAM como tal, no está del todo resuelto, y aún existe una investigación por realizar considerable sobre este campo, por ejemplo el mapeo de zonas de larga extensión.
Varios grupos de investigación en la actualidad trabajan en el problema del SLAM, ya sea utilizando sensores láser, ultrasónicos, de visión o una combinación de estos, aplicando variadas técnicas de procesamiento de las lecturas para el manejo de ruido, limitaciones de rango e incertidumbres presentes en estas, además de la implementación de algoritmos probabilísticos para la corrección de la localización obtenida a través de odómetros, para los cuales el desarrollo se encuentra enfocado en la optimización de recursos que permita un menor coste computacional, logrando de esta manera realizar el SLAM en ambientes cada vez de mayor tamaño. La principal motivación de este trabajo es realizar apropiación del conocimiento en esta área, mediante investigación e implementación de algunas técnicas utilizadas en la resolución del problema del SLAM, abriendo las puertas a investigaciones futuras que profundicen en temas de optimización, desarrollo e implementación en tiempo real de los algoritmos.

## 2.1. Metodología para SLAM

El proceso de SLAM se fundamenta en una sucesión de pasos, donde el objetivo de la metodología planteada es utilizar información del medio ambiente donde se mueve el robot para restablecer su posición. Dado que la odometría del robot (que en principio proporciona la ubicación de los robots) es frecuentemente errónea, no es lo suficientemente confiable para definir la posición del robot en el medio. Como apoyo a los sensores de odometría se recurre a los de alcance para corregir la posición del robot, por ejemplo el láser. Esto se consigue mediante la extracción de características del medio ambiente y observando repetidamente cuando el robot se mueve alrededor.

Para solucionar el problema del SLAM se utiliza un mapa estocástico, que a diferencia de la localización a partir de un mapa a priori, se debe mantener y actualizar a medida que el robot está realizando el mapa automáticamente. La (Fig. 2.1) muestra un esquema que incorpora la construcción del mapa y el mantenimiento del mismo, en el circuito de localización del robot. Cuando existen cambios de odometría debido al movimiento del robot, la incertidumbre relativa se actualiza, las marcas son extraídas del medio ambiente desde la nueva posición, luego se intenta asociar estos puntos de referencia a las observaciones que previamente han sido vistas, estas se utilizan para corregir la posición del robot. En este diagrama, las observaciones estimadas afectan la creación o descarte de nuevas características en el mapa, las cuales se agregan como

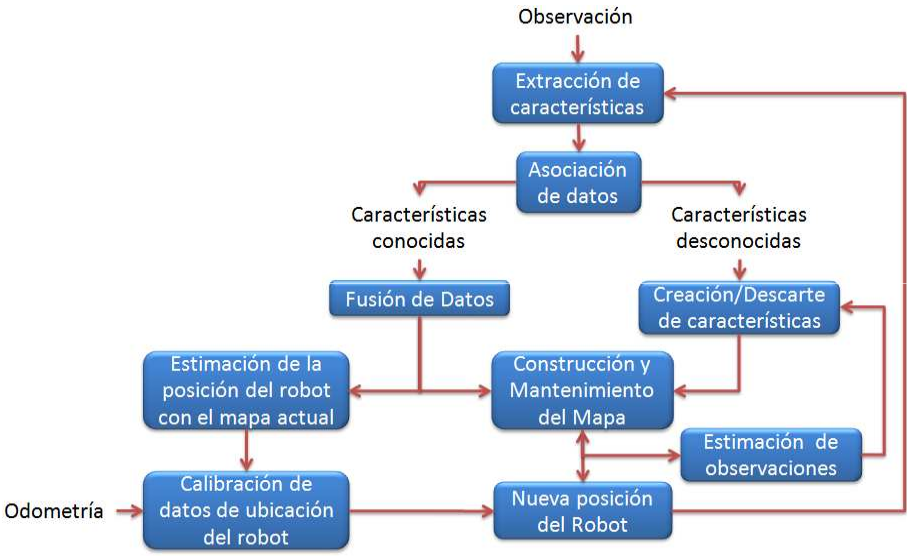


Figura 2.1: Desarrollo del proceso de localización y mapeo simultáneo

nuevas observaciones para que puedan ser asociadas más adelante. Las incertidumbres de todas estas cantidades deben considerarse en todo el proceso.

## 2.2. Formulación del problema

Considerando que un robot móvil se desplaza a través de un ambiente, haciendo observaciones relativas de un número de puntos de referencia usando un sensor, en un instante de tiempo $k$ (Fig. 2.2), las siguientes notaciones son definidas como:

$x_k$: Vector de estado que describe la posición y orientación del robot.
$u_k$: Vector de control, aplicado en el instante $k-1$ para llevar el vehículo al estado $x_k$ en el instante $k$.
$m_i$: Vector que describe la posición del $i$ésimo punto de referencia cuya localización se asume invariante en el tiempo.
$z_{(k,i)}$: Observación tomada desde el vehículo del punto de referencia $i$ésimo en el instante $k$.

Cuando existen múltiples observaciones de puntos de referencias en un mismo instante o cuando estas no son muy relevantes, la observación se denota como $z_k$.

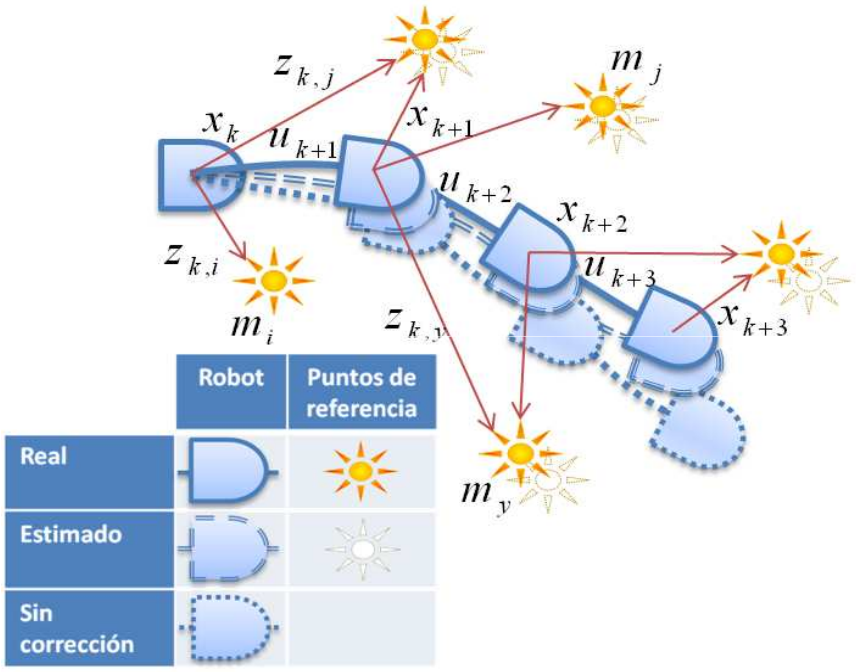


Figura 2.2: Esquema de localización y mapeo simultáneo

La estimación y la extracción de características (puntos de referencia) deben ser realizadas al mismo tiempo, a través de observaciones realizadas entre la verdadera posición del robot (no conocida) y los puntos de referencia pasados. [18] Además, también se definen algunos parámetros como:
Historial de la ubicación del vehículo:

$$X_{0:k} = \{x_0, x_1, \ldots, x_k\} = \{X_{0:k-1} x_k\} \tag{2.1}$$

Historial de entradas de control:

$$U_{0:k} = \{u_1, u_2, \ldots, u_k\} = \{U_{0:k-1} u_k\} \tag{2.2}$$

Todos los puntos de referencia:

$$m = \{m_1, m_2, \ldots, m_n\} \tag{2.3}$$

Todas las observaciones de puntos de referencia:

$$Z_{0:k} = \{z_1, z_2, \ldots, z_k\} = \{Z_{0:k-1} z_k\} \tag{2.4}$$

Para dar solución al problema del SLAM de forma probabilística, se requiere que la distribución de probabilidad sea calculada en todos los instantes $k$, el teorema de

Bayes permite escribirla condicionada al conjunto de todos los datos recogidos hasta el instante $k$, como lo expresa la Eq. (2.5).

$$P(x_k, m|Z_{0:k}, U_{0:k}, x_0) \tag{2.5}$$

Si se considera que el único estado existente es $x_k$,$m$ y $x_0$, (hipótesis de Markov), la Eq. (2.5) puede ser escrita de la siguiente manera:

$$P(x_k, m|Z_{0:k}, U_{0:k}, x_0) = P(z_k|x_k, m)P(x_k, m|Z_{0:k-1}, U_{0:k}, x_0) \tag{2.6}$$

$$= P(z_k|x_k, m) \int P(x_k|x_{k-1}, u_k)P(x_{k-1}, m|Z_{0:k-1}, U_{0:k-1}, x_0)dx_{k-1} \tag{2.7}$$

Se puede expresar entonces:

$$P(x_k, m|Z_{0:k-1}, U_{0:k}, x_0) = \int P(x_k|x_{k-1}, u_k)P(x_{k-1}, m|Z_{0:k-1}, U_{0:k-1}, x_0)dx_{k-1} \tag{2.8}$$

Así, el problema puede ser solucionado de manera recursiva, haciendo uso de datos proporcionados por los sensores (con su probabilidad) en cada instante $k$, y la función de probabilidad del estado del instante $k-1$. Este cálculo demanda que un modelo de transición y un observador del estado (funciones generativas) sean definidos describiendo el efecto de la entrada de control (sistema odométrico) y observación (percepción externa) respectivamente.

Modelo de observación $P(z_k|x_k, m)$
Modelo de movimiento $P(x_k|x_{k-1}, u_k)$

Estas dos funciones son generalmente invariantes en el tiempo por lo que no dependen de la variable $k$, ya que las observaciones están condicionadas a el mapa dado y el estado actual del robot, y el próximo estado del robot depende de su estado pasado y de la entrada de control aplicada al mismo.
Vale la pena señalar que el problema de construcción de mapas pueden ser formulado como el cálculo de la densidad condicional $P(m|X_{0:k}, Z_{0:k}, U_{0:k})$. Esto supone que la ubicación del robot $x_k$ se conoce (o al menos deterministamente) en todo momento, sujeto de conocimiento de la ubicación inicial. El mapa $m$ se construye fusionando las observaciones desde distintas ubicaciones. Por el contrario, el problema de localización se puede formular como el cálculo de la distribución de probabilidad $P(x_k|Z_{0:k}, U_{0:k}, x_0)$. Esto supone que la ubicación de los puntos de referencia se conoce con certeza y el objetivo es calcular una estimación de la localización del vehículo con respecto a estos puntos.
El modelo de observación $P(z_k|x_k, m)$ hace explícita la dependencia de las observaciones tanto en el vehículo como de los puntos de referencia. De ello se deduce que la distribución conjunta no puede ser dividida de la manera obvia y que una división de

este tipo lleva a estimaciones inconsistentes. Gran parte del error entre un punto de referencia estimado y el real es debido a una sola fuente: los errores que se producen en el conocimiento de donde está ubicado el robot cuando las observaciones a los puntos de referencia se realizan. A su vez, esto implica que los errores en las estimaciones de ubicación de los puntos están muy correlacionados. En la práctica, esto significa que la posición relativa entre dos puntos de referencia $m_i$-$m_j$, se puede conocer con gran precisión, aun cuando la ubicación absoluta de un punto de referencia $m_i$ es bastante incierta. En forma probabilística, esto quiere decir que la densidad de probabilidad conjunta para el par de puntos de referencia $P(m_i, m_j)$, es alcanzable incluso cuando la densidad marginal $P(m_i)$ es muy dispersa. [18]

## 2.3. Soluciones al problema de SLAM

En robótica la construcción de mapas se remonta a hace 33 años, y desde la década de 1990 los enfoques probabilísticos (es decir, Filtros de Kalman (KF), Filtros de Partículas (PF) y la maximización de la expectativa (EM)) se han convertido en dominantes en la resolución del problema del SLAM. Las tres técnicas son derivaciones matemáticas de la regla de Bayes recursiva. La principal razón de esta popularidad técnicas probabilísticas es el hecho de que el mapeo del robot se caracteriza por poseer incertidumbre y el ruido en los sensores, y los algoritmos probabilísticos abordan el problema explícitamente modelando las distintas fuentes de ruido y sus efectos sobre las mediciones [49].

### 2.3.1. Filtro de Kalman:

La localización y el proceso de construcción de mapas consisten en generar la mejor estimación para el sistema de estados dada la información disponible al sistema. Esto se puede lograr con una predicción recursiva, en tres etapas del proceso (comprensión de la predicción, la observación y actualización de medidas) conocido como el filtro de Kalman extendido (EKF) [15].El filtro de Kalman es un estimador recursivo de mínimos cuadrados y produce, entre otras cosas, en un tiempo $i$ un mínimo de estimación del error cuadrático medio $\hat{x}(i|j)$ del estado $x(i)$ dada una secuencia de observaciones hasta el tiempo $j$, $Z^j = z(1) \dots z(j)$.

$$\hat{x}(i|j) = E[x(i)|Z^j] \tag{2.9}$$

Para el algoritmo de localización y mapeo simultaneo, el EKF es utilizado para estimar (aposteriori) la posición del vehiculo $\hat{x}_v^+(k)$ a lo largo de las posiciones de $n_f$ características observadas $\hat{x}_v^+(k)$, $i = 1 \dots n_f$. La estimación del estado consta del estado actual del vehículo, sus estimaciones, así como las características relacionadas con otras

ya observadas.

$$\hat{x}^+(k) = \begin{bmatrix} \hat{x}_v^+(k) \\ \hat{x}_v^1(k) \\ . \\ . \\ . \\ \hat{x}_{n_f}^+(k) \end{bmatrix} \tag{2.10}$$

La matriz de covarianza aposteriori para el estado estimado está definida a través de

$$P^+(k) = E[(x(k) - \hat{x}^+(k))(x(k) - \hat{x}^+(k))^T | Z^k] \tag{2.11}$$

Este define el error cuadrático medio y el error de las correlaciones en cada uno de los estados estimados. Para el caso del filtro para SLAM, la matriz de covarianza toma la forma siguiente utilizando $P_{vv}^+(k)$ para representar las covarianzas del vehículo, $P_{mm}^+(k)$ para representar las covarianzas de las características del mapa y $P_{vm}^+(k)$ para representar el covarianza cruzada entre el vehículo y el mapa.

$$P^+(k) = \begin{bmatrix} P_{vv}^+(k) & P_{vm}^+(k) \\ P_{vm}^{+T}(k) & P_{mm}^+(k) \end{bmatrix} \tag{2.12}$$

La etapa de predicción del filtro, utiliza un modelo del movimiento del vehículo, $f(\hat{x}_v^+(k-1), u(k))$ para generar una estimación apriori de la posición del vehículo, $\hat{x}_v^-(k)$, en el instante $k$ dada la información disponible en el instante $k-1$. Los puntos de referencia en general se suponen fijos. Juntos, estos dos modelos generan como resultado la matriz aumentada del estado durante el ciclo de predicción del filtro.

$$\begin{bmatrix} \hat{x}_v^- \\ \hat{x}_m^- \end{bmatrix} = \begin{bmatrix} f(\hat{x}_v^+(k-1), u(k)) \\ \hat{x}_m^+(k-1) \end{bmatrix} \tag{2.13}$$

La matriz de covarianza apriori también debe ser propagada a través del modelo del vehículo como parte de la predicción. El Filtro de Kalman extendido linealiza la propagación de la incertidumbre sobre la estimación del estado actual $\hat{x}^+(k-1)$ usando el jacobiano $\bigtriangledown_x f(k)$ de $f$ evaluada en $\hat{x}^+(k-1)$ como

$$P^-(k) = \bigtriangledown_x f(k) P^+(k-1) \bigtriangledown_x f^T(k) + Q(k) \tag{2.14}$$

Para el algoritmo de SLAM, este paso en el filtro puede ser simplificado debido a la suposición de que las características son estacionarias. Esto permite que la complejidad de calcular el pronóstico de la covarianza disminuya al exigir sólo las variaciones asociadas con el vehículo y que los términos de covarianza cruzada entre el vehículo y el mapa sean actualizados durante la etapa de predicción. La fusión de la observación en la estimación del estado se logra calculando primero una observación apriori predicha, $\hat{z}^-(k)$, utilizando el modelo de observación, $h$ como:

$$\hat{z}^-(k) = h(\hat{x}^-(k)) \tag{2.15}$$

Cuando se reciban las observaciones de los sensores del vehículo, estas deben estar asociadas con características particulares en el ambiente. La diferencia entre la observación real, $z(k)$, recibida de los sensores del sistema y la observación predicha,$\hat{z}^-(k)$, se denomina la innovación $v(k)$.

$$v(k) = z(k) - \hat{z}^-(k) \tag{2.16}$$

La covarianza de la innovación, $S(k)$ se calcula, a partir de la estimación del estado actual de covarianza, $P^-(k)$, el Jacobiano del modelo de observación, $\bigtriangledown_x h(k)$, y la covarianza de la observación del modelo $R(k)$,

$$S(k) = \bigtriangledown_x h(k) P^-(k) \bigtriangledown_x h^T(k) + R(k) \tag{2.17}$$

Las innovaciones y sus covarianzas asociadas pueden ser utilizados para validar las mediciones antes de que sean incorporados en las estimaciones de filtrado. El cálculo de la covarianza de la innovación se puede simplificar al señalar que cada observación es sólo función de la característica observada. Una vez que la observación se ha asociado con una característica particular en el mapa, la estimación de estado se puede actualizar por medio de la matriz de ganancia óptima $W(k)$, Esta matriz de ganancia proporciona una suma ponderada de la predicción y la observación y se calcula utilizando la covarianza de la innovación, $S(k)$ y la covarianza estado predicho $P^-(k)$. El factor de ponderación es proporcional a $P^-(k)$ e inversamente proporcional a la covarianza de la innovación. Esto se utiliza para calcular la actualización de estado $\hat{x}^+(k)$, así como la actualización del estado de covarianza $P^+(k)$.

$$\hat{x}^+(k) = \hat{x}^-(k) + W(k)v(k) \tag{2.18}$$

$$P^+(k) = P^-(k) - W(k)S(k)W^T(k) \tag{2.19}$$

Donde

$$W(k) = P^-(k) \bigtriangledown_x h^T(k) S^{-1}(k) \tag{2.20}$$

Cuando una nueva característica se observa, su estimación debe ser correctamente inicializada y se agregada al vector de estado. Teniendo en cuenta una estimación del estado actual, $\hat{x}^-(k)$, compuesto por el estado del vehículo, $\hat{x}_v^-(k)$, y los estados del mapa, $\hat{x}_m^-(k)$, una observación de relación entre el vehículo y la nueva característica,$z(k)$, y un modelo de función de inicialización, $g_i(.,.)$ se asignan a la estimación actual del estado del vehículo y a la observación a una nueva estimación de característica, la estimación inicial del estado característica es

$$\hat{x}_i^+(k) = g_i(\hat{x}_v^-(k), z(k)) \tag{2.21}$$

Estas estimaciones del nuevo estado se adjuntan al vector de estado como nuevos elementos de características del mapa. Las covarianzas de las nuevas características estimadas deben ser correctamente inicializadas desde la estimación inicial, esta depende de la estimación actual del vehículo, por lo que se correlaciona con el resto del vehículo

y otras estimaciones del estado mapa. Ignorar la correlación entre la estimación del nuevo estado y el resto del mapa, puede dar lugar a incoherencias en el proceso de filtrado. La matriz de covarianza es primero aumentada con la covarianza de la observación, y los términos de la covarianza cruzada entre los elementos del estado actual y las estimaciones del nuevo estado se calculan.

$$P^{*-}(k) = \begin{bmatrix} P^{-}_{vv}(k) & P^{-}_{vm}(k) & 0 \\ P^{-}_{vm}(k) & P^{-}_{mm}(k) & 0 \\ 0 & 0 & R(k) \end{bmatrix} \tag{2.22}$$

La covarianza final se calcula mediante la proyección de la matriz de covarianza aumentada a través del jacobiano $\bigtriangledown_x g(k)$ de la función de inicialización,$g_i$, con respecto a los estados aumentados

$$P^{+}(k) = \bigtriangledown_x g(k) P^{*-}(k) \bigtriangledown_x g^T(k) \tag{2.23}$$

Una inicialización adecuada de las estimaciones de las características, es necesaria para mantener su coherencia y para generar la correcta covarianza cruzada entre la función y las estimaciones del vehículo [3].

### 2.3.2. Métodos basados en filtros de partículas

El filtro de partículas, es un filtro bayesiano recursivo que se implementa en las simulaciones de Monte Carlo. Ejecuta estimación SMC (Sequential Monte-Carlo) por un conjunto de grupos de puntos aleatorios ("partículas"), que representan la probabilidad a posteriori bayesiana. A diferencia de los filtros paramétricos (por ejemplo, KF), PF representa la distribución de un conjunto de muestras tomadas de esta distribución, lo que lo hace capaz de manejar sensores altamente no lineales y el ruido no gaussiano. Sin embargo, esta capacidad produce un aumento de la complejidad computacional sobre la dimensión del Estado como nuevos puntos de referencia se detecten, llegando a ser no apto para aplicaciones en tiempo real. Por esta razón, PF sólo ha sido aplicado con éxito a la localización, es decir, determinar la posición y orientación del robot, pero no para el mapeo, es decir, la posición del punto de referencia y su orientación [52]. Una suposición común en SLAM es que las marcas se encuentran al azar e independientemente distribuidas en el ambiente. Bajo este supuesto, la correlación entre las variables de los puntos de referencias $m$ en la estimación SLAM, se consigue sólo a través de la incertidumbre de la trayectoria del robot (las variables $n_k$ representan la correspondencia entre las observaciones y los puntos de referencia en el mapa). El "filtro de partículas Rao-Blackwellized "(RBPF) es un método para realizar la estimación, donde primero se factoriza de acuerdo con la independencia de los estados:

$$P(x_k, m | Z_{0:k}, U_{0:k}, n_{0:k}) = \underbrace{P(x_k | Z_{0:k}, U_{0:k}, n_{0:k})}_{\text{a través de trayectorias}} \prod_{i=1}^{n} \underbrace{P(m_i | Z_{0:k}, U_{0:k}, n_{0:k})}_{\text{a través de puntos de referencia}} \tag{2.24}$$

El cálculo se estima a través de la trayectoria de forma no-paramétrica con $n$ muestras ("partículas"). Acondicionado en los valores incluidos en la muestra, las variables son puntos de referencia independientes, y cada uno se estima por separado, por lo general con un pequeño filtro de Kalman extendido (EKF) de tamaño constante. La estimación de la trayectoria con las muestras se realiza a través del muestreo de importancia secuencial y re-muestreo (SISR), comúnmente conocido como filtrado de re-muestreo o filtrado de partículas. (Filtrado de partículas es un método general para el muestreo de alta dimensión, la distribución de muestras se complica por la construcción de forma secuencial). La idea básica es utilizar para cada partícula $\phi_k^i = X_{0:k-1}^i, m^i$ un modelo de movimiento $P(x_k|u_k, x_{k-1})$ como una distribución propuesta para proyectar el estado del robot por muestreo, es decir:

$$x_k^i \sim P(x_k|x_{k-1}^i, u_k) \tag{2.25}$$

Una vez que cada partícula se proyecta, el paso de "predicción"se completa y la probabilidad a posteriori está representada por las partículas.

$$P(X_{0:k}|u_k, z_{0:k}, n_{0:k-1}) \tag{2.26}$$

A continuación, las muestras se ponderan según la probabilidad de medición del sensor, es decir:

$$w_k^i = w_{k-1}^i P(z_k|x_k^i, n_(k), m_{n_k}^i) \tag{2.27}$$

Las muestras son ponderadas (asintóticamente) desde la probabilidad a posteriori deseada. Por último, se realiza un paso de re-muestreo N veces a partir del conjunto de partículas, con reemplazo, de acuerdo a los pesos $w_k^i$. Los pesos se restablecen de manera uniforme. Esto suele mejorar la representación de la trayectoria a posteriori, porque con el tiempo, la mayor parte del peso se concentra en tan sólo unas pocas partículas. El re-muestreo asigna más partículas a las zonas de alta probabilidad, lo que conduce a una mejor estimación de trayectoria futura [4].

### 2.3.3. Métodos basados en maximización de la expectativa

La estimación de EM (expectation maximization) es un algoritmo estadístico que se desarrolló en el contexto de máxima similitud (ML máximum likelihood) de estimación y ofrece una solución óptima, siendo una opción ideal para la construcción del mapa, pero no para la localización. El algoritmo EM es capaz de construir un mapa cuando la posición del robot es conocida, por ejemplo, por medio de la expectativa. EM itera dos pasos: un paso de expectativa (paso E-), donde la probabilidad a posteriori de la localización del robot se calcula para un mapa dado, y el paso de maximización (M-paso), en la que el mapa más probable es calculado teniendo en cuenta las expectativas de localización. El resultado final es una serie de mapas cada vez más precisos [49]. Formalmente, la función que está siendo maximizada es la expectativa sobre la probabilidad de los datos de $d_k$ y la ruta del robot $X_{0:k}$

$$m_{n+1} = \arg\max E_{s^t}[logP(d_k, x_k|m)|m_{0:n}, d_k] \tag{2.28}$$

Esta ecuación se deriva del filtro de Bayes bajo varios supuestos. El mapa (i+1)ésimo se obtiene del mapa i por la maximización de una función de probabilidad. El logaritmo es una función monótonica, por lo que maximizar el logaritmo es equivalente a la maximización de la probabilidad. Parte de la probabilidad que es maximizada corresponde a la ruta del robot. Sin embargo, en el mapeo la ruta es desconocida. La Eq. (2.28), por lo tanto, calcula la expectativa de esta posibilidad en todos los caminos posibles que el robot pueda haber tenido. Bajo los supuestos, la Eq. (2.28) puede ser re-expresada en la siguiente integral:

$$m_{n+1} = \arg\max \sum_{\tau} \int P(x_\tau | m_{0:n}, d_k) log P(z_\tau, x_\tau | m) d_{s_\tau} \tag{2.29}$$

El lado derecho contiene el término $P(x_\tau | m_{0:n} d_(k))$, que es la probabilidad a posteriori de la posición $x_\tau$ condicionada a los datos $d_k$ y al mapa $i$ de $m_{0:n}$, el problema específico es de localización del robot, ya que se da el mapa $i$-ésimo y sólo se necesita calcular la probabilidad a posteriori de la posición del robot. La diferencia clave para establecer la localización es que en nuestro caso, los datos en todo el intervalo de tiempo $\{0, \ldots, k\}$ se utilizan para estimar la probabilidad a posteriori de la ubicación en el tiempo $\tau$, incluso para $\tau < k$. Por lo tanto, hay que incorporar datos pasados y futuros en relación con el paso del tiempo $\tau$. Por suerte, bajo supuestos esto puede lograrse mediante la doble ejecución de filtros de Bayes: una en el tiempo, y otra hacia atrás en el tiempo. El paso hacia adelante nos da una probabilidad a posteriori $P(x_\tau | m_{0:n}, d_\tau)$ condicionada a todos los datos previos al tiempo $\tau$. El paso hacia atrás nos da probabilidad a $P(x_\tau m_{0:n}, d_{\tau+1}, \ldots, d_k)$ condicionada a todos los datos obtenidos después del tiempo $\tau$. La multiplicación de estas dos estimaciones y la normalización posterior nos da la probabilidad deseada $P(x_\tau | m_{0:n}, d_k)$. Este cálculo es conocido como el paso E en EM, ya que calcula las expectativas (probabilidades) de diversas posiciones en todos los momentos en el tiempo. El paso final, el paso M, es la maximización de la Eq. (2.29). Aquí las expectativas $P(x_\tau | m_{0:n} d_k)$ son fijas, como los obtenidos en el paso E. El objetivo del paso M es encontrar un nuevo mapa $m$ que maximice la probabilidad de registro de las mediciones del sensor $\log P(z_\tau | x_\tau, m)$, para todo $\tau$ y todas las posiciones $x_\tau$ y bajo la expectativa calculada en el paso E. Desafortunadamente, no se conoce una solución de forma cerrada a este problema de maximización de alta dimensión. Un enfoque común es resolver el problema para cada localización en el mapa $\langle x, y \rangle$, de forma independiente. Esto supone que el mapa está representado por un número finito de lugares. La maximización del componente racional es entonces relativamente sencillo. Las implementaciones existentes del paradigma de la EM se basan en representaciones de red para todas las densidades involucradas en el proceso de estimación. También dependen de mapas probabilísticos en vez de ausencia o único mapa, que tiene un efecto suavizante en la maximización y evita quedar atrapado en máximos locales durante la optimización [49].

| Método | Pros | Contras | Otros aspectos |
|---|---|---|---|
| **KF/EKF** | Alta Convergencia, Maneja incertidumbre | Supuesto Gussiano Lento en mapas de varias dimensiones | Incremental, No maneja correspondencias, Puede caer en mínimos locales |
| **PF** | Maneja no linealidades, Maneja ruido no gaussiano | Crece en complejidad, Solo tiene éxito en localización | Sufre degeneración debido a la dependencia de datos pasados |
| **EM** | Óptimo para la construcción de mapas, Resuelve asociación de datos | Ineficiente por el crecimiento del costo, Inestable para grandes escenarios Solo tiene éxito en construcción de mapas | Maneja mínimos locales, No apto para ambientes dinámicos |

Tabla 2.1: Lista de ventajas y desventajas de las diferentes estrategias de SLAM [3] [49]

### 2.3.4. Comparación (pros y contras) de métodos de solución del SLAM

Casi ninguno de los métodos actuales pueden realizar mapas compatibles para las áreas grandes, debido principalmente al aumento en el costo computacional y de las incertidumbres. Por lo tanto este es posiblemente el tema más importante que necesita ser mejorado. Algunas publicaciones recientes abordan el problema mediante el uso de múltiples mapas, o sub-mapas que se utilizan últimamente para construir un mapa global más grande [20] [39] [46] [3]. Sin embargo, estos métodos se basan mucho en asumir la asociación de datos propiamente dicha, que es otra cuestión importante que necesita ser mejorada. La Tabla 1 proporciona una lista de ventajas y desventajas de las diferentes estrategias de SLAM en términos del método utilizado para hacer frente a las incertidumbres. En esencia, los métodos que todavía no están resueltos son los que permiten implementaciones a gran escala en cada ambientes no estructurados, es decir, bajo el agua, y sobre todo en situaciones en las que otras soluciones actuales no están disponibles o no fiables. De acuerdo con la investigación bibliográfica, las soluciones de SLAM se pueden mejorar ya sea mediante la formulación más eficiente y coherente de los algoritmos de filtrado de grandes escenarios, y resolver de una manera muy fuerte el problema de asociación de datos. Para el primer caso, los diferentes filtros que se aplican en el marco SLAM deben ser estudiados, por ejemplo, el Filtro de Kalman extendido comprimido (CEKF) o filtros de información. El segundo problema es actualmente resuelto usando SIFT y SURF, que parecen ser soluciones considerablemente buenas para el problema de asociación de datos, sin embargo se convierten en costosas computacionalmente cuando se manejan mapas de alta dimensionalidad [3].

## 2.4. Implementación

SLAM, en su forma sencilla, escala cuadráticamente con el número de puntos de referencia en un mapa. Para la aplicación en tiempo real, esta escala es potencialmente una limitación sustancial en el uso de métodos de SLAM. Un segundo gran obstáculo que superar en la aplicación de los métodos de SLAM es la correcta asociación de las observaciones de puntos de referencia con puntos ubicados en el mapa (y su respectiva extracción), una asociación incorrecta puede llevar a una falla catastrófica del algoritmo de SLAM. La asociación de datos es particularmente importante cuando un vehículo vuelve a una región previamente mapeada después de una larga excursión, llamado el problema de bucle cerrado. Los métodos de asociación incluyen la validación de procesos por lotes que explotan las limitaciones inherentes a la formulación de SLAM, los métodos basados en la apariencia, y técnicas multi-hipótesis [18].

## 2.5. Complejidad computacional

La incertidumbre en las estimaciones de los puntos de referencia disminuye monotonicamente, estas incertidumbres convergen a cero y la incertidumbre en el vehículo y ubicaciones absolutas en el mapa alcanzan una cota inferior. La correlación cruzada en la matriz de covarianza del mapa mantienen el conocimiento de las relaciones relativas entre las estimaciones y los puntos de referencia en que se apoyan las propiedades de convergencia. Así la propagación de la matriz de covarianza del mapa completo es esencial para la solución del problema de SLAM. Sin embargo, el uso de la matriz de covarianza del mapa completo en cada paso en el problema de construcción de mapas, causa importantes problemas de cálculo como el aumento del número de puntos de referencia $n$, el cálculo necesario en cada paso se incrementa como $n^3$ y el almacenamiento requerido para el mapa como $n^2$. A medida que el rango sobre el cual se desea operar un algoritmo de SLAM aumenta (y por lo tanto el número de puntos de referencia aumenta) llega a ser esencial desarrollar una versión computacionalmente tratable del algoritmo de construcción de mapa de SLAM que mantenga las propiedades de ser coherente y no divergente [15].

### 2.5.1. Mejoramiento de la eficiencia computacional

La formulación basada en estados del problema de SLAM involucra la estimación de un Estado conjunto compuesto por la posición del robot y las ubicaciones de los puntos de referencia observados. Esta formulación del problema tiene una estructura peculiar, el modelo de proceso sólo afecta a los estados de posición del vehículo y el modelo de observación sólo hace referencia a un solo par vehículo - punto de referencia. Una amplia gama de técnicas se han desarrollado para aprovechar esta estructura especial en la limitación de la complejidad computacional del algoritmo de SLAM. Técnicas para mejorar la eficiencia computacional pueden caracterizarse por ser óptimas o conservadoras.

En los algoritmos óptimos el objetivo es reducir cálculos que se necesitan al resolver las estimaciones y covarianzas. Los algoritmos conservadores dan lugar a estimaciones que tienen una mayor incertidumbre o covarianza que el resultado óptimo. Por lo general, los algoritmos conservadores, si bien son menos precisos, son computacionalmente más eficientes y, por tanto, de valor en las implementaciones reales.
Algoritmos con las incertidumbres o covarianzas menores que los de la solución óptima se denominan y se consideran incompatibles y soluciones no válidas al problema de SLAM. El enfoque directo a la reducción de la complejidad computacional implica la explotación de la estructura del problema de SLAM en la reformulación del tiempo y las ecuaciones de actualización de observación para limitar los cálculos necesarios. El cómputo de tiempo de actualización puede ser limitado usando métodos de aumento de estado El cálculo de la actualización de la observación puede limitarse mediante particiones de las ecuaciones de actualización. Estos pasos resultan en una estimación óptima del SLAM con cálculo reducido. La reformulación de la representación estándar de SLAM espacio-espacio en forma de información permite dividir la información resultante de la matriz, para ser explotados en la reducción de la computación. Los algoritmos resultantes son generalmente conservadores, pero todavía producen una buena estimación con un esfuerzo de cómputo mucho menor. Los métodos de sub mapeo explotan la idea de que un mapa puede ser dividido en regiones con sistemas de coordenadas locales y dispuestas en forma jerárquica. Las actualizaciones pueden ocurrir en un marco local con actualizaciones periódicas entre estos. Las técnicas de sub mapeo generalmente ofrecen una estimación conservadora en el contexto global [18].

## 2.6. Representación del ambiente

Una amplia variedad de técnicas de localización y mapeo están basadas en representaciones del medio, que se encuentra compuesto por un grupo de elementos característicos detectables por el sistema sensorial del robot (mapas basados en características) [22]. Un escenario típico es el de un robot midiendo su ubicación relativa y/o desviación con respecto a los puntos de referencia puntuales. En este escenario, los algoritmos de localización con puntos de referencia conocidos y las técnicas de SLAM se han desarrollado para la descripción estadística de la incertidumbre del sensor y en marcos de lecturas limitadas.
Para la representación de puntos de referencia se hace uso de los datos recolectados por el sensor, algunos algoritmos los representan como puntos en el ambiente, otros realizan una descripción de la forma del grupo de lecturas (características geométricas), este último es de los más utilizados por ser compacto (menor capacidad de almacenamiento) sin una pérdida significativa de información, en los que se incluye la extracción de líneas, curvaturas, esquinas, entre otras. Con el mejoramiento de la capacidad computacional de los procesadores, ha hecho que se apliquen técnicas de segmentación y extracción de características mediante el procesamiento de imágenes del medio.
Entre muchas figuras geométricas, el segmento de línea es la más simple. Es fácil de des-

cribir la mayoría de los entornos semi y estructurados con segmentos de línea. Muchos algoritmos se han propuesto utilizar las características de la línea extraídos de datos en 2D, en [12] proponen un método de segmentación de la línea inspirada en un algoritmo de visión por ordenador, para usar con un mapa a priori como una aproximación a la localización del robot, [54] introduce un algoritmo de construcción concurrente de mapas dinámicos basados en las características geométricas (líneas y círculos) utilizando un escáner láser, [2] utiliza un escáner 2D con un método de segmentación basado en regresión lineal en la localización basada en mapas, [26] presenta una técnica para la adquisición y el seguimiento de la posición de un robot móvil con un escáner láser mediante la extracción de líneas ortogonales (paredes) en un entorno de oficina. Por último, [41] sugiere un algoritmo de extracción en línea con conexión de la línea ponderado para la construcción de mapas basado en líneas [38].
En las lecturas de los sensores externos hay dos tipos de características que son importantes poder detectar en las lecturas (si se quiere comprimir datos) [2]: los puntos de rotura y los segmentos. Los primeros revelan interrupciones en el proceso de la captura de datos, y se originan habitualmente debido a la presencia de cuerpos o áreas que dificultan la localización de otros elementos más distantes. El descubrimiento de puntos de rotura permite catalogar las lecturas en conjuntos nombrados clústeres. Por otra parte, los fragmentos o segmentos están constituidos por conjuntos de puntos próximos que establecen entre sí una forma definida, por ejemplo una línea recta.

### 2.6.1. Clustering

Dado que existen numerosos problemas en los que el análisis de grupos es necesario, han surgido diversas soluciones que se adecuan a cada una de las necesidades específicas de cada problema. Todos estos algoritmos se pueden clasificar en base a una serie de criterios, dependiendo de cuál sea el resultado final, de cómo se escojan los parámetros de entrada, de la estructura del algoritmo, etc. A pesar de la gran cantidad de técnicas de agrupamiento existentes en la literatura, todas pueden ser clasificadas en uno de los siguientes cuatro tipos de agrupamiento:

•**Algoritmos de agrupamiento particionales.** Son aquellos que obtienen como resultado una única partición de los datos iniciales, en lugar de una estructura de agrupamiento con varios niveles de particiones.

•**Algoritmos de agrupamiento jerárquicos.** Organizan los datos en estructuras jerárquicas de acuerdo a la matriz de proximidades. Los resultados de estos algoritmos son, por lo general, mostrados en un árbol binario o en un dendograma.

•**Algoritmos de agrupamiento probabilísticos.** Desde el punto de vista probabilístico, se asume que los objetos son generados de acuerdo a algunas distribuciones probabilísticas. Objetos en distintos grupos son generados por distintas distribuciones de probabilidad o son derivados de distintos tipos de funciones de densidad, o de las

mismas familias pero con distintos parámetros.

•**Algoritmos de agrupamiento basados en densidades.** Estos algoritmos aplican criterios locales de grupo. Los grupos son tenidos en cuenta como regiones en el espacio de datos de gran densidad de objetos, y los cuales están separados por regiones de menor densidad (ruido). Estas regiones pueden tener cualquier forma y pueden estar distribuidas de cualquier manera.

Por otro lado, debido a que el agrupamiento de datos tiene una aplicabilidad muy extensa y variada, y por tanto resuelve problemas de muy distinta índole, se han de tener en cuenta una serie de temas para la aplicación de estas técnicas. Según [25] la clasificación de todos estos temas se resume en:

•**Jerárquicos o particionales.** Este criterio está relacionado con la estructura del resultado final. Los métodos jerárquicos producen una serie de particiones anidadas mientras que los particionales producen una única partición. La clasificación tradicional de los algoritmos de agrupamiento se basa en este criterio.

•**Aglomerativos o divisivos.** Este aspecto está relacionado con la estructura algorítmica y la operativa. Un método aglomerativo comienza con cada dato en un grupo distinto, y va sucesivamente mezclando o uniendo grupos hasta que algún criterio de parada se cumpla. Los métodos divisivos, por el contrario, comienzan introduciendo todos los datos en un grupo y van dividiendo sucesivamente este grupo hasta que se cumple con un criterio de parada.

•**Monotéicos o politéicos.** Esta clasificación se refiere al uso secuencial o simultáneo de las características en el proceso de agrupamiento. La mayoría de los algoritmos son politéicos, es decir, que la mayoría de las características se tienen en cuenta a la hora de calcular las distancias entre los datos, y las decisiones son llevadas a cabo en base a estas distancias. En cambio, un algoritmo monotéico considera las características secuencialmente para dividir la colección de datos dada.

•**Duros o difusos (fuzzy).** Un algoritmo de agrupamiento duro asigna cada objeto a un único grupo durante su operación y a la salida. Por el contrario, un algoritmo de agrupamiento difuso puede asignar a cada objeto o dato a más de un grupo con un cierto grado de pertenencia. Un algoritmo de agrupamiento difuso se puede convertir en uno duro, si asignamos a cada objeto el cluster con mayor grado de pertenencia.

•**Deterministas o estocásticos.** Esta cuestión es más relevante en métodos particionales diseñados para optimizar una función de error cuadrática. Esta optimización puede llevarse a cabo usando técnicas tradicionales o a través de búsquedas aleatorias en el espacio de todas las posibles etiquetas actuales. Incrementales o no incrementales. Este aspecto se debe tener en cuenta cuando el espacio de datos a ser agrupado es

muy grande, y existen limitaciones de memoria o de tiempo de ejecución. Entonces, la arquitectura del algoritmo, puede verse afectada. [10]

### 2.6.2. Extracción de líneas

Una de las ventajas de usar descripciones geométricas en lugar de puntos particulares es que se pueden minimizar las necesidades de almacenamiento [1] [9]. Así, para almacenar la descripción geométrica de un segmento únicamente hacen falta los puntos correspondientes a los extremos y los parámetros de la ecuación de la recta correspondiente. Por lo tanto, utilizar descripciones geométricas permitiría conservar la escalabilidad aun cuando el ambiente tratado sea de superficies extensas. En medios interiores, como universidades, oficinas, o naves industriales, es común poder describir el entorno mediante líneas, dado que los objetos más comunes son paredes, armarios o mesas. La gran mayoría de los métodos utilizados hoy en día para la detección de rectas son adaptaciones de algoritmos consolidados procedentes del campo de visión artificial [11].
En estadística, un *outlier* significa un único dato que es numéricamente distante del resto de las observaciones. Por el contrario, el término *inlier* se utiliza para referirse a las observaciones que no se caracterizan por estar desalineadas. La tasa de ruptura (punto de ruptura) se define como el porcentaje de los valores *outliers* que un determinado método de cálculo puede tolerar antes de que la estimación pueda ser considerada como errónea. Para una tasa de ruptura de 0 porciento sólo una medida diferente hace que el resultado no sea como el esperado. Por el contrario, para una tasa de ruptura de 50 porciento, que es el máximo que se puede tolerar, no sería posible distinguir los valores *outliers*. Los métodos de estimación que son capaces de tolerar un cierto porcentaje de los valores *outliers*, es decir, tienen una mayor tasa de ruptura del 0 porciento se llaman robustos. Por otra parte se sabe que los estimadores no robustos a los que hacen frente a los valores *outliers* proporcionarían una estimación poco o nada precisa [11].

#### 2.6.2.1. Algoritmos no robustos

En un algoritmo no robusto basta que entre los datos se encuentre un único punto *outlier* para que la recta estimada sea completamente errónea, usualmente estos puntos que hacen que la recta estimada se desvíe por completo, se les conoce como puntos palanca (*leverage points*) o puntos envenenados (*poisoned points*). A continuación se presentarán algunos algoritmos que, a pesar de no ser robustos, son de bastante populares a pesar de tener la particularidad de que son generalmente muy sensibles a cambios en sus parámetros.

• ***Successive Edge Following (SEF)***. Este algoritmo ("Seguimiento de Extremos Consecutivos") [6], [47] considera que un segmento termina cuando la diferencia entre la distancia entre puntos y su siguiente distancia excede un determinado valor

umbral fijado con antelación, como lo que se pretende es conseguir un conjunto de líneas, para ello es necesario realizar una etapa de post-procesamiento, para ajustar una línea a cada conjunto mediante algún método de regresión [47]. Como ventaja principal de este procedimiento destaca su rapidez, mientras que algunos de sus inconvenientes son la dependencia de un umbral no variable y su correspondiente dificultad de ajuste.

• ***Line Tracking (LT)***. El método se basa en calcular la ecuación de la recta ajustada sobre los últimos $n-1$ puntos [34] [38]. A continuación se determina la distancia del punto actual a la recta. Si dicha distancia supera un umbral fijado, entonces se comienza una nueva línea. En caso contrario, se recalculan los parámetros de la recta que mejor se ajuste a los últimos $n$ puntos.

• ***End Point Fit (IEPF)***. [7] [47] El Algoritmo define una línea tomando los puntos inicial y final de las lecturas. A continuación se busca el punto más alejado con respecto a dicha línea. Si el punto encontrado está a una distancia mayor que un determinado valor umbral, entonces se divide el scan en dos intervalos. Después de esto el algoritmo comienza recursivamente con cada uno de los segmentos restantes. El procedimiento termina una vez que la distancia al punto más alejado sea menor que el umbral fijado, siempre y cuando esta circunstancia se cumpla para todos y cada uno de los segmentos en los que ha sido dividido la lectura.

• ***Split and Merge***. [19], [28], [35], [45] El algoritmo está compuesto principalmente por dos partes. La primera fase es recursiva y consiste en dividir (*split*) los segmentos disponibles en otros más pequeños, mientras que la segunda sirve para fusionar (*merge*) los segmentos que sean prácticamente colineales. La primera fase es casi idéntica al algoritmo IEPF, salvo por una pequeña diferencia. En el IEPF las líneas se construyen simplemente conectando el primer y último punto de un conjunto de puntos. A diferencia de éste, en el Split and Merge las líneas se construyen ajustándose a un conjunto de puntos.

#### 2.6.2.2. Algoritmos robustos

Existen una serie de algoritmos que, al contrario que los del apartado anterior, son capaces de ofrecer una estimación correcta aunque entre los datos aparezcan puntos *outliers*. Es decir, tienen la particularidad de ser robustos ante fallos o datos que no tienen similitud con el resto. Dentro de esta categoría, se analizarán a continuación el método de mínimos cuadrados, el algoritmo ransac, y la Transformada de Hough.

• **Mínimos cuadrados**. El método de estimación del estado de mínima mediana de cuadrados se basa en el método desarrollado para modelos no lineales y utilizando técnicas de matrices dispersas de manera que el tiempo usado en el procesamiento numérico sea lo más reducido posible. En el método de mínimos cuadrados la función objetivo a minimizar, a partir de la cual se obtienen los coeficientes de la regresión, es

el sumatorio de los residuales al cuadrado.

•**Ransac**. El objetivo general del método consiste en ajustar un modelo matemático para un conjunto de datos experimentales. El ajuste también es robusto, ya que reconoce la presencia de valores desalineados en los datos [41]. Se inicia con un conjunto de puntos, y se ajusta a una línea de dos puntos seleccionados al azar. Después se establece un conjunto de consenso, que forman parte de los puntos que se pueden considerar que están cerca de la línea. Si el número de puntos del conjunto de consenso (C) es mayor que un cierto umbral T, a continuación, se vuelve a reajustar la línea para el conjunto de puntos de consenso (con los mínimos cuadrados, por ejemplo). De lo contrario el algoritmo se repite hasta que se alcanza un número máximo de iteraciones establecido de antemano.

•**Transformada de Hough**. Del campo de procesamiento digital de imágenes procede la Transformada de Hough, una de las técnicas más populares de extracción de características. La transformada original únicamente es capaz de extraer líneas en una imagen dada, aunque se puede generalizar con el fin de identificar formas más complejas, como curvas, por ejemplo. El objetivo de la Transformada de Hough es detectar un conjunto de píxeles que formen una línea recta. Los píxeles se obtienen de una imagen binaria en la que el color negro identifica obstáculos y el color blanco espacios vacíos, o viceversa. Por lo tanto, se parte de un conjunto de puntos que representan los obstáculos en la imagen, y se desea encontrar el conjunto de líneas que se ajustan a $P$. Para ello hay que transformar cada punto de $P$, ubicado en el plano cartesiano $XY$, en una curva sinusoidal en el plano $P$. La peculiaridad de esta transformación está en que si los puntos analizados en el plano $XY$ forman una recta, entonces las curvas sinusoidales generadas por la transformación se cortarán en un único punto. Esto tiene mucho sentido, pues si en el espacio de transformación $P$ se toma un único punto fijo y se sustituye, entonces resultará que hay infinitas combinaciones de valores $(x, y)$, con la característica de que estarán situados sobre una misma recta. [11]

## 2.7. Asociación de datos

La asociación de datos ha sido siempre un tema crítico para las implementaciones prácticas del SLAM. Antes de la fusión de datos en el mapa, las nuevas mediciones se asocian con puntos de referencia de mapas existentes, y, después de la fusión, estas asociaciones no se pueden revisar. El problema es que una sola asociación de datos incorrectos puede provocar divergencias en la estimación del mapa, a menudo causando una falla calamitosa del algoritmo de localización. El problema de la asociación de datos es el de la coincidencia de puntos de referencia observados desde diferentes (láser) escaneos. También se le refiere a la re-observación de puntos de referencia. En la práctica los siguientes problemas pueden surgir en la asociación de datos:

·Es posible que no se vuelva a observar todos los puntos de referencia de paso en el tiempo.

·Es posible que se observe algo como un punto de referencia, pero no siempre lo vemos de nuevo.

·Es posible que erróneamente se asocie una marca a un punto de referencia anteriormente visto.

Como se indicaba anteriormente los puntos de referencia deben ser fáciles para volver a observar. Como tal los dos primeros casos mencionados no son aceptables para un punto de referencia. En otras palabras, son catalogados puntos de referencia incorrectos. Incluso si se tiene un algoritmo de extracción de características muy bueno, es posible que se necesite definir una política adecuada de asociación de datos para minimizar este inconveniente. [43] El último problema en el que erróneamente es asociado un punto de referencia puede ser devastador, ya que significa que el robot se cree que está en algún lugar diferente de donde lo que realmente está. A continuación se describirá algunos métodos utilizados para la realización de la asociación de datos:

• **Distancia de Mahalanobis**: analiza cómo las medidas pueden adaptarse para predecir las características geométricas usando una prueba de validación estadística. Consideremos el caso en que $n$ características geométricas están siendo seguidas y $n$ mediciones se encuentran en el marco de la imagen siguiente. En principio, cualquier vector de medición puede tener su origen en alguna de las características geométricas y hay $n^2$ posibles combinaciones de asignación. En la práctica, algunas mediciones son más probables que se originen de un rastreo que de otro. Una medida de la distancia por lo tanto es necesaria para cuantificar la similitud, entre menor distancia exista entre una medida y su valor estimado, más probable es que se haya originado en dicho valor.

• **Vecino más cercano**: es el algoritmo más simple y sub óptimo de asociación de datos. Esto supone que cada medida se origina en la característica correspondiente más cercana, donde el más cercano se suele definir con la distancia de Mahalanobis. Lo atrayente de este algoritmo es su simplicidad, tanto conceptual como de cómputo. Para las correspondencias del vecino más cercano, siempre existe la posibilidad finita de que la asociación sea incorrecta y esto puede conducir a graves consecuencias. El algoritmo del vecino más próximo realiza las decisiones de asignación basado únicamente en el marco de la imagen actual. Sin embargo, mucha más información está disponible mediante la revisión las imágenes siguientes. Mejoras significativas en las correspondencias pueden lograrse posponiendo el proceso de decisión con la esperanza de que las medidas futuras clarifiquen las ambigüedades actuales.

• **Filtro *Track-Splittin***: Las características geométricas de interés son seguidas,

cuando en un instante, más de una medida se encuentra dentro de su región de validación, en lugar de asignar arbitrariamente la medición más cercana a la característica, se forma un árbol de rastreo, las ramas indican las asignaciones alternativas de las mediciones de las características. No hay decisión sobre la asignación en esta etapa. En cambio, las decisiones se posponen hasta que las medidas adicionales se han reunido para apoyar o refutar las asignaciones anteriores. El supuesto implícito es que las ambigüedades se resuelven con medidas de futuro.

•**Verosimilitud conjunta (*joint likelihood*)**: El método produce particiones disjuntas de medición a fin que la medición se asigne a una sola característica geométrica. La idea básica es, primero, agrupar las medidas en pistas factibles. Este conjunto de pistas no son necesariamente disjuntos, y por lo tanto un subconjunto de las pistas disjuntas debe ser seleccionadas. Sin embargo, hay muchos conjuntos lógicos, y así que se realiza una búsqueda para encontrar el mejor conjunto de pistas inconexas. Una medida de verosimilitud conjunta se utiliza para cuantificar cual conjunto de pistas es el mejor. Las mediciones se clasifican ya sea como originarios de las características geométricas en el entorno o bien se consideran espurios debido al ruido. La primera etapa del algoritmo es la construcción de pistas de viabilidad, en el que las secuencias de las mediciones se agrupan en las pistas factibles que sean razonables para incorporar en una hipótesis. La viabilidad de pista se puede probar en la misma forma que el algoritmo *track-splitting*. A continuación, una hipótesis debe seleccionar un conjunto de pistas inconexas del conjunto factible, y la probabilidad de la hipótesis debe ser calculada. Así como el filtro *track-splitting*, se calcula la probabilidad de una pista, estimando la probabilidad conjunta de una partición. Por último, la restricción de que todas las pistas en una partición lógicos deben ser disjuntos debe ser impuesta. Diferentes conjuntos de partición posible, pueden tener un número diferente de particiones de pistas, pero para cualquier conjunto posible de las particiones todas las mediciones deben ser tenidas en cuenta. La principal desventaja del enfoque Verosimilitud conjunta es que es un proceso por partes. Una formulación recursiva en línea es posible, pero el rendimiento de los algoritmos puestos en línea puede ser significativamente peor que las técnicas de proceso por lotes. Un segundo problema es que la iniciación y terminación de las características geométricas no está explícitamente manejada por el algoritmo, pero se lleva a cabo en lugar de la etapa de factibilidad pista.

•**Algoritmo de hipótesis múltiple**: Por desgracia, el *track-splitting*, la verosimilitud conjunta *joint likelihood*, y algoritmos de varias hipótesis tienen complejidad exponencial, y, mientras que la heurística puede ser usada para restringir el espacio de búsqueda, grandes cantidades de memoria y recursos de cómputo puede ser necesaria. Para algunas aplicaciones, estos requisitos no pueden cumplirse, por tiempo real y / o restricciones de costo. Aproximaciones subóptimas pueden desarrollarse porque tienen la ventaja de tener memoria finita y cálculos, como el algoritmo del filtro de asociación de datos por probabilidad conjunta [13].

# Capítulo 3

# SISTEMA DEL EKF-SLAM

En este capítulo se estudiarán las características físicas (robot y sensores) de la plataforma de donde se obtuvieron los datos para la realización de pruebas y evaluaciones. Esta descripción ayudará a entender las capacidades del sistema y cuáles son sus limitaciones. Adicionalmente se explica el procesamiento al que son sometidos los datos provenientes de los sensores de proximidad para la extracción de las características y su posterior asociación.

El sistema de desarrollo para la programación y prueba del algoritmo de SLAM mostrado en la Fig. (3.1) se compone de una plataforma de desarrollo que incluye, un robot móvil dotado de un sensor propioceptivo denominado odómetro y exteroceptivos correspondientes a un laser y un cinturón de ultrasonidos, y un computador que procesa los datos suministrados por el robot durante su recorrido, extrayendo y asociando características del entorno, además de encargarse de los cálculos del filtro de Kalman.

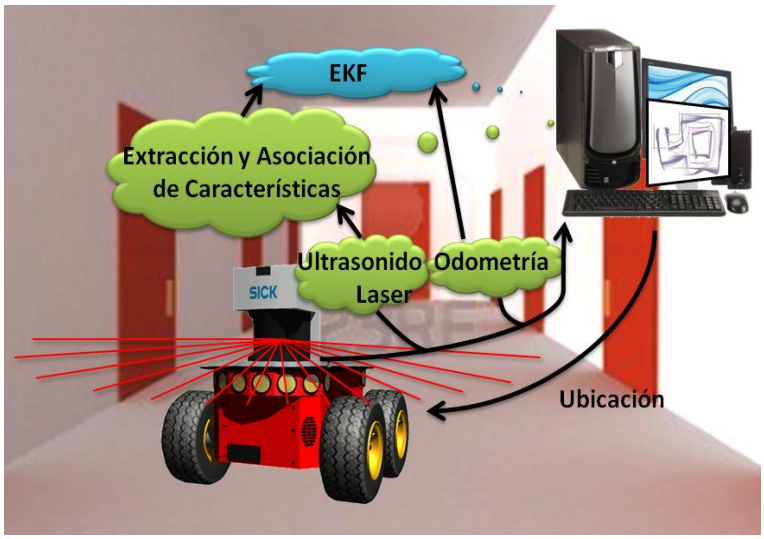


Figura 3.1: Sistema del EKF-SLAM.

## 3.1. Plataforma de Desarrollo

El robot Pioneer 3-DX es una plataforma robótica popular para educación, investigación, creación de prototipos, exhibiciones y otros proyectos. Este robot puede estar equipado con un ordenador integrado en una única placa de formato EBX basada en Pentium, la cual se utiliza para comunicaciones de alto nivel y funciones de control. Aunque el P3-DX ofrece la opción de ordenador incorporado, puede ser equipado con computadoras portátiles, gracias a los 23 kg de carga permitida. El principal inconveniente de la utilización de un ordenador portátil a bordo es la pérdida de la superficie de carga donde otras opciones de sensor o actuador pueden ser instalados.
Las principales características hardware de esta unidad son: comunicaciones basadas en ethernet (opcional), láser (opcional), hasta 252 watios-hora de baterías intercambiables en caliente (las cuales añaden bastante peso al robot), anillo frontal de 8 sensores SONAR, anillo posterior de 8 sensores sonar (opcional), dos motores independientes, 2 ruedas de 19 cm. y una rueda de giro libre. La configuración de las ruedas es de tipo diferencial, estas estan colocadas en el eje perpendicular a la direccion del robot, donde cada rueda es controlada por un motor, de tal forma que el giro del robot queda determinado por la diferencia de velocidad de las ruedas. Otras opciones interesantes son los parachoques, pinzas, visión, telémetros, brújula, etc., todos ellos dispositivos que se pueden adquirir del fabricante.

### 3.1.1. Sensores para la navegación

Para navegar por cualquier tipo de ambiente, los robots requieren de sensores que reconozcan el medio que los envuelve, por lo que se pueden aprovisionar con diversidad de sensores idóneos para el registro de estímulos (color, distancia, fuerza, presión, inclinación, etc). Los sensores son comúnmente clasificados como: propioceptivos y exteroceptivos. Los primeros obtienen el estado interno del robot, mientras que los segundos miden estímulos que se producen en el exterior del robot y su función es suministrar al robot una representación del medio en el que se halla. A continuación se presentarán los sensores más usados para la navegación.

#### 3.1.1.1. Sensores Propioceptivos

Algunos de los sensores propioceptivos de uso más comun en la navegación, son:

•**Odométros**: Son dispositivos generalmente electromecánico que convierten la posición angular de un eje a un código digital o tren de pulsos, mediante los cuales se pueden obtener una estimación de la velocidad y la distancia recorrida. La odometría de un robot brinda un error que es creciente con el tiempo a causa de deslizamientos que se presentan en los elementos móviles.

•**Acelerómetros**: Es un dispositivo electromecánico que mide las fuerzas de aceleración. Estas fuerzas pueden ser estáticas, como la constante de fuerza de la gravedad, o podrían ser dinámico - causado por el movimiento o vibración del acelerómetro. Algunas desventajas de los acelerómetros es que con dificultad detectan las aceleraciones de pequeña magnitud y son muy sensibles a irregularidades en el terreno.

•**Giróscopos**: Es un dispositivo que se utiliza principalmente para la navegación y medición de la velocidad angular. Los giroscopios están disponibles que pueden medir la velocidad de rotación en $x$, $y$ y $z$. En robótica se usan con acelerómetros, para detectar cambios en la orientación de los robots.

La principal ventaja de los sensores propioceptivos, es que no dependen de estímulos externos, ya que sólo miden estados internos del robot. Como inconveniente, resulta difícil efectuar estimaciones fiables, ya que los errores se acumulan continuamente, al no disponer de información del mundo exterior.

#### 3.1.1.2. Sensores Exteroceptivos

Este apartado puntualiza algunos de sensores exteroceptivos más sonados en la robótica:

•**Visión**: Son dispositivos utilizados para la adquisición electrónica de imágenes en movimiento, esta información puede ser procesada para realizar inspecciones y posicionamiento. Son muy importantes, debido a que proporcionan una gran cantidad de información del entorno, probablemente más que ningún otro tipo de sensor.

•**Sónar**: Son dispositivos que evaluan los atributos de un objetivo mediante la interpretación de los ecos de la radio o las ondas de sonido, respectivamente. Los sensores ultrasónicos generan ondas sonoras de alta frecuencia y evaluan el eco que es recibido de nuevo por el sensor. Se calcula el intervalo de tiempo entre el envío de la señal y la recepción del eco para determinar la distancia de un objeto.

•**Láser** Es un dispositivo que al igual que el sonar, utiliza el principio de tiempo de vuelo para medir distancias, pero lo hace con señales electromagnéticas, las cuales tienen una velocidad de propagación de 0,3 m/ns, lo que lo hace mas preciso y costoso.

En contraste a los propioceptivos, los sensores exteroceptivos simplemente detectan lo acontecido en el exterior del robot. Las medidas de este tipo de sensores normalmente son interpretadas por el robot para extraer características del entorno y construir un modelo del mismo.

#### 3.1.1.3. Sensores Comunmente Utilizados en Slam

Para el paso de obtención de observaciones del entorno, requerido en el proceso de SLAM, el Robot Pioneer 3-DX fue equipado con los siguientes sensores:

•**Encoders** Cada motor del robot tiene acoplado un encoder que permite la medición de velocidades y desplazamientos del robot, estos enconders tienen una resolución 500 pulsos por vuelta gracias a los cuales puede llevar a cabo una estimación odométrica de la posición del robot.

•**Sonares** Un total de ocho sensores de este tipo se encuentran incluidos en el robot. El ángulo entre sensores con dos direcciones consecutivas es de 20 grados a excepción de los sensores laterales para los que el ángulo es de 40 grados. La sensibilidad oscila entre 10 centímetros a cuatro metros.

•**Laser SICK LMS 200** El sensor láser de medición SICK LMS 200 es un sensor de distancia de medición extremadamente precisa. El LMS 200 puede ser conectado a través de RS-232 o RS-422, proporcionando mediciones de distancia en un área de 180 grados hasta 80 metros de distancia.

## 3.2. Extracción de Características

Para el presente trabajo las características extraídas del entorno son representadas por líneas rectas que se adquieren de los datos obtenidos a través de un escáner laser y un cinturón de sonares equipados en un robot móvil. Debido a la naturaleza y cantidad de información suministrada por cada uno de los sensores, se implementaron diferentes técnicas de segmentación de los datos, que luego son nuevamente procesados para conseguir la parametrización de línea.
Para los datos provenientes del escáner laser, se aplica inicialmente el método RANSAC *Random Sample Consensus* para eliminar datos espurios y obtener solo aquellos que describan una línea recta. En el caso del cinturón de sonares, los datos de las lecturas son más contaminados en comparación de los del laser, por tal motivo se hace necesario la implementación de un método más robusto, para este caso el algoritmo de extracción de lecturas pertenecientes a la línea es apoyado en la transformada de Hough. La división y parametrización de rectas en ambos casos se realiza con el método SEF *Successive Edge Following* y la técnica TLS *Total Least Square*.

### 3.2.1. Lecturas de los Sensores

El escáner láser y el cinturon de sonares describen un corte 2D del ambiente de trabajo donde el robot navega, esto lo hace a través de una serie de puntos específicos en el sistema de coordenadas polares $(r_R, \theta_R)$, cuyo origen es la ubicación del sensor (Fig. 3.2). Se considera un robot navegando en un entorno 2D y $x_k = [x_r, y_r, \theta_r]$ denota que su ubicación (donde $(x_r, y_r)$ y $\theta_r$ son la posición y orientación respectivamente) en el instante $k$, en un marco de referencia global $(x_G, y_G)$. El espacio de trabajo es descrito por características estáticas rectilíneas (tales como partes de las paredes, puertas, estantes) que son detectadas por el sistema sensorial del robot.
El robot está equipado con sensores de proximidad (anillos de sonar y/o escáner láser) que proporcionan una serie de pasos $s$ y medidas $A^s = [r_R^A, \theta_R^A]$, que son parámetrizados

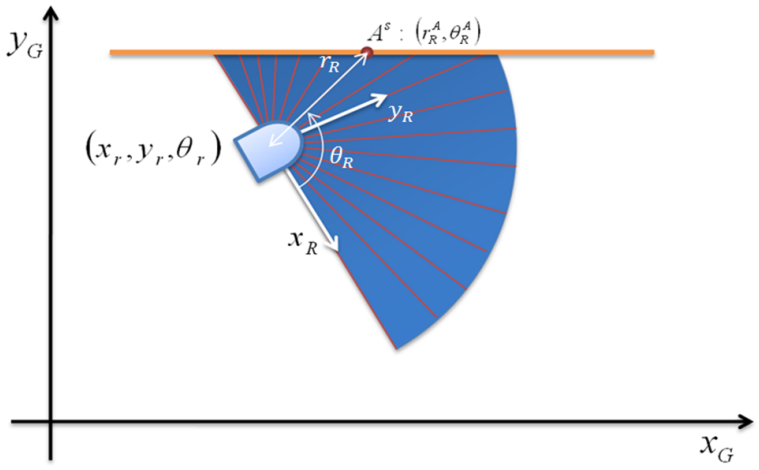


Figura 3.2: Parámetros del escáner láser

por su distancia $r_R^A$ desde el origen hasta el punto en la dirección de detección $\theta_R^A$. La pareja $A^s = [r_R^A, \theta_R^A]$ puede considerarse como las coordenadas polares de la lectura $s$ en un robot centrado en un sistema de referencia. Las coordenadas cartesianas se denotan por:

$$A^s = [x_R^A, y_R^A] = [r_R^A cos(\theta_R^A), r_R^A sin(\theta_R^A)] \tag{3.1}$$

Para el sensor laser utilizado los pasos suministrados son $s = 180$, conclusión que se deriva de Capitulo 3 donde se establece que la resolución angular del escaner es de un grado sexagesimal. En el caso del cinturón de sonares, este solo posee 8 sensores, por lo que se obtendra un total de $s = 8$ lecturas por barrido.

### 3.2.2. RANSAC

El algoritmo RANSAC (*Random Sample And Consensus*) se introdujo por primera vez por Fischler y Bolles [21] en 1981 como un método para estimar los parámetros de un modelo determinado a partir de un conjunto de datos contaminados por grandes cantidades de valores atípicos. El porcentaje de los valores *outliers* que pueden ser manejados por RANSAC puede ser mayor que el 50 porciento del conjunto de datos completo. Tal porcentaje, conocido también como el punto de ruptura, que se supone que es comúnmente el límite práctico para muchas otras técnicas de uso común para la estimación de parámetros (como todos los regresión por mínimos cuadrados o técnicas robustas al menos como media de cuadrados [24] [44] [36] [56] [48] [33]. El dispositivo de medición utilizado para aplicar este metodo es un escáner láser, estos son muy precisos, eficientes y la salida no requiere de muchos cálculos para procesar. El escaneo láser se procesa para calcular los parámetros que describen la recta líneas en el medio ambiente.

El objetivo general del método consiste en ajustar un modelo matemático para un conjunto de datos experimentales. El ajuste también es robusto, ya que reconoce la presencia de valores *outliers* en los datos. Algunos métodos de ajuste utilizan todos los datos disponibles para obtener una solución inicial y luego tratar de eliminar los puntos que no forman parte del modelo. El procedimiento RANSAC funciona de la manera opuesta, se compone de dos pasos que se repiten de forma iterativa (hipótesis y el marco de la prueba) [57]:

**Hipótesis** En primer lugar establece un mínimo de muestra (MSS) seleccionadas al azar del conjunto de datos de entrada y los parámetros del modelo se calculan utilizando sólo los elementos del MSS. La cardinalidad de los MSS es el más pequeño para determinar los parámetros del modelo (a diferencia de otros enfoques, tales como los mínimos cuadrados, donde los parámetros se estiman utilizando todos los datos disponibles, posiblemente con un peso adecuado).

**Prueba** En el segundo paso del RANSAC se evalúa que los elementos del conjunto de datos sean coherentes con el modelo instanciado con los parámetros estimados en el primer paso. El conjunto de tales elementos se llama conjunto consenso.

El siguiente pseudo-código pertenece al algoritmo del RANSAC para la extracción de la línea. Se inicia con un conjunto de puntos (lecturas laser), y se ajusta a una línea de dos puntos seleccionados al azar. Después se establece un conjunto de consenso, que se asume forman parte de los puntos que se pueden considerar que están cerca de la línea. Si el número de puntos del conjunto de consenso es mayor que un cierto umbral, a continuación, se vuelve a reajustar la línea para el conjunto de puntos de consenso (con los mínimos cuadrados, por ejemplo). De lo contrario el algoritmo se repite hasta que se alcanza un número máximo de iteraciones establecido de antemano.

**Algoritmo 1** RANSAC

**Entrada:** $A = \{A_1, A_2, ..., A_s\}$, un conjunto de puntos.
  $P \leftarrow$ Muestra aleatoria de 2 puntos de $A$
  **repetir**
    $R \leftarrow$ Modelo de la línea utilizando $P$
    $C \leftarrow$ subconjunto de puntos de $A$ que se ajustan al modelo $R$ con un margen de error (grupo consenso).
    **si** $N_c \geq T$ **entonces**
      $R^*nuevo \leftarrow$nuevo modelo de línea ajustado a $C$
    **si no**
      $P \leftarrow$ muestra aleatoria de 2 puntos de $A$
    **fin si**
  **hasta que** máximo número de iteraciones o el consenso sea encontrado

El margen de error se puede determinar experimentalmente. Por ejemplo, puede hacer un juicio de datos erróneos, se calcula el error medio y luego se establece el

margen de error a una o dos desviaciones estándar en comparación con el error medio. Por otro lado, el número máximo de iteraciones se puede considerar como el número $k$ de ensayos que se llevarán a cabo con el fin de obtener un conjunto de $n$ datos pertenecientes al modelo correcto con una probabilidad de $z$. Si $w$ se define como la probabilidad antes de que un punto esté dentro del margen de error del modelo, $k$ se obtiene con la Eq. (3.2).

$$k = \frac{\log(1-z)}{\log(1-w^n)} \tag{3.2}$$

Para determinar el número de puntos del conjunto de consenso se puede hacer de la siguiente manera. Sea y la probabilidad de que un punto está dentro del margen de error de un modelo incorrecto. Se pretende que sea muy pequeña. Aunque es muy difícil determinar con precisión y, es razonable suponer que será inferior a $w$.

### 3.2.3. Transformada de Hough

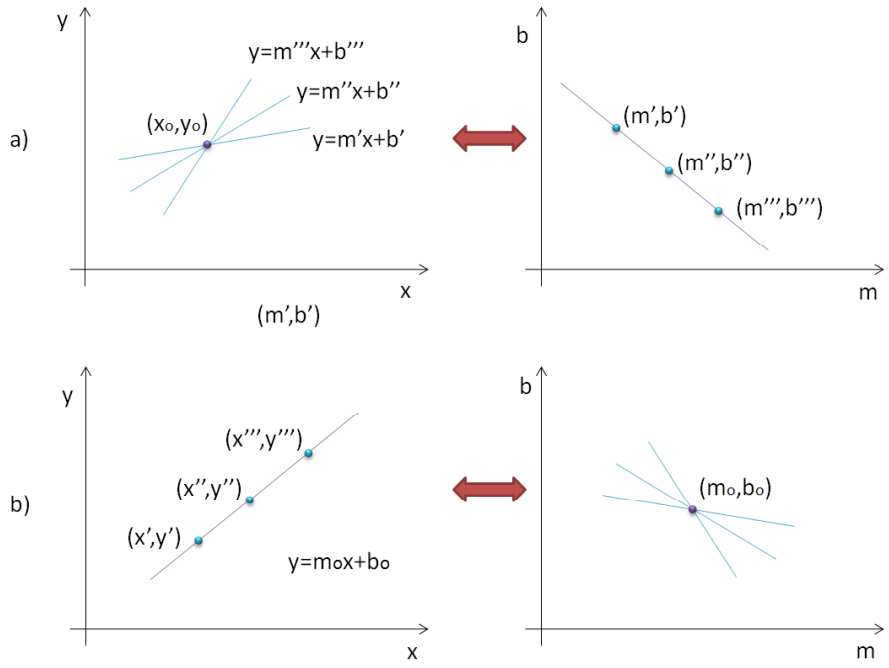


Figura 3.3: Transformaciones punto a linea. a) Un punto con lineas de varios parametros $(m, b)$ a través de estos se transforma en una línea recta en el plano de parametros, b) Puntos colineales se trasnforman en líneas rectas que se intersectan en un único punto en el plano de parametros

Una línea recta en el plano de coordenadas $x - y$ puede ser descrita completamente por solo dos parametros, la pendiente y la intersección con $y$. La parametrización mas

familiar es:

$$y = m_o x + b_o \tag{3.3}$$

donde $m_o$ es la pendiente y $b_o$ es la intersección con el eje $y$. Esta línea, cuando se dibuja en el plano de parametros $m - b$, se convierte en un punto. Esto se puede observar, cuando, dado un punto $(x_o, y_o)$, el conjunto de líneas a través de $(x_o, y_o)$ (donde cada una es parametrizada por algún par $(m, b)$) forman en una línea recta en el plano de parametros $(m - b)$ (Fig. 3.3).

Esta es evidentemente la ecuación de transformación que indica la relación lineal entre $m$ y $b$:

$$b = -x_o m + y_o \tag{3.4}$$

Además, se observa que en caso, los puntos que se encuentran en línea en el plano de coordenadas $(x - y)$, se transforman en sus respectivas líneas en el plano de parámetros, estas líneas que se encuentran en el plano de parámetros tendran un unico punto de intersección. Las coordenadas $(m, b)$ del punto de intersección son exactamente la pendiente y el punto de intersección (parámetros) de la línea que describe los puntos en el plano de coordenadas. Pero para esta parametrización, existen singularidades cuando la pendiente, la variable independiente en este caso, no tiene límite (cuando una línea en el plano de coodenadas es paralela a el eje $y$, la pendiente tiende a infinito).

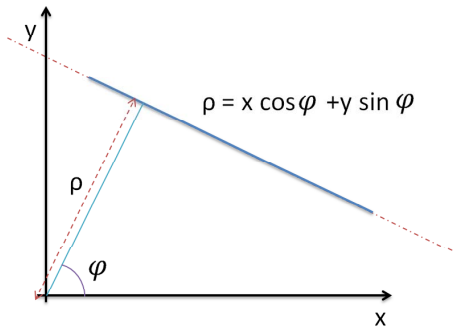


Figura 3.4: Parametrización normal

Como es presentado por Duda y Hart en [17], resulta más conveniente para describir la línea recta utilizar la parametrización normal (Fig. 3.4). En la parametrización normal, una línea en el plano de coordenadas es descrita completamente por dos parámetros,$\varphi$, el ángulo de la normal desde el orígen, y $\rho$, su distancia del origen.
Siguiendo la transformación punto-línea, dado un punto $(x_o, y_o)$ en el plano de coordenadas, se crea un grupo de líneas que pasan a través de $(x_o, y_o)$, cada una de las cuales es parametrizada por algún par $(\varphi, \rho)$, correspondiente a la transformación en una curva senosoidal en el plano de parametros $(\varphi - \rho)$. La ecuación de la curva corresponde a:

$$\rho_o = x cos\varphi + y sin\varphi \tag{3.5}$$

En este caso, los puntos que se encuentran sobre una línea en el plano de coordenadas $(x - y)$ son transformados en sus respectivas curvas senosidales en el plano de parametros, estas curvas en el plano de parametros se intersectaran en un único punto, las coordenadas $(\varphi, \rho)$ de este punto de intersección son exactamente el ángulo (u orientación) de la normal y la distancia desde el origen a la línea que pasa a través de los puntos en el plano de coordenadas. La (Fig. 3.5) ilustra como son las relaciones punto-curva. [53]

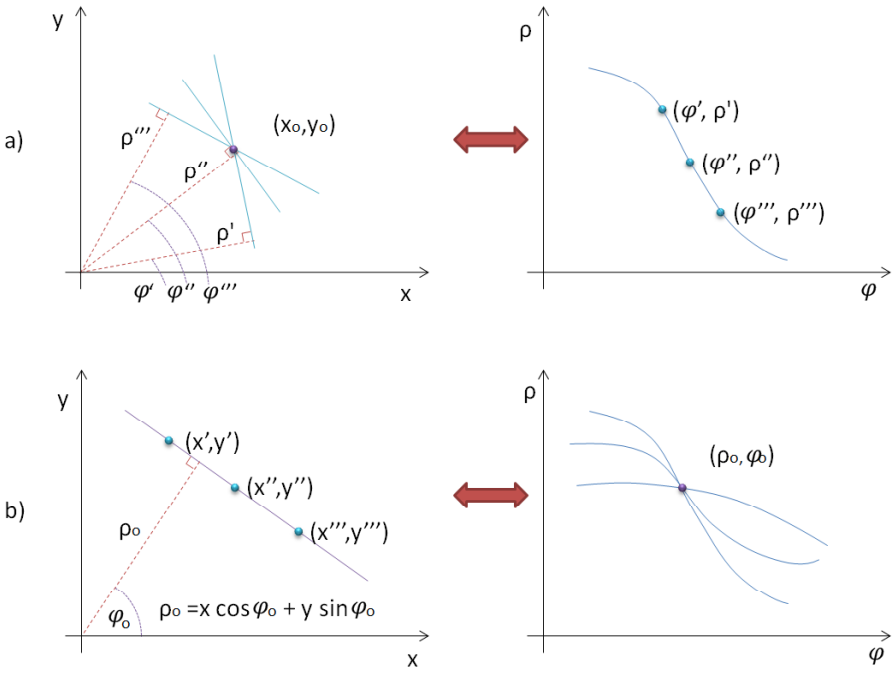


Figura 3.5: Transformaciones punto-curva. a) un punto con lineas a con varios parámetros $(\varphi, \rho)$ pasando a través de ella se transforma en una curva senosoidal en el plano de parámetros. b) Puntos colineales se transforman en curvas que se intersectan en un unico punto en el plano de parámetros.

La transformada de Hough hace uso de la transformación punto-curva para la detección de líneas. Este se basa en la premisa que puntos colineales en el plano de coordenadas serán transformadas en curvas con un unico punto de intersección en el plano de parametros, así, para encontrar una línea, el procedimiento corresponde en tranformar los puntos a curvas en el plano de parametros y localizar las intersecciones de las curvas para determinar la presencia de líneas.

Analizar el plano de parametros para obtener intersecciones precisas, sería una tarea tediosa e ineficiente, por tal motivo se recurre a la discretización del espacio de parametros, obteniendo una grilla 2D donde cada celda representa una región en la cual las intersecciones de las curvas podrían corresponder para ajustar los puntos colineales,

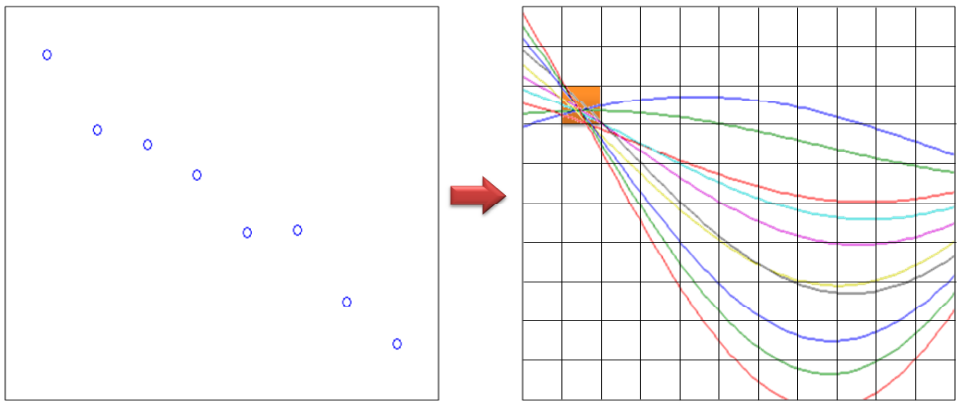

Figura 3.6: Transformada de Hough con grilla 2D

la resolución de la grilla estará basada en que tan ajustada deberá estar la línea a los puntos. A continuación se muestra el algoritmo de la transformada de Hough para la segmentación de líneas:

**Algoritmo 2** Transformada de Hough

**Entrada:** $(x, y)$, un conjunto de puntos en coordenadas cartesianas, $Th$, umbral de votación para identificación de líneas.
$\varphi_d \leftarrow$ vector de ángulos discreto de 0 a $\pi$.
**para** $i = 0$ hasta la longitud del conjunto de datos **hacer**
$\rho(i) = x(i)\cos(\varphi_d) + y(i)\sin(\varphi_d)$
**fin para**
$\rho_d \leftarrow$ parámetro $\rho$ discretizado.
$im \leftarrow$ cálculo de votación para cada pareja $(\varphi_d, \rho_d)$.
$[r, c] \leftarrow$ filas y columnas de $im$ cuya votación es superior a el umbral $Th$.
Extracción de los parámetros de las rectas $(\rho_{lin}, \varphi_{lin})$

### 3.2.4. Successive Edge Following (SEF)

Este algoritmo (Seguimiento de Extremos Consecutivos) [47] [6] trabaja directamente con las distancias medidas por el sensor, que se encuentran en formato polar $(\rho, \phi)$. Por consiguiente, no se necesita llevar a cabo ninguna transformación a coordenadas cartesianas $(x, y)$. En este método, se considera que un segmento termina cuando la diferencia entre una distancia y su siguiente $(|\rho_{j+1} - \rho_j|)$ es mayor a un valor umbral $(d_{max})$ anteriormente fijado.

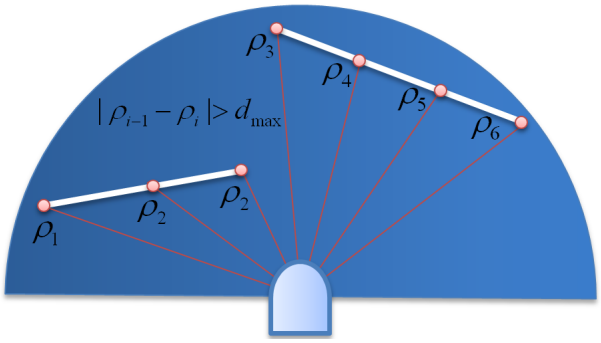


Figura 3.7: Seguimiento de Extremos Consecutivos

Al terminar el algoritmo lo único que se obtiene es una serie de conjuntos de distancias, pero éste no es el objetivo. Lo que se pretende es conseguir un conjunto de líneas, y para ello es necesario realizar una etapa de post-procesamiento, para ajustar una línea a cada conjunto mediante algún método de regresión [47] [6].

La principal ventaja de este algoritmo es su rapidez, más, tiene el inconveniente de ser dependiente de un umbral que no varia y su adquisición ajustandolo a los datos. Además se pueden producir errores de detección en situaciones determinadas, como por ejemplo al colocar el sensor frente a un rincón (donde la diferencia entre las distancias consecutivas es poca), o una superficie donde se describa una línea recta con puntos muy separados entre sí. El algoritmo seguimiento de extremos consecutivos es el siguiente:

**Algoritmo 3** SEF

**Entrada:** $\rho_1, \ldots, \rho_n$, un conjunto de medidas de distancia
  $i \leftarrow 1$
  **para** $j = 1$ hasta $n - 1$ **hacer**
    **si** $\rho_{j+1} - \rho_j > Th$ **entonces**
      $S \leftarrow \{\rho_i, \ldots, \rho_j\}$
      Almacenar $S$
      $i \leftarrow j + 1$
    **fin si**
  **fin para**

### 3.2.5. TLS Total Least Squares

Difícilmente se podría nombrar a otro método que se utilice con tanta frecuencia como el método que se conoce como el *Least Squares*, aproximación que en estadística se considera como un estimador de máxima verosimilitud de un modelo de regresión lineal bajo criterios estándar (distribución normal residual de media cero y matriz de covarianza que es un múltiplo de la identidad). Del mismo modo, el TLS *total least*

*square* es un estimador de máxima verosimilitud en el modelo de errores invariantes.
El TLS fue desarrollado por Golub y Van Loan [23] como una técnica de solución para un sistema sobre determinado de ecuaciones y luego perfeccionado por Van Huffel [51], es una extensión del usual método de mínimos cuadrados, permitiendo manejar también ciertas inconsistencias en la matriz de datos. La popularidad del TLS es probablemente debido al hecho de que la minimización del residuo puede ser hecha cercanamente a la forma analítica, este método es líneal y es consistente con el principio de ortogonalidad de la aproximación óptima en sentido del error cuadrático medio.
TLS es un método de ajuste que se utiliza cuando hay errores tanto en el vector de observación y en la matriz de datos, por eso es una buena opción para obtener los parámetros apropiados de la línea final. El TLS minimiza la suma del cuadrado de las distancias de los puntos a la recta (medidas en perpendicular). En el caso del problema del TLS, se quiere tomar en cuenta las perturbaciones en la matriz de sensibilidad, para ello el TLS se define como la minimización sobre el rango y perturbaciones de la matriz de sensibilidad de tal forma que la distancia normal euclidiana entre la recta y los puntos que la componen sea la menor posible.
En el presente trabajo los puntos que han sido clasificados como *inliers*, se usan para calcular los parámetros de la recta. Para un conjunto de puntos $(x, y)$ en coordenadas cartesianas con una desviación de $\sigma$, se halla la diferencia entre las coordenadas de cada punto y la media aritmética correspondiente, así:

$$dx = x - \frac{1}{n}\sum_{i=1}^{n} x_i \tag{3.6}$$

$$dy = y - \frac{1}{n}\sum_{i=1}^{n} y_i \tag{3.7}$$

Se definen las variables intermedias $N$ y $D$ para dar mayor claridad al proceso:

$$N = -2\sum \frac{dx * dy}{\sigma^2} \tag{3.8}$$

$$D = \frac{1}{\sigma^2}\sum (dy^2 - dx^2) \tag{3.9}$$

Finalmente se obtienen los parametros de la línea que es descrita por el conjunto de puntos iniciales:

$$\theta = \frac{atan(\frac{N}{D})}{2} \tag{3.10}$$

$$\rho = x_{mean} * cos(\theta) + y_{mean} * sin(\theta) \tag{3.11}$$

Cabe anotar que los parámetros de la recta obtenidos a través de la aplicación del TLS están dados en forma polar paramétrica, de esta forma se ingresan los parámetros a el filtro extendido de Kalman para realizar el proceso de SLAM.
Cuando tenemos variables correspondientes a valores de mediciones experimentales, estas tienen incertidumbres debido a limitaciones de medición (por ejemplo, la precisión

del instrumento) que se propagan en la combinación de variables en determinada función. Para propagar la incertidumbre y obtener la matriz de covarianza de cada línea tenemos:
Sea $(r_s, t_s)$ un conjunto de puntos en coordenadas polares,$(\sigma)$ la desviación estándar por cada punto, y $(r_s \cos(t_s), r_s \sin(t_s))$ su correspondiente en coordenadas cartesianas, como las ecuaciones para obtener los parametros de la línea son un conjunto de combinaciones no-lineales de la variable $r_s$, estas deben ser linealizadas por aproximación a una expansión de primer orden en serie de Taylor, así:

$$N = -2 \sum \frac{dx * dy}{\sigma^2} \tag{3.12}$$

$$N = -2 \sum \frac{(y - y_{mean}) * (x - x_{mean})}{\sigma^2} \tag{3.13}$$

$$N = -2 \sum \frac{x * y - y_{mean} * x - x_{mean} * y + y_{mean} * x_{mean}}{\sigma^2} \tag{3.14}$$

$$N = -2 \sum \frac{r_s^2 \cos(t_s) \sin(t_s) - r_s y_{mean} \cos(t_s) - r_s x_{mean} \sin(t_s) + y_{mean} x_{mean}}{\sigma^2} \tag{3.15}$$

$$\frac{\partial N}{\partial r_p} = \frac{2}{\sigma^2}(x_{mean} * sin(t_s) + y_{mean} * cos(t_s) - r_s * sin(2 * t_s)) \tag{3.16}$$

$$D = \frac{1}{\sigma^2} \sum (dy^2 - dx^2) \tag{3.17}$$

$$D = \frac{1}{\sigma^2} \sum ((y - y_{mean})^2 - (x - x_{mean})^2) \tag{3.18}$$

$$D = \frac{1}{\sigma^2} \sum (y^2 - 2 * y * y_{mean} + y_{mean}^2 - x^2 + 2 * x * x_{mean} - x_{mean}^2) \tag{3.19}$$

$$D = \frac{1}{\sigma^2} \sum (r_s^2 \sin(t_s)^2 + 2 r_s (\cos(t_s) x_{mean} - \sin(t_s) y_{mean}) + y_{mean}^2 - r_s^2 cos(t_s)^2 - x_{mean}^2) \tag{3.20}$$

$$\frac{\partial D}{\partial r_s} = \frac{1}{\sigma^2} \sum (2 * r_s \sin(t_s)^2 - 2 \sin(t_s) * y_{mean} - 2 * r_s cos(t_s)^2 + 2 \cos(t_s) * x_{mean}) \tag{3.21}$$

$$\frac{\partial D}{\partial r_s} = \frac{2}{\sigma^2} \sum (-2 * r_s \cos(2 * t_s) - \sin(t_s) * y_{mean} + \cos(t_s) * x_{mean}) \tag{3.22}$$

$$\theta_l = \frac{atan(\frac{N}{D})}{2} \tag{3.23}$$

$$\frac{\partial \theta_l}{\partial r_s} = \frac{1}{2} \frac{D * \frac{\partial N}{\partial \rho} - N * \frac{\partial D}{\partial \rho}}{D^2 - N^2} \tag{3.24}$$

$$\rho = x_{mean} * cos(\theta_l) + y_{mean} * sin(\theta_l) \tag{3.25}$$

$$\frac{\partial \rho}{\partial r_s} = \frac{\partial \theta}{\partial r_s} * \frac{\partial \rho}{\partial \theta} + \frac{1}{n} * cos(\theta - \theta_l) \tag{3.26}$$

Las componentes de la matriz de covarianza son:

$$\sigma\theta_l = \frac{\partial\theta_l}{\partial r_s}^2 * \sigma^2 \tag{3.27}$$

$$\sigma\rho = \frac{\partial\rho}{\partial r_s}^2 * \sigma^2 \tag{3.28}$$

$$\sigma\theta_l\rho = ((\frac{\partial\theta_l}{\partial r_s} * \frac{\partial\rho}{\partial r_s}) * \sigma)^2 \tag{3.29}$$

Obteniendo de esta forma la matriz de covarianza de la recta:

$$Cov_L = \begin{bmatrix} \sigma\theta_l & \sigma\theta_l\rho \\ \sigma\theta\rho & \sigma\rho \end{bmatrix} \tag{3.30}$$

## 3.3. Asociación de Características

La asociación de características es sin duda una parte muy importante del algoritmo de SLAM. La correcta correspondencia de una característica observada con una perteneciente al vector de estado es esencial para la construcción de un mapa consistente. Una asociación de datos incorrecta es generará problemas para construir un mapa confiable y una sola asociación equivocada puede conducir a un fracaso en la construcción eficiente de mapas.
La asociación de las características relaciona la información observada en la iteración actual que se obtiene respecto a la posición del robot, con una característica observada en iteraciones anteriores y que se encuentra almacenada en el vector de estado. Función de asociación depende en gran medida el error en el robot, entre mayor es el error en la posición del robot, más difícil se vuelve a asociar características. Como se utiliza un sensores basados en el registro de medio ambiente, las características, que se extraen de los datos del área de navegación, sólo se pueden distinguir por su ubicación. Para aceptar sólo las asociaciones más probables, las únicas características consideradas son las que se encuentran en proximidades de la observación [30].
La disponibilidad de un modelo estocástico, tanto para el mapa como para las mediciones nos permite comprobar cada correspondencia de las características observadas mediante la predicción de la ubicación de las caracteristicas del mapa con relación a la referencia del sensor, y determinar la compatibilidad con una prueba de hipótesis sobre la innovación y la covarianza de cada pareja posible.
Las características se asocian con una ventana de validación que define el error máximo admisible entre la observación y la predicción. Una ventana de validación basada en la distancia de Mahalanobis, con algunas restricciones se ha utilizado para la asociación características en este trabajo.

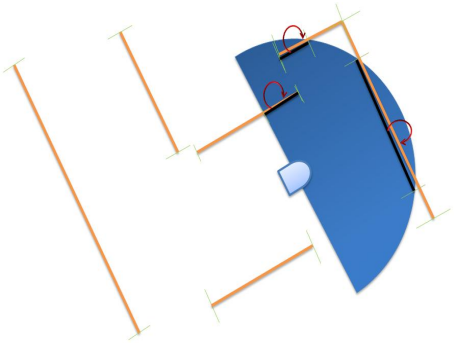

Figura 3.8: Compatibilidad Individual. El mapa color naranja corresponde al predicho en el instante $k$ cuando las observaciones en negro son medidas.

### 3.3.1. Distancia de Mahalanobis

La validación de hipótesis coincidentes con la distancia de Mahalanobis es ampliamente utilizado en aplicaciones de robótica y en general las técnicas de asociación de datos, la distancia de Mahalanobis es definida por la innovación y su covarianza, se compara con un umbral definido por la distribución de $\chi^2$ para validar que la hipótesis coincida.
La distancia de Mahalanobis, también conocida como la distancia estadística, es una distancia que para cada uno de sus componentes (las variables) toma su variabilidad en cuenta al determinar su distancia al centro correspondiente. Así, los componentes con alta variabilidad reciben menos peso que los componentes con baja variabilidad. Esto se logra ajustando la base de los componentes considerados, es decir, para dos puntos $X_i = (\rho_{11i}, \theta_{12i})$ y $Y_i = (\rho_{12i}, \theta_{22i})$ la distancia de Mahalanobis esta dada por:

$$d_M = \sqrt{(\rho_{1i} - \rho_{2i})^T C^{-1} (\theta_{1i} - \theta_{2i})} \tag{3.31}$$

Donde $C_{nxn}$ es una matriz de covarianza no singular (por lo tanto simétrica definida positiva ). La $d_M$ es un método estándar que se utiliza en la asociación de datos para el seguimiento de características. [42]
Después del proceso de detección, $m$ estimaciones de características, y $n$ mediciones están disponibles. En un principio, hay $mn$ hipótesis de emparejamiento, la validación se realiza mediante una prueba estadística basada en la distancia de Mahalanobis y su aproximacion por la distribución $\chi^2$. [37]:

$$v^T C^{-1} v \leq \chi^2 \tag{3.32}$$

Donde $v$ es el vector entre el estado de la caracteristica predicha y una medición precisa. Este test debe ser teoricamente calculado para $mxn$ hipótesis.

### 3.3.2. Reducción de hipótesis

La técnica de asociación de datos implementada realiza pruebas de validación para cada hipótesis de emparejamiento con el fin de trabajar con un conjunto reducido de

hipótesis validadas.

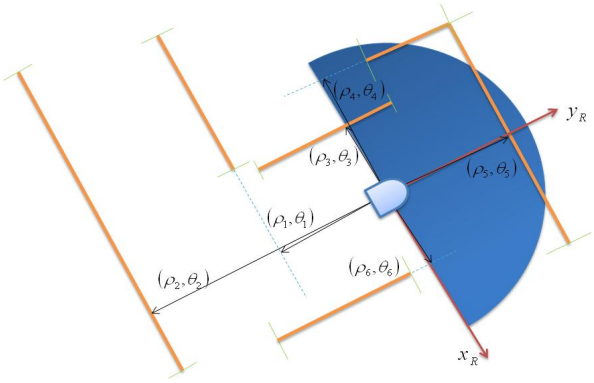


Figura 3.9: Características que han sido predecidas, con sus respectivos parámetros.

La primera de ellas consiste en descartar aquellas predicciones que podrían estar ubicada fuera del alcance del sensor, ya que ninguna observación conseguiría ser asociada a estas. Para la disminución en el número de predicciones a asociar, se eliminan aquellas características predichas que poseen un $\rho$ mayor al alcance del robot y un $\theta$ mayor a $-\pi/2$ y menor a $\pi/2$, conservando de esta forma solo las predicciones que pueden ser vistas desde el estado actual del robot. La Fig. 3.9 representa un paso de predicción de características, donde todos los hitos del vector de estado son desplazados y rotados simbolizando las posibles observaciones que podría tener el robot en el estado actual.
La asociación de datos indica si las rectas que se miden en determinado instante corresponden con alguna que ya se tenga en el vector de estados, sin embargo existen algunas restricciones a tener en cuenta en el momento de asegurar que dos rectas son las mismas.

#### 3.3.2.1. Vectores normales a las rectas

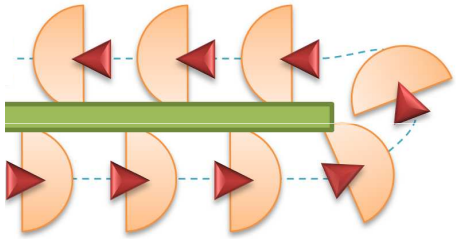

Figura 3.10: Robot Navegando

El robot atraviesa un pasillo donde ve paredes pero retorna por el otro lado de la misma, en principio sería la misma pared, pero un laser no brinda información de textura, por lo que no se puede asumir que es la misma, así, resulta mejor identificarlas con diferentes rectas.

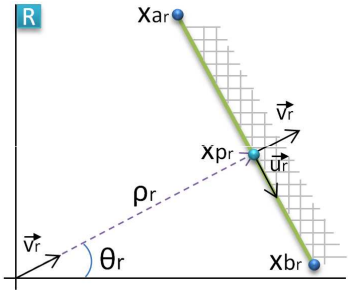


Figura 3.11: Parámetros de las rectas detectadas

Para realizar la identificación de las rectas, la primera vez que se ve la recta se calcula su vector normal en la referencia del robot, $\vec{u_r}$ representa el vector director y $\vec{v_r}$ el vector normal saliente.$\vec{u_r}.\vec{v_r} = 0$ ya que existe ortogonalidad entre los vectores.

$$\vec{u_r} = \begin{bmatrix} -sin(\theta) \\ cos(\theta) \end{bmatrix} \tag{3.33}$$

$$\vec{v_r} = \begin{bmatrix} cos(\theta) \\ sin(\theta) \end{bmatrix} \tag{3.34}$$

Cuando se observa de nuevo la pared por otro lado $z = (\rho_r, \theta_r)$ el algoritmo de asociación debe indicar que las rectas no son las mismas, si bien puede ser la misma pared, para el algoritmo de SLAM es importante registrar el lado correspondiente, para ello, se realizan los siguientes pasos:

•Traslación del vector $vecv_w$ a la referencia del robot (es una predicción desde el punto de vista vectorial): Se invierte el robot $X_{RW}$=$\ominus X_{WR}$. Se realiza el calculo de la rotacion $H_{RW}$ Se obtiene los vectores trasladados $\vec{hv} = H_{RW}.\vec{v_w}$

•Trasladación los puntos extremos de las rectas: $\vec{h_a}$=$H_{RW} \oplus X_{aw}$ $\vec{h_b}$=$H_{RW} \oplus X_{bw}$

•Cálculo del ángulo entre los vectores $\vec{v_r}$ y $\vec{h_v}$

$$cos(\theta) = \frac{\vec{v_r}.\vec{h_v}}{|v_r||h_v|} \tag{3.35}$$

•Si $\theta$>90° las rectas no se corresponden.

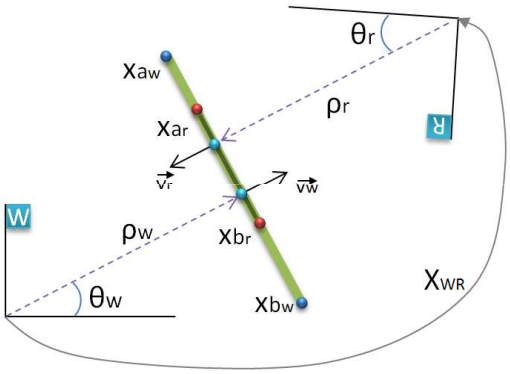


Figura 3.12: Asociación de datos.

#### 3.3.2.2. Proyección de rectas

Existen rectas con vectores normales similares, pero estas pueden estar separadas debido a la existencia de puertas, corredores, etc., o puede ocurrir que las rectas son las mismas, pero debido al corto alcance del sensor o la segmentación son fraccionadas de forma diferente. En este último caso la predicción de la recta y la observación deben ser comparadas desde el punto de vista de sus extremos proyectados. Al proyectar $h_b$

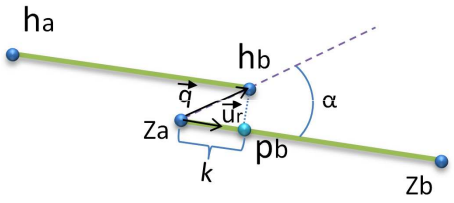


Figura 3.13: Extremos de rectas proyectados.

en $z$, se puede definir si las rectas se encuentran solapadas, para realizar la proyección se ejecutan los siguientes pasos:

$$\vec{q} = h_b - z_a \tag{3.36}$$

•Calculo de $K$

$$\vec{q}.\vec{u_r} = |\vec{q}||\vec{u_r}|cos(\alpha) \tag{3.37}$$

$$K = \vec{q}|cos(\theta) \tag{3.38}$$

$$K = \frac{\vec{q}\vec{u_r}}{|\vec{u_r}|} = \vec{q}.\vec{u_r} \tag{3.39}$$

•Calculo de $P_b$

$$P_b = K\vec{u_r} + z_a \tag{3.40}$$

•$P_b = [p_x, p_y]$, si ($p_x$>$z_{a_x}$) o si ($p_y$>$z_{a_y}$), el punto de proyecta sobre la recta $z$ y hay solape.

### 3.3.3. Algoritmo de Compatibilidad Individual

Para representar la compatibilidad o correspondencia de las observaciones con las carecterísticas del mapa obtenidas en el paso de predicción, se recurre a crear una matriz de compatibilidad de $n$ filas (concerniete a las observaciones) y $m$ columnas (representando las predicciones), cuyas intersecciones indican con un 1 si la observación y la predicción corresponden a la misma característica, de lo contrario contendrá un valor de 0, como lo muestra la Fig. 3.14.

| | $h_1$ | $h_2$ | $h_3$ | $h_4$ | $h_5$ | $h_6$ | $h_7$ | $h_8$ | | $h_m$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $z_1$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 |
| $z_2$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 1 |
| $z_3$ | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 |
| | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| $z_n$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Figura 3.14: Matriz de compatibilidad individual.

Esta matriz de compatibilidad individual se crea a partir de la implementación de los métodos descritos anteriormente, donde se verifican restricciones de observación, sentido de visualización y traslapo entre las líneas observadas y las pronosticadas. Después de realizar esta depuración se evalúa la cercanía, utilizando la distancia de Mahalanobis como referencia para demostrar correspondencia. El algoritmo 4 muestra como se realiza el proceso para la obtención de la matriz de correspondencia.

### 3.3.4. Actualización de los Puntos Extremos

Cuando dos características son asociadas, sus parámetros $(\rho, \theta)$ que se encuentran en el vector de estados se actualizan por medio del Filtro de Kalman Extendido, más, tambien es menester actualizar los puntos extremos de la línea actualizada.
Para las líneas rectas horizontales, las cuales se determinan como aquellas que tienen un parametro $|\rho|$ entre 0.7854 y 2.3562 radianes, el reajuste de los puntos de inicio y final de la línea se hace teniendo en cuenta los valores extremos en el eje $x$ de las líneas

**Algoritmo 4** Compatibilidad Individual de Observaciones

**Entrada:** $z = \{z_1, z_2, ..., z_n\}$ es un conjunto observaciones, $h = \{h_1, h_2, ..., h_m\}$ es un conjunto predicciones.

**para** $i = 0$ hasta $n$ **hacer**
  $\vec{v_z} \leftarrow$ Vector normal saliente de $z_i$
  $z_{ab} \leftarrow$ Coordenadas de los puntos extremos de la observación
  **para** $j = 0$ hasta $m$ **hacer**
    **si** $\rho_{z_i} < SensorRange$ & $(\pi/2 > \theta_{z_i} > -\pi/2)$ **entonces**
      $\vec{v_h} \leftarrow$ Vector normal saliente de $h_j$
      $h_{ab} \leftarrow$ Coordenadas de los puntos extremos de la predicción
      $\theta = \arccos(z_i * v_h^T)$
      Se verifican los vectores normales de las líneas
      **si** $\theta < 90$ **entonces**
        $\vec{q} = h_b - z_a$
        $K = \vec{q}.\vec{u_z}$
        $P_b = K\vec{u_r} + z_a$
        Se verifican si existe solapamiento de las líneas
        **si** $(p_x > z_{a_x})||(p_y > z_{a_y})$ **entonces**
          $e = z_i - h_j$
          **si** $e^T C^{-1} e \leq \chi^2$ **entonces**
            IC(ij)=1
          **si no**
            IC(ij)=0
          **fin si**
        **si no**
          IC(ij)=0
        **fin si**
      **si no**
        IC(ij)=0
      **fin si**
    **si no**
      IC(ij)=0;
    **fin si**
  **fin para**
**fin para**

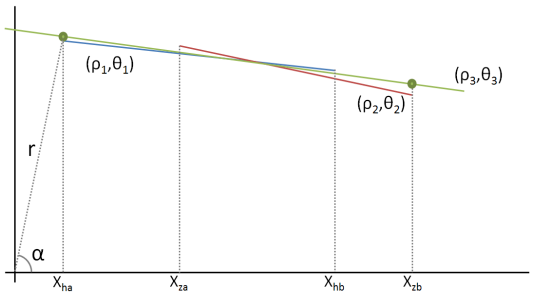


Figura 3.15: Punto extremo de líneas horizontales.

asociadas Fig (3.15). Haciendo uso de los parametros de la línea actualizados $(\rho, \theta)$ y las coordenadas $x$ de los extremos de la recta, se obtienen los nuevos limites de línea realizando una proyección de dichas coordenadas en la recta descrita por el vector de estados, así:

$$r = \frac{\rho}{\cos(\alpha - \theta)} \tag{3.41}$$

$$\cos(\alpha) = \frac{X_{ha}}{r} \tag{3.42}$$

$$\tan(\alpha) = \frac{\rho - X_{ha}\cos(\theta)}{X_{ha}\sin(\theta)} \tag{3.43}$$

En procedimiento silimar se hace para las líneas verticales las cuales se determinan como aquellas que tienen un parametro $|\rho|$ fuera del rango entre 0.7854 y 2.3562 radianes,para este caso, el reajuste se hacen basados en los valores extremos en el eje $y$ de las líneas asociadas Fig(3.16).

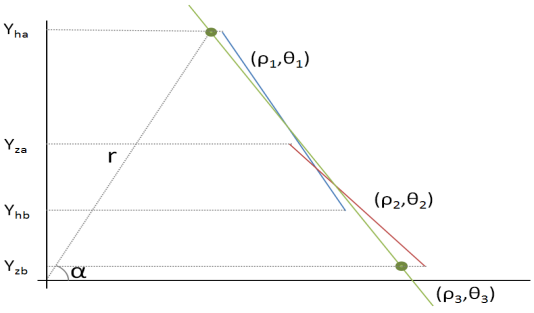


Figura 3.16: Punto extremo de líneas verticales.

Para la obtención de los extremos de la línea se proyecta en el eje y los extremos de las líneas asociadas así:

$$\sin(\alpha) = \frac{Y_{ha}}{r} \tag{3.44}$$

$$\cot(\alpha) = \frac{\rho - Y_{ha}\sin(\theta)}{Y_{ha}\cos(\theta)} \tag{3.45}$$

Para ambos casos los extremos en coordenadas cartecianas de la recta se define como:

$$x = \frac{\rho \cos(\alpha)}{\cos(\alpha - \theta)} \tag{3.46}$$

$$y = \frac{\rho \sin(\alpha)}{\cos(\alpha - \theta)} \tag{3.47}$$

# Capítulo 4

# FILTRO DE KALMAN, EXTENCIÓN E IMPLEMENTACIÓN

El filtro Kalman es un conjunto de ecuaciones matemáticas que proporciona un medio eficiente computacionalmente (recursivo) para estimar el estado de un proceso, mientras que minimiza el error cuadrático medio. El filtro es muy potente en varios aspectos: es compatible con las estimaciones de los estados pasados, presentes y futuros, y lo puede hacer incluso cuando la naturaleza precisa del sistema modelado es desconocida.

## 4.1. Filtro de Kalman

En 1960, R.E. Kalman publicó su artículo que describe una solución recursiva al problema discreto de filtrado lineal [27]. Desde ese momento, debido en gran parte a los avances en la computación digital, el filtro de Kalman ha sido objeto de una amplia investigación y aplicación, particularmente en el área de navegación autónoma o asistida. [5]

### 4.1.1. Proceso a ser estimado

El filtro Kalman aborda el problema general de tratar de estimar el estado $x \in \Re^n$ de un proceso controlado de tiempo discreto que se rige por la ecuación en diferencias estocástica lineal:

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \tag{4.1}$$

Con una medida $z \in \Re^m$ que es:

$$z_k = Hx_k + v_k \tag{4.2}$$

Las variables aleatorias $w_{k-1}$ y $v_k$ representan el ruido del proceso y de medición (respectivamente). Se suponen que son independientes (entre ellas), blanco, y con probabilidad normal distribuciones.

$$p(w) \sim N(0, Q) \tag{4.3}$$

$$p(v) \sim N(0, R) \tag{4.4}$$

En la práctica, las matrices de covarianza del ruido del proceso $Q$ y covarianza del ruido de medición $R$ pueden cambiar con cada paso de tiempo o de la medición, sin embargo, se asumen que son constantes.

La matriz $A$ de $nxn$ de la ecuación en diferencias Eq. (4.1) refiere al estado en el paso de tiempo anterior $k-1$ con el estado en el paso actual $k$, en la ausencia de cualquier función de conducción o de ruido del proceso. En la práctica puede cambiar con cada paso de tiempo, pero se supone que es constante. La matriz $B$ de $nx1$ relaciona la entrada de control opcional $u \in \Re^l$ para el estado $x$. La matriz $H$ de $mxn$ en la ecuación de medición Eq. (4.2) relaciona el estado con la medición $z_k$. En la práctica puede cambiar con cada paso de tiempo o de la medición, pero se supone constante.

### 4.1.2. Origenes Computacionales del Filtro

Definimos $\hat{x_k}^- \in \Re^n$ como una estimación a priori del estado en el paso $k$ dado el conocimiento del proceso antes del paso $k$, y sea $\hat{x_k} \in \Re^n$ nuestro estado estimado a posteriori en el paso $k$ dada la medición $z_k$. Podemos entonces definir a priori y a posteriori los errores de estimación, como

$$e_k^- \equiv x_k - \hat{x_k}^- \tag{4.5}$$

y

$$e_k \equiv x_k - \hat{x_k} \tag{4.6}$$

La estimación a priori de covarianza de error es entonces:

$$P_k^- = E[e_k^- e_k^{-^T}] \tag{4.7}$$

Y una estimación de la covarianza a posteriori el error:

$$P_k^= E[e_k e_k^T] \tag{4.8}$$

Para derivar las ecuaciones para el filtro de Kalman, comenzamos con el objetivo de encontrar una ecuación que calcula una estimación a posteriori del estado $\hat{x}$ como una combinación lineal de una estimación a priori $\hat{x}^-$, y una diferencia ponderada entre una medida real y una predicción de medición como se muestra a continuación en Eq. (4.9).

$$\hat{x} = \hat{x}^- + K(z_k - H\hat{x}_k^-) \tag{4.9}$$

La diferencia $(z_k - H\hat{x}_k^-)$ en Eq. (4.9) se denomina la innovación de medición, o el residuo. El residuo refleja la discrepancia entre la medida prevista $H\hat{x}_k^-$ y la medida real $z_k$. Un residual de cero significa que los dos están completamente de acuerdo.

La matriz $K$ de $nxm$ en Eq. (4.9) es elegida para formar la ganancia o factor de mezcla que minimiza la a posteriori covarianza del error Eq. (4.8). Esta minimización se puede lograr, primero sustituyendo Eq. (4.9) en la definición anterior para $e_k$, sustituyendo esto en Eq. (4.8), realizando las expectativas indicadas, teniendo el derivado de la traza del resultado con respecto a $K$, estableciendo que resultado igual a cero y, a continuación solucionando $K$. Una forma de obtener la $K$ resultante que minimiza Eq. (4.8) viene dada por

$$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1} == \frac{P_k^- H^T}{H P_k^- H^T + R} \tag{4.10}$$

En cuanto a Eq. (4.10) vemos que a medida que la covarianza del error de medición $R$ se aproxima a cero, la ganancia $K$ da un mayor peso al residuo. Específicamente,

$$\lim_{R_k \to 0} K_k = H^{-1} \tag{4.11}$$

Por otro lado, como la covarianza a priori del error de estimación $R$ se aproxima a cero, la ganancia $K$ brinda menor peso a el residuo. Específicamente,

$$\lim_{P_k^- \to 0} K_k = 0 \tag{4.12}$$

Otra forma de pensar acerca de la ponderación de $K$ es que a medida que la covarianza del error de medición $R$ se aproxima a cero, la medición real $z_k$ es de más y más çonfianza", mientras que la medida predicha $H\hat{x}_k^-$ es de confianza cada vez menor. Por otro lado, como la estimación a priori de la covarianza de error $P_k^-$ se aproxima a cero la medición real $z_k$ es de confianza cada vez menor, mientras que la medición predicha $H\hat{x}_k^-$ es de mayor confianza.

### 4.1.3. Origenes Probabilisticos del Filtro

La justificación de Eq. (4.9) se basa en la probabilidad de que la estimación a priori $\hat{x}_k^-$ es condicionada en todas las mediciones anteriores (la regla de Bayes). Por ahora basta señalar que el filtro de Kalman mantiene los dos primeros momentos de la distribución del estado.

$$E[x_k] = \hat{x}_k \tag{4.13}$$

$$E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T] = P_k \tag{4.14}$$

La estimación a posteriori estado Eq. (4.9) refleja la media (el primer momento) de la distribucion del estado que se distribuye normalmente si las condiciones de Eq. (4.3) y Eq. (4.4) se cumplen. La estimación a posteriori de la covarianza del error Eq. (4.8)

refleja la varianza de la distribución del estado (el segundo momento no central). En otras palabras,

$$p(x_k|z_k) \sim N(E[x_k], E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T]) = N(\hat{x}_k, P_k) \tag{4.15}$$

El filtro de Kalman estima un proceso mediante el uso de una forma de control de retroalimentación: el filtro estima el estado del proceso en algún momento y luego obtiene la retroalimentación en forma de (ruidoso) mediciones. Por lo tanto, las ecuaciones para el filtro de Kalman se encuentran en dos grupos: las ecuaciones de tiempo de actualización y las ecuaciones de medición de actualización. Las ecuaciones de actualización de tiempo son las responsables de proyectar hacia adelante (en el tiempo) el estado actual y las estimaciones de la covarianza del error para obtener las estimaciones a priori para el próximo momento.
Las ecuaciones de actualización de medición son responsables de la realimentación, es decir para la incorporación de una nueva medición en la estimación a priori, para obtener una mejora de una estimación a posteriori. Las ecuaciones de actualización de tiempo también puede ser considerado como ecuaciones de predicción, mientras que las ecuaciones de actualización de la medición pueden ser considerados como ecuaciones correctores.

## 4.2. Filtro de Kalman Extendido

En el caso de que el sistema dinámico sea no-lineal, es posible usar una modificación del algoritmo llamada Filtro de Kalman Extendido, el cual linealiza el sistema en torno al identificado realmente, para calcular la ganancia y la dirección de corrección adecuada. En este caso, en vez de haber matrices $A$, $B$ y $H$, hay dos funciones $f$ y $h$ que entregan la transición de estado y la observación (la salida contaminada) respectivamente. Asumimos que el proceso tiene un vector de estado $x \in \Re^n$, este proceso es gobernado por una ecuación no lineal de diferencia:

$$x_k = f(x_{k-1}, u_{k-1}, w_{k-1}) \tag{4.16}$$

Con una medida $z \in \Re^m$ que es:

$$z_k = h(x_k, v_k) \tag{4.17}$$

En este caso la funcion no lineal $f$ relaciona el estado anterior con el actual. Se incluye como parámetros cualquier función de control $u_{k-1}$ y el ruido del proceso de media cero $w_k$. La función no lineal en la ecuación de medición Eq. 4.17 relaciona el estado $x_k$ con la medición $z_k$. Se puede aproximar el vector de estado y la medición sin conocer los valores individuales del ruido como:

$$\tilde{x}_k = f(\hat{x}_{k-1}, u_{k-1}, 0) \tag{4.18}$$

y

$$\tilde{z}_k = h(\tilde{x}_k, 0) \tag{4.19}$$

Donde $\hat{x}_k$ es una estimación a posteriori del estado (desde un paso anterior). Para la estimación de un proceso no lineal y las relaciones de medición, se empieza por escribir nuevas ecuaciones que gobiernan las ecuaciones que linealizan una estimación acerca de Eq. 4.19) y Eq. 4.19.

$$x_k \approx \tilde{x}_k + A(x_{k-1} - \hat{x}_{k-1}) + W w_{k-1}, \tag{4.20}$$

$$z_k \approx \tilde{z}_k + H(x_k - \tilde{x}_k) + V v_k \tag{4.21}$$

Donde:

- $A$ es la matriz jacobiana de derivadas parciales de $f$ con respecto a $x$.
- $W$ es la matriz jacobiana de derivadas parciales de $f$ con respecto a $w$.
- $H$ es la matriz jacobiana de derivadas parciales de $h$ con respecto a $x$.
- $V$ es la matriz jacobiana de derivadas parciales de $h$ con respecto a $v$.

Definimos entonces una nueva notación para predecir el error:

$$\tilde{e}_{x_k} \approx A(x_{k-1} - \hat{x}_{k-1}) + \epsilon_k \tag{4.22}$$

$$\tilde{e}_{z_k} \approx H\tilde{e}_{x_k} + \eta_k \tag{4.23}$$

Donde $\epsilon_k$ y $\eta_k$ representan las nuevas variables aleatorias independientes que tienen media cero y matrices de covarianza $WQW^T$ y $VRV^T$ respectivamente. Para obtener las estimaciones a posteriori del estado desde el proceso original no lineal se tiene que:

$$\hat{x}_k = \tilde{x}_k + \hat{e}_k \tag{4.24}$$

Dado que el valor predicho de $\hat{e}_k$ es cero, la ecuación de filtro Kalman para estimar $\hat{e}_k$ es:

$$\hat{e}_k = K_k \tilde{e}_{z_k} \tag{4.25}$$

así:

$$\hat{x}_k = \tilde{x}_k + K_k(z_k - \tilde{z}_k) \tag{4.26}$$

La ecuación Eq. 4.26) puede utilizarse para la actualización de la medición en el filtro de Kalman extendido. El conjunto completo de ecuaciones EKF se muestra a continuación, cabe anotar que se coloca el subíndice $k$ a la jacobianos A, W, H, y V, para indicar que son diferentes en (y por lo tanto, debe ser recalculado en) cada paso de tiempo.

Ecuaciones de actualización EKF

$$\hat{x}_k^- = f(\hat{x}_{k-1}, \hat{u}_{k-1}, 0) \tag{4.27}$$

$$P_k^- = A_k P_{k-1} A^T + W_k Q_{k-1} W_k^T \tag{4.28}$$

Ecuaciones de actualización de medición EKF

$$K_k = P_k^- H^T (H P_k^- H^T + V_k R_k V_k^T)^{-1} \tag{4.29}$$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - h(\hat{x}_k^-, 0)) \tag{4.30}$$

$$P_k = (I - K_k H) P_k^- \tag{4.31}$$

Como en el filtro de Kalman básico, las ecuaciones de actualización de medición corrigen las estimaciones de estado y covarianza con la medida $z_k$. Una característica importante de la EKF es que el jacobiano $H_k$ en la ecuación de la ganancia de Kalman sirve para propagar correctamente o magnificar sólo el componente correspondiente de la información de medición.

## 4.3. Implementación del Filtro de Kalman Extendido

La EKF es una modificación del filtro de Kalman lineal (LKF) que puede manejar dinámica no lineal y ecuaciones de medición no lineal [29]. El EKF es un estimador óptimo que combina de forma recursiva datos con ruido provenientes sensores con un modelo de la dinámica del sistema. En la implementación del Filtro de Kalman se ha determinado la ubicación del robot como el vector de estado $s \in R^3$ el cual está conformado por las coordenadas cartesianas y su orientación:

$$s = (x_{robot}, y_{robot}, \phi_{robot}) \tag{4.32}$$

Y el estado de control como $u = (\bigtriangledown x_l, \bigtriangledown y_l)$ donde $\bigtriangledown x_l$ y $\bigtriangledown y_l$ representan las lecturas de odometría. El modelo dinámico del sistema (vehículo) se define en forma generalizada como:

$$x_r = f(x_k, u_k) + q_k \tag{4.33}$$

Donde $f$ es un modelo no lineal de la dinámica de un móvil con configuración diferencial, cuyas variables corresponden al estado actual $x_k$ y al de control $u_k$, $q_k$ representa el ruido del proceso cuyos parámetros son: media cero y covarianza $Q_k$, $k$ simboliza un índice de tiempo discreto, el modelo será actualizado en intervalos de tiempo $t = kT$ para un periodo $T$. Con estas especificaciones se define la función $f$ como:

$$f_x = x_{k+1} = x_k + \bigtriangledown x_l \cos(\phi_k) - \bigtriangledown y_l \sin(\phi_k) \tag{4.34}$$

$$f_y = y_{k+1} = y_k + \bigtriangledown x_l \sin(\phi_k) + \bigtriangledown y_l \cos(\phi_k) \tag{4.35}$$

$$f_\phi = \phi_{k+1} = \phi_k + \bigtriangledown \phi_l \tag{4.36}$$

Donde $\bigtriangledown \phi_l = \arctan \frac{\bigtriangledown x_l}{\bigtriangledown y_l}$ Para el desarrollo de las ecuaciones pertenecientes al filtro de Kalman se han de tener en cuenta las siguientes notaciones: $x_{k|k}$ Es la estimación

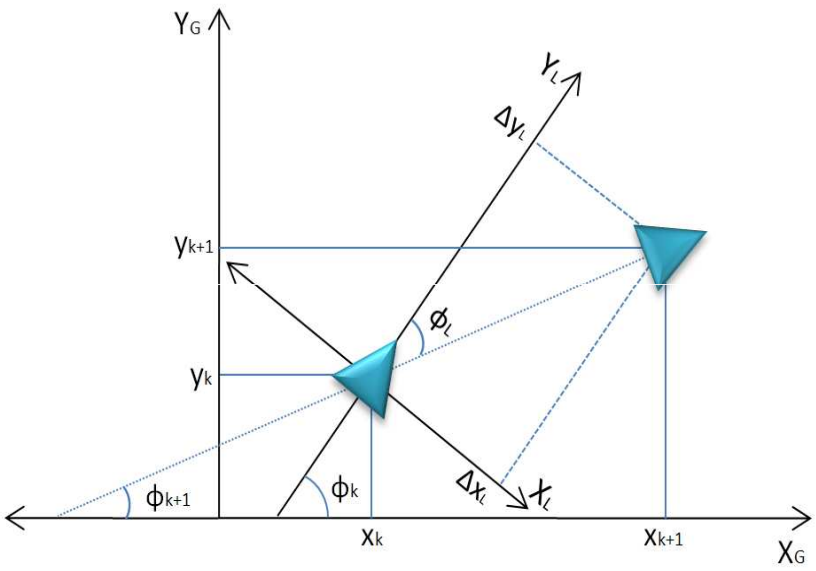


Figura 4.1: Parámetros del filtro del Kalman

del estado $x$ en el instante $k$ dando las medidas disponibles (estimación). $x_{k+1|k}$ Es la estimación de $x$ en el instante $k+1$ dado las $k$ primeras medidas (predicción). El estado del robot está definido por:

$$x_{k|k} = X_k + \eta_k \tag{4.37}$$

El vector de estado corresponde a un vector columna $x = [x_r^T, x_1^T, x_2^T, x_3^T, \ldots, x_N^T]^T$ que contiene la posición estimada actual del robot en coordenadas cartesianas $x_r = [x_r, y_r, \theta_r]^T$, junto con las posiciones de todos los hitos observados $x_i = [\theta_i, \rho_i]^T$ Por tanto, este vector tendrá dimensiones $(3 + 2n)$x1, siendo $n$ el número de hitos. $X$ es denominado como el vector de estados y $\eta$ es el error en la estimación.

La matriz de covarianza $P_{k|k} = E[\eta_k \eta_k^T]$ contiene las relaciones de la posición del robot consigo mismo, con las posiciones de los hitos, las relaciones entre las posiciones de los hitos y las relaciones cruzadas entre hitos.

$$P = \begin{bmatrix} P^{rr} & P^{ri} & \ldots & P^{rn} \\ P^{ir} & P^{ii} & \ldots & P^{in} \\ \vdots & & & \vdots \\ P^{nr} & P^{ni} & \ldots & P^{nn} \end{bmatrix} \tag{4.38}$$

La primera submatriz $P^{rr}$ es la covarianza de la posición del robot por tanto será una matriz 3x3. Las tres primeras filas de la matriz, representadas por $Pri$ , corresponden a las correlaciones entre el robot y los diferentes hitos, al igual que las tres primeras columnas que son las correlaciones de los hitos con el estado del robot. La primera

será una matriz 3x2$n$ mientras que la segunda será 2$n$x3, donde $n$ es el número de hitos del vector de estados. Las matrices restantes $P^{ii}$, $P^{in}$ y $P^{ni}$ que constituyen una matriz de 2$n$x2$n$ es simetrica y se conforma de la covarianza de los hitos consigo mismos, y la covarianza cruzada entre hitos.

### 4.3.1. Filtro de Kalman Extendido en el Proceso del SLAM

El robot y el mapa se expresan en un único vector de estados $X$ con su correspondiente estimación del error de covarianza $P$ en cada momento. El EKF se utiliza para estimar el estado $X$ y la covarianza $P$ dada la medición $Z$. La estimación se origina con un proceso de predicción, ocasionado por el desplazamiento del robot móvil, y un paso de actualización, que ocurre en el momento que los hitos son re-observados. En el momento en que un hito nuevo es detectado, este debe ser adicionado al vector de estados y a la matriz de covarianza respectivamente. Así el progreso de creación del mapa se fracciona en tres partes: desplazamiento del vehículo (reajuste de odometría), composición de los hitos y re-observación de los mismos. La figura 4.2 muestra el diagrama de pasos general que se seguirá:

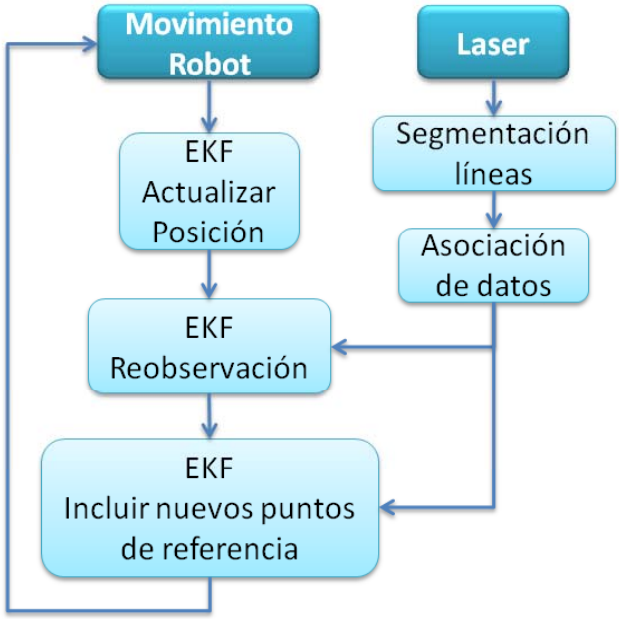


Figura 4.2: Proceso de SLAM

Cada que se produzca un movimiento del robot (cambio en la odometría) se realiza el cálculo de la nueva posición del mismo, actualizando las coordenadas $x$, $y$ y $\theta$ haciendo uso de las señales de control generadas para alcanzar el estado actual. Con las lecturas proporcionadas por el sensor laser, se extraen las características del entorno, las cuales son procesadas junto con las características anteriormente detectadas, aquellas

que son asociadas a una característica pasada corregirán la posición actual del robot. Conociendo la estimación de las coordenadas de la posición anterior y el movimiento producido, se realiza el cálculo de la discrepancia entre la posición observada de los hitos y la estimada presumiendo que el robot se halle efectivamente en el lugar que enseña la odometría, para realizar el ajusta el estado actual del robot. Las características que se han observado por primera vez se incluyen en el conjunto de hitos reconocidos para futuras iteraciones y las reobservadas se ajustan con los nuevos parámetros.

#### 4.3.1.1. Predicción

**Desplazamiento del vehículo**

El robot móvil se mueve el estado estimado del mismo en el instante $k+1$ está basado la expectativa en el modelo del estado de transición por:

$$x_{r_{k+1|k}} = E[f(X_k, u_k)] \approx f(x_{r_{k|k}}, u_k) \tag{4.39}$$

Como suponemos que el ruido que afecta el proceso tiene una media igual a cero, el valor se hace cero al evaluarlo con el valor estimado $x_r$ para hallar $x_{r_{k|k}}$. La covarianza $P^{rr}_{k|k}$ se propaga por el modelo lineal del estado de la transición del EKF proporcionando $P^{rr}_{k|k}$ :

$$P^{rr}_{(k+1|k)} = J_x P^{rr}_{(k|k)} J_x^T + Q_k \tag{4.40}$$

Donde $Q_k = E[q_k q_k^T]$, $J_x$ es el jacobiano de $f$ con respecto a $x$. Este procedimiento brinda una posición estimada del vehículo basada únicamente en el modelo del robot y los valores de odometría. El jacobiano del modelo de predicción está estrictamente relacionado con el modelo de predicción. Este precisa como predecir la posición del robot desde de la posición anterior.

$$f = \begin{bmatrix} x_R + \bigtriangledown x_l \cos(\theta_R) - \bigtriangledown y_l \sin(\theta_R) \\ y_R + \bigtriangledown x_l \sin(\theta_R) + \bigtriangledown y_l \cos(\theta_R) \\ \theta_R + \bigtriangledown \theta_l \end{bmatrix} \tag{4.41}$$

Donde $x_R$ y $y_R$ son las coordenadas del robot, $\theta_R$ es la rotación, $\bigtriangledown x_l$ $\bigtriangledown y_l$ $\bigtriangledown \theta_l$ es el cambio de las coordenadas respectivamente. Asumiendo que se puede acceder directamente a los incrementos a través de señales del robot y que se va a trabajar con un modelo linealizado, el jacobiano del modelo de predicción quedara como:

$$J_x = \begin{bmatrix} 1 & 0 & -\bigtriangledown x \sin(\theta_R) - \bigtriangledown y \cos(\theta_R) \\ 0 & 1 & \bigtriangledown x \cos(\theta_R) - \bigtriangledown y \sin(\theta_R) \\ 0 & 0 & 1 \end{bmatrix} \tag{4.42}$$

Se ha asumido que el proceso tiene un ruido gaussiano proporcional a los controles dados al robot. La matriz $Q$ se calcula como $Q = J_{\bigtriangledown x} C J^T_{\bigtriangledown x}$ donde $J_{\bigtriangledown x}$ es la matriz jacobiana de derivadas parciales de $f$ con respecto $\bigtriangledown x$ :

$$J_{\bigtriangledown x} = \begin{bmatrix} \cos(\theta_R) & -\sin(\theta_R) & 0 \\ \sin(\theta_R) & \cos(\theta_R) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.43}$$

La matriz $C$ es una representación de cómo de exacta es la odometría. El valor suele estimarse realizando experimentos y comprobando cual es el valor que mejor se ajusta a los resultados obtenidos.

También es menester actualizar las correlaciones cruzadas robot-hito

$$P_{(k+1|k)^r i} = J_x P_{(k|k)^r i} \tag{4.44}$$

Sin embargo no es necesario renovar las correlaciones cruzadas entre los hitos, debido que la ubicación del hito en el plano no es afectada por cambios en el estado del robot.

**Predicción de Observaciones**

En este punto queremos saber como serían las rectas si se observan desde la posición actual del robot, para luego compararlas con las observaciones actuales. En el instante $k$ se tiene en vector de estado de la forma $X = [x_r^T, x_1^T, x_2^T, x_3^T, \ldots, x_N^T]^T$, donde $x_r$ es la posición actual del robot y $x_i$ las posiciones de todos los hitos observados en la referencia global. $h$ o la predicción es una funcion que depende del vector de estado $X_k$, donde:

$$h = \ominus X_{WR_k} * \oplus X_{WFn_k} \tag{4.45}$$

Para trasladar la referencia global del robot al punto de referencia local se hace necesario calcular un vector de transformacion inversa, el cual viene dado por:

$$\ominus X_{WR_k} = \begin{bmatrix} -x_R \cos(\theta_R) - y_R \sin(\theta_R) \\ x_R \sin(\theta_R) - y_R \cos(\theta_R) \\ -\theta_R \end{bmatrix} \tag{4.46}$$

Luego se calcula la transformación de las hitos haciendo uso del modelo de medida, que define como obtener la estimación del rango y de la orientación para un punto de referencia.

$$\oplus X_{WFn_k} = \begin{bmatrix} \rho_L + x_{WR_k} \cos(\theta_{WR_k} + \theta_L) + y_{WR_k} \sin(\theta_{WR_k} + \theta_L) \\ \theta_{WR_k} + \theta_L \end{bmatrix} \tag{4.47}$$

Donde $\rho_L$ y $\theta_L$ representan la posición almacenada del hito, $x_{WR_k}$ , $y_{WR_k}$ y $\theta_{WR_k}$ simbolizan la posición del robot segun la transformación inversa obtenida anteriormente. Esto proporcionará una estimación para la posición del punto de referencia visto desde la nueva posición del robot, dando como resultado una función de predicción $h$ de la siguiente forma:

$$h = \begin{bmatrix} \rho_L - x_R \cos(\theta_L) - y_R \sin(\theta_L) \\ \theta_L - \theta_R \end{bmatrix} \tag{4.48}$$

Dado que la predicción se realiza a travez de una funcion no líneal debido a las transformaciones presentadas, y dado que tambien se debe obtener la matriz de covarianza, se deber realizar una linealización al rededor del punto $x_k$, para propagar de esta manera la matriz $P$ a travez de la trasformación. La matriz de Jacobianos $H_x$ de la predicción es compuesta por $H_r$ y $H_f$ donde:

$$H_r = \frac{\partial h}{\partial x_R} = \begin{bmatrix} -\cos(\theta_L) & -\sin(\theta_L) & 0) \\ 0 & 0 & -1 \end{bmatrix} \tag{4.49}$$

$$H_f = \frac{\partial h}{\partial x_L} = \begin{bmatrix} 1 & x_R \sin(\theta_L) - y_R \cos(\theta_L) \\ 0 & 1 \end{bmatrix} \tag{4.50}$$

Para todas las predicciones, tenemos una matriz final de jacobianos $H$ de la forma:

$$H = \begin{bmatrix} H_{r1} & H_{f1} & 0 & \dots & 0 \\ H_{r2} & 0 & H_{f2} & \dots & 0 \\ \vdots & & & \vdots & \\ H_{2n} & 0 & 0 & \dots & H_{fn} \end{bmatrix} \tag{4.51}$$

Propagando la matriz de jacobianos en la predicción, así:

$$S = H P_k H^T \tag{4.52}$$

#### 4.3.1.2. Re-observación de los hitos (Actualización)

Cuando un hito presente en el vector de estados es observado se utiliza el paso de actualización del EKF para actualizar el estado del mapa incluyendo la posición del robot. La ecuación del modelo de observación para el punto de referencia $i$ (en forma de línea) tiene la forma

$$z_{ik+1} = \begin{bmatrix} \rho_{ik+1} \\ \theta_{ik+1} \end{bmatrix} = \begin{bmatrix} x_r cos(\phi_r + \theta_i) + y_r sin(\phi_r + \theta_i) + \rho_i \\ \phi_r + \theta_i \end{bmatrix} + n_{zi} \tag{4.53}$$

$$= h_i(x_{k+1|k}) + n_{zi} \tag{4.54}$$

Se considera que el ruido de todo el proceso $n_{zi}$ es un ruido blanco gaussiano de covarianza $R_i$. Si los $N$ hitos son observados el modelo de la observación estará dado por las siguientes ecuaciones:

$$z_{k+1} = \begin{bmatrix} z_{ik+1} \\ \vdots \\ z_{Nk+1} \end{bmatrix} \tag{4.55}$$

$$h = \begin{bmatrix} h_1 \\ \vdots \\ h_N \end{bmatrix} \tag{4.56}$$

$$R_{k+1} = \begin{bmatrix} R_1 & \dots & 0 \\ \vdots & \dots & \vdots \\ 0 & \dots & R_N \end{bmatrix} \tag{4.57}$$

Denotando al jacobiano de $h$ como $H$, las ecuaciones del proceso de actualización son:

$$x_{k+1|k+1} = x_{k+1|k} + K_{k+1}(z_{k+1} - h(x_{k+1|k})) \tag{4.58}$$

$$P_{k+1|k+1} = (I - K_{k+1}H_x)P_{k+1|k} \tag{4.59}$$

Donde $K_{k+1}$ es la ganancia de Kalman que se calcula como

$$K_{k+1} = P_{k+1|k+1}H_x^T(H_xP_{k+1|k+1}H_x^T + R_{k+1})^{-1} \tag{4.60}$$

La ganancia de Kalman o factor de mezcla establece la cantidad de influencia del error entre nuestra estimación y la medida, por lo que le proporcionará un mayor o menor peso a la odometría en relación con los hitos observados. Es decir, si se parte de la base de que el sistema de medida para observar los hitos es inferior en precisión que el sistema de odometría, se valorará más la estimación de la posición obtenida por la odometría que la deducida a partir de los hitos observados. Por el contrario si el sistema de observación de los hitos es superior al utilizado para adquirir el cambio en la odometría, se considerará más la posición estimada por los hitos. Usando la predicción de la posición actual es posible estimar dónde deberían encontrarse los hitos. Normalmente hay ciertas discrepancias, a la diferencia entre la posición predicha del robot y la posición real se le conoce como innovación.

#### 4.3.1.3. Integración del nuevo hito

Cuando un nuevo hito $L_{new} = [\rho, \theta]$ es visto y validado el estado del mismo $x_{N+1}$ se incorpora al vector de estado del sistema.

$$x_{N+1} = m(x_{k+1|k+1}, L_{new}) = \begin{bmatrix} (x_{r_{k+1|k+1}} cos(\Phi_{r_{k+1|k+1}} + \theta) + y_{r_{k+1|k+1}} sin(\Phi_{r_{k+1|k+1}} + \theta) + \rho \\ \Phi_{r_{k+1|k+1}} + \theta \end{bmatrix} \tag{4.61}$$

$$x_{k+1|k+1} \leftarrow \begin{bmatrix} x_{k+1|k+1} \\ x_{N+1} \end{bmatrix} \tag{4.62}$$

Seguidamente de completa la matriz de covarianza P para qué comprenda al nuevo hito, para su realización se tendrá que calcular siguientes sub-matrices que son la covarianza del hito N +1, la covarianza cruzada hito-robot y la covarianza cruzada hito-hito [16]

$$P^{N+1|N+1} = J_{x_r}P^{rr}_{k+1|k+1}J^T_{x_r} + J_zRJ^T_z \tag{4.63}$$

$$P^{rN+1} = P^{N+1r^T} = P^{rr}_{k+1|k+1}J^T_{x_r} \tag{4.64}$$

$$P^{N+1i}_{k+1|k+1} = P^{iN+1}_{k+1|k+1}{}^{T} = J_{x_r} P^{ri}_{k+1|k+1} \tag{4.65}$$

Las matrices $J_{x_r}$ y $J_z$ son los jacobianos de $m$ con respecto al estado del robot $x_r$ y al punto de referencia $L_{new}$.

$$J_{xr} = \begin{bmatrix} cos(\phi_1 + \theta) & sin(\phi_1 + \theta) & -x_1 sin(\phi_1 + \theta) + y_1 cos(\phi_1 + \theta) \\ 0 & 0 & 1 \end{bmatrix} \tag{4.66}$$

$J_{xr}$ muestra cuánto cambia el rango y la orientación de las líneas (hitos) cuando varian $x_R$ , $y_R$ y $\theta_R$ . El primer elemento de la primera fila es el cambio del rango con respecto a $x_r$; el segundo, respecto a $y_R$ y el tercer valor es con respecto a $\theta_R$. La segunda fila proporciona la misma información pero con relación a la orientación.

$$J_z = \begin{bmatrix} 1 & -x_1 sin(\phi_1 + \theta) + y_1 cos(\phi_1 + \theta) \\ 0 & 1 \end{bmatrix} \tag{4.67}$$

### 4.3.2. Complejidad Computacional

Asumiendo que el vehículo está equipado con un sensor limitado en rango y orientación, la cantidad de mediciones obtenidas en cualquier ubicación será más o menos constante. Supongamos que en algún paso $k$:

- El mapa contiene $n$ características.
- El sensor proporciona $m$ mediciones.
- $r$ de los cuales corresponden a características re-observadas.
- $s = m - r$ corresponden a nuevas características.

Esto corresponde a una trayectoria de exploración, donde el tamaño del mapa es proporcional al número de pasos que se han llevado a cabo.

#### 4.3.2.1. Coste computacional por paso

La complejidad computacional que se lleva a cabo el ciclo de movimiento, sensado y actualización del proceso de EKF en paso $k$ involucra en el cálculo del mapa predicho, que requiere la obtención también el cálculo de los correspondientes jacobianos, y la actualización mapa, que requiere el cálculo de la correspondiente jacobiano, la matriz de ganancia de Kalman, así como la innovación y su covarianza. Para el cálculo de la matriz de la covarianza de innovacion de $rxr$ se requiere $rn^2 + r^2n$ multiplicaciones y $rn^2 + r^2n + r^2$ sumas, es decir, $O(n^2)$ operaciones. Pero dado que la matriz $Hk$ tiene componentes iguales a cero, se representa co un tamaño efectivo de $rxc$, así, el cálculo requiere $rcn + r^2c$ multiplicaciones y $rcn + r^2n + r^2$ sumas, es decir, $O(n)$ operaciones. El mismo análisis lleva a la conclusión de que el costo de la computación tanto la covarianza de predicción y la matriz de ganancia de Kalman es $O(n)$, y que el mayor costo de una actualización de EKF SLAM es el cálculo de la matriz de covarianza, que

es $O(n2)$. Por lo tanto, el coste computacional por paso de EKF es cuadrática en el tamaño del mapa [40]:

$$C_{EKF,k} = O(n^2) \tag{4.68}$$

#### 4.3.2.2. Coste computacional total

En una trayectoria de exploración un número constante de las nuevas características se añaden al mapa a cada paso. Así, para mapear un entorno de tamaño $n$, $n/s$ pasos son necesarios. El costo total de EKF SLAM será entonces:

$$C_{EKF,k} = \sum_{k=1}^{n/s} O((ks)^2) \tag{4.69}$$

$$C_{EKF,k} = \sum_{k=1}^{n/s} O(k^2) \tag{4.70}$$

La suma de potencia cuadrado se sabe que es:

$$\sum_{k=1}^{n} k^2 = (1/6)n(n+1)(2n+1) \tag{4.71}$$

Por lo tanto, el costo total de EKF SLAM es cúbica:

$$C_{EKF,k} = O(\frac{(n/s)(n/s+1)(2n/s+1)}{6}) \tag{4.72}$$

$$C_{EKF,k} = O(\frac{n^3}{s^3}) \tag{4.73}$$

$$C_{EKF,k} = O(n^3) \tag{4.74}$$

# Capítulo 5

# PRUEBA Y ANALISIS DE RESULTADOS

Los resultados de los distintos experimentos realizados con el algoritmo propuesto de mapeo y localización simultánea son expuestos en este capítulo. Las pruebas se hicieron para dos ambientes diferentes, el primero de ellos corresponde a entornos generados artificialmente a través de software de simulación, donde el robot (simulado) navega en condiciones ideales, para obtener de esta manera la posición real del robot, de forma paralela se supone el estado de la odometría agregando ruido a los movimientos ejecutados por el vehículo. El segundo ambiente atañe a entornos comunes de interiores, donde el robot realizó recorridos de diferentes distancias. Para ambos ambientes se registraron los datos tanto de los sensores laser como de los ultrasonidos para realizar la extracción de características del mapa, esencial en el proceso de SLAM. Adicionalmente se presenta una aplicación en la aérea de dibujo en ingeniería, correspondiente a levantamiento de planos de ambientes estructurados, desarrollado en el presente proyecto.

## 5.1. Medio Ambiente Artificialmente Modelado

Esta sección muestra un conjunto de resultados obtenidos con mapas simulados, que han sido construidos con el propósito de poner a prueba el EKF-SLAM, ya que para comparar los resultados con la realidad del medio, es esencial que toda la información se encuentre en el mismo marco de referencia. Sin embargo, en experimentos de laboratorio el marco de referencia del SLAM, la odometría y el mapa verdadero no están por lo general, en el mismo marco. Es por esto que se realiza una simulación ideal donde los valores de odometría y laser son exactos, datos que luego son alterados con la adición de ruido, obteniendo de esta forma, dos conjuntos de datos (real y obtenido de sensores con ruido) en el mismo marco, cabe anotar que el conjunto de datos con presencia de ruido es procesado por el algoritmo de EKF-SLAM, cuyo resultado puede ser comparado con la realidad del mapa y recorrido.

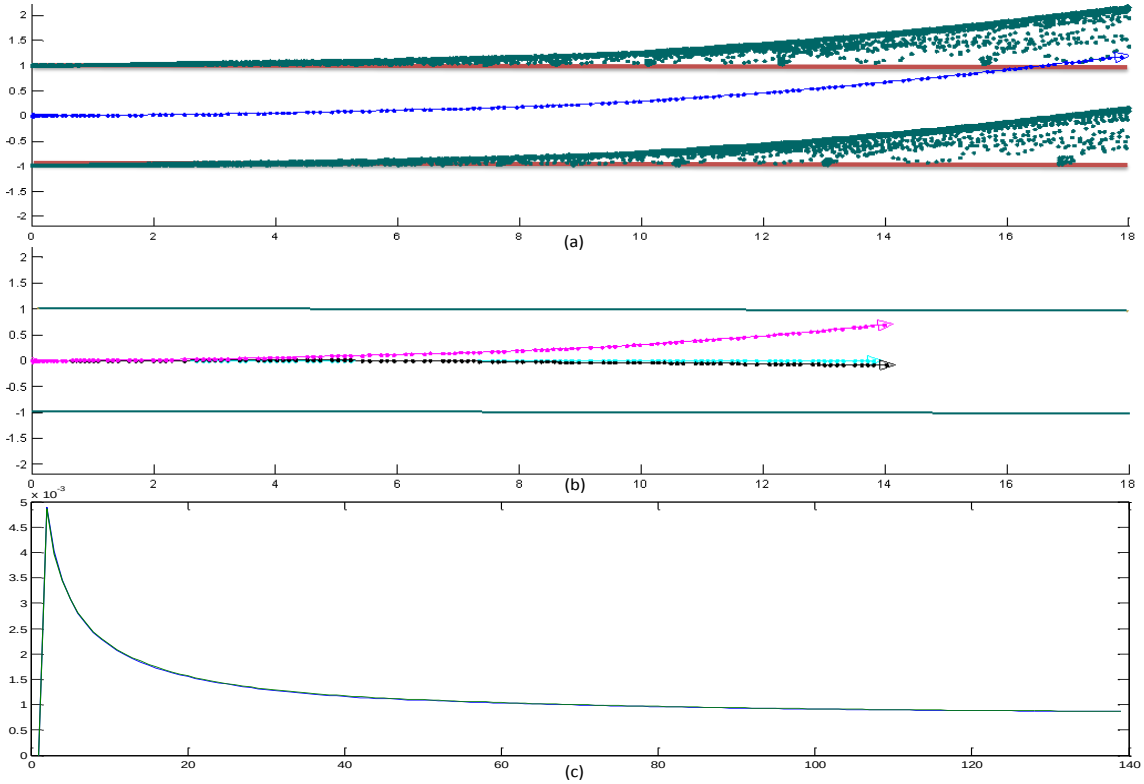


Figura 5.1: Resultado de la ejecución del EKF-SLAM en un pasillo, utilizando un escaner Laser.

La Fig(5.1) y Fig(5.2) con tres sub-figuras cada una es el resultado del EKF-SLAM se ejecutado en un mapa artificial, en este caso un pasillo de 18 metros de largo y 2 de ancho, las sub-Figuras Fig(5.1)Fig(5.2(a) muestran el mapa real contrastado con los datos adquiridos a partir de la odometría y los sensores de laser y ultrasonido (afectados por ruido). Las segundas sub-Figuras Fig(5.1)Fig(5.2(b) se han obtenido después de 138 iteraciones del algoritmo, donde se muestra el estado del robot real (azul), estimado (negro) y según odometría (magenta), junto con el mapa construido, apreciando de manera gráfica la corrección del error de odometría. Las terceras sub-Figuras Fig(5.1)Fig(5.2(c) representan la evolución de la incertidumbre de las características conforme se realizan las iteraciones del SLAM, se observa cómo, a medida que el proceso continua, y las características del mapa (líneas) son re-observados en cada paso y su incertidumbre disminuye, esta disminución es más significativa para el sensor laser que para el de ultrasonido, al ser este tipo de sensor de mayor precisión.

Para el mapa construido con asistencia de sensores de ultrasonido las características del entorno comienzan a ser evidentes después de una serie de pasos (30 para esta prueba), ya que en cada instante cuando se escanea el medio se obtiene como máximo un número de 8 puntos de distancia a obstáculos, los cuales deben ser pasados por un filtro que indique cuales se encuentran en el rango de detección, estos datos son

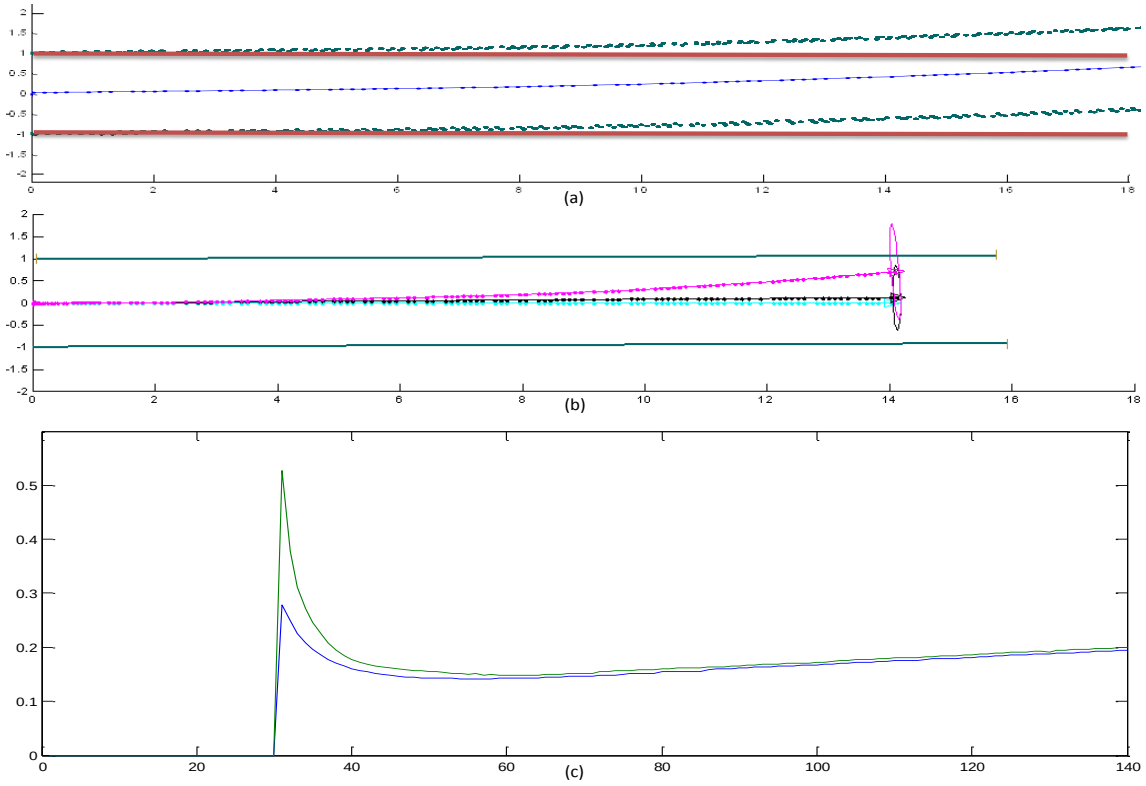


Figura 5.2: Resultado de la ejecución del EKF-SLAM en un pasillo, utilizando un cinturon de ultrasonidos.

insuficientes para asegurar la presencia de líneas rectas, y por lo tanto es menester ir acumulando sistematicamente las lecturas pasadas con las actuales, para que el algoritmo de segmentación detecte con esta información correctamente los puntos de referencia.

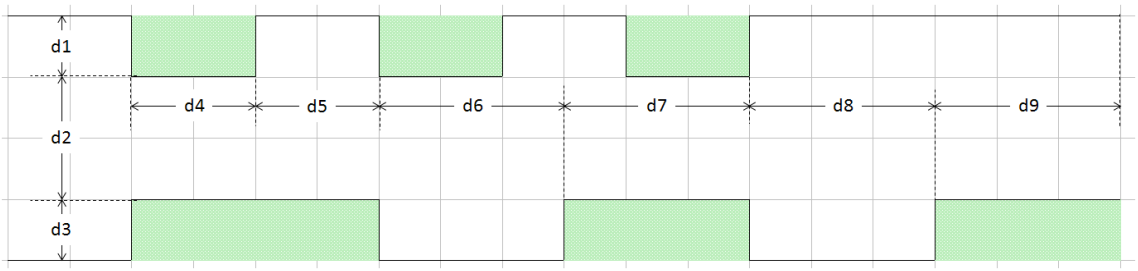


Figura 5.3: Mapa modelado en Mapper3

El entorno de la Fig(5.3) fue creado en el software de modelamiento de mapas Mapper3 (mobilerobots), con una configuración y medidas predispuestas $d$ (que se especifican en la misma figura ) con el ánimo de crear un mapa compuesto por líneas de

diferentes características en párametros y longitudes, poniendo a prueba tanto el algoritmo EKF-SLAM como el de segmentación, para cada caso. Al disponer las medidas del entorno se facilita la evaluación del resultado conseguido al final del recorrido.

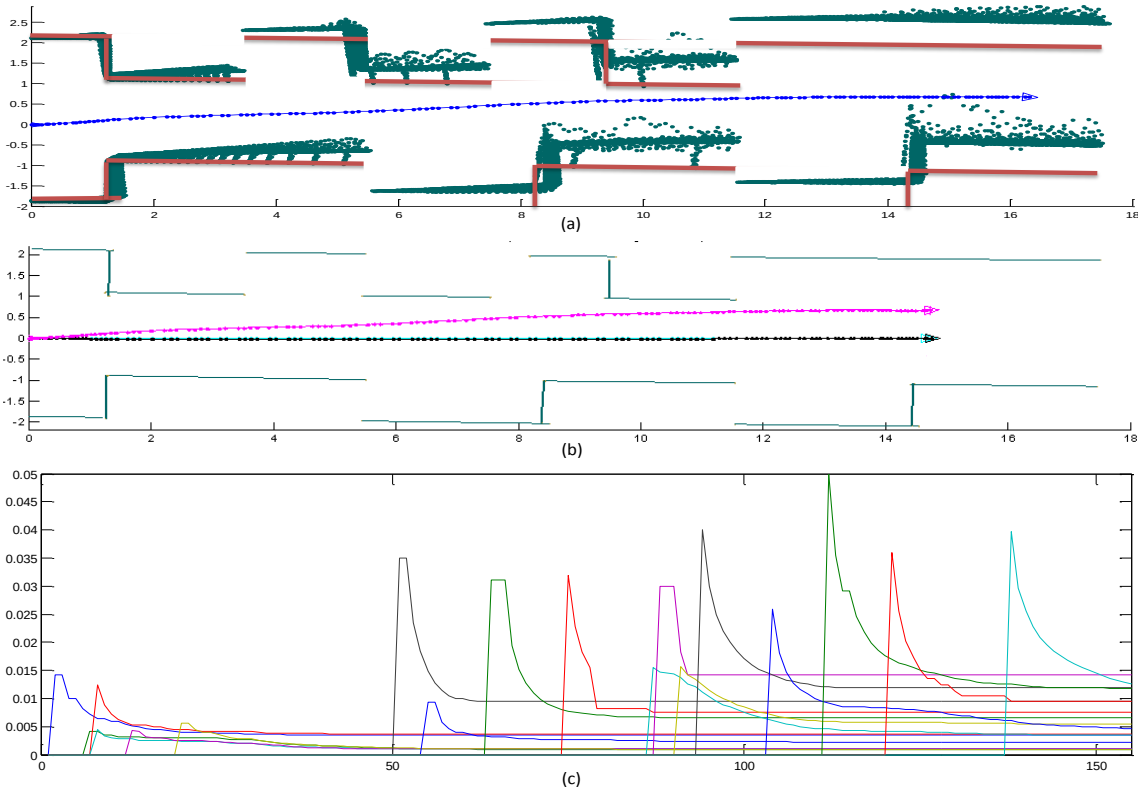


Figura 5.4: Resultado de la ejecución del EKF-SLAM para el mapa modelado.

Las Figuras Fig(5.4) Fig(5.5) con tres sub-figuras cada una es el resultado del EKF-SLAM para el mapa diseñado ,donde robot se ha simulado realizando un recorrido en línea recta, las sub-Figuras Fig(5.4)Fig(5.5)(a) muestran el mapa simulado en contraste con los datos de odometría y sensores afectados por ruido. Las segundas sub-Figuras Fig(5.4)Fig(5.5)(b) se obtuvieron después de 160 iteraciones del algoritmo, donde se muestra la ubicacion real del robot (azul), la posición estimada (negro) y según la odometría (magenta) para cada iteración, junto con el mapa que ha sido construido, por su parte las terceras sub-Figuras Fig(5.4)Fig(5.5)(c) muestran la evolución de la incertidumbre de las características de cada mapa. Con el fin de comparar numéricamente, tanto el mapa creado como la corrección de odometría realizados a través del SLAM-EKF, se definieron algunas medidas de distancia que describen la disposición del mapa, las cuales son confrontadas con aquellas obtenidas tras la ejecución del proceso de SLAM.

La Tabla 5.1 registra el valor las longitudes definidas (Fig.(5.3)) en metros del entorno registrado por el robot y las resultantes de la ejecución del algoritmo de EKF-SLAM con los diferentes sensores de proximidad, además de un par de columnas co-

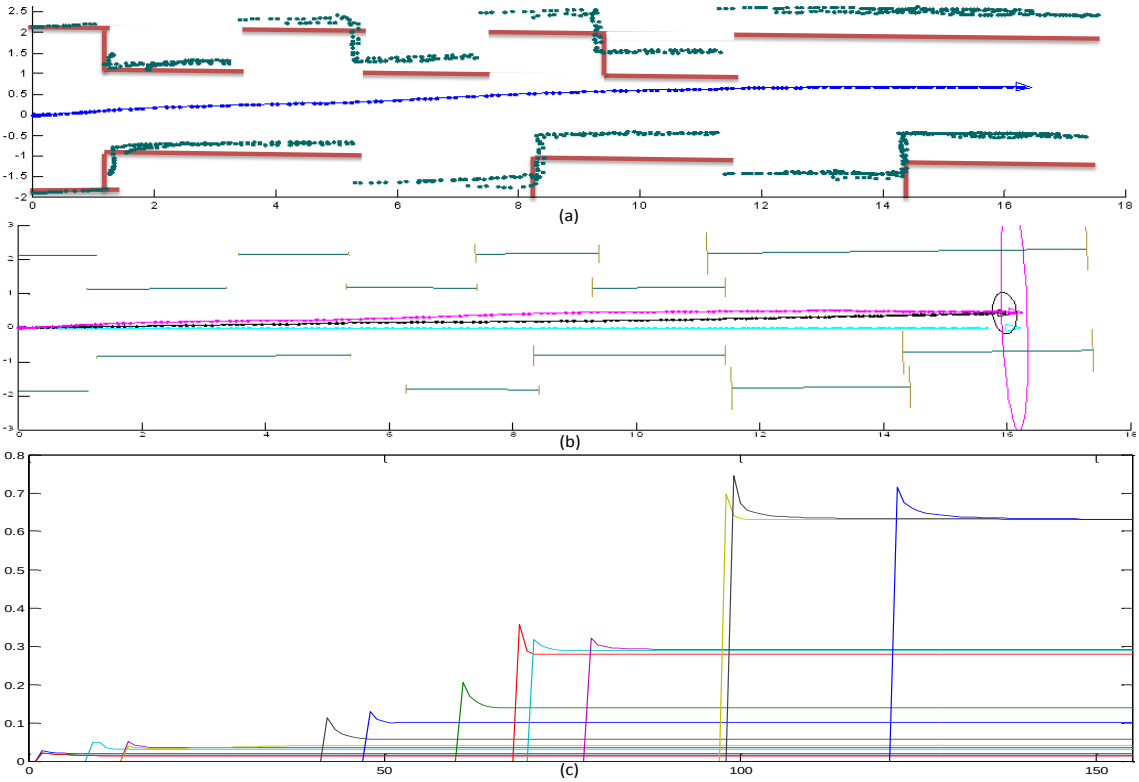


Figura 5.5: Resultado de la ejecución del EKF-SLAM para el mapa modelado.

| Distancia | SLAM Laser | SLAM US | Mapa Real | Error Laser % | Error US % |
|---|---|---|---|---|---|
| d1 | 1,0012 | 0,9884 | 1 | 0,12 % | 1,16 % |
| d2 | 1,9614 | 1,9845 | 2 | 1,93 % | 0,78 % |
| d3 | 0,9948 | 1,0050 | 1 | 0,52 % | 0,50 % |
| d4 | 2,0645 | 2,0532 | 2 | 3,22 % | 2,66 % |
| d5 | 1,9330 | 1,8858 | 2 | 3,35 % | 5,71 % |
| d6 | 3,0777 | 3,0523 | 3 | 2,59 % | 1,74 % |
| d7 | 3,0022 | 2.8945 | 3 | 0,07 % | 3,52 % |
| d8 | 3,0547 | 3,0902 | 3 | 1,82 % | 3,01 % |
| d9 | 2,9868 | 3.0739 | 3 | 0,44 % | 2,46 % |

Tabla 5.1: Comparación entre distancias características del mapa construido utilizando un escaner laser y sensores de ultrasonido.

rrespondientes a el error porcentual existente entre ellas. De igual manera las Figuras Fig(5.6) y Fig(5.7) enseñan las graficas del error de posición del robot móvil, realizando un contraste entre el dato de odometría y el estimado por el filtro (azul) respecto a la ubicación real (conocida) del robot móvil, evidenciando el error acumulativo del odómetro, y la corrección del mismo. La estimación de ubicación es indudablemente

mejor para el proceso en el que interviene el escáner laser como sensor de apoyo, las líneas detectadas a través de los datos de este sensor posee una menor incertidumbre debido a factores como el error de detección y la cantidad de datos que definen una recta, además que en todas las iteraciones del algoritmo los datos son suficientes para segmentar las características, evaluarlas y corregir el error de odometría.

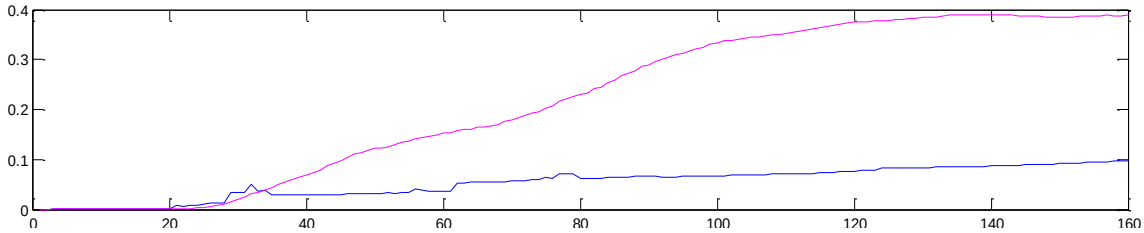


Figura 5.6: Error en cada paso iteración del proceso de EKF-SLAM (Laser), Error de odometría en magneta y error de la posición estimada en azul

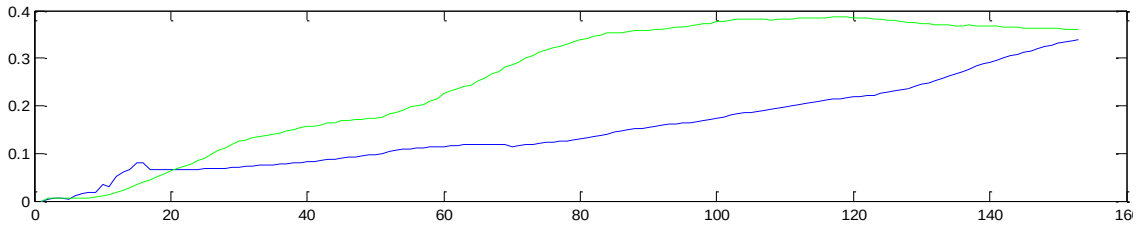


Figura 5.7: Error en cada paso iteración del proceso de EKF-SLAM (Ultrasonido), Error de odometría en verde y error de la posición estimada en azul

La tarea de detección se llama usualmente detección de bucles, mientras que la minimización del error es el cierre del bucle. En la Fig(5.8) se muestra el proceso del EKF-SLAM (laser) en un recorrido por un mapa donde las primeras características identificadas son reobservadas después de un largo recorrido sin ser asociadas con el entorno actual. Una característica de baja incertidumbre es una característica vista en al menos una de las iteraciones anteriores del algoritmo, cuya re-observación implica la reducción de la varianza del vector de estado. Esta propiedad, llamada de cierre del bucle se refiere al fenómeno que ocurre cuando el robot reconoce una zona del mapa ya observada tras haber generado una zona nueva del mapa, por lo cual debe corregirse el error acumulado en la trayectoria, esto es evidente en la Fig(5.9), donde se observa como los errores del mapa son reducidos abruptamente en el paso 179 (reobservación de las primeras líneas) debido a la covarianza entre las características y el robot.

## 5.2. Experimentos en entornos de interiores

La ejecuciones que se realizaron en el laboratorio de Robótica Móvil del grupo de investigación PSI de la Universidad de Valle, consistieron de navegaciones controladas

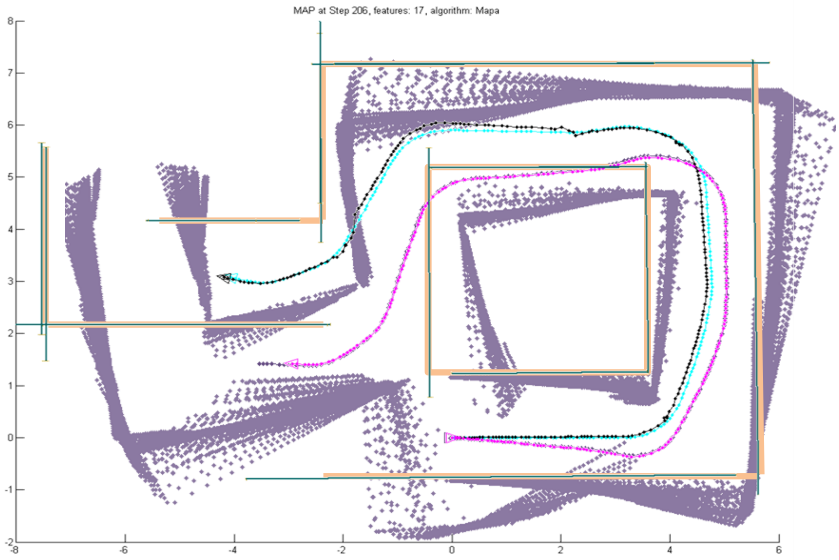


Figura 5.8: Resultado de EKF-SLAM (laser) para un mapa cerrado.

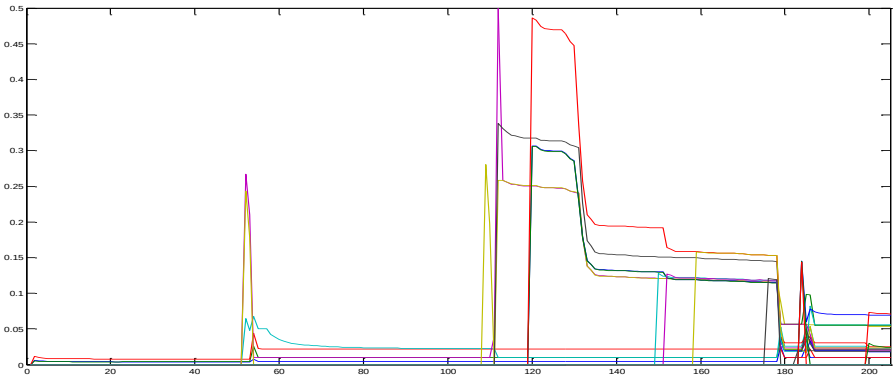

Figura 5.9: Error de las características del mapa cerrado.

del robot móvil Pioneer 3DX al rededor de un mapa ambientado con cajas, puertas y paredes, mientras se registraban los valores de odometría y de los sensores de proximidad (laser y ultrasonido), el promedio de la traslación entre dos capturas de datos consecutivas fue aproximadamente de 11 cm. La información de los sensores fue segmentada y parametrizada por los algoritmos implementados para luego ser procesada por el filtro de Kalman (algoritmo EKF-SLAM). Las figuras Fig(5.10) y Fig(5.11), exponen la comparación gráfica entre el mapa construido por el robot utilizando el algoritmo EKF-SLAM (con sensores laser y ultrasonido respectivamente) en verde y el mapa real en color naranja, además se muestra el recorrido según la odometría del vehículo y la ubicación estimada en cada iteración del algoritmo. Los cúmulos de puntos en el caso de la Fig(5.10) se deben a la presencia de error en la odometría del robot que afecta a su vez la orientación de las lecturas del escáner laser. En la figura Fig(5.12), se grafican los errores de las características del mapa construido con el escáner laser, comprobando

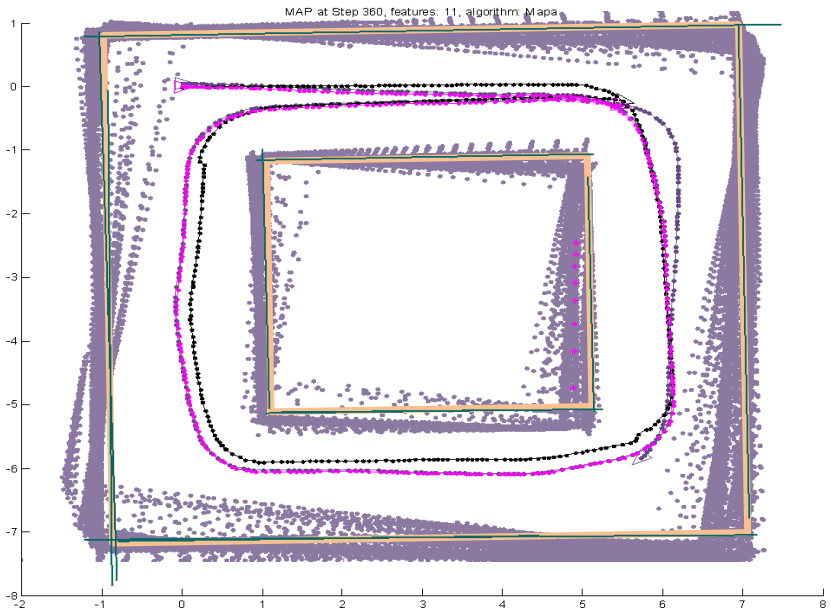


Figura 5.10: Resultado EKF-SLAM (laser) con datos reales.

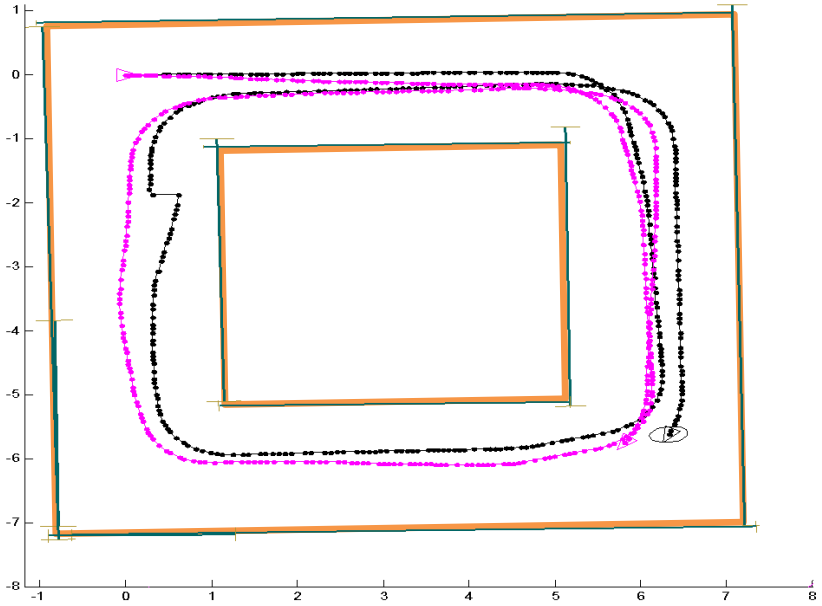

Figura 5.11: Resultado EKF-SLAM (ultrasonido) con datos reales.

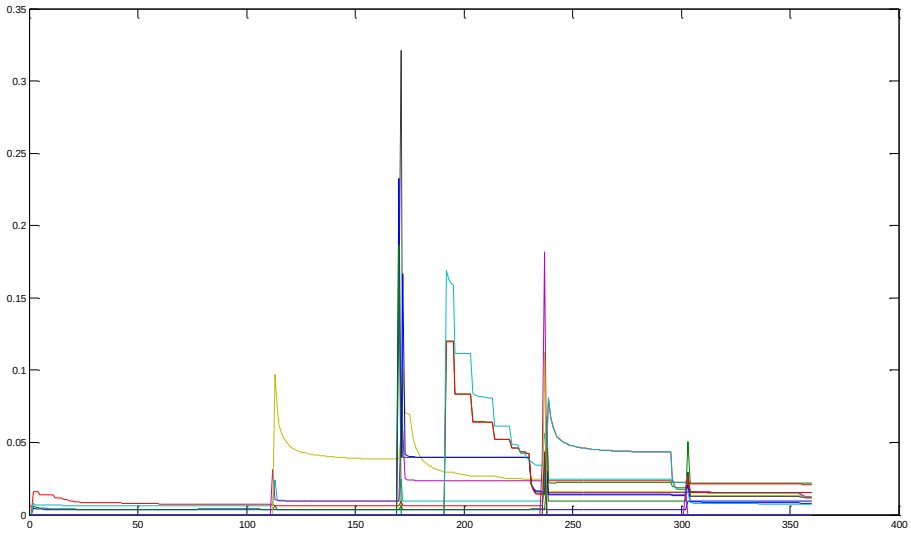


Figura 5.12: Error del mapa resultante del EKF-SLAM (laser)

| Distancia | SLAM Laser | SLAM US | Mapa Real | Error Laser % | Error US % |
|---|---|---|---|---|---|
| d1 | 2,9822 | 2,9839 | 3 | 0,593 % | 0,54 % |
| d2 | 0,9998 | 1,0110 | 1 | 0,020 % | 1,10 % |
| d3 | 1,9824 | 1,9730 | 2 | 0,880 % | 1,35 % |
| d4 | 5,9677 | - | 6 | 0,538 % | - % |
| d5 | 8,0028 | 8,1233 | 8 | 0,035 % | 1,54 % |
| d6 | 1,9899 | 1,8807 | 2 | 0,505 % | 5,97 % |
| d7 | 0,9639 | 0,9789 | 1 | 3,610 % | 2,11 % |

Tabla 5.2: Comparación entre distancias características de un mapa real utilizando sensor laser.

el comportamiento de disminución de la incertidumbre cuando existe reobservación de una característica y la propiedad de cierre de bucle para un mapa real.

Al igual que como se realizó con las pruebas de simulación, se estableció un mapa real (Fig.(5.14)) con medidas conocidas, por donde navegó el robot, registrando el entorno con el escáner laser y los sensores de ultrasonido, al mismo tiempo que se obtenía los datos de odometría. La información de los sensores de proximidad de segmentaban mientras se procesaban los datos a través de algoritmo EKF-SLAM, creando el mapa del entorno. El mapa resultante compuesto de puntos extremos de líneas, fue dibujado y acotado como se muestra en la figura Fig(5.14) exponiendo el entorno construido usando un escáner laser. Las distancias establecidas para comparación se extrajeron de los mapas resultantes y se presentan en la tabla 5.2.

El error es notablemente mayor en el proceso de EKF-SLAM con sensores de Ultrasonido, esto se debe a varios factores, al inicio del recorrido los datos para la extracción de características del mapa son insuficientes para realizar una detección confiable de

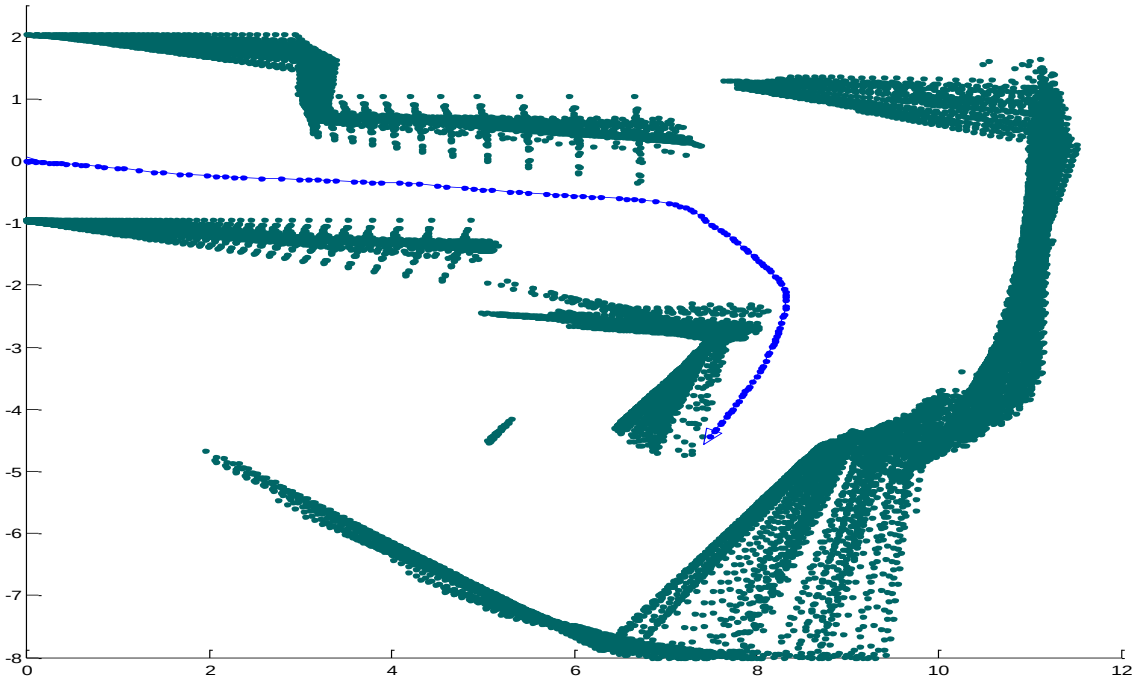


Figura 5.13: Mapa según los datos sin procesar de los sensores.

alguna línea del entorno, por lo tanto la odometría estimada no sufrirá corrección hasta que se comiencen a observar los puntos de referencia, otro causante del error en esta prueba es la naturaleza del sensor, este sensor posee un ruido significativo en comparación con el escáner laser, los datos de los sensores de ultrasonido se ven afectados por el campo de actuación de forma cónica, la estructura de la superficie, falsos ecos, ángulo de incidencia y reflexión, entre otros.

## 5.3. Aplicaciones

Con el fin de ratificar la importancia y potencialidad del estudio del problema de SLAM, se ha realizado cambios en el código original de mapeo, buscando que este genere un archivo externo que contenga el mapa procesado. El mapa final se encuentra compuesto de puntos extremos de líneas rectas que lo componen, que son escritos al final del proceso en un archivo ".txt". El archivo del mapa es cargado y dibujado en el ambiente AutoCad por medio de un programa desarrollado en Autolisp (lenguaje de programación que permite desarrollar programas y funciones para el manejo de entidades del tipo gráfico orientadas al uso específico de AutoCAD). La figuras Fig(5.15) y Fig(5.16) muestran ejemplos de los dibujos de los mapas en el ambiente de diseño Autocad.

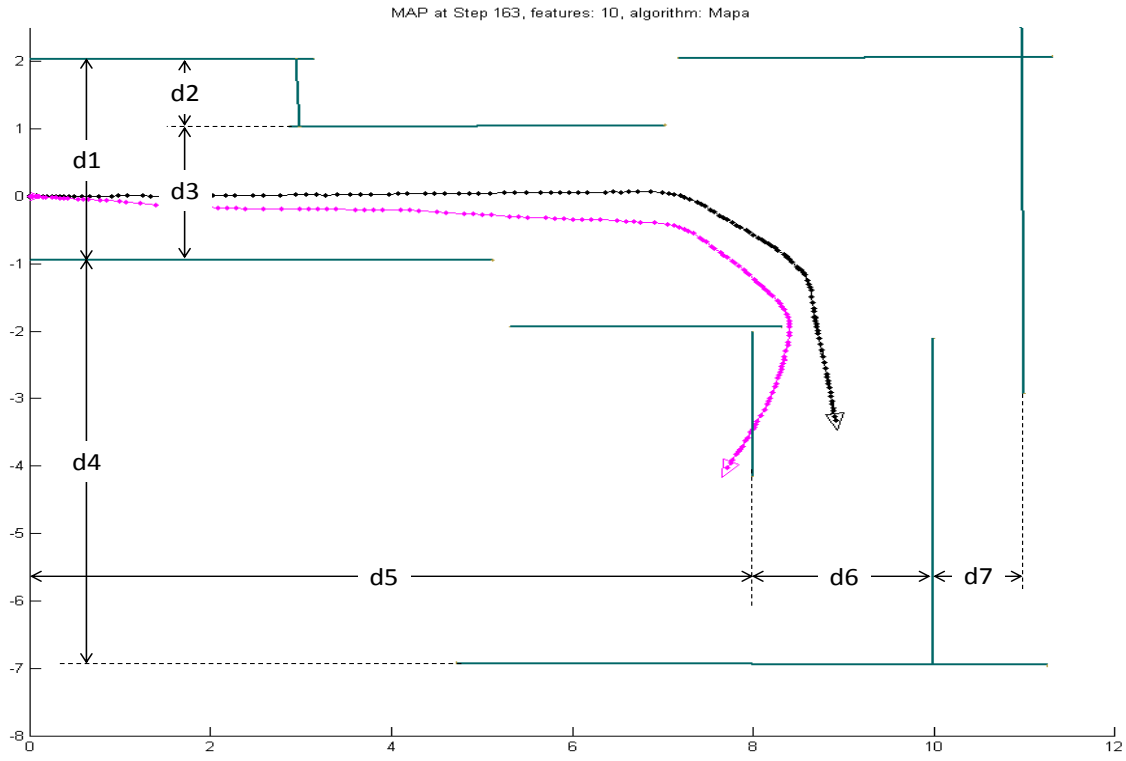


Figura 5.14: Resultado de EKF-SLAM (laser)

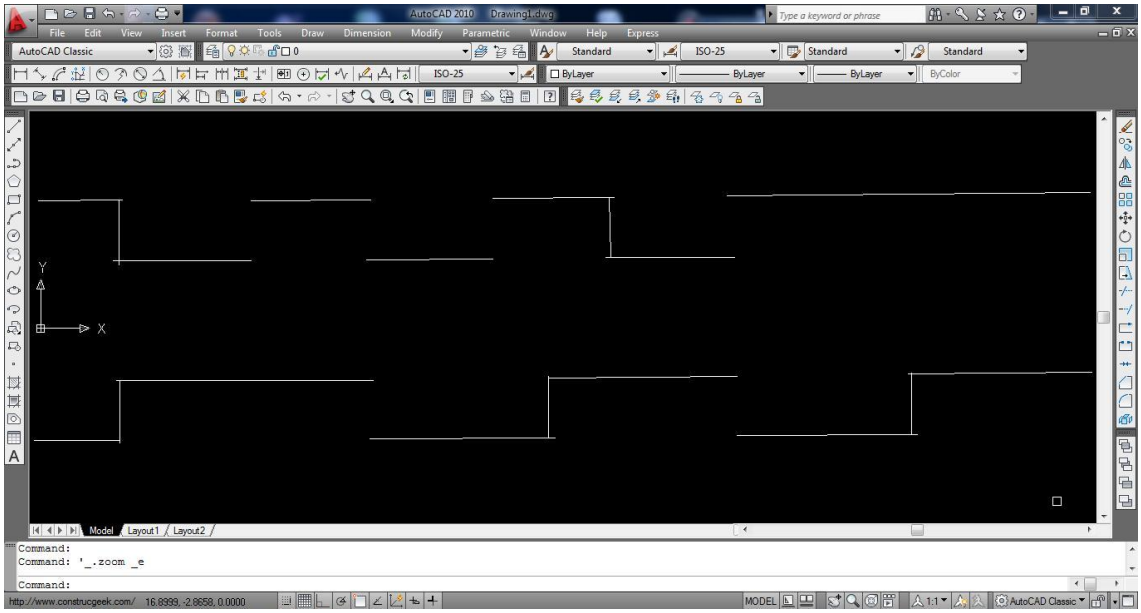


Figura 5.15: Mapa en Autocad

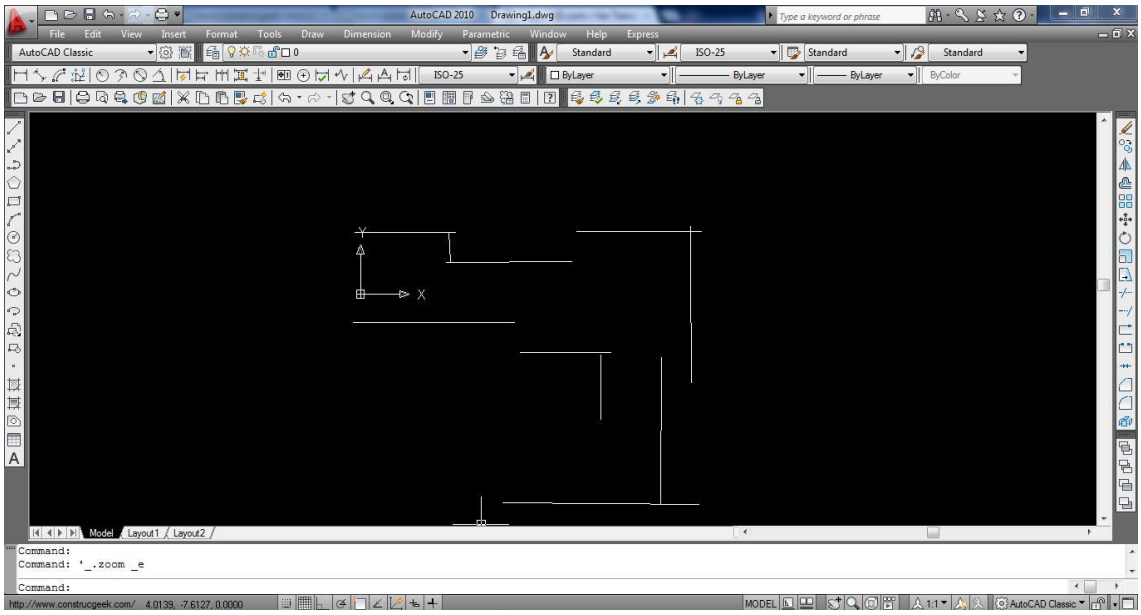


Figura 5.16: Mapa en Autocad

# Capítulo 6

# CONCLUSIONES

## 6.1. Segmentación y Asociación de datos

Dependendiendo del error del sensor y su calibración se pueden utilizar diferentes tipos de algoritmos para realizar la extracción de puntos de referencia (como split and merge, RANSAC o transformada de Hough), esto depende principalmente de la cantidad de datos espurios presentes en las lecturas, lo cual demanda mayor robustez en la segmentación. Con la implementación del algoritmo de SLAM propuesto la varianza y el error, de las variables de localización del robot y las que describen el mapa son notablemente menores en comparación a los datos basados en los sensores de odometría. La técnica de SLAM implementada basada en líneas rectas como puntos de referencia, resulta adecuada para realizar el mapeo de ambientes estructurados, tanto por la descripción del ambiente, como por la compacidad de los datos, ya que una sola característica detectada y actualizada resulta ser muy representativa (en longitud) para la construcción del mapa, haciendo que el crecimiento del tamaño del mismo sea reducido, y por lo tanto el tiempo de cómputo menor.

En la experimentación del algoritmo de SLAM implementado, la importancia de la correcta asociación de características resultaba evidente. Los errores en el robot finalmente conduciría a la ruptura del proceso de asociación, lo que lleva al fracaso del algoritmo de SLAM. Un desarrollo eficiente de SLAM requiere mantener el error de la ubicación del robot revisada mediante la observación de las características con regularidad. El tema de la función de correspondencia basada en la cercanía a menudo conducen a la adecuación de las características cercanas en el espacio, pero no topológicamente, con lo que se realiza un mapa erróneo.

## 6.2. Filtro de Kalman Extendido

La no linealidad está implícito en el sistema, esto hace que sea esencial tratar con ella. El sistema EKF controla la no linealidad a traves del uso de una aproximación de primer orden serie de Taylor de las funciones no lineales. Esta aproximación se

ha documentado ser demasiado confiada y en algunos casos conducen a estimaciones erróneas y, finalmente, a mapas ineficientes. A pesar de que el sistema de EKF-SLAM se ha llevado a cabo en nuestros experimentos de características de linea utilizando aproximaciones, fue en tiempos de alta linealidad en que los errores no conduciría a una ruptura del algoritmo. Este problema viene implícito con el sistema de EKF-SLAM, a pesar de la reducción en los errores de linealidad.

Los mapas desarrollados en este trabajo asumen el medio ambiente semi estático y por lo tanto, el algoritmo está diseñado para manejar entornos medianamente dinámicos. Sin embargo, teniendo en cuenta el trabajo reciente de la localización en entornos dinámicos, el desarrollo de un sistema de SLAM funcionar en entornos dinámicos de seguimiento de rasgos estáticos descartando los no estáticos.

## 6.3. Aplicaciones

AutoCAD es un software diseño CAD, dibujo, modelado de estructuras, dibujo arquitectónico e ingeniería en 2D y 3D más utilizado por arquitectos e ingenieros de todo el mundo. Gracias a sus avanzadas y convenientes características, en la actualidad es una pieza fundamental en cualquier estudio de diseño arquitectónico o ingeniería industrial, y es utilizado habitualmente para el desarrollo y elaboración de complejas piezas de dibujo técnico en dos dimensiones y para creación de modelos tridimensionales.
AutoLISP es una adaptación del lenguaje de programación LISP y forma parte integral del paquete AutoCAD. Autolisp es un pequeño subconjunto del CommonLISP, y por ello se ajusta muy estrechamente a la misma sintaxis y convenciones, pero consta de muchas funciones específicas de AutoCAD. Con AutoLISP, se pueden escribir programas y generar funciones de macros con un lenguaje potente y de alto nivel, apropiado para las aplicaciones de gráficos. AutoLISP es la más potente herramienta para optimizar la ejecución de AutoCAD. Le habilita para automatizar AutoCAD incluso más allá de lo que puede llevar a cabo usando macros. Con AutoLISP, se pueden escribir programas y generar funciones de macros con un lenguaje potente y de alto nivel, apropiado para las aplicaciones de gráficos.
El dibujo obtenido a través del programa desarrollado en AutoLisp brinda una aproximación a un levantamiento de mapas de interiores para determinar la configuración del ambiente y la posición sobre la superficie de elementos o instalaciones, que podría ser utilizado en las fases de los proyectos arquitectónicos y de ingeniería civil.

## 6.4. Resumen

Se llega a la finalización de esta tesis con la esperanza que sirva como una referencia informativa para los interesados en el campo de la robótica móvil. Esperamos que el código documentado y el sistema implementado servirá como buenos puntos de partida para entender los conceptos del mapeo. Si bien el presente proyecto (utilizando el Filtro de Kalman Extendido) no posee un alto grado de innovación a nivel internacional,

creemos que esta tesis ofrece una visión global del EKF-SLAM al mismo tiempo que discute sobre los detalles específicos del sistema que son necesarios para la aplicación de un algoritmo de SLAM. Terminamos esta tesis en la previsión de que pueda servir como fuente de conocimiento para la investigación, contribuir al crecimiento de la comunidad robótica y generar nuevos proyectos de investigación como el mapeo de zonas extensas.

# Apéndice A

# PUBLICACIONES

El presente trabajo dió lugar a las siguientes publicaciones y congresos nacionales e internacionales:

1. J.S. Berrío, L.M.Paz, E.F. Caicedo. Algoritmo de extracción de líneas a partir de datos obtenidos por un scaner laser 2D. IEEE LARC - LARS - CCAC IASCW 2011, Bogotá, Colombia, October 2010, CFP11LAR-CDR.

2. J.S. Berrío, L.M.Paz, E.F. Caicedo. Segmentation and parameterization of 2D lines based on Mean Shift Clustering, 2012 Workshop on Engineering Applications, Bogotá, Colombia, Mayo, 2012.''http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6220085''

3. J.S. Berrío, S.A. Orozco, E.F. Caicedo. Lines extraction in laser scans through the integration of the Hough transform and SEF, 2012 Workshop on Engineering Applications, Bogotá, Colombia, Mayo, 2012. ''http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6220086''

# Bibliografía


[1] F. Amigoni, S. Gasparini, and M. Gini. Building segment-based maps without pose information. *Proceedings of the IEEE*, 94(7):1340 –1359, july 2006. ISSN 0018-9219. doi: 10.1109/JPROC.2006.876925.

[2] K. O. Arras and R. Siegwart. Feature Extraction and Scene Interpretation for Map-Based Navigation and Map Building. In *Symposium on Intelligent Systems and Advanced Manufacturing*, 1997.

[3] Josep Aulinas. *3D Visual SLAM applied to large-scale underwater scenarios*. Universidad de Girons, España, 2006.

[4] Kristopher R. Beevers. Sampling strategies for particle filtering slam. In *IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION (ICRA 2007)*, 2007.

[5] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[6] G.A. Borges and M.-J. Aldon. A split-and-merge segmentation algorithm for line extraction in 2d range images. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, volume 1, pages 441 –444 vol.1, 2000. doi: 10.1109/ICPR.2000.905371.

[7] Geovany Araujo Borges and Marie-JosÃ© Aldon. Line extraction in 2d range images for mobile robotics. *Journal of Intelligent amp; Robotic Systems*, 40:267–297, 2004. ISSN 0921-0296. URL `http://dx.doi.org/10.1023/B:JINT.0000038945.55712.65`. 10.1023/B:JINT.0000038945.55712.65.

[8] R. A. Brooks. A robust layered control system for a mobile robot. In *International Conference on Robotics and Automation*, 1985.

[9] E. Brunskill and N. Roy. Slam using incremental probabilistic pca and dimensionality reduction. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 342 – 347, april 2005. doi: 10.1109/ROBOT.2005.1570142.

[10] Margarita Gallardo Campos. *Aplicación de técnicas de clustering para la mejora del aprendizaje*. Universidad Carlos III de Madrid, España, 2009.

[11] Carlos Fernández Caramés. *Tecnicas de navegacion de robots basados en sistemas de medicion por laser*. Universidad de Salamanca, Septiembre 2007.

[12] J. A. Castellanos and J. D. Tardós. Laser-based segmentation and localization for a mobile robot. In F. Pin, M. Jamshidi, and P. Dauchez, editors, *Robotics and Manufacturing vol. 6 - Procs. of the 6th Int. Symposium (ISRAM)*, pages 101–108. ASME Press, New York, NY, 1996.

[13] Ingemar J. Cox. A review of statistical data association techniques for motion correspondence. *International Journal of Computer Vision*, 10:53–66, 1993.

[14] Fabio Gagliardi Cozman and Eric Krotkov. Automatic mountain detection and pose estimation for teleoperation of lunar rovers. In *Proceedings of the 5th International Symposium on Experimental Robotics V*, pages 207–215, London, UK, 1998. Springer-Verlag. ISBN 3-540-76218-3. URL `http://portal.acm.org/citation.cfm?id=645625.757990`.

[15] G. Dissanayake, H. Durrant-Whyte, and T. Bailey. A computationally efficient solution to the simultaneous localisation and map building (slam) problem. In *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, volume 2, pages 1009 –1014 vol.2, 2000. doi: 10.1109/ROBOT.2000.844732.

[16] JULIA VERDEJO DOMÍNGUEZ. *Implementación del FKE para SLAM*. Escuela de ingenieros de Sevilla, España, 2008.

[17] Richard O. Duda and Peter E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM*, 15:11–15, January 1972. ISSN 0001-0782. doi: http://doi.acm.org/10.1145/361237.361242. URL `http://doi.acm.org/10.1145/361237.361242`.

[18] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localisation and mapping (slam): Part i the essential algorithms. *IEEE ROBOTICS AND AUTOMATION MAGAZINE*, 2:2006, 2006.

[19] T. Einsele. Real-time self-localization in unknown indoor environment using a panorama laser range finder. In *Intelligent Robots and Systems, 1997. IROS '97., Proceedings of the 1997 IEEE/RSJ International Conference on*, volume 2, pages 697 –702 vol.2, sep 1997. doi: 10.1109/IROS.1997.655087.

[20] C. Estrada, J. Neira, and J.D. Tardos. Hierarchical slam: Real-time accurate mapping of large environments. *Robotics, IEEE Transactions on*, 21(4):588 – 596, aug. 2005. ISSN 1552-3098. doi: 10.1109/TRO.2005.844673.

[21] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24:381–395, June 1981. ISSN 0001-0782. doi: http://doi.acm.org/10.1145/358669.358692. URL `http://doi.acm.org/10.1145/358669.358692`.

[22] A. Garulli, A. Giannitrapani, A. Rossi, and A. Vicino. Mobile robot SLAM for line-based environment representation. In *European Control Conference 2005*, Decision and Control, page 2005, 2041-2046.

[23] Gene H. Golub and Charles Van Loan. An analysis of the total least squares problem. Technical report, Cornell University, Ithaca, NY, USA, 1980.

[24] Peter J. Huber, John Wiley, Sons., and Wiley InterScience (Online Service). *Robust statistics / Peter J. Huber*. Wiley, New York :, 1981. ISBN 0471418056 0471725250. URL `http://www.loc.gov/catdir/toc/onix01/80018627.htmlhttp://ezproxy.lib.monash.edu.au/login?url=http://dx.doi.org/10.1002/0471725250`.

[25] Anil K. Jain, Robert P. W. Duin, and Jianchang Mao. Statistical pattern recognition: A review. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 22(1):4–37, 2000.

[26] Patric Jensfelt and Henrik I. Christensen. Laser based position acquisition and tracking in an indoor environment. In *in Proc. Int. Symp. Robotics and Automation*, 1998.

[27] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME Journal of Basic Engineering*, (82 (Series D)):35–45, 1960. URL `http://www.cs.unc.edu/~{}welch/kalman/media/pdf/Kalman1960.pdf`.

[28] C.T.M. Keat, C. Pradalier, and C. Laugier. Vehicle detection and car park mapping using laser scanner. In *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 2054 – 2060, aug. 2005. doi: 10.1109/IROS.2005.1545233.

[29] Sewan Kim and Younggie Kim. Robot localization using ultrasonic sensors. In *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 4, pages 3762 – 3766 vol.4, sept.-2 oct. 2004. doi: 10.1109/IROS.2004.1390000.

[30] Gururaj Kosuru. *Design and Implementation of an EKF based SLAM algorithm on a mobile robot*. PhD thesis, International Institute of Information Technology, october 2011.

[31] Benjamin Kuipers and Yung-Tai Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *JOURNAL OF ROBOTICS AND AUTONOMOUS SYSTEMS*, 8:47–63, 1991.

[32] T. S. Levitt and D. T. Lawton. Qualitative navigation for mobile robots. *Artif. Intell.*, 44:305–360, July 1990. ISSN 0004-3702. doi: 10.1016/0004-3702(90)90027-W. URL `http://portal.acm.org/citation.cfm?id=101743.101745`.

[33] R. Martinez-Cantin, J.A. Castellanos, J.D. Tardos, and J.M.M. Montiel. Adaptive scale robust segmentation for 2d laser scanner. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 796 –801, Octubre 2006.

[34] R. Martinez-Cantin, J.A. Castellanos, J.D. Tardos, and J.M.M. Montiel. Adaptive scale robust segmentation for 2d laser scanner. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 796 –801, oct. 2006. doi: 10.1109/IROS.2006.281671.

[35] F. Mastrogiovanni, A. Sgorbissa, and R. Zaccaria. On the tips of one's toes: self-localization in a dynamic environment. In *Computational Intelligence in Robotics and Automation, 2005. CIRA 2005. Proceedings. 2005 IEEE International Symposium on*, pages 341 – 346, june 2005. doi: 10.1109/CIRA.2005.1554300.

[36] Jerry M. Mendel. *Lessons in Estimation Theory for Signal Processing, Communications*. Prentice Hall, 1995.

[37] J. M. M Montiel and L. Montano. Efficient validation of matching hypotheses using mahalanobis distance. *Engineering Applications of Artificial Ingelligence*, 11 (3):439–448, 1998.

[38] Viet Nguyen, Stefan Gächter, Agostino Martinelli, Nicola Tomatis, and Roland Siegwart. A comparison of line extraction algorithms using 2d range data for indoor mobile robotics. *Auton. Robots*, 23:97–111, August 2007. ISSN 0929-5593. doi: 10.1007/s10514-007-9034-y. URL `http://portal.acm.org/citation.cfm?id=1286035.1286041`.

[39] Lina M. Paz, Juan D. Tardós, and José Neira. Divide and Conquer: EKF SLAM in $O(n)$. *Aceepted in Transactions on Robotics (in Print)*, 24(5), October 2008. URL `http://webdiis.unizar.es/~lpaz/publications_archivos/papers/dcslam.pdf`.

[40] Lina Maria Paz. *Divide and Conquer: EKF SLAM in O(n)*. PhD thesis, Universidad de Zaragoza, November 2008.

[41] Samuel T. Pfister, Stergios I. Roumeliotis, and Joel W. Burdick. Weighted line fitting algorithms for mobile robot map building and efficient data representation. In *In ICRA*, pages 14–19, 2003.

[42] Raquel R. Pinho, João Manuel, R. S. Tavares, and Miguel V. Correia. Efficient approximation of the mahalanobis distance for tracking with the kalman filter. In *in CompIMAGE - Computational Modelling of Objects Represented in Images: Fundamentals, Methods and Applications*, pages 84–92, 2006.

[43] Soren Rissgaard. *SLAM for Dummies*. MIT cognitive robotics spring 2005, Massachuset, 2005.

[44] Peter J. Rousseeuw and Mia Hubert. *Robust statistics for outlier detection*, volume 1. John Wiley and Sons, Inc., 2011. doi: 10.1002/widm.2. URL `http://dx.doi.org/10.1002/widm.2`.

[45] Daniel Sack and Wolfram Burgard. A comparison of methods for line extraction from range data. In *In Proc. of the 5th IFAC Symposium on Intelligent Autonomous Vehicles (IAV*, 2004.

[46] S. Se, D.G. Lowe, and J.J. Little. Vision-based global localization and mapping for mobile robots. *Robotics, IEEE Transactions on*, 21(3):364 – 375, june 2005. ISSN 1552-3098. doi: 10.1109/TRO.2004.839228.

[47] Ali Siadat, Axel Kaske, Siegfried KLAUSMANN, Michel DUFAUT, and René HUSSON. An optimized segmentation method for a 2d laser-scanner applied to mobile robot navigation, 1997.

[48] Charles V. Stewart. Robust parameter estimation in computer vision. *SIAM Reviews*, 41:513–537, 1999.

[49] Sebastian Thrun. Robotic mapping: A survey. In *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002.

[50] Günther Schmidt Uwe D. Hannebeck. Set theoretic localization of fast mobile robots using an angle measurement technique, 1996.

[51] Sabine Van Huffel and Joos Vandewalle. Analysis and solution of the nongeneric total least squares problem. *SIAM J. Matrix Anal. Appl.*, 9:360–372, November 1988. ISSN 0895-4798. doi: 10.1137/0609030. URL `http://dl.acm.org/citation.cfm?id=58030.58037`.

[52] Stefan B. Williams, Gamini Dissanayake, and Hugh Durrant-whyte. Field deployment of the simultaneous localisation and mapping algorithm, 2002.

[53] X. Yun, K. Latt, and J.S. Glennon. Mobile robot localization using the hough transform and neural networks. In *Intelligent Control (ISIC), 1998. Held jointly with IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA), Intelligent Systems and Semiotics (ISAS), Proceedings*, pages 393 –400, sep 1998. doi: 10.1109/ISIC.1998.713694.

[54] L. Zhang and B.K. Ghosh. Line segment based map building and localization using 2d laser rangefinder. In *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, volume 3, pages 2538 –2543 vol.3, 2000. doi: 10.1109/ROBOT.2000.846410.

[55] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *Int. J. Comput. Vision*, 13:119–152, October 1994. ISSN 0920-5691. doi: 10.1007/BF01427149. URL `http://portal.acm.org/citation.cfm?id=195967.195970`.

[56] Zhengyou Zhang, Zhengyou Zhang, Programme Robotique, and Projet Robotvis. Parameter estimation techniques: A tutorial with application to conic fitting. *Image and Vision Computing*, 15:59–76, 1997.

[57] Marco Zuliani. *RANSAC for Dummies*. University of California, Santa Barbara, Octubre 2009.